A FRAMEWORK FOR ADAPTIVE MCMC TARGETING MULTIMODAL DISTRIBUTIONS

By Emilia Pompe∗,§, Chris Holmes†,§ and Krzysztof Łatuszyński‡,¶

University of Oxford§ and University of Warwick¶

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 $\pi$ 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 $X \times I$, where $X$ is the original state space of $\pi$ and $I$ 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

∗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.
†Supported by the MRC, the EPSRC, the Alan Turing Institute, Health Data Research UK, and the Li Ka Shing foundation.
‡Supported by the Royal Society via the University Research Fellowship scheme.
MSC 2010 subject classifications: Primary 60J05, 65C05; secondary 62F15
Keywords and phrases: multimodal distribution, adaptive MCMC, ergodicity
strongly on the good between-mode mixing of the Markov kernel used within the SMC algorithm (see [32]). 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, 25], astrophysics [14, 15, 39] and sensor network localisation [22].

Moreover, multimodality is an inherent issue of Bayesian mixture models (e.g. [24]), 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 \( \pi \) 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.

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 \( \pi \) 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 \( \pi \) are located, addressing challenge (1) above, and also to move between these regions, addressing challenge (2). The examples of such methods, which incorporate \( \pi^\beta \) by augmenting the state space, are parallel tempering proposed by [18] and its adaptive version [28], simulated tempering [27], tempered transitions [30] and the equi-energy sampler [25]. 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 non-overlapping $\epsilon$-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 [26] as an extension of Riemannian Manifold HMC [19]. 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 [44, 43] 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 multi-domain 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 $\pi$.

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 [41, 39].

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 [35, 5, 20]. 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, 34, 16, 7, 11].

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 which region the current state belongs to. Besides, adapting parameters of the local proposal distributions on the fly must be based on samples that actually come from the corresponding region.

The multi-domain 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 [26], 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 [29], 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 $X \times I$, where $I = \{1, \ldots, N\}$, 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 non-uniform 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 [34], as opposed to identifying regeneration times, which allows 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. 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. 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 $\pi$ 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 $\mathcal{I} := \{1, \ldots, N\}$ is a finite set of indices of the modes of $\pi$. We defer the discussion about finding the modes to Section 2.2. Here $\gamma$ 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

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

where $Q_i(\mu_i, \Sigma_{\gamma,i})$ is an elliptical distribution (such as the normal or the multivariate $t$ distribution) centred at $\mu_i$ 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 $\tilde{\pi}$,
E. POMPE, C. HOLMES AND K. ŁATUSZYSKI

respectively. Firstly, 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
\( \tilde{\pi}_\gamma(B \times I) = \pi(B) \). Hence, \( \pi \) 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 \( \pi \).

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 \( \epsilon \), respectively.

- **Local move**: Given the current state of the chain \((x, i)\) and the current
  parameter \( \gamma \), 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 \( \gamma \), 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 build
on its stability properties to establish ergodicity of JAMS in Section 4. The
acceptance probability formula for local moves is given by:

\[
\alpha_{\gamma,L} ((x, i) \rightarrow (y, i)) = \min \left[ 1, \frac{\tilde{\pi}_\gamma(y, i) \sum w_{\gamma,j} Q_j(\mu_j, \Sigma_{\gamma,j}(y))}{\tilde{\pi}_\gamma(x, i) \sum w_{\gamma,j} Q_j(\mu_j, \Sigma_{\gamma,j}(x))} \right].
\]

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 \( \mu_k \)
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 \( R_{\gamma,J,ik}(\cdot) \). For
independent proposal jumps, the acceptance probability is equal to

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

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

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

(2.2)
The required equality is satisfied for
\[
y := \mu_k + \Lambda_{\gamma,k} \Lambda_{\gamma,k}^{-1}(x - \mu_i),
\]
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 \textit{deterministic jumps}. The acceptance probability is then given by
\[
\alpha_{\gamma,j}(\,(x,i) \rightarrow (y,k)) = \min \left[ 1, \frac{\tilde{\pi}_\gamma(y,k) \ a_{\gamma,ki} \ \sqrt{\det \Sigma_{\gamma,k}}}{\tilde{\pi}_\gamma(x,i) \ a_{\gamma,ik} \ \sqrt{\det \Sigma_{\gamma,i}}} \right].
\]

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.

