Luengo *et al. EURASIP Journal on Advances in Signal Processing* (2020) 2020:25
https://doi.org/10.1186/s13634-020-00675-6

EURASIP Journal on Advances in Signal Processing

REVIEW Open Access

A survey of Monte Carlo methods for parameter estimation

David Luengo[1*], Luca Martino[2,3], Mónica Bugallo[4], Víctor Elvira[5] and Simo Särkkä[6]

Abstract

Statistical signal processing applications usually require the estimation of some parameters of interest given a set of observed data. These estimates are typically obtained either by solving a multi-variate optimization problem, as in the maximum likelihood (ML) or maximum a posteriori (MAP) estimators, or by performing a multi-dimensional integration, as in the minimum mean squared error (MMSE) estimators. Unfortunately, analytical expressions for these estimators cannot be found in most real-world applications, and the Monte Carlo (MC) methodology is one feasible approach. MC methods proceed by drawing random samples, either from the desired distribution or from a simpler one, and using them to compute consistent estimators. The most important families of MC algorithms are the Markov chain MC (MCMC) and importance sampling (IS). On the one hand, MCMC methods draw samples from a proposal density, building then an ergodic Markov chain whose stationary distribution is the desired distribution by accepting or rejecting those candidate samples as the new state of the chain. On the other hand, IS techniques draw samples from a simple proposal density and then assign them suitable weights that measure their quality in some appropriate way. In this paper, we perform a thorough review of MC methods for the estimation of static parameters in signal processing applications. A historical note on the development of MC schemes is also provided, followed by the basic MC method and a brief description of the rejection sampling (RS) algorithm, as well as three sections describing many of the most relevant MCMC and IS algorithms, and their combined use. Finally, five numerical examples (including the estimation of the parameters of a chaotic system, a localization problem in wireless sensor networks and a spectral analysis application) are provided in order to demonstrate the performance of the described approaches.

Keywords: Statistical signal processing, Bayesian inference, Monte Carlo methods, Metropolis-Hastings algorithm, Gibbs sampler, MH-within-Gibbs, Adaptive MCMC, Importance sampling, Population Monte Carlo

1 Introduction

1.1 Motivation: parameter estimation in statistical signal processing applications

Statistical inference deals with the estimation of a set of unknowns given a collection of observed data contaminated by noise and possibly by some other types of distortions and interferences [1]. In many signal processing applications, this typically amounts to inferring some static parameters of interest from the noisy observations [2–4][1]. For instance, in *denoising* applications, the aim is reconstructing the original signal (e.g., an audio recording or an image) from the noisy observations [5]. An extended version of this problem occurs in *blind deconvolution*, where a noisy filtered signal is available and the goal is to recover both the unknown filter and the input [6][2]. Finally, as a third application, *target localization/tracking* in wireless sensor networks requires estimating/tracking

*Correspondence: david.luengo@upm.es
[1]Universidad Politécnica de Madrid, ETSIST, C/Nikola Tesla, s/n, 28031 Madrid, Spain
Full list of author information is available at the end of the article

[1]Note that, although we concentrate here on the parameter estimation problem, other related signal processing problems (model selection, prediction, classification, etc.) can be addressed in a similar way.
[2]Closely related problems are blind equalization (where the input is a digital communications signal), blind identification (where the main goal is to recover the filter) [7], and blind source separation (where multiple input signals have to be separated from a collection of multiple outputs) [8].

the location of the target (maybe jointly with some parameters of the system, such as the noise variance, the propagation constant or even the position of the sensors) from measurements recorded by the sensors [9, 10].

In the *Bayesian framework*, all the aforementioned problems are addressed by formulating a *prior distribution*, which should gather all the available information about the parameters of interest external to the data, and assuming an input-output model (the *likelihood*), that incorporates our knowledge or lack thereof on the way in which the observed data relate to the unknown parameters [11]. Then, *Bayes theorem* allows us to obtain the *posterior distribution*, which takes into account both the effect of the prior information and the observed data in an optimal way. Finally, the desired Bayesian point estimators are obtained by minimizing a pre-defined cost function that can typically be expressed either as some integral measure with respect to (w.r.t.) the posterior or as some optimization problem. For instance, the well-known minimum mean squared error (MMSE) estimator corresponds to the conditional mean of the parameters of interest given the data (i.e., the expected value of the posterior distribution), whereas the maximum a posteriori (MAP) estimator corresponds to the value of the parameters where the posterior attains its highest peak[3]. Note that a similar procedure is also followed by *frequentist methods* (i.e., in the end they also attempt to minimize some cost function which is either expressed as some integral measure or formulated as an optimization problem), even though they are completely different from a conceptual point of view. Indeed, the frequentist maximum likelihood (ML) estimator simply corresponds to the Bayesian MAP estimator with a uniform prior. Hence, although we focus on Bayesian approaches in the sequel, all the techniques mentioned here are also applicable in a frequentist context.

Unfortunately, obtaining closed-form expressions for any of these estimators is usually impossible in real-world problems. This issue can be circumvented by using approximate estimators (e.g., heuristic estimators in the frequentist context or variational Bayesian approximations) or by restricting the class of models that were considered (e.g., in the case of Bayesian inference by using only conjugate priors). However, with the increase in computational power and the extensive development of Monte Carlo methods, Bayesian inference has been freed from the use of a restricted class of models and much more complicated problems can now be tackled in a realistic way. In the following section, we briefly review the history of Monte Carlo methods, pointing out the key developments and some of the most relevant algorithms that will be described in detail throughout the paper. Note that, apart from MC methods, there are several alternative techniques for approximating integrals in statistical inference problems [12]: asymptotic methods, multiple quadrature approaches, and subregion adaptive integration. However, these schemes cannot be applied in high-dimensional problems and MC algorithms become the only feasible approach in many practical applications. Another related topic which is not covered here due to space constraints is variational Bayesian inference. However, the interested reader can check some of the existing tutorials (and references therein) for an overview these methods [13, 14].

1.2 Framework: Monte Carlo methods

The so-called Monte Carlo (MC) methods encompass a large class of stochastic simulation techniques that can be used to solve many optimization and inference problems in science and engineering. Essentially, MC methods proceed by obtaining a large pool of potential values of the desired parameters and substituting the integrations by sample averages. In practice, these parameter values can be obtained either by physically replicating the desired experiment or by characterizing it probabilistically and generating a set of random realizations.

The origin of MC methods can be traced back to Buffon's experiments to compute an empirical value on the *St. Petersburg game*,[4] and the formulation of his famous experiment (nowadays commonly known as *Buffon's needle*) to calculate the value of π [16, 17][5]. Buffon's needle experiment became quite well known after it was mentioned by Laplace in 1812 [18], and several scientists attempted to replicate his experiment during the last quarter of the ninteenth century [19–22][6]. Meanwhile, other statisticians were experimenting with different mechanisms to generate random numbers (e.g., using cards, a roulette or dice) to verify empirically, through some kind of primitive stochastic simulation, their complicated statistical procedures [26]. Another example of simulation in statistical computations occurred at the beginning of the twentieth century, when William Gosset ("Student") published his famous papers, where he investigated the

[3] Note that the MAP estimator can also be expressed in an integral form [2, 3], but the maximization approach is much more useful from a practical point of view.

[4] The St. Petersburg game consists of tossing a fair coin repeteadly until a head occurs [15]. The payoff then is 2^k, where k is the number of tosses required. Buffon's goal was computing the expected payoff of the game in practice (theoretically it is infinite), which turned out to be 4.9106 in his experiment.

[5] Buffon's needle experiment consists of dropping a needle of length ℓ on a grid of parallel lines uniformly separated by distance $d > \ell$ and counting the number of times that the needles intersect the lines (n) out of the N experiments. This empirical intersection probability, $\hat{p} = \frac{n}{N}$, can be used to obtain an approximate value of π, since $p = \frac{2\ell}{\pi d}$, and thus $\pi \approx \frac{2\ell}{\hat{p}d}$.

[6] Actually, Lazzarini's experimental approximation of $\pi \approx 3.1415929$ (accurate to six decimal places), provided in [21], has been disputed and several authors have suggested that he did not perform a fair experiment [23–25].

distribution of the t-statistic and the correlation coefficient [27, 28][7]. Finally, Leonard H. C. Tippett devised a way to systematically draw random numbers for his experiments on extreme value distributions and published a list of random digits that was used by subsequent researchers [31, 32]. However, all these approaches occurred before the advent of computers and aimed only at solving some particular problem at hand, not at providing some general simulation method (except for Galton's approach [33], which provided a generic way to draw normal random variables (RVs) for all types of applications, but failed to gain widespread acceptance).

In spite of all these early attempts to perform stochastic simulation (a.k.a. statistical sampling), the formulation of the MC method as we know it today did not happen until the construction of the first computers in the 1940s[8]. Stanislaw Ulam, a Polish mathematician working at Los Alamos National Laboratory, devised the MC method while convalescing from an illness in 1946 [36, 38]. He was playing solitaire and trying to calculate the probability of success (a difficult combinatorial problem) when he realized that an easier way to accomplish that task (at least in an approximate way) would be to play a certain number of hands and compute the empirical success probability. On his return to Los Alamos, he learnt of the new computers that were being built from his close friend John von Neumann, a consultor both at Los Alamos and the Ballistics Research Laboratory (where the first computer, the ENIAC, was being developed), and discussed the possibility of developing a computer-based implementation of his idea to solve difficult problems in statistical physics. Von Neumann immediately recognized the relevance of Ulam's idea and sketched an approach to solve neutron diffusion/multiplication problems through computer-based statistical sampling in a 1947 letter to Robert Richtmyer (head of the Theoretical Division at Los Alamos) [38]. The method was then successfully tested on 9 neutron transport problems using ENIAC and Nicholas Metropolis coined the name "Monte Carlo", inspired by an uncle of Stan Ulam who borrowed money from relatives because "he just had to go to Monte Carlo" [36, 37]. The seminal paper on MC was then published in 1949 [39], more powerful computers were developed (like the MANIAC in Los Alamos [35]), and many physicists started using computer-based MC methods to obtain approximate solutions to their problems [40]. MC methods required an extensive supply of random numbers, and the development of the essential random number generators required by MC methods also started during those years. For instance, von Neumann described the rejection sampling (RS) method in a 1947 letter to Ulam [38] (although it was not published until 1951 [41]) and Lehmer introduced linear congruential random number generators in 1951 [42].

The next milestone in statistical sampling was the development of the Metropolis-Hastings (MH) algorithm. The MH algorithm was initially devised by Nicholas Metropolis et al. in 1953 as a general method to speed up the computation of the properties of substances composed of interacting individual molecules [43]. The idea is rather simple: random uniformly distributed moves of particles around their current position were proposed; if the global energy of the system was decreased, these moves were always accepted; otherwise, they were accepted only with some non-null probability that depended on the energy increase (the larger the increase, the less likely the move to be accepted). Rejected moves were also used to compute the desired averages. Metropolis et al. proved that the method was ergodic and samples were drawn from the desired distribution. This approach can be seen as a Markov chain, with an RS sampling step at the core to ensure that the chain has the desired invariant probability density function (PDF), and thus Markov chain Monte Carlo (MCMC) methods were born. A symmetric proposal density was considered in [43]. In 1970, Hastings showed that non-symmetric proposal densities could also be used [44], thus allowing for much more flexibility in the method, and proposed a generic acceptance probability that guaranteed the ergodicity of the chain. In the meantime, a different acceptance probability rule had been proposed by Barker in 1965 [45], and it remained to be seen which rule was better. This issue was settled in 1973 by Peskun (a Ph.D. student of Hastings), who proved that the Hastings acceptance rule was optimal [46]. The MH algorithm was extensively used by the physics community since the beginning, but few statisticians or engineers were aware of it until the 1990s [47].

Another crucial event in the history of MC methods was the introduction, by Stuart Geman and Donald Geman in 1984, of a novel MCMC algorithm, the Gibbs sampler, for the Bayesian restoration of images [48]. The Gibbs sampler became very popular soon afterwards, thanks to the classical 1990 paper of Gelfand and Smith [49], who gave examples of how the Gibbs sampler could be applied in Bayesian inference. Andrew Gelman showed in 1992 that the Gibbs sampler was a particular case of the MH algorithm [50], thus causing a renewed interest in the MH algorithm by statisticians. Then, Tierney wrote an influential paper on the history and theory of the MH algorithm in 1994 [51], where he showed how it could be used to deal with non-standard distributions in Bayesian

[7] William S. Gosset published his two famous papers under the pseudonym "Student", after attaining permission from his employer *Arthur Guinness & Sons* of Dublin, to avoid conflicts with other employees who were forbidden from publishing papers in scientific journals [29, 30].

