A FAST UNIVERSAL SELF-TUNED SAMPLER WITHIN GIBBS SAMPLING

L. Martino[1], *H. Yang*[2], *D. Luengo*[3], *J. Kanniainen*[2], *J. Corander*[1]

[1] Dep. of Mathematics and Statistics, University of Helsinki, 00014 Helsinki (Finland).
[2] Dep. of Industrial Engineering, Tampere University of Technology, Tampere (Finland).
[3] Dep. of Signal Theory and Communications, Universidad Politécnica de Madrid, 28031 Madrid (Spain).

ABSTRACT

Bayesian inference often requires efficient numerical approximation algorithms, such as sequential Monte Carlo (SMC) and Markov chain Monte Carlo (MCMC) methods. The Gibbs sampler is a well-known MCMC technique, widely applied in many signal processing problems. Drawing samples from univariate full-conditional distributions efficiently is essential for the practical application of the Gibbs sampler. In this work, we present a simple, self-tuned and extremely efficient MCMC algorithm which produces virtually independent samples from these univariate target densities. The proposal density used is self-tuned and tailored to the specific target, but it is not adaptive. Instead, the proposal is adjusted during an initial optimization stage, following a simple and extremely effective procedure. Hence, we have named the newly proposed approach as FUSS (Fast Universal Self-tuned Sampler), as it can be used to sample from any bounded univariate distribution and also from any bounded multi-variate distribution, either directly or by embedding it within a Gibbs sampler. Numerical experiments, on several synthetic data sets (including a challenging parameter estimation problem in a chaotic system) and a high-dimensional financial signal processing problem, show its good performance in terms of speed and estimation accuracy.

1. INTRODUCTION

Bayesian methods, and their implementations by means of sophisticated Monte Carlo techniques (1; 2), have become very popular over the last two decades. Indeed, many practical statistical signal processing problems demand procedures for drawing from probability distributions with non-standard forms, such as Markov chain Monte Carlo (MCMC) methods (3; 4) and particle filters (5; 6; 7). MCMC techniques generate samples from a target probability density function (pdf) by drawing from a simpler proposal pdf (1; 8) and generating a Markov chain. The two most widely applied MCMC approaches are the Metropolis-Hastings (MH) algorithm and the Gibbs sampler (1; 2).

The Gibbs sampling technique is extensively used in Bayesian inference (9) to generate samples from multivariate target densities, drawing each component of the samples from univariate full-conditional densities (10; 11; 12).[1] When the multivariate target can be easily factorized into univariate conditional pdfs, the key point for the successful application of the Gibbs sampler is the ability to draw efficiently from these univariate pdfs (1; 2; 10). The best scenario for Gibbs sampling occurs when exact samplers for each full-conditional are available. Otherwise, another exact sampling technique, like rejection sampling (RS) or an MH-type algorithm, is typically used *within* the Gibbs sampler to draw from the complicated full-conditionals. In the first case, samples generated from the RS algorithm are independent, but the acceptance rate can be very low. In the second case, we have an approach where an internal MCMC (the MH method) is applied *inside* another external MCMC (the Gibbs sampler). Therefore, the typical problems of the *external*-MCMC (long "burn-in" period, large correlation, etc.) could raise dramatically if the *internal*-MCMC is not extremely efficient. Indeed, although the Gibbs sampler needs only one sample from each full-conditional, several iterations are typically performed to avoid the "burn-in" period of the *internal*-MCMC.[2]

In order to avoid these problems, several automatic and self-tuning samplers have been proposed: *adaptive rejection sampling* (ARS) (14; 15), *Adaptive Rejection Metropolis Sampling* (ARMS) (16; 17; 18), *Independent Doubly Adaptive*

[1]Blockwise Gibbs sampling approaches, where several random variables are updated simultaneously, have been proposed to speed up the convergence of the Gibbs sampler (13). However, unless direct sampling from the multi-variate full-conditionals is feasible, these approaches result in an increased difficulty of drawing samples and a higher computational cost per iteration. Furthermore, the performance of the overall algorithm can decrease if the blocks are not properly chosen, especially when direct sampling from the multi-variate full-conditionals is unfeasible.

[2]Note that, from a theoretical point of view, performing a single iteration of the internal MH is enough to guarantee the ergodicity of the Gibbs sampler. However, the convergence of the chain can be very slow. Namely, the performance of the resulting estimator built from those samples can be very poor if the proposal pdf is not very similar to the full-conditionals. Hence, several iterations of the internal MH algorithm are typically required in order to achieve the desired level of performance (more as the proposal differs more from the target). See the numerical examples in Section 6 for further details on this issue.

Rejection Metropolis Sampling (IA^{2}RMS) (19; 20), *Adaptive Sticky Metropolis* (ASM) (21), etc. ARS builds a piecewise linear proposal on the target's log-domain, starting with a reduced number of support points and incorporating new points whenever a candidate sample is rejected. Unfortunately, since it is based on the rejection sampling technique, the proposal must be always above the target, a requirement which is only fulfilled by log-concave targets. In order to solve this issue, ARMS introduces a Metropolis-Hastings step, thus obtaining a universal sampler which is able to draw virtually from any target pdf. However, the adaptive structure in ARMS has an important restriction: support points cannot be added inside regions where the proposal is below the target. Recently, the IA^{2}RMS and ASM algorithms have been proposed to overcome this drawback, introducing more flexibility in the mechanism used to add points to the support set and decoupling it from the proposal construction.

All the previous methods build an adaptive sequence of proposal pdfs via some interpolation procedure given a set of support points. The proposal is updated when a new support point is incorporated, according to some statistical criterion. However, although these methods can attain a very good performance, the results show a dependence on the initial set of support points. Another drawback is the difficulty of ensuring their ergodicity, especially in applications within Gibbs sampling (2; 17), as the Markovian nature of the chain is lost due to the adaptive nature of the proposal, which may depend on all the previous samples. Other related works, where a non-adaptive proposal pdf is built via interpolation procedures can be found in literature (22; 23; 24). Furthermore, different types of generic adaptive MH schemes based on independent proposals have also been studied (25; 26). However, in general, the considered proposal pdf has a fixed parametric form so that the complete adaptation of the proposal is not possible.

In this work, we present a novel algorithm that follows a complementary strategy: start with a large number of support points and remove many of them following some pruning strategy.[3] The key idea is starting with a thin uniform grid that covers the effective support of the target and discard those support points that do not provide relevant information according to some pre-defined criterion. The idea of using a grid and a piecewise linear constant function to approximate a uni-variate full conditional was already proposed by the griddy Gibbs sampler (27). However, their approach is substantially different from ours. On the one hand, they simply select a few support points heuristically, instead of starting with a large set of points and selecting the best ones in a principled way. On the other hand, the proposal in the griddy Gibbs sampler does not take into account the tails of the target (the proposal is set to zero outside of the interval covered by the grid), thus providing a poor performance for slowly decaying tails (e.g., heavy tailed distributions). Furthermore, this griddy approximation of the uni-variate full conditionals is not embedded within another *inner* Monte Carlo method, thus leading to an approximate sampler, unlike our scheme, which results in an exact sampler.

The resulting method is fast and extremely efficient (it yields virtually independent samples), even for highly multimodal and complicated targets. The dependence on the initial set of points is also drastically reduced, since the algorithm only requires an approximate knowledge of the effective support of the target pdf. Moreover, unlike previous approaches, the proposal is self-tuned during the initialization stage, without any adaptation afterwards. Hence, ergodicity is not an issue and the convergence of the chain to the target distribution is always guaranteed. For these reasons, we call the new method FUSS ("Fast Universal Self-tuned Sampler") since, with this sampler, there is no "fuss" about convergence or tuning. The FUSS algorithm is particularly well suited for multi-modal and spiky target densities (i.e., densities with several sharp and narrow modes), where virtually all of the existing MCMC techniques fail. This kind of target pdfs often appears in practical applications, e.g., in ecology, bioinformatics and financial inference problems (see Sections 6 and 7).

The rest of the paper is organized as follows. Sections 2 and 3 are devoted to recalling the general framework and describing the structure of the novel technique. Details about the proposal construction and generation are given in Section 4. Different pruning algorithms are then introduced in Section 5. Sections 6 and 7 provide numerical results on several uni-variate and multi-variate pdfs, including a challenging parameter estimation problem in a chaotic system, as well as a multi-dimensional and multi-modal inference problem in financial signal processing. Finally, Section 8 contains some brief final remarks.

2. PROBLEM STATEMENT

Bayesian inference often requires drawing samples from complicated multivariate posterior pdfs, $\pi(\mathbf{x}|\mathbf{y})$ with $\mathbf{x} \in \mathcal{X}^D \subseteq \mathbb{R}^D$. A common approach, when direct sampling from $\pi(\mathbf{x}|\mathbf{y})$ is unfeasible, is using a Gibbs sampler (2). At the i-th iteration, a Gibbs sampler obtains the d-th component ($d = 1, \dots, D$) of $\mathbf{x}$, x_d, by drawing from the full conditional pdf of x_d given all the previously generated components (2; 9; 28), i.e.,

$$x_d^{(i)} \sim \bar{\pi}(x_d|\mathbf{x}_{1:d-1}^{(i)}, \mathbf{x}_{d:D}^{(i-1)}) = \bar{\pi}(x_d) \propto \pi(x_d), \tag{1}$$

[3]Note that all of the previous methods typically follow the opposite strategy: start with a reduced number of support points and keep adding points in order to improve the proposal adaptively.

Table 1. General structure of the FUSS algorithm.

1. **Initialization:** Choose a set of support points, $\mathcal{S}_M = \{s_1, \ldots, s_M\}$, such that $s_1 < s_2 < \ldots < s_M$.
2. **Pruning:** Remove support points according to a pre-specified criterion, attaining a final set $\mathcal{S}_m$, with $m < M$.
3. **Construction:** Build a proposal function $p(x|\mathcal{S}_m)$ given $\mathcal{S}_m$, using some appropriate pre-defined mechanism.
4. **MCMC algorithm:** Perform K steps of an MCMC method using $p(x|\mathcal{S}_m)$ as proposal pdf, thus yielding a set of samples $\{x_1, \ldots, x_K\}$.

where $\bar{\pi}(x_d)$ is the normalized target pdf, $\pi(x_d)$ denotes its unnormalized counterpart (note that we have dropped the dependence on $\mathbf{x}_{1:d-1}^{(i)}$ and $\mathbf{x}_{d:D}^{(i-1)}$ to simplify the notation), $x_d \in \mathcal{X}$ and the initial vector is typically drawn from the prior (i.e., $\mathbf{x}^{(0)} \sim \bar{\pi}_0(\mathbf{x})$), but can also be set to some fixed value when no prior information is available or it is unreliable.

However, even sampling from the univariate pdfs in Eq. (1) can often be complicated. In these cases, a common approach is to use another Monte Carlo technique (e.g., rejection sampling (RS) or the Metropolis-Hastings (MH) algorithm) within the Gibbs sampler, drawing candidates from a simpler proposal pdf,

$$\bar{p}(x) \propto p(x) = e^{W(x)},$$

where $\bar{p}(x)$ and $p(x)$ denote the normalized and unnormalized proposal respectively, $W(x)$ is a "potential" function and $x \in \mathbb{R}$. The best case occurs when an RS technique can be applied, since it yields independent and identically distributed (i.i.d.) samples. However, RS requires $p(x) \geq \pi(x)$ for all $x \in \mathcal{X}$, which may be hard to guarantee in practice. For instance, the adaptive rejection sampling (ARS) technique can be applied only to log-concave target pdfs (15). Thus, the use of another MCMC method becomes almost mandatory in practical applications. In this case, the performance of this approach depends strictly on the choice of $p(x)$. Our aim is designing an efficient fast sampler to draw from the univariate target pdf,

$$\pi(x) = e^{V(x)}, \quad x \in \mathcal{X} \subseteq \mathbb{R}, \tag{2}$$

where $V(x) = \log[\pi(x)]$ is a generic function, i.e., $\pi(x)$ can be multimodal and/or heavy tailed for example.

3. STRUCTURE OF THE ALGORITHM

FUSS is an MCMC approach based on an independent proposal pdf, built through a simple interpolation procedure, like the ones shown in the next section. The general structure of FUSS is given in Table 1. The first 3 steps consist of an optimization procedure (applied only once, during the algorithm's initialization stage), designed to obtain a good proposal density, tailored to the shape of the target. Step 4 contains the MCMC iterations, which are repeated K times. In this work, we consider two possible techniques for Step 4 of FUSS. The first one is the well known *Metropolis-Hastings (MH) algorithm* (2), which is recalled in Table 2. In this case, the resulting method is denoted as FUSS-MH. The second one is the *rejection chain (RC) algorithm* (29; 30), which is shown in Table 3. In this case, a rejection sampling (RS) test is initially performed, and an MH step is applied only when a sample is accepted, thus ensuring that samples are drawn from the target pdf. We denote this second method as FUSS-RC. On the one hand, FUSS-RC is slower than FUSS-MH, since the chain does not move forward when a sample is rejected in the RS test. On the other hand, FUSS-RC yields samples with a lower correlation, due to the application of the RS test. We test and compare the performance of both schemes through numerical simulations in Sections 6 and 7. The notation $a \wedge b$, used in Tables 2 and 3, denotes the minimum between two real values, i.e., $a \wedge b = \min\{a, b\}$.

3.1. Important remarks

It is important to emphasize some aspects of our approach. First of all, we remark again that steps 1 to 3 of the general FUSS algorithm in Table 1 are performed only once. The success of the FUSS algorithms lies on the speed in performing these steps and the quality of the final proposal density. Indeed, since the shape of the proposal is tailored to $\pi(x)$, the samples generated will always be virtually independent.

