2018 European Control Conference (ECC)
June 12-15, 2018. Limassol, Cyprus

Rao-Blackwellized Particle Gibbs Kernels for Smoothing in Jump Markov Nonlinear Models

Milan Papež[1]

Abstract— **Jump Markov nonlinear models (JMNMs) characterize a dynamical system by a finite number of presumably nonlinear and possibly non-Gaussian state-space configurations that switch according to a discrete-valued hidden Markov process. In this context, the smoothing problem – the task of estimating fixed points or sequences of hidden variables given all available data – is of key relevance to many objectives of statistical inference, including the estimation of static parameters. The present paper proposes a particle Gibbs with ancestor sampling (PGAS)-based smoother for JMNMs. The design methodology relies on integrating out the discrete process in order to increase the efficiency through Rao-Blackwellization. The experimental evaluation illustrates that the proposed method achieves higher estimation accuracy in less computational time compared to the original PGAS procedure.**

I. INTRODUCTION

Particle Markov chain Monte Carlo (PMCMC) methods [1] have recently emerged as an efficient tool to perform statistical inference in general state-space models (SSMs, [2]). These algorithms apply sequential Monte Carlo (SMC, [3]) to tackle the issue of constructing high-dimensional proposal kernels in MCMC [4]. This makes them particularly well suited for addressing the smoothing problem in jump Markov nonlinear models (JMNMs). The particle Gibbs with ancestor sampling (PGAS) kernel [5], which can be seen as a PMCMC smoother, has proved to be a serious competitor to the prominent SMC-based smoothing strategies such as the backward simulator [6] and generalized SMC two-filter smoother [7]. For a thorough review of existing SMC-based smoothers, see [8] and references therein.

The development in this paper is motivated by the recent progress in constructing PG kernels specifically tailored for jump Markov linear models (JMLMs) [9], [10]. The methods therein exploit the linear Gaussian substructure of the model to increase efficiency through Rao-Blackwellization. This is achieved by using the Kalman filter (KF) to design the conditional variants of either the discrete particle filter [11] or Rao-Blackwellized particle filter (RBPF, [12]). A common aspect of these PG methods lies in that the backward information filter (BIF, [13]) is used to further increase the effect of Rao-Blackwellization and to improve the mixing properties [4] via ancestor sampling or backward simulation.

The problem with JMNMs is that their nonlinear character prevents us from applying Rao-Blackwellization in the same sense as with JMLMs; nevertheless, there is still a tractable substructure to exploit. The present paper is concerned with the design of a Rao-Blackwellized PGAS (RBPGAS) kernel that takes advantage of the hierarchical structure formed by the discrete latent process. The method builds on the RBPF proposed in [14], which is similar to that introduced in [12] except it replaces the above-discussed KF with a finite state-space filter; conversely, the particle filter (PF) focuses on the remaining (continuous-valued) part of the latent process. However, the design of a finite state-space BIF turns out to be more intricate in this context as it requires us to introduce a sequence of artificial probability distributions to change the scale of the associated backward recursion.

This research has been financially supported by the Ministry of Education, Youth and Sports of the Czech republic under the project CEITEC 2020 (LQ1601).

[1]Milan Papež is with Central European Institute of Technology, Brno University of Technology, Technická 12, 616 00 Brno, Czech Republic, e-mail: milan.papez@ceitec.vutbr.cz

II. BACKGROUND

A. Jump Markov Nonlinear Models

The generic form of the discrete-time JMNM considered in the present paper is defined by

$$c_t|c_{t-1} \sim p(c_t|c_{t-1}), \tag{1a}$$

$$z_t|c_t, z_{t-1} \sim f(z_t|c_t, z_{t-1}), \tag{1b}$$

$$y_t|c_t, z_t \sim g(y_t|c_t, z_t), \tag{1c}$$

where the states and measurements are denoted by $z_t \in \mathsf{Z} \subseteq \mathbb{R}^{n_z}$ and $y_t \in \mathsf{Y} \subseteq \mathbb{R}^{n_y}$, respectively. The activity of the current regime of the model is indicated by the discrete *mode* variable $c_t \in \mathsf{C} \coloneqq \{1, \ldots, K\}$. We assume to have access only to the measurements y_t, while the state z_t and mode c_t variables are considered hidden. Furthermore, for all $c_t \in \mathsf{C}$, the model is characterized by its state transition and observation probability densities $f(\cdot)$ and $g(\cdot)$, respectively. The switching between the modes is governed by the conditional probability distribution $p(\cdot)$. At the initial time step, the hidden variables are distributed according to $z_1 \sim \mu(z_1|c_1)$ and $c_1 \sim p(c_1)$.

B. Problem Formulation

Let $x_{1:T} \coloneqq (x_1, \ldots, x_T)$ denote a generic sequence of variables defined on some product space X^T, for an integer $T > 0$ denoting the final time point. The aim of this study is to design an efficient PMCMC smoother targeting the joint smoothing density given by

$$p(c_{1:T}, z_{1:T}|y_{1:T}) = \frac{p(c_{1:T}, z_{1:T}, y_{1:T})}{p(y_{1:T})}. \tag{2}$$

However, the density (2) is intractable even in situations where (1b) and (1c) are linear and Gaussian. The reason consists in that the marginal likelihood $p(y_{1:T})$ contains

summation over K^T values, which is always impossible to compute exactly, except for small data sets. Despite this, we consider (1b) and (1c) nonlinear and non-Gaussian, making the situation more difficult as the integral over Z^T in the marginal likelihood $p(y_{1:T})$ cannot be evaluated either.

