The Annals of Statistics
2020, Vol. 48, No. 5, 2930–2952
https://doi.org/10.1214/19-AOS1916
© Institute of Mathematical Statistics, 2020

A FRAMEWORK FOR ADAPTIVE MCMC TARGETING MULTIMODAL DISTRIBUTIONS

BY EMILIA POMPE[1,*], CHRIS HOLMES[1,†] AND KRZYSZTOF ŁATUSZYŃSKI[2]

[1]*Department of Statistics, University of Oxford,* [*]*emilia.pompe@stats.ox.ac.uk;* [†]*cholmes@stats.ox.ac.uk*
[2]*Department of Statistics, University of Warwick,* *K.G.Latuszynski@warwick.ac.uk*

We propose a new Monte Carlo method for sampling from multimodal distributions. The idea of this technique is based on splitting the task into two: finding the modes of a target distribution π and sampling, given the knowledge of the locations of the modes. The sampling algorithm relies on steps of two types: local ones, preserving the mode; and jumps to regions associated with different modes. Besides, the method learns the optimal parameters of the algorithm, while it runs, without requiring user intervention. Our technique should be considered as a flexible framework, in which the design of moves can follow various strategies known from the broad MCMC literature.

In order to design an adaptive scheme that facilitates both local and jump moves, we introduce an auxiliary variable representing each mode, and we define a new target distribution $\tilde{\pi}$ on an augmented state space $\mathcal{X} \times \mathcal{I}$, where $\mathcal{X}$ is the original state space of π and $\mathcal{I}$ is the set of the modes. As the algorithm runs and updates its parameters, the target distribution $\tilde{\pi}$ also keeps being modified. This motivates a new class of algorithms, Auxiliary Variable Adaptive MCMC. We prove general ergodic results for the whole class before specialising to the case of our algorithm.

1. Introduction.

Poor mixing of standard Markov chain Monte Carlo (MCMC) methods on multimodal target distributions with isolated modes is a well-described problem in statistics. Due to their dynamics these algorithms struggle with crossing low-probability barriers separating the modes and, thus, take a long time before moving from one mode to another, even in low dimensions. Sequential Monte Carlo (SMC) has often empirically proven to outperform MCMC on this task; its robust behaviour, however, relies strongly on the good between-mode mixing of the Markov kernel used within the SMC algorithm (see [31]). Therefore, constructing an MCMC algorithm which enables fast exploration of the state space for complicated target functions is of great interest, especially as multimodal distributions are common in applications. The examples include, but are not limited to, problems in genetics [12, 24], astrophysics [14, 15, 39] and sensor network localisation [21].

Moreover, multimodality is an inherent issue of Bayesian mixture models (e.g., [23]), where it may be caused by label-switching or, more generally, by model identifiability issues or model misspecification (see [13]).

Designing MCMC algorithms for sampling from a multimodal target distribution π on a state space $\mathcal{X}$ needs to address three fundamental challenges:

(1) Identifying high probability regions where the modes are located.
(2) Moving between the modes by crossing low-probability barriers.
(3) Sampling efficiently within the modes by accounting for inhomogeneity between them and their local geometry.

Received January 2019; revised September 2019.
MSC2020 subject classifications. Primary 60J05, 65C05; secondary 62F15.
Key words and phrases. Multimodal distribution, adaptive MCMC, ergodicity.

Existing MCMC methodology for multimodal distributions usually identifies these challenges separately and a systematic way of addressing (1)–(3) is not available. In this paper we introduce a unifying framework for responding to challenges (1)–(3) simultaneously via a novel design of Auxiliary Variable Adaptive MCMC. The framework allows us to split the sampling task into mode finding, between-region jump moves and local moves. In addition, it incorporates parameter adaptations for optimisation of the local and jump kernels and identification of local regions.

1.1. *Other approaches*. Numerous MCMC methods have been proposed to address the issue of multimodality. The most popular approach is based on tempering. The idea behind this type of methods relies on an observation that raising a multimodal distribution π to the power $\beta \in (0, 1)$ makes the modes "flatter," and, as a result, it is more likely to accept moves to the low probability regions. Hence, it is easier to explore the state space and find the regions where the modes of π are located, addressing challenge (1) above, and also to move between these regions, addressing challenge (2). The examples of such methods, which incorporate π^β by augmenting the state space, are parallel tempering proposed by [18] and its adaptive version [27], simulated tempering [26], tempered transitions [29] and the equi-energy sampler [24]. Despite their popularity, tempering-based approaches, as noticed by [45], tend to mix between modes exponentially slowly in dimension if the modes have different local covariance structures. Addressing this issue is an area of active research [40].

Another strand of research is optimisation-based methods which address challenge (1) by running preliminary optimisation searches in order to identify local maxima of the target distribution. They use this information in their between-mode proposal design to overcome challenge (2). A method called Smart Darting Monte Carlo, introduced in [2], relies on moves of two types: jumps between the modes, allowed only in nonoverlapping ϵ-spheres around the local maxima identified earlier; and local moves (Random Walk Metropolis steps). This technique was generalised in [38] by allowing the jumping regions to overlap and have an arbitrary volume and shape. The authors of [1] went one step further by introducing updates of the jumping regions and parameters of the proposal distribution at regeneration times, hence, the name of their method—Regeneration Darting Monte Carlo (RDMC). This includes a possibility of adding new locations of the modes at regeneration times if they are detected by optimisation searches running on separate cores. Another optimisation-based method, Wormhole Hamiltonian Monte Carlo, was introduced by [25] as an extension of Riemanian Manifold HMC. As before, updates of the parameters of this algorithm are allowed at regeneration times.

We will see later that the algorithm we propose also falls into the category of optimisation-based methods.

The Wang–Landau algorithm [43, 44] or its adaptive version proposed by [10] belong to the exploratory strategies that aim to push the algorithm away from well-known regions and visit new ones, hence, addressing challenge (1). The multidomain sampling technique, proposed in [46], combines the idea of the Wang–Landau algorithm with the optimisation-based approach. This algorithm relies on partitioning the state space into domains of attraction of the modes. Local moves are Random Walk Metropolis steps proposed from a distribution depending on the domain of attraction of the current state. Jumps between the modes follow the independence sampler scheme, where the new states are proposed from a mixture of Gaussian distributions approximating π.

Other common approaches include the localised normal-symmetric Random Walk Metropolis algorithm [5], MultiNest algorithms based on nested sampling [14, 15] and Metropolis–Hastings algorithms with a special design of the proposal distribution accounting for the necessity of moving between the modes [39, 41].

1.2. *Contribution.* The importance of addressing challenge (3) stems from the fact that, when modes have distinct shapes, different local proposal distributions will work well in regions associated with different modes. However, the majority of the methods described above only employ a single transition kernel, regardless of the region.

In applied problems optimal parameters of the MCMC kernels are unknown; therefore, recent approaches involve tuning them while the algorithm runs. In case of unimodal target distributions, Adaptive MCMC techniques prove to be useful [5, 19, 36]. The parameters of the involved transition kernels can be learned on the fly as the simulation progresses, based on the samples observed so far. The adaptive algorithms remain ergodic under suitable regularity conditions [3, 7, 11, 16, 35].

In case of multimodal distributions, an analogous idea would be to apply these Adaptive MCMC methods separately to regions associated with different modes, to improve the within-mode mixing. Note that in order to sample from different proposal distributions in regions associated with different modes, one needs to control at each step of the algorithm to which region the current state belongs. Besides, adapting parameters of the local proposal distributions on the fly must be based on samples that actually come from the corresponding region.

The multidomain sampler [46] discussed earlier has a mechanism of assigning samples to regions. However, in their setting keeping track of the regions requires running a gradient ascent procedure at each MCMC step which imposes a high computational burden on the whole algorithm. Other optimisation-based approaches known in the literature (e.g., [38] and [1]) tend to ignore the necessity of assigning samples to regions and the possibility of moving between the modes via local steps.

Another issue is that adaptive optimisation-based methods presented above, such as those of [1] and [25], allow for adaptations only at regeneration times. In high dimensions regenerations happen rarely which, in practice, makes the adaptive scheme prohibitively inefficient. Besides, identifying regeneration times using the method of [28], as authors of both algorithms propose, requires case-specific calculations which precludes any generic implementation.

We aim to remedy these shortcomings by proposing a framework for designing an adaptive algorithm on an augmented state space $\mathcal{X} \times \mathcal{I}$, where $\mathcal{I} = \{1, \dots, N\}$, and the auxiliary variable i of the resulting sample (x, i) encodes the corresponding region for x. Local MCMC kernels update x only, while jump kernels that move between the modes update x and i simultaneously. Furthermore, the design of the target distribution on the augmented state space prevents the algorithm from moving to a region associated with a different mode via local steps. In the sequel we make specific choices for the adaptive scheme, the local and jump kernels as well as the burn-in routine used for setting up initial values of the parameters of the algorithm. However, the design is modular and different approaches can be incorporated in the framework. Besides, it allows for a multicore implementation of a large part of the algorithm.

This approach motivates introducing a new class of algorithms, Auxiliary Variable Adaptive MCMC, where not only transition kernels are allowed to be modified on the fly but also the augmented target distributions. It turns out that, apart from our method, there is a wide range of algorithms that belong to this class, including adaptive parallel tempering or adaptive versions of pseudo-marginal MCMC. Thus, our general ergodicity results, proved for the whole class under standard regularity conditions, can potentially be useful for analysing other methods.

The remainder of the paper is organised as follows. In Section 2 we present our algorithm, the Jumping Adaptive Multimodal Sampler (JAMS), and discuss its properties. In Section 3 we define the Auxiliary Variable Adaptive MCMC class and establish convergence in distribution and a Weak Law of Large Numbers for this class, under the uniform and the nonuniform scenario. We present theoretical results specialised to the case of our proposed algorithm in Section 4. Ergodicity is derived here from the analogues of the Containment and Diminishing Adaptation conditions introduced in [35], as opposed to identifying regeneration times which allow us to circumvent the issues described above. The proofs of all our theorems along with some additional comments about the theoretical results are gathered in Supplementary Material A [32]. Section 5 demonstrates the performance of our method on two synthetic and one real data example. Additional details of our numerical experiments are available in Supplementary Material B [32]. We conclude with a summary of our results and a discussion in Section 6.

2. Jumping Adaptive Multimodal Sampler (JAMS).

2.1. *Main algorithm.* Let π be the multimodal target distribution of interest defined on $(\mathcal{X}, \mathcal{B}(\mathcal{X}))$. We introduce a collection of target distributions $\{\tilde{\pi}_\gamma\}_{\gamma \in \mathcal{Y}}$ on the augmented state space $\mathcal{X} \times \mathcal{I}$, where $\mathcal{I} := \{1, \dots, N\}$ is a finite set of indices of the modes of π. We defer the discussion about finding the modes to Section 2.2. Here, γ denotes the design parameter of the algorithm that may be adapted on the fly. For a fixed $\gamma \in \mathcal{Y}$, $\tilde{\pi}_\gamma$ is defined as

$$\tilde{\pi}_\gamma(x, i) := \pi(x) \frac{w_{\gamma,i} Q_i(\mu_i, \Sigma_{\gamma,i})(x)}{\sum_{j \in \mathcal{I}} w_{\gamma,j} Q_j(\mu_j, \Sigma_{\gamma,j})(x)}, \tag{2.1}$$

where $Q_i(\mu_i, \Sigma_{\gamma,i})$ is an elliptical distribution (such as the normal or the multivariate t distribution) centred at μ_i with covariance matrix $\Sigma_{\gamma,i}$. We shall think of $\{\mu_i\}_{i \in \mathcal{I}}$ and $\{\Sigma_{\gamma,i}\}_{i \in \mathcal{I}}$ as locations and covariances of the modes of π, respectively. First, notice that constructing a Markov chain targeting $\tilde{\pi}_\gamma$ provides a natural way of identifying the mode at each step by recording the auxiliary variable i. Besides, for each $B \in \mathcal{B}(\mathcal{X})$ and $\gamma \in \mathcal{Y}$ we have $\tilde{\pi}_\gamma(B \times \mathcal{I}) = \pi(B)$. Hence, π is the marginal distribution of $\tilde{\pi}_\gamma$ for each $\gamma \in \mathcal{Y}$; so, sampling from $\tilde{\pi}_\gamma$ can be used to generate samples from π.

