Statistics and Computing (2021) 31:61
https://doi.org/10.1007/s11222-021-10038-2


# Ensemble slice sampling

## Parallel, black-box and gradient-free inference for correlated & multimodal distributions

**Minas Karamanis**[1] · **Florian Beutler**[1]

Received: 16 November 2020 / Accepted: 28 July 2021 / Published online: 15 August 2021


**Abstract**
Slice sampling has emerged as a powerful Markov Chain Monte Carlo algorithm that adapts to the characteristics of the target distribution with minimal hand-tuning. However, Slice Sampling's performance is highly sensitive to the user-specified initial length scale hyperparameter and the method generally struggles with poorly scaled or strongly correlated distributions. This paper introduces Ensemble Slice Sampling (ESS), a new class of algorithms that bypasses such difficulties by adaptively tuning the initial length scale and utilising an ensemble of parallel walkers in order to efficiently handle strong correlations between parameters. These affine-invariant algorithms are trivial to construct, require no hand-tuning, and can easily be implemented in parallel computing environments. Empirical tests show that Ensemble Slice Sampling can improve efficiency by more than an order of magnitude compared to conventional MCMC methods on a broad range of highly correlated target distributions. In cases of strongly multimodal target distributions, Ensemble Slice Sampling can sample efficiently even in high dimensions. We argue that the parallel, black-box and gradient-free nature of the method renders it ideal for use in scientific fields such as physics, astrophysics and cosmology which are dominated by a wide variety of computationally expensive and non-differentiable models.

**Keywords** Markov Chain Monte Carlo · Bayesian inference · Slice sampling · Adaptive Monte Carlo · Probabilistic data analysis

## 1 Introduction

*Bayesian inference and data analysis* has become an integral part of modern science. This is partly due to the ability of Markov Chain Monte Carlo (MCMC) algorithms to generate samples from intractable probability distributions. MCMC methods produce a sequence of samples, called a *Markov chain*, that has the target distribution as its equilibrium distribution. The more samples are included, the more closely the distribution of the samples approaches the target distribution. The Markov chain can then be used to numerically approximate expectation values (e.g. parameter uncertainties, marginalised distributions).

✉ Minas Karamanis
minas.karamanis@ed.ac.uk

Florian Beutler
florian.beutler@ed.ac.uk

[1] Institute for Astronomy, University of Edinburgh, Royal Observatory, Blackford Hill, Edinburgh EH9 3HJ, UK

Common MCMC methods entail a significant amount of time spent hand-tuning the hyperparameters of the algorithm to optimise its efficiency with respect to a target distribution. The emerging and routine use of such mathematical tools in science calls for the development of black-box MCMC algorithms that require no hand-tuning at all. This need led to the development of adaptive MCMC methods like the Adaptive Metropolis algorithm (Haario et al. 2001) which tunes its proposal scale based on the sample covariance matrix. Unfortunately, most of those algorithms still include a significant number of hyperparameters (e.g. components of the covariance matrix) rendering the adaptation noisy. Furthermore, the tuning is usually performed on the basis of prior knowledge, such as one or more long preliminary runs which further slow down the sampling. Last but not least, there is no reason to believe that a single Metropolis proposal scale is optimal for the whole distribution (i.e. the appropriate scale could vary from one part of the distribution to another). Another approach to deal with those issues would be to develop methods that by construction require no or minimal hand-tuning.

An archetypal such method is the Slice Sampler (Neal 2003), which has only one hyperparameter, the initial length scale.

It should be noted that powerful adaptive methods that require no hand-tuning (although they do require preliminary runs) already exist. Most notable of them is the No U-Turn Sampler (NUTS) (Hoffman and Gelman 2014), an adaptive extension of Hamiltonian Monte Carlo (HMC) (Neal et al. 2011). However, such methods rely on the gradient of the log probability density function. This requirement is the reason why these methods are limited in their application in quantitative fields such as physics, astrophysics and cosmology, which are dominated by computationally expensive non-differentiable models. Thus, our objective in this paper is to introduce a parallel, black-box and gradient-free method that can be used in the aforementioned scientific fields.

This paper presents Ensemble Slice Sampling (ESS), an extension of the Standard Slice Sampling method. ESS naturally inherits most of the benefits of Standard Slice Sampling, such as the acceptance rate of 1, and most importantly the ability to adapt to the characteristics of a target distribution without any hand-tuning at all. Furthermore, we will show that ESS's performance is insensitive to linear correlations between the parameters, thus enabling efficient sampling even in highly demanding scenarios. We will also demonstrate ESS's performance in strongly multimodal target distributions and show that the method samples efficiently even in high dimensions. Finally, the method can easily be implemented in parallel taking advantage of multiple CPUs thus facilitating Bayesian inference in cases of computationally expensive models.

Our implementation of ESS is inspired by Tran and Ninness (2015). However, our method improves upon that by extending the direction choices (e.g. Gaussian and global move), adaptively tuning the initial proposal scale and parallelising the algorithm. Nishihara et al. (2014) developed a general algorithm based on the elliptical slice sampling method (Murray et al. 2010) and a Gaussian Mixture approximation to the target distribution. ESS utilises an ensemble of parallel and interacting chains, called walkers. Other methods that are based on the ensemble paradigm include the Affine-Invariant Ensemble Sampler (Goodman and Weare 2010) and the Differential Evolution MCMC (Ter Braak 2006) along with its various extensions (ter Braak and Vrugt 2008; Vrugt et al. 2009), as well as more recent approaches that are based on Langevin diffusion dynamics (Garbuno-Inigo et al. 2020a, b) and the time discretisation of stochastic differential equations (Leimkuhler et al. 2018) in order to achieve substantial speedups.

In Sect. 2, we will briefly discuss the Standard Slice Sampling algorithm. In Sect. 3, we will introduce the Ensemble Slice Sampling method. In Sect. 4 we will investigate the empirical evaluation of the algorithm. We reserve Sects. 5 and 6 for discussion and conclusion, respectively.

## 2 Standard Slice Sampling

*Slice Sampling* is based on the idea that sampling from a distribution $p(x)$ whose density is proportional to $f(x)$ is equivalent to uniformly sampling from the region underneath the graph of $f(x)$. More formally, in the univariate case, we introduce an auxiliary variable, the height $y$, thus defining the joint distribution $p(x, y)$, which is uniform over the region $U = \{(x, y) : 0 < y < f(x)\}$. To sample from the marginal density for $x$, $p(x)$, we sample from $p(x, y)$ and then we ignore the $y$ values.

Generating samples from $p(x, y)$ is not trivial, so we might consider defining a Markov chain that will converge to that distribution. The simplest, in principle, way to construct such a Markov chain is via Gibbs sampling. Given the current $x$, we sample $y$ from the conditional distribution of $y$ given $x$, which is uniform over the range $(0, f(x))$. Then, we sample the new $x$ from the *slice* $S = \{x : y < f(x)\}$.

Generating a sample from the slice $S$ may still be difficult, since we generally do not know the exact form of $S$. In that case, we can update $x$ based on a procedure that leaves the uniform distribution of $S$ invariant. Neal (2003) proposed the following method:

Given the current state $x_0$, the next one is generated as:

1. Draw $y_0$ uniformly from $(0, f(x_0))$, thus defining the horizontal slice $S = \{x : y_0 < f(x)\}$,
2. Find an interval $I = (L, R)$ that contains all, or much, of $S$ (e.g. using the stepping-out procedure defined bellow),
3. Draw the new point $x_1$ uniformly from $I \cap S$.

In order to find the interval $I$, Neal (2003) proposed to use the *stepping-out* procedure that works by randomly positioning an interval of length $\mu$ around the point $x_0$ and then expanding it in steps of size $\mu$ until both ends are outside of the slice. The new point $x_1$ is found using the *shrinking* procedure, in which points are uniformly sampled from $I$ until a point inside $S$ is found. Points outside $S$ are used to shrink the interval $I$. The stepping-out and shrinking procedures are illustrated in Fig. 1. By construction, the stepping-out and shrinking procedures can adaptively tune a poor estimate of the length scale $\mu$ of the initial interval. The length scale $\mu$ is the only free hyperparameter of the algorithm. For a detailed review of the method we direct the reader to Neal (2003) and MacKay (2003) (also Exercise 30.12 in that text).

It is important to mention here that for multimodal distributions, there is no guarantee that the slice would cross any of the other modes, especially if the length scale is underestimated initially. Ideally, in order to provide a large enough initial value of the scale factor $\mu$, prior knowledge of the distance between the modes is required. As we will show in the next section, Ensemble Slice Sampling does not suffer

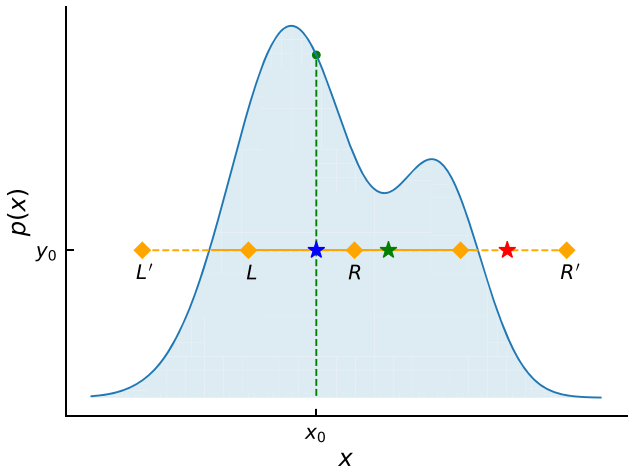


**Fig. 1** The plot shows the univariate slice sampling method. Given an initial value $x_0$, a value $y_0$ is uniformly sampled along the vertical slice $(0, f(x_0))$ (green dashed line) thus defining the initial point (blue star). An interval $(L, R)$ is randomly positioned horizontally around the initial point, and then it is expanded in steps of size $\mu = R - L$ until both of its ends $L'$, $R'$ are outside the slice. The new point (green star) is generated by repeatedly sampling uniformly from the expanded interval $(L', R')$ until a point is found inside the slice. Points outside the slice (e.g. the red star) are used to shrink the interval $(L', R')$ by moving $L'$ or in this case $R'$ to that point and accelerate the sampling procedure. (Color figure online)

from this complication and can handle strongly multimodal distributions efficiently.

## 3 Ensemble Slice Sampling

The univariate slice sampling scheme can be used to sample from multivariate distributions by sampling repeatedly along each coordinate axis in turn (one parameter at a time) or by sampling along randomly selected directions (MacKay 2003). Using either of those choices, the Standard Slice Sampler performs acceptably in cases with no strong correlations in parameter space. The overall performance of the algorithm generally depends on the number of expansions and contractions during the stepping-out and shrinking procedures, respectively. Ideally, we would like to minimise that number. A reasonable initial estimate of the length scale is still required in order to reduce the amount of time spent expanding or contracting the initial interval.

However, when strong correlations are present two issues arise. First, there is no single value of the initial length scale that minimises the computational cost of the stepping-out and shrinking procedures along all directions in parameter space. The second problem concerns the choice of direction. In particular, neither the component-wise choice (one parameter at a time) nor the random choice is suitable in strongly correlated cases. Using such choices results in highly autocorrelated samples.

