Submitted to the Annals of Applied Statistics

SUPPLEMENTARY INFORMATION: JOINT MODELING OF MULTIPLE TIME SERIES VIA THE BETA PROCESS WITH APPLICATION TO MOTION CAPTURE SEGMENTATION

By Emily B. Fox*, Michael C. Hughes†, Erik B. Sudderth†, and Michael I. Jordan‡

*University of Washington**, *Brown University*†, *and University of California, Berkeley*‡

CONTENTS

APPENDIX A: BETA PROCESS AND INDIAN BUFFET PROCESS

Here, we discuss the connections between the *beta process* (BP), the Bernoulli process (BeP), and the *Indian buffet process* (IBP). The IBP is the predictive distribution over the feature assignments generated by a BP-BeP hierarchy, marginalizing away the global "coin-flipping" probabilities ω_k used to decide which objects possess which features. We make use of the IBP in all of our MCMC inference algorithms for feature assignments. For more information, consult (Ghahramani et al., 2006) for an introduction to the IBP and (Thibaux and Jordan, 2007) for the formal relationship between the IBP and the beta process.

Formally, we employ a beta-Bernoulli process hierarchy. Given mass parameter α, concentration parameter c, and base measure B_0, we generate feature assignments as

$$\begin{aligned} B \mid B_0 &\sim \mathrm{BP}(\alpha, c, B_0) \\ X_i \mid B &\sim \mathrm{BeP}(B), \quad i = 1, \ldots, N. \end{aligned} \tag{A.1}$$

The random measure B (a beta process realization) defines a set of weights on the global collection of features (which we interpret as human exercise behaviors in our motion capture application). Each *time series* i is associated with a draw from a Bernoulli process, X_i. The Bernoulli process realization $X_i = \sum_k f_{ik}\delta_{\theta_k}$ defines the feature vector $\boldsymbol{f}_i$ for time series i, indicating its available subset of global behaviors.

The probability X_i contains feature θ_k after having observed $X_1, \ldots, X_{i-1}$ is equal to the expected mass of that atom:

$$p(f_{ik} = 1 \mid X_1, \ldots, X_{i-1}) = \mathbb{E}_{B|X_1,\ldots,X_{i-1}}[\omega_k], \tag{A.2}$$

where $\mathbb{E}_B[\cdot]$ means to take the expectation with respect to the distribution of B. Because beta process priors are conjugate to the Bernoulli process (Kim, 1999), the posterior distribution given N samples $X_i \sim \mathrm{BeP}(B)$ is a beta process with updated parameters:

$$B \mid X_1, \ldots, X_N, B_0, c \sim \mathrm{BP}\left(c + N, \frac{c}{c+N}B_0 + \sum_{k=1}^{K_+} \frac{m_k}{c+N}\delta_{\theta_k}\right). \tag{A.3}$$

Here, m_k denotes the number of time series X_i that select the k^{th} feature θ_k (i.e., $f_{ik} = 1$). For simplicity, we have reordered the feature indices to list first the K_+ features used by at least one time series.

Using the posterior distribution defined in Eq. (A.3), we consider the discrete and continuous portions of the base measure separately.

Generically, for a discrete base measure B_0 containing atoms, a sample $B \sim \mathrm{BP}(c, B_0)$ necessarily contains each original atom θ_k with associated weights

$$\omega_k \sim \mathrm{Beta}(cq_k, c(1-q_k)), \tag{A.4}$$

where $q_k \in (0,1)$ denotes the mass of the k^{th} atom in B_0. Based on the form of Eq. (A.3), the discrete component is a collection of atoms at locations $\theta_1, \ldots, \theta_{K_+}$, each with weight

$$q_k = \frac{m_k}{c+i-1}. \tag{A.5}$$

For each instantiated feature $k \in \{1, \ldots, K_+\}$, we have

$$\omega_k \sim \mathrm{Beta}((c+i-1)q_k, (c+i-1)(1-q_k)) \tag{A.6}$$

such that the expected weight is simply q_k, implying that the i^{th} time series chooses feature k with probability proportional to m_k, the number of time series already possessing k. This result exactly matches the IBP generative process: each successive "customer" possesses a pre-existing feature with probability m_k.

Again generically, given a continuous base measure B_0, explicitly representing a beta process realization $B \sim \mathrm{BP}(c, B_0)$ as an atomic measure is not straightforward due to the improper beta distribution on atom weights $\omega_k \sim \mathrm{Beta}(cb_k, c(1-b_k))$, with $b_k = B_0(\theta_k)$ tending to zero for any continuous B_0. However, realizing a Bernoulli process draw $X_i \sim \mathrm{BeP}(B)$ is straightforward via the connection to Poisson processes: draw $L \sim \mathrm{Poisson}(\alpha)$ and independently sample L atoms, with locations $\theta_\ell \sim \frac{B_0(\theta)}{\alpha}$. All L atoms are included in the Bernoulli process realization, yielding $X_i = \sum_{\ell=1}^{L} \delta_{\theta_\ell}$.

In the case of the posterior distribution defined in Eq. (A.3), the continuous portion of the posterior is $\frac{c}{c+i-1} B_0$. The Poisson process defined by this rate function generates

$$\mathrm{Poisson}\left(\frac{c}{c+i-1} B_0(\Theta)\right) = \mathrm{Poisson}\left(\frac{c}{c+i-1}\alpha\right) \tag{A.7}$$

new atoms in X_i that do not appear in $X_1, \ldots, X_{i-1}$. Following this argument, the first time series simply chooses $\mathrm{Poisson}(\alpha)$ features. If we specialize this process to $c = 1$, we arrive at the one-parameter IBP process for generating unique features. Each successive "customer" i possesses $\mathrm{Poisson}(\alpha/i)$ brand-new features.

APPENDIX B: MARGINAL PROBABILITY COMPUTATIONS

Here, we discuss two crucial probability computations for the BP-AR-HMM. Both computations marginalize out key variables to simplify calculations and enable efficient posterior mixing in the sampler.

First, we show how to compute the marginal probability of a *discrete variable* configuration: $p(\boldsymbol{F}, \boldsymbol{z}, \boldsymbol{y})$. This computation is frequently used in our sampler. It allows our efficient split-merge and data-driven birth moves to avoid proposing values for high-dimensional continuous parameters. Furthermore, this computation allows us to create trace plots of log probability over time, allowing "apples to apples" comparison of different sampler configurations even when they differ in the number of instantiated features.

Next, we show how to compute the probability of a single sequence's observed data $p(\boldsymbol{y}^{(i)}|\boldsymbol{\theta}, \boldsymbol{\eta}^{(i)}, \boldsymbol{f}_i)$, marginalizing away the discrete state sequence $\boldsymbol{z}^{(i)}$. This computation is critical for our sampler for the *shared* features of the feature matrix $\boldsymbol{F}$. Given the assigned features of the sequence i in question, we show that this computation is a straight-forward application of the sum-product (or forward-backward) dynamic programming algorithm.

B.1. Computing $p(\boldsymbol{F}, \boldsymbol{z}, \boldsymbol{y})$ marginalizing over continuous parameters. Here, we discuss computing the joint posterior probability of the collapsed configuration $\boldsymbol{F}, \boldsymbol{z}$; that is, the discrete assignments of features and state sequences for *all* sequences. This marginalizes over the two sets of *continuous* HMM parameters: transition weights η and emission parameters θ.

$$p(\boldsymbol{F}, \boldsymbol{z}, \boldsymbol{y}) = \int_{\boldsymbol{\theta}} \int_{\boldsymbol{\eta}} p(\boldsymbol{F}, \boldsymbol{z}, \boldsymbol{\theta}, \boldsymbol{\eta}, \boldsymbol{y}) \tag{B.1}$$

Note that this marginalization is only tractable so long as the prior distributions of both η, θ enjoy conjugacy.

Given the conditional independence assumptions encoded in our graphical model, this computation decomposes in straight-forward fashion:

$$p(\boldsymbol{F}, \boldsymbol{z}, \boldsymbol{y}) = p(\boldsymbol{F}|\alpha, c) p(\boldsymbol{y}|\boldsymbol{z}) \prod_{i=1}^{N} p(\boldsymbol{z}^{(i)}|\gamma, \kappa) \tag{B.2}$$

We discuss computation of each individual term below.

Feature matrix term $p(\boldsymbol{F}|\alpha, c)$. We are given the two-parameter Indian Buffet process prior on $\boldsymbol{F}$ with mass α and concentration c (often, we fix

$c = 1$ to recover the usual IBP). We assume the matrix $\boldsymbol{F}$ has N rows and K_+ columns with at least one non-zero entry.

Ghahramani et al. discuss computing this probability at length (Ghahramani et al., 2006). The first step is to obtain the *left-ordered* form $[\boldsymbol{F}]$ of the matrix $\boldsymbol{F}$. This maps every $\boldsymbol{F}$ onto a unique matrix whose non-zero entries span only the first K_+ columns. Since the columns of this matrix are exchangeable, this mapping provides an equivalence class representation.

In practice, we achieve this form by rearranging the columns so that they are sorted from largest to smallest according to the integer value each column represents when interpreted as a base-2 number. For example, if the matrix has $N = 3$ rows and its first column has binary assignments [1 0 1], we would convert this into a decimal number $h = 5$, and sort all columns $F_{\cdot k}$ by their corresponding value $h(F_{\cdot k})$. This value h is sometimes called a "history".

After rearranging the columns, we obtain a set of sufficient statistics: K_h counts the number of columns (features) with "history" h, and m_k counts the number of sequences that possess each feature k.

$$K_h = \sum_{k=1}^{K_+} \delta_{h,h([F]_{\cdot k})}, \quad m_k = \sum_{i=1}^{N} f_{ik} \tag{B.3}$$

Given these statistics, Ghahramani et al. show (their eq. 21) that

$$p([\boldsymbol{F}]|\alpha, c) = \frac{\alpha^{K_+} c^{K_+}}{\prod_{h \geq 1} K_h!} \exp\left[-\alpha \sum_{i=1}^{N} \frac{c}{c+i-1}\right] \prod_{k=1}^{K_+} B(m_k, N - m_k + c) \tag{B.4}$$

where $B(\cdot)$ is the beta function: $B(x, y) = \frac{\Gamma(x)\Gamma(y)}{\Gamma(x+y)}$ for scalar real x, y.

State sequence term $p(\boldsymbol{z}^{(i)}|\gamma, \kappa)$.. Under our proposed BP-AR-HMM model, each sequence has *independent* transition parameters, so we handled the computation for each sequence (indexed by i) separately. First we write:

$$p(\boldsymbol{z}^{(i)}|\boldsymbol{f}_i, \gamma, \kappa) = \int_{\pi^{(i)}} p(\boldsymbol{z}^{(i)}|\pi^{(i)}) p(\pi^{(i)}|\boldsymbol{f}_i, \gamma, \kappa) d\pi^{(i)} \tag{B.5}$$

Recall that $\pi^{(i)}$ given $\boldsymbol{f}_i$ has a finite Dirichlet prior with $K_i = \sum_k f_{ik}$ states, and that $p(\boldsymbol{z}^{(i)}|\pi^{(i)})$ is just a Markovian discrete distribution. Thus, via conjugacy we can attain a closed-form expression. Let $Q(\vec{w})$ define the

normalization constant of a Dirichlet distribution given a vector of K_i parameters $\vec{w}$.

$$Q([w_1, \dots w_{K_i}]) = \frac{\Gamma(\prod_{k=1}^{K_i} w_k)}{\sum_{k=1}^{K_i} \Gamma(w_k)} \tag{B.6}$$

Then we have the following closed-form expression for the total probability of discrete state sequence $BFz^{(i)}$. We use sufficient statistics $N_{kk'}^{(i)}$ to count the number of transitions from k to k' in the sequence $\boldsymbol{z}^{(i)} = [z_1^{(i)}, \dots z_{T_i}^{(i)}]$.

$$p(\boldsymbol{z}^{(i)}|\boldsymbol{f}_i, \gamma, \kappa) = \prod_{k=1}^{K_i} \frac{Q([\dots N_{k,k'}^{(i)} + \gamma + \delta_{kk'}\kappa \dots])}{Q([\dots \gamma + \delta_{kk'}\kappa \dots])} \tag{B.7}$$

Observed data term $p(\boldsymbol{y}|\boldsymbol{z})$. Given the segmentation all sequences encoded by $\boldsymbol{z}$, we can aggregate sufficient statistics for all data assigned to each distinct behavior and compute the likelihood under the prior base measure $B_0(\cdot)$ collapsing away the emission parameters θ. Let $\boldsymbol{Y}_k = \{y_t^{(i)} : z_t^{(i)} = k\}$ and $\tilde{\boldsymbol{Y}}_k = \{\tilde{y}_t^{(i)} : z_t^{(i)} = k\}$. Then for our auto-regressive model we have

$$p(\boldsymbol{y}|\boldsymbol{z}) = \prod_{k=1}^{K_+} \int_{\theta_k} p(\boldsymbol{Y}_k|\tilde{\boldsymbol{Y}}_{\boldsymbol{k}}, \theta_k) B_0(\theta_k) d\theta_k \tag{B.8}$$

Recall that each emission parameter is parameterized as $\theta_k = (\Sigma_k, \boldsymbol{A}_k)$ for our AR(1) likelihood, where both Σ_k and $\boldsymbol{A}_k$ are D-by-D matrices where D is the dimension of each observation $y_t^{(i)}$. For the base measure B_0, we have a Matrix Normal - Inverse Wishart (MNIW) prior. It takes four parameters: degrees-of-freedom n_0, scale matrix S_0 (defining covariance of Σ_k), scale matrix K (which together with S_0 defines the covariance of A_k), and finally mean matrix M.

$$\begin{aligned} \Sigma_k \mid n_0, S_0 &\sim \mathrm{IW}(n_0, S_0) \\ \boldsymbol{A}_k \mid \Sigma_k, M, K &\sim \mathcal{MN}(\boldsymbol{A}_k; M, \Sigma_k, K), \end{aligned} \tag{B.9}$$

Executing the necessary calculus and algebra, we find the computation reduces to a closed-form function of sufficient statistics:

(B.10)

$$p(\boldsymbol{Y}_k|\tilde{\boldsymbol{Y}}_k) = p(\boldsymbol{Y}_k|M, K, S_0, n_0)$$
$$= \int\int p(\boldsymbol{Y}_k|\boldsymbol{A}_k, \Sigma_k)p(\boldsymbol{A}_k|M, \Sigma_k, K)p(\Sigma_k|n_0, S_0)d\Sigma_k d\boldsymbol{A}_k$$

(B.11)
$$= \frac{1}{(2\pi)^{\frac{n_k d}{2}}} \frac{\Gamma_d(\frac{n_k+n_0}{2})}{\Gamma_d(\frac{n_0}{2})} \frac{|S_0|^{\frac{n_0}{2}}}{|S^{(k)}_{y|\tilde{y}}|^{\frac{n_k+n_0}{2}}} \frac{|K|^{\frac{1}{2}}}{|S^{(k)}_{\tilde{y}\tilde{y}}|^{\frac{1}{2}}}$$

where $\Gamma_D(\cdot)$ is the D-dimensional gamma function, $|\cdot|$ denotes the determinant, n_k counts the number of observations in set $\boldsymbol{Y}_k$, and sufficient statistics $S^{(k)}_{\cdot,\cdot}$ are defined below

(B.12)
$$S^{(k)}_{\tilde{y}\tilde{y}} = \sum_{(t,i)|z^{(i)}_t=k} \tilde{\mathbf{y}}^{(i)}_t \tilde{\mathbf{y}}^{(i)^T}_t + \boldsymbol{K} \qquad S^{(k)}_{y\tilde{y}} = \sum_{(t,i)|z^{(i)}_t=k} \mathbf{y}^{(i)}_t \tilde{\mathbf{y}}^{(i)^T}_t + \boldsymbol{MK}$$
$$S^{(k)}_{yy} = \sum_{(t,i)|z^{(i)}_t=k} \mathbf{y}^{(i)}_t \mathbf{y}^{(i)^T}_t + \boldsymbol{MKM}^T \qquad S^{(k)}_{y|\tilde{y}} = S^{(k)}_{yy} - S^{(k)}_{y\tilde{y}} S^{-(k)}_{\tilde{y}\tilde{y}} S^{(k)^T}_{\tilde{y}\tilde{y}}.$$

We emphasize that calculation of $p(\boldsymbol{y}|\boldsymbol{z})$ is straightforward even with alternative generating distributions for the observed data (e.g. Gaussian or Multinomial). So long as this distribution has the appropriate conjugate prior, this quantity can be computed in closed-form as functions of sufficient statistics.

B.2. Computing $p(\boldsymbol{y^{(i)}}|\boldsymbol{f_i}, \boldsymbol{\theta}, \boldsymbol{\eta^{(i)}})$ marginalizing state sequence $\boldsymbol{z^{(i)}}$. Here, we discuss computation of the probability of a single sequence's observed data under a particular feature assignment:

(B.13)
$$p(\boldsymbol{y}^{(i)}|\boldsymbol{f}_i, \boldsymbol{\theta}, \boldsymbol{\eta}^{(i)}) = \sum_{\boldsymbol{z}^{(i)}} p(\boldsymbol{y}^{(i)}, \boldsymbol{z}^{(i)}|\boldsymbol{f}_i, \boldsymbol{\theta}, \boldsymbol{\eta}^{(i)})$$

This is sometimes called the "marginal likelihood" of sequence i, though we prefer the more descriptive term "likelihood marginalizing over discrete state sequence."