C. Sequential Monte Carlo

SMC [3] refers to a general class of algorithms suitable for approximating a sequence of (intractable) target densities $\{\pi_t(x_{1:t})\}_{t=1}^T$. We consider each of these densities to be defined on a product space X^t and to have the form

$$\pi_t(x_{1:t}) = \gamma_t(x_{1:t})/Z_t, \tag{3}$$

where $\gamma_t(x_{1:t})$ and $Z_t = \int \gamma_t(x_{1:t})dx_{1:t}$ most often constitute the complete data likelihood and the marginal likelihood, respectively, with $Z_t > 0$ for all $t = 1, \ldots, T$. The SMC approximation embodies an empirical measure represented by

$$\hat{\pi}_t(dx_{1:t}) = \sum_{i=1}^N w_t^i \delta_{x_{1:t}^i}(dx_{1:t}), \tag{4}$$

which is completely specified by the weighted particle system $\{x_{1:t}^i, w_t^i\}_{i=1}^N$. Here, the samples $\{x_{1:t}^i\}_{i=1}^N$ are termed particle trajectories and represent a hypothetical evolution of the true trajectory $x_{1:t}$, while the weights $\{w_t^i\}_{i=1}^N$ assess the contribution of the corresponding particle trajectories to the resulting approximation.

SMC methods are based on the repetitive use of sequential importance sampling and resampling in order to propagate (4) in time. Let us assume we have the previously generated particle system $\{x_{1:t-1}^i, w_{t-1}^i\}_{i=1}^N$. The *recursive step* of an SMC method begins with resampling. This procedure consists of sampling a set of ancestor indices $\{a_t^i\}_{i=1}^N$ from

$$\mathbb{P}(a_t^i = j) = w_{t-1}^j, \qquad j = 1, \ldots, N. \tag{5}$$

The indices are then applied in the sequential importance sampling approach. This proceeds by first drawing the particles $\{x_t^i\}_{i=1}^N$ from the proposal density as

$$x_t^i \sim q_t(\cdot|x_{1:t-1}^{a_t^i}), \tag{6}$$

where the index a_t^i is used to assign the parent trajectory to the offspring particle x_t^i. The set of ancestors $\{a_{1:t}^i\}_{i=1}^N$ thus serves for tracing the genealogy of the particles. Subsequently, we extend the previous trajectories according to

$$x_{1:t}^i \coloneqq \{x_{1:t-1}^{a_t^i}, x_t^i\}. \tag{7}$$

The recursive step is concluded by computing the normalized importance weights $w_t^i \propto W_t(x_{1:t}^i)$ for $i = 1, \ldots, N$, where the unnormalized weight function is defined by

$$W_t(x_{1:t}) \coloneqq \frac{\gamma_t(x_{1:t})}{\gamma_{t-1}(x_{1:t-1})q_t(x_t|x_{1:t-1})}. \tag{8}$$

The *initial step* applies the standard importance sampling, that is, we first draw the particles $\{x_1^i\}_{i=1}^N$ from the initial proposal density $x_1^i \sim q_1(\cdot)$, and then calculate the importance weights $w_1^i \propto W_1(x_1^i)$ for $i = 1, \ldots, N$, where $W_1(x_1) = \gamma_1(x_1)/q_t(x_1)$.

D. Particle Markov Chain Monte Carlo Smoothing

MCMC [4] constitutes a family of methods appropriate for approximating a target density $\pi(x)$ defined on some space X. The idea underlying MCMC is to simulate a Markov chain $\{x[k]\}_{k=1}^R$ according to

$$x[k] \sim \mathcal{K}(\cdot|x[k-1]),$$

where $\mathcal{K}$ is a Markov transition kernel on X, and R denotes the number of MCMC iterations. The chain is initialized by sampling from an initial density $x[1] \sim \nu(x)$. If the kernel $\mathcal{K}$ is ergodic and admits the target density $\pi(x)$ as its stationary density, then the chain $\{x[k]\}_{k=1}^R$ can be used to approximate expectations in the sense

$$\frac{1}{R}\sum_{k=1}^R \varphi(x[k]) \xrightarrow{a.s.} \mathbb{E}_\pi\left[\varphi(x)\right] \tag{9}$$

for $R \to \infty$, where $\xrightarrow{a.s.}$ labels almost sure convergence, φ is a suitable function on X, and $\mathbb{E}_\pi\left[\varphi(x)\right]$ is the expected value with respect to $\pi(x)$.