[8] Apparently, Enrico Fermi was the first one to make a systematic use of statistical sampling techniques to compute approximations to all kind of physical quantities of interest while working in Rome (i.e., before 1938). However, he never wrote anything about it and we only have an indirect account of this fact from his student Emilio Segrè [34] (see also [35–37]).

inference. Simple explanations of the Gibbs sampler and the MH algorithm also appeared in the 1990s [52, 53], and those two methods started being applied for all sort of problems during the following years: medicine [54], econometrics [55], biostatistics [56], phylogenetic inference [57], etc. Indeed, the MH algorithm has become so popular since its re-discovery in the early 1990s that it was named one of the top 10 algorithms in the 20th century by the *IEEE Computing in Science & Engineering Magazine* [58].

The first signal processing applications of MCMC followed soon after Geman and Geman's publication of the Gibbs sampler (indeed, their original application involved a signal processing problem: the denoising of images). In the 1990s, both the MH algorithm and the Gibbs sampler were applied to several signal processing problems: blind identification, deconvolution, and equalization [59–63]; denoising and restoration of missing samples in digital audio recordings [5, 64–66]; reconstruction of the images obtained in computed tomography [67, 68]; parameter estimation of time-varying autoregressive (AR) models [69, 70]; etc. Then, Fitzgerald published the first tutorial on MCMC methods for signal processing applications in 2001 [71], and the first special issue on MC methods for statistical signal processing (edited by Petar Djuric and Simon Godsill) appeared in 2002 [72]. During these years, tutorial papers on the related areas of signal processing for wireless communications and machine learning also appeared [73, 74], as well as another review paper on MC methods for statistical signal processing [75].

The second large family of Monte Carlo methods are the so-called importance sampling (IS) and its adaptive versions (AIS). Unlike MCMC techniques, where candidate samples can be either accepted or discarded, IS methods employ all the generated candidates, assigning them a weight according to their "quality". IS was first used in statistical physics in order to estimate the probability of nuclear particles to penetrate shields [76]. During the following decades, IS was extensively used as a variance reduction technique (especially for rare event simulation) in a large variety of applications: operations research [77], simulation of stochastic processes [78], other problems in statistical physics [79, 80], digital communications [81, 82], computer reliability [83], inventory systems [84], etc. In the 1970s and 1980s, several authors also applied the IS principle in Bayesian inference problems when direct sampling from the posterior distribution was either impossible or impractical [85–87]. The limitations of the IS approach were also recognized at this time: the performance of IS-based estimators critically depends on the choice of the proposal, with good proposals leading to a substantial decrease in variance and bad proposals resulting in a very poor performance (with a potentially infinite variance from a theoretical point of view). In order to solve these issues, the multiple IS (MIS) approach and alternative weighting schemes (like the so called deterministic mixture (DM)) were proposed in the 1990s [88–91]. During these years, sequential importance sampling (SIS) methods (a.k.a. particle filters) were also developed as an alternative to the Kalman filter for the estimation of dynamic parameters [92, 93]. These methods are also based on the IS methodology, with weights that are sequentially updated as new observations become available. See the companion tutorial in this special issue for a detailed review of sequential Monte Carlo (SMC) methods, which essentially correspond to SIS with resampling [93].

However, IS techniques did not become widely known to all computational statistics, machine learning and statistical signal processing practitioners until the 2000s. In 2001, Iba published a cross-disciplinary survey in which he grouped several algorithms where "a set of 'walkers' or 'particles' is used as a representation of a high-dimensional vector" under the generic name of *population Monte Carlo algorithms* [94]. Soon afterwards, Cappé et al. published their influential population Monte Carlo (PMC) paper [95], where they borrowed the name coined by Iba for their proposed AIS framework. In short, [95] showed that previously drawn samples can be used to adapt the proposal in order to reduce the variance of the desired estimators. The original PMC algorithm considered a set of Gaussian proposals with different variances and means selected from the previous population through a multinomial resampling step, where particles were selected with a probability proportional to their IS weights. This classical or standard PMC algorithm is numerically unstable and shows a poor performance in many practical applications, but opened the door to other improved PMC algorithms, like the mixture PMC (M-PMC) [96] or the recent deterministic mixture PMC (DM-PMC) [97]. Furthermore, the success of PMC-based approaches renewed the interest in IS techniques for the estimation of static parameters, encouraging authors to develop other AIS methods, like the adaptive multiple importance sampling (AMIS) [98] or the adaptive population importance sampling (APIS) [99] algorithms.

Finally, let us remark that many important advances have occurred in the field of Monte Carlo methods during the last 20 years: adaptive MCMC techniques that increase the acceptance rate and decrease the correlation among samples, gradient-based MCMC methods which improve the performance in high-dimensional parameter spaces, multiple candidate MCMC algorithms for a higher efficiency in sample generation, generalized sampling and weighting schemes in MIS algorithms for a reduced variance of the desired estimators, the combination of MCMC and AIS techniques in order to exploit their complementary strong points and minimize their

drawbacks, etc. A detailed description of all these methods is provided in this survey. Other important topics that are also briefly covered here are the following: the pseudo-marginal MCMC framework [100], approximate Bayesian computation (ABC) [101, 102], the application of Monte Carlo algorithms in a big data context [103], noisy MCMC methods, and approximated likelihood algorithms.

1.3 Related articles, books, and software packages

The literature on MC methods is rather vast, with many technical reports, journal papers, conference articles, books, and book chapters that cover different aspects of the many existing MC algorithms. In this section, we provide a brief summary (which intends to be illustrative rather than exhaustive) of the articles and books that provide a tutorial introduction or an overview of several aspects of MC methods and closely related topics. At the end of the section we also describe some of the most relevant software packages which are freely available to implement several important MC algorithms. Note that these articles, books, and/or software packages often concentrate on some particular class of MC algorithms, and the user has to select the most appropriate family of methods and software for the specific problem. In particular, note that different MCMC methods have different convergence properties, and therefore we encourage users to be careful and select the most reliable algorithm for their problem.

On the one hand, many excellent books are entirely devoted to the general theory and practice of MC methods [104–109]. However, none of these books is specifically written with signal processing practitioners in mind and they are 5–14 years old, thus not covering several important recently developed algorithms. On the other hand, several books are also devoted to specific classes of MC methods. For instance, [110] and [111] focus on particle filters for tracking applications and random set models respectively, [112] details several different state-space processors (including those based on particle filters), [113] is entirely devoted to the theoretical and practical aspects of SMC methods, and [114] covers Bayesian filtering and smoothing techniques from Kalman to particle filters. Finally, several books address the related topic of random variable generation [115–119], which is an essential issue for MC algorithms, and some of these books also contain one or more chapters on MC methods (e.g., Chapter 7 of [118, 119]).

There are also many other journal papers and conference articles that provide tutorial descriptions of MC methods, but they are either more than 10 years old, differ in scope from the present paper, or cover only some specific class of MC algorithms. The first tutorial on MC methods for signal processing practitioners (as far as we know), covering classical MC techniques (e.g., the MH algorithm, the Gibbs sampler, and reversible jump MCMC) for parameter estimation and model selection, appeared in 2001 [71]. Similar tutorials for wireless communications [73], including also SIS and SMC schemes, and machine learning [74], where simulated annealing and the MC-EM algorithm are described, shortly followed. Then, another tutorial on MC methods for signal processing was published in 2004 and focused on recent advances in MCMC algorithms and particle filters [120]. More recently, Green et al. published a tutorial on Bayesian computation that partially overlaps with the current survey (e.g., it includes MALA, the HMC algorithm, and particle MCMC) [121]. A survey specifically focused on different Multiple Try MCMC methods can be found in [122], whereas Robert et al. [123] have recently published in arXiv another overview on algorithms to accelerate MCMC that briefly discusses several methods included in this paper (like MTM, HMC, or adaptive MCMC). Several surveys that concentrate exclusively on importance sampling methods have also been published recently [124–126].

Finally, note that many toolboxes and specific software implementations (in Matlab, Python, R, and other programming languages) of the different algorithms described in this survey are freely available online. Due to their importance, let us mention three of the main existing environments for MC computation: BUGS, JAGS, and Stan[9]. On the one hand, BUGS (Bayesian inference Using Gibbs Sampling) is a software package that allows the user to specify a statistical model by simply stating the relationships between related variables [127–129]. The software includes an "expert system" that determines the appropriate MCMC scheme (based on the Gibbs sampler) for analysing the specified model. On the other hand, JAGS (Just Another Gibbs Sampler) is a program for the analysis of Bayesian hierarchical models using MCMC simulation [130]. It provides a cross-platform engine for the BUGS language, allowing users to write their own functions, distributions, and samplers. Finally, Stan is a flexible probabilistic programming language that allows users to specify their statistical models and then perform Bayesian inference using MCMC methods (NUTS and HMC), ABC or ML estimation [131, 132]. Stan has Python and R interfaces, as well as wrapper packages for Matlab, Julia, Stata, and Mathematica.

1.4 Acronyms, notation, and organization

Table 1 provides a list of the acronyms used throughout the paper, whereas Table 2 summarizes the main notation.

[9] Further information about them can be found in their respective web sites: JAGS (`http://mcmc-jags.sourceforge.net`), BUGS (`http://www.openbugs.net/w/FrontPage`), and Stan (`http://mc-stan.org`).

Table 1 List of acronyms used

ABC	Approximate Bayesian computation	MC	Monte Carlo
ADS	Adaptive direction sampling	MCMC	Markov chain Monte Carlo
AGM-MH	Adaptive Gaussian mixture Metropolis-Hastings	MH	Metropolis-Hastings
AIS	Adaptive importance sampling	MIS	Multiple importance sampling
AISM	Adaptive independent sticky metropolis	ML	Maximum likelihood
AM	Adaptive Metropolis	MMALA	Riemann manifold MALA
AMCMC	Adaptive Markov chain Monte Carlo	MMSE	Minimum mean squared error
AMIS	Adaptive multiple importance sampling	M-PMC	Mixture population Monte Carlo
APIS	Adaptive population importance sampling	MRF	Markov random field
ARS	Adaptive rejection sampling	MSE	Mean squared error
ARMS	Adaptive rejection Metropolis sampling	MTM	Multiple-try Metropolis
CDF	Cumulative distribution function	NUTS	No U-turn sampler
CLT	Central Limit Theorem	OFDM	Orthogonal frequency division multiplexing
DA	Data augmentation	PDF	Probability density function
DM	Deterministic mixture	PMC	Population Monte Carlo
DR	Delayed rejection	PMH	Particle Metropolis-Hastings
FUSS	Fast universal self-tuned sampler	PWC	Piecewise constant
GMS	Group Metropolis sampling	PWL	Piecewise linear
HMC	Hamiltonian Monte Carlo	RMHMC	Riemann manifold HMC
IA^{2}RMS	Independent doubly adaptive rejection Metropolis sampling	RS	Rejection sampling
IID	Independent and identically distributed	RV	Random variable
IS	Importance sampling	SDE	Stochastic differential equation
LAIS	Layered adaptive importance sampling	SIS	Sequential importance sampling
MALA	Metropolis adjusted Langevin algorithm	SMC	Sequential Monte Carlo
MAP	Maximum a posteriori	WSN	Wireless sensor network

Moreover, the following rules will be followed regarding the notation:

- Vectors and matrices will be denoted in boldface (e.g., $\mathbf{y}$ and $\mathbf{C}$), with $\text{vec}\{\mathbf{y}_1, \ldots, \mathbf{y}_L\}$ denoting the vectorization operation, i.e., the stacking of a set of vectors $(\mathbf{y}_1, \ldots, \mathbf{y}_L)$ of dimension $D_{\mathbf{y}} \times 1$ in order to construct a single vector $\mathbf{y} \in \mathbb{R}^{LD_{\mathbf{y}}}$. Capital boldface symbols are used for matrices, whereas lowercase boldface symbols are used for vectors.
- The notation $\boldsymbol{\theta}_{\neg i}$ will be used to denote a vector with the ith component removed, i.e., $\boldsymbol{\theta}_{\neg i} = [\theta_1, \ldots, \theta_{i-1}, \theta_{i+1}, \ldots, \theta_{D_\theta}]^\top$.
- Capital letters will be used to denote random variables (e.g., X), while lowercase letters are used for their realizations (e.g., x).
- When required, properly normalized PDFs will be indicated by using a bar (e.g., $\bar{\pi}$ and $\bar{q}$), whereas their non-negative unnormalized versions will be indicated by the same letter without the bar (e.g., π and q).
- The notation $x \sim p(X)$ indicates that a realization x of the random variable X is drawn from the PDF p.
- We use an argument-wise notation for the different normalized and unnormalized densities used throughout the text. For instance, $\pi(\boldsymbol{\theta})$ denotes the D_θ-dimensional target, whereas $\pi(\theta_d|\boldsymbol{\theta}_{\neg d})$ denotes the one-dimensional full conditional density of the dth parameter.
- The notation $\mathbb{E}_p(g)$ will be used to denote the mathematical expectation of the function g w.r.t. the PDF p.