Our approach would be to target each of those two issues individually. The resulting algorithm, Ensemble Slice Sampling (ESS), is invariant under affine transformations of the parameter space, meaning that its performance is not sensitive to linear correlations. Furthermore, ESS minimises the computational cost of finding the slice by adaptively tuning the initial length scale. Last but not least, unlike most MCMC methods, ESS is trivially parallelisable, thus enabling the data analyst to take advantage of modern high-performance computing facilities with multiple CPUs.

### 3.1 Adaptively tuning the length scale

Let us first consider the effect of the initial length scale on the performance of the univariate slice sampling method. For instance, if the initial length scale is $\lambda$ times smaller than the actual size of the slice, then the stepping-out procedure would require $\mathcal{O}(\lambda)$ steps in order to fix this. However, in this case, since the final interval is an accurate approximation of the slice there would probably be no contractions during the shrinking phase. On the other hand, when the initial length scale is larger than the actual slice then the number of expansions would be either one or zero. In this case though, there would be a number of contractions.

#### 3.1.1 Stochastic approximation

As the task is to minimise the total number of expansions and contractions, we employ and adapt the *Robbins–Monro* inspired stochastic approximation algorithm (Robbins and Monro 1951) of Tibbits et al. (2014). Ideally, based on the reasoning of the previous paragraph, only one expansion and one contraction will take place. Therefore, the target ratio of number of expansions to total number of expansions and contractions is $1/2$. To achieve this, we update the length scale $\mu$ based on the following recursive formula:

$$\mu^{(t+1)} = 2\mu^{(t)} \frac{N_{\mathrm{e}}^{(t)}}{N_{\mathrm{e}}^{(t)} + N_{\mathrm{c}}^{(t)}}, \tag{1}$$

where $N_{\mathrm{e}}^{(t)}$ and $N_{\mathrm{c}}^{(t)}$ are the number of expansions and contractions during iteration $t$. It is easy to see that when the fraction $N_{\mathrm{e}}^{(t)}/(N_{\mathrm{e}}^{(t)} + N_{\mathrm{c}}^{(t)})$ is larger than $1/2$ the length scale $\mu$ will be increased. In the case where the fraction is smaller than $1/2$ the length scale $\mu$ will be decreased accordingly. The optimization can stop either when the fraction is close to $1/2$ within a threshold or when a maximum number of tuning steps has been completed. The pseudocode for the first case is shown in Algorithm 1. In order to preserve detailed balance, it is important to be sure that the adaptation stops after

a finite number of iterations. In practice, this happens after $\mathcal{O}(10)$ iterations. An alternative would be to use diminishing adaptation (Roberts and Rosenthal 2007) but we found that our method is sufficient in practice (see Sect. 4.3 for more details).

**Algorithm 1** Function to tune the length scale $\mu$.

1: **function** TuneLengthScale($t$, $\mu^{(t)}$, $N_e^{(t)}$, $N_c^{(t)}$, $M^{\text{adapt}}$)
2: **if** $t \leq M^{\text{adapt}}$ **then**
3: Compute $\mu^{(t+1)}$ using Equation 1,
4: **return** $\mu^{(t+1)}$
5: **else**
6: **return** $\mu^{(t)}$
7: **end if**

## 3.2 The choice of direction & parallelization

In cases where the parameters are correlated, we can accelerate mixing by moving more frequently along certain directions in parameter space. One way of achieving this is to exploit some prior knowledge about the covariance of the target distribution. However, such an approach would either require significant hand-tuning or noisy estimations of the sample covariance matrix during an initial run of the sampler. For that reason, we employ a different approach to exploit the covariance structure of the target distribution and preserve the hand-tuning-free nature of the algorithm.

### 3.2.1 Ensemble of walkers

Following the example of Goodman and Weare (2010), we define an ensemble of parallel chains, called walkers. In our case though, each walker is an individual slice sampler. The sampling proceeds by moving one walker at a time by slice sampling along a direction defined by a subset of the rest of walkers of the ensemble. As long as the aforementioned direction does not depend on the position of the current walker, the resulting algorithm preserves the detailed balance of the chain. Moreover, assuming that the distribution of the walkers resembles the correlated target distribution, the chosen direction will prefer directions of correlated parameters.

We define an ensemble of $N$ parallel walkers as the collection $S = \{\mathbf{X_1}, \ldots, \mathbf{X_N}\}$. The position of each individual walker $\mathbf{X_k}$ is a vector $\mathbf{X_k} \in \mathbb{R}^D$, and therefore, we can think of the ensemble $S$ as being in $\mathbb{R}^{ND}$. Assuming that each walker is drawn independently from the target distribution with density $p$, then the target distribution for the ensemble would be the product

$$P(\mathbf{X_1}, \ldots, \mathbf{X_N}) = \prod_{k=1}^{N} p(\mathbf{X_k})\,. \tag{2}$$

The Markov chain of the ensemble would preserve the product density of Eq. 2 without the individual walker trajectories being Markov. Indeed, the position of $\mathbf{X_k}$ at iteration $t+1$ can depend on $\mathbf{X_j}$ at iteration $t$ with $j \neq k$.

Given the walker $\mathbf{X_k}$ that is to be updated there are arbitrary many ways off defining a direction vector from the complementary ensemble $S_{[k]} = \{\mathbf{X_j},\ \forall j \neq k\}$. Here, we will discuss a few of them. Following the convention in the ensemble MCMC literature, we call those recipes of defining direction vectors, *moves*. Although the use of the ensemble might seem equivalent to that of a sample covariance matrix in the Adaptive Metropolis algorithm (Haario et al. 2001), the first has a higher information content as it encodes both linear and nonlinear correlations. Indeed, having an ensemble of walkers allows for arbitrary many policies for choosing the appropriate directions along which the walkers move in parameter space. As we will shortly see, one of the choices (i.e. the Gaussian move, introduced later in this section) is indeed the slice sampling analogue of a covariance matrix. However, other choices (i.e. differential move or Global move) can take advantage of the non-Gaussian nature of the ensemble distribution and thus propose more informative moves. As it will be discussed later in this section, those advanced moves make no assumption of Gaussianity for the target distribution. Furthermore, as we will show in the last part of this section, the ensemble can also be easily parallelised.

**Algorithm 2** Function to return a differential move direction vector.

1: **function** DifferentialMove($k$, $\mu$, $S$)
2: Draw two walkers $\mathbf{X_l}$, and $\mathbf{X_m}$ uniformly and without replacement from the complementary ensemble $S$,
3: Compute direction vector $\boldsymbol{\eta}_k$ using Equation 6,
4: **return** $\boldsymbol{\eta}_k$

### 3.2.2 Affine transformations and invariance

Affine invariance is a property of certain MCMC samplers first introduced in the MCMC literature by Goodman and Weare 2010. An MCMC algorithm is said to be affine invariant if its performance is invariant under the bijective mapping $g : \mathbb{R}^D \to \mathbb{R}^D$ of the form $\mathbf{Y} = A\mathbf{X} + b$ where $A \in \mathbb{R}^{D\times D}$ is a matrix and $b \in \mathbb{R}^D$ is a vector. Linear transformations of this form are called affine transformations and describe rotations, rescaling along specific axes as well as translations in parameter space. Assuming that $\mathbf{X}$ has the probability density $p(\mathbf{X})$, then $\mathbf{Y} = A\mathbf{X} + b$ has the probability density

$$p_{A,b}(\mathbf{Y}) = p(A\mathbf{X} + b) \propto p(\mathbf{X})\,. \tag{3}$$

Given a density $p$ as well as an MCMC transition operator $\mathcal{T}$ such that $\mathbf{X}(t+1) = \mathcal{T}\big(\mathbf{X}(t); p\big)$ for any iteration $t$ we call the operator $\mathcal{T}$ affine invariant if

$$\mathcal{T}\big(A\mathbf{X} + b; p_{A,b}\big) = A\,\mathcal{T}\big(\mathbf{X}; p\big) + b \tag{4}$$

for $\forall A \in \mathbb{R}^{D\times D}$ and $\forall b \in \mathbb{R}^{D}$. In case of an ensemble of walkers, we define an affine transformation from $\mathbb{R}^{ND}$ to $\mathbb{R}^{ND}$ as

$$S = \{\mathbf{X_1}, \ldots, \mathbf{X_N}\} \xrightarrow{A,b} \{A\mathbf{X_1} + b, \ldots, A\mathbf{X_N} + b\}\,. \tag{5}$$

The property of affine invariance is of paramount importance for the development of efficient MCMC methods. As we have discussed already, proposing samples more frequently along certain directions can accelerate sampling by moving further away in parameter space. Given that most realistic applications are highly skewed or anisotropic and are characterised by some degree of correlation between their parameters, affine-invariant methods are an obvious choice of a tool that can be used in order to achieve high levels of efficiency.

#### 3.2.3 Differential move

The differential direction choice works by moving the walker $\mathbf{X}_k$ based on two randomly chosen walkers $\mathbf{X}_l$ and $\mathbf{X}_m$ of the complementary ensemble $S_{[k]} = \{\mathbf{X_j},\ \forall j \neq k\}$ (Gilks et al. 1994), see Fig. 2 for a graphical explanation. In particular, we move the walker $\mathbf{X}_k$ by slice sampling along the vector $\boldsymbol{\eta}_k$ defined by the difference between the walkers $\mathbf{X}_l$ and $\mathbf{X}_m$. It is important to notice here that the vector $\boldsymbol{\eta}_k$ is not a unit vector and thus carries information about both the length scale and the optimal direction of movement. It will also prove to be more intuitive to include the initial length scale $\mu$ in the definition of the direction vector in the following way:

$$\boldsymbol{\eta}_k = \mu\big(\mathbf{X}_l - \mathbf{X}_m\big)\,. \tag{6}$$

The pseudocode for a function that, given the value of $\mu$ and the complementary ensemble $S$, returns a differential direction vector $\boldsymbol{\eta}_k$ is shown in Algorithm 2. Furthermore, the differential move is clearly affine invariant. Assuming that the distribution of the ensemble of walkers follows the target distribution and the latter is highly elongated or stretched along a certain direction then the proposed direction given by Eq. 6 will share the same directional asymmetry.

#### 3.2.4 Gaussian move

The direction vector $\boldsymbol{\eta}_k$ can also be drawn from a normal distribution with the zero mean and the covariance matrix equal to the sample covariance of the complementary ensemble $S_{[k]}$,

$$\mathbf{C}_S = \frac{1}{|S|}\sum_{j\in S}\big(\mathbf{X}_j - \bar{\mathbf{X}}_S\big)\big(\mathbf{X}_j - \bar{\mathbf{X}}_S\big)^t\,. \tag{7}$$

We chose to include the initial length scale $\mu$ in this definition as well:

$$\frac{\boldsymbol{\eta}_k}{2\mu} \sim \mathcal{N}\big(\mathbf{0}, \mathbf{C}_S\big)\,. \tag{8}$$

The factor of 2 is used so that the magnitude of the direction vectors are consistent with those sampled using the differential direction choice in the case of Gaussian-distributed walkers.