An MCMC smoother respects these ideas with the only difference being that the target density π is defined on X^T. However, it is mostly difficult to efficiently sample from the kernel $\mathcal{K}$ when the dimension T is large. This issue is addressed here on the basis of a specific PMCMC method known as PG [1]. The procedure relies on the *conditional* SMC (CSMC) update, which is nearly the same as the original SMC method described above, except one particle trajectory $x'_{1:T}$ is specified in advance. We refer to $x'_{1:T}$ as the reference trajectory. The consequence of this change is that one samples from (5) and (6) only for $i = 1, \ldots, N-1$, while the remaining ancestor index and particle are set as $a_t^N \coloneqq N$ and $x_t^N \coloneqq x'_t$, respectively. The rest of the operations remain unchanged. After finishing the run of the CSMC method, a new reference trajectory is received by sampling k with $\mathbb{P}(k = i) = w_T^i$ and selecting $x_{1:T}^k$ from $\{x_{1:T}^i\}_{i=1}^N$. The fact that the technique requires a state trajectory as the input and returns another state trajectory as the output defines a Markov kernel on X^T, which is called the PG kernel [5]. It is shown in [1] that the kernel is ergodic and admits $\pi_T(x_{1:T})$ as the stationary density.

E. Particle Gibbs with Ancestor Sampling Kernel

The mixing properties of the basic PG kernel may be poor when the number of particles N is low or the dimension T is high. The reason consists in that the (C)SMC algorithm is quite inefficient in updating the past values of the particle trajectories (7). In fact, for some time points not too far from the current time t, the number of identical particle trajectories is often very close or equal to N. Such a loss in diversity is commonly known as particle path degeneracy [15]. The consequence of the phenomenon is that the consecutive trajectories generated by the PG kernel will be highly correlated, and the ability of the PMCMC smoother to explore the space of trajectories X^T will thus become unsatisfactory.

Algorithm 1 PGAS Kernel [5]

Inputs: $x'_{1:T} = x_{1:T}[k-1]$.
Outputs: $x_{1:T}[k]$ and $\{x^i_{1:T}, w^i_T\}_{i=1}^N$.

A. **Initial step:** $(t = 1)$

1. Sample $x^i_1 \sim q_1(\cdot)$ for $i = 1, \ldots, N-1$ and set $x^N_1 := x'_1$.
2. Compute $w^i_1 \propto W_1(x^i_1)$ for $i = 1, \ldots, N$.

B. **Recursive step:** $(t = 2, \ldots, T)$

1. Sample a^i_t with $\mathbb{P}(a^i_t = j) = w^j_{t-1}$ for $i = 1, \ldots, N-1$.
2. Sample $x^i_t \sim q_t(\cdot|x^{a^i_t}_{1:t-1})$ for $i = 1, \ldots, N-1$.
3. Sample a^N_t using (10) and set $x^N_t = x'_t$.
4. Set $x^i_{1:t} := \{x^i_t, x^{a^i_t}_{1:t-1}\}$ for $i = 1, \ldots, N$.
5. Compute $w^i_t \propto W_t(x^i_{1:t})$ according to (8), for $i = 1, \ldots, N$.

C. **Final step:**

1. Sample k with $\mathbb{P}(k = i) = w^i_T$ and set $x_{1:T}[k] = x^k_{1:T}$.

To improve the mixing properties, let us adopt the method referred to as PGAS [5]. The procedure enhances the original PG kernel by introducing an additional sampling step that generates new values of the ancestor indices a^N_t, instead of just setting them as $a^N_t := N$. The Nth trajectory is constructed, for all $t \geq 2$, by concatenating one of the historical trajectories with a future part of the reference trajectory according to

$$x^N_{1:T} := \{x^{a^N_t}_{1:t-1}, x'_{t:T}\},$$

where the connection between the two partial paths is determined by the ancestor index a^N_t. After the complete pass through the data, the Nth trajectory no longer coincides with the reference trajectory, as in the case of the fundamental PG kernel, but rather becomes fragmented into pieces. The consecutive sampled trajectories are then significantly less correlated, thus providing a substantial improvement of the mixing properties. The ancestor a^N_t is sampled from

$$\mathbb{P}(a^N_t = i) = w^i_{t-1|T}, \qquad i = 1, \ldots, N, \tag{10}$$

where

$$w^i_{t-1|T} \propto w^i_{t-1} \frac{\gamma_T(\{x^i_{1:t-1}, x'_{t:T}\})}{\gamma_{t-1}(x^i_{1:t-1})} \tag{11}$$

is the probability of connecting the ith historical trajectory with the future one, see [5] for the derivation of (11) and the proof of ergodicity. The method is recalled in Algorithm 1. Note the CSMC procedure is represented by steps A and B, and the basic PG kernel is obtained by setting $a^N_t := N$ in step B3.

III. SMOOTHER DESIGN

A. Design Objectives

A possible approach to design the algorithm consists in directly targeting the join smoothing density (2). This would require us to simply define $\gamma_t(x_{1:t}) := p(z_{1:t}, c_{1:t}, y_{1:t})$ and $Z_t := p(y_{1:t})$ in the above framework. The CSMC procedure in Algorithm 1 would then be represented by the conditional PF [1] operating with the composite state variable $x_t := (c_t, z_t)$. However, it was noticed in [16] that the PFs sampling this joint state may suffer from the degeneracy of the mode variables when the mixing properties of the transition kernel (1a) are poor. To suppress this problem, an RBPF that marginalizes out the mode variable was proposed in [14]. The development in the present paper is based on introducing a conditional version of this RBPF.