Regarding the structure of the paper, let us remark that we concentrate on the use of MCMC methods for the estimation of static parameters, although the extension of some of these techniques to a dynamical setting will be occasionally discussed. This choice is motivated by two facts: the need to keep the length of the tutorial within reasonable bounds and the existence of two recent review papers on AIS methods [126, 133]. However, two sections detailing the different IS and AIS techniques, as well as the use of IS-within-MCMC, have also been included for the sake of completeness. Regarding the selection of the methods covered, we have tried to include the most

Table 2 Summary of the main notation used throughout the paper

Notation	Description
D_y	Dimension of the data.
L	Number of data available.
$\mathbf{y} \in \mathbb{R}^{LD_y}$	LD_y-dimensional observations vector, $\mathbf{y} = \text{vec}\{\mathbf{y}_1, \ldots, \mathbf{y}_L\}$ with $\mathbf{y}_i \in \mathbb{R}^{D_y}$ for $i = 1, \ldots, L$.
D_θ	Dimension of the parameter space.
$\Theta = \Theta_1 \times \cdots \times \Theta_{D_\theta}$	Feature space for the parameter vector $\boldsymbol{\theta}$.
$\boldsymbol{\theta} \in \mathbb{R}^{D_\theta}$	D_θ-dimensional parameter vector, $\boldsymbol{\theta} = [\theta_1, \ldots, \theta_{D_\theta}]$ with $\theta_d \in \Theta_d$ for $d = 1, \ldots, D_\theta$.
$\boldsymbol{\theta}^{(m)}$	mth sample of the parameter vector in MC and RS.
$\boldsymbol{\theta}^{(t)}$	Sample of the parameter vector at the tth iteration in MCMC methods.
$\bar{\pi}(\boldsymbol{\theta}\|\mathbf{y}) \equiv \bar{\pi}(\boldsymbol{\theta})$	Target (i.e., posterior) PDF.
$\pi(\boldsymbol{\theta}\|\mathbf{y}) \equiv \pi(\boldsymbol{\theta})$	Target function (i.e., non-negative but unnormalized).
$p_0(\boldsymbol{\theta})$	Prior probability density function.
$\ell(\mathbf{y}\|\boldsymbol{\theta})$	Likelihood.
$Z(\mathbf{y})$	Normalizing constant of the target (a.k.a. partition function, marginal likelihood, or model evidence).
$\bar{\pi}(\theta_d\|\boldsymbol{\theta}_{\neg d})$	Full conditional PDF for the dth parameter given all the other parameters (used in the Gibbs sampler).
T	Number of Monte Carlo iterations performed.
T_b	Number of iterations for the burn-in period in MCMC.
N	Number of proposals used in multiple IS approaches.
M	Number of samples drawn in the MC algorithm, RS and IS approaches. Usually $M \geq N$ in MIS.
$\bar{q}(\boldsymbol{\theta})$, $\bar{q}_t(\boldsymbol{\theta})$, $\bar{q}_{m,t}(\boldsymbol{\theta})$	Proposal PDF.
$q(\boldsymbol{\theta})$, $q_t(\boldsymbol{\theta})$, $q_{m,t}(\boldsymbol{\theta})$	Proposal function (i.e., non-negative but unnormalized) for $t = 1, \ldots, T$ and $m = 1, \ldots, M$.
$w_{m,t}(\boldsymbol{\theta})$	Unnormalized weight of the mth particle ($m = 1, \ldots, M$) at the tth iteration ($t = 1, \ldots, T$) for AIS approaches.
$w_{m,t}(\boldsymbol{\theta})$	Normalized weight of the mth particle ($m = 1, \ldots, M$) at the tth iteration ($t = 1, \ldots, T$) for AIS approaches.
$\hat{\pi}(\boldsymbol{\theta})$	Random measure used to approximate the target at the tth iteration.
$\boldsymbol{\mathcal{N}}(\mu, \mathbf{C})$, $\boldsymbol{\mathcal{N}}(\cdot\|\mu, \mathbf{C})$	Gaussian PDF with mean μ and covariance $\mathbf{C}$.
$\boldsymbol{\mathcal{U}}(\mathcal{I})$	Uniform PDF within the interval $\mathcal{I}$.

relevant MC algorithms from the different families, following a chronological order from the classical (and usually simpler) MC methods to the more recent and sophisticated ones. Finally, note that the main focus of the paper is describing the different MC algorithms in a unified way by using a consistent notation which is amenable to signal processing practitioners. However, some theoretical aspects are also briefly discussed, as well as the main advantages and limitations of each algorithm.

The rest of the paper is organized as follows. First of all, the mathematical background is provided in Section 2. The Bayesian framework for statistical inference and the basic MC algorithm are briefly reviewed here (Section 2.1), altogether with RS, which lies at the heart of MCMC methods (Section 2.2). Then, Section 3 describes in detail many of the most relevant MCMC algorithms for signal processing applications: the MH algorithm, the Gibbs sampler, and their combined use (Section 3.1); adaptive MCMC methods (Section 3.2); gradient-based algorithms (Section 3.3); and other advanced MCMC schemes (Section 3.4). A short dicussion on MCMC convergence diagnostics (Section 3.5) is also included here. This is followed by Section 4, where IS techniques are described: standard IS vs. multiple IS (Section 4.1); adaptive IS (Section 4.2); group IS (Section 4.7); and sequential IS (Section 4.8). Some convergence results on IS and AIS (Section 4.3) are also included here, as well as a short discussion on the variance of the IS estimator and the choice of the optimal proposal (Section 4.4), a note on the estimation of the effective sample size (Section 4.5), and a description of proper weighting schemes (Section 4.6). This is followed by the description of different schemes for the use of IS-within-MCMC in Section 5: multiple try approaches for static (Section 5.1) and dynamic parameters (Section 5.2); pseudo-marginal MCMC methods (Section 5.3); noisy MCMC algorithms (Section 5.4); and approximated likelihood methods (Section 5.4.2). Finally, the performance of many of the described methods is demonstrated through several numerical simulations in Section 6: two simple examples for MCMC and IS methods (Sections 6.1 and 6.2); the estimation of the

parameters of a chaotic system (Section 6.3); a localization problem in wireless sensor networks (Section 6.4); and a spectral estimation application (Section 6.5). A discussion of the reviewed methods concludes the paper in Section 7.

2 Mathematical formulation

2.1 Bayesian inference and the Monte Carlo method

Let us assume that we have a dataset, $\mathbf{y} = \text{vec}\{\mathbf{y}_1, \ldots, \mathbf{y}_L\} \in \mathbb{R}^{LD_{\mathbf{y}}}$ with $\mathbf{y}_i \in \mathbb{R}^{D_{\mathbf{y}}}$, which depends on some *static* parameter vector, $\boldsymbol{\theta} = [\theta_1, \ldots, \theta_{D_\theta}]^\top \in \Theta \subseteq \mathbb{R}^{D_\theta}$. From a Bayesian point of view, all the information required to solve any task related to $\boldsymbol{\theta}$ (e.g., inference or optimization problems) is contained in the *posterior* or *target* PDF, $\bar{\pi}(\boldsymbol{\theta}|\mathbf{y})$. Using Bayes rule, this posterior can be expressed as

$$\bar{\pi}(\boldsymbol{\theta}|\mathbf{y}) = \frac{\ell(\mathbf{y}|\boldsymbol{\theta})p_0(\boldsymbol{\theta})}{Z(\mathbf{y})} = \frac{\pi(\boldsymbol{\theta}|\mathbf{y})}{Z(\mathbf{y})}, \tag{1}$$

where $\ell(\mathbf{y}|\boldsymbol{\theta})$ is the *likelihood*, that depends on the statistical input-output model assumed; $p_0(\boldsymbol{\theta})$ is the *prior* PDF, which summarizes all the information available about $\boldsymbol{\theta}$ external to the observation of the data; $Z(\mathbf{y})$ is the *marginal likelihood* (a.k.a. as model evidence or partition function in some contexts), a normalizing term which does not depend on $\boldsymbol{\theta}$; and $\pi(\boldsymbol{\theta}|\mathbf{y})$ is the *target function*, a non-negative definite function (i.e., $\pi(\boldsymbol{\theta}|\mathbf{y}) \geq 0$ for all $\boldsymbol{\theta} \in \Theta \subseteq \mathbb{R}^{D_\theta}$ and $\mathbf{y} \in \mathbb{R}^{LD_{\mathbf{y}}}$) such that $\int_\Theta \pi(\boldsymbol{\theta}|\mathbf{y})\, d\boldsymbol{\theta} = Z(\mathbf{y})$ with $Z(\mathbf{y}) \neq 1$ in general.

Now, let us assume that we want to compute the following integral,

$$\begin{aligned} I = \mathbb{E}_{\bar{\pi}}(g(\boldsymbol{\theta})) &= \int_\Theta g(\boldsymbol{\theta})\bar{\pi}(\boldsymbol{\theta}|\mathbf{y})d\boldsymbol{\theta} \\ &= \frac{1}{Z(\mathbf{y})}\int_\Theta g(\boldsymbol{\theta})\pi(\boldsymbol{\theta}|\mathbf{y})d\boldsymbol{\theta} < \infty, \end{aligned} \tag{2}$$

where $g(\boldsymbol{\theta})$ can be any integrable function w.r.t. $\bar{\pi}(\boldsymbol{\theta}|\mathbf{y})$. For instance, when $g(\boldsymbol{\theta}) = \boldsymbol{\theta}$ this integral becomes the well-known minimum mean squared error (MMSE) estimator of the parameter $\boldsymbol{\theta}$ [2–4],

$$\hat{\boldsymbol{\theta}}_{\text{MMSE}} = \mathbb{E}_{\bar{\pi}}(\boldsymbol{\theta}) = \int_\Theta \boldsymbol{\theta}\bar{\pi}(\boldsymbol{\theta}|\mathbf{y})d\boldsymbol{\theta}, \tag{3}$$

which is widely used in many statistical signal processing applications and corresponds to the conditional expectation of $\boldsymbol{\theta}$ w.r.t. the posterior PDF.

Unfortunately, obtaining an analytical solution of these integrals is usually unfeasible in many practical problems of interest. In these cases, an approximate solution of (2) can be obtained through the Monte Carlo (MC) method shown in Algorithm 1. Essentially, the MC method simply consists of obtaining a set of independent and identically distributed (IID) samples of the parameter vector to be inferred and using them to approximate the desired integral by means of an unweighted sum. These M samples, $\boldsymbol{\theta}^{(m)}$, can be obtained either by sampling directly from the target PDF (i.e., the posterior $\bar{\pi}(\boldsymbol{\theta}|\mathbf{y})$), as shown in Algorithm 1, or by replicating the physical procedure where the desired parameters are involved. Note that the subindex M in $\widehat{I}_M$ denotes the number of samples involved in the estimation.

Algorithm 1 Monte Carlo (MC) approximation of the integral in Eq. (2)

1. Draw $\boldsymbol{\theta}^{(m)} \sim \bar{\pi}(\boldsymbol{\theta}|\mathbf{y})$ for $m = 1, \ldots, M$.
2. Approximate the integral in Eq. (2) as

$$\widehat{I}_M = \frac{1}{M}\sum_{m=1}^{M} g(\boldsymbol{\theta}^{(m)}). \tag{4}$$

The MC estimate of I provided by Eq. (4) is unbiased, i.e., $\mathbb{E}_{\bar{\pi}}(\widehat{I}_M) = I$. Moreover, by the *strong law of large numbers*, $\widehat{I}_M \to I$ almost surely (a.s.) as $M \to \infty$ [104]. Furthermore, if $g(\boldsymbol{\theta})$ is square integrable w.r.t. $\bar{\pi}(\boldsymbol{\theta}|\mathbf{y})$, then we can use the *central limit theorem* (CLT) to state the following result [104]:

$$\frac{\widehat{I}_M - I}{\sqrt{V_M}} \xrightarrow{d} \boldsymbol{\mathcal{N}}(0,1) \qquad \text{as} \quad M \to \infty, \tag{5}$$

where $\xrightarrow{d}$ denotes convergence in distribution, and

$$V_M = \frac{1}{M}\mathbb{E}_{\bar{\pi}}((g(\boldsymbol{\theta}) - I)^2) = \frac{1}{M}\int_\Theta (g(\boldsymbol{\theta}) - I)^2 \bar{\pi}(\boldsymbol{\theta}|\mathbf{y})\, d\boldsymbol{\theta}. \tag{6}$$

Note that (5) is equivalent to stating that $\widehat{I}_M \xrightarrow{d} \boldsymbol{\mathcal{N}}(I, V_M)$ as $M \to \infty$.