The pseudocode for a function that, given the value of $\mu$ and the complementary ensemble $S$, returns a Gaussian direction vector $\boldsymbol{\eta}_k$ is shown in Algorithm 3. See Fig. 3 for a graphical explanation of the method. Moreover, just like the differential move, the Gaussian move is also affine invariant. In the limit in which the number of walkers is very large and the target distribution is normal, the first reduces to the second. Alternatively, assuming that the distribution of walkers follows the target distribution then the covariance matrix of the ensemble would be the same as that of independently drawn samples from the target density. Therefore, any anisotropy characterising the target density would also be present in the distribution of proposed directions given by Eq. 8.

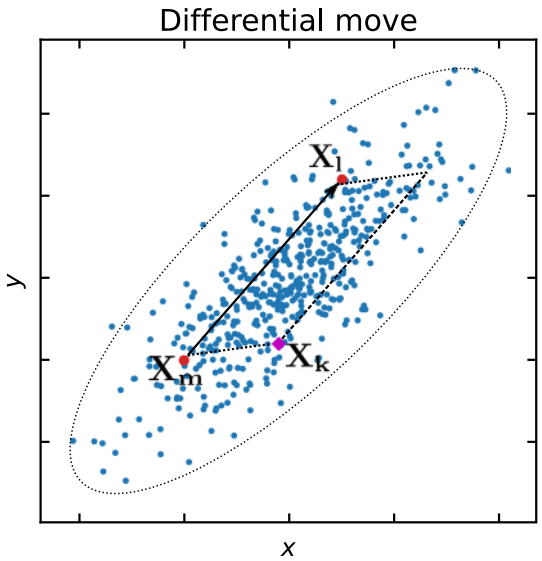


**Fig. 2** The plot shows the differential direction move. Two walkers (red) are uniformly sampled from the complementary ensemble (blue). Their positions define the direction vector (solid black). The selected walker (magenta) then moves by slice sampling along the parallel direction (dashed black). (Color figure online)

**Algorithm 3** Function to return a Gaussian Move direction vector.

1: **function** GaussianMove($k$, $\mu$, $S$)
2: Estimate sample covariance $\mathbf{C}_S$ of the walkers in the complementary ensemble $S$ using Equation 7,
3: Sample $\boldsymbol{\eta}_k/(2\mu) \sim \mathcal{N}(\mathbf{0}, \mathbf{C}_S)$,
4: **return** $\boldsymbol{\eta}_k$

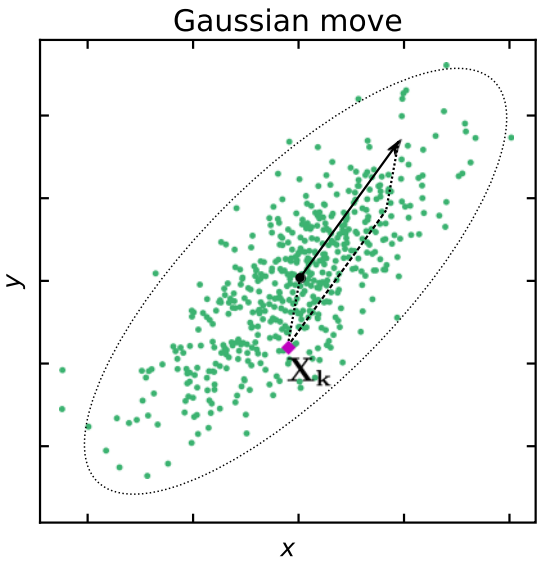


**Fig. 3** The plot shows the Gaussian direction move. A direction vector (solid black) is sampled from the Gaussian-approximated distribution of the walkers of the complementary ensemble (green). The selected walker (magenta) then moves by slice sampling along the parallel direction (dashed black). (Color figure online)

#### 3.2.5 Global move

ESS and its variations described so far (i.e. differential move, Gaussian move) have as much difficulty traversing the low probability regions between modes/peaks in multimodal distributions as most local MCMC methods (e.g. Metropolis, Hamiltonian Monte Carlo, slice sampling, etc.). Indeed, multimodal distributions are often the most challenging cases to sample from. Fortunately, Ensemble Slice Sampling's flexibility allows to construct advanced moves which are specifically designed to handle multimodal cases even in moderate to high-dimensional parameter spaces. The *global move* is such an example.

We first fit a *Gaussian Mixture* to the distribution of the walkers of the complementary ensemble $S_{[k]}$ using *Variational Inference*. To avoid defining the number of components of the Gaussian Mixture, we use a *Dirichlet process* as the prior distribution for the Gaussian Mixture weights[1] (Görür and Rasmussen 2010). The exact details of the construction of the Dirchlet process Gaussian Mixture (DPGM) are beyond the scope of this work and we direct the reader to Görür and Rasmussen (2010) and Bishop (2006) for more details. One of the major benefits of fitting the DPGM using variational inference compared to the expectation–maximisation (EM) algorithm (Dempster et al. 1977) that is often used is the improved stability. In particular, the use of priors in the variational Bayesian treatment guarantees that Gaussian components do not collapse into specific data points. This regularisation due to the priors leads to component covariance matrices that do not diverge even when the number of data points (i.e. walkers in our case) in a component is lower than the number of dimensions. In our case, this means that even if the number of walkers located in a mode of the target distribution is small DPGM would still identify that mode correctly. In such cases, the covariance of the component that corresponds to that mode would be over-estimated. This however does not affect the performance of the Global move as the latter does not rely on exact estimates of the component covariance matrices.[2]

In practice, we recommend using more than the minimum number of walkers in cases of multimodal distributions (e.g. at least two times as many in bimodal cases). We found that the computational overhead introduced by the variational fitting of the DPGM is negligible compared to the computational cost of the evaluation of the model and posterior distribution in common problems in physics, astrophysics and cosmology. Indeed, the cost is comparable, and only a few times higher than the Differential or Gaussian move. The reason for that is the relatively small number of walkers (i.e. $\mathcal{O}(10 - 10^3)$) that simplifies the fitting procedure.

Once fitting is done, we have a list of the means and covariance matrices of the components of the Gaussian Mixture. As the ensemble of walkers traces the structure of the target distribution, we can use the knowledge of the means and covariance matrices of the Gaussian Mixture to construct efficient direction vectors. Ideally, we prefer direction vectors that connect different modes. This way, the walkers will be encouraged to move along those directions that would otherwise be very unlikely to be chosen.

We uniformly select two walkers of the complementary ensemble and identify the Gaussian components to which they belong, say $i$ and $j$. There are two distinct cases, and we will treat them as such. In case A, $i = j$, meaning that the selected walkers originate from the same component. In case B, $i \neq j$, meaning that the two walkers belong to different components and thus probably different peaks of the target distribution.

As we will show next, only in case B, we can define a direction vector that favours mode-jumping behaviour. In case A,

[1] To this end, we use the Scikit-Learn implementation of the Dirichlet process Gaussian Mixture.

[2] Indeed, the covariance matrix of a component only enters through Eq. 10, but then it is rescaled by the factor $\gamma$.

we can sample a direction vector from the Gaussian component that the two select walkers belong to:[3]

$$\frac{\boldsymbol{\eta}_k}{2\mu} \sim \mathcal{N}\left(\mathbf{0}, \boldsymbol{C}_{i=j}\right), \tag{9}$$

where $\boldsymbol{C}_{i=j}$ is the covariance matrix of the $\mathrm{i_{th}}$ (or equivalently $\mathrm{j_{th}}$) component. Just as in the Gaussian move, the mean of the proposal distribution is zero so that we can interpret $\boldsymbol{\eta}$ as a direction vector.

In case B, where the two selected walkers belong to different components, $i \neq j$, we will follow a different procedure to facilitate long jumps in parameter space. We will sample two vectors, one from each component:

$$\boldsymbol{\eta}_{k,n} \sim \mathcal{N}\left(\boldsymbol{\mu}_n, \gamma \boldsymbol{C}_n\right), \tag{10}$$

for $n = i$ or $n = j$. Here, $\boldsymbol{\mu}_n$ is the mean of the nth component and $\boldsymbol{C}_n$ is its covariance matrix. In practise, we also rescale the covariance by a factor of $\gamma = 0.001$, which results in direction vectors with lower variance in their orientation. $\gamma < 1$ ensures that the chosen direction vector is close to the vector connecting the two peaks of the distribution. Finally, the direction vector will be defined as:

$$\boldsymbol{\eta}_k = 2\left(\boldsymbol{\eta}_{k,i} - \boldsymbol{\eta}_{k,j}\right). \tag{11}$$

The factor of 2 here is chosen to better facilitate mode-jumping. There is also no factor of $\mu$ in the aforementioned expression, since in this case there is no need for the scale factor to be tuned.

The pseudocode for a function that, given the complementary ensemble $S$, returns a Global direction vector $\boldsymbol{\eta}_k$ is shown in Algorithm 4. See Fig. 4 for a graphical explanation of the method. It should be noted that for the global move to work at least one walker needs to be present on each well separated mode.

Here, we introduced three general and distinct moves that can be used in a broad range of cases. In general, the global move requires a higher number of walkers than the differential or Gaussian move in order to perform well. We found that the differential and Gaussian moves are good choices for most target distributions, whereas the global move is only necessary in highly dimensional and multimodal cases. One can use the information in the complementary ensemble to construct more moves tailor-made for specific problems. Such additional moves might include kernel density estimation or clustering methods and as long as the information used comes from the complementary ensemble (and not from the walker that would be updated) the detailed balance is preserved.

[3] In practice, we use uniformly sample two walkers from the list of walkers that DPGM identified in that mode. This step removes any dependency on covariance matrix estimates.

**Algorithm 4** Function to return a global move direction vector.

1: **function** GlobalMove($k$, $\mu$, $S$)
2: Fit Dirichlet process Gaussian Mixture (DPGM) to the complementary ensemble $S_{[k]}$,
3: If $N$ is the number of components of the DPGM then select two components $i$, $j$ uniformly such that $i \neq j$,
4: **if** $i = j$ **then**
5: Sample $\boldsymbol{\eta}_k/(2\mu) \sim \mathcal{N}\left(\mathbf{0}, \boldsymbol{C}_{i=j}\right)$,
6: **else**
7: Sample $\boldsymbol{\eta}_{k,n} \sim \mathcal{N}\left(\boldsymbol{\mu}_n, \gamma \boldsymbol{C}_n\right)$ for $n = i, j$,
8: Compute direction vector $\boldsymbol{\eta}_k$ using Equation 11,
9: **end if**
10: **return** $\boldsymbol{\eta}_k$

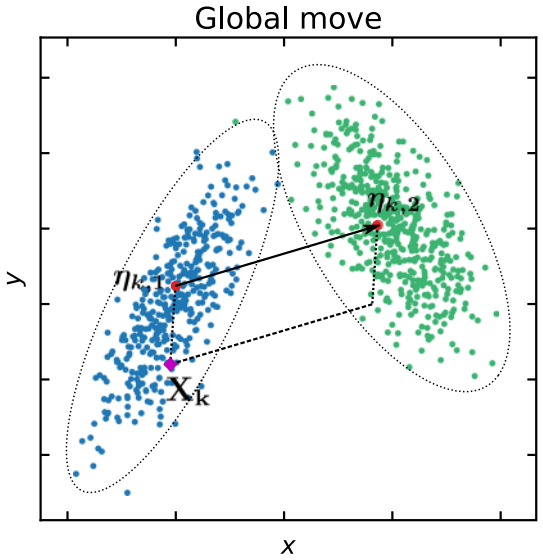


**Fig. 4** The plot shows the global direction move assuming that the uniformly selected pair of walkers of the complementary ensemble belongs to different components (blue and green). A position (red) is sampled from each component (using the rescaled by $\gamma$ covariance matrix). Those two points (red) define the direction vector (black) connecting the two modes (blue and green). The selected walker (magenta) then moves by slice sampling along the parallel direction (dashed). (Color figure online)