The sampling algorithm that we propose is summarised in Algorithm 1. The method relies on MCMC steps of two types, performed with probabilities $1 - \epsilon$ and ϵ, respectively:

- *Local move*: Given the current state of the chain (x, i) and the current parameter γ, a local kernel $\tilde{P}_{\gamma,L,i}$ invariant with respect to $\tilde{\pi}_\gamma$ is used to update x, while i remains fixed; hence, the mode is preserved.
- *Jump move*: Given the current state of the chain (x, i) and the current parameter γ, a new mode k is proposed with probability $a_{\gamma,ik}$. Then, a new point y is proposed using a distribution $R_{\gamma,J,ik}(x, \cdot)$. The new pair is accepted or rejected using the standard Metropolis–Hastings formula such that the jump kernel is invariant with respect to $\tilde{\pi}_\gamma$.

Our choice for the local kernel is Random Walk Metropolis (RWM) with proposal $R_{\gamma,L,i}(x, \cdot)$ that follows either the normal or the t distribution. This allows us to employ well-developed adaptation strategies for RWM and to build on its stability properties to establish ergodicity of JAMS in Section 4. The acceptance probability formula for local moves is given by

$$\alpha_{\gamma,L}\big((x,i) \to (y,i)\big) = \min\left[1, \frac{\pi(y) Q_i(\mu_i, \Sigma_{\gamma,i})(y)}{\pi(x) Q_i(\mu_i, \Sigma_{\gamma,i})(x)} \frac{\sum_{j \in \mathcal{I}} w_{\gamma,j} Q_j(\mu_j, \Sigma_{\gamma,j})(x)}{\sum_{j \in \mathcal{I}} w_{\gamma,j} Q_j(\mu_j, \Sigma_{\gamma,j})(y)}\right]. \tag{2.2}$$

As for the jump moves, we consider two different methods of proposing a new point y associated with mode k. The first one, which we call *independent proposal jumps*, is to draw y from an elliptical distribution centred at μ_k with covariance matrix $\Sigma_{\gamma,k}$, independently from the current point (x,i). Since there is no dependence on x and i, in case of independent proposal jumps the proposal distribution to mode k will be denoted by $R_{\gamma,J,k}(\cdot)$. For independent proposal jumps, the acceptance probability is equal to

$$\alpha_{\gamma,J}\big((x,i)\to(y,k)\big)=\min\left[1,\frac{\tilde{\pi}_\gamma(y,k)}{\tilde{\pi}_\gamma(x,i)}\frac{a_{\gamma,ki}R_{\gamma,J,i}(x)}{a_{\gamma,ik}R_{\gamma,J,k}(y)}\right]. \tag{2.3}$$

Alternatively, given that the current state is (x,i), we can propose a "corresponding" point y in mode k such that the Mahalanobis distance between the point and its corresponding mode is preserved, that is

$$(x-\mu_i)^T\Sigma_{\gamma,i}^{-1}(x-\mu_i)=(y-\mu_k)^T\Sigma_{\gamma,k}^{-1}(y-\mu_k).$$

The required equality is satisfied for

$$y:=\mu_k+\Lambda_{\gamma,k}\Lambda_{\gamma,i}^{-1}(x-\mu_i), \tag{2.4}$$

where

$$\Sigma_{\gamma,i}=\Lambda_{\gamma,i}\Lambda_{\gamma,i}^T \quad\text{and}\quad \Sigma_{\gamma,k}=\Lambda_{\gamma,k}\Lambda_{\gamma,k}^T.$$

Herein, this method will be called *deterministic jumps*. The acceptance probability is then given by

$$\alpha_{\gamma,J}\big((x,i)\to(y,k)\big)=\min\left[1,\frac{\tilde{\pi}_\gamma(y,k)}{\tilde{\pi}_\gamma(x,i)}\frac{a_{\gamma,ki}\sqrt{\det\Sigma_{\gamma,k}}}{a_{\gamma,ik}\sqrt{\det\Sigma_{\gamma,i}}}\right]. \tag{2.5}$$

In both cases the design of the jump moves takes into account the shapes of the two modes involved which helps achieving high acceptance rates and, consequently, improves the between-mode mixing.

As presented in Algorithm 1, the method involves learning the parameters on the fly. We design an adaptation scheme of three lists of parameters: covariance matrices (used both for adapting the target distribution $\tilde{\pi}_\gamma$ and the proposal distributions), weights $w_{\gamma,i}$ and probabilities $a_{\gamma,ik}$ of proposing mode k in a jump from mode i. Hence, formally, $\mathcal{Y}$ refers to the product space of $\Sigma_{\gamma,i}$, $w_{\gamma,i}$ and $a_{\gamma,ik}$ for $i,k\in\{1,\dots,N\}$ restricted by $\sum_{j\in\mathcal{I}}w_{\gamma,j}=1$ and $\sum_{k\in\mathcal{I}}a_{\gamma,ik}=1$ for each $\gamma\in\mathcal{Y}$ and each $i\in\mathcal{I}$. An adaptive scheme for $w_{\gamma,i}$ and $a_{\gamma,ik}$ that follows an intuitive heuristic is discussed briefly in Supplementary Material B [32].

Our method of adapting the covariance matrices $\Sigma_{\gamma,i}$ is presented in Algorithm 2. For every $i\in\mathcal{I}$, the matrix $\Sigma_{\gamma,i}$ is based on the empirical covariance matrix of the samples from the region associated with mode i obtained so far. This is possible in our framework by keeping track of the auxiliary variable i. Updates are performed every certain number of iterations (denoted by AC_2 in Algorithm 2). This method follows the classical Adaptive Metropolis methodology (cf. [20, 36]) applied separately to the covariance structure associated with each mode. For the local proposal distributions the covariance matrices are additionally scaled by the factor $2.38^2/d$ which is commonly used as optimal for Adaptive Metropolis algorithms [33, 34]. Since representing a covariance matrix in high dimensions reliably typically requires a large number of samples, we do not apply this method straightaway. Instead, we perform adaptive scaling, aiming to achieve the optimal acceptance rate (typically fixed at 0.234; see [33, 34]) for local moves, until the number of samples observed in a given mode exceeds a prespecified constant (denoted by AC_1 in Algorithm 2).

It is worth outlining that this special construction of the target distribution $\tilde{\pi}_\gamma$ makes it unlikely for the algorithm to escape via local steps from the mode it is assigned to and settle in another one. Indeed, if a proposed point y is very distant from the current mode μ_i and close to another mode μ_k, the acceptance probability becomes very small due to the expression $Q_i(\mu_i,\Sigma_{\gamma,i})(y)$ in the numerator of (2.2) and $Q_k(\mu_k,\Sigma_{\gamma,k})(y)$ in the denominator, as $Q_i(\mu_i,\Sigma_{\gamma,i})(y)$ will typically be tiny in such case and $Q_k(\mu_k,\Sigma_{\gamma,k})(y)$ will be large.

Algorithm 1 JAMS: main algorithm, iteration $n+1$

```
1: Input: current point $(x_n, i_n)$, list of modes $\{\mu_1, \dots \mu_N\}$, constant $\epsilon \in (0,1)$, parameter
   $\gamma_n = \{\Sigma_{\gamma_n,i}, w_{\gamma_n,i}, a_{\gamma_n,ik}\}_{i,k\in\{1,\dots,N\}}$, empirical means $m_1, \dots, m_N$ and covariance matri-
   ces $S_1, \dots, S_N$.
2: Generate $u \sim U[0,1]$.
3: if $u > \epsilon$ then
4:     Local move:
5:     Propose a new value $y \sim R_{\gamma_n,L,i_n}(x_n, \cdot)$.
6:     Accept $y$ with probability $\alpha_{\gamma_n,L}((x_n, i_n) \to (y, i_n))$.
7:     if $y$ accepted then
8:         $(x_{n+1}, i_{n+1}) := (y, i_n)$.
9:     else
10:        $(x_{n+1}, i_{n+1}) := (x_n, i_n)$.
11:    end if
12: else
13:    Jump move:
14:    Propose a new mode $k \sim (a_{\gamma_n,i1}, \dots, a_{\gamma_n,iN})$.
15:    Propose a new value $y \sim R_{\gamma_n,J,ik}(x_n, \cdot)$.
16:    Accept $(y,k)$ with probability $\alpha_{\gamma_n,J}((x_n, i_n) \to (y, k))$.
17:    if $(y,k)$ accepted then
18:        $(x_{n+1}, i_{n+1}) := (y, k)$.
19:    else
20:        $(x_{n+1}, i_{n+1}) := (x_n, i_n)$.
21:    end if
22: end if
23: Update the empirical mean $m_{i_{n+1}}$ and covariance matrix $S_{i_{n+1}}$ by including $x_{n+1}$.
24: Update the parameter $\gamma_n$ to $\gamma_{n+1}$ according to Algorithm 2.
25: return New sample $(x_{n+1}, i_{n+1})$, parameter $\gamma_{n+1}$, $m_{i_{n+1}}$ and $S_{i_{n+1}}$.
```

2.2. *Burn-in algorithm.* Algorithm 1 takes mode locations $\{\mu_1, \dots, \mu_N\}$ and initial values of the matrices $\{\Sigma_{\gamma_0,1}, \dots, \Sigma_{\gamma_0,N}\}$ as input. Since further improvements in the estimation of $\Sigma_{\gamma,i}$ are possible only after some samples in mode i have been observed, matrices

Algorithm 2 Updating the parameters, iteration $n+1$

```
1: Input: (in addition to the parameters of iteration $n+1$ of the main algorithm) number
   of samples observed so far in each mode $n_1, \dots, n_N$, auxiliary matrices $\tilde{\Sigma}_1, \dots, \tilde{\Sigma}_N$,
   positive integers $AC_1$ and $AC_2$, constants $\alpha, \alpha_{\text{opt}} \in (0,1)$, $\beta > 0$.
2: if $n_{i_{n+1}} < AC_1$ then
3:     if Local move then
4:         $\tilde{\Sigma}_{i_{n+1}} := \exp(n_{i_{n+1}}^{-\alpha}(\alpha_{\gamma_n,L} - \alpha_{\text{opt}}))\tilde{\Sigma}_{i_{n+1}}$.
5:         $\Sigma_{\gamma_{n+1},i_{n+1}} := \tilde{\Sigma}_{i_{n+1}} + \beta I_d$.
6:     end if
7: else
8:     if $n_{i_{n+1}}$ is divisible by $AC_2$ then
9:         $\Sigma_{\gamma_{n+1},i_{n+1}} := S_{i_{n+1}} + \beta I_d$.
10:        Update $w_{\gamma_n,i}$ and $a_{\gamma_n,ik}$ to $w_{\gamma_{n+1},i}$ and $a_{\gamma_{n+1},ik}$ for $i,k = 1, \dots, N$.
11:    end if
12: end if
```

$\{\Sigma_{\gamma_0,1}, \dots, \Sigma_{\gamma_0,N}\}$ need to represent well the shapes of the corresponding modes so that jumps to all the modes are accepted reasonably quickly. We address the issue of setting up these values by introducing a burn-in algorithm which runs before the main MCMC sampler (Algorithm 1) is initiated. Alternatively, one may prefer a version of this method in which the burn-in routine continues running in parallel to the main sampler on multiple cores, for example, to include modes possibly missed in the initial run. We sketch different stages of the burn-in routine below; additional details are given in Supplementary Material B [32].