Achieving this calculation efficiently requires a variant of the sum-product algorithm applied to the chain graph of the AR-HMM. Recall that given a fixed feature assignment $\boldsymbol{f}_i$ for each sequence in question, the BP-AR-HMM reduces to a set of finite-dimensional AR-HMMs, each of which is

described by its set of feature-constrained transition distributions $\pi^{(i)}$ (a deterministic normalization of $\eta^{(i)}$) along with the shared library of VAR parameters $\theta_k = \{\boldsymbol{A}_k, \Sigma_k\}$. The derivations below directly follow those for the standard HMM (Rabiner, 1989). For simplicity we drop the superscript i notation and use z_t, y_t instead of $z_t^{(i)}, y_t^{(i)}$ as we assume a particular sequence has be chosen.

First, we define a set of *forward messages*

$$\alpha_t(z_t) \triangleq p(\boldsymbol{y}_1, \ldots, \boldsymbol{y}_t, z_t), \tag{B.14}$$

which satisfy the recursion

$$\alpha_{t+1}(z_{t+1}) = p(\boldsymbol{y}_{t+1} \mid z_{t+1}, \tilde{\boldsymbol{y}}_{t+1}) \sum_{z_t} p(\boldsymbol{y}_1, \ldots, \boldsymbol{y}_t \mid z_t) p(z_{t+1} \mid z_t) p(z_t) \tag{B.15}$$

$$= p(\boldsymbol{y}_{t+1} \mid z_{t+1}, \tilde{\boldsymbol{y}}_{t+1}) \sum_{z_t} \alpha_t(z_t) p(z_{t+1} \mid z_t) \tag{B.16}$$

$$= \mathcal{N}(\boldsymbol{y}_{t+1}; \boldsymbol{A}_{z_{t+1}} \tilde{\boldsymbol{y}}_{t+1}, \Sigma_{z_{t+1}}) \sum_{z_t} \alpha_t(z_t) \pi_{z_t}(z_{t+1}). \tag{B.17}$$

The messages are initialized as

$$\alpha_1(z_1) = p(\boldsymbol{y}_1, \tilde{\boldsymbol{y}}_1, z_1) = \mathcal{N}(\boldsymbol{y}_1; \boldsymbol{A}_{z_1} \tilde{\boldsymbol{y}}_1, \Sigma_{z_1}) \pi_0(z_1). \tag{B.18}$$

After running the recursion from $t = 1, \ldots, T$ to obtain $\alpha_2, \ldots \alpha_T$, the desired likelihood is simply computed by summing over the components of the forward message at time T:

$$p(\boldsymbol{y}_1, \ldots, \boldsymbol{y}_T | \boldsymbol{f}_i, \theta, \pi^{(i)}) = \sum_{z_T} \alpha_T(z_T). \tag{B.19}$$

Note that for the BP-HMM, at each step the forward message for time series i is computed by summing $z_t^{(i)}$ over the finite collection of possible HMM state indices specified by that time series's feature vector $\boldsymbol{f}_i$. That is, whenever we have a $\sum_{z_t}$ in the above equations, recall that the range of the sum indicator is $z_t \in \{k : f_{i,k} = 1\}$.

Numerical stability. Frequently, for long sequences and high-dimensional observations the above algorithm can result in numerical underflow. This occurs because the joint probability $p(y_1, \ldots y_{T_i})$ can be a very small number. To make the computation numerically stable for longer sequences, we recommend implementing the revised recursion:

$$\tilde{\alpha}_t(z_{t+1}) \triangleq p(\boldsymbol{y}_{t+1}, z_{t+1}|\boldsymbol{y}_1, \ldots, \boldsymbol{y}_t) = p(\boldsymbol{y}_{t+1}|z_{t+1}) \sum_{z_t} \pi_{z_t z_{t+1}} \tilde{\alpha}_t(z_t) \tag{B.20}$$

$$\tilde{s}_{t+1} \triangleq p(\boldsymbol{y}_t|\boldsymbol{y}_1, \ldots, \boldsymbol{y}_{t-1}) = \sum_{z_{t+1}} \tilde{\alpha}_{t+1}(z_{t+1}) \tag{B.21}$$

By defining each successive quantity as a distribution only over $\boldsymbol{y}_t, z_t$ (instead of the full joint) we obtain improved numerical stability.

The dynamic programming algorithm is straightforward. First initialize:

$$\tilde{\alpha}_1(z_1) = \mathcal{N}(\boldsymbol{y}_1; \boldsymbol{A}_{z_1} \tilde{\boldsymbol{y}}_1, \Sigma_{z_1}) \pi_0(z_1) \tag{B.22}$$

$$\tilde{s}_1 = \sum_{z_1} \tilde{\alpha}_1(z_1) \tag{B.23}$$

$$\tilde{\alpha}_1(z_1) = \frac{1}{\tilde{s}_1} \tilde{\alpha}_1(z_1) \tag{B.24}$$

. Then iterate from $t = 1, \ldots T-1$ and do:

$$\tilde{\alpha}_{t+1}(z_{t+1}) = \mathcal{N}(\boldsymbol{y}_{t+1}; \boldsymbol{A}_{z_{t+1}} \tilde{\boldsymbol{y}}_{t+1}, \Sigma_{z_{t+1}}) \sum_{z_t} \alpha_t(z_t) \pi_{z_t}(z_{t+1}). \tag{B.25}$$

$$\tilde{s}_{t+1} = \sum_{z_{t+1}} \tilde{\alpha}_{t+1}(z_{t+1}) \tag{B.26}$$

$$\tilde{\alpha}_{t+1}(z_{t+1}) = \frac{1}{\tilde{s}_{t+1}} \tilde{\alpha}_{t+1}(z_{t+1}) \tag{B.27}$$

We then compute the *logarithm* of the likelihood of interest:

$$\log p(\boldsymbol{y}_1, \ldots, \boldsymbol{y}_T|\boldsymbol{f}_i, \theta, \pi^{(i)}) = \sum_{t=1}^{T} \log \tilde{s}_t \tag{B.28}$$

APPENDIX C: MCMC ALGORITHM FOR THE BP-AR-HMM

Here we give the formal details of the overall MCMC sampling algorithm for the BP-AR-HMM advocated in our main paper.

The high-level algorithm is `BP-AR-HMM-MCMC`, listed in Alg. C.1. This algorithm modifies every single member variable of the Markov state. Completing all the steps in this algorithm represent one "iteration" of our sampler. This includes our novel birth-death moves modifying discrete variables as well as our sequentially-allocated split-merge moves. Note that some moves (such as our split-merge moves) may be attempted multiple times, since the acceptance rates of these moves are typically low but the potential benefits

Algorithm C.1 BP-AR-HMM-MCMC($\boldsymbol{y}, \Psi, \alpha, c, \gamma, \kappa, \lambda$)

Input:

$\boldsymbol{y}, \tilde{\boldsymbol{y}}$: N sequences of observed data (current and lagged observations)

$\Psi^{t-1} = (\boldsymbol{F}, \boldsymbol{z}, \theta, \eta, \alpha, c, \gamma, \kappa)$: input state of Markov chain (iter. $t-1$)

$\boldsymbol{F}$: N-by-K_+ binary feature matrix

$\boldsymbol{z}$: N discrete state sequences. Each $\boldsymbol{z}_t^{(i)} = k \in \{1, 2, \dots K_+\}$ s.t. $f_{ik} = 1$

$\theta = \theta_1, \dots \theta_k$: Emission parameters (one set per feature k)

$\eta = \eta^{(1)} \dots \eta^{(N)}$: Transition weights (one set per sequence i)

α, c : BP hyperparameters

γ, κ : HMM transition hyperparameters

$\lambda = (n_0, S_0, M, K)$: MNIW prior hyperparameters

Output:

$\Psi^t = (\boldsymbol{F}, \boldsymbol{z}, \theta, \eta, \alpha, c, \gamma, \kappa)$: next state of Markov chain (iter. t)

Procedure:

for sequence $i \in \{1, 2, \dots N\}$:
  for feature $k \in \{1, 2, \dots K_+\}$ owned by ≥ 1 other seq. ($\exists j \neq i$ s.t. $f_{jk} = 1$):
    $m_k^{(-i)} \leftarrow \sum_{i' \neq i} f_{i'k}$
    $f_{ik} \sim$ `SampleSharedFeature`($\boldsymbol{y}^{(i)}$, $\boldsymbol{f}_i$, $m_k^{(-i)}$, θ, $\eta^{(i)}$, (α, c)) (Alg. C.2)
  $\boldsymbol{z}^{(i)} \sim$ `SampleStateSequence`($\boldsymbol{y}^{(i)}$, $\boldsymbol{f}_i, \theta, \eta$) (Alg. C.3)
  $\boldsymbol{f}_i, \boldsymbol{z}^{(i)} \sim$ `SampleBirthDeath`($\boldsymbol{y}^{(i)}$, $\boldsymbol{f}_i$, $\boldsymbol{z}$, (α, c), (γ, κ), λ (Alg. D.1)
for trial $r = 1,2, \dots R$:
  $\boldsymbol{F}, \boldsymbol{z} \sim$ `SampleSplitMerge`($\boldsymbol{y}$, $\boldsymbol{F}, \boldsymbol{z}$, (α, c), (γ, κ), λ)
for feature $k \in \{1, 2, \dots K_+\}$:
  $\theta_k \sim$ `SampleEmissionParams`($\{\boldsymbol{y}_t^{(i)} : z_t^{(i)} = k\}$, λ) (Alg. C.4)
for sequence $i \in \{1, 2, \dots N\}$:
  $\eta^{(i)} \sim$ `SampleTransitionWeights`($\boldsymbol{z}^{(i)}$, (γ, κ)) (Alg. C.5)
$\alpha, c \sim$ `SampleBPHypers`($\boldsymbol{F}$, (α, c)) (Alg. F.1)
$\gamma, \kappa \sim$ `SampleHMMTransitionHypers`($\boldsymbol{z}$, (γ, κ)) (Alg. F.2)

of an accepted move are very high due to the possible large changes in the global discrete assignments.

We also specify the detailed sampling algorithms for some specific variable updates in 5 here in this section. Note that the algorithms for birth-death moves are in Supplement D and the split-merge algorithms are in Supplement E. Furthermore, the hyperparameter sampling algorithms are in Supplement F.

This leaves the following basic moves: sampling a single entry of the binary feature matrix f_{ik} (Alg. C.2), block sampling the state sequence for one time series $\boldsymbol{z}^{(i)}$ (Alg. C.3), sampling emission parameters θ (Alg. C.4), and sampling transition parameters η (Alg. C.5).

Sampling Shared Features. We sample each entry in $\boldsymbol{F}$ one-at-a-time using Algorithm C.2, deciding whether each binary assignment is "on" or "off." Note that this algorithm is only valid for features that have at least one other time series (excluding the current sequence) that possess it. That is, we can only use this algorithm to modify entry f_{ik} if there exists a sequence $j \neq i$ such that $f_{jk} = 1$. This requirement comes because the prior on the binary assignment of a particular entry of $\boldsymbol{F}$ we use employs the IBP predictive distribution for features that already exist. Entries that do not fit this requirement are termed *unique* features and are sampled according to the birth-death reversible jump procedures in D.

This algorithm proposes a new value $f_{ik}^* = \neg f_{ik}$, that is, the complement of the existing binary assignment of feature k in $\boldsymbol{f}_i$. This proposal is either accepted or rejected according to a Metropolis-Hastings procedure. To efficiently determine the likelihood that feature k should be present or absent in a given sequence (denoted by L in the algorithm), we marginalize over all possible state sequences $\boldsymbol{z}$, using the efficient dynamic programming procedure in B.2. This computation uses all features available to $\boldsymbol{f}_i$, including or excluding k depending on the proposal. Thus, this computation requires fixing the entire i-th row of $\boldsymbol{F}$. The prior probability of possessing feature k (denoted by P in the algorithm) is easily computed via the predictive distribution defined in the IBP process, which requires on the number of other sequences $m_k^{(-i)}$ that possess feature k. Thus, this computation requires fixing the entire k-th column of the matrix $\boldsymbol{F}$. Because the proposal distribution is deterministic, the acceptance ratio just involves the product of P and L terms for both current and candidate assignments.

Sampling state sequence. For a particular time series, we sample the entire hidden discrete state sequence $\boldsymbol{z}^{(i)}$ all at once given sequence-specific feature assignments $\boldsymbol{f}_i$, sequence-specific transition parameters $\eta^{(i)}$,

Algorithm C.2 SampleSharedFeature($\boldsymbol{y}^{(i)}, \tilde{\boldsymbol{y}}^{(i)}, \boldsymbol{f}_i, m_k^{(-i)}, \theta, \eta^{(i)}, \alpha, c$)

MH proposal that attempts to flip single entry in binary matrix to its complement

Input:

$\boldsymbol{y}^{(i)}, \tilde{\boldsymbol{y}}^{(i)}$: observations and lagged observations for seq. i

$\boldsymbol{f}_i$: current binary feature vector for seq. i

$m_k^{(-i)}$: usage count of feature k in $\boldsymbol{F}$, ignoring sequence i

Output:

$f_{ik}^{(new)}$: new binary assignment for feature k in seq. i

Procedure:

1: Obtain proposed binary assignment

$$\boldsymbol{f}_i^* \leftarrow [f_{i1} \;\; f_{i2} \;\; \dots \underbrace{\neg f_{ik}}_{\text{flip feat.}k} \dots \;\; f_{iK_+}]$$

2: Compute probability of each feature configuration under IBP prior:

$$P^* \leftarrow f_{ik}^* p_{on} + (1 - f_{ik}^*)(1 - p_{on})$$

$$P \leftarrow f_{ik} p_{on} + (1 - f_{ik})(1 - p_{on}), \qquad p_{on} \leftarrow \frac{m_k^{(-i)}}{N - m_k^{(-i)} + c}$$

3: Construct feature-constrained transition distributions:

$$\pi_k^* \propto [\eta_{k1}^{(i)} \dots \eta_{kK_+}^{(i)}] \otimes \boldsymbol{f}_i^*, \qquad k \in \{k : f_{ik}^* = 1\}$$

$$\pi_k \propto [\eta_{k1}^{(i)} \dots \eta_{kK_+}^{(i)}] \otimes \boldsymbol{f}_i, \qquad k \in \{k : f_{ik} = 1\}$$

4: Compute likelihoods of observed data under each assignment [Alg. B.27]

$$L^* \leftarrow p(\boldsymbol{y}^{(i)} | \tilde{\boldsymbol{y}}^{(i)}, \boldsymbol{f}_i^*, \theta, \pi^*)$$

$$L \leftarrow p(\boldsymbol{y}^{(i)} | \tilde{\boldsymbol{y}}^{(i)}, \boldsymbol{f}_i, \theta, \pi)$$

5: Compute acceptance ratio $\rho = \frac{P^* L^*}{P L}$

6: Sample new assignment: rand $\sim$ Unif$(0, 1)$

$$f_{ik}^{(new)} \leftarrow \begin{cases} f_{ik}^* & \text{if rand } < \min(\rho, 1) \\ f_{ik} & \text{otherwise} \end{cases}$$

and global emission parameters θ. This block sampling procedure is sometimes termed "backward-filtering, forward sampling" due to the dynamic programming algorithm employed (see Scott (2002) for details). Sampling the sequence of T_i discrete labels all at once (as a block) yields much more efficient mixing than iteratively sampling each entry individually.

The initial state distribution $\pi_0^{(i)}$ for time series i must be treated specially. We choose to simply enforce a uniform distribution over all of the K_i possible features available in $\boldsymbol{f}_i$,

$$\pi_{0k}^{(i)} = \begin{cases} \frac{1}{K_i} & \text{if } f_{ik} = 1 \\ 0 & \text{o.w.} \end{cases} \tag{C.1}$$

Alternatives are certainly possible. We choose this uniform distribution because there is little we can learn about the initial state distribution from data. Because we use a sequence-specific transition model, for each sequence there is only one observed data point, $z_1^{(i)}$, that influences this distribution's

posterior. By enforcing the uniform distribution we simplify inference (since we need not sample this value) and lose little in model expressiveness.