Unfortunately, Algorithm 1 cannot be applied in many practical problems, because we cannot draw samples directly from $\bar{\pi}(\boldsymbol{\theta}|\mathbf{y})$. In these cases, if we can perform point-wise evaluations of the target function, $\pi(\boldsymbol{\theta}|\mathbf{y}) = \ell(\mathbf{y}|\boldsymbol{\theta})p_0(\boldsymbol{\theta})$, we can apply other types of Monte Carlo algorithms: rejection sampling (RS) schemes, Markov chain Monte Carlo (MCMC) techniques, and importance sampling (IS) methods. These two large classes of algorithms, MCMC and IS, are the core of this paper and will be described in detail in the rest of this work. Before, we briefly recall the basis of the RS approach, which is one of the key ingredients of MCMC methods, in the following section.

2.2 Rejection sampling (RS)

The RS method is a classical Monte Carlo technique for *universal sampling* that can be used to generate samples virtually from any target density $\bar{\pi}(\boldsymbol{\theta})$ by drawing from

a simpler proposal density $\bar{q}(\boldsymbol{\theta})$[10]. The sample is either accepted or rejected by applying an adequate test to the ratio of the two PDFs, and it can be easily proved that accepted samples are actually distributed according to the target density [115]. The RS algorithm was originally proposed by John von Neumann in a 1947 letter to Stan Ulam [38], but it was not published until 1951 [41]. In its original formulation, von Neumann considered only a uniform proposal PDF, but the algorithm was later generalized to allow drawing samples from any proposal density from which sampling is straightforward. In the standard RS algorithm [41, 115], we first draw a sample from the proposal PDF, $\boldsymbol{\theta'} \sim q(\boldsymbol{\theta})$ and then accept it with probability

$$p_A(\boldsymbol{\theta'}) = \frac{\pi(\boldsymbol{\theta'})}{Cq(\boldsymbol{\theta'})} \leq 1, \tag{7}$$

where C is a constant such that $Cq(\boldsymbol{\theta})$ is an envelope function for $\pi(\boldsymbol{\theta})$, i.e., $Cq(\boldsymbol{\theta}) \geq \pi(\boldsymbol{\theta})$ for all $\boldsymbol{\theta} \in \Theta$. We can summarize this procedure in an equivalent way: at the tth iteration, draw a sample $\boldsymbol{\theta}^{(t)} \sim \bar{q}(\boldsymbol{\theta})$ and $u \sim \boldsymbol{\mathcal{U}}([0,1])$; if $u \leq \frac{\pi(\boldsymbol{\theta}^{(t)})}{Cq(\boldsymbol{\theta}^{(t)})}$, accept $\boldsymbol{\theta}^{(t)}$, otherwise, reject it; when the desired number of samples have been drawn from $\bar{\pi}(\boldsymbol{\theta})$, stop. Algorithm 2 summarizes the generation of M samples from the target PDF using the standard RS algorithm.

Algorithm 2 Rejection Sampling (RS) method.

1. **Initialization:** Choose a proposal function, $q(\boldsymbol{\theta})$, and the required number of samples from the target, M. Find an upper bound, $C \geq \frac{\pi(\boldsymbol{\theta})}{q(\boldsymbol{\theta})}$ for all $\boldsymbol{\theta} \in \Theta$, and let $t = m = 1$.
2. `WHILE` $m \leq M$:
   (a) Draw $\boldsymbol{\theta}^{(t)} \sim \bar{q}(\boldsymbol{\theta})$ and $u \sim \boldsymbol{\mathcal{U}}([0,1))$.
   (b) If $u \leq \frac{\pi(\boldsymbol{\theta}^{(t)})}{Cq(\boldsymbol{\theta}^{(t)})}$, accept $\boldsymbol{\theta}^{(t)}$, setting $\boldsymbol{\theta}^{(m)} = \boldsymbol{\theta}^{(t)}$ and letting $m = m + 1$.
   (c) Set $t = t + 1$ regardless of whether $\boldsymbol{\theta}^{(t)}$ has been accepted or not.
3. Approximate the integral in Eq. (2) using Eq. (4).

The RS algorithm is a simple MC method for approximating the integral in Eq. (2) that can be universally applied as long as the upper bound C can be found. However, it has several important drawbacks that hinder its practical application:

1. For complicated targets, finding a bound C such that $Cq(\boldsymbol{\theta}) \geq \pi(\boldsymbol{\theta})$ for all $\boldsymbol{\theta} \in \Theta$ can be difficult, especially for high-dimensional parameter spaces.
2. Even if this bound can be found, the RS algorithm can be very inefficient if the ratio $\frac{\pi(\boldsymbol{\theta})}{Cq(\boldsymbol{\theta})}$ is small for a large portion of the parameter space. Indeed, the acceptance probability of the RS algorithm is given by

$$P_A = \int_\Theta \frac{\pi(\boldsymbol{\theta})}{Cq(\boldsymbol{\theta})} \bar{q}(\boldsymbol{\theta})\, d\boldsymbol{\theta} = \frac{Z_\pi}{CZ_q}, \tag{8}$$

   where $Z_\pi = \int_\Theta \pi(\boldsymbol{\theta})\, d\boldsymbol{\theta}$ and $Z_q = \int_\Theta q(\boldsymbol{\theta})\, d\boldsymbol{\theta}$. Depending on the target and the proposal selected, this P_A can be very low (this happens when $CZ_q \gg Z_\pi$), thus rendering the RS algorithm useless in practice. For this reason, many RS approaches have been specifically designed for drawing efficiently from a specific target distribution [134, 135]. For example, efficient random number generators based on RS schemes can be found for the Gamma, Beta, and Nakagami distributions [136–140].
3. The number of iterations required to generate M samples, T, is a random variable with an expected value $\mathbb{E}(T) = \frac{M}{P_A}$ and P_A given by (8). Hence, the exact time required to generate M valid samples cannot be set a priori, and this can be a serious problem in many applications.

One way to tackle some of these difficulties is by constructing the proposal $q(\boldsymbol{\theta})$ adaptively, using some of the so called adaptive RS (ARS) methods. The ARS algorithm was originally proposed by Gilks and Wild in 1992 [141], and several generalized ARS algorithms have been proposed since then [142–150]. However, the need to have $Cq(\boldsymbol{\theta}) \geq \pi(\boldsymbol{\theta})$ for all $\boldsymbol{\theta} \in \Theta$ and the difficulty of constructing the adaptive proposals in high-dimensional parameter spaces limit the applicability of those generalized ARS algorithms [119, 151], rendering MCMC and IS approaches more efficient in general, and thus preferable for practical applications. For further information see Chapters 3 and 4 in [119].

3 Markov chain Monte Carlo (MCMC)

According to Definition 7.1 of [104], an MCMC method is any method producing an ergodic Markov chain whose stationary density is the desired target PDF, $\bar{\pi}(\boldsymbol{\theta})$. In the following, we detail some of the most relevant MCMC algorithms, starting from the basic building blocks (the MH algorithm and the Gibbs sampler) in Section 3.1, and ending up with several advanced adaptive (Section 3.2), gradient-based (Section 3.3), and other advanced MCMC schemes (Section 3.4). Note that we focus on describing the different algorithms rather than on their theoretical properties, although a brief discussion on the validity of the MH algorithm (due to its importance as the basis of most MCMC algorithms) is provided in Section 3.1.2.

[10] For the sake of simplicity, in the sequel, we drop the dependence on the data ($\mathbf{y}$) from the target, i.e., we use $\bar{\pi}(\boldsymbol{\theta}) \equiv \bar{\pi}(\boldsymbol{\theta}|\mathbf{y})$ and $\pi(\boldsymbol{\theta}) \equiv \pi(\boldsymbol{\theta}|\mathbf{y})$.

3.1 MCMC basic building blocks: Metropolis-Hastings and Gibbs samplers

3.1.1 Metropolis-Hastings (MH) algorithm

As mentioned in the introduction, the MH algorithm was initially devised by Nicholas Metropolis et al. in 1953 as a general method to speed up the computation of the properties of substances composed of interacting individual molecules [43]. In a two-dimensional square with N particles, studying these properties requires computing $2N$-dimensional integrals, an impossible task (both analytically or numerically) for the large values of N required in practice. A naive MC simulation would consist in drawing particle configurations uniformly at random and assigning them weights proportional to $\exp(-E/K\mathcal{T})$, where E is the energy of the configuration, $K \approx 1.38 \times 10^{-23}$ is Boltzmann's constant and $\mathcal{T}$ is the absolute temperature (in Kelvin degrees)[11]. Nevertheless, Metropolis et al. devised the following alternative modified MC scheme:[12]

1. Place the N particles in an initial arbitrary configuration (e.g., in a regular lattice) within the square.
2. At the tth iteration ($t = 1, \ldots, T$) and for $i = 1, \ldots, N$:[13]
   (a) Propose a move for the ith particle, located at $\boldsymbol{\theta}_i^{(t-1)}$, according to the following rule:
   $$\boldsymbol{\theta}_i' = \boldsymbol{\theta}_i^{(t-1)} + \kappa \boldsymbol{\xi}^{(t)}, \tag{9}$$
   where κ is a user-defined parameter that represents the maximum allowed displacement per iteration, while $\boldsymbol{\xi}^{(t)} \sim \mathcal{U}([-1,1)\times[-1,1))$.
   (b) Compute the change in energy caused by the move, $\Delta E_i^{(t)}$. If the energy is not increased (i.e., $\Delta E_i^{(t)} \leq 0$), then accept the move, setting $\boldsymbol{\theta}_i^{(t)} = \boldsymbol{\theta}_i'$. Otherwise (i.e., if the energy is increased and thus $\Delta E_i^{(t)} > 0$), accept the move with probability $\exp(-\Delta E_i^{(t)}/K\mathcal{T})$, thus setting $\boldsymbol{\theta}_i^{(t)} = \boldsymbol{\theta}_i'$, or reject it with probability $1 - \exp(-\Delta E_i^{(t)}/K\mathcal{T})$, thus letting $\boldsymbol{\theta}_i^{(t)} = \boldsymbol{\theta}_i^{(t-1)}$.
3. All the different configurations, $\boldsymbol{\theta}_i^{(t)}$ for $t = 1, \ldots, T$, are used to compute the desired averages.

Intuitively, the MH algorithm can be seen as a generalized rejection sampler whose proposal depends on the result of the previous iteration (i.e., on $\boldsymbol{\theta}^{(t-1)}$). Furthermore, the acceptance rate also depends on $\boldsymbol{\theta}^{(t-1)}$ and the value of $\boldsymbol{\theta}^{(t-1)}$ is re-used whenever a candidate sample $\boldsymbol{\theta}'$ is rejected. This creates an undesired effect, since the drawn samples are no longer independent as in the RS algorithm, but allows us to work with proposal densities that may lie below the target. This is due to the fact that the underlying Markov chain has the desired target as the limiting invariant distribution (e.g., see [104] for a rigorous proof). Another useful perspective is to view the method as a thinning of a random walk in precisely the right way to ensure convergence to the correct target. Loosely speaking, the chain is thinned by discarding those candidates which correspond to moves from the current state that happen too often, and this is done with the right probability to ensure that the invariant distribution of the Markov chain is exactly the desired target. See the excellent tutorial (but rigorous) exposition of the MH algorithm provided by Chib and Greenberg for further information about this issue [53].

In this algorithm, the proposal for the tth iteration and the ith particle is $\bar{q}(\boldsymbol{\theta}_i|\boldsymbol{\theta}_{1:N}^{(t-1)}) = \bar{q}(\boldsymbol{\theta}_i|\boldsymbol{\theta}_i^{(t-1)}) = \mathcal{U}([\theta_{i,1}^{(t-1)}-\kappa, \theta_{i,1}^{(t-1)}+\kappa)\times[\theta_{i,2}^{(t-1)}-\kappa, \theta_{i,2}^{(t-1)}+\kappa))$, whereas the target is $\bar{\pi}(\boldsymbol{\theta}) \propto \exp(-E(\boldsymbol{\theta})/K\mathcal{T})$. The acceptance probability is then given by

$$\alpha\left(\mathbf{x}_i', \mathbf{x}_{1:N}^{(t-1)}\right) = \min\left[1, \exp(-\Delta E_i^{(t)}/K\mathcal{T})\right], \tag{10}$$

with $\Delta E_i^{(t)} = E\left(\boldsymbol{\theta}_1^{(t-1)}, \ldots, \boldsymbol{\theta}_{i-1}^{(t-1)}, \boldsymbol{\theta}_i', \boldsymbol{\theta}_{i+1}^{(t-1)}, \ldots, \boldsymbol{\theta}_N^{(t-1)}\right) - E\left(\boldsymbol{\theta}_1^{(t-1)}, \ldots, \boldsymbol{\theta}_N^{(t-1)}\right)$. This acceptance probability guarantees the ergodicity of the chain and the convergence of the algorithm to the desired target PDF [43], but is not the only valid acceptance rule. Indeed, in 1965 Barker proposed an alternative acceptance probability for the computation of radial distribution functions in plasmas [45]:

$$\alpha(\boldsymbol{\theta}_i', \boldsymbol{\theta}_{1:N}^{(t-1)}) = \frac{1}{1 + \exp(-\Delta E_i^{(t)}/K\mathcal{T})}. \tag{11}$$

Soon afterwards, Hastings generalized these two acceptance probabilities, allowing for non-symmetric proposals (unlike the proposals considered both by Metropolis and Barker, which were both symmetric) [44]. Using our notation, where the parameters to be estimated are denoted as $\boldsymbol{\theta}$, the two acceptance rules (α_M and α_B denote the generalization of Metropolis' and Barker's acceptance rules, respectively) become:

[11] Nowadays this would be considered an application of the IS methodology, since the particles are not drawn from the desired target density, $\bar{\pi}(\boldsymbol{\theta}) \propto \exp(-E(\boldsymbol{\theta})/K\mathcal{T})$, but from a uniform random proposal. Thus, they are assigned the corresponding standard IS weights, $w(\boldsymbol{\theta}) \propto \exp(-E(\boldsymbol{\theta})/K\mathcal{T})$, for the subsequent integration (see Section 4.2 for a description of IS methods).