2.2.1. *Starting points for the optimisation procedure.* We sample the starting points for optimisation searches uniformly on a compact set, which is a product of intervals $[L_1, U_1] \times \cdots \times [L_d, U_d]$, provided by the user. Note that if the domain of attraction of each mode overlaps with $[L_1, U_1] \times \cdots \times [L_d, U_d]$, then, asymptotically, all modes will be found, as we will have at least one starting point in each domain.

When dealing with Bayesian models, one can, alternatively, sample the starting points from the prior distribution.

2.2.2. *Mode finding via an optimisation procedure.* The BFGS optimisation algorithm [30] is initiated from every starting point. The BFGS method method provides the optimum point and the Hessian matrix at this point which is particularly useful in the next step of mode merging.

For numerical reasons, instead of working directly with π, we typically use the BFGS algorithm to find the local minima of $-\log(\pi)$.

2.2.3. *Mode merging.* Starting the optimisation procedure from different points belonging to the same basin of attraction will take us to points which are close to the true local maxima but numerically different, an issue that seems to be ignored in optimisation-based MCMC literature.

We deal with this by classifying two vectors, m_i and m_j, as corresponding to the same mode if the squared Mahalanobis distance between them is smaller than some prespecified value. If we let H_i and H_j denote the Hessian matrices of $-\log(\pi)$ at m_i and m_j, respectively, the above Mahalanobis distance is calculated for H_i^{-1} and H_j^{-1} (for symmetry, we average over these two values). This method is scale invariant as the Hessian captures the local shape and scale.

2.2.4. *Initial covariance matrix estimation.* In order to find initial covariance matrix estimates $\Sigma_{\gamma_0,1}, \dots, \Sigma_{\gamma_0,N}$ that accurately reflect the geometry of different modes, we employ the augmented target machinery of Algorithm 1 in the following way. We run Algorithm 1 without jumps, that is, with $\epsilon = 0$, in parallel, starting from each of the modes $\mu_1, \dots, \mu_N$. This implies that we run N chains, and each of them adapts only the matrix Σ_i corresponding to the mode μ_i which was its starting point. We make a number of rounds of this procedure, and after each round we update the target distribution $\tilde{\pi}$ by exchanging the knowledge about the adapted covariance matrices between cores. The final covariance matrices passed to the main MCMC sampler are calculated based on the samples collected in all rounds.

The reason why we exchange information between rounds, despite the additional cost of communication between cores, is that we want the sampler adapting $\Sigma_{k,i}$ to know where the regions associated with other modes are so that it is less likely to visit those regions and contaminate the estimate. Essentially, the initial covariance estimation revisits the problem of collecting samples only from the corresponding regions, discussed earlier.

The initial value of the matrix corresponding to mode i is the inverse of the Hessian evaluated at μ_i. The values of $w_{\gamma,i}$ and $a_{\gamma,ik}$ are set to $1/N$ and not updated during those runs. In Supplementary Material B [32] we propose a method of choosing the number of rounds automatically based on monitoring the inhomogeneity factor (see [36] and [37]).

2.3. *Further comments.* It is important to point out that the auxiliary variable approach presented above should be thought of as a flexible framework rather than one specific method. The BFGS algorithm used for mode finding could be replaced with another optimisation procedure. Similarly, instead of the Random Walk Metropolis algorithm, local moves could be performed with a different MCMC sampler, such as HMC or MALA. One could also consider another scheme for updating the parameters, for example, combining adaptive scaling with covariance matrix estimation (see [42]).

3. Auxiliary Variable Adaptive MCMC.

We introduce a general class of Auxiliary Variable Adaptive MCMC algorithms, as follows.

Recall that $\pi(\cdot)$ is a fixed target probability density on $(\mathcal{X}, \mathcal{B}(\mathcal{X}))$. For an auxiliary pair $(\Phi, \mathcal{B}(\Phi))$, define $\tilde{\mathcal{X}} := \mathcal{X} \times \Phi$, and, for an index set $\mathcal{Y}$, consider a family of probability measures $\{\tilde{\pi}_\gamma(\cdot)\}_{\gamma \in \mathcal{Y}}$ on $(\tilde{\mathcal{X}}, \mathcal{B}(\tilde{\mathcal{X}}))$ such that

$$\tilde{\pi}_\gamma(B \times \Phi) = \pi(B) \quad \text{for every } B \in \mathcal{B}(\mathcal{X}) \text{ and } \gamma \in \mathcal{Y}. \tag{3.1}$$

Let $\{\tilde{P}_\gamma\}_{\gamma \in \mathcal{Y}}$ be a collection of Markov chain transition kernels on $(\tilde{\mathcal{X}}, \mathcal{B}(\tilde{\mathcal{X}}))$ such that each $\tilde{P}_\gamma$ has $\tilde{\pi}_\gamma$ as its invariant distribution and is Harris ergodic.

To define the dynamics of the Auxiliary Variable Adaptive MCMC sequence $\{(\tilde{X}_n, \Gamma_n)\}_{n=0}^\infty$ where Γ represents a random variable taking values in $(\mathcal{Y}, \mathcal{B}(\mathcal{Y}))$, denote its filtration as $\mathcal{G}_n := \sigma\{\tilde{X}_0, \dots, \tilde{X}_n, \Gamma_0, \dots, \Gamma_n\}$. The conditional distribution of Γ_{n+1}, given $\mathcal{G}_n$, will be specified by the adaptive algorithm being used, such as Algorithm 1, while the dynamics of the $\tilde{X}$ coordinate follows

$$\mathbb{P}[\tilde{X}_{n+1} \in \tilde{B} | \tilde{X}_n = \tilde{x}, \Gamma_n = \gamma, \mathcal{G}_{n-1}] = \tilde{P}_\gamma(\tilde{x}, \tilde{B}) \tag{3.2}$$

for $\tilde{x} \in \tilde{\mathcal{X}}, \gamma \in \mathcal{Y}, \tilde{B} \in \mathcal{B}(\tilde{\mathcal{X}})$. Note that, depending on the adaptive update rule for Γ_n, the sequence $\{(\tilde{X}_n, \Gamma_n)\}_{n=0}^\infty$, defined above, is not necessarily a Markov chain. By $\tilde{A}_n(\cdot)$ denote the distribution of the $\tilde{\mathcal{X}}$-marginal of $\{(\tilde{X}_n, \Gamma_n)\}_{n=0}^\infty$ at time n, conditionally on the starting points, that is,

$$\tilde{A}_n(\tilde{B}) := \mathbb{P}[\tilde{X}_n \in \tilde{B} | \tilde{X}_0 = \tilde{x}, \Gamma_0 = \gamma] \quad \text{for } \tilde{B} \in \mathcal{B}(\tilde{\mathcal{X}}).$$

By $A_n(\cdot)$ denote the further marginalisation of $\tilde{A}_n(\cdot)$ onto the space of interest $\mathcal{X}$, namely,

$$A_n(B) := \tilde{A}_n(B \times \Phi) \quad \text{for } B \in \mathcal{B}(\mathcal{X}).$$

Finally, in order to define ergodicity of the Auxiliary Variable Adaptive MCMC, let

$$T_n(\tilde{x}, \gamma) := \|A_n(\cdot) - \pi(\cdot)\|_{\mathrm{TV}} = \sup_{B \in \mathcal{B}(\mathcal{X})} |A_n(B) - \pi(B)|.$$

DEFINITION 3.1. We say that the Auxiliary Variable Adaptive MCMC algorithm generating $\{(\tilde{X}_n, \Gamma_n)\}_{n=0}^\infty$ is *ergodic*, if

$$\lim_{n \to \infty} T_n(\tilde{x}, \gamma) = 0 \quad \text{for all } \tilde{x} \in \tilde{\mathcal{X}}, \gamma \in \mathcal{Y}.$$

It can be easily checked that JAMS belongs to the class defined above. There exist other algorithms falling into this category; therefore, the results presented in this paper, in particular Theorems 3.2, 3.3 and 3.4, may be useful for analysing their ergodicity. Examples of other algorithms in this class include adaptive parallel tempering [27] and adaptive versions of pseudo-marginal algorithms [4, 6]. A more detailed discussion on this may be found in Supplementary Material A [32].

3.1. *Theoretical results for the class.* The two main approaches to verifying ergodicity of Adaptive MCMC are based on martingale approximations [3, 7, 16] or coupling [35]. Here, we extend the latter to the Auxiliary Variable Adaptive MCMC class by constructing explicit couplings. In particular, ergodicity of this class of algorithms will be verified for the uniform and the nonuniform case, providing results analogous to Theorems 1 and 2 of [35].

For the uniform case analogues of the usual conditions of Simultaneous Uniform Ergodicity and Diminishing Adaptation will be required.

THEOREM 3.2 (Ergodicity—uniform case). *Consider an Auxiliary Variable Adaptive MCMC algorithm on a state space $\tilde{\mathcal{X}} = \mathcal{X} \times \Phi$, following dynamics* (3.2) *with a family of transition kernels $\{\tilde{P}_\gamma\}_{\gamma \in \mathcal{Y}}$ satisfying* (3.1) *and such that $\tilde{P}_\gamma$ is $\tilde{\pi}_\gamma$-ergodic for each $\gamma \in \mathcal{Y}$. If conditions* (*a*) *and* (*b*) *below are satisfied, then the algorithm is ergodic in the sense of Definition* 3.1:

(a) (*Simultaneous Uniform Ergodicity*) *For all $\varepsilon > 0$, there exists $N = N(\varepsilon) \in \mathbb{N}$ such that*

$$\|\tilde{P}_\gamma^N(\tilde{x}, \cdot) - \tilde{\pi}_\gamma(\cdot)\|_{\mathrm{TV}} \le \varepsilon, \qquad \text{for all} \tilde{x} \in \tilde{\mathcal{X}} \text{ and } \gamma \in \mathcal{Y}.$$

(b) (*Diminishing Adaptation*) *The random variable*

$$D_n := \sup_{\tilde{x} \in \tilde{\mathcal{X}}} \|\tilde{P}_{\Gamma_{n+1}}(\tilde{x}, \cdot) - \tilde{P}_{\Gamma_n}(\tilde{x}, \cdot)\|_{\mathrm{TV}} \tag{3.3}$$

converges to 0 *in probability.*

In fact, assumption (a) of Theorem 3.2 can be relaxed. To this end, define the ε-convergence time as

$$M_\varepsilon(\tilde{x}, \gamma) := \inf\{k \ge 1 : \|\tilde{P}_\gamma^k(\tilde{x}, \cdot) - \tilde{\pi}_\gamma(\cdot)\|_{\mathrm{TV}} \le \varepsilon\}. \tag{3.4}$$

It is enough that the random variable $M_\varepsilon(\tilde{X}_n, \Gamma_n)$ is bounded in probability. Precisely, the following ergodicity result holds for the nonuniform case.