Algorithm C.3 SampleStateSequence($\boldsymbol{y}^{(i)}$, $\tilde{\boldsymbol{y}}^{(i)}$, $\boldsymbol{f}_i, \theta, \eta^{(i)}$)

Block sample discrete state assignments to each timestep in seq. i

Input:

$\boldsymbol{y}^{(i)}, \tilde{\boldsymbol{y}}^{(i)}$: observations and lagged observations for seq. i

$\boldsymbol{f}_i$: current binary feature vector for seq. i

$\theta = (\theta_1, \dots \theta_{K_+})$: emission parameters (one set per feature)

$\theta_k = (A_k, \Sigma_k)$: VAR likelihood parameterized by coefs A_k and covariance Σ_k

Output:

$\boldsymbol{z}^{(i)}$: sample of i's discrete state sequence drawn from conditional posterior

Procedure:

1: Precompute soft evidence

for $t \in \{1, 2, \dots T_i\}$:

$\ell_{tk} \triangleq p(\boldsymbol{y}_t^{(i)} | z_t^{(i)} = k) = \mathcal{N}(\boldsymbol{y}_t^{(i)} \mid \boldsymbol{A}_k \tilde{\boldsymbol{y}}_t^{(i)},\ \Sigma_k) \quad k \in \{k : f_{ik} = 1\}$

2: Create feature constrained transition weights

$\pi_{k\cdot} \propto [\ \eta_{k1}^{(i)} \dots \eta_{kK_+}^{(i)}\ ] \odot \boldsymbol{f}_i, \quad k \in \{k : f_{ik} = 1\}$

3: Initialize M: T_i-by-K_i table for dynamic programming

$M_{T_i,k} \leftarrow 1, \quad M_{t,k} \triangleq p(\boldsymbol{y}_{t+1:T_i}^{(i)} | z_t^{(i)} = k, \tilde{\boldsymbol{y}}_t^{(i)}, \pi, \theta)$

4: Backward pass of dynamic programming.

for $t \in \{T_i - 1, \dots 1\}$:

for $k \in \{k : f_{ik} = 1\}$:

$M_{t,k} \leftarrow \sum_{k' : f_{ik'} = 1} [\pi_k(k') \cdot \ell_{t+1,k'} \cdot M_{t+1,k'}]$

5: Forward sample discrete state sequence (initialize in special start state $z_0^{(i)} = 0$):

for $t \in \{1, 2, \dots T_i\}$:

$\vec{p}_k \leftarrow \pi_{z_{t-1}^{(i)},k} \cdot \ell_{t,k} \cdot M_{t,k} \qquad$ for $k \in \{k : f_{ik} = 1\}$

$\vec{p} \leftarrow \vec{p} / \sum_k \vec{p}_k \qquad$ *normalize*

$z_t^{(i)} \sim \mathrm{Cat}(\vec{p})$

Sampling emission parameters $\boldsymbol{\theta_k}$. Given the feature assignments $\boldsymbol{F}$, the observed data $\boldsymbol{y}$, and the feature assignments for each timestep of data encoded in $\boldsymbol{z}$, each individual feature k has a conditionally independent posterior over its emission parameters $\theta_k = (A_k, \Sigma_k)$.

Because our emission distribution (auto-regressive Gaussian) lies in the exponential family and has a conjugate posterior given our MNIW prior, this posterior is found simply by aggregating data from any timestep in any sequence assigned to feature k. We denote this collection of data assigned to k as $\boldsymbol{Y}_k = \{y_t^{(i)} : z_k^{(i)}\}$. We then compute sufficient statistics on the set $\boldsymbol{Y}_k$, and using these sufficient statistics to build the parameters of the MNIW posterior distribution over A_k, Σ_k. See Alg. C.4 for details.

Algorithm C.4 SampleEmissionParams($\boldsymbol{Y}_k, \tilde{\boldsymbol{Y}}_k, \lambda$)

Gibbs sampling algorithm for HMM emission parameter set θ_k for specific feature k

Input:

$\boldsymbol{Y}_k = \{y_t^{(i)} : z_t^{(i)} = k\}$: set of all observed data from any seq. assigned to feature k

$\tilde{\boldsymbol{Y}}_k = \{\tilde{y}_t^{(i)} : z_t^{(i)} = k\}$: set of all lagged observations from any seq. assigned to k

$\lambda = (n_0, S_0, M, K)$: MNIW prior hyperparameters

Output:

$\theta_k = (A_k, \Sigma_k)$: VAR likelihood parameters drawn from conditional posterior

Procedure:

1: Compute sufficient statistics $S_{\cdot}^{(k)}$ according to Eq. B.12.

2: Compute parameters of MNIW posterior distribution

degrees of freedom $n_k \leftarrow n_0 + |\boldsymbol{Y}_k|$

scale matrix $S_k \leftarrow S_{y|\tilde{y}}^{(k)}$

mean matrix $M_k \leftarrow S_{y\tilde{y}}^{(k)} [S_{\tilde{y}\tilde{y}}^{(k)}]^{-1}$

coef. scale matrix $K_k \leftarrow S_{\tilde{y}\tilde{y}}^{(k)}$

3: Sample new emission parameters $\theta_k = (A_k, \Sigma_k)$

$$\Sigma_k \sim \mathcal{IW}(n_k, S_k)$$
$$A_k \mid \Sigma_k \sim \mathcal{MN}(M_k, \Sigma_k, K_k)$$

Sampling transition parameters $\boldsymbol{\eta^{(i)}}$. Given the feature matrix $\boldsymbol{F}$ and state sequences $\boldsymbol{z}$, our BP-HMM model yields posterior distributions for the transition parameters η such that each sequence's specific parameters $\eta^{(i)}$ are conditionally independent. We give the algorithm $\eta^{(i)}$ in Alg. C.5.

To understand this algorithm, it helps to review the generative process for the state sequence $\boldsymbol{z}^{(i)}$. Note that we assume a uniform initial state distribution, but this detail is not crucial to this algorithm.

$$\begin{aligned}
\eta_{jk}^{(i)} &\sim \text{Gamma}(\gamma + \delta_{jk}\kappa, 1) \\
\pi_{jk}^{(i)} &\leftarrow \begin{cases} \eta_{jk}^{(i)}/C_j & \text{if } f_{ij} = 1 \text{ and } f_{ik} = 1 \\ 0 & \text{o.w.} \end{cases}, \quad C_j \leftarrow \sum_{k':f_{ik'}=1} \eta_{jk'}^{(i)} \\
z_1^{(i)} &\sim \text{Unif}(\{k : f_{ik} = 1\}) \\
z_t^{(i)} &\sim \text{Cat}(\pi_{z_{t-1}^{(i)}}^{(i)}), \quad \text{for } t = 2, 3, \ldots T_i
\end{aligned} \tag{C.2}$$

Equivalently, we can define explicitly the generative probability for the transition distribution $\pi_j^{(i)}$ out of state j as

$$\pi_j^{(i)} \sim \text{Dir}([\gamma, \gamma, \ldots \underset{\delta_{jk}}{\gamma + \kappa}, \ldots \gamma] \odot \boldsymbol{f}_i) \tag{C.3}$$

Because features not present in $\boldsymbol{f}_i$ have zero probability of occurring in $\boldsymbol{z}^{(i)}$, we only need to sample the entries of $\eta^{(i)}$ that corresponding to features k satisfying $f_{ik} = 1$. Given the set of "on" features for sequence i, the posterior of interest factors nicely so that each set of weights for transitioning "out-of" a particular feature k are conditionally independent. That is, we may simply iterate over the set of features $\{k : f_{ik} = 1\}$, and sample each set $\eta^{(i)}_{k\cdot}$ indepedently.

For features k' not possessed in $\boldsymbol{f}_i$, we need not maintain the associated entries of $\eta^{(i)}$ at all. This includes either the whole set of out-going weights-from k': $\eta^{(i)}_{k'\cdot}$, or the weights for incoming transitions to k' from some other feature j: $\eta^{(i)}_{jk'}$. Avoiding maintaining these entries can save memory in implementation. When adding some feature k' to a sequence (e.g. when proposing to flip entry $f_{ik'}$ from "off" to "on" via Alg. C.2), we can simply draw the necessary entries of $\eta^{(i)}$ from the prior. For each feature $j \in \{j : f_{ij} = 1\}$, we have

$$\eta^{(i)}_{jk'} \sim \text{Gamma}(\gamma + \delta_{jk'}\kappa, 1), \quad \eta^{(i)}_{k'j} \sim \text{Gamma}(\gamma + \delta_{k'j}\kappa, 1) \tag{C.4}$$

As described in the main text, the sampling procedure for the set of weights for transitioning "out-of" some state j is accomplished via an auxiliary variable method. Usually, sampling a set of Dirichlet random variables is done by drawing a set of independent Gamma random variables and normalizing them so the resulting set of numbers sums to one. Instead, here we need to sample from the posterior of random variables $\eta^{(i)}_j$with a Gamma prior and a normalized-Gamma likelihood. We can accomplish this by drawing the normalized transition distribution π_j for the sequence, and then unnormalizing via multiplication by a scale factor C_j. This scale factor represents the total sum of the unconstrained weights in question.

APPENDIX D: MCMC BIRTH-DEATH PROPOSALS

Here, we provide details about our birth-death MCMC proposal that adds or deletes a single feature to one particular sequence. We first discuss our novel approach that directly proposes changes to discrete assignment variables $\boldsymbol{f}_i, \boldsymbol{z}^{(i)}$. Later, we consider older methods that instead propose continuous parameters $\boldsymbol{f}_i, \theta, \eta^{(i)}$.

D.1. BP-HMM birth-death with discrete parameters. Our novel birth-death sampler directly proposes modifications to all the latent discrete assignments associated with the current sequence. This proposal move modifies two quantities: (1) add or delete a single entry in binary feature assignment vector $\boldsymbol{f}_i$, and (2) sample fresh values of the state sequence $\boldsymbol{z}^{(i)}$ using

Algorithm C.5 SampleTransitionWeights($\boldsymbol{f}_i, \boldsymbol{z}^{(i)}, \gamma, \kappa$)

Gibbs sampler for HMM transition weight set $\eta^{(i)}$ for available features of seq. i

Input:

$\boldsymbol{z}^{(i)} = [\, z_1^{(i)}, z_2^{(i)}, \dots z_{T_i}^{(i)} \,]$: state trajectory for seq. i, $\quad z_t^{(i)} \in \{k : f_{ik} = 1\}$

γ : transition hyperparameter (pseudocounts for each possible transition)

κ : self-transition bias hyperparameter

Output:

$\eta^{(i)}$: set of transition weights drawn from conditional posterior

Procedure:

1: Compute sufficient statistics above Markovian transitions in $\boldsymbol{z}^{(i)}$

$$N_{kk'} = \sum_{t=2}^{T_i} \delta_{z_{t-1}^{(i)},k} \delta_{z_t^{(i)},k'} \quad \text{for } k, k' \in \{k : f_{ik} = 1\}$$

2: **for** each feature $k \in \{k : f_{ik} = 1\}$:

Sample constrained transition distribution for each feature:

$$\pi_{k,\cdot} \sim \text{Dir}(\, [\, \dots N_{kk'} + \gamma + \delta_{k,k'}\kappa \dots \,] \odot \boldsymbol{f}_i \,)$$

Sample scale factor

$$C_k \sim \text{Gamma}(K_i\gamma + \kappa)$$

Apply scale factor to distribution to create unconstrained transition weight $\eta_{k,k'}^{(i)} \leftarrow C_k \cdot \pi_{kk'}$

only the features available in the newly modified $\boldsymbol{f}_i$. Each proposed configuration $\boldsymbol{f}_i^*, \boldsymbol{z}^{*(i)}$ is jointly accepted or rejected in a single Metropolis-Hastings step.

The overall procedure is formalized in Alg. D.1. Given the i-th sequence to modify and the current sampler configuration, we first identify the set of unique features $\mathcal{U}_i$ possessed in $\boldsymbol{f}_i$. If the size of this set $|\mathcal{U}_i|$ is zero, we must always choose a birth move. However, if some existing features are unique, we either choose a birth with probability $1/2$, or otherwise choose to delete a member of $\mathcal{U}_i$ uniformly at random.

Regardless of what move we select, we must choose randomly a *window* $\mathcal{W}$ of the current time series in advance. This window enables our data-driven proposals. We constrain $\mathcal{W}$ to have some predefined minimum and maximum length (in applications these can be chosen to suggest appropriate time scales for the duration of a single behavior). We choose the length for the current window uniformly within the given bounds, and given the length L choose the start positionof the window uniformly within all possible positions in the time series $T_0 \in \{1, 2, \dots T_i - L + 1\}$. This selection process is done without any knowledge of the sampler configuration to maintain a valid transition

kernel.

Given this window $\mathcal{W}$ and the chosen move type, we proceed to construct the proposed new configuration $\boldsymbol{f}_i^*, \boldsymbol{z}^{*(i)}$. This involves the proposal algorithms outlined in Alg. D.2 (for birth moves) and in Alg. D.3 (for death moves). Both moves rely on deterministic HMM parameters $\hat{\theta}, \hat{\eta}$ to perform block sampling of the candidate state sequence. These are obtained deterministically given the data and the input configuration $\boldsymbol{z}$, as formalized in Alg. D.4. In addition to sampling the new candidate configuration, both these proposal algorithms *also* compute the necessary forward and reverse probabilities required in the acceptance ratio.

Consider the example of a birth move. To maintain reversibility, we need to consider both the forward move (birth proposal of the candidate configuration with one extra feature), and the reverse move (death proposal of the newborn feature that returns to the exact old configuration). The forward probability consists of (1) the birth move to create $\boldsymbol{f}_i^*$, denoted $q_{f-\text{fwd}}()$, and (2) sampling the particular chosen $\boldsymbol{z}^{*(i)}$ given the previous assignments $\boldsymbol{z}^{(i)}$, denoted $q_{z-\text{fwd}}()$. The reverse probability similarly has two terms, one for deciding to delete the feature in question $q_{f-\text{rev}}()$, and the other for landing back at the exact original state sequence $q_{z-\text{rev}}()$.

The details of computing this reverse probability are a bit tricky. Essentially, instead of actually *sampling* a state sequence from the reverse move proposal algorithm (in our running example, a death move), we wish to just compute the probability of arriving back at some target state sequence $\boldsymbol{z}^{(i)}$ known in advance. To do so, we compute the necessary HMM parameters and walk through that sampling process of our block sampler (Alg. C.3) step-by-step. At each point where the algorithm requires *sampling*, we instead force the algorithm to choose the target value and record the probability of doing so via the current conditional posterior. This of course conditions on the past decisions we've made to force the sampler to propose the target value.

Given these forward and reverse move probabilities as well as the joint probability of the new and current configurations, we accept a birth move with probability $\min(1, \rho)$, where

$$\rho_{\text{birth in seq. } i} = \frac{p(\boldsymbol{y}, \boldsymbol{z}^*, \boldsymbol{f}_i^*)}{p(\boldsymbol{y}, \boldsymbol{z}, \boldsymbol{f}_i)} \frac{q_{z-\text{rev}}(\boldsymbol{z}^{(i)} | \boldsymbol{f}_i, \boldsymbol{z}^*, \boldsymbol{y})}{q_{z-\text{fwd}}(\boldsymbol{z}^{*(i)} | \boldsymbol{f}_i^*, \boldsymbol{z}, \boldsymbol{y})} \frac{q_{\text{f-rev}}(\boldsymbol{f}_i | \boldsymbol{f}_i^*)}{q_{\text{f-fwd}}(\boldsymbol{f}_i^* | \boldsymbol{f}_i)} \tag{D.1}$$

Note that for simplicity we ignore the dependence on the lagged observations $\tilde{\boldsymbol{y}}$ here.

Algorithm D.1 SampleBirthDeath$(i, \boldsymbol{y}, \tilde{\boldsymbol{y}}, \boldsymbol{F}, \boldsymbol{z}\ )$

Data-driven MH proposal that attempts to add/remove feature from seq. i

Input:

i : integer id of the sequence whose assignments we'll modify

$\boldsymbol{y}, \tilde{\boldsymbol{y}}$: observations and lagged observations for all sequences

$\boldsymbol{F} = \{\boldsymbol{f}_i\}_{i=1}^N$: binary feature assignments for all items

$\boldsymbol{z}$: state sequences for all items

Output:

$\boldsymbol{f}_i^{new}$, $\boldsymbol{z}^{(i)new}$: new assignments for item (sequence) i

Subprocedures:

$p_{\text{birth}}(\mathcal{U}_i)$: function that determines the probability of performing a birth move given the current configuration's set of unique feature ids $\mathcal{U}_i$

$$\text{we set } p_{\text{birth}}(\mathcal{U}_i) = \begin{cases} \frac{1}{2} & \text{if } |\mathcal{U}_i| > 0 \\ 1 & \text{otherwise} \end{cases} \text{, though other possibilities exist}$$

Procedure:

1: Identify set $\mathcal{U}_i$ of features unique to sequence i in $\boldsymbol{F}$

2: Select random contiguous window $\mathcal{W} = \{T_0, T_0+1, \ldots T_0+L-1\}$ of length L

$$L \sim \text{Unif}(L_{\text{min}}, L\text{max}), \quad T_0 | L \sim \text{Unif}(\{1, 2, \ldots T_i - L + 1\})$$

3: Choose either *birth* or *death* move

`movetype` $\sim$ Bern($p_{\text{birth}}(\mathcal{U}_i)$)

4: Propose new assignments using chosen move

keep track of forward and reverse transition probs Q

if `movetype` $= 1$: *birth move*

$\boldsymbol{f}_i^*, \boldsymbol{z}^{*(i)}, Q_{f-\text{fwd}}, Q_{z-\text{fwd}}, Q_{f-\text{rev}}, Q_{f-\text{rev}} \leftarrow$ `BirthProposal`($i, \boldsymbol{f}_i, \mathcal{U}_i, \boldsymbol{y}, \tilde{\boldsymbol{y}}, \boldsymbol{z}, \mathcal{W}$)

else: death move

$\boldsymbol{f}_i^*, \boldsymbol{z}^{*(i)}, Q_{f-\text{fwd}}, Q_{z-\text{fwd}}, Q_{f-\text{rev}}, Q_{f-\text{rev}} \leftarrow$ `DeathProposal`($i, \boldsymbol{f}_i, \mathcal{U}_i, \boldsymbol{y}, \tilde{\boldsymbol{y}}, \boldsymbol{z}, \mathcal{W}$)