The initial support points in $\mathcal{S}_M$ play the role of internal "tuning" parameters of the FUSS algorithm. After the pruning step, the proposal $p(x|\mathcal{S}_m)$ is built according to the final support set, $\mathcal{S}_m$ with $m < M$, and remains fixed for the rest of the

Table 2. The Metropolis-Hastings (MH) method used in Step 4 of the FUSS-MH algorithm.

```
1. Set $k = 0$ and choose $x_0$.
2. Draw $x' \sim \bar{p}(x) \propto p(x|\mathcal{S}_m)$ and $u' \sim \mathcal{U}([0,1])$.
3. Set $x_{k+1} = x'$ with probability
       $\alpha_{MH} = 1 \wedge \frac{\pi(x')p(x_k|\mathcal{S}_m)}{\pi(x_k)p(x'|\mathcal{S}_m)}$.   (3)
   Otherwise, set $x_{k+1} = x_k$ with probability $1 - \alpha_{MH}$.
4. If $k \leq K$, set $k = k + 1$ and repeat from step 3.2. Otherwise, stop.
```

Table 3. The rejection chain (RC) method used in Step 4 of the FUSS-RC algorithm.

```
1. Set $k = 0$ and choose $x_0$.
2. Draw $x' \sim \bar{p}(x) \propto p(x|\mathcal{S}_m)$ and $u' \sim \mathcal{U}([0,1])$.
3. If $u' \geq \frac{\pi(x')}{p(x'|\mathcal{S}_m)}$, then go back to step 3.2.
4. If $u' \leq \frac{\pi(x')}{p(x'|\mathcal{S}_m)}$, set $x_{k+1} = x'$ with probability
       $\alpha_{RC} = 1 \wedge \frac{\pi(x')\left[\pi(x_k) \wedge p(x_k|\mathcal{S}_m)\right]}{\pi(x_k)\left[\pi(x') \wedge p(x'|\mathcal{S}_m)\right]}$.   (4)
   Otherwise, set $x_{k+1} = x_k$ with probability $1 - \alpha_{RC}$.
5. If $k \leq K$, set $k = k + 1$ and repeat from step 3.2. Otherwise, stop.
```

algorithm. Consequently, the FUSS techniques become standard non-adaptive MCMC methods after step 3, thus avoiding any issue about the ergodicity.

The possibility of applying FUSS directly for drawing from multidimensional distributions depends on the ability to construct efficiently the proposal pdf via interpolation in dimensions higher than one (step 3 in the general structure of FUSS). Although a strategy for building a multidimensional proposal is described in Section 4.3, in this work we focus our attention on the application of FUSS within Gibbs sampling to draw from multi-variate pdfs. This allows us to concentrate on designing very efficient procedures for sampling from univariate full-conditional pdfs. Furthermore, note also that all the operations in both algorithms (FUSS-MH and FUSS-RC) can be implemented in the log-domain, thus evaluating the functions $V(x) = \log[\pi(x)]$ and $W(x) = \log[p(x|\mathcal{S}_m)]$, which may be more convenient for many applications.

As a final remark, note that the FUSS-RC algorithm becomes a standard rejection sampler when $p(x|\mathcal{S}_m) \geq \pi(x)$ for all $x \in \mathcal{X}$, thus providing independent identically distributed (i.i.d.) samples from $\bar{\pi}(x)$. Indeed, note that in this scenario the probability of accepting the new state is always $\alpha_{RC} = 1$, i.e., any candidate movement that has survived the rejection test is automatically accepted. This is due to the fact that, in FUSS-RC after the rejection test, the proposed samples are distributed as

$$\bar{q}(x) \propto q(x) = \pi(x) \wedge p(x|\mathcal{S}_m).$$

Finally, note also that $q(x)$ is closer, in terms of L_1 distance, to $\pi(x)$ than to $p(x|\mathcal{S}_m)$. Hence, FUSS-RC produces less correlated samples than FUSS-MH at the expense of an increased computational cost per iteration.[4]

3.2. Initialization and general strategy for FUSS

The initial set $\mathcal{S}_M$ should cover all the high probability regions of the target $\pi(x)$. In general, if no prior information about $\pi(x)$ is available, we suggest the following FUSS approach: choose a large, dense, uniform initial grid of support points, such

[4] In the extreme (and trivial) case when proposal and target are the same function (i.e. $q(x) = \pi(x)$), the correlation is zero, since we are drawing i.i.d. samples directly from the target.

that $s_{i+1} - s_i = \epsilon$ for $i = 1, \dots, M-1$, i.e.,

$$\mathcal{S}_M = \{s_1, s_2 = s_1 + \epsilon, \dots, s_M = s_1 + (M-1)\epsilon\},$$

in order to capture all the main features of the target. The grid could be thicker in the regions where the user desires to focus the computational effort. The number of support points can then be drastically reduced according to some pre-defined pruning criterion (see Section 5 for several proposed approaches), thus obtaining the final set $\mathcal{S}_m$, which is used to build a stepwise approximation of the target pdf. We emphasize again that the resulting proposal pdf is self-tuned and tailored to the target, but is non-adaptive, since it does not vary during the run of the chain (i.e., it is "adapted" offline, before running the Markov chain).

4. CONSTRUCTION OF THE PROPOSAL DENSITY

In this section we describe in detail the construction of the proposal density, both for one-dimensional and multidimensional pdfs. In many applications it is advantageous to evaluate the target pdf in the log-domain. Therefore, in the following we only consider the construction of the proposal function in the log-domain.

4.1. One-dimensional construction

Let us consider the set of support points after the pruning step (see Section 5),

$$\mathcal{S}_m = \{s_1, s_2, \dots, s_m\} \subset \mathcal{X},$$

where $s_1 < \ldots < s_m$, and we define the intervals $\mathcal{I}_0 = (-\infty, s_1]$, $\mathcal{I}_j = (s_j, s_{j+1}]$ for $j = 1, ..., m-1$ and $\mathcal{I}_m = (s_m, +\infty)$. The unnormalized proposal pdf is then given by

$$p(x|\mathcal{S}_m) = e^{W(x)},$$

where $W(x)$ is built using a piecewise constant approximation for all the intervals, with the exception of the first and last intervals (i.e., the tails), where a non-constant function is used. Mathematically,

$$W(x) = w_i(x) = \max\left[V(s_i), V(s_{i+1})\right] \mathbb{I}_{\mathcal{I}_i}(x), \tag{5}$$

where $1 \leq i \leq m-1$ and

$$\mathbb{I}_{\mathcal{I}_i}(x) = \begin{cases} 1, & x \in \mathcal{I}_i = (s_i, s_{i+1}], \\ 0, & x \neq \mathcal{I}_i = (s_i, s_{i+1}]. \end{cases} \tag{6}$$

The maximum in (5) is selected in order to satisfy the inequality, $W(x) \geq V(x)$, in as many regions as possible. Let us recall again that FUSS-RC becomes a standard rejection sampler (thus providing independent samples) when $p(x|\mathcal{S}_m) \geq \pi(x)$ for all $x \in \mathcal{X}$, which is equivalent to $W(x) \geq V(x)$ for all $x \in \mathcal{X}$. In the first and last intervals, $\mathcal{I}_0$ and $\mathcal{I}_m$ respectively, we have

$$W(x) = w_j(x), \quad j \in \{0, m\}, \quad x \in \mathcal{I}_j,$$

where $w_j(x)$ represents a generic log-tail function. For instance, in order to have a light-tailed proposal, linear functions can be selected for $w_j(x)$ when $j \in \{0, m\}$. Figures 1 and 8 provide some specific examples. For further details, see A.

4.2. Variate generation from $\bar{p}(x)$

The proposal $\bar{p}(x) \propto p(x|\mathcal{S}_m)$ is composed of $m+1$ pieces, including the two tails. Therefore, $p(x|\mathcal{S}_m))$ can be seen as a finite mixture

$$p(x|\mathcal{S}_m) = \sum_{i=0}^{m} \eta_i \phi_i(x),$$

with $\sum_{i=0}^{m} \eta_i = 1$, whereas $\phi_i(x) = \exp\left(w_i(x)\right)$ for all $x \in \mathcal{I}_i$, and $\phi_i(x) = 0$ for $x \notin \mathcal{I}_i$. Hence, in order to draw a sample from $\bar{p}(x) \propto p(x|\mathcal{S}_m)$, it is necessary to perform the following steps:

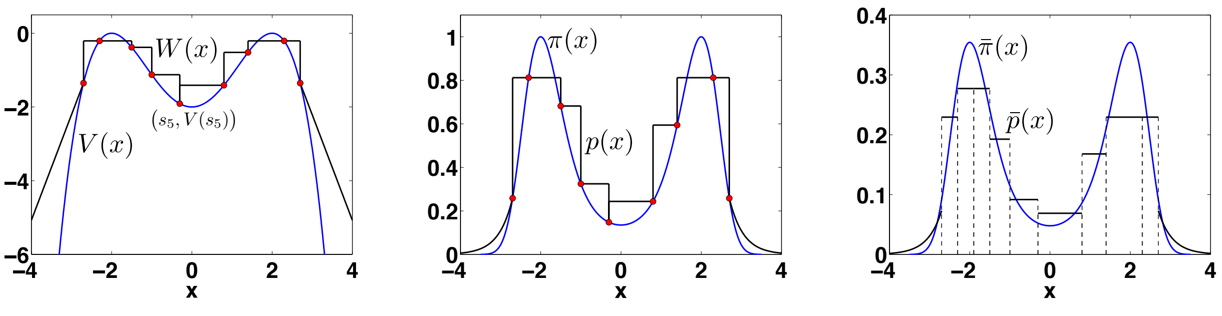


Fig. 1. Example of the proposal construction. **(a)** Construction procedure with $m = 9$ support points, in the log-domain. In this case, we have two light tails (two straight lines in the log-domain). **(b)** The corresponding unnormalized densities $p(x) = e^{W(x)}$ and $\pi(x) = e^{V(x)}$. **(c)** The corresponding normalized densities $\bar{p}(x) \propto p(x)$ and $\bar{\pi}(x) \propto \pi(x)$.

1. Compute the area A_i below each piece, $i = 0, \dots, m$. This is straightforward and can be done analytically, both for the rectangular pieces, $A_i = (s_{i+1} - s_i) \exp(\max[V(s_i), V(s_{i+1})])$ for $i = 1, \dots, m-1$, and for the tails considered here (see A for further details). Then, normalize them,

$$\eta_i = \frac{A_i}{\sum_{j=1}^{m} A_j}, \qquad \text{for} \qquad i = 0, \dots, m.$$

2. Choose a piece (i.e., an index $j^* \in \{0, \dots, m\}$) according to the weights η_i for $i = 0, \dots, m$.
3. Given the index j^*, draw $x' \sim \bar{\phi}_{j^*}(x) \propto \phi_{j^*}(x) = \exp(w_{j^*}(x))$.

It is important to remark that the process of calculating the areas A_i (and then the weights η_i) only has to be performed once, before running the Markov chain.

4.3. Multidimensional construction

In this work, we focus on the application of the FUSS algorithm within a Gibbs sampler, drawing from univariate full-conditional densities. However, FUSS can also be applied to draw directly from a multivariate target density, $\pi(\mathbf{x})$ with $\mathbf{x} \in \mathcal{X}^D \subseteq \mathbb{R}^D$ with $D > 1$. This approach can be useful both for generating samples directly from the target (using FUSS as a stand-alone algorithm) and for updating jointly several variables within a block Gibbs sampler.

In this section, we describe briefly how to build a multidimensional proposal equivalent to the one described in Section 4.1. Let us consider a bounded target density, $\pi(\mathbf{x})$ with $\mathbf{x} \in \mathcal{X}^D \subseteq \mathbb{R}^D$ and $D > 1$. Let us define now D sets of support points, each one forming a grid within $\mathcal{X}$:

$$\begin{aligned} \mathcal{S}_{m_1} &= \{s_{1,1}, \dots, s_{1,m_1}\}, \\ &\vdots \\ \mathcal{S}_{m_d} &= \{s_{d,1}, \dots, s_{d,m_d}\}, \\ &\vdots \\ \mathcal{S}_{m_D} &= \{s_{D,1}, \dots, s_{D,m_D}\}, \end{aligned} \tag{7}$$

where the elements of each set are sorted in ascending order, i.e., $s_{d,1} < s_{d,2} < \dots < s_{d,m_d}$ for all $d = 1, \dots, D$. This grid divides $\mathcal{X}^D$ into $\prod_{d=1}^{D}(m_d - 1)$ hyper-rectangles delimited by 2^D vertices. Namely, a generic hyper-rectangle is defined as

$$\mathcal{I}_{j_1,\dots,j_L} = [s_{1,j_1}, s_{1,j_1+1}] \times [s_{2,j_2}, s_{2,j_2+1}] \times \dots \times [s_{D,j_D}, s_{D,j_D+1}],$$

where $j_d \in \{1, \dots, m_d - 1\}$ with $d = 1, \dots, D$. Therefore, a grid is created with a rectangular boundary defined by

$$\mathcal{R}_B = \prod_{d=1}^{D} [\min s_{d,j_d}, \max s_{d,j_d+1}] \,.$$

Table 4. Pruning algorithm P1

```
1. Given $\mathcal{S}_M = \{s_1, \ldots, s_M\}$, decide the desired number $m$ of final support points (or a rate of reduction $\frac{m}{M}$).
2. Sort $\pi(s_i)$, $i = 1, \ldots, M$, in a decreasing order:
       $\pi(s_{r_1}) \geq \pi(s_{r_2}) \geq .... \geq \pi(s_{r_M})$.
3. Return $\mathcal{S}_m = \{s_{r_1}, s_{r_2}, \ldots, s_{r_m}\}$.
```

Within $\mathcal{R}_B$, a piecewise constant proposal pdf can be easily built by evaluating the target pdf at the vertices of the hyper-rectangle $\mathcal{I}_{j_1,\ldots,j_D}$, in a similar fashion to Eq. (5). Namely, denoting as $\mathcal{V}_{j_1,\ldots,j_D}$ the set of 2^D vertices of $\mathcal{I}_{j_1,\ldots,j_D}$, we can define

$$p(\mathbf{x}|\mathcal{S}_{m_1}, \ldots, \mathcal{S}_{m_D}) = \max_{\mathbf{x} \in \mathcal{V}_{j_1,\ldots,j_D}} \pi(\mathbf{x}), \quad \forall \mathbf{x} \in \mathcal{I}_{j_1,\ldots,j_D} \subset \mathcal{R}_B, \tag{8}$$

i.e., $p(\mathbf{x})$ is constant within $\mathcal{I}_{j_1,\ldots,j_D}$, taking a value equal to the maximum value attained by the target $\pi(\mathbf{x})$ at the 2^D vertices of $\mathcal{I}_{j_1,\ldots,j_D}$. Outside of $\mathcal{R}_B$, the proposal pdf can be constructed by using D independent exponential pieces, i.e.,

$$p(\mathbf{x}|\mathcal{S}_{m_1}, \ldots, \mathcal{S}_{m_D}) = \prod_{d=1}^{D} \exp\left(-\xi_d x_d - \zeta_d\right), \quad \mathbf{x} \notin \mathcal{R}_B, \tag{9}$$

where every ξ_d and ζ_d are obtained applying the interpolation procedure described in Eq. (5) independently to each component of the state space. Note that the function in Eq. (9) can be analytically integrated and it is possible to draw easily from it, following the procedure described in Section 4.2.