THEOREM 3.3 (Ergodicity—nonuniform case). *Consider an Auxiliary Variable Adaptive MCMC algorithm, under the assumptions of Theorem* 3.2, *and replace condition* (*a*) *with the following*:

(a) (*Containment*) *For all $\varepsilon > 0$ and all $\tilde{\delta} > 0$, there exists $N = N(\varepsilon, \tilde{\delta})$ such that*

$$\mathbb{P}(M_\varepsilon(\tilde{X}_n, \Gamma_n) > N | \tilde{X}_0 = \tilde{x}, \Gamma_0 = \gamma) \le \tilde{\delta} \quad \text{for all } n \in \mathbb{N}. \tag{3.5}$$

Then, the algorithm is ergodic in the sense of Definition 3.1.

In our proofs of Theorems 3.2 and 3.3, we introduce two auxiliary processes $\{(\tilde{X}_n^m, \Gamma_n^m)\}_{n=0}^\infty$ and $\{(\tilde{X}_n^i, \Gamma_n^i)\}_{n=0}^\infty$ on $\mathcal{X} \times \Phi$, to be thought of as "Markovian" and "intermediate." The latter interpolates between the Markovian one and the original adaptive process. Crucial for our proof is the coupling construction which shows that the total variation distance between the two pairs of processes goes to 0. We then observe that the total variation distance between processes on $\mathcal{X} \times \Phi$ bounds from above the distance between their marginalisations onto $\mathcal{X}$. Finally, we use the triangle inequality to conclude ergodicity of the adaptive process. All the processes are constructed explicitly on the same probability space which makes our proofs adaptable to other cases.

We use the same coupling construction to establish the Weak Law of Large Numbers for the class of Auxiliary Variable Adaptive MCMC algorithms for both the uniform and the nonuniform case. By letting Φ be a singleton, our result applies to the standard Adaptive MCMC setting and extends the result of [35] where the WLLN was provided for the uniform case only.

THEOREM 3.4 (WLLN). *Consider an Auxiliary Variable Adaptive MCMC algorithm, as in Theorem 3.3, together with assumptions a) and b) of this theorem. Let $g : \mathcal{X} \to \mathbb{R}$ be a bounded measurable function. Then,*

$$\frac{\sum_{i=1}^{n} g(X_i)}{n} \to \pi(g) \quad \textit{in probability as } n \to \infty.$$

While Containment is a weaker condition than Simultaneous Uniform Ergodicity, it is less tractable, and in the standard Adaptive MCMC setting drift conditions are typically used to verify it [9, 35]. Lemma 3.5 helps verifying Containment via geometric drift conditions in the Auxiliary Variable framework.

LEMMA 3.5. *Assume that the following conditions are satisfied:*

(a) *For each $\gamma \in \mathcal{Y}$ $\|\tilde{P}_{\gamma}^{k}(\tilde{x}, \cdot) - \tilde{\pi}_{\gamma}(\cdot)\|_{\mathrm{TV}} \to 0$ as $k \to \infty$.*

(b) *There exists $\lambda < 1$, $b < \infty$ and a collection of functions $V_{\tilde{\pi}_{\gamma}} : \tilde{\mathcal{X}} \to [1, \infty)$ for $\gamma \in \mathcal{Y}$ such that the following simultaneous drift condition is satisfied:*

$$\tilde{P}_{\gamma} V_{\tilde{\pi}_{\gamma}}(\tilde{x}) \leq \lambda V_{\tilde{\pi}_{\gamma}}(\tilde{x}) + b \quad \textit{for all } \tilde{x} \in \tilde{\mathcal{X}} \textit{ and } \gamma \in \mathcal{Y}, \tag{3.6}$$

where for $\tilde{x} \in \tilde{\mathcal{X}}$

$$\tilde{P}_{\gamma} V_{\tilde{\pi}_{\gamma}}(\tilde{x}) := \mathbb{E}(V_{\tilde{\pi}_{\gamma}}(\tilde{X}_{n+1}) | \tilde{X}_n = \tilde{x}, \Gamma_n = \gamma).$$

Moreover, $V_{\tilde{\pi}_{\gamma}}(\tilde{x})$ is bounded on compact sets as a function of $(\tilde{x}, \gamma)$.

(c) *There exist $\delta > 0$, $v > 2n_0 b/(1 - \lambda^{n_0})$ and a positive integer n_0 such that the following minorisation condition holds: for each $\gamma \in \mathcal{Y}$ we can find a probability measure ν_{γ} on $\tilde{\mathcal{X}}$ satisfying*

$$\tilde{P}_{\gamma}^{n_0}(\tilde{x}, \cdot) \geq \delta \nu_{\gamma}(\cdot) \quad \textit{for all } \tilde{x} \textit{ with } V_{\tilde{\pi}_{\gamma}}(\tilde{x}) \leq v. \tag{3.7}$$

(d) *$\mathcal{Y}$ is compact in some topology.*

(e) *There exists a compact set A such that if $X_n \notin A$, then $\Gamma_{n+1} = \Gamma_n$.*

(f) *$\mathbb{E} V_{\tilde{\pi}_{\Gamma_0}}(\tilde{X}_0) < \infty$.*

Then, the Containment condition (3.5) holds.

Lemma 3.5 bears resemblance to Theorem 3 of [35]. However, in our setting drift functions need to depend on the target distribution $\tilde{\pi}_{\gamma}$, which modifies the form of the standard drift condition, and, as a consequence, we cannot use the proof of this theorem directly to obtain our result. In particular, our lemma additionally assumes that the adaptation happens on a compact set only.

3.2. *Adaptive Increasingly Rarely version of the class.* Adaptive Increasingly Rarely (AIR) MCMC algorithms were introduced in [11] as an alternative to classical Adaptive MCMC methods. While they share the same selftuning properties, their ergodic properties are mathematically easier to analyse, and their computational cost of adaptation is smaller.

The key idea behind the AIR algorithms is to allow for updates of parameters only at prespecified times N_j with an increasing sequence of lags n_k between them. For the sequence $\{n_k\}_{k>1}$, [11] proposed using any scheme that satisfies $c_2 k^{\kappa} \geq n_k \geq c_1 k^{\kappa}$ for some positive c_1, c_2 and κ. In order to ensure that the random variable D_n defined by (3.3) converges to 0 in probability, that is, diminishing adaptation holds, the following modification is introduced. The updates happen at times N_j^*, where

$$N_j^* = \sum_{k=1}^{j} n_k^* \quad \text{with } N_0^* = 0 \text{ and}$$

$$n_k^* = n_k + \text{Uniform}\big[0, \lfloor k^{\kappa^*} \rfloor\big] \quad \text{for some } \kappa^* \in (0, \kappa) \text{ and } n_0^* = 0.$$

Observe that D_n is only positive if $n + 1 \in \{N_j^*\}_{j \geq 1}$. Besides, if $n + 1 > N_k$, then $\mathbb{P}(D_n > 0) \leq \frac{1}{\lfloor k^{\kappa^*} \rfloor}$; so, in particular, D_n goes to 0 as n tends to infinity.

We apply the same idea to Auxiliary Variable Adaptive MCMC algorithms, by adapting the parameters of the transition kernels and the target distributions only at times N_j^*, as described above, so that Diminishing Adaptation is automatically satisfied for these algorithms. In Section 4 we study in detail an AIR version of JAMS.

4. Ergodicity of the Jumping Adaptive Multimodal Sampler.

The main results of this section are stated in Theorems 4.1 and 4.2 which establish convergence of our algorithm to the correct limiting distribution under the uniform and the nonuniform scenario, respectively. To prove these theorems, we will use our results from Section 3.

4.1. *Overview of the assumptions.*

In order to prove ergodic results for JAMS, we consider a slightly modified version of Algorithm 1, called mJAMS herein. While being easier to analyse mathematically, it inherits the main properties of Algorithm 1. The modifications are twofold:

1. We update the parameters only if the most recent sample (x_n, i_n) is such that x_n belongs to some fixed compact set A_{i_n}.
2. We adapt the parameters "increasingly rarely" (see Section 3.2).

In Supplementary Material A [32] we present the pseudocode of the resulting algorithm and make a remark on how to choose the compact sets $A_1, \ldots, A_N$. If jumps are proposed deterministically, we additionally assume that they are allowed only on "jumping regions" $JR_{\gamma,i}$ defined as

$$JR_{\gamma,i} = \big\{x \in \mathcal{X} : (x - \mu_i)^T \Sigma_{\gamma,i}^{-1}(x - \mu_i) \leq R\big\} \tag{4.1}$$

for $i \in \mathcal{I}$ and some $R > 0$. Note that equation (2.4) ensures that if x belongs to $JR_{\gamma,i}$ and we propose a deterministic jump from (x, i) to (y, k), then y must be in $JR_{\gamma,k}$. Thus, the detailed balance condition is satisfied. The reasons for these modifications will become clearer when we present the proofs of the ergodic theorems.

Observe that each matrix $\Sigma_{\gamma,i}$ is based on samples belonging to a compact set A_i; so, these matrices are bounded from above. Since we keep adding βI_d to the covariance matrix at each step, they are also bounded from below. Consequently, there exist positive constants m and M for which

$$m I_d \preceq \Sigma_{\gamma,i}, \preceq M I_d \quad \text{for all } \gamma \in \mathcal{Y} \text{ and } i \in \mathcal{I}. \tag{4.2}$$

As for the adaptive scheme for $w_{\gamma,i}$ and $a_{\gamma,ik}$, we only require that these values be bounded away from 0, that is, there exist ϵ_a and ϵ_w such that

$$w_{\gamma,i} > \epsilon_w \quad \text{and} \quad a_{\gamma,ik} > \epsilon_a \quad \text{for all } \gamma \in \mathcal{Y} \text{ and } i, k \in \mathcal{I}. \tag{4.3}$$

Therefore, the parameter space $\mathcal{Y}$ may be considered as compact.

4.2. *Theoretical results for JAMS.* We begin with the case when the jump moves are proposed independently from distributions $R_{\gamma,J,i}$ with heavier tails than the tails of the target distribution π for all $i \in \mathcal{I}$ and $\gamma \in \mathcal{Y}$, that is,

$$\sup_{x\in\mathcal{X}} \sup_{\gamma\in\mathcal{Y}} \frac{\pi(x)}{R_{\gamma,J,i}(x)} < \infty \quad \text{for each } i \in \mathcal{I}. \tag{4.4}$$

We prove that, under this assumption, Simultaneous Uniform Ergodicity is satisfied for mJAMS and, consequently, by Theorem 3.2 the algorithm is ergodic.