5: Compute joint probability of current and proposed configurations (Eq. B.2)

$$L^* \leftarrow p(\boldsymbol{y}, \tilde{\boldsymbol{y}}, \boldsymbol{F}^*, \boldsymbol{z}^*)$$
$$L \leftarrow p(\boldsymbol{y}, \tilde{\boldsymbol{y}}, \boldsymbol{F}, \boldsymbol{z})$$

6: Decide whether to accept or reject candidate state

$$\rho = \frac{L^*}{L} \frac{Q_{z-\text{rev}} Q_{f-\text{rev}}}{Q_{z-\text{fwd}} Q_{f-\text{fwd}}}, \quad D \sim \text{Bern}\left(\min(\rho, 1)\right), \quad \boldsymbol{f}_i^{new}, \boldsymbol{z}^{(i)new} = \begin{cases} \boldsymbol{f}_i^*, \boldsymbol{z}^{*(i)} & \text{if } D = 1 \\ \boldsymbol{f}_i, \boldsymbol{z} & \text{o.w.} \end{cases}$$

Algorithm D.2 BirthProposal(i, $\boldsymbol{f}_i, \mathcal{U}_i, \boldsymbol{y}, \tilde{\boldsymbol{y}}, \boldsymbol{z}, \mathcal{W}$)

Add new feature at $k^* = K_+ + 1$
$\boldsymbol{f}_i^* \leftarrow [f_{i1}\ f_{i2}\ \dots f_{iK_+}\ 1]$
$\mathcal{U}_i^* \leftarrow \mathcal{U}_i + \{k^*\}$
$Q_{f-\text{fwd}} \leftarrow p_{\text{birth}}(\mathcal{U}_i)$
Obtain HMM parameters, using modified state sequence $\bar{\boldsymbol{z}}$
$\bar{z}_t^{(i)} \leftarrow \begin{cases} k^* & \text{if } t \in \mathcal{W} \\ z_t^{(i)} & \text{otherwise} \end{cases}$
$\hat{\theta}^*, \hat{\eta}^{*(i)} \leftarrow$ `DeterministicHMMParams`($i, \boldsymbol{y}, \tilde{\boldsymbol{y}}, \boldsymbol{f}_i^*, \bar{\boldsymbol{z}}$)
Sample proposed new state sequence
$z^{*(i)}, Q_{z-\text{fwd}} \sim$ `SampleStateSequence`$(\boldsymbol{y}^{(i)}, \tilde{\boldsymbol{y}}^{(i)}, \boldsymbol{f}_i^*, \hat{\theta}^*, \hat{\eta}^{*(i)}$)
Consider reverse move (death at feature $k^* = K_+ + 1$)
$\bar{\boldsymbol{f}}_i \leftarrow [f_{i1} \dots f_{iK_+} 0]$
$Q_{f-\text{rev}} \leftarrow \frac{1-p_{\text{birth}}(\mathcal{U}_i^*)}{|\mathcal{U}_i^*|}$
$\hat{\theta}, \hat{\eta}^{(i)} \leftarrow$ `DeterministicHMMParams`($i, \boldsymbol{y}, \tilde{\boldsymbol{y}}, \bar{\boldsymbol{f}}_i, \boldsymbol{z}^*$)
$Q_{z-\text{rev}} \leftarrow$ `SampleStateSequence`$(\boldsymbol{y}^{(i)}, \tilde{\boldsymbol{y}}^{(i)}, \bar{\boldsymbol{f}}_i, \hat{\theta}, \hat{\eta}^{(i)}$, target=$\boldsymbol{z}$)

Algorithm D.3 DeathProposal(i, $\boldsymbol{f}_i, \mathcal{U}_i, \boldsymbol{y}, \tilde{\boldsymbol{y}}, \boldsymbol{z}, \mathcal{W}$)

Choose specific unique feature to delete
$k^* \sim \text{Unif}(\ \mathcal{U}_i\ )$
$\boldsymbol{f}_i^* \leftarrow [f_{i1}\ f_{i2}\ \dots\ 0\ \dots\ f_{iK_+}]$
$\mathcal{U}_i^* = \mathcal{U}_i - \{k^*\}$
$Q_{f-\text{fwd}} \leftarrow \frac{1-p_{\text{birth}}(\mathcal{U}_i)}{|\mathcal{U}_i|}$
Obtain HMM parameters
$\hat{\theta}^*, \hat{\eta}^{*(i)} \leftarrow$ `DeterministicHMMParams`($i, \boldsymbol{y}, \boldsymbol{f}_i^*, \boldsymbol{z}$)
Sample proposed state sequence
$z^{*(i)}, Q_{z-\text{fwd}} \sim$ `SampleStateSequence`$(\boldsymbol{y}^{(i)}, \tilde{\boldsymbol{y}}^{(i)}, \boldsymbol{f}_i^*, \hat{\theta}^*, \hat{\eta}^{*(i)}$)
Consider reverse move (birth)
$\bar{\boldsymbol{f}}_i \leftarrow [f_{i1}\ f_{i2}\ \dots\ 1\ \dots\ f_{iK_+}]$
$Q_{f-\text{rev}} \leftarrow p_{\text{birth}}(\mathcal{U}_i^*)$
Obtain HMM parameters, using modified state sequence
$\bar{z}_t^{(i)} \leftarrow \begin{cases} k^* & \text{if } t \in \mathcal{W} \\ z_t^{*(i)} & \text{o.w.} \end{cases}$
$\hat{\theta}, \hat{\eta}^{(i)} \leftarrow$ `DeterministicHMMParams`($i, \boldsymbol{y}, \bar{\boldsymbol{f}}_i, \bar{\boldsymbol{z}}$)
Determine probability of returning to original state seq.
$Q_{z-\text{rev}} \leftarrow$ `SampleStateSequence`$(\boldsymbol{y}^{(i)}, \tilde{\boldsymbol{y}}^{(i)}, \bar{\boldsymbol{f}}_i, \hat{\theta}, \hat{\eta}^{(i)}$, target=$\boldsymbol{z}$)

Algorithm D.4 DeterministicHMMParams($i, \boldsymbol{y}, \tilde{\boldsymbol{y}}, \boldsymbol{f}_i, \boldsymbol{z}$)

Get transition weights $\hat{\eta}^{(i)}$ and emission params $\hat{\theta}$ for all features active in $\boldsymbol{f}_i$

1: Construct transition weights using prior mean
  for $k \in \{k : f_{ik} = 1\}$:
    for $k' \in \{k : f_{ik} = 1\}$:
      $\hat{\eta}^{(i)}_{kk'} \leftarrow \gamma + \delta_{kk'}\kappa \triangleq \mathbb{E}[\text{Gamma}(\eta^{(i)}_{kk'}|\gamma + \delta_{kk'}\kappa, 1)]$
2: Obtain sufficient statistics and compute parameters of MNIW posterior distribution
  for $k \in \{k : f_{ik} = 1\}$:
    $S^{(k)}_{\cdot\cdot} \leftarrow$ via Eq. (B.12) using data $\boldsymbol{Y}_k = \{\boldsymbol{y}^{(i')}_t : \boldsymbol{z}^{(i')}_t = k\}$ from any seq. i'
    degrees of freedom $n_k \leftarrow n_0 + |\boldsymbol{Y}_k|$
    scale matrix $S_k \leftarrow S^{(k)}_{y|\tilde{y}}$
    mean matrix $M_k \leftarrow S^{(k)}_{y\tilde{y}}[S^{(k)}_{\tilde{y}\tilde{y}}]^{-1}$
    coef. scale matrix $K_k \leftarrow S^{(k)}_{\tilde{y}\tilde{y}}$
3: Construct emission parameters using posterior mean
  for $k \in \{k : f_{ik} = 1\}$:
    $\hat{\Sigma}_k \leftarrow \frac{S_k}{n_k - D - 1} \triangleq \mathbb{E}[\mathcal{IW}(\Sigma_k|n_k, S_k,\ \boldsymbol{Y}_k, \tilde{\boldsymbol{Y}}_k)]$
    $\hat{A}_k \leftarrow M_k \triangleq \mathbb{E}[\mathcal{MN}(A_k|M_k, \Sigma_k, K_k,\ \boldsymbol{Y}_k, \tilde{\boldsymbol{Y}}_k)]$
    $\hat{\theta}_k \leftarrow (\hat{A}_k, \hat{\Sigma}_k)$

D.2. Review of related work. In contrast to our new approach using a birth-death sampler in discrete assignments, previous work on the BP-AR-HMM as well as other beta process models used a variety of methods to make inferences about the number of features available. This includes truncation, proposals that modify *all* unique features for a sequence at once, and proposals similar to our birth-death approach that add or delete one unique feature at a time.

Let $K_+ = K_+^{-i} + n_i$, where n_i is the number of unique features chosen, and define $\boldsymbol{f}_{-i} = f_{i,1:K_+^{-i}}$ and $\boldsymbol{f}_{+i} = f_{i,K_+^{-i}+1:K_+}$. The posterior distribution over n_i is then given by

$$p(n_i \mid \boldsymbol{f}_i, \mathbf{y}^{(i)}_{1:T_i}, \boldsymbol{\eta}^{(i)}, \theta_{1:K_+^{-i}}, \alpha) \propto \frac{(\frac{\alpha}{N})^{n_i} e^{-\frac{\alpha}{N}}}{n_i!} \iint p(\mathbf{y}^{(i)}_{1:T_i} \mid \boldsymbol{f}_{-i}, \boldsymbol{f}_{+i} = \mathbf{1}, \boldsymbol{\eta}^{(i)}, \boldsymbol{\eta}_+, \theta_{1:K_+^{-i}}, \boldsymbol{\theta}_+)\, dB_0(\boldsymbol{\theta}_+) dH(\boldsymbol{\eta}_+), \tag{D.2}$$

where H is the gamma prior on transition variables $\eta^{(i)}_{jk}$, and B_0 is the base measure of the beta process. The set $\boldsymbol{\theta}_+ = \theta_{K_+^{-i}+1:K_+}$ consists of the parameters of unique features, and $\boldsymbol{\eta}_+$ the transition parameters $\eta^{(i)}_{jk}$ to or from unique features $j, k \in \{K_+^{-i}+1 : K_+\}$. Exact evaluation of this integral is intractable due to dependencies induced by the AR-HMMs.

One approach to this non-conjugacy relies on a truncation of the limiting Bernoulli process (Görür et al., 2006). That is, drawing $n_i \sim \text{Poisson}(\alpha/N)$

is equivalent to setting n_i equal to the number of successes in infinitely many Bernoulli trials, each with probability of success: $\lim_{K\to\infty} \frac{\alpha/K}{\alpha/K+N}$.

Görür et al. (2006) truncate this process, using K^* Bernoulli trials with probability $\frac{\alpha/K^*}{\alpha/K^*+N}$. Meeds et al. (2006) instead consider Metropolis proposals which replace the existing unique features by $n_i \sim \text{Poisson}(\alpha/N)$ new features, with parameters $\boldsymbol{\theta}_+$ drawn from the prior. For high-dimensional data, however, such moves have very low acceptance rates.

D.3. BP-HMM birth-death with continuous parameters. Motivated by the inadequacy of the related work, in past conference publications we developed a reversible jump MCMC sampler (Green, 1995), which proposes to either add one new feature ("birth"), or eliminate one existing feature in $\boldsymbol{f}_{+i}$ ("death") and also proposes the necessary sets of continuous HMM parameters θ, η, marginalizing away the state sequences $\boldsymbol{z}^{(i)}$.

We stress that our novel moves in the discrete assignment space appear in practice much better due to the avoidance of the curse of dimensionality that plagues any proposal of these continuous parameters. Our original work on the BP-HMM originally generated candidate emission parameters from the prior (Fox et al., 2009) and later we suggested an improved data-driven proposal distribution (Hughes et al., 2012). For completeness we briefly discuss both possible proposals below. Importantly, note that our data-driven proposal for emission parameters used a window selection process very similar in spirit to our current birth-death sampler.

We now outline the basic flavor of our continuous HMM parameter proposals. The sampler move takes as input a current configuration of unique feature assignments $\boldsymbol{f}_{+i}$, emission parameters for unique features θ_+, and transition weights for unique features for sequence i: η_+. We propose a new configuration $\boldsymbol{f}'_{+i}, \theta'_+, \eta'_+$. The proposal distribution factors as follows

$$q(\boldsymbol{f}'_{+i}, \boldsymbol{\theta}'_+, \boldsymbol{\eta}'_+ \mid \boldsymbol{f}_{+i}, \boldsymbol{\theta}_+, \boldsymbol{\eta}_+) = q_f(\boldsymbol{f}'_{+i} \mid \boldsymbol{f}_{+i}) \times q_\theta(\boldsymbol{\theta}'_+ \mid \boldsymbol{f}'_{+i}, \boldsymbol{f}_{+i}, \boldsymbol{\theta}_+) \times q_\eta(\boldsymbol{\eta}'_+ \mid \boldsymbol{f}'_{+i}, \boldsymbol{f}_{+i}, \boldsymbol{\eta}_+) \tag{D.3}$$

The feature proposal $q_f(\cdot \mid \cdot)$ encodes the probabilities of birth and death moves; we propose a birth with probability 0.5, and delete each of the n_i existing features with probability $0.5/n_i$. After an accepted move, we have $n'_i = n_i + 1$ features (birth), or $n'_i = n_i - 1$ (death). If the proposal is rejected, we maintain the original $n'_i = n_i$ unique features.

D.3.1. *Prior proposal for emission parameters.* In original work (Fox et al., 2009), we define our emission parameter proposal $q_\theta(\boldsymbol{\theta}'_+ \mid \boldsymbol{f}'_{+i}, \boldsymbol{f}_{+i}, \boldsymbol{\theta}_+)$

equal to:

$$\text{(D.4)} \qquad \begin{cases} b_0(\theta'_{+,n_i+1}) \prod_{k=1}^{n_i} \delta_{\theta_{+,k}}(\theta'_{+,k}), & \text{birth of feature } n_i+1; \\ \prod_{k \neq \ell} \delta_{\theta_{+,k}}(\theta'_{+,k}), & \text{death of feature } \ell. \end{cases}$$

Under a birth proposal, a new parameter θ'_{+,n_i+1} is drawn from the prior and all other parameters remain the same. For a death proposal of feature j, we simply eliminate that parameter from the model. Here, b_0 is the density associated with emission distribution prior: $\alpha^{-1}B_0$.

The distribution $q_\eta(\cdot \mid \cdot)$ is defined similarly. Under a birth, new parameters for transitions into the new state $\eta'_{+,\cdot,n_i+1}$ and away from it $\eta_{+,n_i+1,\cdot}$ are drawn using the gamma prior on transition weights: $\text{Gamma}(\alpha + \kappa\delta_{k,k'}, 1)$. Under a death move, we simply delete all parameters related to the state being deleted.

The Metropolis-Hastings acceptance probability is then given by

$$\text{(D.5)} \quad \rho(\boldsymbol{f}'_{+i}, \boldsymbol{\theta}'_+, \boldsymbol{\eta}'_+ \mid \boldsymbol{f}_{+i}, \boldsymbol{\theta}_+, \boldsymbol{\eta}_+) = \min\{r(\boldsymbol{f}'_{+i}, \boldsymbol{\theta}'_+, \boldsymbol{\eta}'_+ \mid \boldsymbol{f}_{+i}, \boldsymbol{\theta}_+, \boldsymbol{\eta}_+), 1\}.$$

We compactly represent the acceptance ratio $r(\cdot \mid \cdot)$ for either a birth or death move as

$$\text{(D.6)} \qquad \frac{p(\mathbf{y}^{(i)}_{1:T_i} \mid [\boldsymbol{f}_{-i}\, \boldsymbol{f}'_{+i}], \boldsymbol{\theta}', \boldsymbol{\eta}^{(i)}, \boldsymbol{\eta}'_+)\ \text{Poi}(n'_i \mid \alpha/N)\ q_f(\boldsymbol{f}_{+i} \mid \boldsymbol{f}'_{+i})}{p(\mathbf{y}^{(i)}_{1:T_i} \mid [\boldsymbol{f}_{-i}\, \boldsymbol{f}_{+i}], \boldsymbol{\theta}, \boldsymbol{\eta}^{(i)})\ \text{Poi}(n_i \mid \alpha/N)\ q_f(\boldsymbol{f}'_{+i} \mid \boldsymbol{f}_{+i})},$$

where we recall that $n'_i = \sum_k f'_{+ik}$. Because our birth and death proposals do not modify the values of existing parameters, the Jacobian term normally arising in reversible jump MCMC algorithms simply equals one.The proposal terms q_η and q_θ exactly cancel with the terms related to the joint probability $p(\theta)$, $p(\eta)$, so these need not appear in the simplified form of the acceptance ratio.

D.3.2. *Data-driven proposal for emission parameters.* Efficiently adding or deleting features is crucial for good mixing. In moderate- to high-dimensional applications, such as our motion capture, we observe that the birth move outlined above can have low acceptance rates for birth proposals. Because the emission parameters θ_{k^*} for the new feature are drawn from a potentially vague prior b_0, they are unlikely to yield appreciable gains in the likelihood $p(\mathbf{y}^{(i)}_{1:T_i} \mid \boldsymbol{\theta}', \ldots)$ due to the inevitable mismatch between the prior and the empirical distribution. That is, it is unlikely for a randomly sampled parameter to better explain the data than the currently instantiated, data-informed parameters.