### 3.2.6 Parallelising the ensemble

Instead of evolving the ensemble by moving each walker in turn, we can do this in parallel. A naive implementation of this would result in a subtle violation of detailed balance. We can avoid this by splitting the ensemble into two sets of walkers (Foreman-Mackey et al. 2013) of $n_{\mathrm{Walkers}}/2$ each. We can now update the positions of all the walkers in the one set in parallel along directions defined by the walkers of the other set (the complementary ensemble). Then we can perform the same procedure for the other set. In accordance

with Eq. 2, the stationary distribution of the split ensemble would be

$$P(\mathbf{X_1}, \ldots, \mathbf{X_N}) = \prod_{k=1}^{N/2} p(\mathbf{X_k}) \prod_{k=1+N/2}^{N} p(\mathbf{X_k})\,. \tag{12}$$

The method generates samples from the target distribution by simulating a Markov chain which leaves this product distribution invariant. The transition operator $\mathcal{T}_1$ that updates the walkers of the first set (i.e. $k = 1, \ldots, N/2$) uses the walkers of the complementary ensemble (i.e. $k = 1 + N/2, \ldots, N$) and vice versa for the transition operator $\mathcal{T}_2$ that acts on the second set. In the context of ESS, the aforementioned transition operators correspond to a single iteration of Algorithm 5 coupled with one of the moves (e.g. differential move).

It follows from the ensemble splitting technique that the maximum number of CPUs used without any of them being idle is equal to the total number of walkers updated concurrently, that is $n_{\text{Walkers}}/2$. We will also verify this empirically in Sect. 4. Of course, this does not mean that if there are more CPUs available they cannot be used as we can always increase the size of the ensemble to match the available CPUs.

Combining this technique with the stochastic approximation solution of Sect. 3.1 and the choices (moves) of direction and ensemble-splitting technique of this subsection leads to the Ensemble Slice Sampling method of Algorithm 5.[4] Of course, another move (e.g. Gaussian, global) can be used instead of the differential move in Algorithm 5. Finally, the minimum number of walkers used should be twice the number of parameters. Using fewer walkers than that could lead to erroneous sampling from a lower-dimensional parameter space (Ter Braak 2006).

In general, parallelising a slice sampler is not trivial (e.g. as it is for Metropolis) because each update requires an unknown number of probability density evaluations. However, because of the affine invariance (i.e. performance unaffected by linear correlations) induced by the existence of the ensemble, all iterations require on average the same number of probability density evaluations (i.e. usually 5 if the stochastic approximation for the length scale $\mu$ is used). Therefore, the parallelization of Ensemble Slice Sampling is very effective in practice. Furthermore, the benefit of having parallel walkers instead of parallel independent chains (e.g. such as in Metropolis sampling) is clear, the walkers share information about the covariance structure of the distribution thus accelerating mixing.

[4] Perhaps a small detail, but we have included the length scale in the definition of the direction vector $\eta$, and therefore, it does not appear in the definition of the $(L, R)$ interval.

**Algorithm 5** Single Iteration $t$ of Ensemble Slice Sampling.

Given $t$, $f$, $\mu^{(t)}$, $S_{[0]}$, $S_{[1]}$, $M^{\text{adapt}}$:
Initialise $N_e^{(t)} = 0$ and $N_c^{(t)} = 0$,
**for** $i = 0, 1$ **do**
  **for** $k = 1, ..., N/2$ **do**
    $k \leftarrow k + iN/2$
    Compute direction vector $\boldsymbol{\eta}_k \leftarrow$ DifferentialMove($k$, $\mu^{(t)}$, $S_{[i]}$)
    Sample $Y \sim$ Uniform(0, $f(\mathbf{X_k}^{(t)})$)
    Sample $U \sim$ Uniform(0, 1)
    Set $L \leftarrow -U$, and $R \leftarrow L + 1$
    **while** $Y < f(L)$ **do**
      $L \leftarrow L - 1$
      $N_e^{(t)} \leftarrow N_e^{(t)} + 1$
    **end while**
    **while** $Y < f(R)$ **do**
      $R \leftarrow R + 1$
      $N_e^{(t)} \leftarrow N_e^{(t)} + 1$
    **end while**
    **while** True **do**
      Sample $X' \sim$ Uniform($L$, $R$)
      Set $Y' \leftarrow f(X'\boldsymbol{\eta}_k + \mathbf{X_k}^{(t)})$
      **if** $Y < Y'$ **then**
        **break**
      **end if**
      **if** $X' < 0$ **then**
        $L \leftarrow X'$
        $N_c^{(t)} \leftarrow N_c^{(t)} + 1$
      **else**
        $R \leftarrow X'$
        $N_c^{(t)} \leftarrow N_c^{(t)} + 1$
      **end if**
    **end while**
    Set $\mathbf{X_k}^{(t+1)} \leftarrow X'\boldsymbol{\eta}_k + \mathbf{X_k}^{(t)}$
  **end for**
**end for**
$\mu^{(t+1)} \leftarrow$ TuneLengthScale($t$, $\mu^{(t)}$, $N_e^{(t)}$, $N_c^{(t)}$, $M^{adapt}$),

## 4 Empirical evaluation

To empirically evaluate the sampling performance of the Ensemble Slice Sampling algorithm, we perform a series of tests. In particular, we compare its ability to sample from two demanding target distributions, namely the *autoregressive process of order 1* and the *correlated funnel*, against the Metropolis and Standard Slice Sampling algorithms. The Metropolis' proposal scale was tuned to achieve the optimal acceptance rate, whereas the initial length scale of Standard Slice Sampling was tuned using the stochastic scheme of Algorithm 1. Ensemble Slice Sampling significantly outperforms both of them. These tests help establish the characteristics and advantages of Ensemble Slice Sampling. Since our objective was to develop a gradient-free black-box method we then proceed to compare Ensemble Slice Sampling with a list of gradient-free ensemble methods such as *Affine Invariant Ensemble Sampling* (AIES), *Differential Evolution Markov Chain* (DEMC) and *Kernel Density Estimate Metropolis* (KM) on a variety of challenging target

distributions. Moreover, we are also interested in assessing the convergence rate of the length scale $\mu$ during the first iterations as well as the parallel scaling of the method in the presence of multiple CPUs. Unless otherwise specified, we use the differential move for the tests. Unlike ESS that has an acceptance rate of 1, AIES's and DEMC's acceptance rate is related to the number of walkers. For that reason, and for the sake of a fair comparison, we made sure the selected number of walkers in all examples would yield the optimal acceptance rate for AIES and DEMC. As we will discuss further in Sect. 5, it makes sense to increase the number of walkers in cases of multimodal distributions or strong nonlinear correlations. In general though, we recommend using the minimum number of walkers (i.e. twice the number of dimensions) as the default choice and increase it only if it is required by a specific application. For more rules and heuristics about the initialisation and number of walkers, we direct the interested reader to Sect. 5.

## 4.1 Performance tests

**Autoregressive process of order 1:** In order to investigate the performance of ESS. In high-dimensional and correlated scenarios, we chose a highly correlated Gaussian as the target distribution. More specifically, the target density is a discrete-time *autoregressive process of order 1*, also known as AR(1). This particular target density is ideally suited for benchmarking MCMC algorithms since the posterior density in many scientific studies often approximates a correlated Gaussian. Apart from that, the AR(1) is commonly used as a prior for time-series analysis.

The AR(1) distribution of a random vector $\boldsymbol{X} = (X_1, ..., X_N)$ is defined recursively as follows:

$$\begin{aligned} X_1 &\sim \mathcal{N}(0, 1)\,, \\ X_2|X_1 &\sim \mathcal{N}(\alpha X_1, \beta^2)\,, \\ &\vdots \\ X_N|X_{N-1} &\sim \mathcal{N}(\alpha X_{N-1}, \beta^2)\,, \end{aligned} \tag{13}$$

where the parameter $\alpha$ controls the degree of correlation between parameters and we chose it to be $\alpha = 0.95$. We set $\beta = \sqrt{1-\alpha^2}$ so that the marginal distribution of all parameters is $\mathcal{N}(0, 1)$. We also set the number of dimensions to $N = 50$.

For each method, we measured the mean *integrated autocorrelation time* (IAT), and the number of effective samples per evaluation of the probability density function, also termed *efficiency* (see "Appendix A" for details). For this test we ran the samplers for $10^7$ iterations. In this example, we used the minimum number of walkers (i.e. 100 walkers) for ESS and the equivalent number of probability evaluations for Metropolis and Slice Sampling with each walker initialised at a position sampled from the distribution $\mathcal{N}(0, 1)$. The results are presented in Table 1. The chain produced by Ensemble Slice Sampling has a significantly shorter IAT (20–40 times) compared to either of the other two methods. Furthermore, Ensemble Slice Sampling, with either Differential or Gaussian move, generates an order of magnitude greater number of independent samples per evaluation of the probability density. In this example, the Differential and Gaussian moves have achieved almost identical IAT values and efficiencies.

**Table 1** The table shows a comparison of the optimally tuned Metropolis, Standard Slice, and Ensemble Slice Sampling with the differential move (ESS-D) and the Gaussian move (ESS-G), respectively, in terms of the integrated autocorrelation time (IAT) and the number of effective samples per evaluation of the probability density (efficiency) multiplied by $10^4$. These metrics are formally defined in "Appendix A". The target distributions are the 50-dimensional autoregressive process of order 1 and the 25-dimensional correlated funnel distribution. The total number of iterations was set to $10^7$

| | Metropolis | Slice | ESS-D | ESS-G |
|---|---|---|---|---|
| Autoregressive process of order 1 | | | | |
| IAT | 4341 | 2075 | 111 | 107 |
| Efficiency | 2.3 | 1.0 | 17.5 | 17.8 |
| Correlated funnel distribution | | | | |
| IAT | – | 3905 | 129 | 141 |
| Efficiency | – | 0.5 | 15.3 | 14.0 |

To assess the mixing rate of Ensemble Slice Sampling, we set the maximum number of probability density evaluations to $3 \times 10^5$ and show the results in Fig. 5. We compare the results of Ensemble Slice Sampling with those obtained via the optimally tuned Metropolis and Standard Slice Sampling methods. Ensemble Slice Sampling significantly outperforms both of them, being the only one with a chain resembling the target distribution in the chosen number of probability evaluations.

### 4.1.1 Correlated funnel

The second test involves a more challenging distribution, namely the correlated funnel distribution adapted from Neal (2003). The funnel, tornado like, structure is common in Bayesian hierarchical models and possesses characteristics that render it a particularly difficult case. The main difficulty originates from the fact that there is a region of the parameter space where the volume of the region is low but the probability density is high, and another region where the opposite holds.