\begin{algorithm}
\caption{JAMS: main algorithm, iteration $n+1$}
\begin{algorithmic}[1]
\State \textbf{Input:} current point $(x_n,i_n)$, list of modes $\{\mu_1, \ldots, \mu_N\}$, constant $\epsilon \in (0,1)$, parameter $\gamma_n = \{\Sigma_{\gamma,i}, w_{\gamma,i}, a_{\gamma,i,k}\}_{i,k \in \{1, \ldots, N\}}$, empirical means $m_1, \ldots, m_N$ and covariance matrices $S_1, \ldots, S_N$.
\State Generate $u \sim U[0,1]$.
\State if $u > \epsilon$ then
\State \quad \textbf{Local move:}
\State \quad Propose a new value $y \sim R_{\gamma_n,L,i_n}(x_n, \cdot)$.
\State \quad Accept $y$ with probability $\alpha_{\gamma,L}(\,(x_n,i_n) \rightarrow (y,i_n))$.
\State \quad if $y$ accepted then
\State \quad \quad $(x_{n+1},i_{n+1}) := (y,i_n)$.
\State \quad else
\State \quad \quad $(x_{n+1},i_{n+1}) := (x_n,i_n)$.
\State \quad end if
\State else
\State \quad \textbf{Jump move:}
\State \quad Propose a new mode $k \sim (a_{\gamma_n,i,1}, \ldots, a_{\gamma_n,i,N})$.
\State \quad Propose a new value $y \sim R_{\gamma_n,J,i_k}(x_n, \cdot)$.
\State \quad Accept $(y,k)$ with probability $\alpha_{\gamma,J}(\,(x_n,i_n) \rightarrow (y,k))$.
\State \quad if $(y,k)$ accepted then
\State \quad \quad $(x_{n+1},i_{n+1}) := (y,k)$.
\State \quad else
\State \quad \quad $(x_{n+1},i_{n+1}) := (x_n,i_n)$.
\State \quad end if
\State end if
\State Update the empirical mean $m_{i_{n+1}}$ and covariance matrix $S_{i_{n+1}}$ by including $x_{n+1}$.
\State Update the parameter $\gamma_n$ to $\gamma_{n+1}$ according to Algorithm 2.
\State \textbf{return} New sample $(x_{n+1},i_{n+1})$, parameter $\gamma_{n+1}$, $m_{i_{n+1}}$ and $S_{i_{n+1}}$.
\end{algorithmic}
\end{algorithm}

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 $i, k \in \{1, \ldots, N\}$ 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.

**Algorithm 2 Upd ating 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 $n_1, \ldots, n_N$, auxiliary matrices $\tilde{\Sigma}_1, \ldots, \tilde{\Sigma}_N$, 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: $\tilde{\Sigma}_{i_{n+1}} := \exp \left( n_{i_{n+1}}^{-\alpha} (\alpha_{\text{opt}}) \right) \tilde{\Sigma}_{i_{n+1}}$.
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 $w_{\gamma_{n+1},i}$ and $a_{\gamma_{n+1},ik}$ to $w_{\gamma_{n+1},i}$ and $a_{\gamma_{n+1},ik}$ for $i, k = 1, \ldots, N$.
11: end if 
12: end if 

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. [21, 35]) 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, 36]. Since representing a covariance matrix in high dimensions reliably typically requires a large number of samples, we do not apply this method straight away. Instead, we perform adaptive scaling, aiming to achieve the optimal acceptance rate (typically fixed at 0.234; see [33, 36]) for local moves, until the number of samples observed in a given mode exceeds a pre-specified 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 pro-
posed point $y$ is very distant from the current mode $\mu_i$ and close to another mode $\mu_k$, 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.

2.2. Burn-in algorithm. Algorithm 1 takes mode locations $\{\mu_1, \ldots, \mu_N\}$ and initial values of the matrices $\{\Sigma_{\gamma,0,1}, \ldots, \Sigma_{\gamma,0,N}\}$ 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 $\{\Sigma_{\gamma,0,1}, \ldots, \Sigma_{\gamma,0,N}\}$ 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, e.g. 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.