Hence, a better solution is to exploit the tractable substructure of the model and thus factorize (2) as

$$p(c_{1:T}, z_{1:T}|y_{1:T}) = p(c_{1:T}|z_{1:T}, y_{1:T}) p(z_{1:T}|y_{1:T}).$$

The main goal is to design a Rao-Blackwellized PG (RBPG) kernel for approximating the second factor. Then, the sampled trajectories are used in the first factor to analytically compute a finite state-space smoother.

B. Conditional Rao-Blackwellized Particle Filter

To show the derivation of the conditional RBPF (CRBPF), let us factorize the extended target density $p(c_t, z_{1:t}|y_{1:t})$ as

$$p(c_t, z_{1:t}|y_{1:t}) = p(c_t|z_{1:t}, y_{1:t}) p(z_{1:t}|y_{1:t}). \tag{12}$$

The first factor embodies the posterior distribution of the mode variable given the sequence $z_{1:t}$, which is computable analytically through the standard filtering recursion based on the forward prediction

$$p(c_t|z_{1:t-1}, y_{1:t-1}) = \sum_{c_{t-1} \in \mathsf{C}} p(c_t|c_{t-1}) p(c_{t-1}|z_{1:t-1}, y_{1:t-1}) \tag{13}$$

and the forward update

$$p(c_t|z_{1:t}, y_{1:t}) \propto p(y_t, z_t|c_t, z_{t-1}) p(c_t|z_{1:t-1}, y_{1:t-1}), \tag{14}$$

forming together a finite state-space filter.

As the second factor is supposed to describe the analytically intractable part of the model, we resort to the above framework to find the approximation. Thus, the CSMC method now targets $p(z_{1:t}|y_{1:t})$, which requires us to define $\gamma_t(x_{1:t}) := p(z_{1:t}, y_{1:t})$ while Z_t remains the same. The CSMC procedure in Algorithm 1 therefore operates with the marginal state variable $x_t := z_t$. In consequence of these changes, the weight function (8) becomes

$$W_t(z_{1:t}) = \frac{p(y_t, z_t|z_{1:t-1}, y_{1:t-1})}{q_t(z_t|z_{1:t-1})}, \tag{15}$$

where the marginal density in the numerator is

$$p(y_t, z_t|z_{1:t-1}, y_{1:t-1}) = \sum_{c_t \in \mathsf{C}} p(y_t, z_t|c_t, z_{t-1}) p(c_t|z_{1:t-1}, y_{1:t-1}). \tag{16}$$

To compute the weights (15) for all the trajectories $\{z^i_{1:t}\}_{i=1}^N$, the recursion given by (13) and (14) needs to be computed for each $i = 1, \ldots, N$; this is crucial to note as some of the steps in Algorithm 1 are computed just for $N-1$ particles. Importantly, too, the posterior distributions (14) extend the original particle system, yielding $\{z^i_{1:t}, p(c_t|z^i_{1:t}, y_{1:t}), w^i_t\}_{i=1}^N$. Construction of the bootstrap proposal density for $q_t(z_t|z_{1:t-1})$ is outlined in [14].

Algorithm 2 Finite State-Space BIF

Inputs: $z'_{1:T}$ and $\{\xi_t(c_t)\}_{t=1}^T$.
Outputs: $\{\tilde{p}(c_t|y_{t:T}, z'_{t:T})\}_{t=1}^T$.
A. **Initial step:** $(t = T)$
1. Compute $\tilde{p}(c_T|y_T, z'_T) \propto p(y_T|c_T, z'_T)\xi_T(c_T)$.
B. **Recursive step:** $(t = T-1, \ldots, 1)$
1. Compute $\tilde{p}(c_t|y_{t+1:T}, z'_{t:T})$ using (22), $\xi_t(c_t)$, and $\xi_{t+1}(c_{t+1})$.
2. Compute $\tilde{p}(c_t|y_{t:T}, z'_{t:T})$ using (23) and $\tilde{p}(c_t|y_{t+1:T}, z'_{t:T})$.

C. Ancestor Sampling Weights

A possible way to acquire an RBPG kernel would be to use the CRBPF in the basic PG kernel. While this is easy to realize, the mixing properties of the resulting sampler may be poor, as discussed above. Hence, let us derive the ancestor sampling to improve upon this issue. Considering we have $\gamma_T(x_{1:T}) \coloneqq p(z_{1:T}, y_{1:T})$, the weight (11) transforms into

$$w^i_{t-1|T} \propto w^i_{t-1} p(y_{t:T}, z'_{t:T}|z^i_{1:t-1}, y_{1:t-1}). \tag{17}$$

The predictive densities in (17) can be computed by running one finite state-space filter for each of the historical state trajectories $z^i_{1:t-1}$ over the time range from $t-1$ to T. However, if such a straightforward implementation were used, the total cost of computing the weights (17) would amount to $\mathcal{O}(TK^2N)$ operations per time step t. The issue can be circumvented as demonstrated through the design of the Rao-Blackwellized particle smoothers (RBPSs) presented within [9], [17], which consists in rewriting the predictive density as

$$p(y_{t:T}, z'_{t:T}|z^i_{1:t-1}, y_{1:t-1}) = \sum_{c_{t-1}\in\mathsf{C}} p(y_{t:T}, z'_{t:T}|z^i_{t-1}, c_{t-1}) p(c_{t-1}|z^i_{1:t-1}, y_{1:t-1}). \tag{18}$$

The summand in (18) is similar to the two-filter smoothing formula [18] that is based on running one filter forward and the other backward in time. In our situation, the forward filter has already been recalled in (13) and (14). The backward filter is designed below.