Suppose we want to sample an N-dimensional vector $\boldsymbol{X} = (X_1, ..., X_N)$ from the correlated funnel distribution. The marginal distribution of $X_1$ is Gaussian with mean zero and unit variance. Conditional on a value of $X_1$, the vector

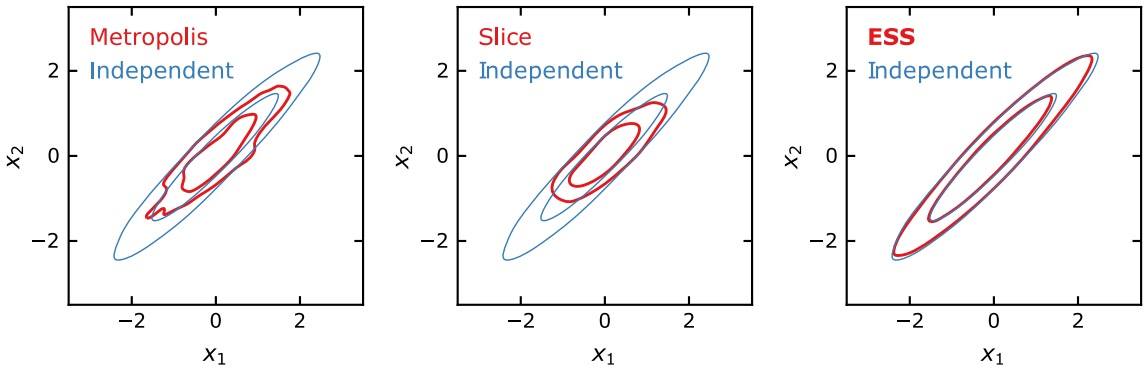


**Fig. 5** The plots compare the 1-sigma and 2-sigma contours generated by the optimised random-walk Metropolis (left), Standard Slice (centre) and Ensemble Slice Sampling (right) methods to those obtained by Independent Sampling (blue) for the AR(1) distribution. All samplers used the same number of probability density evaluations, $3 \times 10^5$. Only the first two dimensions are shown here. (Color figure online)

$\boldsymbol{X}_{2-N} = (X_2, ..., X_N)$ is drawn from a Gaussian with mean zero and a covariance matrix in which the diagonal elements are $\exp(X_1)$, and the non-diagonal equal to $\gamma \exp(X_1)$. If $\gamma = 0$, the parameters $X_2$ to $X_N$ conditional on $X_1$ are independent and the funnel distribution resembles the one proposed by Neal (2003). The value of $\gamma$ controls the degree of correlation between those parameters. When $\gamma = 0$ the parameters are uncorrelated. For the following test, we chose this to be $\gamma = 0.95$. We set the number of parameters $N$ to 25.

Using $10^7$ iterations, we estimated the IAT and the efficiency of the algorithms for this distribution as shown in Table 1. Just like in the AR(1) case, we used the minimum number (i.e. 50) of walkers for ESS with each walker initialised at a position sampled from the distribution $\mathcal{N}(0, 1)$. Since the optimally-tuned Metropolis fails to sample from this particular distribution, we do not quote any results. The Metropolis sampler is unable to successfully explore the region of parameter space with negative $X_1$ values. The presence of strong correlations renders the Ensemble Slice Sampler 30 times more efficient than the Standard Slice Sampling algorithm on this particular example. In this example, the differential move outperforms the Gaussian move in terms of efficiency, albeit by a small margin. In general, we expect the former to be more flexible than the latter since it makes no assumption about the Gaussianity of the target-distribution and recommend it as the default configuration of the algorithm.

To assess the mixing rate of the algorithm on this demanding case, we set the maximum number of evaluations of the probability density function to $3 \times 10^5$. As shown in Fig. 6, the Ensemble Slice Sampling is the only algorithm out of the three whose outcome closely resembles the target distribution. The results of Metropolis were incorrect for both, the limited run with $3 \times 10^5$ iterations and the long run with $10^7$ iterations. In particular, the chain produced using the Metropolis method resemble a converged chain but in fact it is biased in favour of positive values of $x_1$. The problem arises because of the vanishing low probability of accepting a point with highly negative value of $x_1$. This indicates the inability of Metropolis to handle this challenging case. For a more detailed discussion of this problem, we direct the reader to Section 8 of Neal (2003). In general, the correlated funnel is a clear example of a distribution in which a single Metropolis proposal scale is not sufficient for all the sampled regions of parameter space. The locally adaptive nature of ESS solves this issue.

### 4.2 Comparison to other ensemble methods

So far, we have demonstrated Ensemble Slice Sampling's performance in simple, yet challenging, target distributions. The tests performed so far demonstrate ESS's capacity to sample efficiently from highly correlated distributions compared with standard methods such as Metropolis and Slice Sampling. Although the use of Metropolis and Slice Sampling is common, these methods are not considered to be state-of-the-art. For this reason, we will now compare ESS with state-of-the-art gradient-free ensemble MCMC methods.

By far, the two most popular choices[5] of gradient-free ensemble methods are the Affine-Invariant Ensemble Sampling (AIES) (Goodman and Weare 2010) method and the Differential Evolution Monte Carlo (DEMC) (Ter Braak 2006) algorithm supplemented with a Snooker update (ter Braak and Vrugt 2008).

[5] For instance, in the fields of Astrophysics and Cosmology where most models are not differentiable and gradient methods (e.g. Hamiltonian Monte Carlo or NUTS) are not applicable the default choice is the Affine-Invariant Ensemble Sampler (AIES) (Goodman and Weare 2010) as implemented in emcee.

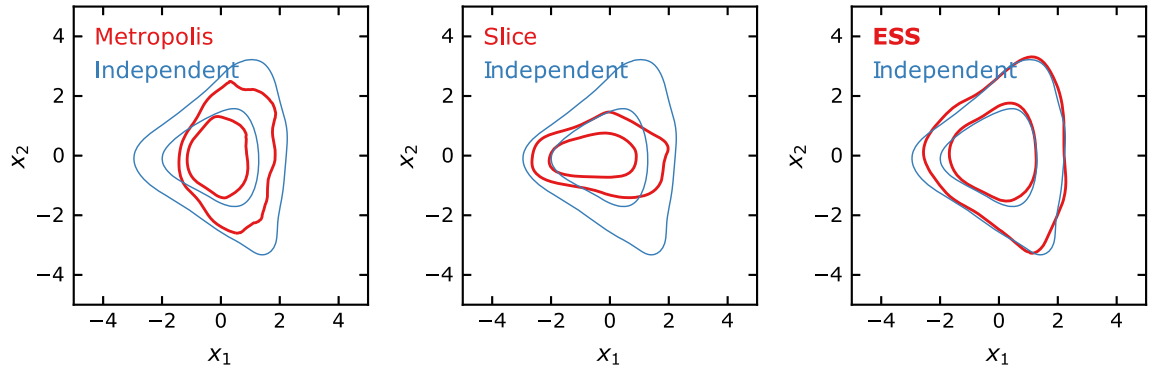


**Fig. 6** The plots compare the 1-sigma and 2-sigma contours generated by the optimised random-walk Metropolis (left), Standard Slice (centre) and Ensemble Slice Sampling (right) methods to those obtained by Independent Sampling (blue) for the correlated funnel distribution. All samplers used the same number of probability density evaluations, $3 \times 10^5$. Only the first two dimensions are shown here. (Color figure online)

In cases of strongly multimodal target distributions, we will also test our method against Sequential Monte Carlo[6] (SMC) (Liu and Chen 1998; Del Moral et al. 2006) and Kernel Density Estimate Metropolis (KM) (Farr and Farr 2015) which are particle methods specifically designed to handle strongly multimodal densities.

#### 4.2.1 Ring distribution

Although all three of the compared methods (i.e. ESS, AIES and DEMC) are affine invariant and thus unaffected by linear correlations, they do however differ significantly in the way they handle nonlinear correlations. In particular, only Ensemble Slice Sampling (ESS) is locally adaptive because of its stepping-out procedure and therefore able to handle nonlinear correlations efficiently.

To illustrate ESS's performance in a case of strong nonlinear correlations, we will use the 16-dimensional ring distribution defined by:

$$\ln \mathcal{L} = -\left[\frac{(x_n^2 + x_1^2 - a)^2}{b}\right]^2 - \sum_{i=1}^{n-1}\left[\frac{(x_i^2 + x_{i+1}^2 - a)^2}{b}\right]^2, \tag{14}$$

where $a = 2$, $b = 1$ and $n = 16$ are the total number of parameters. We also set the number of walkers to be 64 and run the samplers for $10^7$ steps discarding the first half of the chains. Here, we followed the heuristics discussed at the beginning of this section and increased the number of walkers from the minimum of $2 \times 16$ to $4 \times 16$ due to the presence of strong nonlinear correlations in order to achieve the optimal acceptance rate for AIES and DEMC. The number of iterations is large enough for all samplers to converge and provide accurate estimates of the autocorrelation time.

The results are shown in Table 2 and verify that ESS' performance is an order of magnitude better than that of the other methods.

**Table 2** The table shows a comparison of the Affine-Invariant Ensemble Sampling (AIES), Differential Evolution Markov Chain (DEMC) and Ensemble Slice Sampling methods in terms of the integrated autocorrelation time (IAT) and the number of effective samples per evaluation of the probability density (efficiency) multiplied by $10^5$

| | AIES | DEMC | ESS |
|---|---|---|---|
| Ring distribution | | | |
| IAT | 49470 | 91128 | 1675 |
| Efficiency | 2.0 | 1.1 | 12.2 |
| Gaussian shells distribution | | | |
| IAT | 33046 | 2760 | 89 |
| Efficiency | 3.0 | 36.0 | 731.0 |
| Hierarchical Gaussian process regression | | | |
| IAT | 55236 | 30990 | 547 |
| Efficiency | 1.8 | 3.2 | 38.0 |

These metrics are formally defined in "Appendix A". The target distributions are the 16-dimensional ring distribution, the 10-dimensional Gaussian shells distribution and the 13-dimensional hierarchical Gaussian process regression distribution. In all cases, the total number of iterations was set to $10^7$. It should be noted that in the case of the Gaussian shells the global move was used instead of the differential move

#### 4.2.2 Gaussian shells distribution

Another example that demonstrates ESS's performance in cases of nonlinear correlations is the Gaussian Shells distribution defined as:

[6] As there are many different flavours of SMC, we decided to use the one implemented in `PyMC3` which utilises importance sampling, simulated annealing and Metropolis sampling.

$$\mathcal{L}(\boldsymbol{\Theta}) = \text{circ}(\boldsymbol{\Theta}|\mathbf{c}_1, r_1, w_1) + \text{circ}(\boldsymbol{\Theta}|\mathbf{c}_2, r_2, w_2), \tag{15}$$

where

$$\text{circ}(\boldsymbol{\Theta}|\mathbf{c}, r, w) = \frac{1}{\sqrt{2\pi}\, w} \exp\left[ -\frac{1}{2} \frac{(|\Theta - \mathbf{c}| - r)^2}{w^2} \right]. \tag{16}$$

We choose the centres, $\mathbf{c}_1$ and $\mathbf{c}_2$ to be $-3.5$ and 3.5 in the first dimension, respectively, and zero in all others. We take the radius to be $r = 2.0$ and the width $w = 0.1$. In two dimensions, the aforementioned distribution corresponds to two equal-sized Gaussian Shells. In higher dimensions, the geometry of the distribution becomes more complicated and the density becomes multimodal.