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 $[L_1, U_1] \times \ldots \times [L_d, U_d]$, provided by the user. Note that if the domain of attraction of each mode overlaps with $[L_1, U_1] \times \ldots \times [L_d, U_d]$, 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 [31] is initiated from every starting point. The BFGS 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 $\pi$, 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 pre-specified 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 $\Sigma_{\gamma_0,1}, \ldots, \Sigma_{\gamma_0,N}$ 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, i.e., with $\epsilon = 0$, in parallel, starting from each of the modes $\mu_1, \ldots, \mu_N$. This implies that we run $N$ chains and each of them adapts only the matrix $\Sigma_i$ corresponding to the mode $\mu_i$ 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 $\mu_i$. 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 we propose a method of choosing the number of round automatically based on monitoring the inhomogeneity factor (see [35] 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

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

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 $\Gamma$ represents a random variable taking values in $(\mathcal{Y}, \mathcal{B}(\mathcal{Y}))$, denote its filtration as $\mathcal{G}_n := \sigma\{\tilde{X}_0, \ldots, \tilde{X}_n, \Gamma_0, \ldots, \Gamma_n\}$. The conditional distribution of $\Gamma_{n+1}$ 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

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

for $\tilde{x} \in \tilde{\mathcal{X}}, \gamma \in \mathcal{Y}, B \in \mathcal{B}(\tilde{\mathcal{X}})$. Note that depending on the adaptive update rule for $\Gamma_n$, 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 $\tilde{X}$-marginal of $\{(\tilde{X}_n, \Gamma_n)\}_{n=0}^\infty$ at time $n$, conditionally on the starting points, i.e.

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

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

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

**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 [28] and adaptive versions of pseudomarginal algorithms [4, 6]. A more detailed discussion on this may be found in Supplementary Material A.
3.1. Theoretical results for the class. The two main approaches to verifying ergodicity of Adaptive MCMC are based on martingale approximations [3, 16, 7] or coupling [34]. 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 non-uniform case, providing results analogous to Theorems 1 and 2 of [34].

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 $\tilde{X} = X \times \Phi$, following dynamics (3.2) with a family of transition kernels $\{\tilde{P}_\gamma\}_{\gamma \in \Upsilon}$ satisfying (3.1) and such that $\tilde{P}_\gamma$ is $\tilde{\pi}_\gamma$-ergodic for each $\gamma \in \Upsilon$. 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

$$\|\tilde{P}_\gamma^N(\tilde{x}, \cdot) - \tilde{\pi}_\gamma(\cdot)\|_{TV} \leq \varepsilon, \quad \text{for all} \quad \tilde{x} \in \tilde{X} \quad \text{and} \quad \gamma \in \Upsilon.$$

(b) (Diminishing Adaptation). The random variable

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

converges to 0 in probability.

In fact assumption (a) of Theorem 3.2 can be relaxed. To this end, define the $\varepsilon$–convergence time as

$$(3.4) \quad M_\varepsilon(\tilde{x}, \gamma) := \inf\{k \geq 1 : \|\tilde{P}_\gamma^k(\tilde{x}, \cdot) - \tilde{\pi}_\gamma(\cdot)\|_{TV} \leq \varepsilon\}.$$ 

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 non-uniform case.

**Theorem 3.3 (Ergodicity - non-uniform 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

$$(3.5) \quad \Pr\left(M_\varepsilon(\tilde{X}_n, \Gamma_n) > N \mid \tilde{X}_0 = \tilde{x}, \Gamma_0 = \gamma\right) \leq \tilde{\delta} \quad \text{for all} \quad n \in \mathbb{N}.$$
Then the algorithm is ergodic in the sense of Definition 3.1.

In our proof 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 non-uniform case. By letting \( \Phi \) be a singleton, our result applies to the standard Adaptive MCMC setting and extends the result of [34] 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 \( g : \mathcal{X} \rightarrow \mathbb{R} \) be a bounded measurable function. Then

\[
\frac{\sum_{i=1}^n g(X_i)}{n} \rightarrow \pi(g) \quad \text{in probability as } n \rightarrow \infty.
\]

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 [34, 9]. 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 \( \gamma \in \mathcal{Y} \) \( \| \tilde{P}_\gamma^k (\tilde{x}, \cdot) - \tilde{\pi}_\gamma (\cdot) \|_{TV} \rightarrow 0 \) as \( k \rightarrow \infty \).