5. PRUNING ALGORITHMS

The computational cost of drawing from the proposal pdf $\bar{p}(x)$ (and, as a consequence, the speed of the algorithm) depends on the number of support points used and on the complexity of constructing each piece. Since we are considering simple uniform pieces, the speed is then mainly hindered by the total number of points. Moreover, since we consider initially a thin grid (composed of a large number of support points), the application of some pruning strategy to discard unnecessary points is essential in order to obtain an efficient algorithm. Indeed, the application of the pruning step has two advantages: it speeds up the algorithm and allows the use of a large number of initial support points to capture all the important features of the target.

In the following, we present some possible pruning criteria, sorted in increasing level of complexity. For the sake of simplicity, we assume a bounded target $\pi(x)$. The first two procedures, P1 and P2, are shown in Tables 4 and 5 respectively. They are based on the simple idea of pruning all the points s_r corresponding to "small" values of the target. They are the simplest and fastest approaches, but they also present several limitations: they are not advisable for heavy tailed distributions and their performance is quite sensitive to the dispersion of the target. Therefore, P1 and P2 should be used carefully with complicated pdfs, although they can be very efficient for simple densities.

More refined pruning techniques can be easily devised. An example is the procedure P3 in Table 6. The underlying idea is removing support points in areas where the target is virtually flat, i.e., $|\pi(s_{i+1}) - \pi(s_i)| \approx 0$. Moreover, if a uniform grid is used at the first iteration (i.e., $s_{i+1} - s_i = \epsilon$), the ratio $\frac{\pi(s_{i+1}) - \pi(s_i)}{\epsilon}$ is an estimation of the first derivative of the target. Hence, imposing a condition on $|\pi(s_{i+1}) - \pi(s_i)|$ is equivalent to imposing a condition on the first derivative of $\pi(x)$. Clearly, this procedure could be repeated until achieving the desired rate of reduction, as shown in Table 6, where the procedure is iterated until the pruning condition is no longer verified. Indeed, this criterion can be modified in order to obtain an optimal mimimax pruning strategy, as described in the following section.

5.1. Optimal minimax pruning strategy

The performance of a rejection sampler or an independent Metropolis algorithm is related to the L_1 distance between the target and the proposal (2),

$$D_{p|\pi}(\mathbb{R}) = \int_{-\infty}^{\infty} |p(x|\mathcal{S}_M) - \pi(x)| dx. \tag{12}$$

Table 5. Pruning algorithm P2

1. Given $\mathcal{S}_M = \{s_1, \ldots, s_M\}$, choose a value $\delta \in (0, 1)$.
2. Find all the support points, $s_{k_j} \in \mathcal{S}_M$, such that
$$\pi(s_{k_j}) \leq \delta \max_{1 \leq i \leq M} \pi(s_i). \tag{10}$$
3. Return $S_m = S_M \setminus \{s_{k_1}, \ldots, s_{k_G}\}$, where G is the number of points satisfying the inequality in Eq. (10).

Table 6. Pruning algorithm P3

```
1. Given $\mathcal{S}_M = \{s_1, \ldots, s_M\}$, choose a value $\delta > 0$ and set $\mathcal{S}^{(0)} = \mathcal{S}_M$, $G = M$, $r = 1$ and $L = \max_{0 \leq i \leq M} |\pi(s_{i+1}) - \pi(s_i)|$, with $s_0 = -\infty$ and $s_M = +\infty$.
2. While $G \neq 0$:
    (a) Find all the support points $s_{k_j} \in \mathcal{S}^{(r)}$ such that
            $\left|\pi(s_{k_j+1}) - \pi(s_{k_j})\right| \leq \delta L,$   (11)
    (b) Set $\mathcal{S}^{(r)} = \mathcal{S}^{(r-1)} \setminus \{s_{k_1}, \ldots, s_{k_G}\}$, where $G = |\{s_{k_1}, \ldots, s_{k_G}\}|$.
    (c) Set $r = r + 1$.
3. Return $\mathcal{S}_m = \mathcal{S}^{(r)}$.
```

Taking into account the procedure used to build the proposal and described in detail in Section 4, we can write

$$D_{p|\pi}(\mathbb{R}) = \sum_{j=0}^{M} D_{p|\pi}(\mathcal{I}_j),$$

where $D_{p|\pi}(\mathcal{I}_j)$ denotes the local L_1 distance within the i-th interval ($0 \leq j \leq M$),

$$\begin{aligned} D_{p|\pi}(\mathcal{I}_j) &= \int_{\mathcal{I}_j} |p(x|\mathcal{S}_M) - \pi(x)| dx \\ &= \int_{s_j}^{s_{j+1}} |p(x|\mathcal{S}_M) - \pi(x)| dx, \end{aligned} \tag{13}$$

where $s_0 = -\infty$ and $s_{M+1} = +\infty$, since $\mathcal{I}_0 = (-\infty, s_1]$ and $\mathcal{I}_M = [s_M, \infty)$. Moreover, recall that $s_1 < s_2 < \ldots < s_M$.

The essential consideration now is the following: when a support point is removed, the distance between the target and the proposal generally increases, thus leading to a worse performance of the algorithm. Hence, an optimal criterion for pruning support points is discarding those points that lead to the smallest increase in the L_1 distance between $p(x)$ and $\pi(x)$. This can be seen as a minimax pruning criterion, since its goal is minimizing the maximum increase in $D_{p|\pi}(\mathbb{R})$. Let us assume that the j-th support point ($2 \leq j \leq M-1$) is pruned, then the distance between the target and the proposal becomes now

$$\widetilde{D}_{p|\pi}(\mathbb{R}) = \sum_{\substack{i=0 \\ i \neq j-1, j}}^{M} D_{p|\pi}(E_i) = D_{p|\pi}(\mathbb{R}) + D_{p|\pi}(\mathcal{I}_{j-1} \cup \mathcal{I}_j) - \left[D_{p|\pi}(\mathcal{I}_{j-1}) + D_{p|\pi}(\mathcal{I}_j)\right], \tag{14}$$

where $D_{p|\pi}(\mathcal{I}_{j-1} \cup \mathcal{I}_j)$ denotes the distance between the target and the proposal in the new interval, $\mathcal{I}_{j-1} \cup \mathcal{I}_j$, created by removing the j-th support point. Hence, the optimal support point to be pruned is the one that minimizes the increase in the distance between $p(x)$ and $\pi(x)$, given by (14), i.e.,

$$k = \underset{2 \leq j \leq M-1}{\arg\min} \left\{D_{p|\pi}(\mathcal{I}_{j-1} \cup \mathcal{I}_j) - \left[D_{p|\pi}(\mathcal{I}_{j-1}) + D_{p|\pi}(\mathcal{I}_j)\right]\right\}, \tag{15}$$

Table 7. Pruning algorithm P4

```
1. Choose a value $\delta > 0$. Given $\mathcal{S}_M = \{s_1, ..., s_M\}$, set $\mathcal{S}^{(0)} = \mathcal{S}_M$, $m = M$, $n = 0$ and
   $$L = \max_{1 \leq j \leq \lfloor \frac{m-1}{2} \rfloor} (s_{2j+1} - s_{2j-1})|\pi(s_{2j+1}) - \pi(s_{2j-1})|.$$
2. FOR $r = 1, \ldots, R = \lfloor \frac{m-1}{2} \rfloor$:
   (a) Compute $b_r = (s_{2r+1} - s_{2r-1})|\pi(s_{2r+1}) - \pi(s_{2r-1})|$.
   (b) If $b_r \leq \delta L$, set $\mathcal{S}^{(r)} = \mathcal{S}^{(r-1)} \setminus \{s_{2r}\}$ and $n = n + 1$.
   (c) Otherwise, if $b_r > \delta L$, set $\mathcal{S}^{(r)} = \mathcal{S}^{(r-1)}$.
3. If $n > 0$ set $n = 0$, $\mathcal{S}^{(0)} = \mathcal{S}^{(R)}$, $m = |\mathcal{S}^{(R)}|$ and repeat from step 2.
4. Otherwise, if $n = 0$, return $\mathcal{S}_m = \mathcal{S}^{(R)}$.
```

where the constant term $D_{p|\pi}(\mathbb{R})$ has been removed. Since the proposal $p(x|\mathcal{S}_M)$ is a piecewise constant function, taking values

$$\exp(w_i) = \exp(\max[V(s_i), V(s_{i+1})]),$$

with the exception of the tails, and considering a *continuous* target pdf, it can be easily shown that

$$D_{p|\pi}(\mathcal{I}_j) \leq B_{j-1,j+1} = (s_{j+1} - s_{j-1})|\pi(s_{j+1}) - \pi(s_{j-1})|,$$

i.e., $B_{j-1,j+1}$ is an upper bound on the distance $\widetilde{D}_{p|\pi}(\mathbb{R})$. Therefore, by pruning the support point that minimizes $B_{j-1,j+1}$ we are indeed minimizing the maximum possible increase in the L_1 distance between the target and the proposal pdfs. This observation is the theoretical basis of the minimax optimal pruning strategy detailed in Table 7. The pruning algorithm P4 applies this criterion recursively, performing several iterations through the support set and discarding those points such that $B_{2r-1,2r+1} \leq \delta L$, where $L = \max_{1 \leq j \leq \lfloor \frac{m-1}{2} \rfloor} (s_{2j+1} - s_{2j-1})|\pi(s_{2j+1}) - \pi(s_{2j-1})|$. The algorithm stops automatically when no support points are pruned at one iteration, thus providing the user with a black-box algorithm where only one easily interpretable parameter (δ) has to be chosen. Figure 5.1 depicts several examples of proposals obtained after applying the pruning P4 with different values of δ for a target pdf $\pi(x) = 0.2\mathcal{N}(x; -1, 1) + 0.8\mathcal{N}(x; 3, 1)$, where $\mathcal{N}(x; \mu, \sigma^2)$ denotes a Gaussian pdf with mean μ and variance σ^2.

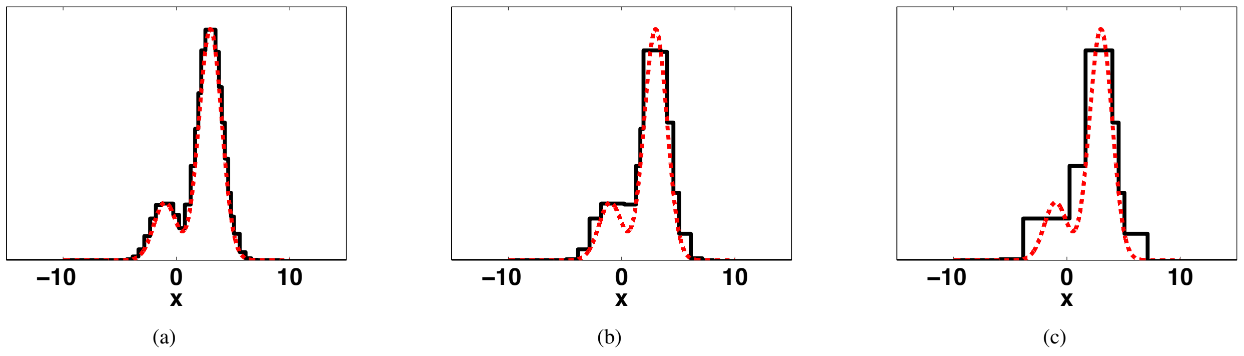


Fig. 2. Examples of proposal construction starting with $S_M = \{-100 : 0.01 : 100\}$ and after applying the pruning algorithm P4: **(a)** with the final number of points $m = 37$, **(b)** $m = 18$, **(c)** and $m = 10$, respectively. The considered target pdf is showed in dashed line.

6. NUMERICAL EXAMPLES FOR UNIVARIATE DENSITIES

In this section, we consider two different univariate target densities in order to test the performance of FUSS as a stand-alone algorithm: the Nakagami pdf (widely used to simulate fading in wireless communication systems) and a Gaussian mixture that contains four modes (three of them very narrow). The FUSS algorithm is compared to other two well-known MCMC methods: the Metropolis-Hastings (MH) algorithm and the slice sampler.

6.1. Unimodal target pdf: Nakami distribution

First of all we consider a Nakagami target distribution, i.e.,

$$\bar{\pi}(x) \propto \pi(x) = x^{2\beta-1} \exp\left(-\frac{\beta}{\Omega}x^2\right), \quad x > 0, \tag{16}$$

with $\beta \geq 0.5$ and $\Omega > 0$. The Nakagami distribution is widely used for the simulation of fading channels in wireless communications (31; 32; 33). When β is an integer or half-integer (i.e., $\beta = \frac{n}{2}$ with $n \in \mathbb{N}$), independent samples can be directly generated through the square root of a sum of squares of n zero-mean i.i.d. Gaussian random variables (33). However, for generic values of β there is not direct method to sample from it, and several alternative approaches have been considered (31; 32). Here, our goal is to estimate the expected value of $X \sim \bar{\pi}(x)$, $\mu = E[X] = \frac{\Gamma(\beta+\frac{1}{2})}{\Gamma(\beta)}\sqrt{\frac{\Omega}{\beta}}$, and the variance, $\sigma^2 = \Omega\left(1 - \frac{1}{\beta}\left(\frac{\Gamma(\beta+\frac{1}{2})}{\Gamma(\beta)}\right)^2\right)$ for $\Omega = 1$ and $\beta = 4.6$.

We apply the FUSS methods using different pruning procedures (P2, P3 and P4) with different values of the threshold parameter δ. We always use an initial set $\mathcal{S}_M = \{0.01, 0.02, 0.03, \ldots, 10^3\}$, i.e., $s_i = 10^{-2} \cdot i$ for $i = 1, \ldots, M = 10^5$ points. We also test a standard MH technique (2) with a random walk proposal

$$\bar{p}(x_k|x_{k-1}) \propto \exp\left\{\frac{-(x_k - x_{k-1})^2}{2\sigma_p^2}\right\},$$

with different values of σ_p. Furthermore, we consider another well-known methodology, the *slice sampling* technique (2, Chapter 8). In order to obtain a fair (and reproducible) comparison of the computational time, we use the corresponding Matlab functions provided by MathWorks: `mhsample.m` and `slicesample.m` respectively.