[12] Note that we have used $\boldsymbol{\theta}$ to represent the positions of the particles (which are the parameters to be inferred), as in the rest of the paper, instead of the original notation of [43], which used $\mathbf{x}$.

[13] Note that the way in which the samples are drawn actually corresponds to the *MH-one-at-a-time* algorithm [75], which is equivalent to the well-known MH-within-Gibbs algorithm (see Section 3.1.4) with one iteration of the internal MH method per iteration of the external systematic scan Gibbs sampler (see Section 3.1.3), since the particles are moved one at a time and the proposed moves are not performed according to the desired target PDF, but using a simpler proposal (and thus they can be accepted or rejected).

$$\alpha_M(\boldsymbol{\theta}', \boldsymbol{\theta}^{(t-1)}) = \min\left[1, \frac{\pi(\boldsymbol{\theta}')q(\boldsymbol{\theta}^{(t-1)}|\boldsymbol{\theta}')}{\pi(\boldsymbol{\theta}^{(t-1)})q(\boldsymbol{\theta}'|\boldsymbol{\theta}^{(t-1)})}\right], \tag{12a}$$

$$\alpha_B(\boldsymbol{\theta}', \boldsymbol{\theta}^{(t-1)}) = \frac{\pi(\boldsymbol{\theta}')q(\boldsymbol{\theta}^{(t-1)}|\boldsymbol{\theta}')}{\pi(\boldsymbol{\theta}')q(\boldsymbol{\theta}^{(t-1)}|\boldsymbol{\theta}') + \pi(\boldsymbol{\theta}^{(t-1)})q(\boldsymbol{\theta}'|\boldsymbol{\theta}^{(t-1)})}, \tag{12b}$$

Finally, in 1973 Peskun proved that the acceptance rule of Eq. (12a) was optimal [46], and this settled the structure of the algorithm used nowadays [152].

The MH algorithm with the acceptance rule of Eq. (12a) is summarized in Algorithm 3. The burn-in period (T_b) is the number of initial samples removed from the empirical average in Eq. (14), which is used to compute the desired estimator, in order to guarantee that the chain has converged approximately to its stationary distribution. This period can be estimated automatically (e.g., see Section 3.5 for a brief discussion on this issue and [153] for a comparative review of different techniques to assess the convergence of a Markov chain and thus determine the burn-in period) or set to some pre-defined value, and is required by all MCMC algorithms.

Algorithm 3 Metropolis-Hastings (MH) algorithm.

1 **Initialization:** Choose a proposal function $q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t-1)})$, an initial state $\boldsymbol{\theta}^{(0)}$, the total number of iterations (T), and the burn-in period (T_b).

2 FOR $t = 1, \ldots, T$:

(a) Draw $\boldsymbol{\theta}' \sim q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t-1)})$ and $u \sim \mathcal{U}([0, 1))$.

(b) Compute the acceptance probability:

$$\alpha_t \equiv \alpha(\boldsymbol{\theta}', \boldsymbol{\theta}^{(t-1)}) = \min\left[1, \frac{\pi(\boldsymbol{\theta}')q(\boldsymbol{\theta}^{(t-1)}|\boldsymbol{\theta}')}{\pi(\boldsymbol{\theta}^{(t-1)})q(\boldsymbol{\theta}'|\boldsymbol{\theta}^{(t-1)})}\right]. \tag{13}$$

(c) If $u \leq \alpha_t$, accept $\boldsymbol{\theta}'$ and set $\boldsymbol{\theta}^{(t)} = \boldsymbol{\theta}'$. Otherwise (i.e., if $u > \alpha_t$), reject $\boldsymbol{\theta}'$ and set $\boldsymbol{\theta}^{(t)} = \boldsymbol{\theta}^{(t-1)}$.

3 Approximate the integral in Eq. (2) as

$$\widehat{I}_{T-T_b} = \frac{1}{T - T_b} \sum_{t=T_b+1}^{T} g(\boldsymbol{\theta}^{(t)}). \tag{14}$$

One of the main advantages of the MH algorithm is that it is a very generic method that admits the use of almost any proposal and target PDFs. However, although the algorithm is valid regardless of the shape and parameters of the proposal PDF (see Section 3.1.2 for a brief review of the specific conditions for the validity of the MH algorithm), the speed of convergence and the quality of the estimators obtained substantially depend on the quality of this proposal. Many choices are possible, but here we will only consider the two most widely used (see [53] for a brief discussion on five different families of proposals):

- *Independent MH:* The proposal is fixed and does not depend on the current state of the chain, i.e., $\bar{q}(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t-1)}) = \bar{q}(\boldsymbol{\theta})$. For instance, a widely used choice in this case is a multi-variate Gaussian PDF with fixed mean vector and covariance matrices: $\bar{q}(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\mu}, \mathbf{C})$. An independent proposal can be considered a *global proposal*, since it can generate candidate samples in the whole state space regardless of the current state of the chain. This type of proposal fosters the exploration of the state space, but its performance can be poor for complicated target PDFs (especially for high-dimensional state spaces, where it can be difficult to find a good parameterization).
- *Random walk MH:* The proposal is centered on the current state of the chain, i.e., the proposed candidate at the tth iteration can be expressed as $\boldsymbol{\theta}' = \boldsymbol{\theta}^{(t-1)} + \boldsymbol{\vartheta}'$, where $\boldsymbol{\vartheta}' \sim p(\boldsymbol{\vartheta}|\mathbf{0}, \mathbf{C}_\vartheta)$ and $p(\boldsymbol{\vartheta}|\mu, \mathbf{C}_\vartheta)$ is an arbitrary PDF specified using a location parameter μ and a scale parameter $\mathbf{C}$. For instance, using a Gaussian PDF for $\boldsymbol{\vartheta}$ we have $\boldsymbol{\vartheta}' \sim \mathcal{N}(\boldsymbol{\vartheta}|\mathbf{0}, \mathbf{C}_\vartheta)$, which implies that $\boldsymbol{\theta}' \sim \bar{q}(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t-1)}) = \mathcal{N}(\boldsymbol{\vartheta}|\boldsymbol{\theta}^{(t-1)}, \mathbf{C}_\vartheta)$. If the PDF of $\boldsymbol{\vartheta}$ is symmetric (i.e., $q(\boldsymbol{\vartheta}|\boldsymbol{\vartheta}^{(t-1)}) = q(\boldsymbol{\vartheta}^{(t-1)}|\boldsymbol{\vartheta})$), then the acceptance rule becomes:

$$\alpha(\boldsymbol{\theta}', \boldsymbol{\theta}^{(t-1)}) = \min\left[1, \frac{\pi(\boldsymbol{\theta}')}{\pi(\boldsymbol{\theta}^{(t-1)})}\right]. \tag{15}$$

This is the type of proposal used by Metropolis et al. (with a uniform distribution for $\boldsymbol{\vartheta}$) in [43], which led them to the simplified acceptance probability shown in Eq. (10). A random walk proposal can be seen as a local proposal, since it is centered on the current state of the chain. Hence, the random walk MH algorithm encourages a more local exploration around the current state.

A critical issue for the good performance of the MH algorithm is the acceptance rate (AR), which depends on the variance of the proposal PDF and should be neither too high nor too low. On the one hand, a high variance typically leads to a low AR, thus implying that the MH algorithm gets stuck because most candidate samples are rejected. On the other hand, a low variance can easily lead to a high AR, as only local moves around previously accepted samples are proposed, but can result in the MH algorithm failing to explore the target. The seminal work of Roberts, Gelman, and Wilks proved, for the random walk MH algorithm and in a simplified setting,

that the proposal's variance should be tuned in such a way that the average acceptance rate is roughly 1/4 [154]. In [155], the same authors delved deeper into this issue, showing that the optimal acceptance rate is approximately 44% for $D_\theta = 1$ and declines to 23% when $D_\theta \to \infty$. These results can be extended to different settings and other methods based on the MH algorithm, like MH-within-Gibbs or Hamiltonian MC (see Sections 3.1.4 and 3.3.2, respectively), and have lead to the practical rule of thumb of choosing the variance of the proposal in order to ensure and acceptance rate between 25 and 40%. However, let us remark that several authors have proved that the optimal AR can be substantially different for other settings/methods. For instance, Bédard and Rosenthal have recently warned that the asymptotically optimal AR can be significantly different from the well-known 0.234 AR when the target's components are not independent [156]. Indeed, in [157, 158] Bédard showed that 0.234 is the upper limit for the AR in the simplified model considered, but much lower ARs can actually be optimal. Other authors have also found that higher acceptance rates can be optimal for other algorithms that make use of gradient information, like the simplified Langevin algorithm (SLA) or the modified adaptive Langevin algorithm (MALA) (see Section 3.3.1) [159, 160].

Finally, let us remark that the local and global proposals, used by the independent and random walk MH algorithms respectively, can be combined. For instance, [161] proposes using the following *small world proposal*:

$$q(\boldsymbol{\theta}) = (1-p)\cdot q_G(\boldsymbol{\theta}) + p\cdot q_L(\boldsymbol{\theta}), \tag{16}$$

where $q_L(\boldsymbol{\theta})$ is a local proposal centered around the current state of the chain, $q_G(\boldsymbol{\theta})$ is a global proposal that allows for "wild" moves far away from the current state, and p is a small probability. Using this proposal leads to an MH algorithm with improved performance, especially for complicated heterogeneous spaces and multi-modal distributions, and can turn slowly mixing into rapidly mixing chains [161, 162].

3.1.2 Validity of the Metropolis-Hastings algorithm

Let us now take at the conditions when the MH algorithm (Alg. 3) produces samples from the desired target PDF. In order to analyze its output, let us first notice that the states $\boldsymbol{\theta}^{(1)}, \boldsymbol{\theta}^{(2)}, \ldots$ form a Markov chain with a certain transition density $K(\boldsymbol{\theta}^{(t)}|\boldsymbol{\theta}^{(t-1)})$. The key trick of the MH algorithm is that the algorithm has been constructed in such a way that the stationary PDF of the Markov chain is the target PDF:

$$\bar{\pi}(\boldsymbol{\theta}') = \int_\Theta K(\boldsymbol{\theta}'|\boldsymbol{\theta})\bar{\pi}(\boldsymbol{\theta})d\boldsymbol{\theta}. \tag{17}$$

One way to ensure the above is the *detailed balance condition*, which demands that

$$K(\boldsymbol{\theta}'|\boldsymbol{\theta})\bar{\pi}(\boldsymbol{\theta}) = K(\boldsymbol{\theta}|\boldsymbol{\theta}')\bar{\pi}(\boldsymbol{\theta}'). \tag{18}$$

Integrating both sides over $\boldsymbol{\theta}$ and recalling $\int_\Theta K(\boldsymbol{\theta}|\boldsymbol{\theta}')d\boldsymbol{\theta} = 1$ now gives

$$\int_\Theta K(\boldsymbol{\theta}'|\boldsymbol{\theta})\bar{\pi}(\boldsymbol{\theta})d\boldsymbol{\theta} = \int_\Theta K(\boldsymbol{\theta}|\boldsymbol{\theta}')\bar{\pi}(\boldsymbol{\theta}')d\boldsymbol{\theta} = \bar{\pi}(\boldsymbol{\theta}'), \tag{19}$$

which shows that $\bar{\pi}(\boldsymbol{\theta})$ is the stationary PDF of the Markov chain. Furthermore, this condition also ensures that the Markov chain is reversible [104–106, 152]. The transition PDF of the MH algorithm consist of two parts—the PDF of the accepted samples and the PDF of the rejected samples. It can thus be written in the following form:

$$K(\boldsymbol{\theta}'|\boldsymbol{\theta}) = \alpha(\boldsymbol{\theta}', \boldsymbol{\theta})\, q(\boldsymbol{\theta}'|\boldsymbol{\theta}) + \left(1 - \int_\Theta \alpha(\boldsymbol{\theta}', \boldsymbol{\theta}) q(\boldsymbol{\theta}'|\boldsymbol{\theta})d\boldsymbol{\theta}'\right)\delta(\boldsymbol{\theta}' - \boldsymbol{\theta}). \tag{20}$$

By direct computation, it can be easily verified that the detailed balance condition is satisfied (see also Theorem 7.2 of [104]).