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

\begin{equation}
\tilde{P}_\gamma V_{\tilde{\pi}_\gamma} (\tilde{x}) \leq \lambda V_{\tilde{\pi}_\gamma} (\tilde{x}) + b \quad \text{for all } \tilde{x} \in \tilde{\mathcal{X}} \text{ and } \gamma \in \mathcal{Y},
\end{equation}

where for \( \tilde{x} \in \tilde{\mathcal{X}} \)

\[
\tilde{P}_\gamma V_{\tilde{\pi}_\gamma} (\tilde{x}) := E \left( V_{\tilde{\pi}_\gamma} (\tilde{X}_{n+1}) | \tilde{X}_n = \tilde{x}, \Gamma_n = \gamma \right).
\]
Moreover, \( V_{\tilde{\pi}}(\tilde{x}) \) is bounded on compact sets as a function of \((\tilde{x}, \gamma)\).

c) There exist \( \delta > 0, v > 2n_0b/(1 - \lambda^{n_0}) \) 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 \( \nu_\gamma \) on \( \tilde{X} \) satisfying

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

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) \( EV_{\tilde{\pi}_{\Gamma_0}}(\tilde{X}_0) < \infty. \)

Then the Containment condition (3.5) holds.

Lemma 3.5 bears resemblance to Theorem 3 of [34]. 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 self-tuning 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 pre-specified 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 \( \kappa \). In order to ensure that the random variable \( D_n \) defined by (3.3) converges to 0 in probability, i.e. Diminishing Adaptation holds, the following modification is introduced. The updates happen at times \( N_j^* \), where

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

and \( n_k^* = n_k + \text{Uniform}[0, \lfloor k^\kappa^* \rfloor] \) for some \( \kappa^* \in (0, \kappa) \) 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 non-uniform 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 two-fold:

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 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

\[
JR_{\gamma,i} = \{ x \in \mathcal{X} : (x - \mu_i)^T \Sigma_{\gamma,i}^{-1} (x - \mu_i) \leq R \}
\]

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 \(\beta I_d\) to the covariance matrix at each step, they are also bounded from below. Consequently, there exist positive constants \(m\) and \(M\) for which

\[
mI_d \preceq \Sigma_{\gamma,i} \preceq MI_d \quad \text{for all } \gamma \in \mathcal{Y} \text{ and } i \in \mathcal{I}.
\]

As for the adaptive scheme for \(w_{\gamma,i}\) and \(a_{\gamma,ik}\), we only require that these values be bounded away from 0, i.e. there exist \(\epsilon_a\) and \(\epsilon_w\) such that

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

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 $\pi$ for all $i \in I$ and $\gamma \in \mathcal{Y}$, i.e.

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

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 $\pi$ and the proposal distributions $R_{\gamma,J,i}$ satisfies (4.4). Then mJAMS is ergodic.

When the tails of the distribution $\pi$ are heavier than 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 $i \in I$, $\gamma \in \mathcal{Y}$ the proposal distribution for local moves $R_{\gamma,L,i}$ follows an elliptical distribution parameterized by $\Sigma_{\gamma,i}$. Furthermore, the family of distributions $R_{\gamma,L,i}(0, \cdot)$, $\gamma \in \mathcal{Y}$, has uniformly bounded probability density functions, and for any compact set $C \subset \mathcal{X}$ we have

\[ \inf_{x,y \in C} \inf_{\gamma \in \mathcal{Y}} R_{\gamma,L,i}(x,y) > 0 \quad \text{for each } i \in I. \]  

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

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

c) The target distribution $\pi$ is super-exponential, i.e. it is positive with continuous first derivatives and satisfies

\[ \lim_{|x| \to \infty} \frac{x}{|x|} \cdot \nabla \log \pi(x) = -\infty. \]
d) Every $Q_i$, $i \in I$, is an elliptical distribution parametrised by $\Sigma_{\gamma,i}$ positive on $\mathcal{X}$ and additionally, the following condition is satisfied:

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

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

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

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

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

Then mJAMS is ergodic.