D. Finite State-Space Backward Information Filter

The BIF [13] facilitates the computation of the likelihood term in the summand of (18) under a closed-form solution. In the present context, the recursive step of such a filter iterates, for $t = T-1, \ldots, 1$, over the backward prediction

$$p(y_{t+1:T}, z_{t+1:T}|z_t, c_t) = \sum_{c_{t+1}\in\mathsf{C}} p(y_{t+1:T}, z_{t+2:T}|z_{t+1}, c_{t+1}) p(z_{t+1}, c_{t+1}|c_t, z_t) \tag{19}$$

and the backward update

$$p(y_{t:T}, z_{t+1:T}|z_t, c_t) = p(y_t|c_t, z_t) p(y_{t+1:T}, z_{t+1:T}|z_t, c_t). \tag{20}$$

The initial step computes only the observation density (1c) at $t = T$. A similar recursive form has recently been used to develop an RBPS for mixed linear/nonlinear SSMs in [19].

The difficulty encountered with (19) and (20) lies in that neither of these is a probability density in the arguments z_t and c_t. Therefore, integrating with respect to these variables might lead to results which are not finite. To overcome this issue, Briers et al. [20] proposed to design a sequence of *artificial* probability distributions to change the scale of such problematic measures. Here, this approach is applied only to the mode variable c_t as the state variable z_t is fixed at this stage of the design. The sought recursion is introduced below.

Proposition 1 (Rescaled backward recursion): Let $\xi_t(c_t)$ denote the artificial (prior) distribution related to an *auxiliary* (posterior) distribution $\tilde{p}(c_t|y_{t:T}, z_{t:T})$ according to

$$\tilde{p}(c_t|y_{t:T}, z_{t:T}) \propto p(y_{t:T}, z_{t+1:T}|z_t, c_t)\xi_t(c_t). \tag{21}$$

Then, the rescaled version of the backward prediction (19) is

$$\tilde{p}(c_t|y_{t+1:T}, z_{t:T}) \propto \sum_{c_{t+1}\in\mathsf{C}} \tilde{p}(c_{t+1}|y_{t+1:T}, z_{t+1:T}) \frac{p(z_{t+1}, c_{t+1}|c_t, z_t)\xi_t(c_t)}{\xi_{t+1}(c_{t+1})}, \tag{22}$$

and, similarly, the backward update (20) becomes

$$\tilde{p}(c_t|y_{t:T}, z_{t:T}) \propto p(y_t|c_t, z_t)\tilde{p}(c_t|y_{t+1:T}, z_{t:T}). \tag{23}$$

Proof: The result follows immediately from multiplying both sides of (19) and (20) by $\xi_t(c_t)$ and noticing that a definition analogous to (21) holds also for $\tilde{p}(c_t|y_{t+1:T}, z_{t:T})$. ∎

To compute the predictive density by means of this reformulated backward recursion, we first substitute from (21) into (19) to receive

$$p(y_{t:T}, z'_{t:T}|z^i_{t-1}, c_{t-1}) \propto \sum_{c_t\in\mathsf{C}} \frac{\tilde{p}(c_t|y_{t:T}, z'_{t:T})}{\xi_t(c_t)} p(z'_t, c_t|c_{t-1}, z^i_{t-1}), \tag{24}$$

and, subsequently, we plug this result into (18). Note that $\tilde{p}(c_t|y_{t:T}, z'_{t:T})$ needs to be computed only for the reference trajectory as it does not contain any part of the historical particle trajectories. Thus, the distributions $\tilde{p}(c_t|y_{t:T}, z'_{t:T})$ can be precomputed in a single backward pass through the data. It is also important to mention the total cost of computing (17) becomes $\mathcal{O}(K^2N)$ operations in a single time step t. The finite state-space BIF is summarized in Algorithm 2.

The artificial distribution is introduced into the backward recursion such that its choice can be made arbitrarily. Multiple recommendations regarding how to define $\xi_t(c_t)$ are contained in [20]; here, the marginal and independent prior is chosen, namely, $\xi_t(c_t) \coloneqq p(c_t) = p(c_t|z_t)$, where

$$p(c_t) = \sum_{c_{t-1}\in\mathsf{C}} p(c_t|c_{t-1}) p(c_{t-1}). \tag{25}$$

Note this choice simplifies the ratio in (22) to the product $p(z_{t+1}|c_t, z_t)p(c_t|c_{t+1})$, where $p(c_t|c_{t+1})$ is the backward transition kernel of the mode variable. Other choices of $\xi_t(c_t)$ are discussed later in the text.

The proposed RBPGAS kernel is summarized in Algorithm 3, where the convention $z_{1:0} \coloneqq \varnothing$ and $y_{1:0} \coloneqq \varnothing$ holds in step A4. Although not explicitly expressed in Algorithm 3, the posterior distributions $p(c_{t-1}|z^i_{1:t-1}, y_{1:t-1})$ are resampled, together with the particle trajectories $z^i_{1:t-1}$, by applying the ancestor indices a^i_t before their use in (13).