To ameliorate these issues, we suggest a *data-driven* reversible jump proposal (Tu and Zhu, 2002; Hughes and Sudderth, 2012), which improves on the prior by combining it with the posterior of θ_{k^*} given data in a randomly chosen subwindow W of the current sequence. This proposal was thoroughly advocated in (Hughes et al., 2012). We draw θ_{k^*} from a proposal distribution which is a *mixture* of this posterior and the prior over θ (denoted $p_\theta(\cdot)$). Formally, the resulting proposal distribution is

$$q_\theta(\theta) = \frac{1}{2}p_\theta(\theta) + \frac{1}{2}p(\theta \mid \{\mathbf{y}_t^{(i)} : t \in W\}) \tag{D.7}$$

This mixture strikes a balance between successfully creating new features while also making death moves possible. We found experimentally that using only the posterior over the subwindow W as a proposal distribution had low acceptance of death moves due to the sharply-peaked reverse probability.

Under this data-driven proposal, the acceptance ratio r_{birth} for a birth move to a candidate state which adds feature k^* then becomes

$$\frac{p(\mathbf{y}_{1:T_i}^{(i)} \mid [\boldsymbol{f}_{-i}\, \boldsymbol{f}'_{+i}], \boldsymbol{\theta}, \theta_{k^*}, \boldsymbol{\eta}^{(i)}, \boldsymbol{\eta}'_{+}) \operatorname{Poi}(n'_i \mid \alpha/N)\, q_f(\boldsymbol{f}_{+i} \mid \boldsymbol{f}'_{+i}) p_\theta(\theta_{k^*})}{p(\mathbf{y}_{1:T_i}^{(i)} \mid [\boldsymbol{f}_{-i}\, \boldsymbol{f}_{+i}], \boldsymbol{\theta}, \boldsymbol{\eta}^{(i)}) \operatorname{Poi}(n_i \mid \alpha/N)\, q_f(\boldsymbol{f}'_{+i} \mid \boldsymbol{f}_{+i}) q_\theta(\theta_{k^*})}, \tag{D.8}$$

The above is similar to Eq. (D.6), but includes terms to account for the data-driven proposal of θ_{k^*}. As outlined above, the Jacobian remains unity since no existing parameters are modified. The acceptance ratio for a death move under the data-driven framework is the reciprocol of Eq. (D.8).

Note that no term in the acceptance ratio accounts for the choice of the subwindow W. We can omit such a term because when updating the unique features of a particular object i, W is chosen uniformly at random from all possible windows in time series i [1]. This choice is independent of the choice to perform birth or death moves, and its probability does not depend on current model parameters. Thus, each particular random choice of subwindow W defines a valid pair of birth-death moves satisfying detailed balance, and we need not include such a choice in an acceptance ratio (Tierney, 1994).

APPENDIX E: MCMC SPLIT-MERGE PROPOSALS

Here, we provide additional algorithmic details on the process of our novel split-merge moves. To review, our SM moves operate on the Markov state $\Psi = (\boldsymbol{F}, \boldsymbol{z})$ of the BP-HMM MCMC chain, where $\boldsymbol{F}$ denotes the binary

[1] We recommend enforcing duration requirement $L_{min} \leq |W| \leq L_{max}$ when selecting W, where L are user-specified parameters. This does not alter validity, but allows biasing the window to leverage sufficient data to achieve higher acceptance rates.

feature assignment matrix and $\boldsymbol{z} = \{z^{(1)} \dots z^{(N)}\}$ denotes the HMM state sequences for each time series in the collection. We will use f_i or $f^{(i)}$ to denote the i-th row of $\boldsymbol{F}$.

First, we provide a high-level overview of a split-merge move in Alg. E.1. Next, we give details on the construction of a *merge* proposal in Alg. E.3, complementing the algorithm provided in the main paper (Alg. 1) for a *split* proposal. Finally, we provide a restricted Gibbs (RG) sampling algorithm (Alg. E.4) for drawing individual feature assignments k_a, k_b for one sequence within the sequentially allocated split proposal. This includes important details about how the transition probability is calculated for the acceptance ratio. Our hope is that this information might provide a knowledgable practioner sufficient details to fully understand and reproduce our split-merge MCMC moves.

Overall Split-Merge Procedure. As outlined in the main paper, completing one SM update requires several steps: selecting anchor items, selecting features to determine the appropriate move (split or merge), constructing the proposed assignment variables $\boldsymbol{F}^*, \boldsymbol{z}^*$ under the chosen move, and finally accepting or rejecting via Metropolis-Hastings. This high-level procedure is specified in Alg. E.1.

In general, the proposal move (whether a split or a merge) involves selecting a pair of features to modify k_i, k_j and then sampling a candidate configuration $\boldsymbol{F}^*, \boldsymbol{z}^*$. To maintain detailed balance, in the acceptance ratio we consider *reversing* this move: selecting features to modify k_i^*, k_j^* from the new configuration $\Psi^* = (\boldsymbol{F}^*, \boldsymbol{z}^*)$ that will allow returning to the original configuration $\Psi = (\boldsymbol{F}, \boldsymbol{z})$. The generic formula for computing the acceptance ratio of a split-merge move, agnostic to whether a split or a merge is actually proposed, is then

$$\rho = \frac{p(\boldsymbol{y}, \tilde{\boldsymbol{y}}, \Psi^*)}{p(\boldsymbol{y}, \tilde{\boldsymbol{y}}, \Psi)} \frac{q(\Psi|\Psi^*, i, j, k_i^*, k_j^*)}{q(\Psi^*|\Psi, i, j, k_i, k_j)} \frac{q_k(k_i^*, k_j^*|\Psi^*, i, j)}{q_k(k_i, k_j|\Psi, i, j)} \tag{E.2}$$

Note that for simplicity, in our notation above (and in the algorithm) we have omitted the dependence on data $\boldsymbol{y}, \tilde{\boldsymbol{y}}$ in proposal distributions $q()$, though both the choice of features k_i, k_j and the proposal of Ψ^* do in fact depend crucially on this data.

To fill in details explicitly, the accept ratio for a split move that creates new features k_a, k_b from single feature $k_m = k_i = k_j$ is:

$$\rho_{\text{split}}|i, j = \frac{p(\boldsymbol{y}, \tilde{\boldsymbol{y}}, \Psi^*)}{p(\boldsymbol{y}, \tilde{\boldsymbol{y}}, \Psi)} \frac{q_{\text{merge}}(\Psi|\Psi^*, i, j, k_a, k_b)}{q_{\text{split}}(\Psi^*|\Psi, i, j, k_m, k_m)} \frac{q_k(k_a, k_b|\Psi^*, i, j)}{q_k(k_m, k_m|\Psi, i, j)} \tag{E.3}$$

Algorithm E.1 SampleSplitMerge($\boldsymbol{y}, \tilde{\boldsymbol{y}}, \boldsymbol{F}, \boldsymbol{z}, (\alpha, c), (\gamma, \kappa), \lambda$)

Input:

$\boldsymbol{y}, \tilde{\boldsymbol{y}}$: observations and lagged observations for all time series
$\Psi = (F, \mathbf{z})$: current state of Markov chain discrete variables
α, c : BP hyperparameters
γ, κ : HMM transition hyperparameters

Output:

$\Psi^{\text{new}} = (\boldsymbol{F}^{\text{new}}, \boldsymbol{z}^{\text{new}})$: new state of Markov chain discrete variables

Procedure:

Select anchor items i, j uniformly at random from all data items
Select a feature from each anchor item to modify
    $k_i, k_j, q_{k-\text{fwd}} \leftarrow$ `FeatureSelect`$(\boldsymbol{y}, \tilde{\boldsymbol{y}}, \Psi^*, i, j, \text{target feats.} = (k_a, k_b))$
if $k_i = k_j$
    $\Psi^*, q_{\text{fwd}} \leftarrow$ `SplitProposal`$(\ \Psi,\ \boldsymbol{y},\ \tilde{\boldsymbol{y}}, i, j, \text{orig.} = k_i, \text{new} = (k_a, k_b)\ )$
    $q_{k-\text{rev}} \leftarrow$ `FeatureSelect`$(\ \boldsymbol{y},\ \tilde{\boldsymbol{y}}, \Psi^*, i, j, \text{target feats.} = (k_a, k_b))$
    $q_{\text{rev}} \leftarrow$ `MergeProposal`$(\ \Psi^*,\ \boldsymbol{y},\ \tilde{\boldsymbol{y}}, i, j, \text{orig.} = (k_a, k_b), \text{new} = k_i, \text{target} = \Psi\ )$
else:
    $\Psi^*, q_{\text{fwd}} \leftarrow$ `MergeProposal`$(\ \Psi,\ \boldsymbol{y},\ \tilde{\boldsymbol{y}}, i, j, k_i, k_j, k_m\ )$
    $q_{k-\text{rev}} \leftarrow$ `FeatureSelect`$(\ \boldsymbol{y},\ \tilde{\boldsymbol{y}}, \Psi^*, i, j, \text{target feats.} = (k_m, k_m))$
    $q_{\text{rev}} \leftarrow$ `SplitProposal`$(\ \Psi^*,\ \boldsymbol{y},\ \tilde{\boldsymbol{y}}, i, j, \text{orig.} = k_m, \text{new} = (k_i, k_j), \text{target} = \Psi\ )$
Calc. joint log prob of current and proposed states via Eq. B.2
    $L^* \leftarrow p(\boldsymbol{y},\ \tilde{\boldsymbol{y}}, \Psi^*), \qquad L \leftarrow p(\boldsymbol{y},\ \tilde{\boldsymbol{y}}, \Psi)$
Compute acceptance ratio ρ

$$\rho \leftarrow \frac{L^*}{L} \frac{q_{\text{rev}}}{q_{\text{fwd}}} \frac{q_{k-\text{rev}}}{q_{k-\text{fwd}}} \triangleq \frac{p(\boldsymbol{y}, \tilde{\boldsymbol{y}}, \Psi^*)}{p(\boldsymbol{y}, \tilde{\boldsymbol{y}}, \Psi)} \frac{q(\Psi|\Psi^*, i, j, k_i^*, k_j^*)}{q(\Psi^*|\Psi, i, j, k_i, k_j)} \frac{q_k(k_i^*, k_j^*|\Psi^*, i, j)}{q_k(k_i, k_j|\Psi, i, j)} \tag{E.1}$$

Decide whether to accept or reject

$$\Psi^{\text{new}} \leftarrow \begin{cases} \Psi^* & \text{with prob. } \min(\rho, 1), \\ \Psi & \text{otherwise} \end{cases}$$

Similarly, the ratio for a merge move that combines features k_a, k_b into a single new feature k_m

$$\rho_{\text{merge}}|i, j = \frac{p(\boldsymbol{y}, \tilde{\boldsymbol{y}}, \Psi^*)}{p(\boldsymbol{y}, \tilde{\boldsymbol{y}}, \Psi)} \frac{q_{\text{split}}(\Psi|\Psi^*, i, j, k_m, k_m)}{q_{\text{merge}}(\Psi^*|\Psi, i, j, k_a, k_b)} \frac{q_k(k_m, k_m|\Psi^*, i, j)}{q_k(k_a, k_b|\Psi, i, j)} \tag{E.4}$$

Next, we focus on the proposal construction procedures, which we call `MergeProposal` and `SplitProposal` in Alg. E.1. Note that the algorithm in the main paper specifies `SplitProposal`. Here, we provide the complementary sequential allocation procedure used to create a merge proposal.

E.1. Feature Selection Distribution. A crucial step in the split-merge framework for our latent feature model is the selection of which fea-

tures to modify, which directly determines whether we split or merge. The formal process of feature selection is outlined in Alg. E.2.

As described in the main text (Section 6), this procedure takes in the current sampler configuration and two anchor sequences, and samples two feature labels k_i, k_j such that $f_{ik_i} = 1$ and $f_{jk_j} = 1$. Naively sampling k_i, k_j uniformly at random will frequently result in merges that do not make sense, so we first select k_i at random, and then bias the selection of k_j so that features that explain similar data as k_i will be more likely to be selected. This is done via a marginal likelihood ratio, which indicates how much the emission model prefers to explain the data attached to k_i, k_j together versus separately. Additionally, when possible (i.e. when $f_{jk_i} = 1$) we encourage the selection of $k_j = k_i$ (which results in a split move proposal) by enforcing that this choice has probability 2/3). This requires careful bookkeeping but is perfectly valid.

E.2. Merge Proposal using Sequential Allocation . Constructing a merge proposal, which combines input features k_a, k_b into a new feature k_m, follows a sequential allocation process, similar to the `SplitProposal` outlined in the main paper (Alg. 1). Here, given fixed choice of k_a, k_b the proposed feature assignments are deterministic: we set $f^*_{\ell,k_m} = 1$ if ℓ is in the active set $\mathcal{S}$, and $f^*_{\ell,k_m} = 0$ otherwise. However, we still completely update the state sequences of the active set. Alg. E.3 specifies the necessary steps. Note that sequential allocation is used here to iteratively improve the emission parameters used for the new "merged" feature k_m.

E.3. Sampling Feature Assignments for Split Proposal. During a split proposal (outlined in Alg. 1), we visit each sequence ℓ within the active set and choose its binary assignments $f^{(\ell)}_{k_a,k_b}$ to the new features k_a, k_b. The algorithm for updating $f^{(\ell)}_{k_a,k_b}$ is given in Alg. E.4.

`SampleSplitFeatures` is a restricted Gibbs (RG) algorithm that is successively applied to each individual sequence. We sweep through the active set $\mathcal{S}$ and for each item ℓ we sample only assignments to new features k_a, k_b based on the assignments made to previously visited sequences. We call this set of previously visited items $\mathcal{S}_{\text{prev}}$, which is a subset of $\mathcal{S}$. The possible binary assignments made are *restricted*: we must enforce that each item $\ell \in \mathcal{S}$ possesses at least one of the new features.

Two things are important to highlight about this process. First, we must track the proposal probabilities q_f of making each random choice. These are used in the acceptance ratio of Alg. E.1. Performing this bookkeeping using logarithms allows more numerically stable computations. Second, we must

Algorithm E.2 FeatureSelect($\boldsymbol{y}$, $\tilde{\boldsymbol{y}}, \boldsymbol{F}, \boldsymbol{z}, i, j$)

Input:

$\boldsymbol{y}, \tilde{\boldsymbol{y}}$: observations and lagged observations

$\boldsymbol{F}, \boldsymbol{z}$: current discrete assignments for feature possession and state sequences.

i, j : anchor sequence ids

Optional Input:

$k_i^{\text{target}}, k_j^{\text{target}}$: specific feature ids to choose.

If provided, just compute probability q_k of choosing these features.

Output:

k_i : feature possessed by i to modify. Must satisfy $f_{ik_i} = 1$.

k_j : feature possessed by j to modify. Must satisfy $f_{jk_j} = 1$.

$q_k \triangleq q_k(k_i, k_j | \boldsymbol{y}, \tilde{\boldsymbol{y}}, \boldsymbol{F}, \boldsymbol{z}, i, j)$: probability of choosing k_i, k_j via this procedure

Procedure:

1: Select k_i uniformly from all features possessed by i:

$k_i \sim \text{Unif}(\ \{k : f_{ik} = 1\}\ ), \quad q_{k_i} = \frac{1}{|\{k : f_{ik}=1\}|}$

2: For each candidate feature id for k_j, give it a selection probability p_k

High probability for features more similar to k_i

for $k \in \{k : f_{jk} = 1\}$:

$p_k \leftarrow m(Y_k \cup Y_{k_i}) / [m(Y_k) \cdot m(Y_{k_i})]$

where $Y_k \triangleq \{y_t^{(\ell)} : z_t^{(\ell)} = k\}, \tilde{Y}_k \triangleq \{\tilde{y}_t^{(\ell)} : z_t^{(\ell)} = k\}$,

and the marginal likelihood (Eq. B.11) is

$$
\begin{aligned}
m(Y_k) &\triangleq p(Y_k | \tilde{Y}_k, M, K, S_0, n_0) \\
&= \int \int p(Y_k | \tilde{Y}_k, \boldsymbol{A}_k, \Sigma_k) p(\boldsymbol{A}_k | M, \Sigma_k, K) p(\Sigma_k | n_0, S_0) d\Sigma_k d\boldsymbol{A}_k \\
&= \frac{1}{(2\pi)^{\frac{n_k d}{2}}} \frac{\Gamma_d(\frac{n_k+n_0}{2})}{\Gamma_d(\frac{n_0}{2})} \frac{|S_0|^{\frac{n_0}{2}}}{|S_{y|\tilde{y}}^{(k)}|^{\frac{n_k+n_0}{2}}} \frac{|K|^{\frac{1}{2}}}{|S_{\tilde{y}\tilde{y}}^{(k)}|^{\frac{1}{2}}},
\end{aligned}
$$

3: Compute prob. of choosing a feature distinct from k_i (leads to a merge move)

$$C_j = \sum_{k : f_{jk}=1} \mathbb{I}[k \neq k_i] p_k, \quad \mathbb{I} \text{ is the indicator function}$$

4: If a split move (selecting $k_i = k_j$) is possible, make this move have 2/3 probability

if $f_{jk_i} = 1$: $p_{k_i} = 2C_j$

5: Sample the feature assignment for k_j:

$k_j \sim \text{Discrete}(\vec{p})$ where $\vec{p}$ has length K_+ and entry $p_k = 0$ unless $f_{jk} = 1$

6: **If** target features specified:

Force output features to target values. $k_i \leftarrow k_i^{\text{target}}, k_j \leftarrow k_j^{\text{target}}$

7: Compute prob. of sampling this outcome: $q_k \leftarrow p_{k_j} q_{k_i}$

be able to compute the probability of *reverse* proposals, which are needed to satisfy the detailed-balance requirement of our Metropolis-Hastings split-merge proposal framework.