For all these techniques, we choose $x_0 \in \mathcal{U}[0, 10]$, $K = 5000$, and we consider all the generated samples without removing any burn-in period. We have performed $3 \cdot 10^4$ independent runs. The results are shown in Tables 8 and 9. These tables provide the *Mean Square Errors* (MSE) in the estimation of μ and σ^2, the linear correlation among the samples at lag-1, $\rho(1)$, the acceptance rate ($0 \leq$AR≤ 1) in the rejection sampling (RS) step,[5] the final number of support points after the pruning stage, m, and the computational time for the whole algorithm (normalized w.r.t. the time required by the standard MH method).

We can see that both FUSS algorithms (FUSS-MH and FUSS-RC) always outperform the standard MH and slice sampling techniques. In all cases, the FUSS algorithms are faster, with the only exception of FUSS-RC using P2 and $\delta = 0.9$, which is slightly slower than MH, but faster than slice sampling. Moreover, both FUSS-MH and FUSS-RC attain the performance of an *exact sampler* in the estimation of μ, since

$$\text{MSE}(\mu) \geq \frac{\sigma^2}{K} = 1.0560\ 10^{-5}.$$

This optimal behaviour of both samplers is due to the fact that they are able to draw virtually independent samples (lag-1 correlation $\rho(1) < 10^{-3}$ in all cases for FUSS-RC and $\rho(1) < 10^{-2}$ for FUSS-MH with P4 and $\delta \geq 0.5$), just like the ones that would be obtained from an exact sampler. On the one hand, FUSS-MH is always faster than FUSS-RC, due to the lack of the rejection sampling (RS) test. On the other hand, FUSS-RC provides better results (thanks to this RS test) and is still faster than the standard MH and the slice sampler. Note also that the time required by FUSS-MH always increases when m becomes larger, whereas in FUSS-RC the computational cost can also decrease when m grows due to an improvement in the acceptance rate. Finally, let us remark that the performance of the pruning procedures P3 and P4 is clearly better than P2. As expected, the best one is P4, owing to the fact that P4 selects the final support points according to a minimax optimality criterion, as discussed in Section 5.1. This results in a better quality of the estimators (in terms of MSE), a lower correlation among samples (i.e., a smaller value of $\rho(1)$), and a reduced computational time (especially for FUSS-RC, as the AR increases).[6]

[5]We remark that the AR is not the acceptance probability α in the FUSS-MH and MH methods. The AR is the averaged number of samples accepted in the rejection sampling step of FUSS-RC, which is the only algorithm that includes an RS step. This means that the total number of iterations of FUSS-RC is greater than $K = 5000$ (depending on the acceptance rate), whereas for the other methods AR $= 1$ and $K = 5000$.

[6]The benefit of using P4 can be seen more clearly in the next example, where a multimodal target pdf with narrow modes and a smaller number of iterations K is considered.

		FUSS-MH				**FUSS-RC**			
Pruning		$\delta= 0.9$	$\delta= 0.5$	$\delta= 0.3$	$\delta= 0.01$	$\delta= 0.9$	$\delta= 0.5$	$\delta= 0.3$	$\delta= 0.01$
P2	MSE(μ)	$3.13\ 10^{-5}$	$2.15\ 10^{-5}$	$1.68\ 10^{-5}$	$1.10\ 10^{-5}$	$1.12\ 10^{-5}$	$1.09\ 10^{-5}$	$1.06\ 10^{-5}$	$1.05\ 10^{-5}$
	MSE(σ^2)	$3.94\ 10^{-6}$	$2.25\ 10^{-6}$	$1.65\ 10^{-6}$	$1.05\ 10^{-6}$	$1.21\ 10^{-6}$	$1.13\ 10^{-6}$	$1.12\ 10^{-6}$	$1.12\ 10^{-6}$
	$\rho(1)$	0.4811	0.3288	0.2142	0.0087	$-8.77\ 10^{-4}$	$-2.26\ 10^{-4}$	$-1.30\ 10^{-4}$	$-1.05\ 10^{-4}$
	AR	1	1	1	1	0.3926	0.6354	0.7677	0.9779
	m	23	56	73	139	23	56	73	139
	Time	0.6683	0.6742	0.6781	0.6875	1.3476	0.8263	0.7927	0.7328
P3	MSE(μ)	$1.11\ 10^{-5}$	$1.09\ 10^{-5}$	$1.06\ 10^{-5}$	$1.05\ 10^{-5}$	$1.10\ 10^{-5}$	$1.07\ 10^{-5}$	$1.05\ 10^{-5}$	$1.05\ 10^{-5}$
	MSE(σ^2)	$1.20\ 10^{-6}$	$1.16\ 10^{-6}$	$1.15\ 10^{-6}$	$1.13\ 10^{-6}$	$1.14\ 10^{-6}$	$1.11\ 10^{-6}$	$1.11\ 10^{-5}$	$1.10\ 10^{-6}$
	$\rho(1)$	0.0200	0.0124	0.0077	0.0053	$-2.35\ 10^{-4}$	$-2.50\ 10^{-4}$	$-2.80\ 10^{-4}$	$-1.85\ 10^{-4}$
	AR	1	1	1	1	0.9570	0.9630	0.9787	0.9830
	m	50	88	106	166	50	88	106	166
	Time	0.6712	0.6816	0.6859	0.7019	0.7676	0.7408	0.7424	0.7426
P4	MSE(μ)	$1.10\ 10^{-5}$	$1.09\ 10^{-5}$	$1.06\ 10^{-5}$	$1.05\ 10^{-5}$	$1.10\ 10^{-5}$	$1.07\ 10^{-5}$	$1.05\ 10^{-5}$	$1.05\ 10^{-5}$
	MSE(σ^2)	$1.19\ 10^{-6}$	$1.14\ 10^{-6}$	$1.12\ 10^{-6}$	$1.10\ 10^{-6}$	$1.13\ 10^{-6}$	$1.10\ 10^{-6}$	$1.09\ 10^{-6}$	$1.08\ 10^{-6}$
	$\rho(1)$	0.0133	0.0096	0.0076	0.0053	$1.27\ 10^{-4}$	$-2.61\ 10^{-4}$	$-2.45\ 10^{-4}$	$-2.62\ 10^{-4}$
	AR	1	1	1	1	0.9666	0.9769	0.9800	0.9832
	m	71	109	121	177	71	109	121	177
	Time	0.6849	0.6937	0.6957	0.7105	0.7339	0.7397	0.7380	0.7502

Table 8. Results of FUSS for the Nakagami target ($\beta = 4.6$ and $\Omega = 1$) with different pruning procedures (P2, P3 and P4) and $K = 5000$. The table shows the MSE in the estimation of the mean and variance (μ and σ^2), the correlation at lag-1 ($\rho(1)$), the acceptance rate (AR) for FUSS-RC (for FUSS-MH AR=1, since there is no RS test), the number of support points in the final grid (m), and the time required by the whole algorithm (normalized w.r.t. the time required by a standard MH sampler).

		$\sigma_p= 0.2$	$\sigma_p= 0.5$	$\sigma_p= 0.8$	$\sigma_p= 1$	$\sigma_p= 2$	$\sigma_p= 3$	$\sigma_p= 4$
MH	MSE(μ)	0.0021	$3.95\ 10^{-4}$	$1.98\ 10^{-4}$	$1.52\ 10^{-4}$	$1.43\ 10^{-4}$	$1.90\ 10^{-4}$	$2.52\ 10^{-4}$
	MSE(σ^2)	0.0513	0.0091	0.0039	0.0027	$9.20\ 10^{-4}$	$6.18\ 10^{-4}$	$5.69\ 10^{-4}$
	$\rho(1)$	0.8935	0.7495	0.7433	0.7611	0.8389	0.8808	0.9043
	AR	1	1	1	1	1	1	1
	Time	1	1	1	1	1	1	1
Slice sampling								
MSE(μ)= $1.24\ 10^{-5}$	MSE(σ^2)= $2.27\ 10^{-5}$		$\rho(1)$= 0.0229		AR=1	Time= 2.5037		

Table 9. Results of the standard MH and slice sampling methods for the Nakagami target ($\beta = 4.6$ and $\Omega = 1$) with different standard deviations (σ_p) and $K = 5000$. The meaning of the parameters displayed is the same as in Table 8.

6.2. Multimodal target pdf: mixture of Gaussians

In this second example we test the FUSS approach to draw from a multimodal target density. More specifically, we consider a mixture of 4 Gaussian pdfs,

$$\bar{\pi}(x) = \sum_{i=1}^{4} \mathcal{N}(x; \mu_i, \sigma_i^2), \tag{17}$$

where $\mu_1 = -7$, $\sigma_1 = 0.1$, $\mu_2 = 0$, $\sigma_2 = 1$, $\mu_3 = 8$, $\sigma_3 = 0.2$, $\mu_4 = 15$ and $\sigma_4 = 0.1$. This choice leads to 4 clearly separated modes (three of them quite narrow), as shown in Figure 3(a).[7] We perform K iterations of the chain, taking into account all the generated samples (K) to perform the estimation (without removing any burn-in period). We test again the performance of the FUSS algorithm, the standard MH method with a Gaussian random walk proposal pdf (with variance σ_p^2) and the slice sampler, as in the previous example. Furthermore, we also apply two more sophisticated MCMC techniques: the *Metropolis adjusted Langevin algorithm* (MALA) (34) and the *Hamiltonian Monte Carlo* (HMC) method (35). Both methodologies incorporate the gradient information in their respective proposal mechanisms. Hence, they should provide a better performance whenever it is possible to evaluate the gradient of the target. However, they are also more costly (compared to a standard MH method) due to the evaluations of the gradient required. We consider a Gaussian pdf with a standard deviation $\sigma_p = 20$ for the diffusion noise

[7]Note that, from a Monte Carlo point of view, the problem becomes harder as the number of modes increases. This is due to the fact that our interest here is not discovering the number of modes (as in spectral estimation), but being able to construct a proposal that mimicks the target as closely as possible. Therefore, having clearly separated and narrow modes results in a more challenging problem than having partially overlapping (and thus wider) modes.

in MALA and for the kinetic distribution in HMC.[8] We test different values of the discretization parameter (ϵ) for MALA, and different values of ϵ and L (length of the trajectory) in HMC.[9]

For the sake of simplicity, we have only considered the FUSS-MH algorithm in this example.[10] For all the different algorithms, the initial state of the chain has been chosen as $x_0 \sim \mathcal{U}([-10, 20])$. As in the previous example, we have used the Matlab functions directly provided by MathWorks for the MH and slice samplers (`mhsample.m` and `slicesample.m` respectively). For HMC we have used the code provided in (36, Chapter 30), whereas for MALA we have developed our own Matlab code, since there is no code publicly available as far as we know. For FUSS-MH, we use an initial set $\mathcal{S}_M = \{-10^3, -10^3 + 0.01, \ldots, 10^3 - 0.01, 10^3\}$, i.e. $\mathcal{S}_M = \{s_1, \ldots, s_M\}$ with $s_i = s_1 + (i-1)\epsilon$ for $s_1 = -10^3$, $\epsilon = 10^{-2}$ and $M = 2 \cdot 10^5 + 1$ points.

Table 10 shows the results (using $K = 200$ samples, different pruning algorithms (P2, P3 and P4) and values of δ) for FUSS-MH: MSE in the estimation of the mean (μ) and variance (σ^2) of the target, correlation at lag-1 ($\rho(1)$), number of final support points (m), and time required by the whole algorithm (normalized w.r.t. the time required by the standard MH algorithm using $K = 200$). The results of the standard MH (using again $K = 200$ samples and different values of σ_p) are provided in Table 11, whereas Table 12 shows the results (in terms of MSE and elapsed time) for the MH and slice samplers when the number of iterations of the Markov chain (K) is increased from $K = 200$ up to $K = 2 \cdot 10^4$. Finally, Tables 13 and 14 provide the results for MALA and HMC respectively. Furthermore, in Figures 3(b) and (c) we depict respectively the log-MSE and log-time of MH and slice sampling as function of K, compared to the results for FUSS-MH (which do not depend on K). All the results are averaged over $3 \cdot 10^4$ runs.

		FUSS-MH			
Pruning		$\boldsymbol{\delta}$**= 0.9**	$\boldsymbol{\delta}$**= 0.5**	$\boldsymbol{\delta}$**= 0.3**	$\boldsymbol{\delta}$**= 0.01**
P2	MSE(μ)	11.38	7.33	4.30	0.4680
	MSE(σ^2)	409.02	288.35	211.78	19.52
	$\rho(1)$	0.9142	0.8917	0.8419	0.1468
	m	18	46	103	662
	Time	1.03	1.04	1.06	1.17
P3	MSE(μ)	1.7683	1.1368	0.8686	0.3679
	MSE(σ^2)	78.96	49.82	35.92	15.3513
	$\rho(1)$	0.6693	0.5284	0.4224	0.0296
	m	61	99	135	497
	Time	1.20	1.27	1.31	1.35
P4	MSE(μ)	0.3786	0.3662	0.3638	0.3526
	MSE(σ^2)	15.53	15.31	15.10	14.53
	$\rho(1)$	0.0446	0.0306	0.0247	0.0093
	m	145	195	223	605
	Time	1.23	1.26	1.28	1.40

Table 10. Results of FUSS-MH for the mixture of Gaussians target in Eq. (17) using different pruning procedures (P2, P3 and P4) and $K = 200$. The table shows the MSE in the estimation of the mean and variance (μ and σ^2), the correlation at lag-1 ($\rho(1)$), the number of support points in the final grid (m), and the time required by the whole algorithm (normalized w.r.t. the time required by the standard MH sampler with $K = 200$).

		$\boldsymbol{\sigma_p}$**= 2**	$\boldsymbol{\sigma_p}$**= 8**	$\boldsymbol{\sigma_p}$**= 14**	$\boldsymbol{\sigma_p}$**= 20**	$\boldsymbol{\sigma_p}$**= 25**	$\boldsymbol{\sigma_p}$**= 30**
MH	MSE(μ)	34.92	24.64	19.62	19.46	19.80	20.57
	MSE(σ^2)	4.43 10^3	1.36 10^3	1.13 10^3	1.09 10^3	1.37 10^3	1.51 10^3
	$\rho(1)$	0.7481	0.9465	0.9425	0.9424	0.9445	0.9459
	Time	1	1	1	1	1	1

Table 11. Results of the standard MH method for the mixture of Gaussians target in Eq. (17) using $K = 200$. The table shows the MSE in the estimation of the mean and variance (μ and σ^2), the correlation at lag-1 ($\rho(1)$), and the time required by the whole algorithm (normalized to 1 in order to compare with the alternative methods). The best results are obtained for $\sigma_p = 20$, but the MSE is always larger than using FUSS-MH.