The most challenging part of the proof is verifying drift condition (3.6), which we do for $V_{\tilde{T}_{\gamma}}((x,i)) = c \tilde{T}_{\gamma}(x,i)^{-s}$, $c > 0$, $s \in (0, 1)$, separately for local and jump kernels. When proving the result for local moves, we show that $\limsup_{|x| \to \infty} \sup_{\gamma \in \mathcal{Y}} \frac{\tilde{E}_{\gamma,L,i}V_{\tilde{T}_{\gamma}}((x,i))}{V_{\tilde{T}_{\gamma}}((x,i))} < 1$, following the proof of Theorem 4.1 of [23]. 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 [23], to show that for large $x$ the value of the integral of $R_{\gamma,L,i}(x, y)$ min $\left[1, \frac{\tilde{T}_{\gamma}(y,i)}{\tilde{T}_{\gamma}(x,i)} \right] \frac{V_{\tilde{T}_{\gamma}}((y,i))}{V_{\tilde{T}_{\gamma}}((x,i))}$ with respect to $y$ is arbitrarily small on each set. The main difference between our approach and that of [23] of 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 $\tilde{T}_{\gamma}$ does not "deviate too much" from $\pi$. This condition can be easily verified if every $Q_i$, $i \in I$ follows the $t$ distribution with the same number of degrees of freedom.

For the jump kernels the idea is to show that $\sup_{\gamma \in \mathcal{Y}} \tilde{V}_{\tilde{T}_{\gamma}}((y,k))$ is "not too large" when the algorithm jumps to $(y,k)$. For deterministic jumps performed on jumping regions it follows from the boundedness of $\tilde{V}_{\tilde{T}_{\gamma}}((x,i))$ on
compact sets. For independent proposal jumps we show that under (4.10)
\sup_{\gamma \in \mathcal{Y}} \int_{\mathcal{X}} R_{\gamma, l, k}(y) V_{\mathcal{N}}((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 \( \gamma^\ast \in \mathcal{Y} \)
\begin{equation}(4.11) \limsup_{|x| \to \infty} \int_{r(x)} R_{\gamma^\ast, l, i}(x, y) \, dy < 1 \quad \text{for each } i \in \mathcal{I}.
\end{equation}
Then condition (4.6) holds.

**Lemma 4.4.** Consider mJAMS together with conditions a), c) and d) of
Theorem 4.2. Assume additionally that the target distribution \( \pi \) satisfies
\begin{equation}(4.12) \limsup_{|x| \to \infty} \frac{x}{|x|} \cdot \frac{\nabla \pi(x)}{|
\nabla \pi(x)|} < 0.
\end{equation}
Then condition (4.6) holds.

The following corollary shows that mJAMS in a standard setting is success-
ful at targeting mixtures of normal distributions.

**Corollary 4.5.** Let the target distribution \( \pi \) be given by
\[ \pi(x) \propto w_1 \exp(-p_1(x)) + \ldots + w_n \exp(-p_n(x)), \]
where \( w_i > 0 \) and \( p_i \) is a polynomial of order \( \geq 2 \) for each \( i = 1, \ldots, n \). If additionally \( Q_i \) for \( i \in \mathcal{I} \) follows the multivariate \( t \) distribution with the same
number of degrees of freedom, and \( R_{\gamma, l, i}(0, \cdot) \) 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 \( P(\tau < t) \to 1 \) as \( t \to \infty \), where \( \tau \) 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 (non-adaptive) MCMC methods this is typically measured by the rate at which the spectral gap of a transition kernel decreases. In Supplementary Material A we analyse the spectral gap of an individual JAMS kernel. To obtain a formal result, we consider the following target density:

\[
\pi(x) = \frac{1}{2} N \left( -\frac{(1,\ldots,1)}{d}, \sigma_1^2 I_d \right) + \frac{1}{2} N \left( (1,\ldots,1), \sigma_2^2 I_d \right).
\]

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 non-adaptive 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 fifteen 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 adap-
tive parallel tempering [28], 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 implemen-
tation, 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.
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 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 5 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 \(-\underbrace{1,\ldots,1}_{d} \in \mathbb{R}^d\). 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
Table 1

| d   | 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    |

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.

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 non-elliptical modes, we used a modified and more challenging version of a classic example introduced in [25] (a mixture of 20 bivariate Gaussian distributions), studied also later by [28] 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 7 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 ten 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 7,000 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 5 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 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
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).