For our test, we set the number of dimensions to 10 and the number of walkers to 40 due to the existence of two modes. Since this target distribution exhibits some mild multimodal behaviour we opt for the global move instead of the default differential move although the latter also performs acceptably in this case. The total number of iterations was set to $10^7$, and the first half of the chains was discarded. The results are presented in Table 2. ESS's autocorrelation time is 2–3 orders of magnitude lower than that of the other methods and the efficiency is higher by 1–2 orders of magnitude, respectively.

#### 4.2.3 Hierarchical Gaussian process regression

To illustrate ESS's performance in a real-world example, we will use a modelling problem concerning the concentration of $CO_2$ in the atmosphere adapted from Chapter 5 of Rasmussen (2003). The data consist of monthly measurements of the mean $CO_2$ concentration in the atmosphere measured at the *Mauna Loa Observatory* (Keeling and Whorf 2004) in *Hawaii* since 1958. Our goal is to model the concentration of $CO_2$ as a function of time. To this end, we will employ a *hierarchical Gaussian process* model with a composite covariance function designed to take care of the properties of the data. In particular, the covariance function (kernel) is the sum of following four distinct terms:

$$k_1(r) = \theta_1^2 \exp\left( -\frac{r^2}{2\theta_2} \right), \tag{17}$$

where $r = x - x'$ that describes the smooth trend of the data,

$$k_2(r) = \theta_3^2 \exp\left[ -\frac{r^2}{2\theta_4} - \theta_5 \sin^2\left( \frac{\pi r}{\theta_6} \right) \right], \tag{18}$$

that describes the seasonal component,

$$k_3(r) = \theta_7^2 \left[ 1 + \frac{r^2}{2\theta_8 \theta_9} \right]^{-\theta_8}, \tag{19}$$

which encodes medium-term irregularities, and finally:

$$k_4(r) = \theta_{10}^2 \exp\left( -\frac{r^2}{2\theta_{11}} \right) + \theta_{12}^2 \delta_{ij}, \tag{20}$$

that describes the noise. We also fit the mean of the data, having in total 13 parameters to sample.

We sample this target distribution using 36 walkers for $10^7$ iterations, and we discard the first half of the chains. The number of walkers that was used corresponds to 1.5 times the minimum number. We found that this value results in the optimal acceptance rate for AIES and DEMC. For this example, we use the differential move of ESS. The results are presented in Table 2. The integrated autocorrelation time of ESS is 2 orders of magnitude lower than that of the other methods and its efficiency is more than an order of magnitude higher. The performance is weakly sensitive to the choice of the number of walkers.

#### 4.2.4 Bayesian object detection

Another real-world example with many applications in the field of *astronomy* is *Bayesian object detection*. The following model adapted from Feroz and Hobson (2008) can be used with a few adjustments to detect astronomical objects in telescope images often hidden in background noise.

We assume that the 2D circular objects present in the image are described by the Gaussian profile:

$$\mathbf{G}(x, y; \boldsymbol{\theta}) = A \exp\left[ -\frac{(x - X)^2 + (y - Y)^2}{2R^2} \right], \tag{21}$$

where $\boldsymbol{\Theta} = (X, Y, A, R)$ are parameters that define the coordinate position, the amplitude and the size of the object, respectively. Then, the data can be described as:

$$\mathbf{D} = \mathbf{N} + \sum_{i=1}^{n_{\text{Obj}}} \mathbf{G}(\boldsymbol{\theta_i}), \tag{22}$$

where $n_{\text{Obj}}$ is the number of objects in the image and $\mathbf{N}$ is an additive Gaussian noise term.

Assuming a $200 \times 200$ pixel-wide image, we can create a simulated dataset by sampling the coordinate positions $(X, Y)$ of the objects from $\mathcal{U}(0, 200)$ and their amplitude $A$ and size $R$ from $\mathcal{U}(1, 2)$ and $\mathcal{U}(3, 7)$, respectively. We sample $n_{\text{Obj}} = 8$ objects in total. Finally, we sample the noise $\mathbf{N}$ from $\mathcal{N}(0, 4)$. In practice we create a dataset of 100 such images and one such example is shown in Fig. 7. Notice that the objects are hardly visible as they are obscured by the background noise, this makes the task of identifying those objects very challenging.

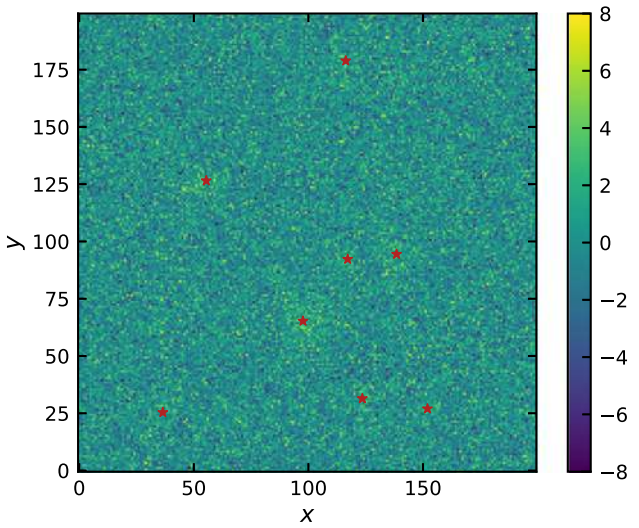


**Fig. 7** The plot shows a simulated image used in the Bayesian object detection exercise. There are 8 circular objects included here. As the objects are hardly visible due to the background noise, their centres are marked with red stars

Following the construction of the simulated dataset, the posterior probability density function is defined as:

$$P(\boldsymbol{\theta}|\mathbf{D}) \propto \exp\left\{\frac{[\mathbf{G}(\boldsymbol{\theta}) - \mathbf{D}]^2}{2\sigma^2}\right\} P(\boldsymbol{\theta})\,, \tag{23}$$

where $\sigma = 2$ is the standard deviation of the $\mathbf{N}$ noise term. The prior $P(\boldsymbol{\theta})$ can be decomposed as the product of prior distributions of $X$, $Y$, $A$ and $R$. We used uniform priors for all of these parameters with limits (0, 200) for $X$ and $Y$, (1, 2) for $A$ and (2, 9) for $R$. It is important to mention here that the posterior does not include any prior information about the exact or maximum number of objects in the data. In that sense, the sampler is agnostic about the exact number, positions and characteristics (i.e. amplitude and size) of the objects that it seeks to detect.

We sampled the posterior distribution using 200 walkers (initialised from the prior distribution) for each image in our dataset (i.e. 100 images in total) using Ensemble Slice Sampling (ESS), Affine-Invariant Ensemble Sampling (AIES) and Differential Evolution Markov Chain (DEMC). Although the posterior distribution is multimodal (i.e. 8 modes) we used the differential move since the number of dimensions is low and there is no reason to use more sophisticated moves like the global move. We used a large enough ensemble of walkers due to the potential presence of multiple modes so that all three samplers are able to resolve them.

We ran each sampler for $10^4$ iterations in total and we discarded the first half of the chains. We found that, on average for the 100 images, ESS identifies correctly 7 out of 8 objects in the image, whereas AIES and DEMC identify 4 and 5, respectively.

In cases where the objects are well-separated ESS often identifies correctly 8 out of 8. Its accuracy falls to 7/8 in cases where two of the objects are very close to each other or overlap. In those cases, ESS identifies the merged object as a single object.

#### 4.2.5 Gaussian Mixture

One strength of ESS is its ability to sample from strongly multimodal distributions in high dimensions. To demonstrate this, we will utilise a Gaussian Mixture of two components centred at $-\mathbf{0.5}$ and $+\mathbf{0.5}$ with standard deviation of $\mathbf{0.1}$. We also put 1/3 of the probability mass in one mode and 2/3 in the other.

We first set this distribution at 10 dimensions and we sample this using 80 walkers for $10^5$ steps. The distance between the two modes in this case is approximately 32 standard deviations. We then increase the number of dimensions to 50 and we sample it using 400 walkers for $10^5$ iterations. In this case, the actual distance between the two modes is approximately 71 standard deviations. The total number of iterations was set to $10^7$ for all methods but the SMC.

This problem consists of two, well-separated, modes and thus requires using at least twice the minimum number of walkers (i.e. at least 40 for the 10-dimensional case and 200 for the 50-dimensional one). Although the aforementioned configuration was sufficient for ESS to provide accurate estimates, we opted instead for twice that number (i.e. 80 walkers for the 10-dimensional cases and 400 for the 50-dimensional one) in order to satisfy the requirements of the other samplers, mainly the Kernel Density Estimate Metropolis (KM), but also AIES and DEMC. For the Sequential Monte Carlo (SMC) sampler, we used 2000 and 20,000 independent chains for the low and high-dimensional case, respectively. The temperature ladder that interpolates between the prior and posterior distribution was chosen adaptively guaranteeing an effective sample size of 90% the physical size of the ensemble. Our implementation of SMC was based on that of `PyMC3` using and an independent Metropolis mutation kernel.

The results for the 10-dimensional and 50-dimensional cases are plotted in Figs. 8 and 9, respectively. In the 10-dimensional case, both ESS (differential and global move) and SMC managed to sample from the target, whereas AIES, DEMC and KM failed to do so. In the 50-dimensional case, only the Ensemble Slice Sampling with the global move manages to sample correctly from this challenging target distribution. In practice, $\text{ESS}_G$ is able to handle similar cases in even higher number of dimensions and with more than 2 modes.