[8] The value $\sigma_p = 20$, used both in MALA and HMC, corresponds to the optimal scale parameter for a random walk MH method obtained in the simulations (see Table 11).

[9] Note that MALA can also be interpreted as a special case of HMC with $L = 1$ (35).

[10] As shown in the previous numerical example, FUSS-RC provides in general better results at the expense of a slight increase in the computational time.

		$\mathbf{K = 200}$	$\mathbf{K = 10^3}$	$\mathbf{K = 2 \cdot 10^3}$	$\mathbf{K = 5 \cdot 10^3}$	$\mathbf{K = 10^4}$	$\mathbf{K = 1.5 \cdot 10^4}$	$\mathbf{K = 2 \cdot 10^4}$
MH $\boldsymbol{\sigma_p}$**= 20**	MSE(μ)	19.46	6.16	3.16	1.35	0.6735	0.4540	0.3341
	MSE(σ^2)	1.09 10^3	263.36	123.37	44.69	22.48	14.45	10.79
	Time	1	4.55	8.75	21.51	41.48	61.44	81.06
Slice	MSE(μ)	22.13	6.08	3.25	1.26	0.6136	0.4177	0.2911
	MSE(σ^2)	1.10 10^3	171.13	68.91	23.98	10.48	7.40	5.85
	Time	2.37	11.04	21.62	52.66	103.78	154.38	204.57

Table 12. Results of the slice sampling and the standard MH method for the mixture of Gaussians target in Eq. (17) using $\sigma_p = 20$ and varying the total number of iterations of the chain (K) from $K = 200$ up to $K = 2 \cdot 10^4$. The table shows the MSE in the estimation of the mean and variance (μ and σ^2), and the time required by the whole algorithm (normalized w.r.t. the time required by the standard MH sampler with $K = 200$).

		$\mathbf{K = 1.5 \cdot 10^4}$								
		ϵ**= 0.05**	ϵ**= 0.1**	ϵ**= 0.5**	ϵ**= 0.7**	ϵ**= 0.8**	ϵ**= 0.9**	ϵ**= 1**	ϵ**= 1.2**	ϵ**= 1.5**
MALA	MSE(μ)	0.4686	0.5038	0.5269	0.4135	0.4616	0.4701	0.4413	0.4748	0.4828
	MSE(σ^2)	14.09	16.80	18.03	14.61	14.69	15.36	11.19	13.75	14.73
	Time	61.59	61.59	61.59	61.59	61.59	61.59	61.59	61.59	61.59

Table 13. Results of MALA (34) for the mixture of Gaussians target in Eq. (17) using $K = 1.5 \cdot 10^4$, $\sigma_p = 20$ (the standard deviation of the driving noise), and varying the parameter ϵ. The table shows the MSE in the estimation of the mean and variance (μ and σ^2), and the time required by the whole algorithm (normalized w.r.t. the time required by the standard MH sampler with $K = 200$).

		$\mathbf{K = 1.5 \cdot 10^4}$						
		$\mathbf{L = 2}$	$\mathbf{L = 3}$	$\mathbf{L = 5}$	$\mathbf{L = 8}$	$\mathbf{L = 9}$	$\mathbf{L = 10}$	$\mathbf{L = 15}$
HMC ϵ**= 0.1**	MSE(μ)	1.06	0.9829	0.9453	0.5058	0.3987	0.6217	0.8448
	MSE(σ^2)	21.98	20.56	18.83	15.42	16.74	39.42	61.15
	Time	62.63	73.28	79.94	131.68	141.08	156.76	215.58
HMC ϵ**= 0.2**	MSE(μ)	0.5295	0.3445	1.05	0.8356	0.5183	0.9699	0.4694
	MSE(σ^2)	10.52	10.21	25.11	25.75	35.53	28.41	21.47
	Time	62.63	73.28	79.94	131.68	141.08	156.76	215.58
HMC ϵ**= 0.5**	MSE(μ)	0.4408	0.5173	0.5213	0.3259	1.12	0.6557	0.6134
	MSE(σ^2)	12.53	13.76	13.91	21.40	24.93	30.13	25.05
	Time	62.63	73.28	79.94	131.68	141.08	156.76	215.58

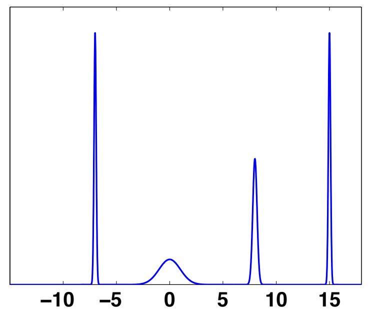
Table 14. Results of HMC for the mixture of Gaussians target in Eq. (17) using the code in (36, Chapter 30) with $K = 1.5 \cdot 10^4$, $\sigma_p = 20$ (σ_p^2 represents the inverse of the mass in the kinetic energy equation), and varying the parameters ϵ (the discretization step size of the leapfrog method within HMC (35)) and L (the total number of leapfrog steps). The table shows the MSE in the estimation of the mean and variance (μ and σ^2), and the time required by the whole algorithm (normalized w.r.t. the time required by the standard MH sampler with $K = 200$).

Taking into account all these results we can obtain the following conclusions:

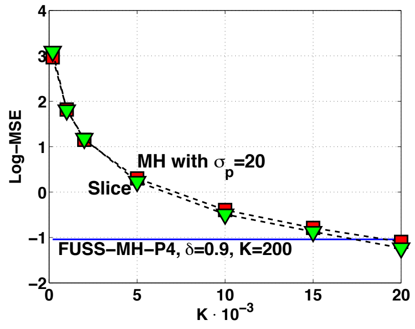
- For a fixed value of K, FUSS-MH outperforms MH, slice sampling, MALA and HMC. All of these methods are able to obtain similar results to FUSS when properly parameterized, but they are much more sensitive to the choice of the parameters and they require a much larger number of iterations ($K = 2 \cdot 10^4$ for MH and slice sampling, and $K = 1.5 \cdot 10^4$ for MALA and HMC), thus implying a much higher computational cost.

- The results also highlight the importance of using an adaptive or self-tuned method: the results vary considerably with the choice of the parameters for MH, slice sampling, MALA and HMC. Furthermore, FUSS-MH attains the theoretical limit (in terms of MSE in the estimation of μ) for independent samples, MSE(μ) $\geq \frac{\sigma^2}{K} = \frac{68.765}{200} = 0.3438$ for $K = 200$, whereas the MSE of the competing methods is always much higher than this limit, MSE(μ) $\geq \frac{\sigma^2}{K} = \frac{68.765}{2 \cdot 10^4} = 3.438 \cdot 10^{-3}$ for $K = 2 \cdot 10^4$. This is due to the fact that FUSS-MH is able to adapt the entire shape of the proposal pdf according to the target, thus providing virtually independent samples.

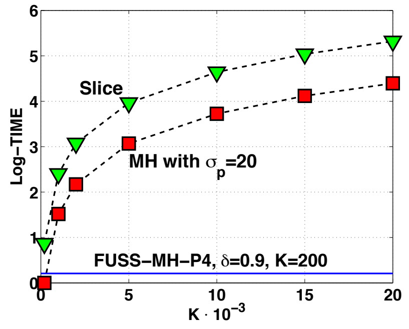
- Tables 13 and 14 show that, unlike FUSS, MALA and HMC suffer when applied to multimodal distributions with narrow modes. In this specific scenario, the use of gradient information can even reduce the exploratory behaviour of the chain



(a)



(b)



(c)

Fig. 3. **(a)** The multimodal target pdf $\pi(x)$ in Eq. (17). **(b)** Log-MSE as function of $\frac{K}{10^3}$ for the slice (triangles) and the MH (squares) methods, with $\sigma_p = 20$. The solid line shows the log-MSE, $\log(0.3526) = -1.0424$, of FUSS-MH-P4 with $\delta = 0.01$, achieved using only $K = 200$. **(c)** Log-time (normalized w.r.t. the time required by the standard MH sampler with $K = 200$) as function of K for the slice (triangles) and the MH (squares) methods. The solid line corresponds to FUSS-MH-P4 with $\delta = 0.01$ and $K = 200$.

for some values of the parameters. The results for MALA and HMC are comparable to the standard MH method (with a similar number of samples, $K = 2 \cdot 10^4$) and to FUSS-MH-P4($\delta = 0.01$) using only $K = 200$ samples. Namely, using FUSS we can save more than 98% of the computational time.

- FUSS-MH using the pruning procedure P4 attains virtually the theoretical bound for the MSE (i.e., 0.3438 for the estimation of μ using $K = 200$), achievable using independent samples. Similar results are attained using different choices of the threshold δ (showing the robustness of the approach w.r.t. the parameterization), due to the fact that FUSS-MH-P4 produces virtually independent samples.

7. NUMERICAL EXAMPLES FOR MULTIVARIATE DENSITIES

In this section, we consider four multi-variate densities in order to test the performance of FUSS-within-Gibbs: two bivariate multimodal pdfs, a challenging problem of parameter estimation in a chaotic system, and a high-dimensional and multi-modal problem in financial signal processing. The FUSS algorithm is compared to other two well-known MCMC-within-Gibbs methods: the Metropolis-Hastings (MH) algorithm and adaprive rejection Metropolis sampling (ARMS). Furthermore, the bivariate version of the stand-alone FUSS algorithm is also tested in Section 7.2, showing that it can outperform all the MCMC-within-Gibbs approaches.

The FUSS-within-Gibbs algorithm is summarized in Table 15. The algorithm in Table 15 is essentially a standard Gibbs sampler, where FUSS is used to draw from the full-conditionals at each iteration. Note that the set of support points is always re-calculated before applying the FUSS algorithm. This leads to an optimal design of the proposal at each iteration and does not result in a large computational cost, as shown in the simulations. However, if the target is not expected to change substantially between two consecutive Gibbs iterations, the update could be performed only once out of P iterations (as done in the griddy Gibbs sampler), thus reducing even more the computational cost. Note also that the alternative methods tested (MH-within-Gibbs and ARMS-within-Gibbs) are similar to the algorithm in Table 15, but removing steps 2(b) and 2(c) and using instead some other proposal function, as described later.

7.1. FUSS-within-Gibbs: simple bivariate pdf

In order to show the importance of using an adequate MCMC technique *within* a Gibbs sampler, in this section we provide a simple example using different methods for drawing from the full-conditional pdfs. Let us consider the bivariate target density

$$\bar{\pi}(x_1, x_2) \propto \pi(x_1, x_2) = \exp\left(-\frac{(x_1^2 - A + Bx_2)^2}{4} - \frac{x_1^2}{2\sigma_1^2} - \frac{x_2^2}{2\sigma_2^2}\right), \tag{18}$$

where $A = 16$, $B = 10^{-2}$ and $\sigma_1^2 = \sigma_2^2 = \frac{10^4}{2}$. Densities with this analytic form are often used in the literature to evaluate the performance of different Monte Carlo algorithms (37; 38). We apply a Gibbs sampler to draw from $\bar{\pi}(x_1, x_2)$. To generate

Table 15. General structure of the FUSS-within-Gibbs algorithm.

```
1. Initialization: Select the number of Gibbs iterations, $N_G$, and the number of FUSS iterations per Gibbs iteration, $K$. Choose a set of support points, $\mathcal{S}_M = \{s_1, \ldots, s_M\}$, such that $s_1 < s_2 < \ldots < s_M$.
2. FOR $t = 1, \ldots, N_G$:
       FOR $i = 1, \ldots, D$:
       (a) Let the target be $\pi(x) = \pi(x_i^{(t)} | x_1^{(t)}, \ldots, x_{i-1}^{(t)}, x_{i+1}^{(t)}, \ldots, x_D^{(t-1)})$.
       (b) Apply some of the pruning procedures described in Section 5, using the initial support set $\mathcal{S}_M$ and the target $\pi(x)$, to obtain the final support set $\mathcal{S}_i^{(t)}$.
       (c) Build a proposal function $p(x|\mathcal{S}_i^{(t)})$, using some appropriate pre-defined mechanism, as described in Section 4.
       (d) Perform $K$ steps of an MCMC method using $p(x|\mathcal{S}_i^{(t)})$ as proposal pdf. Take the last one as $x_i^{(t)}$.
```

samples from the full conditional pdfs, $\pi(x_1|x_2)$ and $\pi(x_2|x_1)$, we use FUSS-MH-P4 with $\delta = 0.9$, starting with a uniform grid $\mathcal{S}_M = \{-10^4, -10^4+0.01, \ldots, 10^4-0.01, 10^4\}$, i.e. $\mathcal{S}_M = \{s_1, \ldots, s_M\}$ with $s_i = s_1+(i-1)\epsilon$ for $s_1 = -10^4$, $\epsilon = 10^{-2}$ and $M = 2 \cdot 10^6+1$ points. We also apply the ARMS method within the Gibbs sampler (16). ARMS uses an interpolation procedure to build the proposal using a set of support points, similarly to FUSS. However, unlike FUSS, ARMS starts with few support points and incorporates new ones adaptively. We choose for ARMS the initial set $\{-10, -6, -4.3, -0.01, 3.2, 3.8, 4.3, 7, 10\}$. Finally, we also consider a standard MH algorithm with a random walk proposal

$$\bar{p}(x_i^{(t)}|x_i^{(t-1)}) \propto \exp\left(-\frac{(x_i^{(t)} - x_i^{(t-1)})^2}{2\sigma_p^2}\right),$$

with $\sigma_p \in \{1, 2, 10\}$, initial state $x_i^{(0)} \in \mathcal{U}([-5, 5])$, and $x_i^{(t)}$ denoting the value of x_i ($i \in \{1, 2\}$) at the t-th iteration of the MH algorithm. We perform $N_G = 2000$ iterations of the Gibbs sampler, using all the samples to estimate four statistics which involve the first four moments of the target: mean, variance, skewness and kurtosis. In each iteration, we draw K samples from each full-conditional, taking only the last one and continuing the Gibbs cycle.