That is, if we perform a merge from Ψ to Ψ^*, the acceptance ratio requires

Algorithm E.3 MergeProposal(Ψ,$\boldsymbol{y}$, $\tilde{\boldsymbol{y}}$, i, j, k_a, k_b, k_m)

Input:

$\Psi = (\boldsymbol{F}, \boldsymbol{z})$: current discrete assignments in Markov chain state

$\boldsymbol{y}, \tilde{\boldsymbol{y}}$: observations and lagged observations

i, j : anchor sequences ids

k_a, k_b : ids of features to merge

k_m : intended label of new

Optional Input:

Ψ^{target} : target configuration of $\boldsymbol{F}, \boldsymbol{z}$

If provided, do not sample. Instead return probability of sampling Ψ^{target}

Output:

$\Psi = (\boldsymbol{F}, \boldsymbol{z})$: proposed discrete assignments in Markov chain state

will have one fewer feature than input config.

Procedure:

1: Construct active set $\mathcal{S} = \{\ell : f_{\ell k_a} = 1 \text{ or } f_{\ell k_b} = 1\}$

2: Propose new feature matrix.

Start with input assignments: $\boldsymbol{F}^* \leftarrow \boldsymbol{F}$

Delete old features k_a, k_b: **for** $\ell \in \mathcal{S}$: $f^*_{\ell k_a} = 0, \quad f^*_{\ell k_b} = 0$

Add new feature k_m: **for** $\ell \in \mathcal{S}$: $f^*_{\ell k_m} = 1$

3: Initialize state sequence with old features replaced with k_m

$\boldsymbol{z}^* \leftarrow \boldsymbol{z}$

$\boldsymbol{z}^{*(i)}_t \leftarrow k_m$ if $z^{(i)}_t = k_a$ or $z^{(i)}_t = k_b$

$\boldsymbol{z}^{*(j)}_t \leftarrow k_m$ if $z^{(j)}_t = k_a$ or $z^{(j)}_t = k_b$

4: Obtain non-random transition and emission parameters

$\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\eta}} \leftarrow$ `DeterministicHMMParams`$(\boldsymbol{y}, \tilde{\boldsymbol{y}}, \boldsymbol{F}^*, \boldsymbol{z}^*)$ (Alg. D.4).

5: **for** non-anchor items ℓ in active set $\mathcal{S}$:

sample state sequence

$z^{*(\ell)} \sim$ `SampleStateSequence`$(\ell, \boldsymbol{y}, \boldsymbol{f}^*_\ell, \hat{\theta}, \hat{\eta}^{(\ell)})$ (Alg. C.3)

update emission parameters to reflect recent assignment

$\hat{\theta}_{k_m} \leftarrow$ posterior mean of $p(\theta_{k_m} | \mathbf{y}, \mathbf{z}^*)$

6: Sample state sequence for anchor items i, j

$z^{*(i)} \sim$ `SampleStateSequence`$(i, \boldsymbol{y}, \boldsymbol{f}^*_i, \hat{\theta}, \hat{\eta}^{(i)})$

$z^{*(j)} \sim$ `SampleStateSequence`$(j, \boldsymbol{y}, \boldsymbol{f}^*_j, \hat{\theta}, \hat{\eta}^{(j)})$

computing the probability of obtaining Ψ from Ψ^* via a split. In this setting, we know in advance the destination state Ψ, so we don't actually need to sample anything, but we do need to walk-through the sampling process step by step and tally the probability of making each discrete choice that leads from Ψ^* back to Ψ. Practically, this can be achieved by augmenting the sampling procedure to take this desired destination state as an optional argument. When the split construction process is called with this known destination state $\Psi^{\text{target}} = (\boldsymbol{F}^{\text{target}}, \boldsymbol{z}^{\text{target}})$, the procedure does not randomly assign variables, but instead computes the probability of assigning to the provided target values.

Note that an analogous reverse probability calculation occurs in Alg. C.3

Algorithm E.4 SampleSplitFeatures($\ell, [k_a, k_b], \boldsymbol{F}, \hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\eta}}, \boldsymbol{y}^{(\ell)}, \mathcal{S}_{\text{prev}}, \mathbf{F}^{\text{target}}$)

Input:

ℓ : id of sequence to update
$[k_a, k_b]$: ids of features created by current split move
$\boldsymbol{F}$: binary feature matrix
$\hat{\theta}, \hat{\eta}$: deterministic HMM emission, transition parameters
$\mathcal{S}_{\text{prev}}$: set of previously updated items
$\boldsymbol{y}, \tilde{\boldsymbol{y}}$: observations and lagged observations

Optional Input:

$\boldsymbol{F}^{\text{target}}$: desired output configuration (for computing reverse probabilities

Output:

$f_{\ell,[k_a,k_b]}$: updated feature assignments for item ℓ
Note enforced restriction: item ℓ must possess at least one of k_a, k_b
q_f : prob. of sampling new feature assignment under this proposal

Procedure:

1: Define m_a, m_b as counts of how many previously seen items possess features k_a, k_b.

$$m_a = \sum_{n \in \mathcal{S}_{\text{prev}}} f_{n,k_a} \qquad m_b = \sum_{n \in \mathcal{S}_{\text{prev}}} f_{n,k_b} \qquad N_{\text{prev}} = |\mathcal{S}_{\text{prev}}|$$

2: Construct Gibbs conditional probabilities for each possible value of $f_{\ell,[k_a k_b]}$

$$p_{[1\ 0]} \leftarrow \frac{m_a}{N_{\text{prev}} + c} \frac{N_{\text{prev}} - m_b + c}{N_{\text{prev}} + c} p(\boldsymbol{y}^{(\ell)} | \tilde{\boldsymbol{y}}, \boldsymbol{f}_\ell([1\ 0]), \hat{\eta}^{(\ell)}, \hat{\boldsymbol{\theta}})$$

$$p_{[0\ 1]} \leftarrow \frac{N_{\text{prev}} - m_a + c}{N_{\text{prev}} + c} \frac{m_b}{N_{\text{prev}} + c} p(\boldsymbol{y}^{(\ell)} | \tilde{\boldsymbol{y}}, \boldsymbol{f}_\ell([0\ 1]), \hat{\eta}^{(\ell)}, \hat{\boldsymbol{\theta}})$$

$$p_{[1\ 1]} \leftarrow \frac{m_a}{N_{\text{prev}} + c} \frac{m_b}{N_{\text{prev}} + c} p(\boldsymbol{y}^{(\ell)} | \tilde{\boldsymbol{y}}, \boldsymbol{f}_\ell([1\ 1]), , \hat{\eta}^{(\ell)}, \hat{\boldsymbol{\theta}})$$

where $\boldsymbol{f}_\ell([a\ b])$ is a length K_+ vector whose k-th entry is: $\begin{cases} a & \text{if } k = k_a \\ b & \text{if } k = k_b \\ f_{\ell k} & \text{o.w.} \end{cases}$

3: Compute normalization constant: $Z_p \leftarrow p_{[1\ 0]} + p_{[0\ 1]} + p_{[1\ 1]}$
4: **if** $\boldsymbol{F}^{\text{target}}$ exists: **then**
5: $\quad f_{\ell,[k_a\ k_b]} \leftarrow f^{\text{target}}_{\ell,[k_a\ k_b]}$
6: **else**
7: $\quad f_{\ell,[k_a\ k_b]} \sim \begin{cases} [1\ 0] & \text{with prob. } p_{[1\ 0]}/Z_p \\ [0\ 1] & \text{with prob. } p_{[0\ 1]}/Z_p \\ [1\ 1] & \text{with prob. } p_{[1\ 1]}/Z_p \end{cases}$
8: **end if**
9: $q_f \leftarrow \begin{cases} p_{[1\ 0]}/Z_p \text{ if } f_{\ell,[k_a k_b]} = [1\ 0] \\ p_{[0\ 1]}/Z_p \text{ if } f_{\ell,[k_a k_b]} = [0\ 1] \\ p_{[1\ 1]}/Z_p \text{ if } f_{\ell,[k_a k_b]} = [1\ 1] \end{cases}$ *Record prob. of sampling new feature vector*

for sampling the state sequence $z^{(\ell)}$ for some item ℓ in both split and merge moves. A total probability q_z of transitioning from $\boldsymbol{z}$ to $\boldsymbol{z}^*$ is computed by

considering each sequence in turn and tallying up the transition probabilities at each discrete assignment made by the Gibbs block sampler of $\boldsymbol{z}^{(\ell)}$ given fixed $\boldsymbol{f}_\ell$ and point estimates $\hat{\boldsymbol{\theta}}, \hat{\eta}^{(\ell)}$. When computing reverse moves, a target configuration is provided, and rather than actually making random assignments the algorithm computes the probability of the sequence of choices that leads to the target state.

APPENDIX F: MCMC SAMPLING FOR HYPERPARAMETERS

Here we consider MCMC sampling updates for the hyperparameters of our BP-AR-HMM. First, we describe updates for the mass α and concentration c of the IBP prior on feature matrices, which determine the number of features and the level of sharing. Later, we outline the update for the dynamics hyperparameters γ, κ, which respectively influence the uniformity and stickiness of the transition distributions for each sequence.

F.1. Sampling IBP Hyperparameters. We use the two-parameter version of the IBP, as derived in (Ghahramani et al., 2006). We recall that this induces the following prior on *left-ordered form* feature matrices $\boldsymbol{F}$

(F.1)
$$p([\boldsymbol{F}]|\alpha, c) = \frac{\alpha^{K_+} c^{K_+}}{\prod_{h\geq 1} K_h!} \exp\left[-\alpha \sum_{i=1}^{N} \frac{c}{c+i-1}\right] \prod_{k=1}^{K_+} B(m_k, N - m_k + c)$$

where, as before, K_+ is the number of total distinct features with at least one positive entry in $\boldsymbol{F}$. As in Görür et al. (2006), we place a conjugate $\text{Gamma}(a_\alpha, b_\alpha)$ prior on α, which leads to the following posterior distribution on α:

$$p(\alpha \mid \boldsymbol{F}, c, a_\alpha, b_\alpha) \propto \alpha^{K_+} \exp\left(-\alpha \sum_{n=1}^{N} \frac{c}{c+n-1}\right) \cdot \alpha^{a_\alpha - 1} \exp(-b_\alpha \alpha)$$

(F.2)
$$= \text{Gamma}\left(a_\alpha + K_+, b_\alpha + \sum_{n=1}^{N} \frac{c}{c+n-1}\right)$$

There is no corresponding conjugate prior for c. Using a $\text{Gamma}(a_c, b_c)$ prior, we have the following posterior

$$p(c \mid \boldsymbol{F}, \alpha, a_c, b_c) \propto c^{K_+} \exp\left(-\alpha \sum_{n=1}^{N} \frac{c}{c+n-1}\right) \prod_{k=1}^{K_+} B(m_k, N - m_k + c)$$

(F.3)
$$\cdot\, c^{a_c - 1} e^{-b_c c}$$

To sample c, we use a Gamma random walk proposal distribution $q_c(\cdot)$ with mean set to the current value and fixed variance σ_c^2. This is parameterized as $q_c(c'|c) = \text{Gamma}(\frac{c^2}{\sigma_c^2}, \frac{c}{\sigma_c^2})$. To decide to accept or reject this proposal, the usual Metropolis-Hastings ratio is applied. We accept with probability $\min(1, r)$, where

$$r = \frac{p(\boldsymbol{F}|\alpha, c')p(c'|a_c, b_c)}{p(\boldsymbol{F}|\alpha, c)p(c|a_c, b_c)} \cdot \frac{q_c(c|c')}{q_c(c'|c)} \tag{F.4}$$

Let $a = c^2/\sigma_c^2$ and $b = c/\sigma_c^2$. Similarly, let $a' = c'^2/\sigma_c^2$ and $b' = c'/\sigma_c^2$. The ratio of proposal terms reduces to

$$\frac{q(c|c')}{q(c'|c)} = \frac{\text{Gamma}(c|a', b')}{\text{Gamma}(c'|a, b)} = \frac{\frac{b'^{a'}}{\Gamma(a')}}{\frac{b^a}{\Gamma(a)}} \frac{c^{a'-1}e^{-b'c}}{c'^{a-1}e^{-bc'}} \tag{F.5}$$

We can thus simplify the overall ratio r as:

$$\begin{aligned} r &= \frac{\ell(c')}{\ell(c)} \frac{(c')^{a_c-1}e^{-b_c c'}}{c^{a_c-1}e^{-b_c c}} \frac{\Gamma(a)}{\Gamma(a')} \frac{(b')^{a'}}{b^a} \frac{c^{a'-1}e^{-b'c}}{c'^{a-1}e^{-bc'}} \\ &= \frac{\ell(c')}{\ell(c)} \frac{(c')^{a_c+a'-a}e^{-(b_c-b)c'}}{c^{a_c+a-a'}e^{-(b_c-b')c}} \frac{\Gamma(a)}{\Gamma(a')} (\sigma_c^2)^{a-a'} \\ &= \frac{\ell(c')}{\ell(c)} \frac{(c')^{a_c+a'-a}}{c^{a_c+a-a'}} e^{-(c'-c)b_c} \frac{\Gamma(a)}{\Gamma(a')} (\sigma_c^2)^{a-a'} \end{aligned} \tag{F.6}$$

where we used the facts and definitions $b^a = c^a/(\sigma_c^2)^a$, $b'c = c'b$. We further define $\ell(c) \triangleq p(\boldsymbol{F}|\alpha, c)$ is

$$\ell(c) = c^{K_+} \exp(-\alpha \sum_{n=1}^{N} \frac{c}{c+n-1}) \prod_{k=1}^{K_+} B(m_k, N - m_k + c) \tag{F.7}$$

F.2. Sampling Transition Parameters. Recall that the Markovian transition distribution π that produce the state sequence $\boldsymbol{z}$ is a deterministic transform of the sequence-specific weights η. The prior $\pi_{k\cdot}^{(i)}|f_{ik} = 1 \sim \text{Dir}([\ldots \gamma + \delta_{kk'}\kappa \ldots])$ is governed by two hyperparameters: a symmetric parameter γ and the additional sticky bias parameter κ.

Transition hyperparameters are assigned gamma priors $\gamma \sim \text{Gamma}(a_\gamma, b_\gamma)$ and $\kappa \sim \text{Gamma}(a_\kappa, b_\kappa)$. Because of non-conjugacy we rely on Metropolis-Hastings proposals to iteratively sample γ given κ, and κ given γ. This is formalized in Alg. F.2. Each step of this iterative procedure uses a Metropolis-Hastings random walk gamma proposal distribution $q_\gamma(\cdot \mid \cdot)$ or $q_\kappa(\cdot \mid \cdot)$, with

Algorithm F.1 SampleBPHypers()

Input:

$\boldsymbol{F} = \{\boldsymbol{f}_i\}$: binary feature assignments for all time series

α, c : IBP hyperparameters

Output:

α, c : new HMM hyperparameter values drawn from conditional posterior

Sampler Algorithm Settings:

σ_c^2 : proposal distribution variance

R_{IBP} : number of iterations to attempt proposals

Procedure:

for iterations $r = 1, 2, \ldots R_{\text{IBP}}$:

Sample $\alpha | c, \boldsymbol{F}$

$$\alpha \sim \text{Gamma}\left(a_\alpha + K_+, b_\alpha + \sum_{n=1}^{N} \frac{c}{c+n-1}\right)$$

Sample $c | \alpha, \boldsymbol{F}$

$$c' | c \sim \text{Gamma}(\text{mean} = c, \text{var} = \sigma_c^2)$$

$$c \leftarrow \begin{cases} c' \text{ with prob } \min(1, r), & r \text{ defined in Eq. F.6} \\ c \text{ otherwise} \end{cases}$$

Algorithm F.2 SampleHMMTransitionHypers()

Input:

$\boldsymbol{F} = \{\boldsymbol{f}_i\}$: binary feature assignments for all time series

$\boldsymbol{\eta} = \{\eta^{(i)}\}$: HMM transition weights for all time series

γ, κ : HMM transition hyperparameters

Output:

γ, κ : new HMM hyperparameters drawn from conditional posterior

Sampler Algorithm Settings:

$\sigma_\gamma^2, \sigma_\kappa^2$: proposal distribution variances

R_{HMM} : number of iterations to attempt proposals

Procedure:

for iterations $r = 1, 2, \ldots R_{\text{HMM}}$:

Sample $\gamma | \kappa, \boldsymbol{F}, \boldsymbol{\eta}$

$$\gamma' | \gamma \sim \text{Gamma}(\text{mean} = \gamma, \text{var} = \sigma_\gamma^2)$$

$$\gamma \leftarrow \begin{cases} \gamma' \text{ with prob. } \min(1, r_\gamma), & r_\gamma \text{ defined in Eq. F.11} \\ \gamma \text{ otherwise} \end{cases}$$

Sample $\kappa | \gamma, \boldsymbol{F}, \boldsymbol{\eta}$

$$\kappa' | \kappa \sim \text{Gamma}(\text{mean} = \kappa, \text{var} = \sigma_\kappa^2)$$

$$\kappa \leftarrow \begin{cases} \kappa' \text{ with prob. } \min(1, r_\kappa), & r_\kappa \text{ defined in Eq. F.12} \\ \kappa \text{ otherwise} \end{cases}$$

fixed variance σ_γ^2 or σ_κ^2, and mean set to the current value. This is random walk update is similar to the updates used to sample c in Alg. F.1.