THEOREM 4.1. *Consider the mJAMS algorithm, and assume that the relationship between the target distribution π and the proposal distributions $R_{\gamma,J,i}$ satisfies* (4.4). *Then, mJAMS is ergodic.*

When the tails of the distribution π are heavier, then the tails of the proposal distributions $R_{\gamma,J,i}$, or, when the jumps follow the deterministic scheme, Simultaneous Uniform Ergodicity does not hold. However, it turns out that, under some additional regularity conditions, mJAMS is still ergodic, as it satisfies the assumptions of Lemma 3.5.

THEOREM 4.2. *Consider the mJAMS algorithm, and assume that the following conditions are satisfied*:

(a) *For each $i \in \mathcal{I}, \gamma \in \mathcal{Y}$, the proposal distribution for local moves $R_{\gamma,L,i}$ follows an elliptical distribution parametrised by $\Sigma_{\gamma,i}$. Furthermore, the family of distributions, $R_{\gamma,L,i}(\mathbf{0},\cdot)$, $\gamma \in \mathcal{Y}$, has uniformly bounded probability density functions, and, for any compact set $C \subset \mathcal{X}$, we have*

$$\inf_{x,y\in C} \inf_{\gamma\in\mathcal{Y}} R_{\gamma,L,i}(x,y) > 0 \quad \textit{for each } i \in \mathcal{I}. \tag{4.5}$$

(b) *Let $r_{\gamma,i}(x)$ be the rejection set for local moves, that is, $r_{\gamma,i}(x) := \{y \in \mathcal{X} : \tilde{\pi}_\gamma(y,i) < \tilde{\pi}_\gamma(x,i)\}$. We assume that*

$$\limsup_{|x|\to\infty} \sup_{\gamma\in\mathcal{Y}} \int_{r_{\gamma,i}(x)} R_{\gamma,L,i}(x,y)\,dy < 1 \quad \textit{for each } i \in \mathcal{I}. \tag{4.6}$$

(c) *The target distribution π is superexponential, that is, it is positive with continuous first derivatives and satisfies*

$$\lim_{|x|\to\infty} \frac{x}{|x|} \cdot \nabla \log \pi(x) = -\infty. \tag{4.7}$$

(d) *Every Q_i, $i \in \mathcal{I}$, is an elliptical distribution parametrised by $\Sigma_{\gamma,i}$ positive on $\mathcal{X}$ and, additionally, the following condition is satisfied*:

$$\sup_{x\in\mathcal{X}} \frac{Q_i(\mu_i, \Sigma_{\gamma_1,i})(x)}{Q_k(\mu_k, \Sigma_{\gamma_2,k})(x)} < \infty \quad \textit{for all } i,k \in \mathcal{I} \textit{ and } \gamma_1,\gamma_2 \in \mathcal{Y}. \tag{4.8}$$

Additionally, one of the following two conditions for jump moves holds:

(e1) *Jump moves follow the procedure for deterministic jumps, as described in Section* 2.1.
(e2) *Jump moves follow the independent proposal procedure, as described in Section* 2.1. *The proposal distributions for jumps have uniformly bounded probability density functions and satisfy*

$$\inf_{x\in B(\mu_i,r)} \inf_{\gamma\in\mathcal{Y}} R_{\gamma,J,i}(x) > 0 \quad \textit{for each } i \in \mathcal{I} \textit{ and some } r > 0, \tag{4.9}$$

where $B(\mu_i, r)$ is a ball of radius r and centre μ_i. Moreover, the relationship between the target distribution $R_{\gamma,J,i}$ is given by

$$\sup_{x\in\mathcal{X}}\sup_{\gamma\in\mathcal{Y}}\frac{R_{\gamma,J,i}(x)}{\pi(x)^{s_J}}<\infty \quad \textit{for each } i\in\mathcal{I} \textit{ and some } s_J\in(0,1]. \tag{4.10}$$

Then, mJAMS is ergodic.

The most challenging part of the proof is verifying drift condition (3.6), which we do for $V_{\tilde{\pi}_\gamma}((x,i)) = c\tilde{\pi}_\gamma(x,i)^{-s}$, $c > 0$, $s \in (0,1)$, separately for local and jump kernels. When proving the result for local moves, we show that $\limsup_{|x|\to\infty}\sup_{\gamma\in\mathcal{Y}}\frac{\tilde{P}_{\gamma,L,i}V_{\tilde{\pi}_\gamma}((x,i))}{V_{\tilde{\pi}_\gamma}((x,i))} < 1$, following the proof of Theorem 4.1 of [22]. That is, we consider analogues of the regularity conditions of this theorem (assumptions (b) and (c) above), and we split the state space $\mathcal{X}$ into three disjoint sets, like in [22], to show that, for large x, the value of the integral of $R_{\gamma,L,i}(x,y)\min[1,\frac{\tilde{\pi}_\gamma(y,i)}{\tilde{\pi}_\gamma(x,i)}]\frac{V_{\tilde{\pi}_\gamma}((y,i))}{V_{\tilde{\pi}_\gamma}((x,i))}$ with respect to y is arbitrarily small on each set. The main difference between our approach and that of [22] is that we work with adaptive kernels and target distributions which necessitates bounding the suprema of the involved expressions over $\gamma \in \mathcal{Y}$. Besides, assumption (d) is introduced to ensure that $\tilde{\pi}_\gamma$ does not "deviate too much" from π. This condition can be easily verified if every Q_i, $i \in \mathcal{I}$ follows the t distribution with the same number of degrees of freedom.

For the jump kernels the idea is to show that $\sup_{\gamma\in\mathcal{Y}} V_{\tilde{\pi}_\gamma}((y,k))$ is "not too large" when the algorithm jumps to (y,k). For deterministic jumps performed on jumping regions, it follows from the boundedness of $V_{\tilde{\pi}_\gamma}((x,i))$ on compact sets. For independent proposal jumps, we show that under (4.10) $\sup_{\gamma\in\mathcal{Y}}\int_{\mathcal{X}} R_{\gamma,J,k}(y)V_{\tilde{\pi}_\gamma}((y,k))\,dy < \infty$.

Note that condition a) holds automatically for our algorithm if we assume that the proposal distributions for local moves follow either the normal or the t distribution (see Section 2.1) and when (4.2) holds. Condition (4.9) is satisfied if the proposal distributions for jumps follow, for example, the normal distribution. The following lemmas are useful in verifying assumption b) of Theorem 4.2:

LEMMA 4.3. *Let $r(x) := \{y \in \mathcal{X} : \pi(y) < \pi(x)\}$ and $a(x) := \{y \in \mathcal{X} : \pi(y) \geq \pi(x)\}$. Consider mJAMS together with conditions a), c) and d) of Theorem 4.2. Assume, additionally, that for some $\gamma^* \in \mathcal{Y}$*

$$\limsup_{|x|\to\infty}\int_{r(x)} R_{\gamma^*,L,i}(x,y)\,dy < 1 \quad \textit{for each } i\in\mathcal{I}. \tag{4.11}$$

Then, condition (4.6) holds.

LEMMA 4.4. *Consider mJAMS together with conditions* (a), (c) *and* (d) *of Theorem 4.2. Additionally, assume that the target distribution π satisfies*

$$\limsup_{|x|\to\infty}\frac{x}{|x|}\cdot\frac{\nabla\pi(x)}{|\nabla\pi(x)|} < 0. \tag{4.12}$$

Then, condition (4.6) holds.

The following corollary shows that mJAMS in a standard setting is successful at targeting mixtures of normal distributions:

COROLLARY 4.5. *Let the target distribution π be given by*

$$\pi(x) \propto w_1 \exp(-p_1(x)) + \cdots + w_n \exp(-p_n(x)),$$

where $w_i > 0$ and p_i is a polynomial of order ≥ 2 for each $i = 1, \dots, n$. Additionally, if Q_i for $i \in \mathcal{I}$ follows the multivariate t distribution with the same number of degrees of freedom and $R_{\gamma,L,i}(\mathbf{0}, \cdot)$ follows the normal distribution, the assumptions of Lemma 4.4 are satisfied.

The result stated below is a corollary from Theorem 3.4 and establishes the Weak Law of Large Numbers for our algorithm.

THEOREM 4.6. *Consider mJAMS, and assume that conditions of either Theorem 4.1 or Theorem 4.2 are satisfied. Then, the Weak Law of Large Numbers holds for all bounded and measurable functions.*

REMARK 4.7. Note that Theorems 4.1, 4.2 and 4.6 are based on an assumption that the list of modes is fixed. Let us now consider mJAMS in the version with mode finding running in parallel to the main MCMC sampler. Assume, additionally, that $\mathbb{P}(\tau < t) \to 1$ as $t \to \infty$, where τ is the time of adding the last mode. In this case Theorems 4.1, 4.2 and 4.6 still hold. Indeed, as the parallel burn-in algorithm runs independently of mJAMS, we can rephrase all the probabilistic limiting statements in the proofs on the set $C_t := \{\tau < t\}$ and then let $t \to \infty$.

An important question to ask is how the performance of the algorithm changes with the dimension of the state space. In case of (nonadaptive) MCMC methods this is typically measured by the rate at which the spectral gap of a transition kernel decreases. In Supplementary Material A [32] we analyse the spectral gap of an individual JAMS kernel. To obtain a formal result, we consider the following target density:

$$\pi(x) = \frac{1}{2} N\Big(-\underbrace{(1, \dots, 1)}_{d}, \sigma_1^2 I_d\Big) + \frac{1}{2} N\Big(\underbrace{(1, \dots, 1)}_{d}, \sigma_2^2 I_d\Big). \tag{4.13}$$

Parallel tempering is known to mix torpidly on this example (see [45]) for any choice of temperatures when $\sigma_1 \neq \sigma_2$. We show that, given the locations of the modes and a certain rate of approximation of the proposal covariance matrices to the true ones, an individual JAMS kernel with independent proposal jumps mixes rapidly on this target density. The proof of this result relies on observing that the jump kernel following the independent proposal scheme is, in fact, an independence sampler kernel; thus, we can use a result from [17] to obtain its spectral gap.

We also argue, without proving a formal result, why the number of iterations of covariance matrix estimation in the burn-in algorithm required to obtain the above rate of approximation is polynomial with respect to the dimension. Hence, given the locations of the modes, this would give an overall polynomial time of an algorithm composed of the covariance matrix estimation in the burn-in algorithm, followed by a nonadaptive version of JAMS targeting (4.13).

5. Examples.

In this section we present empirical results for our method (Algorithm 1 preceded by the burn-in algorithm). We test its performance on three examples: the first one is a mixture of two Gaussians motivated by [45]; the second one is a mixture of 15 multivariate t distributions and five banana-shaped ones; the third one is a Bayesian model for sensor network localisation. Our implementation admits three versions, varying in the way the jumps between modes are performed. In particular, we consider here the deterministic jump and two independent proposal jumps with Gaussian and t-distributed proposals.

Additionally, we compare the performance of our algorithm against adaptive parallel tempering [27], which was chosen here as it is the refined version of the most commonly used MCMC method for multimodal distributions (parallel tempering). What is more, this algorithm has a generic implementation, where the user only needs to provide the target density function. In order to make a comparison between the efficiency of these algorithms, among other things, we analyse the Root Mean Square Error (RMSE) divided by the square root of the dimension of the state space, given a computational budget. We measure the computational cost by the number of evaluations of the target distribution (and its gradient, if applicable), as this is typically the dominating factor in real data examples. Herein, we define RMSE as the Euclidean distance between the true d-dimensional expected value (if known) and its empirical estimate based on MCMC samples.