...method significantly outperforms APT with a smaller computational 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 [22], analysed later by [1], [26] and, in a modified version by [39]. There are 11 sensors with locations \(x_1, \ldots, x_{11}\) scattered on a space \([0, 1]^2\). The locations of sensors \(x_1, \ldots, x_8\) 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 \(\exp\left(-\frac{\|x_i - x_j\|^2}{2 \times 0.3^2}\right)\). 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 \(w := \{w_{ij}\}\).
Fig 4: Boxplots of the values of RMSE/√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.

|         | deterministic | Gaussian | t-distributed |
|---------|---------------|----------|---------------|
| d=10    | Lowest 0.27   | Highest 0.77 | Lowest 0.12   | Highest 0.52   | Lowest 0.20   | Highest 0.65   |
|         |               |          |               |               |               |
| d=20    | Lowest 0.20   | Highest 0.75 | Lowest 0.00   | Highest 0.35   | Lowest 0.11   | Highest 0.48   |

**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.

The goal of the study is to make inference about the unknown locations $x_i = (z_{i1}, z_{i2})$ for $i = 1, \ldots, 8$ given $y$ and $w$. Following [1] and [26] we put a uniform prior on $[0, 1]$ for each of the coordinates $z_{i1}$ and $z_{i2}$ for $i = 1, \ldots, 8$. The resulting posterior distribution is given by

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

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 non-identifiable which results in multimodal-
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.

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 we assign an even higher computational budget to adaptive parallel tempering allowing for 5 temperatures 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. Firstly, 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. Secondly, 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, e.g. [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.

Acknowledgements. 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.

SUPPLEMENTARY MATERIAL

Supplementary Material A and B: (supplementary_material.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, pages 108–116.
[2] Andricioaei, I., Straub, J. E., and Voter, A. F. (2001). Smart darting Monte Carlo. The Journal of Chemical Physics, 114(16):6994–7000.
[3] Andrieu, C. and Moulines, E. (2006). On the ergodicity properties of some adaptive MCMC algorithms. Ann. Appl. Probab., 16(3):1462–1505.
[4] Andrieu, C. and Roberts, G. O. (2009). The pseudo-marginal approach for efficient Monte Carlo computations. The Annals of Statistics, 37(2):697–725.
[5] Andrieu, C. and Thomas, J. (2008). A tutorial on adaptive mcmc. Statistics and computing, 18(4):343–373.
[6] Andrieu, C. and Vihola, M. (2015). Convergence properties of pseudo-marginal Markov chain Monte Carlo algorithms. The Annals of Applied Probability, 25(2):1030–1077.
[7] Atchadé, Y. and Fort, G. (2010). Limit theorems for some adaptive MCMC algorithms with subgeometric kernels. Bernoulli, 16(1):116–154.
[8] Atchadé, Y. F., Roberts, G. O., and Rosenthal, J. S. (2011). Towards optimal scaling of Metropolis-coupled Markov chain Monte Carlo. Statistics and Computing, 21(4):555–568.
[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):1–54.
[10] Bornn, L., Jacob, P. E., Del Moral, P., and Doucet, A. (2013). An adaptive interacting Wang–Landau algorithm for automatic density exploration. *Journal of Computational and Graphical Statistics*, 22(3):749–773.

[11] Chimiròv, C., Latuszyński, K., and Roberts, G. (2018). Air Markov Chain Monte Carlo. *arXiv preprint arXiv:1801.09309*.

[12] Craiu, R., Rosenthal, J., and Yang, C. (2009). Learn from thy neighbor: parallel-chain and regional adaptive MCMC. *Journal of the American Statistical Association*, 104(488):1454–1466.

[13] Drton, M. and Richardson, T. S. (2004). Multimodality of the likelihood in the bivariate seemingly unrelated regressions model. *Biometrika*, 91(2):383–392.

[14] Feroz, F., Hobson, M., and Bridges, M. (2009). MultiNest: an efficient and robust Bayesian inference tool for cosmology and particle physics. *Monthly Notices of the Royal Astronomical Society*, 398(4):1601–1614.

[15] Feroz, F., Hobson, M., Cameron, E., and Pettitt, A. (2013). Importance nested sampling and the MultiNest algorithm. *arXiv preprint 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(6):3262–3289.

[17] Găsemyr, J. (2006). The spectrum of the independent Metropolis–Hastings algorithm. *Journal of Theoretical Probability*, 19(1):152–165.

[18] Geyer, C. J. (1991). Markov chain Monte Carlo maximum likelihood. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 53(2):123–214.

[19] Green, P. J., Latuszyński, K., Pereyra, M., and Robert, C. P. (2015). Bayesian computation: a summary of the current state, and samples backwards and forwards. *Stat. Comput.*, 25(4):835–862.