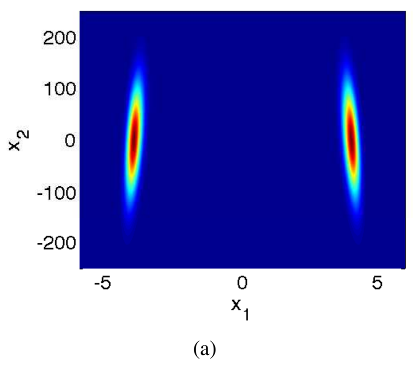
Table 16 provides the numerical results (averaged over 1000 runs) for the *Mean Absolute Error* (MAE) and the time required by the Gibbs sampler. The time is normalized by considering 1 to be the time elapsed when using ARMS-within-Gibbs with $K = 50$. Figure 4(a) illustrates the target $\bar{\pi}(\mathbf{x})$, whereas Figures 4(b)-(c) depict the MAE and the time required by ARMS and FUSS-MH-P4 as function of K. Observing Table 16 and Fig. 4(b), we notice that FUSS-MH-P4 outperforms ARMS for all values of K in the estimation of the four central moments: FUSS-MH-P4 always achieves MAEs close to zero. As shown also in Fig. 4(c), FUSS-MH-P4 is slightly slower than ARMS for $K \in \{3, 5, 10\}$ due to the pruning stage. However, the computational time in FUSS-MH-P4 remains virtually constant with K, whereas in ARMS it increases with K, since ARMS keeps adding new support points to improve the proposal pdf, hence becoming more costly for $K > 15$. Regarding the use of MH-within-Gibbs, the results depend largely on the choice of the variance of the proposal, σ_p^2, showing the need of adaptive or self-tuned MCMC strategies. Indeed, for an inadequate scale parameter (e.g., $\sigma_p = 1$ or $\sigma_p = 2$), even the use of $K = 1000$ provides bad results. On the other hand, when a good σ_p is selected (i.e., $\sigma_p = 10$), MH with $K = 100$ and $K = 1000$ provides virtually the same performance as FUSS-MH-P4, although at the expense of an increased computational cost.

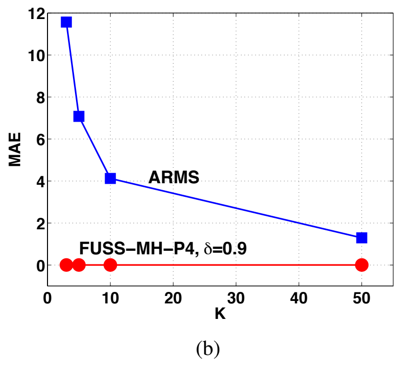
7.2. FUSS-within-Gibbs and bivariate FUSS: donut target pdf

In order to assess the performance of the FUSS algorithm when the variables of the target are strongly correlated, we consider here a bivariate *donut* target density:

$$\pi(x_1, x_2) = \exp\left(-\frac{(x_1^2 - A + Bx_2^2)^2}{4}\right), \tag{19}$$

with $A = 10$ and $B = 0.01$. Figure 5(a) depicts the proposal approximation of $\pi(x_1, x_2)$, obtained using the multi-variate version of FUSS described in Section 4.3, which is virtually undistinguishable from the true target pdf. Moreover, Figures 5(a) and (b) illustrate two slices of the target function $\pi(x_1, x_2)$, obtained fixing $x_2 = 10$ and $x_1 = 3$ respectively. Note that both the support and the shape of the conditional pdfs vary considerably.





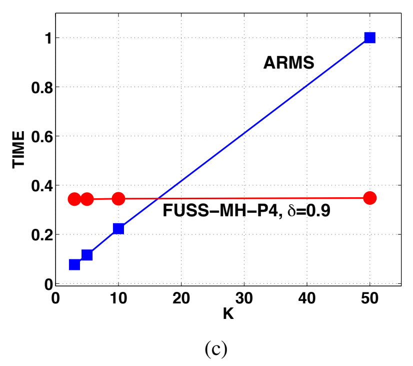


Fig. 4. **(a)** Contour plot of the target pdf $\bar{\pi}(x_1, x_2)$. **(b)** MAE in estimation of the kurtosis (first component) as a function of K, using ARMS-within-Gibbs (squares) and FUSS-within-Gibbs (circles). **(c)** Time required by ARMS (squares) and FUSS (circles) within the Gibbs sampler as a function of K (normalized w.r.t. the time required by ARMS-within-Gibbs using $K = 50$).

			MAE				
Technique		K	**Mean**	**Variance**	**Skewness**	**Kurtosis**	**Time**
FUSS-MH-P4 $\delta = 0.9$		3	0.0735	0.0365	0.0369	0.0022	0.343
		5	0.0735	0.0361	0.0367	0.0022	0.343
		10	0.0724	0.0354	0.0365	0.0021	0.345
		50	0.0721	0.0355	0.0364	0.0021	0.348
ARMS		3	3.408	11.580	3.384	11.572	0.077
		5	3.151	9.839	2.650	7.079	0.116
		10	2.798	7.665	2.024	4.124	0.223
		50	1.918	3.407	1.134	1.292	1.000
MH	$\sigma_p = 1$	100	3.509	12.308	3.671	13.666	0.540
	$\sigma_p = 2$	100	1.756	3.077	0.9782	0.9633	0.540
	$\sigma_p = 10$	100	0.0756	0.0376	0.0368	0.0025	0.540
	$\sigma_p = 1$	1000	3.508	12.302	3.665	13.624	3.229
	$\sigma_p = 2$	1000	1.601	2.560	0.8741	0.7691	3.229
	$\sigma_p = 10$	1000	0.0743	0.0360	0.0363	0.0021	3.229

Table 16. Mean absolute error (MAE) in the estimation of the first component of four statistics (mean, variance skewness and kurtosis) of $\bar{\pi}(x_1, x_2)$, and simulation time for the whole algorithm (normalized w.r.t. the time required by ARMS using $K = 50$). All the techniques are used within a Gibbs sampler ($N_G = 2000$ total iterations), performing K iterations per Gibbs iteration for each full-conditional.

We use different Monte Carlo techniques in order to compute the variances in the diagonal of the covariance matrix of $\pi(x_1, x_2)$, $\sigma_1^2 \approx 5$ and $\sigma_2^2 \approx 500$ respectively. We calculate the *relative error* (that is the ratio of the mean absolute error (MAE) and the true value) for each component and compute the arithmetic mean of the two values obtained.

First of all, we apply two MCMC-within-Gibbs techniques (performing $N_G = 1000$ iterations of the Gibbs sampler): FUSS-MH-P2 with $\delta = 0.01$ and an MH scheme. For the FUSS technique, we consider only $K = 3$ steps for each full conditional and $\mathcal{S}_M = \{-10^4 : 0.1 : 10^4\}$. For the MH scheme, we set $K = 10$ and use a random walk proposal $\bar{p}(x_i^{(t)}|x_i^{(t-1)}) \propto \exp\left(-\frac{(x_i^{(t)}-x_i^{(t-1)})^2}{2\sigma_p^2}\right)$ for $i \in \{1, 2\}$. We test several values of σ_p. The initial value is always set to $x_i^{(0)} = -2$ for both (FUSS and MH). Furthermore, we apply directly the bivariate version of FUSS-MH-P2 with $\delta = 0.01$, using the proposal constructed following the procedure described in Section 4.3. We use the initial support set $\mathcal{S}_{M_i} = \{-10^4 : 0.1 : 10^4\}$ for $i \in \{1, 2\}$, and different number of iterations $K \in \{30, 50, 100, 1000\}$. We compare with a standard bivariate MH algorithm using again a random walk Gaussian proposal pdf, i.e., $\bar{p}(\mathbf{x}^{(t)}|\mathbf{x}^{(t-1)}) = \mathcal{N}(\mathbf{x}^{(t)}|\mathbf{x}^{(t-1)}, \mathbf{\Sigma}_p)$, with $\mathbf{\Sigma}_p = [\sigma_p^2 \;\, 0; \; 0 \;\, \sigma_p^2]^\top$ and testing again several values of σ_p.

Tables 17 and 18 show the results in terms of relative error. Note that the FUSS schemes always outperform the MH methods, even with an optimal choice of the scale parameter σ_p. Furthermore, FUSS obtains better results using a smaller number of iterations K, thus saving a good deal of valuable computational power. Moreover, the cost of drawing from the

FUSS proposal pdf is small due to the application of the pruning procedure. Finally, note that the stand-alone version of FUSS provides better results than FUSS-within-Gibbs (which in turn provides better results than MH and MH-within-Gibbs). This is due to the strong correlation among x_1 and x_2, which slows down the convergence of the Gibbs sampler, and might be solved (at least partially) using an appropriate reparameterization or a better sweep strategy (e.g., a random scan approach) for the Gibbs sampler (2).

		Relative Error						
Technique within Gibbs	**K**	$\sigma_p = 2$	$\sigma_p = 5$	$\sigma_p = 8$	$\sigma_p = 10$	$\sigma_p = 12$	$\sigma_p = 15$	$\sigma_p = 20$
MH	10	0.6654	0.4994	0.0949	0.1118	0.1889	0.2479	0.3046
FUSS-MH-P2, $\delta = 0.01$	3	0.0856						

Table 17. Mean relative error in the estimation of the trace of the covariance matrix, for the donut target pdf ($\sigma_1^2 = 5$ and $\sigma_2^2 = 500$) using MH or FUSS within a Gibbs sampler with $N_G = 1000$ iterations. We use $K = 10$ steps for each iteration of Gibbs for MH, whereas only $K = 3$ for FUSS.

		Relative Error								
Technique	**K**	$\sigma_p = 2$	$\sigma_p = 5$	$\sigma_p = 8$	$\sigma_p = 10$	$\sigma_p = 12$	$\sigma_p = 15$	$\sigma_p = 18$	$\sigma_p = 20$	$\sigma_p = 25$
Bidim. MH	1000	0.5889	0.4209	0.3168	0.2819	0.2560	0.2337	0.2161	0.2301	0.2426
Bidim. FUSS-MH-P2, $\delta = 0.01$	30	0.1947								
	50	0.1249								
	100	0.0731								
	1000	0.0225								

Table 18. Mean relative error in the estimation of the trace of the covariance matrix, for the donut target pdf ($\sigma_1^2 = 5$ and $\sigma_2^2 = 500$) applying directly MH and FUSS-MH in the bidimensional space, with $K = 1000$ iterations for MH and different values of K for the FUSS scheme.

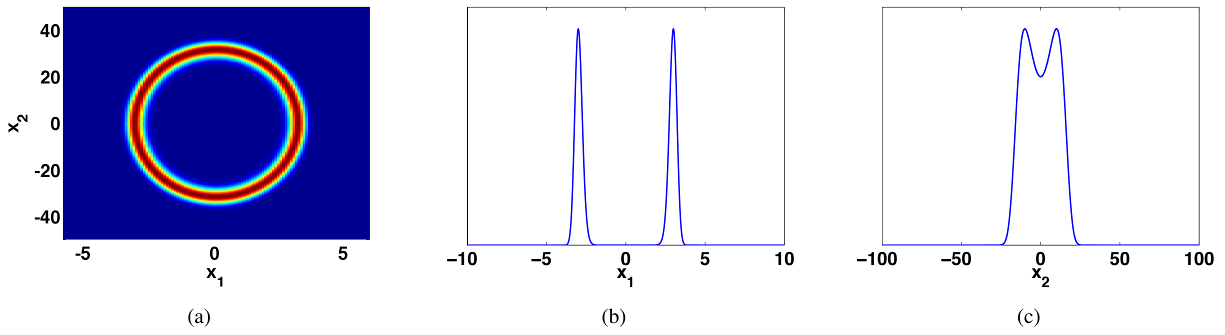


Fig. 5. **(a)** Bidimensional proposal function obtained by the FUSS procedure: it is a excellent approximation of the target function $\pi(x_1, x_2)$. **(b)** Slice of the target function $\pi(x_1, x_2)$ keeping fixed $x_2 = 10$. **(c)** Slice of the target function $\pi(x_1, x_2)$ keeping fixed $x_1 = 3$.

7.3. FUSS within Gibbs: parameter estimation in a chaotic system

The FUSS algorithms are also able to overcome severe computational issues in some statistical frameworks where other, even more sophisticated, MCMC techniques seem to fail (39; 40). In order to show this capability, we address the estimation of fixed parameters of a chaotic system, which is considered a very challenging problem in the literature (39; 40; 41). This type of systems are often utilized for modelling the evolution of population sizes, for instanc in ecology (39). Let us consider a logistic map (42) perturbed by multiplicative noise,

$$z_{t+1} = R\left[z_t\left(1 - \frac{z_t}{\Omega}\right)\right]\exp(\epsilon_t), \quad \epsilon_t \sim \mathcal{N}(0, \lambda^2), \tag{20}$$

with $z_1 \sim \mathcal{U}([0,1])$ and for some unknown parameters $R > 0$ and $\Omega > 0$. Let us assume that a sequence $z_{1:T} = [z_1, \dots, z_T]$ is observed and, for the sake of simplicity, let us consider that λ is known. Under these circumstances the likelihood function is given by

$$p(z_{1:T}|R,\Omega) = \prod_{t=1}^{T-1} p(z_{t+1}|z_t, R, \Omega),$$

where, defining $g(z_t, R, \Omega) = R\left[\, z_t\left(1 - \frac{z_t}{\Omega}\right)\right]$, we have

$$p(z_{t+1}|z_t, R, \Omega) \propto \left|\frac{g(z_t,R,\Omega)}{z_{t+1}}\right| \exp\left(-\frac{\log\left(\frac{z_{t+1}}{g(z_t,R,\Omega)}\right)^2}{2\lambda^2}\right),$$

if $g(z_t,R,\Omega) > 0$, and $p(z_{t+1}|z_t,R,\Omega) = 0$, if $g(z_t,R,\Omega) \leq 0$. Considering uniform priors, $R \sim \mathcal{U}([0,10^4])$ and $\Omega \sim \mathcal{U}([0,10^4])$, our goal is computing the mean of the bivariate posterior pdf, $p(R,\Omega|z_{1:T}) \propto p(z_{1:T}|R,\Omega)$, which corresponds to the minimum mean squared error (MMSE) estimate of the parameters.