In order to depict the variability in the results delivered by both methods, each simulation was repeated 20 times. For exact settings of the experiments as well as some additional results, we refer the reader to Supplementary Material B [32].

5.1. *Mixture of Gaussians.* We looked at the results for the target distribution (4.13) in several different dimensions, d, ranging between 10 and 200, for $\sigma_1^2 = 0.5\sqrt{d/100}$ and $\sigma_2^2 = \sqrt{d/100}$. The results for our method shown below are based on 500,000 iterations of the main algorithm, preceded by the burn-in algorithm including 1500 BFGS runs, started from points sampled uniformly on $[-2, 2]^d$.

The length of the covariance matrix estimation was chosen automatically using the rule described in Supplementary Material B [32] and varied between 3000 (for $d = 10$) to 1,023,000 iterations (for $d = 200$) per mode. For dimensions $d = 10$ and $d = 20$, we ran also the adaptive parallel tempering (APT) algorithm, with 700,000 iterations and five temperatures. Overall, this requires 3,500,000 evaluations of the target density that cannot be performed in parallel, despite the name of the method, as the communication between chains running at different temperatures is needed after every iteration. In the light of the tendency of the parallel tempering algorithm to stay in wider modes, each time the APT algorithm was started in $-\underbrace{(1, \dots, 1)}_{d} \in \mathbb{R}^d$. In order to base our analysis on the same sample size of 500,000 for the two methods, in case of adaptive parallel tempering we applied an initial burn-in period of 200,000 steps.

The results presented in the boxplots of Figure 1, as well as the upper panel of density plots (Figure 3), show that our method outperforms adaptive parallel tempering on this example, even when the latter method is given a much larger computational budget. The summary of the acceptance rates of the jump moves presented in Table 1 demonstrates that the algorithm preserves good mixing between the modes in all its jump versions up to dimension 80. It is remarkable that the deterministic jump ensures excellent mixing even in much higher dimensions, outperforming the remaining two methods (see Figure 2 and the lower panel of Figure 3), with the acceptance rate between 0.64 and 0.97 in dimension 200.

5.2. *Mixture of t and banana-shaped distributions.* In order to test our method on a multimodal target with highly nonelliptical modes, we used a modified and more challenging version of a classic example introduced in [24] (a mixture of 20 bivariate Gaussian distributions), also studied later by [27] and [39].

In our case, instead of the Gaussian distribution, modes 1–5 follow the banana-shaped distribution with t tails, where the shape of the banana for mode k is present in the projection onto coordinates 1 and $2k$. The remaining ones follow the multivariate t distribution with

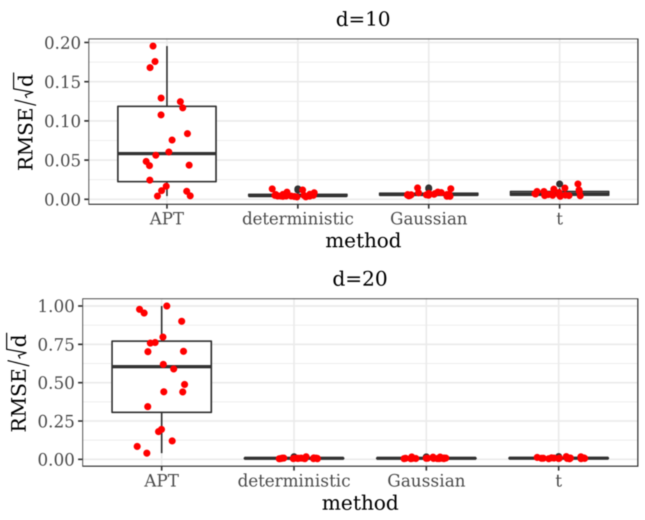

FIG. 1. *Boxplots of the values of RMSE/$\sqrt{d}$ for the mixture of Gaussians across* 20 *runs of the experiment, dimensions* 10 *and* 20. *We compare the results of APT with the three JAMS versions*: *deterministic*, *Gaussian and* t*-distributed jumps. Note different scales on the* y*-axis.*

seven degrees of freedom and the covariance matrices $0.01\sqrt{d}I_d$, where d is the dimension. The weights are assumed to be equal to 0.05. We consider dimensions $d = 10$ and $d = 20$ by repeating the original coordinates of the centres of the modes five and 10 times, respectively.

The results below are based on 500,000 iterations, preceded by 40,000 BFGS runs. The starting points for these runs were sampled uniformly on $[-2, 12]^d$. The number of iterations of the covariance matrix estimation varied between 7000 and 15,000 steps per mode for dimension $d = 10$ and between 15,000 and 63,000 steps per mode for dimension $d = 20$. For adaptive parallel tempering we used 2,100,000 iterations and five temperatures. We applied an initial burn-in period of 600,000 steps, and we thinned the chain keeping every third sample.

In Supplementary Material B [32] we present results for the same example obtained using JAMS in dimensions $d = 50$ and $d = 80$ assuming that the modes of the target distribution are known, since mode finding (in particular, getting to each basin of attraction) is the main bottleneck for this example.

For dimensions $d = 10$ and $d = 20$, all modes were found by the BFGS runs in each of the 20 simulations. Figure 4 illustrates that the empirical means based on JAMS samples approximate well the true expected value of the target distribution, consistently across all experiments and that our method significantly outperforms APT with a smaller computational

TABLE 1
The lowest and the highest value (*across* 20 *runs of the experiment*) *of the acceptance rates of jump moves between the two modes for the mixture of Gaussians for different jump methods and dimensions*

	Deterministic		Gaussian		t-distributed	
	Lowest	Highest	Lowest	Highest	Lowest	Highest
$d = 10$	0.98	0.99	0.85	0.87	0.71	0.73
$d = 20$	0.98	0.99	0.79	0.83	0.66	0.68
$d = 80$	0.91	0.98	0.23	0.41	0.24	0.39
$d = 130$	0.72	0.98	0.04	0.13	0.06	0.15
$d = 160$	0.79	0.97	0.01	0.07	0.03	0.07
$d = 200$	0.64	0.97	0.01	0.05	0.02	0.06

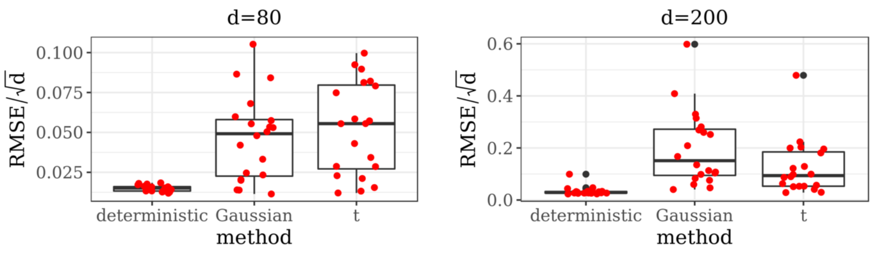

FIG. 2. *Boxplots of the values of RMSE/$\sqrt{d}$ for the mixture of Gaussians across* 20 *runs of the experiment, dimensions* 80 *and* 200. *Note different scales on the y-axis.*

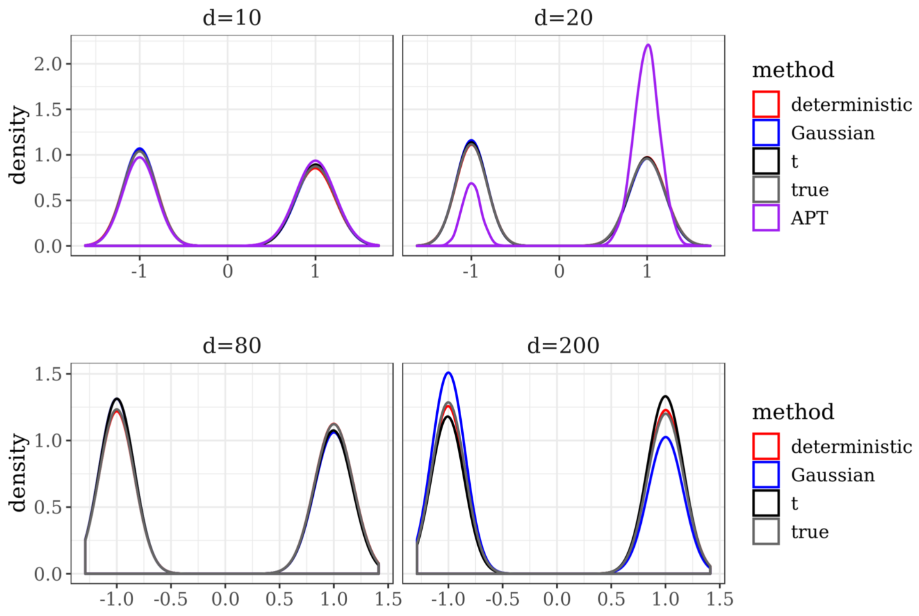

FIG. 3. *Density plots for the mixture of Gaussians in dimensions* 10, 20, 80 *and* 200. *The upper panel shows a comparison between APT and the three JAMS versions. The simulations chosen for the analysis correspond to the median value of RMSE across* 20 *experiments* (*the tenth largest value of RMSE*).

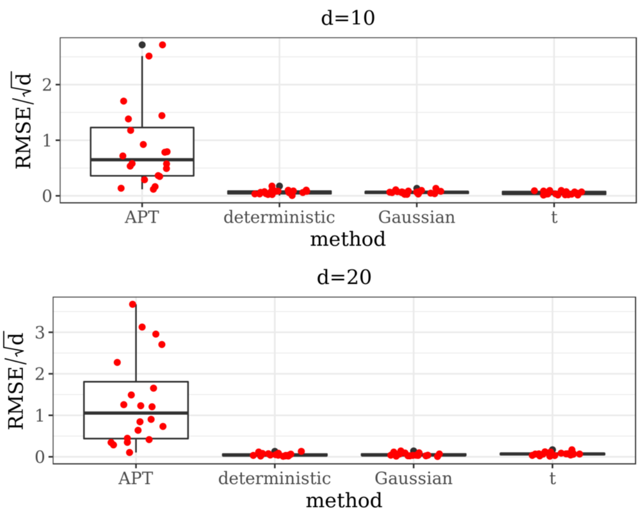

FIG. 4. *Boxplots of the values of RMSE/$\sqrt{d}$ for the mixture of banana-shaped and t-distributions across* 20 *runs of the experiment, dimensions* 10 *and* 20. *We compare the results of APT with the three JAMS versions: deterministic, Gaussian and t-distributed jumps. Note different scales on the y-axis.*