Algorithm 3 RBPGAS Kernel for JMNMs

Inputs: $z'_{1:T} = z_{1:T}[k-1]$.
Outputs: $z_{1:T}[k]$ and $\{z^i_{1:T}, \{p(c_t|z^i_{1:t}, y_{1:t})\}_{t=1}^T, w^i_T\}_{i=1}^N$.

A. **Initial step:** $(t = 1)$

1. Compute the sequence $\{\xi_t(c_t)\}_{t=1}^T$.
2. Use $z'_{1:T}$ and $\{\xi_t(c_t)\}_{t=1}^T$ as the input for Algorithm 2 to produce $\{\tilde{p}(c_t|y_{t:T}, z'_{t:T})\}_{t=1}^T$.
3. Sample $z^i_1 \sim q_1(\cdot)$ for $i = 1, \ldots, N-1$ and set $z^N_1 \coloneqq z'_1$.
4. Compute $p(c_1|z^i_1, y_1)$ and $p(y_1, z^i_1)$ according to (14) and (16), respectively, for $i = 1, \ldots, N$.
5. Compute $w^i_1 \propto W_1(z^i_1)$ for $i = 1, \ldots, N$.

B. **Recursive step:** $(t = 2, \ldots, T)$

1. Sample a^i_t with $\mathbb{P}(a^i_t = j) = w^j_{t-1}$ for $i = 1, \ldots, N-1$.
2. Sample $z^i_t \sim q_t(\cdot|z^{a^i_t}_{1:t-1})$ for $i = 1, \ldots, N-1$.
3. Compute $w^i_{t-1|T}$ according to (17-18) and (24), using $\xi_t(c_t)$ and $\tilde{p}(c_t|y_{t:T}, z'_{t:T})$, for $i = 1, \ldots, N$.
4. Sample a^N_t with $\mathbb{P}(a^N_t = i) = w^i_{t-1|T}$ and set $z^N_t \coloneqq z'_t$.
5. Set $z^i_{1:t} \coloneqq \{z^i_t, z^{a^i_t}_{1:t-1}\}$ for $i = 1, \ldots, N$.
6. Compute $p(c_t|z^i_{1:t}, y_{1:t})$ and $p(y_t, z^i_t|z^{a^i_t}_{1:t-1}, y_{1:t-1})$ according to (13-14) and (16), respectively, for $i = 1, \ldots, N$.
7. Compute $w^i_t \propto W_t(z^i_{1:t})$ according to (15), for $i = 1, \ldots, N$.

C. **Final step:**

1. Sample k with $\mathbb{P}(k = i) = w^i_T$ and set $z_{1:T}[k] \coloneqq z^k_{1:T}$.

E. Finite State-Space Smoother

To obtain the smoothed estimates of the mode trajectory, we resort to a finite state-space forward-backward smoother conditioned on the state trajectory $z_{1:T}$. The recursive step computes, for $t = T-1, \ldots, 1$, the marginal smoothing distribution according to

$$p(c_t|z_{1:T}, y_{1:T}) = \sum_{c_{t+1} \in \mathsf{C}} \frac{p(c_{t+1}|c_t)p(c_t|z_{1:t}, y_{1:t})}{p(c_{t+1}|z_{1:t}, y_{1:t})} p(c_{t+1}|z_{1:T}, y_{1:T}), \quad (26)$$

where $p(c_t|z_{1:t}, y_{1:t})$ is produced by the forward recursion formed by (13) and (14). For the initial step, the filtering and marginal smoothing distributions are identical. After computing (26) for each of the trajectories $\{z_{1:T}[k]\}_{k=1}^R$ and averaging the results with (9), the maximum *a posteriori* sequence is taken to represent the smoothed estimate.

An alternative approach is to use the auxiliary distributions $\tilde{p}(c_t|y_{t:T}, z'_{t:T})$ in the two-filter smoother [7].

IV. NUMERICAL ILLUSTRATION

This section demonstrates the performance of the proposed RBPGAS kernel (Algorithm 3) in comparison to the PG [1], PGAS [5], RBPG (Algorithm 3 with setting $a^N_t \coloneqq N$ in step B4), and RBPGASnr (Algorithm 3 with the non-rescaled recursion) kernels. Let us consider the nonlinear benchmark model given by

$$z_t = 0.5 z_{t-1} + 25 \frac{z_{t-1}}{1 + z^2_{t-1}} + 8 \cos(1.2t) + v_t, \quad (27a)$$

$$y_t = 0.05 z^2_t + w_t, \quad (27b)$$

where, for $c_t \in \mathsf{C} \coloneqq \{1, 2\}$, $w_t \sim \mathcal{N}(\mu_{c_t}, \sigma^2_{c_t})$ denotes a mode-dependent Gaussian noise variable with the mean μ_{c_t} and variance $\sigma^2_{c_t}$. Furthermore, $v_t \sim \mathcal{N}(0, 1)$ is an independent and identically distributed Gaussian noise variable with zero mean and unit variance. The kernel (1a) is parameterized by the transition probability matrix Π according to $p(c_t = j|c_{t-1} = i) \coloneqq \Pi_{ij}$ with $i, j \in \mathsf{C}$. The diagonal entries of this matrix are set as $\Pi_{11} = 0.6$ and $\Pi_{22} = 0.8$. The mode-dependent means and variances are defined by $\mu_1 = 0$, $\mu_2 = 7$ and $\sigma^2_1 = 4$, $\sigma^2_2 = 1$. The state prior density is mode-independent and Gaussian, $\mu(z_1|c_1) \coloneqq \mathcal{N}(z_1; 0, 1)$. Further, the prior distribution of the mode variable $p(c_1)$ is parameterized by the vector λ with the relation $p(c_1 = i) \coloneqq \lambda_i$, where the first entry is $\lambda_1 = 0.5$; the artificial distribution corresponds to (25). All the algorithms use the associated bootstrap proposal density.