In the experiments, we set $R = 3.7$, $\Omega = 0.4$ and $T = 20$. Furthermore, we take into account different values of λ of the same order of magnitude as considered in (39). Then we apply FUSS-MH-P2, with $\delta = 10^{-3}$ and $K = 10$, within a Gibbs sampler using only $N_G = 50$ iterations ($S_M = \{10^{-4}, 2\cdot 10^{-4}, \dots, 20\}$).[11] We also consider an MH-within-Gibbs approach with random walk proposal, $\bar{p}(x_t|x_{t-1}) \propto \exp\left(\frac{-(x_t - x_{t-1})^2}{2\sigma_p^2}\right)$, with two different values of $\sigma_p \in \{1,2\}$. The initial states of the chains are chosen randomly from $\mathcal{U}([1,5])$ and $\mathcal{U}([0.38,1.5])$ in order to start sampling from the two full conditionals, $p(R|\Omega, z_{1:T})$ and $p(\Omega|R, z_{1:T})$ respectively. In order to compare the performance of both approaches, we also perform an approximate computation of the true value of the mean (of the corresponding posterior pdf) via an expensive deterministic numerical integration procedure, and so we are able to calculate the MSE obtained by FUSS-MH-P2 and MH.

The results, averaged over 1000 independent runs, are shown in Table 19. It can be clearly seen that FUSS-MH-P2 achieves a very small MSE in the estimation of the two desired parameters (especially in the case of Ω) for any value of λ. Comparing with the MSE obtained by the MH algorithm, the benefit of building a proposal tailored to the full-conditionals (as done by FUSS) becomes apparent. Figures 6(a)–(b) provide two examples of conditional log-pdfs, whereas Figure 6(c) shows the "sharp" conditional density corresponding to Figure 6(b). This pdf resembles a delta function: even using sophisticated adaptive techniques it is difficult to recognize the mode of this kind of target pdf.

		$\lambda = 0.001$	$\lambda = 0.005$	$\lambda = 0.01$	$\lambda = 0.05$	$\lambda = 0.08$	$\lambda = 0.10$
FUSS-MH-P2	MSE(R)	0.0071	0.0089	0.0093	0.0138	0.0150	0.0778
	MSE(Ω)	$5.01\ 10^{-5}$	$6.15\ 10^{-5}$	$6.15\ 10^{-5}$	$5.26\ 10^{-5}$	$7.33\ 10^{-5}$	$1.78\ 10^{-4}$
MH (σ_p= 1)	MSE(R)	0.6830	0.7264	0.7067	1.1631	1.3298	1.3293
	MSE(Ω)	0.0373	0.0402	0.0423	0.0399	0.0471	0.0440
MH (σ_p= 2)	MSE(R)	1.3566	1.4906	1.4247	2.0015	2.3042	2.2401
	MSE(Ω)	0.0897	0.1117	0.1041	0.0989	0.1089	0.1125

Table 19. MSEs in estimation of R and Ω using FUSS-MH-P2 and MH inside a Gibbs sampler, with $\delta = 10^{-3}$, $K = 10$ and $N_G = 50$. The observed sequence $z_{1:T}$ is generated with $R = 3.7$, $\Omega = 0.4$, $T = 20$ and different values of λ.

7.4. FUSS within Gibbs: jump diffusion model for stock market index

We consider the inference problem in a model which is combination of the stochastic volatility model and infinite-activity Lévy jumps as in (43). This model outperforms the popular Black-Scholes model by (44) and Heston model by (45) in describing the dynamics of real-life stock prices. Specifically, suppose Y_t is the daily log-return of some stock market index, i.e. $Y_t = \log(S_t)$, where S_t is the daily index value, and let v_t be the instantaneous variance of return, $t \in \mathbb{N}$. We assume that the joint dynamics of the index return and variance can be described by the following stochastic equations,

$$\begin{aligned} Y_{t+1} - Y_t &= \left(r - \frac{1}{2}v_t + \phi_J(-i) + \eta_s v_t\right)\Delta + \\ &\quad + \sqrt{v_t\Delta}\epsilon_{t+1,1} + J_{t+1}, \\ v_{t+1} - v_t &= k(\theta - v_t)\Delta + \sigma_v\sqrt{v_t\Delta}\epsilon_{t+1,2}, \end{aligned} \tag{21}$$

[11]Note that we use the simplest pruning procedure (P2) in order to show that, even in this case, FUSS provides a good performance.

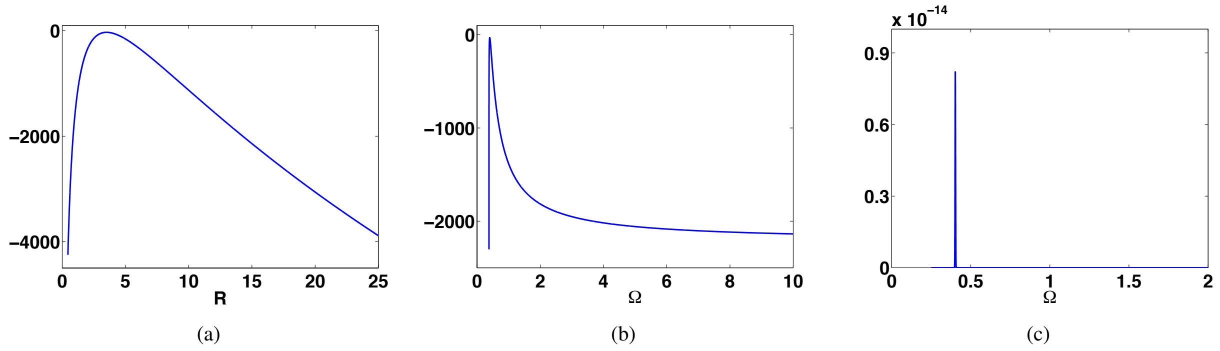


Fig. 6. **(a)-(b)** Examples of (unnormalized) conditional log-pdfs with $\lambda = 0.1$ and considering $T = 20$ observations. **(a)** Fixing $\Omega = 4$. **(b)** Fixing $R = 0.7$. **(c)** The (unormalized) conditional pdf corresponding to Figure (b). Even advanced and adaptive MCMC techniques often fail in drawing samples from this kind of sharp densities.

where r is the risk-free interest rate, $\eta_s v_t$ is the risk premium, and Δ is the time interval of the discretization on the daily basis (we set $\Delta = \frac{1}{252}$). In the variance process, k measures the speed of mean reverting, θ is the long-term mean of variance v_t, and σ_v is the volatility of variance. $\epsilon_{t,1}$ and $\epsilon_{t,2}$ are two noise perturbations distributed according a bivariate Gaussian pdf, specifically,

$$(\epsilon_{t,1}, \epsilon_{t,2}) \sim \mathcal{N}([0,0]^\top, \Sigma),$$

where $\Sigma = [1,\ \rho;\ \rho,\ 1]^\top$. We also consider a Lévy jump J_t in the process of Y_t, and $\phi_J(-i)$ is its jump compensator, defined as $E^Q[e^{iuJ_t}] = e^{-t\phi_J(u)}$ where Q is the so-called risk-neutral probability measure. In our model setting, we employ the Variance Gamma process proposed by (46) as J_t, and for simplicity we assume the jump-relevant parameters γ, ν, and σ remain unchanged under Q, i.e.,

$$\begin{aligned} J_t &= \gamma G_t(1,\nu) + \sigma W_{G_t(1,\nu)}, \\ \phi_J(-i) &= \frac{\log(1 - \gamma\nu - \frac{1}{2}\nu\sigma^2)}{\nu}, \end{aligned} \tag{22}$$

where G_t is an independent Gamma process that we consider known, and $W_{G_t(1,\nu)}$ is a Brownian motion subordinated by G_t. In the market we observe the daily index return $Y_t, t = 1, \ldots, T$. To calibrate the model, we need to filter all the latent variables v_t, J_t, and the fixed parameters $\Theta = \{k, \theta, \rho, \sigma_v, \eta_s, \gamma, \nu, \sigma\}$. In particular, since all the latent variables are varied from day to day, for example $\mathbf{v} = [v_1, \ldots, v_T]^\top$ with $T = 252$ (in one year), the estimation of this model is very challenging owing to the high dimension.

We apply a Gibbs sampler to draw from the complete posterior distribution. The full-conditional pdfs of several latent variables and parameters have a standard form as Gaussian, inverse Gamma and Gamma densities so that direct sampling methods are available. However, the conditional pdfs corresponding to v_t, $t = 1, ..., 252$, and the parameter ν, have a complex analytic form, thus we apply FUSS-RC-P4 for drawing from them. We also apply a standard MH algorithm, with a Gaussian random walk proposal $\bar{p}(x_t|x_{t-1}) \propto \exp\left(-\frac{(x_k - x_{k-1})^2}{2\sigma_p^2}\right)$, to compare the performance. We set $x_0 = 0.02$ (regarding v) or $x_0 = 0.5$ (regarding ν), $K = 3$ for both methods. We set $\sigma_p = 0.01$ for MH. We chose these values of x_0 and σ_p after several attempts, in order to provide the best results of the MH method. Furthermore, for FUSS-RC-P4, we chose $s_1 = 0.0001$, $s_M = 6$, $\Delta_{s_i} = s_{i+1} - s_i = 0.0001$, $\delta = 0.01$. In this example, we change the number of iterations of the Gibbs sampler, $N_g = \{10, 100, 500, 5 \cdot 10^3\}$. For the MH method, we also consider $N_g = 10^4$.

We simulate artificial data by the system using as true parameters $k = 4$, $\theta = 0.02$, $\sigma_v = 0.3$, $\rho = -0.8$, $\eta_s = 0.4$, $\nu = 0.1$, $\gamma = 0.1$, $\sigma = 0.1$, $Y_0 = 7$, $v_0 = 0.02$, and $r = 0.02$ (we recall that the jump component G_t is assumed known). Then we compute the MSEs in the estimation of ν and the entire vector $\mathbf{v}$, i.e., the dimension of the problem is $D = 253$.

The results, averaged over 100 independent runs, are provided in Table 20. FUSS-RC-P4 always outperforms the standard MH method. We can also observe that FUSS-RC-P4 with only $N_g = 10$ provides a smaller MSE($\mathbf{v}$) than the MH method with $N_g = 10^4$.

Figure 5, which compares the filtered paths by FUSS-RC-P4 (dashed line) and MH (dotted-dashed line) with the simulated true path (solid line) of v_t, demonstrates that MH can yield quite poor results if there is no much prior information on v_t. In particular, given that it is a flat curve for initial variance with $v_t = 0.02$ for all $t = 1 \ldots 252$, MH did not achieve the true

Technique	N_g	**MSE(v)**	**MSE(ν)**
FUSS-RC-P4	10	3.989 10^{-5}	1.460 10^{-2}
	100	2.542 10^{-5}	1.642 10^{-3}
	500	1.798 10^{-5}	3.783 10^{-4}
	5000	1.138 10^{-5}	9.119 10^{-5}
MH	10	3.330 10^{-4}	1.289 10^{-1}
	100	3.392 10^{-4}	3.093 10^{-2}
	500	3.289 10^{-4}	1.264 10^{-2}
	5000	2.754 10^{-4}	7.514 10^{-3}
	10^5	7.660 10^{-5}	1.368 10^{-3}

Table 20. MSEs obtained by FUSS-RC-P4 and MH methods for different number of iterations N_g of the Gibbs sampler.

variance dynamics being sensitive to initial flat values whereas FUSS-RC-P4 captured the actual variance dynamics relatively well.

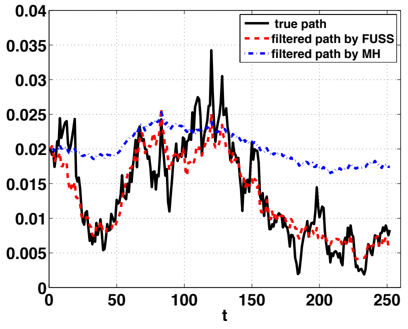


Fig. 7. Filtering of variance path v_t, $t = 1, \ldots, 252$, by FUSS-RC-P4 and $N_g = 5000$ and MH with $N_g = 10^5$. The path is the average of the filtered paths in 100 different experiments.

8. CONCLUSIONS AND FUTURE LINES

In this work, we have introduced FUSS ("Fast Universal Self-tuned Sampler"), a novel and extremely efficient MCMC sampler for drawing from univariate densities. Although the proposal is tailored to the target by means of an initial automatic optimization procedure, FUSS is *not* an adaptive MCMC algorithm, hence its ergodicity is not an issue. We have tested the new technique in several scenarios with different target distributions, including two very challenging practical applications: estimation of the parameters of a chaotic map and a financial signal processing problem. The performance achieved by the new methodology is close to that of an exact sampler, yielding virtually i.i.d. samples. Consequently, the novel technique outperforms other well-known MCMC methods (such as the Metropolis-Hastings (MH) algorithm, the slice sampler, the Metropolis adjusted Langevin algorithm (MALA) or the Hamiltonian Monte Carlo (HMC) method), both in terms of accuracy and speed. Furthermore, the dependence on the initial parameters is also drastically reduced with respect to other MCMC techniques. Numerical simulations also show a clear benefit of using the FUSS schemes within a Gibbs sampler. Future lines include developing more efficient representations for the multi-variate version of FUSS (e.g., using Chua's canonical PWL model (47)), adapting it to sequential inference applications and addressing other problems in machine learning where MCMC-based inference is difficult to implement efficiently (e.g., inference in generalized Poisson mixed models (48) or Gaussian process latent variable models (49)).

9. ACKNOWLEDGEMENTS

This work has been supported by the ERC grant 239784 and AoF grant 251170, the Spanish government through the CONSOLIDER-INGENIO 2010 program (CSD2008-00010) and DISSECT project (TEC2012-38058-C03-01), and the BBVA Foundation through the MG-FIAR project, obtained in "I Convocatoria de Ayudas Fundación BBVA a Investigadores, Innovadores y Creadores Culturales".

References

[1] J. S. Liu, Monte Carlo Strategies in Scientific Computing, Springer, 2004.

[2] C. P. Robert, G. Casella, Monte Carlo Statistical Methods, Springer, 2004.

[3] L. Jing, P. Vadakkepat, Interacting MCMC particle filter for tracking maneuvering target, Digital Signal Processing 20 (2010) 561–574.

[4] W. J. Fitzgerald, Markov chain Monte Carlo methods with applications to signal processing, Signal Processing 81 (1) (2001) 3–18.