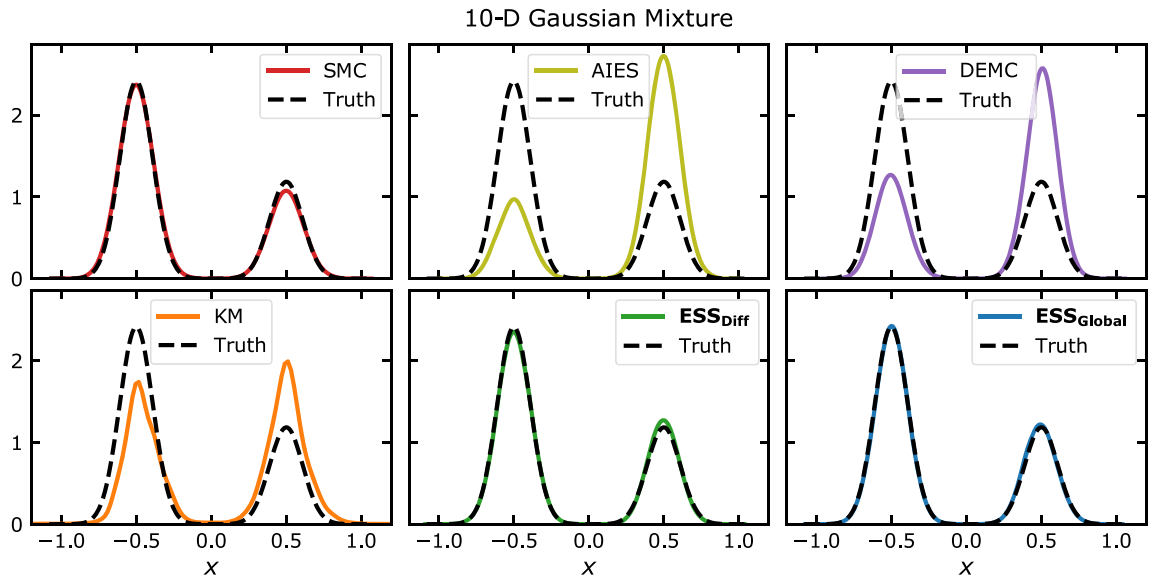


**Fig. 8** The plot compares the results of 6 samplers, namely Sequential Monte Carlo (SMC, red), Affine-Invariant Ensemble Sampling (AIES, yellow), Differential Evolution Markov Chain (DEMC, purple), Kernel Density Estimate Metropolis (KM, orange), Ensemble Slice Sampling using the differential move (ESS, green) and Ensemble Slice Sampling using the global move (ESS, blue). The target distribution is a 10-dimensional Gaussian Mixture. The figure shows the 1D marginal distribution for the first parameter of the 10. (Color figure online)

### 4.3 Convergence of the length scale $\mu$

Figure 10 plots the convergence of the length scale during the first 20 iterations. The target distribution in this example is a 20-dimensional correlated normal distribution. The length scale $\mu$ was initialised from a wide range of possible values. Adaptation is significantly faster when the initial length scale is larger than the optimal one rather than smaller. Another benefit of using a larger initial estimate would be the reduced number of probability evaluations during the first iterations. This is due to the fact that the shrinking procedure is generally faster than the stepping-out procedure.

### 4.4 Parallel scaling

By construction, Ensemble Slice Sampling can be used in parallel computing environments by parallelising the ensemble of walkers as discussed in Sect. 3.2. The maximum number of CPUs used without any of them being idle is equal to the size of complementary ensemble, $n_{\text{Walkers}}/2$. In order to verify this empirically and investigate the scaling of the method for any number of CPUs, we sampled a 10-dimensional normal distribution for $10^5$ iterations with varying number of walkers. The results are plotted in Fig. 11. We sampled the aforementioned distribution multiple times in order to get estimates of the confidence integrals shown in Fig. 11. The required time to do the pre-specified number of iterations scales as $\mathcal{O}(1/n_{\text{CPUs}})$ as long as $n_{\text{CPUs}} \leq n_{\text{Walkers}}/2$. This result does not depend on the specific distribution. We can always use all the available CPUs by matching the size of the complementary ensemble (i.e. half the number of walkers) to the number of CPUs.

## 5 Discussion

In Sect. 4, we provided a quantitative comparison of the efficiency of Ensemble Slice Sampling compared to other methods. In this section, we will provide some qualitative arguments to informally demonstrate the advantages of Ensemble Slice Sampling over other methods. Furthermore, we will briefly discuss some general aspects of the algorithm and place our work in the context of other related algorithms.

After the brief adaptation period is over and the length scale $\mu$ is fixed, the Ensemble Slice Sampling algorithm performs on average 5 evaluations of the probability density per walker per iteration, assuming that either the differential or Gaussian move is used. This is in stark contrast with Metropolis-based MCMC methods that perform 1 evaluation of the probability density per iteration. However, the non-rejection nature of Ensemble Slice Sampling more than compensates for the higher number of evaluations as shown in Sect. 4, thus yielding a very efficient scheme.

One could think of the number of walkers as the only free hyperparameter of Ensemble Slice Sampling. However, choosing the number of walkers is usually trivial. As we mentioned briefly at the end of Sect. 3, there is a minimum limit to that number. In particular, in order for the method to

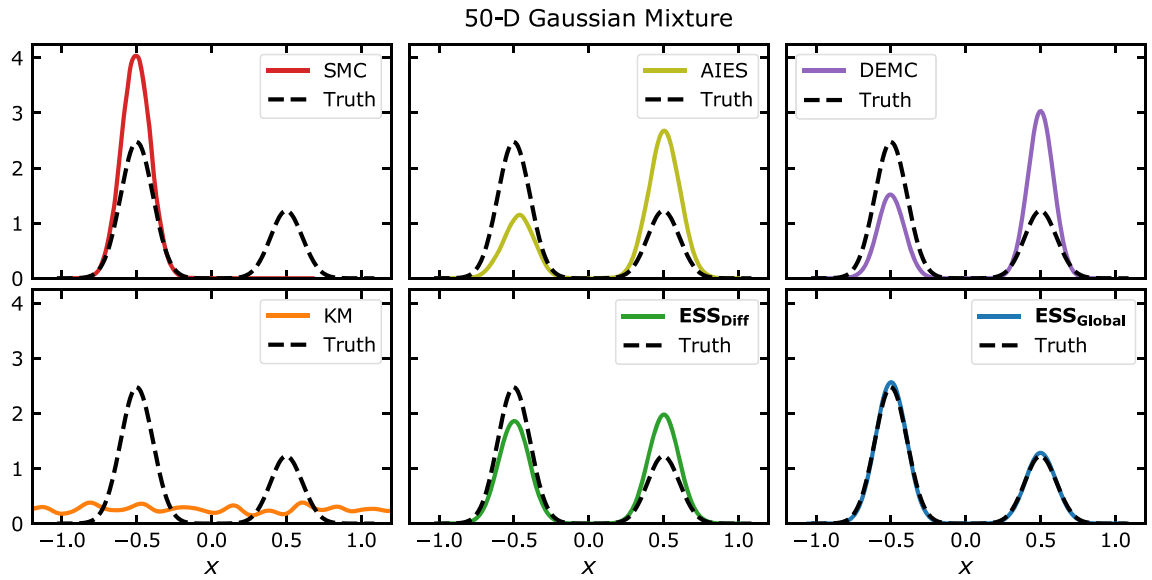


**Fig. 9** The plot compares the results of 6 samplers, namely Sequential Monte Carlo (SMC, red), Affine-Invariant Ensemble Sampling (AIES, yellow), Differential Evolution Markov Chain (DEMC, purple), Kernel Density Estimate Metropolis (KM, orange), Ensemble Slice Sampling using the differential move (ESS, green) and Ensemble Slice Sampling using the global move (ESS, blue). The target distribution is a 50-dimensional Gaussian Mixture. The figure shows the 1D marginal distribution for the first parameter of the 50. (Color figure online)

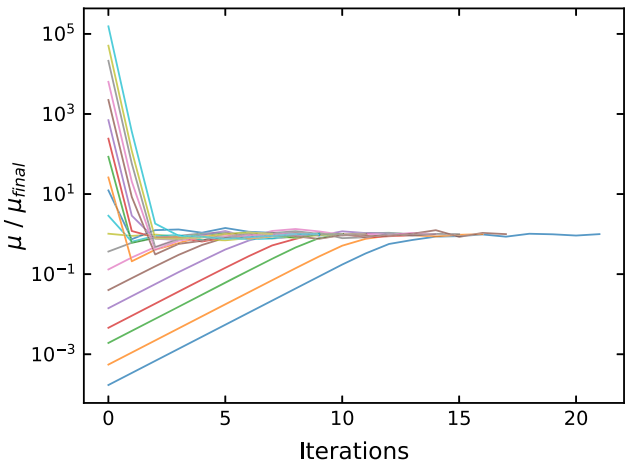


**Fig. 10** The plot shows the adaptation of the length scale $\mu$ as a function of the number of iterations and starting from a wide range of initial values. Each trace is an independent run and the $y$-axis shows the value of $\mu$ divided by the final value of $\mu$. The target distribution in this example is a 20-dimensional correlated normal distribution. Starting from larger $\mu$ values leads to significantly faster adaptation

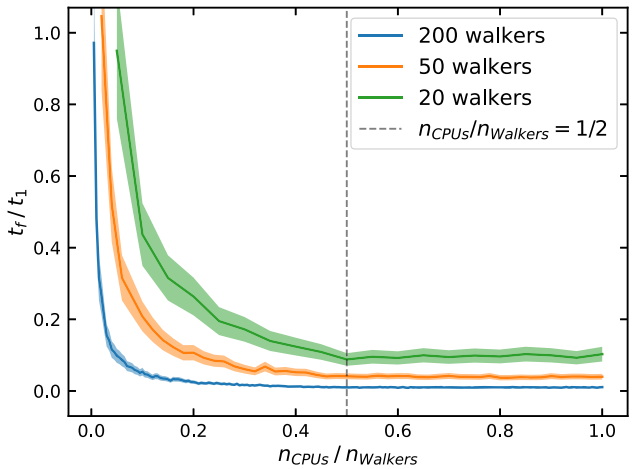


**Fig. 11** The plot shows the time $t_f$ required for ESS to complete a pre-specified number of iterations as a function of the ratio of the number of available CPUs $n_{\text{CPUs}}$ to the total number of walkers $n_{\text{Walkers}}$. The results are normalised with respect to the single CPU case $t_1$. The method scales as $\mathcal{O}(1/n_{\text{CPUs}})$ as long as $n_{\text{CPUs}} \leq n_{\text{Walkers}}/2$ (dashed line). The shaded areas show the $2-\sigma$ intervals

be ergodic, the ensemble should be made of at least $2 \times D$ walkers,[7] where $D$ is the number of dimensions of the problem. Assuming that the initial relative displacements of the walkers span the parameter space (i.e. they do not belong to a lower-than-$D$-dimensional space) the resulting algorithm would be ergodic. As shown in Sect. 4, using a value close to the minimum number of walkers, meaning twice the number of parameters, is generally a good choice. Furthermore, we suggest to increase the number of walkers by a multiplicative factor equal to the number of well separated modes (e.g. four times the number of dimensions in a bimodal density).

[7] The reason that the minimum limit is $2 \times D$ instead of $D + 1$ has to do with the ensemble splitting procedure that we introduced in order to make the method parallel. Splitting the ensemble into two equal parts means that each walker is updated based on the relative displacements of half the ensemble.

Other cases in which increasing the number of walkers can improve the sampling efficiency include target distributions with strong nonlinear correlations between their parameters.

Regarding the initial positions of the walkers, we found that we can reduce the length of the burn-in phase by initialising the walkers from a tight sphere (i.e. normal distribution with a very small variance) close to the *Maximum a Posteriori* (MAP) estimate. In high-dimensional problems, the MAP estimate will not reside in the typical set and the burn-in phase might be longer. We found that the tight sphere initialisation is still an efficient strategy compared to a more dispersed initialisation (Foreman-Mackey et al. 2013). Other approaches include initialising the walkers by sampling from the prior distribution or the *Laplace approximation* of the posterior distribution. In multimodal cases, a prior initialisation is usually a better choice. A brief simulated annealing phase can also be very efficient, particularly in cases with many well separated modes.