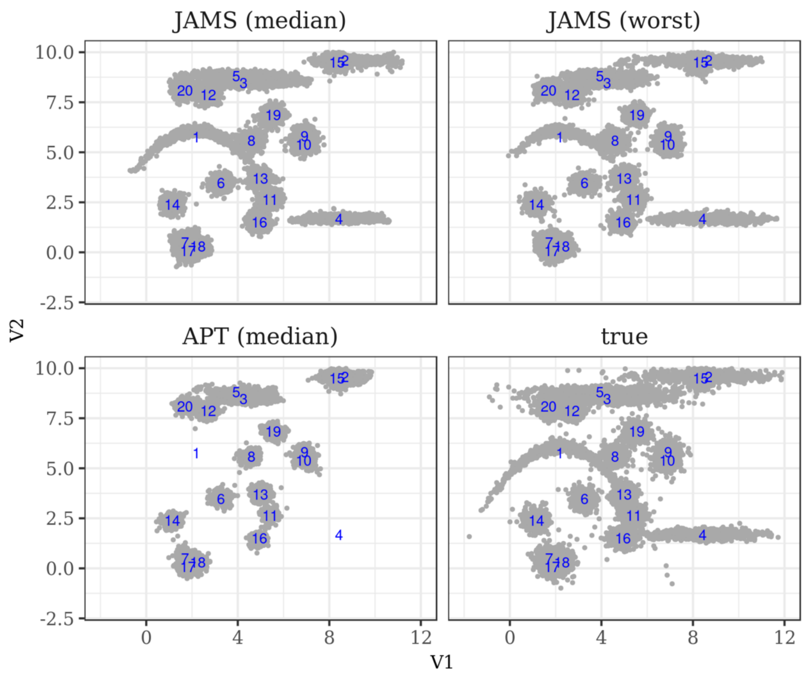

FIG. 5. *Scatterplots of the first and second coordinate of the mixture of t and banana-shaped distributions in dimension* 20. *Recall that modes* 1–5 *follow the banana-shaped distribution with t tails, where the shape of the banana for mode k is present in the projection onto coordinates* 1 *and* $2k$. *The blue numbers denote the centres of the modes. We compare the results obtained using JAMS* (*with t-distributed jumps*) *and APT against samples generated from the true distribution. The simulations chosen for the analysis correspond to the median or maximum* (*"worst"*) *value of RMSE across* 20 *experiments.*

cost. Figure 5 shows that APT struggles with identifying some of the banana-shaped modes, whereas JAMS mixes well between all the modes and reflects well the shape of the true distribution. Due to Gaussian local proposal, both methods exhibit slightly lighter tails than the true distribution. Good between-mode mixing properties of JAMS are confirmed by the results gathered in Table 2.

5.3. *Sensor network localisation.* We consider here an example from [21], analysed later by [1, 25] and, in a modified version, by [39]. There are 11 sensors with locations $x_1, \dots, x_{11}$ scattered on a space $[0, 1]^2$. The locations of sensors $x_1, \dots, x_8$ are unknown; the remaining three locations are known. For any two sensors i and j, we observe the distance y_{ij} between them with probability $\exp(-\frac{\|x_i - x_j\|^2}{2\times 0.3^2})$. Once observed, the distance y_{ij} follows the normal distribution given by $y_{ij} \sim N(\|x_i - x_j\|, 0.02^2)$. Let w_{ij} be equal to 1 when y_{ij} is observed and 0 otherwise, and denote $y := \{y_{ij}\}$ and $w := \{w_{ij}\}$. The goal of the study is to make

TABLE 2
The lowest and the highest value (*across* 20 *runs of the experiment*) *of the acceptance rate of jumps from a given mode, in dimensions* 10 *and* 20, *for the mixture of banana-shaped and t-distributions*

	Deterministic		Gaussian		t-distributed	
	Lowest	Highest	Lowest	Highest	Lowest	Highest
$d = 10$	0.27	0.77	0.12	0.52	0.20	0.65
$d = 20$	0.20	0.75	0.09	0.35	0.11	0.48

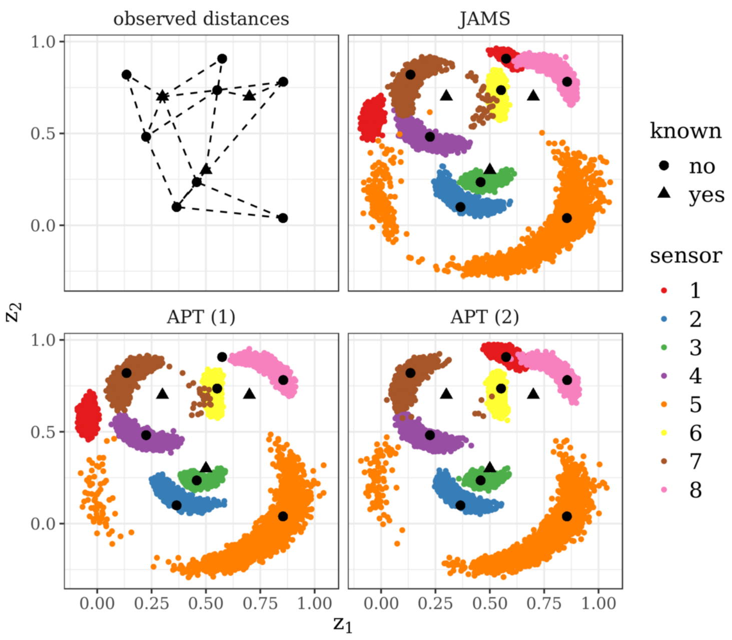

FIG. 6. *Black triangles and dots denote true locations of the sensors with known and unknown locations, respectively. Top left panel: dashed lines represent observed distances between sensors. Top right panel: posterior samples obtained using JAMS (with Gaussian jumps) for locations $x_1, \dots x_8$. Bottom panels: posterior samples using APT for two different starting points.*

inference about the unknown locations $x_i = (z_{i1}, z_{i2})$ for $i = 1, \dots 8$ given y and w. Following [1] and [25], we put a uniform prior on [0, 1] for each of the coordinates z_{i1} and z_{i2} for $i = 1, \dots 8$. The resulting posterior distribution is given by

$$\pi(x_1, \dots x_8 | y, w) \propto \prod_{\substack{j=2,\dots,11 \\ i=1,\dots,8 \\ i<j}} f_{ij}(x_i, x_j | y_{ij}, w_{ij}) \quad \text{where}$$

$$f_{ij}(x_i, x_j | y_{ij}, w_{ij}) = \begin{cases} \exp\left(-\frac{\|x_i - x_j\|^2}{2 \times 0.3^2}\right) \exp\left(-\frac{(y_{ij} - \|x_i - x_j\|)^2}{2 \times 0.02^2}\right) & \text{if } w_{ij} = 1, \\ 1 - \exp\left(-\frac{\|x_i - x_j\|^2}{2 \times 0.3^2}\right) & \text{otherwise.} \end{cases}$$

Since there are few observed distances with known locations (see top left panel of Figure 6), the model is nonidentifiable which results in multimodality of the posterior distribution. We ran JAMS on this example for 500,000 iterations of the main algorithm. This was preceded by 10,000 BFGS runs (with starting points for optimisation sampled from the prior) and covariance matrix estimation (between 7000 and 15,0000 iterations per mode). For parallel tempering we used 700,000 iterations (with a burn-in period of 200,000) and four temperatures. If JAMS is implemented on eight cores, this means that running an APT simulation is about twice as costly as running a JAMS one (see Supplementary Material B [32] for details).

Despite the fact that for all 20 APT experiments the acceptance rates at all temperature levels, as well as for between-temperature swaps, converged to the optimal acceptance rate 0.234 (see [8]), the behaviour of this algorithm was unstable. As shown in Figure 6, in case of APT the estimation of the location of sensor 1 depends on the starting point. In case of JAMS, both modes for x_1 (in red) are represented. Figure 7 illustrates stability of JAMS across all experiments and jump methods. In Supplementary Material B [32] we assign an even higher computational budget to adaptive parallel tempering allowing for five temperatures

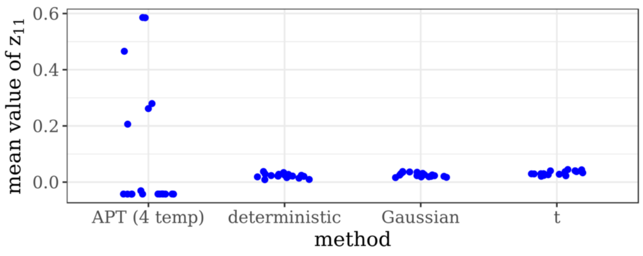

FIG. 7. *A boxplot of the mean value of the first coordinate of sensor* 1 *for* 20 *runs of the experiment for APT* (*with four temperature levels*) *and three versions of JAMS.*

and observe a substantial improvement in mixing and stability, but the results are still worse than those of JAMS.

6. Summary and discussion.

The framework we proposed here is based on three fundamental ideas. First, we split the task into mode finding and sampling from the target distribution. This allows the user to choose freely the best optimisation algorithm available. Second, we base our approach on local moves responsible for mixing within the same mode and jumps that move between the modes. Finally, we account for inhomogeneity between the modes by using different proposal distributions at each mode and adapting their parameters separately.

To develop a methodological approach and prove ergodic results for our algorithm, we introduced the Auxiliary Variable Adaptive MCMC class. There are other adaptive algorithms falling in this category; so, our theoretical results may potentially be useful beyond the scope of JAMS. We have shown that the Auxiliary Variable Adaptive MCMC methods enjoy robust ergodicity properties analogous to those of Adaptive MCMC.

We believe that our approach overcomes, to a large extent, deficiencies of tempering-based methods. Its important advantage is an inbuilt mechanism of remembering the locations of the modes identified, so far, by including them in the augmented target distributions. Hence, in contrast to the tempering-based methods, it does not need to "re-discover" a mode each time in order to move there, which is of particular importance when some basins of attraction are much smaller than the others. By analysing the spectral gap of a single JAMS kernel (with independent proposal jumps) and commenting on the rate of convergence of the estimation of covariance matrices during the burn-in algorithm, we have argued why, for a mixture of Gaussians with known locations of the modes, the required number of iterations should depend polynomially on the dimension. A very interesting research avenue, suggested by the reviewers, would be establishing formal results on the performance of our adaptive algorithm as a function of dimension, possibly also for a larger class of target distributions and the deterministic jump scheme.

Currently, the main bottleneck of the method is mode finding and, in particular, sampling starting points for optimisation runs in such a way that there is at least one point in the basin of attraction of each mode. As well as other optimisation-based algorithms, in cases when a mode is missed by optimisation runs, JAMS will inherit the properties of the standard Random Walk Metropolis algorithm on the region associated with this mode. A potential remedy to this issue could be replacing with certain probability standard RWM with Gaussian or t-distributed proposals with local samplers being Metropolis–Hastings algorithms, using a special design of the proposal distribution, which facilitates crossing low-probability barriers, for example, [39]. Improving the way of sampling the starting points for optimisation runs as well as investigating different strategies of exploring well the state space, even when some modes have been missed, is an area of future work.

Acknowledgments. We thank Louis Aslett, Ewan Cameron, Arnaud Doucet, Geoff Nicholls, Jeffrey Rosenthal, Radek Adamczak, George Deligiannidis, two anonymous referees and the Associate Editor for helpful comments, and Shiwei Lan for pointing us to the data for the sensor network example. We also thank Radu Craiu for providing the data set for the LOH example studied in Supplementary Material B [32].

E. Pompe is supported by the EPSRC and MRC Centre for Doctoral Training in Next Generation Statistical Science: the Oxford-Warwick Statistics Programme, EP/L016710/1, and the Clarendon Fund.

C. Holmes is supported by the MRC, the EPSRC, the Alan Turing Institute, Health Data Research UK, and the Li Ka Shing foundation.