[5] M. F. Bugallo, S. Xu, P. M. Djurić, Performance comparison of EKF and particle filtering methods for maneuvering targets, Digital Signal Processing 17 (2007) 774–786.

[6] P. M. Djurić, J. H. Kotecha, J. Zhang, Y. Huang, T. Ghirmai, M. F. Bugallo, J. Míguez, Particle filtering, IEEE Signal Processing Magazine 20 (5) (2003) 19–38.

[7] J. Míguez, Analysis of selection methods for cost-reference particle filtering with applications to maneuvering target tracking and dynamic optimization, Digital Signal Processing 17 (2007) 787–807.

[8] F. Liang, C. Liu, R. Caroll, Advanced Markov Chain Monte Carlo Methods: Learning from Past Samples, Wiley Series in Computational Statistics, England, 2010.

[9] J. Kotecha, P. M. Djurić, Gibbs sampling approach for generation of truncated multivariate Gaussian random variables, in: Proc. of the IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP), Vol. 3, 1999, pp. 1757–1760.

[10] K. R. Koch, Gibbs sampler by sampling-importance-resampling, Journal of Geodesy 81 (9) (2007) 581–591.

[11] J. K. O. Ruanaidh, W. J. Fitzgerald, Numerical Bayesian Methods Applied to Signal Processing, Springer, 1996.

[12] K. P. Murphy, Machine Learning: A Probabilistic Perspective, The MIT Press, 2012.

[13] G. O. Roberts, S. K. Sahu, Updating schemes, correlation structure, blocking and parameterization for the Gibbs sampler, Journal of the Royal Statistical Society: Series B 59 (2) (1997) 291–317.

[14] W. R. Gilks, Derivative-free Adaptive Rejection Sampling for Gibbs Sampling, Bayesian Statistics 4 (1992) 641–649.

[15] W. R. Gilks, P. Wild, Adaptive Rejection Sampling for Gibbs Sampling, Applied Statistics 41 (2) (1992) 337–348.

[16] W. R. Gilks, N. G. Best, K. K. C. Tan, Adaptive Rejection Metropolis Sampling within Gibbs Sampling, Applied Statistics 44 (4) (1995) 455–472.

[17] W. R. Gilks, R. Neal, N. G. Best, K. K. C. Tan, Corrigidum: Adaptive Rejection Metropolis Sampling within Gibbs Sampling, Applied Statistics 46 (4) (1997) 541–542.

[18] R. Meyer, B. Cai, F. Perron, Adaptive rejection Metropolis sampling using Lagrange interpolation polynomials of degree 2, Computational Statistics and Data Analysis 52 (7) (2008) 3408–3423.

[19] L. Martino, J. Read, D. luengo, Independent doubly adaptive rejection Metropolis sampling, in: Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), 2014, pp. 8048–8052.

[20] L. Martino, J. Read, D. Luengo, Improved adaptive rejection Metropolis sampling algorithms, arXiv:1205.5494 (2012) 1–32.

[21] L. Martino, R. Casarin, F. Leisen, D. Luengo, Adaptive sticky generalized Metropolis, arXiv:1308.3779 (2013) 1–44.

[22] B. Cai, R. Meyer, F. Perron, Metropolis-Hastings algorithms with adaptive proposals, Statistics and Computing 18 (2008) 421–433.

[23] S. Olver, A. Townsend, Fast inverse transform sampling in one and two dimensions, arXiv:1307.1223.

[24] W. Shao, G. Guo, F. Meng, S. Jia, An efficient proposal distribution for Metropolis-Hastings using a b-splines technique, Computational Statistics and Data Analysis 53 (2013) 465–478.

[25] L. Holden, R. Hauge, M. Holden, Adaptive independent Metropolis-Hastings, The Annals of Applied Probability 19 (1) (2009) 395–413.

[26] J. M. Keith, D. P. Kroese, G. Y. Sofronov, Adaptive independence samplers, Statistics and Computing 18 (4) (2008) 409–420.

[27] C. Ritter, M. A. Tanner, Facilitating the Gibbs sampler: The Gibbs stopper and the griddy-Gibbs sampler, Journal of the American Statistical Association 87 (419) (1992) 861–868.

[28] A. Doucet, X. Wang, Monte Carlo methods for signal processing, IEEE Signal Processing Magazine 22 (6) (2005) 152–170.

[29] L. Tierney, Exploring posterior distributions using Markov Chains, Computer Science and Statistics: Proceedings of IEEE 23rd Symp. Interface (1991) 563–570.

[30] L. Tierney, Markov chains for exploring posterior distributions, The Annals of Statistics 22 (4) (1994) 1701–1728.

[31] N. C. Beaulieu, C. Cheng, Efficient Nakagami-m fading channel simulation, IEEE Transactions on Vehicular Technology 54 (2) (2005) 413–424.

[32] D. Luengo, L. Martino, Almost rejectionless sampling from Nakagami-m distributions ($m \geq 1$), IET Electronics Letters 48 (24) (2012) 1559–1561.

[33] J. G. Proakis, Digital Communications, McGraw-Hill, Singapore, 1995.

[34] G. Roberts, R. Tweedie, Exponential convergence of Langevin distributions and their discrete approximations, Bernoulli 2 (1996) 341–363.

[35] R. Neal, Chapter 5 of the Handbook of Markov Chain Monte Carlo, Edited by Steve Brooks, Andrew Gelman, Galin Jones, and Xiao-Li Meng; Chapman and Hall/CRC Press, 2011.

[36] D. J. C. MacKay, Information Theory, Inference and Learning Algorithms, Cambridge University Press, 2003.

[37] H. Haario, E. Saksman, J. Tamminen, Adaptive proposal distribution for random walk Metropolis algorithm, Computational Statistics 14 (1999) 375–395.

[38] L. Martino, J. Read, On the flexibility of the design of multiple try Metropolis schemes, Computational Statistics 28 (6) (2013) 2797–2823.

[39] C. T. Perretti, S. B. Munch, G. Sugihara, Model-free forecasting outperforms the correct mechanistic model for simulated and experimental data, Proceedings of the National Academy of Sciences (PNAS) 110 (13) (2013) 5253–5257.

[40] C. T. Perretti, S. B. Munch, G. Sugihara, Reply to Hartig and Dormann: The true model myth, Proceedings of the National Academy of Sciences (PNAS) 110 (42) (2013) E3976–E3977.

[41] F. Hartig, C. F. Dormann, Does model-free forecasting really outperform the true model?, Proceedings of the National Academy of Sciences (PNAS) 110 (42) (2013) E3975.

[42] A. Boyarsky, P. Góra, Law of Chaos, Birkhöuser, Boston (USA), 1997.

[43] H. Li, M. Wells, C. Yu, A Bayesian analysis of return dynamics with Lévy jumps, Review of Financial Studies 21 (5) (2008) 2345–2378.

[44] F. Black, The pricing of commodity contracts, Journal of financial economics 3 (1) (1976) 167–179.

[45] S. L. Heston, A closed-form solution for options with stochastic volatility with applications to bond and currency options, Review of financial studies 6 (2) (1993) 327–343.

[46] D. Madan, P. Carr, E. Chang, The variance gamma process and option pricing, European finance review 2 (1) (1998) 79–105.

[47] S. Kang, L. O. Chua, A global representation of multidimensional piecewise-linear functions with linear partitions, IEEE Transactions on Circuits and Systems 25 (11) (1978) 938–940.

[48] J. Luts, M. P. Wand, Variational inference for count response semiparametric regression, arXiv:1309.4199.

[49] M. Titsias, N. D. Lawrence, Bayesian Gaussian process latent variable model, in: Proc. 13th Int. Conf. on Artificial Intelligence and Statistics (AISTATS), Chia Laguna Resort, Sardinia (Italy), 2010, pp. 844–851.

A. CONSTRUCTION OF THE TAILS FOR THE PROPOSAL

The choice of the tails for the proposal is important for two reasons: (a) to accelerate the convergence of the chain to the target (especially for heavy-tailed target distributions) and (b) to increase the robustness of the method w.r.t. the initial choice of the set $\mathcal{S}_M$. Indeed, often the construction of tails with a bigger area below them can reduce the dependence on a specific choice of the set of initial support points. In general, a good choice is to build tails such that $p(x) \geq \pi(x)$ for $x \in \mathcal{I}_0$ and $x \in \mathcal{I}_m$. This is always possible when the target pdf has light tails (i.e., convex tails in the log-domain) and also for many heavy-tailed targets that appear in real-world applications. In the following, we show two practical approaches to build both light-tailed and heavy-tailed distributions.

A.1. Light Tails

In order to use light tails, we can simply use two exponential pieces for the first and last intervals of the proposal:

$$p(x|\mathcal{S}_m) = e^{h_0 x + b_0}, \ \ \forall x \in \mathcal{I}_0,$$
$$p(x|\mathcal{S}_m) = e^{h_m x + b_m}, \ \ \forall x \in \mathcal{I}_m.$$

The first linear function, $w_0(x) = h_0 x + b_0$, is the straight line passing through the points $(s_1, V(s_1))$ and $(s_2, V(s_2))$, whereas the second linear function, $w_m(x) = h_m x + b_m$, is the straight line passing through $(s_{m-1}, V(s_{m-1}))$ and the last point $(s_m, V(s_m))$. Note that we can compute the area below each piece analytically, $A_0 = \frac{1}{h_0}\exp(h_0 s_1 + b_0)$ and $A_m = -\frac{1}{h_m}\exp(h_m s_m + b_m)$, and that we can also easily draw from each exponential tail using the inversion method (2).

A.2. Heavy Tails

For heavy tailed constructions, there are also several possibilities. For instance, here we propose to use Pareto pieces, which have the following analytic form

$$p(x|\mathcal{S}_m) = e^{\rho_0}\frac{1}{|x-\mu_0|^{\gamma_0}}, \ \ \forall x \in \mathcal{I}_0,$$
$$p(x|\mathcal{S}_m) = e^{\rho_m}\frac{1}{|x-\mu_m|^{\gamma_m}}, \ \ \forall x \in \mathcal{I}_m,$$

with $\gamma_j > 1$, $j \in \{0, m\}$. In the log-domain, this results in

$$w_0(x) = \rho_0 - \gamma_0 \log(|x-\mu_0|), \text{ for } x \in \mathcal{I}_0,$$
$$w_m(x) = \rho_m - \gamma_m \log(|x-\mu_m|), \text{ for } x \in \mathcal{I}_m.$$

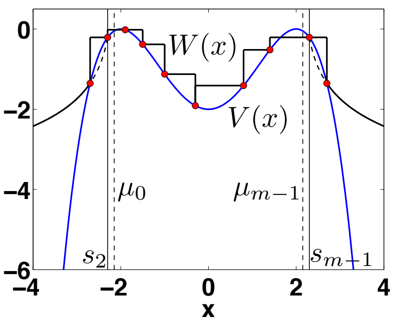
Fixing the parameters μ_j, $j \in \{0, m\}$, the remaining parameters, ρ_j and γ_j, are set in order to satisfy the passing conditions through the points $(s_1, V(s_1))$ and $(s_2, V(s_2))$, and through the points $(s_{m-1}, V(s_{m-1}))$ and $(s_m, V(s_m))$, respectively. The parameters μ_j can be arbitrarily chosen by the user, as long as they fulfill the following inequalities:

$$\mu_0 > s_2, \quad \mu_m < s_{m-1}.$$

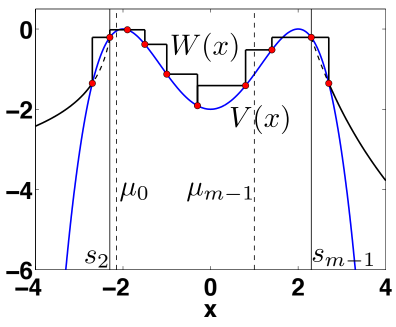
Values of μ_j such that $\mu_0 \approx s_2$ and $\mu_m \approx s_{m-1}$ yield small values of γ_j (close to 1) and, as a consequence, fatter tails. Larger differences in $|\mu_0 - s_2|$ and $|\mu_m - s_{m-1}|$ yield $\gamma_j \to +\infty$, i.e., lighter tails. Several examples of this type of construction are illustrated in Figure 8. As in the previous case, we can compute analytically the integral of $p(x|\mathcal{S}_m)$ in $\mathcal{I}_0$ and $\mathcal{I}_m$:

$$A_0 = \frac{e^{\rho_m}}{\gamma_m - 1}\frac{1}{(s_m - \mu_m)^{\gamma_m - 1}},$$
$$A_m = \frac{e^{\rho_0}}{\gamma_0 - 1}\frac{1}{(\mu_0 - s_1)^{\gamma_0 - 1}}.$$

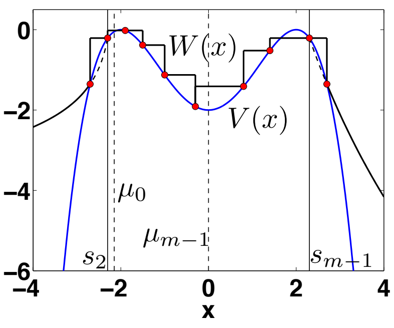
We can also draw samples easily from each Pareto tail using the inversion method (2).



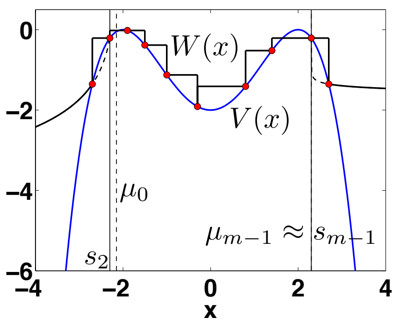
(a)



(b)



(c)



(d)

Fig. 8. Example of the construction of heavy tailed proposal pdfs in the log-domain. **(a)** Example with a specific choice of the parameters $\mu_0 > s_2$ and $\mu_{m-1} < s_{m-1}$ ($m = 10$ in figure). **(b)–(c)** Alternative right log-tail constructions, decreasing the parameter μ_{m-1}. For $\mu_{m-1} \to -\infty$, the log-tail tends to become a straight line passing through $(s_{m-1}, V(s_{m-1}))$ and $(s_m, V(s_m))$. **(d)** Construction of the right log-tail with $\mu_{m-1} \approx s_{m-1}$. In this case, $\mu_{m-1} \to s_{m-1}$, i.e., it tends to become a constant line.