Recent work on the No U-Turn Sampler (Hoffman and Gelman 2014) has attempted to reduce the hand-tuning requirements of Hamiltonian Monte Carlo (Betancourt 2017) using the dual averaging scheme of Nesterov (2009). In order to achieve a similar result, we employed the much simpler stochastic approximation method of Robbins and Monro (1951) to tune the initial length scale $\mu$. The Affine-Invariant Ensemble Sampler (Goodman and Weare 2010) and the Differential Evolution MCMC (Ter Braak 2006) use an ensemble of walkers to perform Metropolis updates. Our method differs by using the information from the ensemble to perform Slice Sampling updates. So why does ESS perform better, as demonstrated, compared to those other methods? The answer lies in the locally adaptive and non-rejection nature of the algorithm (i.e. stepping out and shrinking) that enables both efficient exploration of nonlinear correlations and large steps in parameter space (e.g. using the global move).[8]

For all numerical benchmarks in this paper, we used the publicly available, open-source `Python` implementation of Ensemble Slice Sampling called `zeus`[9] (Karamanis et al. 2021).

## 6 Conclusion

We have presented Ensemble Slice Sampling (ESS), an extension of Standard Slice Sampling that eliminates the latter's dependence on the initial value of the length scale hyperparameter and augments its capacity to sample efficiently and in parallel from highly correlated and strongly multimodal distributions.

In this paper, we have compared Ensemble Slice Sampling with the optimally-tuned Metropolis and Standard Slice Sampling algorithms. We found that, due to its affine invariance, Ensemble Slice Sampling generally converges faster to the target distribution and generates chains of significantly lower autocorrelation. In particular, we found that in the case of AR(1), Ensemble Slice Sampling generates an order of magnitude more independent samples per evaluation of the probability density than Metropolis and Standard Slice Sampling. Similarly, in the case of the correlated funnel distribution, Ensemble Slice Sampling outperforms Standard Slice Sampling by an order of magnitude in terms of efficiency. Furthermore, in this case, Metropolis-based proposals fail to converge at all, demonstrating that a single Metropolis proposal scale is often not sufficient.

When compared to state-of-the-art ensemble methods (i.e. AIES, DEMC), Ensemble Slice Sampling outperforms them by 1–2 orders of magnitude in terms of efficiency for target distributions with nonlinear correlations (e.g. the Ring and Gaussian shells distributions). In the real-world example of hierarchical Gaussian process regression, ESS's efficiency is again superior by 1–2 orders of magnitude. Furthermore, in the Bayesian object detection example ESS achieved higher accuracy compared to AIES and DEMC. Finally, in the strongly multimodal case of the Gaussian Mixture, ESS outperformed all other methods (i.e. SMC, AIES, DEMC, KM) and was the only sampler able to produce reliable results in 50 dimensions.

The consistent high efficiency of the algorithm across a broad range of different problems along with its parallel, black-box and gradient-free nature renders Ensemble Slice Sampling ideal for use in scientific fields such as physics, astrophysics and cosmology, which are dominated by a wide range of computationally expensive and almost always non-differentiable models. The method is flexible and can be extended further using for example tempered transitions (Iba 2001) or subspace sampling (Vrugt et al. 2009).

**Acknowledgements** The authors thank Iain Murray and John Peacock for providing constructive comments on an early draft. The authors would also like to extend their gratitude to the anonymous reviewer and editor for providing comments that helped improve the quality of the manuscript. FB is a Royal Society University Research Fellow. FB is supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (Grant Agreement No. 853291).



[8] Indeed, large steps like the ones in the 50-dimensional Gaussian Mixture example would not have been possible without the non-rejection aspect of the method as most attempts to jump to the other mode would have missed it using Metropolis updates.

[9] The code is available at https://github.com/minaskar/zeus.

## A Estimating the effective sample size

Assuming that the computational bottleneck of a MCMC analysis is the evaluation of the probability density function, which is usually a valid assumption in scientific applications, the *efficiency* can be formally defined as the ratio of the *Effective Sample Size* $N_{\mathrm{Eff}}$ to the total number of probability evaluations for a given chain.

The $N_{\mathrm{Eff}}$ quantifies the number of effectively independent samples of a chain, and it is defined as

$$N_{\mathrm{Eff}} = \frac{n}{\mathrm{IAT}}, \tag{24}$$

where $n$ is the actual number of samples in the chain, and IAT is the *integrated autocorrelation time*. The latter describes the number of steps that the sampler needs to do in order to forget where it started and it is defined as

$$\mathrm{IAT} = 1 + 2\sum_{k=1}^{\infty} \rho(k), \tag{25}$$

where $\rho(k)$ is the *normalised autocorrelation function* at lag $k$. In practise, we truncate the above summation in order to remove noise from the estimate (Sokal 1997).

Given a chain $X(k)$ with $k = 1, 2, ..., n$ the normalised autocorrelation function $\hat{\rho}(k)$ at lag $k$ is estimated as

$$\hat{\rho}(k) = \frac{\hat{c}(k)}{\hat{c}(0)}, \tag{26}$$

where

$$\hat{c}(k) = \frac{1}{n-k}\sum_{m=1}^{n-k} \big[X(k+m) - \bar{X}\big]\big[X(m) - \bar{X}\big], \tag{27}$$

and $\bar{X}$ is the mean of the samples.

In the case of ensemble methods, the IAT of an ensemble of chains is computed by first concatenating the chain from each walker into a single long chain. We found this estimator has lower variance than the Goodman and Weare (2010) estimator and the Foreman-Mackey (2019) estimator.

## References


Betancourt, M.: A conceptual introduction to Hamiltonian Monte Carlo. arXiv preprint arXiv:170102434 (2017)

Bishop, C.M.: Pattern recognition. Mach. Learn. **128**(9), (2006)

Del Moral, P., Doucet, A., Jasra, A.: Sequential Monte Carlo samplers. J. R. Stat. Soc. Ser. B (Stat. Methodol.) **68**(3), 411–436 (2006)

Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the EM algorithm. J. R. Stat. Soc. Ser. B (Methodol.) **39**(1), 1–22 (1977)

Farr. B, Farr, W.M.: Kombine: a kernel-density-based, embarrassingly parallel ensemble sampler. (2015) https://github.com/bfarr/kombine

Feroz, F., Hobson, M.P.: Multimodal nested sampling: an efficient and robust alternative to Markov chain Monte Carlo methods for astronomical data analyses. Mon. Not. R. Astron. Soc. **384**(2), 449–463 (2008)

Foreman-Mackey, D.: Autocorrelation analysis & convergence—EMCEE 3.0.2 documentation. (2019) https://emcee.readthedocs.io/en/stable/tutorials/autocorr/

Foreman-Mackey, D., Hogg, D.W., Lang, D., Goodman, J.: EMCEE: the MCMC hammer. Publ. Astron. Soc. Pac. **125**(925), 306 (2013)

Garbuno-Inigo, A., Hoffmann, F., Li, W., Stuart, A.M.: Interacting Langevin diffusions: gradient structure and ensemble Kalman sampler. SIAM J. Appl. Dyn. Syst. **19**(1), 412–441 (2020a)

Garbuno-Inigo, A., Nüsken, N., Reich, S.: Affine invariant interacting Langevin dynamics for Bayesian inference. SIAM J. Appl. Dyn. Syst. **19**(3), 1633–1658 (2020b)

Gilks, W.R., Roberts, G.O., George, E.I.: Adaptive direction sampling. J. R. Stat. Soc. Ser. D (The Statistician) **43**(1), 179–189 (1994)

Goodman, J., Weare, J.: Ensemble samplers with affine invariance. Commun. Appl. Math. Comput. Sci. **5**(1), 65–80 (2010)

Görür, D., Rasmussen, C.E.: Dirichlet process Gaussian mixture models: choice of the base distribution. J. Comput. Sci. Technol. **25**(4), 653–664 (2010)

Haario, H., Saksman, E., Tamminen, J., et al.: An adaptive metropolis algorithm. Bernoulli **7**(2), 223–242 (2001)

Hoffman, M.D., Gelman, A.: The no-u-turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo. J. Mach. Learn. Res. **15**(1), 1593–1623 (2014)

Iba, Y.: Extended ensemble Monte Carlo. Int. J. Mod. Phys. C **12**(05), 623–656 (2001)

Karamanis, M., Beutler, F., Peacock, J.A.: Zeus: a python implementation of ensemble slice sampling for efficient Bayesian parameter inference. arXiv preprint arXiv:210503468 (2021)

Keeling, C.D., Whorf, T.P.: Atmospheric CO2 concentrations derived from flask air samples at sites in the SiO network. Trends: a compendium of data on Global Change (2004)

Leimkuhler, B., Matthews, C., Weare, J.: Ensemble preconditioning for Markov chain Monte Carlo simulation. Stat. Comput. **28**(2), 277–290 (2018)

Liu, J.S., Chen, R.: Sequential Monte Carlo methods for dynamic systems. J. Am. Stat. Assoc. **93**(443), 1032–1044 (1998)

MacKay, D.J.: Information Theory, Inference and Learning Algorithms. Cambridge University Press, Cambridge (2003)

Murray, I., Adams, R., MacKay, D.: Elliptical slice sampling. In: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, JMLR Workshop and Conference Proceedings, pp. 541–548 (2010)

Neal, R.M.: Slice sampling. Ann. Stat. pp. 705–741 (2003)

Neal, R.M., et al.: MCMC using Hamiltonian dynamics. Handbook of Markov chain Monte Carlo **2**(11), 2 (2011)

Nesterov, Y.: Primal-dual subgradient methods for convex problems. Math. Programm. **120**(1), 221–259 (2009)

Nishihara, R., Murray, I., Adams, R.P.: Parallel MCMC with generalized elliptical slice sampling. J. Mach. Learn. Res. **15**(1), 2087–2112 (2014)

Rasmussen, C.E.: Gaussian processes in machine learning. In: Summer School on Machine Learning, Springer, pp. 63–71 (2003)

Robbins, H., Monro, S.: A stochastic approximation method. Ann. Math. Stat., pp. 400–407 (1951)

Roberts, G.O., Rosenthal, J.S.: Coupling and ergodicity of adaptive Markov chain Monte Carlo algorithms. J. Appl. Prob. **44**(2), 458–475 (2007)

Sokal, A.: Monte Carlo methods in statistical mechanics: foundations and new algorithms. In: Functional Integration, Springer, pp. 131–192 (1997)

Ter Braak, C.J.: A Markov chain Monte Carlo version of the genetic algorithm differential evolution: easy Bayesian computing for real parameter spaces. Stat. Comput. **16**(3), 239–249 (2006)

ter Braak, C.J., Vrugt, J.A.: Differential evolution Markov chain with snooker updater and fewer chains. Stat. Comput. **18**(4), 435–446 (2008)

Tibbits, M.M., Groendyke, C., Haran, M., Liechty, J.C.: Automated factor slice sampling. J. Comput. Gr. Stat. **23**(2), 543–563 (2014)

Tran, K.T., Ninness, B.: Reunderstanding slice sampling as parallel MCMC. In: 2015 IEEE Conference on Control Applications (CCA), IEEE, pp. 1197–1202 (2015)

Vrugt, J.A., Ter Braak, C., Diks, C., Robinson, B.A., Hyman, J.M., Higdon, D.: Accelerating Markov chain Monte Carlo simulation by differential evolution with self-adaptive randomized subspace sampling. Int. J. Nonlinear Sci. Numer. Simul. **10**(3), 273–290 (2009)