The experiment assesses the performance of the compared methods when the number of particles changes according to $N = 2^n$ for $n = 1, \ldots, 9$. For all these values, forty independent runs of the model (27) were performed, each producing the measurement sequence of the length $T = 100$. The algorithms subjected to comparison were tested with the number of iterations $R = 500$. To evaluate resulting estimates, the proposed RBPGAS method with $N = 1024$ particles (a higher number did not lead a significant improvement) was used to compute an "exact" state and mode trajectories for each of the measurement sequences.

As is obvious from Fig. 1, the RBPGAS method achieves the best accuracy with the number of particles $N = 4$, increasing the value does not provide any noticeable improvement. The PGAS procedure approaches the performance of the RBPGAS algorithm as the number of particles increases, with practically no difference in the RMSE and NNZE indicators for $N \geq 32$. The distinction between the methods is most significant at $N = 2$ particles, where the effect of Rao-Blackwellization, brought by the RBPGAS procedure, is most obvious. The PG and RBPG algorithms attain the precision of the RBPGAS method for a much higher N, which is caused by the absence of the ancestor sampling in these methods. This also implies that the PG and RBPG algorithms require substantially more computational resources. Nevertheless, even in this case, it can be seen that the Rao-Blackwellization may substantially increase the accuracy.

Fig. 2 provides a closer look at the situation where the compared algorithms are applied with $N = 2$ particles. We can see that the RMSE of the PGAS and RBPGAS methods is competitive for approximately the first 10^{-1} seconds, with the RBPGAS algorithm starting to be computationally more efficient (in the average behavior) after this time. This is obvious from the median and interquartile range, which decrease more quickly for the RBPGAS procedure. The right part of Fig. 2 reveals that the RBPGAS method achieves lower values of the NNZE in a shorter computational time. For example, we can see that the value 10 is there reached after approximately $2 \cdot 10^{-1}$ seconds with the PGAS method, while the same value is attained after approximately $7 \cdot 10^{-2}$ seconds with the RBPGAS algorithm. It is therefore obvious that the RBPGAS procedure is markedly quicker in approaching the ergodic regime. However, the PG and RBPG kernels deliver a very slow convergence, being still very far from the ergodic regime. The reason is, again, the lack of the

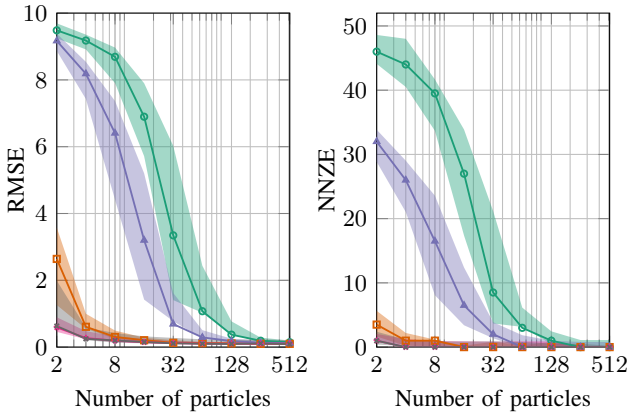

Fig. 1. Left: the RMSE between the exact and estimated state trajectories vs. the number of particles. Right: the number of non-zero elements (NNZE) in the error sequence between the exact and estimated mode trajectories vs. the number of particles. The solid line represents the median, and the shaded area is the interquartile range; both are computed over forty independent runs. The compared algorithms are PG (—○—), PGAS (—□—), RBPG (—△—), RBPGAS (—☆—), and RBPGASnr (—×—).

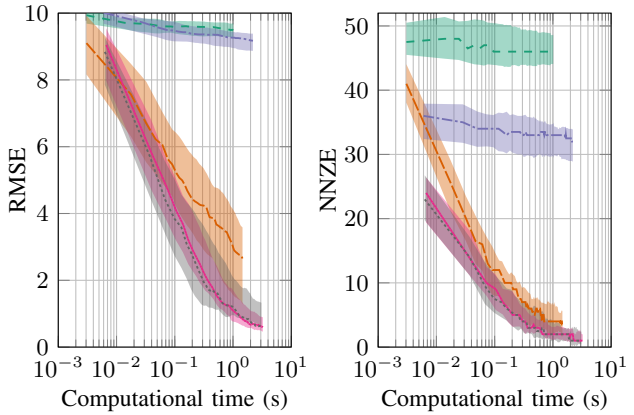

Fig. 2. Left: the RMSE between the exact and estimated state trajectories vs. the computational time (in seconds). Right: the number of non-zero elements (NNZE) in the error sequence between the exact and estimated mode trajectories vs. the computational time. The solid line shows the median, and the shaded area is the interquartile range; both are computed over forty independent runs. The compared methods are PG (- - -), PGAS (———), RBPG (–·–·–), RBPGAS (———), and RBPGASnr (·····).

ancestor sampling. The RBPGASnr method has a higher variance than the RBPGAS procedure, proving the importance of involving the artificial distribution $\xi_t(c_t)$ into the design.