In addition to having the correct stationary PDF, we also need to ensure that the Markov chain is ergodic. The ergodicity property ensures that the Markov chain converges to the stationary distribution with a predefined rate so that we can estimate expectations of the state distributions by computing time averages. A sufficient condition for ergodity is to ensure that the Markov chain is also an aperiodic $\bar{\pi}$-irreducible Harris chain, which can be ensured by the following conditions (see Equations 7.4 and 7.5 and Lemma 7.6 in [104]):[14]

1. The stationary distribution and the proposal PDF satisfy $P[\,\bar{\pi}(\boldsymbol{\theta})q(\boldsymbol{\theta}'|\boldsymbol{\theta}) \le \bar{\pi}(\boldsymbol{\theta}')q(\boldsymbol{\theta}|\boldsymbol{\theta}')] < 1$.
2. The proposal PDF is strictly positive everywhere in the parameter space, i.e., $q(\boldsymbol{\theta}'|\boldsymbol{\theta}) > 0$ for all $\boldsymbol{\theta}', \boldsymbol{\theta} \in \Theta$.

Provided that the detailed balance condition and the aforementioned properties are satisfied, then Corollaries 7.5 and 7.7 in [104] ensure the following ergodicity properties for the MH Markov chain:

$$\lim_{T\to\infty} \sum_{t=1}^{T} g(\boldsymbol{\theta}^{(t)}) = \int_\Theta g(\boldsymbol{\theta})\bar{\pi}(\boldsymbol{\theta})d\boldsymbol{\theta}', \tag{21a}$$

$$\lim_{n\to\infty} \left\| \int_\Theta K^n(\cdot|\boldsymbol{\theta}')\bar{\pi}_0(\boldsymbol{\theta}')d\boldsymbol{\theta} - \bar{\pi} \right\|_{TV} = 0, \tag{21b}$$

where g is an arbitrary L_1 function, $\|\cdot\|_{TV}$ is the total variation norm, K^n denotes the n-step transition kernel, and

[14] These conditions for the proposal density can be slightly relaxed (e.g., see Lemma 7.6 in [104]).

$\bar{\pi}_0$ is an arbitrary initial PDF. Eq. (21a) guarantees that the sample average converges to the true value of the integral, whereas (21b) ensures that the chain's PDF converges to the target PDF regardless of the initial density.

The aforementioned conditions ensure that the chain converges to the target distribution and that time averages can be used to approximate expectations. However, the convergence of the algorithm can still be arbitrarily slow. In order to guarantee that the chain does not get stuck in some region of parameter space for large amounts of time, we need MCMC algorithms which are *geometrically ergodic*. An MCMC algorithm is geometrically ergodic if

$$\left\| \int_{\Theta} K^n(\cdot|\theta')\bar{\pi}_0(\theta')\, d\theta' - \bar{\pi} \right\|_{\mathrm{TV}} < C_{\bar{\pi}_0}\rho^n, \tag{22}$$

for some $C_{\bar{\pi}_0}$ and $0 < \rho < 1$ giving the convergence rate. There are two main reasons why geometric ergodicity is essential. On the one hand, geometric ergodicity guarantees the existence of a Central Limit Theorem which enables error bounds to be developed. On the other hand, without geometric ergodicity algorithms are more-or-less guaranteed to give rise to sample paths with "heavy-tailed excursions" far away from the center of the distribution, thus leading to instability and inaccuracy of the subsequent parameter estimation procedures. See [163] and [164] for a more detailed discussion on geometric ergodicity on the one-dimensional and multi-dimensional cases, respectively.

3.1.3 Gibbs sampler

The Gibbs sampler was introduced by Stuart Geman and Donald Geman in 1984 in order to sample from the Markov Random Field (MRF) induced by the Gibbs distribution [48]. The application considered was the Bayesian restoration of images degraded by blurring, nonlinear deformations, and multiplicative or additive noise[15]. In order to deal with these distortions, Geman and Geman proposed a stochastic relaxation algorithm that relied on iteratively making local random changes in the image based on current values of the pixels. A simulated annealing approach, that gradually lowers the system's "temperature" [165], was used to avoid local maxima. More precisely, using our notation the Gibbs sampler proposed in [48] was the following:

1. Select an arbitrary configuration of the pixels, $\boldsymbol{\theta}^{(0)} = \left[\theta_1^{(0)}, \ldots, \theta_{D_\theta}^{(0)}\right]^\top$.
2. Select the sequence of pixels $(n_1, n_2, \ldots)$ that will be visited for replacement. The sequence used in [48] corresponded to a raster scan of the image (i.e., repeteadly visiting all the sites in some "natural" fixed order), but this sequence does not necessarily have to be periodic.
3. At the tth "epoch" ($t = 1, 2, 3, \ldots$), update the n_tth pixel by drawing a sample from the conditional PDF of θ_{n_t} given the current value of the remaining pixels, $\theta_{n_t}^{(t)} \sim \bar{\pi}\left(\theta_{n_t}|\boldsymbol{\theta}_{\neg n_t}^{(t-1)}\right)$ with $\boldsymbol{\theta}_{\neg n_t}^{(t-1)} = [\theta_1^{(t-1)}, \ldots, \theta_{n_t-1}^{(t-1)}, \theta_{n_t+1}^{(t-1)}, \ldots, \theta_{D_\theta}^{(t-1)}]^\top$.
4. Repeat step 3 until a pre-specified termination condition (e.g., a fixed number of iterations T) is fulfilled.

This approach can be easily generalized and adapted to many practical problems. Algorithm 4 provides a generic version of the Gibbs sampler with an arbitrary selection of the indices to be sampled. As already mentioned in the introduction, Gelman showed that the Gibbs sampler is a particular case of the MH algorithm [50]. This can be easily seen by considering the MH algorithm (Algorithm 3) with a proposal at the tth iteration given by $q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t-1)}) = \pi\left(\theta_{d_t}|\boldsymbol{\theta}_{\neg d_t}^{(t-1)}\right)\delta\left(\boldsymbol{\theta}_{\neg d_t} - \boldsymbol{\theta}_{\neg d_t}^{(t-1)}\right)$, where $\delta(\cdot)$ denotes Dirac's delta. Then, $\boldsymbol{\theta'} = \left[\theta_1^{(t-1)}, \ldots, \theta_{d_t-1}^{(t-1)}, \theta'_{d_t}, \theta_{d_t+1}^{(t-1)}, \ldots, \theta_{D_\theta}^{(t-1)}\right]$ with $\theta'_{d_t} \sim \pi\left(\theta_{d_t}|\boldsymbol{\theta}_{\neg d_t}^{(t-1)}\right)$, just like in the tth iteration of the Gibbs sampler of Algorithm 4. Now, we just need to prove that $\boldsymbol{\theta'}$ is always accepted, as it happens in the Gibbs sampler. Noting that $\pi(\boldsymbol{\theta}) = \pi(\theta_{d_t}|\boldsymbol{\theta}_{\neg d_t})\pi(\boldsymbol{\theta}_{\neg d_t})$ by the chain rule of probability, the ratio inside the acceptance probability (α_t) of the MH algorithm becomes:

$$\begin{aligned}&\frac{\pi(\boldsymbol{\theta'})q\left(\boldsymbol{\theta}^{(t-1)}|\boldsymbol{\theta'}\right)}{\pi\left(\boldsymbol{\theta}^{(t-1)}\right)q\left(\boldsymbol{\theta'}|\boldsymbol{\theta}^{(t-1)}\right)}\\ &\quad= \frac{\pi\left(\theta'_d|\boldsymbol{\theta}_{\neg d_t}^{(t-1)}\right)\pi\left(\boldsymbol{\theta}_{\neg d_t}^{(t-1)}\right)\pi\left(\theta_d^{(t-1)}|\boldsymbol{\theta}_{\neg d_t}^{(t-1)}\right)}{\pi\left(\theta_d^{(t-1)}|\boldsymbol{\theta}_{\neg d_t}^{(t-1)}\right)\pi\left(\boldsymbol{\theta}_{\neg d_t}^{(t-1)}\right)\pi\left(\theta'_d|\boldsymbol{\theta}_{\neg d_t}^{(t-1)}\right)} = 1.\end{aligned}$$

Hence, the proposed sample (drawn from the d_tth full conditional PDF) is always accepted and only the d_tth coordinate is updated at the tth iteration, just like in the Gibbs sampler.

Note that we still have to specify how to select the coordinates to be sampled. In general it may be difficult to determine the best type of scan for a Gibbs sampler, as shown by Roberts and Rosenthal in [166], and many alternative approaches can be devised. However, the three most widely used schemes are the following [104]:

- *Systematic scan:* The parameters are updated according to some pre-specified "canonical" order. Without loss of generality, let us consider that this order is simply $\theta_1, \theta_2, \ldots, \theta_{D_\theta}$. Then, we have the

[15] In [48], the authors use MAP estimators for the Bayesian restoration task, since they believe that "the MAP formulation is well-suited to restoration, particularly for handling general forms of spatial degradation." However, they also state that "minimum mean-square error (MMSE) estimation is also feasible by using the (temporal) ergodicity of the relaxation chain to compute *means* w.r.t. the posterior distribution."

Algorithm 4 Generic Gibbs sampler.

1 **Initialization:** Choose an initial state $\boldsymbol{\theta}^{(0)}$, the total number of iterations (T), and the burn-in period (T_b).
2 FOR $t = 1, \ldots, T$:

(a) Select the coordinate to be sampled, $d_t \in \{1, \ldots, D_\theta\}$, using some of the approaches described below.
(b) Draw $\theta_{d_t}^{(t)} \sim \bar{\pi}(\theta_{d_t} | \boldsymbol{\theta}_{\neg d_t}^{(t-1)})$, with $\boldsymbol{\theta}_{\neg d_t}^{(t-1)} = [\theta_1^{(t-1)}, \ldots, \theta_{d_t-1}^{(t-1)}, \theta_{d_t+1}^{(t-1)}, \ldots, \theta_{D_\theta}^{(t-1)}]^\top$.

3 Approximate the integral in Eq. (2) using Eq. (14).

following sequence of coordinates to be updated: $d_1 = 1, d_2 = 2, \ldots, d_{D_\theta} = D_\theta, d_{D_\theta+1} = 1, d_{D_\theta+2} = 2, \ldots, d_{2D_\theta} = D_\theta, d_{2D_\theta+1} = 1, \ldots$ This can be expressed more compactly as $d_t = ((t-1))_{D_\theta} + 1$, where $((t))_{D_\theta}$ denotes the modulo operation: $((t))_{D_\theta} = m \iff t = kD_\theta + m$ for some $k, m \in \mathbb{Z}$ with $m \in \{0, 1, \ldots, D_\theta - 1\}$ and $-\infty < k < \infty$. In this particular case, the Gibbs sampler in Algorithm 4 can be expressed using a double FOR loop, with the inner loop running sequentially over the different parameters, as shown in Algorithm 5. In this systematic scan Gibbs sampler, which is probably the most widely used version of the algorithm in signal processing applications, one iteration of the Gibbs sampler corresponds to one step of the outer loop. Note that the total number of samples drawn from the full conditional PDFs in Algorithm 5 is TD_θ, whereas in Algorithm 4 only T samples were drawn. Finally, note that the Markov chain induced by the systematic scan Gibbs sampler is non-reversible [104].

- *Symmetric scan:* The coordinates are also explored following a pre-specified deterministic order [104]: first in an ascending order and then in a descending order, and this scheme is repeated periodically, i.e., $d_1 = 1, d_2 = 2, \ldots, d_{D_\theta} = D_\theta, d_{D_\theta+1} = D_\theta - 1, d_{D_\theta+2} = D_\theta - 2, \ldots, d_{2D_\theta-1} = 1, d_{2D_\theta} = 1, d_{2D_\theta+1} = 2, \ldots$ Using the modulo notation, $d_t = \min\{((t-1))_{2D_\theta-2}, ((-t))_{2D_\theta-2}\}$[16]. Unlike the systematic scan, the symmetric scan leads to a reversible Markov chain and can also result in an improved performance. The symmetric Gibbs sampler can also be expressed using a double FOR loop, as shown in Algorithm 6, with one iteration of the Gibbs sampler corresponding to one step of the outer loop. Now, the total number of samples drawn from the full conditional PDFs is $T(2D_\theta - 1)$.
- *Random scan:* This method was proposed originally by Liu et al. [167]. In this case, the parameter to be updated is selected randomly at each iteration, typically following a uniform distribution, i.e., $d_t \sim \boldsymbol{\mathcal{U}}(\{1, 2, \ldots, D_\theta\})$. This scheme also produces a reversible Markov chain and can lead to an improved performance w.r.t. the symmetric scan Gibbs sampler[17].