[20] Haario, H., Saksman, E., and Tamminen, J. (2001). An adaptive Metropolis algorithm. *Bernoulli*, 7(2):223–242.

[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 Journal on Selected Areas in Communications*, 23(4):809–819.

[22] Jasra, A., Holmes, C. C., and Stephens, D. A. (2005). Markov chain Monte Carlo methods and the label switching problem in Bayesian mixture modeling. *Statistical Science*, pages 50–67.

[23] Kou, S., Zhou, Q., and Wong, W. H. (2006). Discussion paper equi-energy sampler with applications in statistical inference and statistical mechanics. *The Annals of Statistics*, pages 1581–1619.

[24] Lan, S., Streets, J., and Shahbaba, B. (2014). Wormhole Hamiltonian Monte Carlo. In *AAAI*, pages 1933–1939.

[25] Marinari, E. and Parisi, G. (1992). Simulated tempering: a new Monte Carlo scheme. *EPL (Europhysics Letters)*, 19(6):451.

[26] Miasojedow, B., Moulines, E., and Vihola, M. (2013). An adaptive parallel tempering algorithm. *Journal of Computational and Graphical Statistics*, 22(3):649–664.

[27] Mykland, P., Tierney, L., and Yu, B. (1995). Regeneration in Markov chain samplers. *Journal of the American Statistical Association*, 90(429):233–241.

[28] Neal, R. M. (1996). Sampling from multimodal distributions using tempered transitions. *Statistics and computing*, 6(4):353–366.

[29] Nocedal, J. and Wright, S. J. (2006). Numerical optimization: Springer series in
operations research and financial engineering. *Springer, New York*

[32] Paulin, D., Jasra, A., and Thiery, A. (2019). Error bounds for sequential Monte Carlo samplers for multimodal distributions. *Bernoulli*, 25(1):310–340.

[33] Roberts, G., Gelman, A., and Gilks, W. (1997). Weak convergence and optimal scaling of random walk Metropolis algorithms. *The Annals of Applied Probability*, 7(1):110–120.

[34] Roberts, G. and Rosenthal, J. (2007). Coupling and ergodicity of adaptive Markov chain Monte Carlo algorithms. *Journal of Applied Probability*, 44(2):458.

[35] Roberts, G. and Rosenthal, J. (2009). Examples of adaptive MCMC. *Journal of Computational and Graphical Statistics*, 18(2):349–367.

[36] Roberts, G. O. and Rosenthal, J. S. (2001). Optimal scaling for various Metropolis-Hastings algorithms. *Statistical science*, 16(4):351–367.

[37] Rosenthal, J. S. (2011). Optimal proposal distributions and adaptive MCMC. *Handbook of Markov Chain Monte Carlo*, 4(10.1201).

[38] Smichidescu, C. and Welling, M. (2011). Generalized darting Monte Carlo. *Pattern Recognition*, 44(10):2738–2748.

[39] Tak, H., Meng, X.-L., and van Dyk, D. A. (2017). A Repelling-Attracting Metropolis Algorithm for Multimodality. *Journal of Computational and Graphical Statistics*.

[40] Tawn, N. G., Roberts, G. O., and Rosenthal, J. S. (2018). Weight-preserving simulated tempering. *Statistics and Computing*, pages 1–15.

[41] Tjelmeland, H. and Hegstad, B. (2001). Mode jumping proposals in MCMC. *Scandinavian Journal of Statistics*, 28(1):205–223.

[42] Vihola, M. (2011). On the stability and ergodicity of adaptive scaling Metropolis algorithms. *Stochastic Processes and their Applications*, 121(12):2839–2860.

[43] Wang, F. and Landau, D. (2001a). Determining the density of states for classical statistical models: A random walk algorithm to produce a flat histogram. *Physical Review E*, 64(5):056101.

[44] Wang, F. and Landau, D. (2001b). Efficient, multiple-range random walk algorithm to calculate the density of states. *Physical review letters*, 86(10):2050.

[45] Woodard, D., Schmidler, S., and Huber, M. (2009). Sufficient conditions for torpid mixing of parallel and simulated tempering. *Electronic Journal of Probability*, 14:780–804.

[46] Zhou, Q. (2011). Multi-Domain Sampling With Applications to Structural Inference of Bayesian Networks. *Journal of the American Statistical Association*, 106(496):1317–1330.