V. DISCUSSION

The uniform prior λ and stationary distribution calculated as the left eigenvector of the matrix Π were used to represent the artificial distribution $\xi_t(c_t)$. Both these options provided results similar to the situation considering (25), with only a minor difference in the variance of the smoothed estimates.

If the transition kernel (1a) has poor mixing properties, the proposed RBPGAS method offers a higher robustness compared to the PGAS algorithm. The trade-off between the estimation accuracy and computational time then starts to be more pronounced, with the proposed RBPGAS method still being the more successful one. The reason lies, as discussed previously, in that the PGAS procedure suffers from the degeneracy around the mode changes [16]. As shown in the experiments, the effectiveness is mainly achieved in terms of a shorter time required to reach the ergodic regime.

The future work will be devoted to the application of the proposed method to the maximum likelihood parameter estimation in JMNMs.

REFERENCES

[1] C. Andrieu, A. Doucet, and R. Holenstein, "Particle Markov chain Monte Carlo methods," *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, vol. 72, no. 3, pp. 269–342, 2010.

[2] O. Cappé, E. Moulines, and T. Ryden, *Inference in Hidden Markov Models*. Springer, 2005.

[3] A. Doucet and A. M. Johansen, "A tutorial on particle filtering and smoothing: Fifteen years later," in *The Oxford Handbook of Nonlinear Filtering*, D. Crisan and B. Rozovsky, Eds. Oxford University Press, 2009.

[4] C. Andrieu, N. de Freitas, A. Doucet, and M. I. Jordan, "An introduction to MCMC for machine learning," *Machine Learning*, vol. 50, no. 1, pp. 5–43, 2003.

[5] F. Lindsten, M. I. Jordan, and T. B. Schön, "Particle Gibbs with ancestor sampling," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 2145–2184, 2014.

[6] S. J. Godsill, A. Doucet, and M. West, "Monte Carlo smoothing for nonlinear time series," *Journal of the american statistical association*, vol. 99, no. 465, pp. 156–168, 2004.

[7] M. Briers, A. Doucet, and S. Maskell, "Smoothing algorithms for state–space models," Cambridge University, Tech. Rep. CUED/F-INFENG/TR.498, 2004.

[8] F. Lindsten and T. B. Schön, "Backward simulation methods for Monte Carlo statistical inference," *Foundations and Trends in Machine Learning*, vol. 6, no. 1, pp. 1–143, 2013.

[9] N. Whiteley, C. Andrieu, and A. Doucet, "Efficient Bayesian inference for switching state-space models using discrete particle Markov chain Monte Carlo methods," *arXiv preprint arXiv:1011.2437*, 2010.

[10] A. Svensson, T. B. Schön, and F. Lindsten, "Identification of jump Markov linear models using particle filters," in *Decision and Control (CDC), 2014 IEEE 53rd Annual Conference on*. IEEE, 2014, pp. 6504–6509.

[11] P. Fearnhead and P. Clifford, "On-line inference for hidden Markov models via particle filters," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 65, no. 4, pp. 887–899, 2003.

[12] A. Doucet, N. J. Gordon, and V. Krishnamurthy, "Particle filters for state estimation of jump Markov linear systems," *IEEE Transactions on signal processing*, vol. 49, no. 3, pp. 613–624, 2001.

[13] D. Q. Mayne, "A solution of the smoothing problem for linear dynamic systems," *Automatica*, vol. 4, no. 2, pp. 73–92, 1966.

[14] E. Özkan, F. Lindsten, C. Fritsche, and F. Gustafsson, "Recursive maximum likelihood identification of jump Markov nonlinear systems," *IEEE Transactions on Signal Processing*, vol. 63, no. 3, pp. 754–765, 2015.

[15] P. E. Jacob, L. M. Murray, and S. Rubenthaler, "Path storage in the particle filter," *Statistics and Computing*, vol. 25, no. 2, pp. 487–496, 2015.

[16] H. Driessen and Y. Boers, "Efficient particle filter for jump Markov nonlinear systems," *IEE Proceedings - Radar, Sonar and Navigation*, vol. 152, no. 5, pp. 323–326, 2005.

[17] S. Särkkä, P. Bunch, and S. J. Godsill, "A backward-simulation based Rao-Blackwellized particle smoother for conditionally linear Gaussian models," *IFAC Proceedings Volumes*, vol. 45, no. 16, pp. 506–511, 2012.

[18] Y. Bresler, "Two-filter formulae for discrete-time non-linear Bayesian smoothing," *International Journal of Control*, vol. 43, no. 2, pp. 629–641, 1986.

[19] F. Lindsten, P. Bunch, S. Särkkä, T. B. Schön, and S. J. Godsill, "Rao-Blackwellized particle smoothers for conditionally linear Gaussian models," *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 2, pp. 353–365, 2016.

[20] M. Briers, A. Doucet, and S. Maskell, "Smoothing algorithms for state–space models," *Annals of the Institute of Statistical Mathematics*, vol. 62, no. 1, pp. 61–89, 2010.