Algorithm 5 Systematic scan Gibbs sampler.

1 **Initialization:** Choose an initial state $\boldsymbol{\theta}^{(0)}$, the total number of iterations (T), and the burn-in period (T_b).
2 FOR $t = 1, \ldots, T$:

(a) Draw $\theta_1^{(t)} \sim \bar{\pi}(\theta_1 | \boldsymbol{\theta}_{2:D_\theta}^{(t-1)})$.
(b) FOR $d = 2, \ldots, D_\theta - 1$:

- Draw $\theta_d^{(t)} \sim \bar{\pi}(\theta_d | \boldsymbol{\theta}_{1:d-1}^{(t)}, \boldsymbol{\theta}_{d+1:D_\theta}^{(t-1)})$.

(c) Draw $\theta_{D_\theta}^{(t)} \sim \bar{\pi}(\theta_{D_\theta} | \boldsymbol{\theta}_{1:D_\theta-1}^{(t)})$.

3 Approximate the integral in Eq. (2) using Eq. (14).

Algorithm 6 Symmetric scan Gibbs sampler.

1 **Initialization:** Choose an initial state $\boldsymbol{\theta}^{(0)}$, the total number of iterations (T), and the burn-in period (T_b).
2 FOR $t = 1, \ldots, T$:

(a) Draw $\theta_1' \sim \bar{\pi}(\theta_1 | \boldsymbol{\theta}_{2:D_\theta}^{(t-1)})$.
(b) FOR $d = 2, \ldots, D_\theta - 1$:

- Draw $\theta_d' \sim \bar{\pi}(\theta_d | \boldsymbol{\theta}_{1:d-1}', \boldsymbol{\theta}_{d+1:D_\theta}^{(t-1)})$.

(c) Draw $\theta_{D_\theta}^{(t)} \sim \bar{\pi}(\theta_{D_\theta} | \boldsymbol{\theta}_{1:D_\theta-1}')$.
(d) FOR $d = D_\theta - 1, \ldots, 2$:

- Draw $\theta_d^{(t)} \sim \bar{\pi}(\theta_d | \boldsymbol{\theta}_{1:d-1}', \boldsymbol{\theta}_{d+1:D_\theta}^{(t)})$.

(e) Draw $\theta_1^{(t)} \sim \bar{\pi}(\theta_1 | \boldsymbol{\theta}_{2:D_\theta}^{(t)})$.

3 Approximate the integral in Eq. (2) using Eq. (14).

Note that only the samples corresponding to the outer loops in Algorithms 5 and 6 (i.e., $\boldsymbol{\theta}^{(t)} = [\theta_1^{(t)}, \ldots, \theta_{D_\theta}^{(t)}]^\top$)

[16] Note that the modulo notation is very convenient, since it leads to a straightforward computation of the sequence of indexes. For instance, in MATLAB the sequence of indexes for the systematic scan Gibbs sampler is obtained as `dt = mod(t-1,Dpar)`, whereas for the symmetric scan it is given by `dt = min(mod(t-1,2*Dpar-2),mod(-t,2*Dpar-2))`, with `Dpar` indicating the dimension of the parameter space. Moreover, since these sequences are deterministic, they can be easily pre-computed and stored for further use when T is fixed a priori.

[17] Note that the sequence of indexes for the random scan Gibbs sampler can also be pre-computed when T is fixed a priori. In this case, this sequence is obtained by the following Matlab command: `dt = randi(Dpar,1,T)`, with `Dpar` indicating again the dimension of the parameter space.

are typically used to compute the approximate estimator of Eq. (14). This entails an inefficient use of the generated samples w.r.t. the generic Gibbs sampler of Algorithm 4, which uses all the drawn samples to compute the approximate estimator of Eq. (14). However, "nothing prevents the use of all the simulations [samples] in integral approximations", as stated by Robert and Casella [104]. Indeed, it has been shown very recently that using all the intermediate samples, both in the Gibbs and MH-within-Gibbs (see Section 3.1.4) samplers, can result in a substantial improvement in performance in some cases [168].

Regarding the convergence of the Gibbs sampler, [48, 169] provide regularity conditions under which the Gibbs sampler is ergodic and the distribution of $\boldsymbol{\theta}^{(t)}$ converges to the target distribution as $t \to \infty$, whereas [52] provides a simple convergence proof. In short, the convergence of the Gibbs sampler essentially requires that all the coordinates keep being updated as the algorithm proceeds, implying that every coordinate is visited infinitely often as $t \to \infty$.

Finally, note that there is no need to sample each of the D_θ parameters individually. Indeed, if a certain subset of parameters can be easily sampled jointly given the rest, then we can group them together inside the loop of Algorithm 4 (and also in Algorithms 5 and 6). Let us assume that the D_θ parameters in $\boldsymbol{\theta} = [\theta_1, \ldots, \theta_{D_\theta}]^\top$ can be grouped into N_g disjoint groups in such a way that $\boldsymbol{\vartheta} = [\vartheta_1, \ldots, \vartheta_{N_g}]^\top$ contains all the parameters to be inferred. Then, Algorithm 4 can be applied on $\boldsymbol{\vartheta}$ instead of $\boldsymbol{\theta}$, drawing $\vartheta_{d_t}^{(t)} \sim \bar{\pi}(\vartheta_{d_t} | \boldsymbol{\vartheta}_{\neg d_t}^{(t-1)})$. This algorithm is known as the *group* or *block Gibbs sampler*. Alternatively, if a subset of parameters can be easily sampled given the rest, we can remove them from the loop of the Gibbs sampler. Without loss of generality, let us assume that we keep the first $\widetilde{D}_\theta$ parameters and leave the remaining parameters outside of the iterations of the Gibbs sampler, i.e., we only draw samples from the reduced set of parameters $\widetilde{\boldsymbol{\theta}} = [\theta_1, \ldots, \theta_{\widetilde{D}_\theta}]^\top$. Then, Algorithm 4 can be applied on $\widetilde{\boldsymbol{\theta}}$ instead of $\boldsymbol{\theta}$, drawing $\theta_{d_t}^{(t)} \sim \bar{\pi}(\theta_{d_t} | \widetilde{\boldsymbol{\theta}}_{\neg d_t}^{(t-1)})$ with $d_t \in \{1, \ldots, \widetilde{D}_\theta\}$. When the chain has converged, then we can easily sample from the remaining parameters given the samples from the first $\widetilde{D}_\theta$ parameters obtained using the Gibbs sampler. This algorithm is known as the *collapsed Gibbs sampler*. Although the addition of auxiliary variables can speed up the convergence of the Gibbs sampler in some cases (e.g., see the data augmentation algorithm in Section 3.1.5), in general grouping or collapsing down variables leads to improved convergence and decreased sample autocovariances, as shown by Liu in [170]. However, let us remark that Liu's proof is highly restrictive and in some cases the uncollapsed sampler can actually converge faster than the collapsed one (e.g., see the counterexample in Appendix A of Terenin et al. [171]). Finally, note also that finding the optimal variables to group or collapse in order to achieve the optimal performance depends on the problem and can be a very difficult task.

The Gibbs sampler is a fundamental algorithm for parameter estimation in many signal processing and machine learning problems. Indeed, it may be the only choice for some models, because it is well-defined even on discrete state spaces where gradients are not available and good Metropolis-Hastings proposals are difficult to construct. Therefore, it has been extensively used in practical applications either as a stand-alone method or combined with the MH algorithm as described in the following section.

3.1.4 MH-within-Gibbs

The Gibbs sampler requires sampling from the full univariate conditional PDFs. Unfortunately, although this should be a much easier task than sampling from the multi-variate posterior PDF, in many real-world applications these conditional PDFs have non-standard forms and we cannot sample directly from them. Initially, some authors tackled this problem by using the RS algorithm (e.g., see [172]), and the adaptive RS (ARS) algorithm was specifically designed for this task [141]. However, as already mentioned before, both the RS and ARS algorithms require finding a bounding constant C such that $Cq(\boldsymbol{\theta}) \geq \pi(\boldsymbol{\theta})$, a task that may be difficult for complicated targets and lead to very inefficient sampling if C is large. In this section, we briefly discuss a widely used technique developed to address this problem, the MH-within-Gibbs algorithm (often also called *Component-wise MH* method), as well as two related methods: the griddy Gibbs sampler and the fast universal self-tuned sampler (FUSS).

In order to sample from non-standard full conditional PDFs, Ritter and Tanner proposed the so called *griddy Gibbs sampler* [173, 174]. Their basic idea was using a set of evaluations from the desired full conditional PDF to build a piecewise approximation from which sampling is straightforward. The tth iteration of the griddy Gibbs sampler for the dth coordinate ($1 \leq d \leq D_\theta$) proceeds as follows:

1. Evaluate the target at some pre-specified set of parameters, $\boldsymbol{\mathcal{S}}_d^{(t)} = \{\theta_{d,1}^{(t)}, \ldots, \theta_{d,K}^{(t)}\}$, obtaining $P_{d,1}^{(t)} = \pi\left(\theta_{d,1}^{(t)} | \boldsymbol{\theta}_{\neg d}^{(t-1)}\right), \ldots, P_{d,K}^{(t)} = \pi\left(\theta_{d,K}^{(t)} | \boldsymbol{\theta}_{\neg d}^{(t-1)}\right)$.
2. Construct an approximate inverse cumulative distribution function (CDF) of the target, $\hat{\Pi}^{-1}\left(\theta_d | \boldsymbol{\theta}_{\neg d}^{(t-1)}, \boldsymbol{\mathcal{S}}_d^{(t)}\right)$, using $P_{d,1}^{(t)}, \ldots, P_{d,K}^{(t)}$ and a piecewise constant (PWC) or piecewise linear (PWL) approximation.

3 Draw $u \sim \mathcal{U}([0,1))$ and apply the *inverse method* [119] to obtain a sample drawn approximately from the target as $\theta_d^{(t)} = \hat{\Pi}^{-1}(u|\boldsymbol{\theta}_{\neg d}^{(t-1)}, \boldsymbol{\mathcal{S}}_d^{(t)})$.

The griddy Gibbs sampler can be easily implemented for univariate full conditional PDFs, and its performance can be improved by using an adaptive grid and allowing the grid to grow if necessary (using the so called *grid grower*), as described in [173, 174]. However, the samples obtained are only approximately distributed according to the target, and building an effective approximation of the inverse CDF in the multi-variate case (e.g., for its use within the block Gibbs sampler) is a challenging task. The first issue can be addressed by using the *Gibbs stopper* [174], where an IS weight is assigned to the drawn samples in order to ensure that they come exactly from the target PDF, but the second one is much more difficult to solve.

In order to sample virtually from any full conditional PDF, the MH algorithm can be used within the Gibbs sampler. This results in a *hybrid sampler* [104], where an *internal* Monte Carlo method (the MH algorithm) is used within another *external* Monte Carlo technique (the Gibbs sampler). Apparently, Geweke and Tanizaki were the first ones to suggest using the MH algorithm within the Gibbs sampler in order to provide a general solution to nonlinear and/or non-Gaussian state space modeling in a Bayesian framework [175, 176]. The MH-within-Gibbs sampler is detailed in Algorithm 7. Note that T_{MH} iterations of the internal MH algorithm are performed per iteration of the external Gibbs sampler and only the last sample drawn from the MH algorithm is typically used for the integral approximation in Eq. (14). Furthermore, usually $T_{MH} = 1$ for the sake of efficiency, but several authors have shown that this is often not the best alternative from the point of view of reducing the variance of the desired estimators for a given computational budget [177]. Note also that the internal MH algorithm should be used to sample only those parameters that cannot be sampled directly (Algorithm 7 assumes that all the parameters require it), and that it can also be easily applied within the block and collapsed Gibbs samplers. Finally, note that Neal and Roberts have shown that the optimal scaling rate for the MH algorithm (which leads to an average acceptance rate of 0.234) also holds for the MH-within-Gibbs sampler regardless of the dimensionality of the update rule [178].

Noting that the piecewise proposal built by the griddy Gibbs sampler could be used to construct very good proposals for the MH-within-Gibbs sampler, Martino et al. recently proposed the fast universal self-tuned sampler (FUSS) within Gibbs algorithm [179]. Essentially, the idea is starting with a very dense grid that roughly covers the whole effective support of the corresponding full conditional PDF and then applying a pruning strategy in order to obtain a sparse grid that contains most of the probability mass of the conditional PDF. The steps performed by the FUSS algorithm, at the tth step of the Gibbs sampler for the dth parameter, are the following:

1. *Initialization:* Choose a large set of support points, $\widetilde{\boldsymbol{\mathcal{S}}}_d^{(t)} = \{\theta_{d,1}^{(t)}, \ldots, \theta_{d,L}^{(t)}\}$, that densely cover the whole effective support of the target.
2. *Pruning:* Remove support points according to a pre-specified and efficient criterion, attaining a final sparse set of support points, $\boldsymbol{\mathcal{S}}_d^{(t)} = \{\theta_{d,1}^{(t)}, \ldots, \theta_{d,K}^{(t)}\}$ with $K \ll L$.
3. *Construction:* Build a proposal function $q(\theta_d|\boldsymbol{\theta}_{\neg d}^{(t-1)}, \boldsymbol{\mathcal{S}}_d^{(t)})$ using some appropriate pre-defined mechanism, typically a PWC or PWL approach.
4. *MH steps:* Perform T_{MH} steps of the internal MH algorithm, as in Algorithm 7, using $q(\theta_d|\boldsymbol{\theta}_{\neg d}^{(t-1)}, \boldsymbol{\mathcal{S}}_d^{(t)})$ as the proposal PDF.