Since the proposal distributions for γ and κ use fixed variance σ_γ^2 or σ_κ^2, and mean equal to the current hyperparameter value, we have

$$q_\gamma(\cdot \mid \gamma) = \text{Gamma}\left(\frac{\gamma^2}{\sigma_\gamma^2}, \frac{\gamma}{\sigma_\gamma^2}\right) \qquad q_\kappa(\cdot \mid \kappa) = \text{Gamma}\left(\frac{\kappa^2}{\sigma_\kappa^2}, \frac{\kappa}{\sigma_\kappa^2}\right). \tag{F.8}$$

Acceptance ratio for γ. To update γ given κ, the acceptance probability is $\min\{r(\gamma' \mid \gamma), 1\}$ with acceptance ratio

$$r(\gamma' \mid \gamma) = \frac{p(\boldsymbol{\pi} \mid \gamma', \kappa, \boldsymbol{F})}{p(\boldsymbol{\pi} \mid \gamma, \kappa, \boldsymbol{F})} \cdot \frac{p(\gamma' \mid a_\gamma, b_\gamma) q(\gamma \mid \gamma', \sigma_\gamma^2)}{p(\gamma \mid a_\gamma, b_\gamma) q(\gamma' \mid \gamma, \sigma_\gamma^2)} \tag{F.9}$$

where $\boldsymbol{\pi} = \{\pi^{(i)}\}$, the set of all sequence-specific transition parameters. We can simplify the right half of this ratio expression using our derivation of F.6. Recalling the definition $K_i = \sum_k f_{ik}$, the likelihood term may be written as

$$\begin{aligned} f(\gamma) &\triangleq p(\boldsymbol{\pi} \mid \gamma, \kappa, \boldsymbol{F}) \\ &= \prod_i \prod_{k=1}^{K_i} \left\{ \frac{\Gamma(\gamma K_i + \kappa)}{\left(\prod_{j=1}^{K_i - 1} \Gamma(\gamma)\right) \Gamma(\gamma + \kappa)} \prod_{j=1}^{K_i} \tilde{\pi}_{kj}^{(i)^{\gamma + \kappa \delta(k,j) - 1}} \right\}. \end{aligned} \tag{F.10}$$

Plugging this likelihood function in place of $\ell(\cdot)$ in Eq. F.6 and redefining appropriate parameters, the acceptance ratio reduces to

$$r_\gamma = \frac{f(\gamma')}{f(\gamma)} \frac{(\gamma')^{a_\gamma + a' - a}}{\gamma^{a_\gamma + a - a'}} e^{-(\gamma' - \gamma) b_\gamma} \frac{\Gamma(a)}{\Gamma(a')} (\sigma_\gamma^2)^{a - a'} \tag{F.11}$$

$$\text{where } a = \frac{\gamma^2}{\sigma_\gamma^2}, a' = \frac{\gamma'^2}{\sigma_\gamma^2}$$

Acceptance Ratio for κ. The corresponding acceptance ratio for κ is

$$r_\kappa = \frac{f(\kappa')}{f(\kappa)} \frac{(\kappa')^{a_\kappa + a' - a}}{\kappa^{a_\kappa + a - a'}} e^{-(\kappa' - \kappa) b_\kappa} \frac{\Gamma(a)}{\Gamma(a')} (\sigma_\kappa^2)^{a - a'} \tag{F.12}$$

$$\text{where } a = \frac{\kappa^2}{\sigma_\kappa^2}, a' = \frac{\kappa'^2}{\sigma_\kappa^2}$$

and where the likelihood function only involves the self-transition terms:

$$f(\kappa) \triangleq \prod_i \frac{\Gamma(\gamma K_i + \kappa)^{K_i}}{\Gamma(\gamma + \kappa)^{K_i}} \prod_{j=1}^{K_i} \tilde{\pi}_{jj}^{(i)^{\gamma + \kappa - 1}} \propto p(\boldsymbol{\pi} \mid \gamma, \kappa, \boldsymbol{F}). \tag{F.13}$$

APPENDIX G: TOY DATA EXPERIMENTS: BASIC SAMPLER

Here, we present experiments that verify our most basic proposed MCMC sampling scheme (without sophisticated moves for adding and removing features) using toy data. To add or delete features, we only use the basic birth-death moves outlined in D.3.1, with emission parameter proposals from the prior. This is done to demonstrate the basic functionality of the overall MCMC algorithm (outlined in C) in recovering the true latent structure of data created via the BP-AR-HMM generative process. We also include here a comparison to an alternative nonparametric time series model – the hierarchical Dirichlet process HMM (HDP-HMM) – to emphasize the benefits of our proposed feature-based model.

Note that later in Section H, we conduct thorough experiments comparing our many candidate algorithms for adding or deleting features, including data-driven birth-death moves and split-merge moves.

To study our proposed BP-HMM model, we first generated time-series by switching among 5 AR(1) models:

$$y_t^{(i)} = a_{z_t^{(i)}} y_{t-1}^{(i)} + e_t^{(i)}(z_t^{(i)}), \tag{G.1}$$

with $a_k \in \{-0.8, -0.6, -0.4, -0.2, 0, 0.2, 0.4, 0.6, 0.8\}$ and process noise covariance Σ_k drawn from an $\mathrm{IW}(3, 0.5)$ prior. The true feature assignments, shown in Fig. 1(b), were sampled from a truncated IBP (Ghahramani et al., 2006) using $\alpha = 10$ and then used to generate the sequences of Fig. 1(a). Each row of the feature matrix corresponds to one of the five time series, and the columns represent the different AR models with a white square indicating that a given time series uses that dynamical mode.

The resulting feature matrix, averaged over 10,000 MCMC samples, is shown in Fig. 1(c). We discard features which explain less than 2% of a time series observations, and ensure consistent labeling across iterations by mapping to the closest ground truth features via Hamming distance. The BP-HMM recovers most of the true latent structure. One noticeable discrepancy is the absence of a_4 (green in Fig. 1(a)) in the fifth time series, which occurs because this feature is used in less than 5% of that series. The nonparametric nature of the model also causes a "tail" in the estimated matrix because of the (infrequent) use of additional features.

G.1. Comparison to the HDP-AR-HMM. As an alternative to the BP-AR-HMM, one might propose an architecture based on the hierarchical Dirichlet process of Teh et al. (2006), such as the HDP-AR-HMMs of Fox et al. (2011a) tied together with a shared set of transition and dynamic parameters. For an HDP-AR-HMM truncated to L possible dynamical modes,

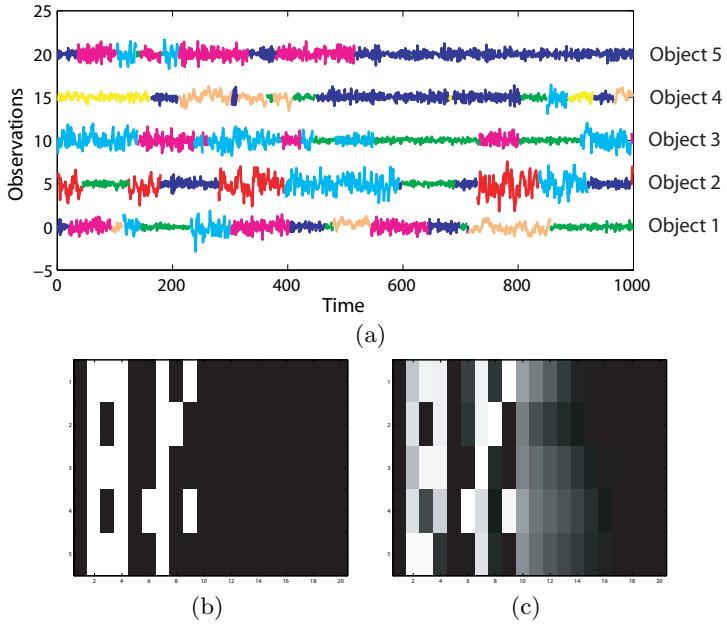

FIG 1. *(a) Observed sequences for 5 switching AR(1) time series colored by true mode sequence. Images of (b) true feature matrix and (c) estimated feature matrix averaged over 10,000 MCMC samples taken from 100 trials every 10th sample. Each row corresponds to a different time series, and each column a different autoregressive model. White indicates active features. Although the true model is defined by only 9 possible features, we show 20 columns to display the uncertain "tail" estimated by the BP-AR-HMM, due to some samples that instantiate additional modes to explain the data. The estimated feature matrices are produced from mode sequences mapped to the ground truth labels according to minimum Hamming distance, and discarding features assigned to* $\leq$ *2% of an item's observations.*

this model is specified as:

$$\begin{aligned} \beta &\sim \mathrm{Dir}(\gamma/L, \ldots, \gamma/L) \\ \pi_j \mid \beta &\sim \mathrm{Dir}(\alpha\beta_1, \ldots, \alpha\beta_{j-1}, \alpha\beta_j + \kappa, \alpha\beta_{j+1}, \ldots, \alpha\beta_L) \\ z_t^{(i)} &\sim \pi_{z_{t-1}^{(i)}}, \quad \mathbf{y}_t^{(i)} = \boldsymbol{A}_{z_t^{(i)}} \tilde{\mathbf{y}}_t^{(i)} + \mathbf{e}_t^{(i)}(z_t^{(i)}). \end{aligned} \tag{G.2}$$

Here, α and γ are a set of concentration parameters that define the HDP and κ is the sticky hyperparameter of the sticky HDP-HMM (Fox et al., 2011b); these hyperparameters are often given priors as well.

Segmentation Performance. To demonstrate the difference between this HDP-AR-HMM and the BP-AR-HMM, we generated data for three switching AR(1) processes. The first two time series, with four times the data points of the third, switched between dynamical modes defined by $a_k \in$

$\{-0.8, -0.4, 0.8\}$ and the third time series used $a_k \in \{-0.3, 0.8\}$. The results shown in Fig. 2 indicate that the multiple HDP-AR-HMM model, which assumes all time series share *exactly* the same transition matrices and dynamic parameters, typically describes the third time series using $a_k \in \{-0.4, 0.8\}$ since this assignment better matches the parameters defined by the other (lengthy) time series. This common grouping of two distinct dynamical modes leads to the large median and 90th Hamming distance quantiles shown in Fig. 2(b). The BP-AR-HMM, on the other hand, *does* distinguish these distinct modes (see Fig. 2(c)), since the penalty in not sharing a behavior is only in the feature matrix; once a unique feature is chosen, it does not matter how the time series employs it. Example segmentations representative of the median Hamming distance error are shown in Fig. 2(d)-(e). These results illustrate that the IBP-based feature model emphasizes *choosing behaviors* rather than assuming all time series perform minor variations of the same dynamics.

For the experiments above, we placed a Gamma$(1, 1)$ prior on α and γ, and a Gamma$(100, 1)$ prior on κ. The gamma proposals used $\sigma_\gamma^2 = 1$ and $\sigma_\kappa^2 = 100$ while the MNIW prior was given $M = 0$, $K = 0.1 * I_d$, $n_0 = d + 2$, and S_0 set to 0.75 times the empirical variance of the joint set of first-difference observations. At initialization, each time series was segmented into five contiguous blocks, with feature labels unique to that sequence.

Predictive Performance. Using the data generation process which created the sequences in Fig. 2(a), we generated a set of 100 held-out test datasets for Objects 1, 2, and 3. Each of the test sequences had length 1000 (in contrast to the training sequences, which had length 2000, 2000, and 500 respectively). Based on 2500 samples taken from 50 chains at MCMC iterations, we computed the log-likelihood of each of the 100 held-out datasets for both the MCMC samples of the BP-AR-HMM and HDP-AR-HMM. The results are summarized in the histogram of Fig. 3.

Since the BP-AR-HMM consistently identifies the unique dynamical mode of $a_k = -0.3$ used by Object 3 while the HDP-AR-HMM does not, we see from Fig. 3 that the mass of the BP-AR-HMM predictive log-likelihood is shifted positively by roughly 100 relative to the HDP-AR-HMM. In addition, the HDP-AR-HMM histogram has a heavy tail, skewed towards lower log-likelihood, whereas the BP-AR-HMM does not.

To summarize, while both the HDP-AR-HMM and BP-AR-HMM define global libraries of infinitely many behaviors, the HDP-AR-HMM assumes each sequence selects the *same* finite subset of behaviors and switches between them according to a *global* transition matrix. In contrast, the BP-

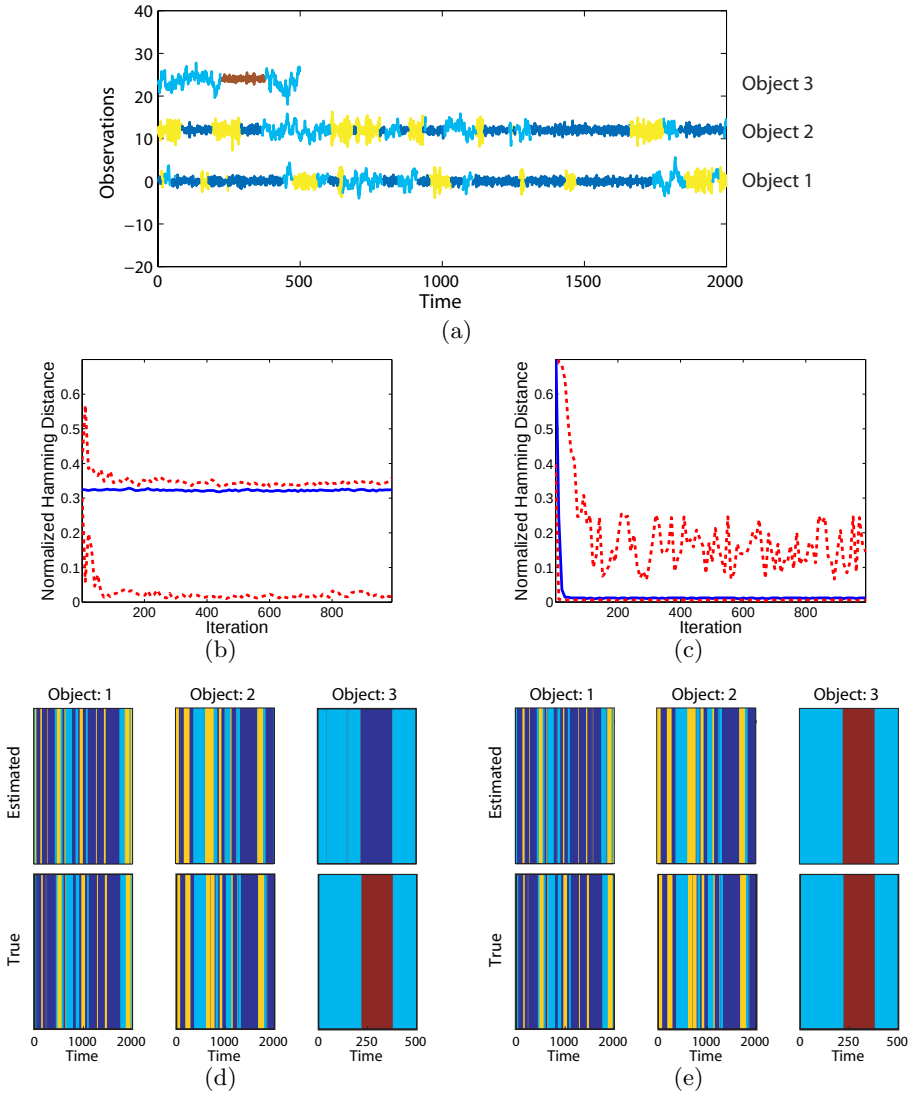

FIG 2. *(a) Observation sequences for 3 switching AR(1) time series colored by true mode sequence. The first two sequences are four times as long as the third. (b)-(c) Focusing solely on the third sequence, the median (solid blue) and* 10^{th} *and* 90^{th} *quantiles (dashed red) of Hamming distance from ground truth over 1000 trials are displayed for the HDP-AR-HMM model (Fox et al., 2011a) and the BP-AR-HMM, respectively. (d)-(e) Examples of typical segmentations into behavior modes for the three time series at MCMC iteration 1000 for the two models. The top and bottom panels display the estimated and true sequences, respectively, and the color coding corresponds exactly to (a). For example, time series 3 switches between two modes colored by cyan and maroon.*

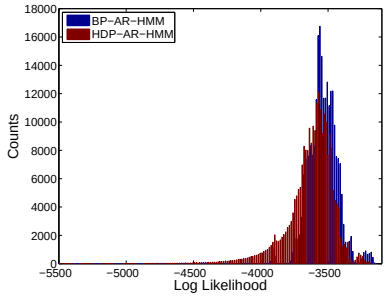

FIG 3. *Histogram of the predictive log-likelihood of 100 held-out data using the inferred parameters sampled every 10th iteration from MCMC iterations 500-1,000 from 50 independent chains for the BP-AR-HMM and HDP-AR-HMM run on the data of Fig. 2(a).*

AR-HMM allows each time series to select its own subset of behaviors *and* its own transition probabilities. The toy data analyzed here matched the HDP-AR-HMM assumptions of global transition matrix and mostly similar behaviors (for Objects 1 and 2). However, the BP-AR-HMM can better model the unique feature used in Object 3, which explains the improved predictive performance and illustrates the benefit of our flexible framework.