K. Łatuszyński is supported by the Royal Society via the University Research Fellowship scheme.

SUPPLEMENTARY MATERIAL

Supplement to "A framework for adaptive MCMC targeting multimodal distributions" (DOI: 10.1214/19-AOS1916SUPP; .pdf). In Supplementary Material A we present the proofs of our theoretical results of Section 3 and Section 4. We discuss also the dependence of the performance of the algorithm on the dimension by analysing the spectral gap of an individual JAMS kernel, and comment on other algorithms in the Auxiliary Variable Adaptive MCMC class. Supplementary Material B contains details of the implementation of our method, an additional simulation example (a multimodal posterior distribution for a Bayesian hierarchical model) and settings of our numerical experiments.

REFERENCES

[1] AHN, S., CHEN, Y. and WELLING, M. (2013). Distributed and adaptive darting Monte Carlo through regenerations. In *Artificial Intelligence and Statistics* 108–116.

[2] ANDRICIOAEI, I., STRAUB, J. E. and VOTER, A. F. (2001). Smart darting Monte Carlo. *J. Chem. Phys.* **114** 6994–7000.

[3] ANDRIEU, C. and MOULINES, É. (2006). On the ergodicity properties of some adaptive MCMC algorithms. *Ann. Appl. Probab.* **16** 1462–1505. MR2260070 https://doi.org/10.1214/105051606000000286

[4] ANDRIEU, C. and ROBERTS, G. O. (2009). The pseudo-marginal approach for efficient Monte Carlo computations. *Ann. Statist.* **37** 697–725. MR2502648 https://doi.org/10.1214/07-AOS574

[5] ANDRIEU, C. and THOMS, J. (2008). A tutorial on adaptive MCMC. *Stat. Comput.* **18** 343–373. MR2461882 https://doi.org/10.1007/s11222-008-9110-y

[6] ANDRIEU, C. and VIHOLA, M. (2015). Convergence properties of pseudo-marginal Markov chain Monte Carlo algorithms. *Ann. Appl. Probab.* **25** 1030–1077. MR3313762 https://doi.org/10.1214/14-AAP1022

[7] ATCHADÉ, Y. and FORT, G. (2010). Limit theorems for some adaptive MCMC algorithms with subgeometric kernels. *Bernoulli* **16** 116–154. MR2648752 https://doi.org/10.3150/09-BEJ199

[8] ATCHADÉ, Y. F., ROBERTS, G. O. and ROSENTHAL, J. S. (2011). Towards optimal scaling of Metropolis-coupled Markov chain Monte Carlo. *Stat. Comput.* **21** 555–568. MR2826692 https://doi.org/10.1007/s11222-010-9192-1

[9] BAI, Y., ROBERTS, G. O. and ROSENTHAL, J. S. (2011). On the containment condition for adaptive Markov chain Monte Carlo algorithms. *Adv. Appl. Stat.* **21** 1–54. MR2849670

[10] BORNN, L., JACOB, P. E., DEL MORAL, P. and DOUCET, A. (2013). An adaptive interacting Wang–Landau algorithm for automatic density exploration. *J. Comput. Graph. Statist.* **22** 749–773. MR3173740 https://doi.org/10.1080/10618600.2012.723569

[11] CHIMISOV, C., ŁATUSZYŃSKI, K. and ROBERTS, G. (2018). Air Markov chain Monte Carlo. Preprint. Available at arXiv:1801.09309.

[12] CRAIU, R. V., ROSENTHAL, J. and YANG, C. (2009). Learn from thy neighbor: Parallel-chain and regional adaptive MCMC. *J. Amer. Statist. Assoc.* **104** 1454–1466. MR2750572 https://doi.org/10.1198/jasa.2009.tm08393

[13] DRTON, M. and RICHARDSON, T. S. (2004). Multimodality of the likelihood in the bivariate seemingly unrelated regressions model. *Biometrika* **91** 383–392. MR2081308 https://doi.org/10.1093/biomet/91.2.383

[14] FEROZ, F., HOBSON, M. P. and BRIDGES, M. (2009). MultiNest: An efficient and robust Bayesian inference tool for cosmology and particle physics. *Mon. Not. R. Astron. Soc.* **398** 1601–1614.

[15] FEROZ, F., HOBSON, M. P., CAMERON, E. and PETTITT, A. N. (2013). Importance nested sampling and the MultiNest algorithm. Preprint. Available at arXiv:1306.2144.

[16] FORT, G., MOULINES, E. and PRIOURET, P. (2011). Convergence of adaptive and interacting Markov chain Monte Carlo algorithms. *Ann. Statist.* **39** 3262–3289. MR3012408 https://doi.org/10.1214/11-AOS938

[17] GÅSEMYR, J. (2006). The spectrum of the independent Metropolis–Hastings algorithm. *J. Theoret. Probab.* **19** 152–165. MR2256484 https://doi.org/10.1007/s10959-006-0009-2

[18] GEYER, C. J. (1991). Markov chain Monte Carlo maximum likelihood.

[19] GREEN, P. J., ŁATUSZYŃSKI, K., PEREYRA, M. and ROBERT, C. P. (2015). Bayesian computation: A summary of the current state, and samples backwards and forwards. *Stat. Comput.* **25** 835–862. MR3360496 https://doi.org/10.1007/s11222-015-9574-5

[20] HAARIO, H., SAKSMAN, E. and TAMMINEN, J. (2001). An adaptive Metropolis algorithm. *Bernoulli* **7** 223–242. MR1828504 https://doi.org/10.2307/3318737

[21] IHLER, A. T., FISHER, J. W., MOSES, R. L. and WILLSKY, A. S. (2005). Nonparametric belief propagation for self-localization of sensor networks. *IEEE J. Sel. Areas Commun.* **23** 809–819.

[22] JARNER, S. F. and HANSEN, E. (2000). Geometric ergodicity of Metropolis algorithms. *Stochastic Process. Appl.* **85** 341–361. MR1731030 https://doi.org/10.1016/S0304-4149(99)00082-4

[23] JASRA, A., HOLMES, C. C. and STEPHENS, D. A. (2005). Markov chain Monte Carlo methods and the label switching problem in Bayesian mixture modeling. *Statist. Sci.* **20** 50–67. MR2182987 https://doi.org/10.1214/088342305000000016

[24] KOU, S. C., ZHOU, Q. and WONG, W. H. (2006). Equi-energy sampler with applications in statistical inference and statistical mechanics. *Ann. Statist.* **34** 1581–1619. MR2283711 https://doi.org/10.1214/009053606000000515

[25] LAN, S., STREETS, J. and SHAHBABA, B. (2014). Wormhole Hamiltonian Monte Carlo. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence* 1953–1959.

[26] MARINARI, E. and PARISI, G. (1992). Simulated tempering: A new Monte Carlo scheme. *Europhys. Lett.* **19** 451–458.

[27] MIASOJEDOW, B., MOULINES, E. and VIHOLA, M. (2013). An adaptive parallel tempering algorithm. *J. Comput. Graph. Statist.* **22** 649–664. MR3173735 https://doi.org/10.1080/10618600.2013.778779

[28] MYKLAND, P., TIERNEY, L. and YU, B. (1995). Regeneration in Markov chain samplers. *J. Amer. Statist. Assoc.* **90** 233–241. MR1325131

[29] NEAL, R. M. (1996). Sampling from multimodal distributions using tempered transitions. *Stat. Comput.* **6** 353–366.

[30] NOCEDAL, J. and WRIGHT, S. J. (2006). *Numerical Optimization*, 2nd ed. *Springer Series in Operations Research and Financial Engineering*. Springer, New York. MR2244940

[31] PAULIN, D., JASRA, A. and THIERY, A. (2019). Error bounds for sequential Monte Carlo samplers for multimodal distributions. *Bernoulli* **25** 310–340. MR3892321 https://doi.org/10.3150/17-bej988

[32] POMPE, E., HOLMES, C. and ŁATUSZYŃSKI, K. (2020). Supplement to "A framework for adaptive MCMC targeting multimodal distributions." https://doi.org/10.1214/19-AOS1916SUPP

[33] ROBERTS, G. O., GELMAN, A. and GILKS, W. R. (1997). Weak convergence and optimal scaling of random walk Metropolis algorithms. *Ann. Appl. Probab.* **7** 110–120. MR1428751 https://doi.org/10.1214/aoap/1034625254

[34] ROBERTS, G. O. and ROSENTHAL, J. S. (2001). Optimal scaling for various Metropolis–Hastings algorithms. *Statist. Sci.* **16** 351–367. MR1888450 https://doi.org/10.1214/ss/1015346320

[35] ROBERTS, G. O. and ROSENTHAL, J. S. (2007). Coupling and ergodicity of adaptive Markov chain Monte Carlo algorithms. *J. Appl. Probab.* **44** 458–475. MR2340211 https://doi.org/10.1239/jap/1183667414

[36] ROBERTS, G. O. and ROSENTHAL, J. S. (2009). Examples of adaptive MCMC. *J. Comput. Graph. Statist.* **18** 349–367. MR2749836 https://doi.org/10.1198/jcgs.2009.06134

[37] ROSENTHAL, J. S. (2011). Optimal proposal distributions and adaptive MCMC. In *Handbook of Markov Chain Monte Carlo. Chapman & Hall/CRC Handb. Mod. Stat. Methods* 93–111. CRC Press, Boca Raton, FL. MR2858446

[38] SMINCHISESCU, C. and WELLING, M. (2011). Generalized darting Monte Carlo. *Pattern Recognit.* **44** 2738–2748.

[39] TAK, H., MENG, X.-L. and VAN DYK, D. A. (2018). A repelling-attracting Metropolis algorithm for multimodality. *J. Comput. Graph. Statist.* **27** 479–490. MR3863751 https://doi.org/10.1080/10618600.2017.1415911

[40] TAWN, N. G., ROBERTS, G. O. and ROSENTHAL, J. S. (2020). Weight-preserving simulated tempering. *Stat. Comput.* **30** 27–41. MR4057469 https://doi.org/10.1007/s11222-019-09863-3

[41] TJELMELAND, H. and HEGSTAD, B. K. (2001). Mode jumping proposals in MCMC. *Scand. J. Stat.* **28** 205–223. MR1844357 https://doi.org/10.1111/1467-9469.00232

[42] VIHOLA, M. (2011). On the stability and ergodicity of adaptive scaling Metropolis algorithms. *Stochastic Process. Appl.* **121** 2839–2860. MR2844543 https://doi.org/10.1016/j.spa.2011.08.006

[43] WANG, F. and LANDAU, D. P. (2001). Determining the density of states for classical statistical models: A random walk algorithm to produce a flat histogram. *Phys. Rev. E* **64** Art. ID 056101.

[44] WANG, F. and LANDAU, D. P. (2001). Efficient, multiple-range random walk algorithm to calculate the density of states. *Phys. Rev. Lett.* **86** 2050–2053. https://doi.org/10.1103/PhysRevLett.86.2050

[45] WOODARD, D. B., SCHMIDLER, S. C. and HUBER, M. (2009). Sufficient conditions for torpid mixing of parallel and simulated tempering. *Electron. J. Probab.* **14** 780–804. MR2495560 https://doi.org/10.1214/EJP.v14-638

[46] ZHOU, Q. (2011). Multi-domain sampling with applications to structural inference of Bayesian networks. *J. Amer. Statist. Assoc.* **106** 1317–1330. MR2896838 https://doi.org/10.1198/jasa.2011.ap10346