APPENDIX H: TOY DATA EXPERIMENTS: SPLIT-MERGE + DATA-DRIVEN SAMPLER

Here we investigate several possible sampling algorithms for adding or deleting features from our BP-HMM model. We use toy data with known ground-truth generative parameters, and compare each algorithm's efficiency.

We consider several candidate algorithms. First, the baseline birth-death moves with emission parameter proposals from the prior (Prior), as presented in Fox et al. (2009). Next, the continuous parameter data-driven birth-death moves (denoted "DD") from Hughes et al. (2012). We also consider two variants of our split-merge moves: (1) "SM-RGS": split-merge moves based on random allocation followed by restricted Gibbs sampling, and (2) "SM-Seq": sequential allocation spit-merge moves (as advocated in Sec. 6 and in Hughes et al. (2012)). Our "SM-RGS" approach attempts to faithfully represent a straight-forward extension of Jain and Neal (2004)'s algorithm for Dirichlet process mixture models to the BP-HMM setting. All key variables are sampled: $\boldsymbol{F}, \boldsymbol{z}, \boldsymbol{\theta}$, and $\boldsymbol{\eta}$, using modifications of the algorithms in C. We'll see that our proposed sequential allocation method (which only samples $\boldsymbol{F}, \boldsymbol{z}$) is much more efficient at adding needed features.

For every experiment, we interleave the chosen move for adding/deleting features with updates to shared features of $\boldsymbol{F}$, as well as updates to HMM

continuous parameters $\boldsymbol{\theta}, \boldsymbol{\eta}$ and block sampling of $\boldsymbol{z}$ as outlined in Sec. 5 and App. C. Note that we do *not* consider the more effective "zDD" data-driven birth-death moves that we use in the motion capture experiments in Sec. 8, as we found these to have similar performance as the regular "DD" moves on the simple toy datasets considered here.

For each dataset and sampling algorithm, we run many independent MCMC chains and rank from "best" to "worst" according to joint probability over the final 20% of the run. Each sampler runs for a fixed length of time, so comparisons are fair despite the fact that one SM iteration may be more costly than a Prior iteration. We fix the BP hyperparameter to $\alpha = 2$ and HMM hyperparameters to $\gamma = 2, \kappa = 200$, to match the high self-transition probability used to generate the data. This allows fair comparison of the different inference techniques for the parameters of interest: $\boldsymbol{F}, \boldsymbol{z}$.

We study performance on two synthetic datasets. Each dataset uses different emission distributions for observed sequence $\mathbf{y}$; we consider Gaussian emissions of 8 dimensions, and a VAR(1) process with 5 dimensions. These diverse formats allow validation that the differences between sampling methods do not depend on a particular choice of likelihood. Each dataset is generated using 8 true features with hand-chosen emission parameters θ.

First, we study how well each sampler *creates* necessary features from scratch. We initialize with just one feature used by all items, and examine how many true states are recovered after one hour of computer time across 10 runs. We show trace plots as well as illustrations of recovered emission parameters θ in Fig. 4. All runs of both SM-Seq and DD moves find all 8 true states within several minutes, while no Prior run or SM-RGS run recovers all true states, remaining stuck with merged versions of true features even after an hour of computation time. SM-RGS seems to fail because it must overcome a random initial configuration, and explore a higher-dimensional space (including θ, η). In contrast, our SM-Seq method only explores the space of discrete assignments, and intelligently allocates data items to newly-proposed features in just one sweep. Note that DD moves add new features most rapidly due to low computational cost, since each proposal requires only a single sequence's data, while SM moves must sweep through all sequences in the active set $\mathcal{S}$ to construct a single proposal.

Next, we study the ability of various BP-HMM samplers to *remove* features starting from a redundant initialization, diagrammed in Fig. 5(a). The data are generated by 8 true states. To create a difficult initial configuration, we divide the sequences into two sets. For the first set, we initialize both $\boldsymbol{f}_i$ and $\boldsymbol{z}^{(i)}$ to the ground truth. For the second set, we initialize with the

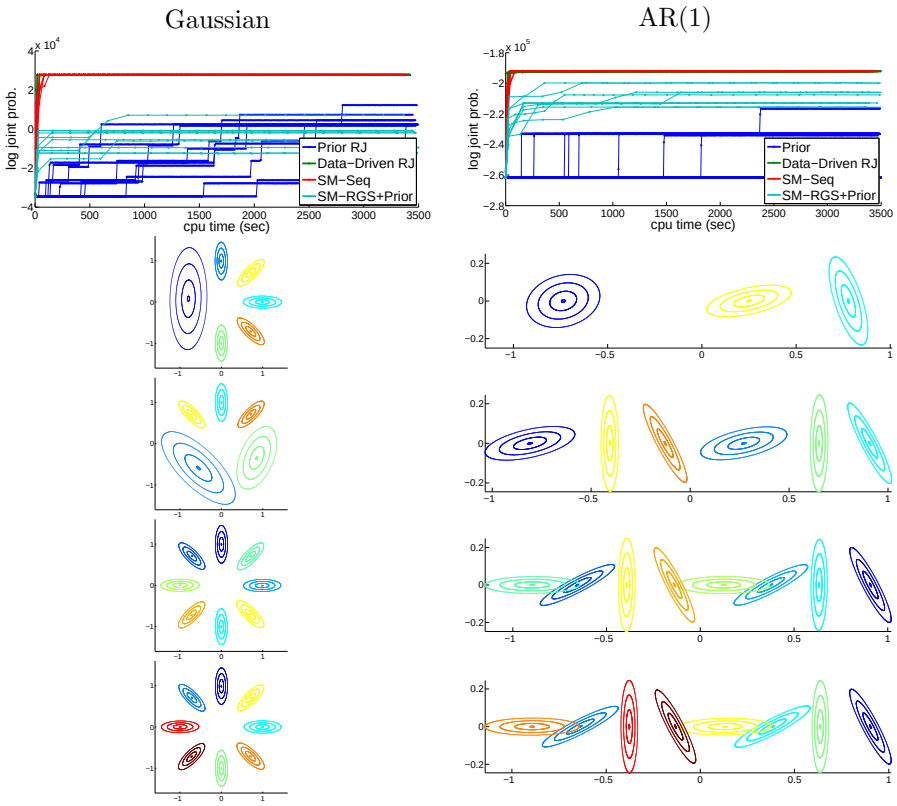

FIG 4. *Feature creation assessment on 100 toy data sequences generated by a BP-HMM with Gaussian (*left*) and AR (*right*) likelihoods. Each run begins with one feature used by all items, and must add new features via MCMC using either sequentially-allocated split-merge (SM-Seq), restricted Gibbs scan split-merge (SM-RGS), or reversible-jump moves with data-driven (DD) or prior (Prior) proposals.* Top: *Trace plots of log joint probability over time.* Bottom: *Emission parameters $\boldsymbol{\theta}$ for all hidden states recovered by each inference method after 1 hour of computer time. We show results for the* top*-ranked runs (by joint log probability) for baselines (Prior=row 1, SM-RGS=row 2) alongside the* worst*-ranked runs of our novel methods (SM-Seq=row 3,DD=row 4) to demonstrate the superiority of our new moves. Gaussian $\theta = (\mu, \Sigma)$ and AR $\theta = (A, \Sigma)$ shown as contour lines of the covariance matrix in first two dimensions, with location determined by μ, A.*

ground truth patterns for $\boldsymbol{f}_i$ and $\boldsymbol{z}^{(i)}$, but with feature labels shifted by 8. The duplication of features creates a challenging scenario for any sampling scheme since it must successfully merge down to one set of true states when each sequence is already stuck in a local optimum.

We show trace plots of all sampler methods and recovered feature matrices in Fig. 5. We observe that for Gaussian and AR likelihoods, the SM moves quickly converge down to a mode with all redundant states removed,

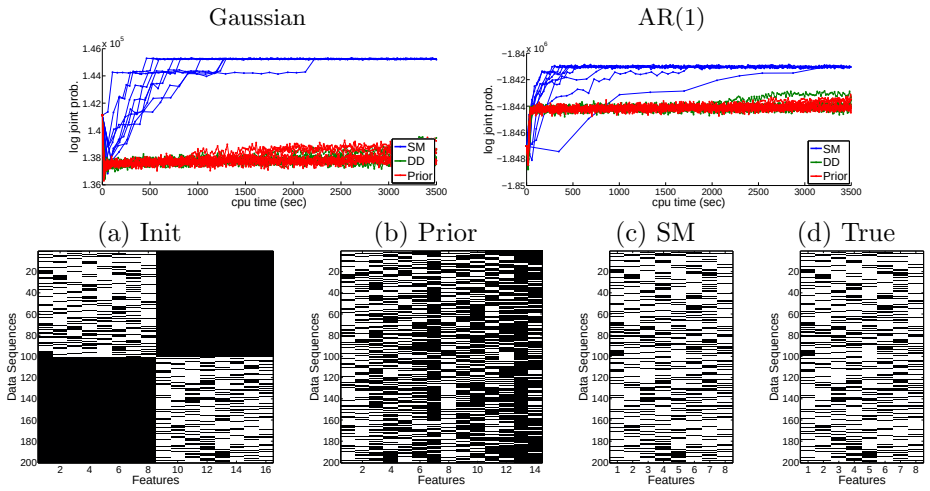

FIG 5. *Feature removal on a "redundant features" task using various sampling methods for BP-HMM toy data.* Top: *trace plots with various emission types.* Bottom: *Initial and recovered feature matrices* $\boldsymbol{F}$ *using autoregressive (AR) sequences. (a) Initial "redundant" configuration, with duplicate copies of the 8 true states, (b)* $\boldsymbol{F}$ *sampled after 1 hour by best Prior run, (c)* $\boldsymbol{F}$ *sampled after 1 hour by best SM-Seq run, and (d) ground truth* $\boldsymbol{F}$ *used in data generation process.*

sometimes within five minutes. In contrast, both DD and Prior reversible jump proposals never merge down to the 8 true features within the alotted hour. Overall, we observe that our sequentially-allocated SM moves make significant improvements over more local samplers in removing redudant features, regardless of the emissions used.

Together, these experiments demonstrate the importance of both DD and split moves for exploring new features, and merge moves for removing features via proposals that can change large portions of the feature assignments $\boldsymbol{F}$ simultaneously. As such, we recommend a sampler that interleaves SM and DD moves in practice. This combination leverages both the speed of the DD moves in feature creation and the efficiency of merge moves in feature deletion.

APPENDIX I: MOCAP EXPERIMENTS: HYPERPARAMETER SETTINGS

For all experiments on motion capture data, regardless of which inference algorithm was employed, we used the same hyperparameter settings. We describe these settings here.

HMM and IBP hyperparameters. We *sample* HMM transition hyperparameters γ, κ and IBP hyperparameters α, c at each iteration of every experiment. The algorithms used are in Alg. F.2 and Alg. F.1, respectively. Each of these procedures involves several iterations where one hyperparameter is sampled with the other fixed, then vice versa. We found that 10-25 iterations yielded reasonable acceptance rates without noticeable computational cost.

MNIW likelihood hyperparameters. The matrix-normal-inverse-Wishart likelihood requires several hyperparameters: scale matrix S_0, degrees of freedom ν_0, and column-wise precision matrix R_0. Together, these specify a prior on autoregressive coefficients A and covariance Σ for a first-order AR process with $D = 12$ dimensional observations as follows:

$$\Sigma \sim \mathcal{W}_D^{-1}(\nu_0, S_0) \tag{I.1}$$

$$A|\Sigma \sim \mathcal{MN}_D(\mathbf{0}, \Sigma, R_0) \tag{I.2}$$

We set $\nu_0 = D + 2$, S_0 to 0.5 times the empirical covariance of first differences of all observation sequences, and $R_0 = 0.5 I_D$, where I is the identity matrix with dimensions D-by-D. We note that in earlier published experiments (Fox et al., 2009) we set S_0 to 5 times the empirical covariance, rather than 0.5 as we do here and in Fox et al. (2009). We made this choice because with our improved inference, we need not specify such a vague prior. Realistically, *individual* discovered behaviors should have less variability than all motion sequences taken together.

We validated our better setting of these parameters by performing MCMC on a "cheating" initialization. Starting from the human annotated ground truth of the small 6 sequence motion capture dataset, we found that under these new hyperparameter settings the sampler stayed in the "ground truth" configuration throughout thousands of iterations. In contrast, a sampler initialized to truth under the more vague prior on S_0 quickly wandered away from this ground truth, gaining in likelihood by deleting some unique behaviors. Because our goal is comparing the segmentation recovered by sampler runs to the ground truth, we found it better to use hyperparameter settings in which ground truth is a (possibly local) mode of the posterior. Note that this difference in likelihood parameter settings might explain why in our experiments the BPHMM sampler from the *prior* performs somewhat worse in terms of Hamming distance (≈ 0.4 vs.≈ 0.3) on the 6 sequence dataset than reported earlier in Fox et al. (2009).

REFERENCES

Fox, E. B., E. B. Sudderth, M. I. Jordan, and A. S. Willsky (2009). Sharing features among dynamical systems with beta processes. In *Advances in Neural Information Processing Systems (NIPS) 22*, Vancouver, Canada.

Fox, E. B., E. B. Sudderth, M. I. Jordan, and A. S. Willsky (2011a). Bayesian nonparametric inference of switching dynamic linear models. *IEEE Transactions on Signal Processing 59*(4), 1569–1585.

Fox, E. B., E. B. Sudderth, M. I. Jordan, and A. S. Willsky (2011b). A sticky HDP-HMM with application to speaker diarization. *Annals of Applied Statistics 5*(2A), 1020–1056.

Ghahramani, Z., T. Griffiths, and P. Sollich (2006). Bayesian nonparametric latent feature models. In *Proc. of the Eighth Valencia International Meeting on Bayesian Statistics (Bayesian Statistics 8)*, Alicante, Spain.

Görür, D., F. Jäkel, and C. E. Rasmussen (2006). A choice model with infinitely many latent features. In *Proc. of the 23rd International Conference on Machine Learning (ICML)*, Pittsburgh, PA, USA.

Green, P. J. (1995). Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika 82*(4), 711–732.

Hughes, M., E. B. Fox, and E. B. Sudderth (2012). Effective split merge monte carlo methods for nonparametric models of sequential data. In *Advances in Neural Information Processing Systems (NIPS) 25*, Lake Tahoe, NV, USA.

Hughes, M. C. and E. B. Sudderth (2012). Nonparametric discovery of activity patterns from video collections. In *The Eighth IEEE Workshop on Perceptual Organization in Computer Vision (POCV)*, Providence, RI, USA.

Jain, S. and R. Neal (2004). A split-merge Markov chain Monte Carlo procedure for the Dirichlet process mixture model. *Journal of Computational and Graphical Statistics 13*(1), 158–182.

Kim, Y. (1999). Nonparametric Bayesian estimators for counting processes. *The Annals of Statistics 27*, 562–588.

Meeds, E., Z. Ghahramani, R. M. Neal, and S. T. Roweis (2006). Modeling dyadic data with binary latent factors. In *Advances in Neural Information Processing Systems (NIPS) 19*, Vancouver, Canada.

Rabiner, L. R. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE 77*(2), 257–286.

Scott, S. (2002). Bayesian methods for hidden Markov models: Recursive computing in the 21st century. *Journal of the American Statistical Association 97*(457), 337–351.

Teh, Y. W., M. I. Jordan, M. J. Beal, and D. M. Blei (2006). Hierarchical Dirichlet processes. *Journal of the American Statistical Association 101*(476), 1566–1581.

Thibaux, R. and M. I. Jordan (2007). Hierarchical beta processes and the Indian buffet process. In *Proc. of the Eleventh International Conference on Artificial Intelligence and Statistics (AISTATS)*, San Juan, Puerto Rico.

Tierney, L. (1994). Markov chains for exploring posterior distributions (with discussion). *The Annals of Statistics 22*(4), 1701–1728.

Tu, Z. and S. C. Zhu (2002). Image segmentation by data-driven Markov chain Monte Carlo. *IEEE Transactions on Pattern Analysis and Machine Intelligence 24*(5), 657–673.

Emily B. Fox
Department of Statistics
University of Washington
Box 354322
Seattle, WA 98195
USA
E-mail: ebfox@stat.washington.edu

Michael C. Hughes and Erik B. Sudderth
Department of Computer Science
Brown University
115 Waterman St, Box 1910
Providence, RI 02912
USA
E-mail: mhughes@cs.brown.edu
sudderth@cs.brown.edu

Michael I. Jordan
Department of Statistics and Department of EECS
427 Evans Hall
Berkeley, CA 94720
USA
E-mail: jordan@stat.berkeley.edu