Since the FUSS algorithm builds a proposal tailored to the target, the acceptance rate of the internal MH algorithm is usually very high and the correlation among the drawn samples very small. This leads to estimators with a reduced variance, especially for very peaky proposals, where other Monte Carlo methods fail (see [179] for

Algorithm 7 MH-within-Gibbs algorithm.

1 **Initialization:** Choose a set of proposal PDFs, $\{\bar{q}(\theta_d|\theta_d^{(t-1)}, \boldsymbol{\theta}_{\neg d})\}_{d=1}^{D_\theta}$, an initial state $\boldsymbol{\theta}^{(0)}$, the total number of iterations (T), the number of iterations of the internal MH algorithm (T_{MH}), and the burn-in period (T_b).

2 FOR $t = 1, \ldots, T$:

(a) Select the coordinate to be sampled, $d_t \in \{1, \ldots, D_\theta\}$, and set $\widetilde{\theta}_{d_t}^{(0)} = \theta_{d_t}^{(t-1)}$.

(b) FOR $t' = 1, \ldots, T_{MH}$:

i. Draw $\theta'_{d_t} \sim \bar{q}(\theta_{d_t}|\widetilde{\theta}_{d_t}^{(t'-1)}, \boldsymbol{\theta}_{\neg d_t}^{(t-1)})$ and $u \sim \mathcal{U}([0,1))$.

ii. Compute the acceptance probability (α_t):

$$\alpha(\boldsymbol{\theta}', \boldsymbol{\theta}_{\neg d_t}^{(t-1)}, \widetilde{\theta}_{d_t}^{(t'-1)}) = \min\left[1, \frac{\pi(\theta'_{d_t}|\boldsymbol{\theta}_{\neg d_t}^{(t-1)}) q(\widetilde{\theta}_{d_t}^{(t'-1)}|\theta'_{d_t}, \boldsymbol{\theta}_{\neg d_t}^{(t-1)})}{\pi(\widetilde{\theta}_{d_t}^{(t'-1)}|\boldsymbol{\theta}_{\neg d_t}^{(t-1)}) q(\theta'_{d_t}|\widetilde{\theta}_{d_t}^{(t'-1)}, \boldsymbol{\theta}_{\neg d_t}^{(t-1)})}\right]$$

iii. If $u \leq \alpha_t$, accept θ'_{d_t} and set $\widetilde{\theta}_{d_t}^{(t')} = \theta'_{d_t}$. Otherwise (i.e., if $u > \alpha_t$), reject θ'_{d_t} and set $\widetilde{\theta}_{d_t}^{(t')} = \widetilde{\theta}_{d_t}^{(t'-1)}$.

(c) Set $\theta_{d_t}^{(t)} = \widetilde{\theta}_{d_t}^{(T_{MH})}$.

3 Approximate the integral in Eq. (2) using Eq. (14).

further details). Finally, note that it is again possible to employ all the T_{MH} samples generated by the internal MH algorithm in the final estimators, as shown in [168].

3.1.5 Other classical MCMC techniques

In this section, we describe other classical approaches for sampling from non-standard multi-variate densities: data augmentation, slice sampling, the hit-and-run algorithm, and adaptive direction sampling. We also discuss briefly the issue of thinning or subsampling the Markov chain, which is often used in signal processing applications to reduce the computational cost and the correlation among the generated samples.

Data augmentation (DA) The *data augmentation* method was originally devised by Tanner and Wong in order to compute posterior distributions for Bayesian inference [180]. The basic idea of data augmentation (DA) is the same one that underlies the well-known and widely used expectation-maximization (E-M) algorithm [181]: in many practical problems, augmenting the observed dataset ($\mathbf{y}$) with a set of latent data ($\mathbf{z}$) leads to an easier analysis of the problem. In the Bayesian inference case, the DA algorithm is based on the assumption that $\pi(\boldsymbol{\theta}|\mathbf{y},\mathbf{z})$ is straightforward to analyze, whereas $\pi(\boldsymbol{\theta}|\mathbf{y})$ is intractable. Another important assumption regards the generation of the latent data ($\mathbf{z}$): they should be easy to draw given the parameters and the observed data. Under these two assumptions, drawing samples from the desired target can be easily accomplished following the iterative approach shown in Algorithm 8. Note that the DA procedure shown in Algorithm 8 is equivalent to the application of the Gibbs sampler of Algorithm 4 on the augmented parameter vector $\boldsymbol{\theta}_a = [\boldsymbol{\theta}, \mathbf{z}_1, \ldots, \mathbf{z}_K]$ [170][18]. Note also that data augmentation is the opposite of integrating out parameters from a model in closed form, as done in the collapsed Gibbs sampler described in Section 3.1.3. Finally, let us remark that, just like it happens with the collapsed Gibbs sampler (cf. the previously mentioned discussion of Liu et al. in Section 3.1.3), DA can either increase or reduce the mixing efficiency.

Slice sampling Several Monte Carlo techniques, like direct methods (e.g., the inverse-of-density method) [119], the rejection sampler (see Section 2.2), and some MCMC algorithms (e.g., the so-called *slice sampler*) rely on a simple result, known as *the fundamental theorem of simulation*.

[18] In fact, Algorithm 8 corresponds to the block Gibbs sampler with $K+1$ groups: $\boldsymbol{\vartheta}_1 = \boldsymbol{\theta}$, $\boldsymbol{\vartheta}_2 = \mathbf{z}_1, \ldots, \boldsymbol{\vartheta}_{K+1} = \mathbf{z}_K$. However, it can be easily converted into a component-wise Gibbs algorithm (with $KD_{\mathbf{y}} + D_{\theta}$ components) by decomposing steps 2(a) and 2(b) into D_{θ} and $KD_{\mathbf{y}}$ draws from the corresponding univariate full conditional PDFs, respectively.

Algorithm 8 Basic Data Augmentation (DA) algorithm.

1 **Initialization:** Select the number of latent data generated per iteration (K), the total number of iterations (T) and the burn-in period (T_b). Obtain an initial set of latent data $(\mathbf{z}_1^{(0)}, \ldots, \mathbf{z}_K^{(0)})$ and construct an initial approximation of the target, $\hat{\pi}^{(0)}(\boldsymbol{\theta}|\mathbf{y}, \mathbf{z}_1^{(0)}, \ldots, \mathbf{z}_K^{(0)}) = \frac{1}{K}\sum_{k=1}^{K} \pi(\boldsymbol{\theta}|\mathbf{y}, \mathbf{z}_k^{(0)})$.

2 FOR $t = 1, \ldots, T$:

(a) Draw $\boldsymbol{\theta}^{(t)} \sim \hat{\pi}^{(t-1)}(\boldsymbol{\theta}|\mathbf{y}, \mathbf{z}_1^{(0)}, \ldots, \mathbf{z}_K^{(0)})$.

(b) Draw $\mathbf{z}_k^{(t)} \sim p(\mathbf{z}|\boldsymbol{\theta}^{(t)}, \mathbf{y})$ for $k = 1, \ldots, K$.

(c) Update the approximation of the target:

$$\hat{\pi}^{(t)}(\boldsymbol{\theta}|\mathbf{y}, \mathbf{z}_1^{(0)}, \ldots, \mathbf{z}_K^{(0)}) = \frac{1}{K}\sum_{k=1}^{K} \pi(\boldsymbol{\theta}|\mathbf{y}, \mathbf{z}_k^{(t)}). \tag{23}$$

3 Approximate the integral in Eq. (2) using Eq. (14).

Theorem 1 *Drawing samples from a random variable* $\boldsymbol{\theta}$ *with density* $\bar{\pi}(\boldsymbol{\theta}) \propto \pi(\boldsymbol{\theta})$ *is equivalent to sampling uniformly on the region defined by*

$$\mathcal{A}_{\pi} = \{(\boldsymbol{\theta}, z) \in \mathbb{R}^2 : 0 \leq z \leq \pi(\boldsymbol{\theta})\}. \tag{24}$$

Namely, considering a realization $(\boldsymbol{\theta}', z')$*, if it is distributed uniformly on* $\mathcal{A}_{\pi}$*, then* $\boldsymbol{\theta}'$ *is a sample from* $\bar{\pi}(\boldsymbol{\theta})$ *[104, 119].*

Therefore, if we are able to draw a vector $(\boldsymbol{\theta}', z')$ uniformly on $\mathcal{A}_{\pi}$ (i.e., the area below the unnormalized target function $\pi(\boldsymbol{\theta})$), then the coordinate $\boldsymbol{\theta}'$ is marginally distributed according to $\bar{\pi}(\boldsymbol{\theta})$. The variable z plays the role of an auxiliary variable which is introduced in order to ease the sampling procedure, just like the latent data in the data augmentation algorithm.

The slice sampler is precisely a Gibbs sampling method that can be applied for drawing samples uniformly from $\mathcal{A}_{\pi}$. Let us define the set

$$\mathcal{O}(z) = \{\text{all } \boldsymbol{\theta} \in \mathbb{R}^{\mathcal{D}_{\theta}} \text{ such that } \pi(\boldsymbol{\theta}) \geq z\}. \tag{25}$$

The slice sampler is given in Algorithm 9.

Algorithm 9 The slice sampler.

1 **Initialization:** Choose an initial state $\boldsymbol{\theta}^{(0)}$, the total number of iterations (T), and the burn-in period (T_b).

2 FOR $t = 1, \ldots, T$:

(a) Draw $z^{(t)}$ uniformly in the interval $[0, \pi(\boldsymbol{\theta}^{(t-1)})]$,

(b) Draw $\boldsymbol{\theta}^{(t)}$ uniformly in the set $\mathcal{O}(z^{(t)})$.

3 Approximate the integral in Eq. (2) using Eq. (14).

The slice sampling algorithm generates a Markov chain over $\mathcal{A}_\pi$, producing samples uniformly distributed in $\mathcal{A}_\pi$ after the burn-in period. However, performing step 2b is often virtually impossible (even for unidimensional target PDFs), since it requires the inversion of $\pi(\boldsymbol{\theta})$ in order to determine the set $\mathcal{O}(z)$. The difficulty of this inversion is due to the fact that $\pi(\boldsymbol{\theta})$ is usually a non-monotonic function, implying that the set $\mathcal{O}(z)$ is typically formed by the union of disjoint sets which are difficult to determine. Fortunately, several practical procedures have been suggested for this purpose. See [182] for further information on this issue.

Hit-and-run Another important class of methods that can be used both for global optimization and Bayesian inference are the so called *hit-and-run algorithms*, which are a collection of efficient sampling techniques that use random walks to explore the parameter space. Sampling through random walks was independently proposed by Boneh and Golan [183] and Smith [184, 185], and this class of methods were later renamed as hit-and-run algorithms [186]. The generic hit-and-run algorithm is shown in Algorithm 10. The basic idea is determining a random direction in the D_θ-dimensional parameter space using the proposal $q(\boldsymbol{\theta})$ and then selecting a random point along that direction with a probability proportional to the target PDF evaluated along the chosen direction.

Algorithm 10 Hit-and-run algorithm.

1. **Initialization:** Choose a proposal function over the unit hypersphere, $q(\boldsymbol{\theta}) \neq 0 \iff \boldsymbol{\theta} \in \mathcal{B} = \{\boldsymbol{\theta} \in \Theta \subseteq \mathbb{R}^{D_\theta} : \|\boldsymbol{\theta}\| = 1\}$, the initial state, $\boldsymbol{\theta}^{(0)}$, the total number of iterations (T) and the burn-in period (T_b).
2. FOR $t = 1, \ldots, T$:
   (a) Draw $\boldsymbol{\theta'} \sim \bar{q}(\boldsymbol{\theta})$.
   (b) Find the set $\Lambda_t = \{\lambda \in \mathbb{R} : \boldsymbol{\theta}^{(t-1)} + \lambda\boldsymbol{\theta'} \in \Theta \subseteq \mathbb{R}^{D_\theta}\}$.
   (c) Draw $\lambda_t \sim p^{(t)}(\lambda) \propto \pi(\boldsymbol{\theta}^{(t-1)} + \lambda\boldsymbol{\theta'})$ with $\lambda \in \Lambda_t$.
   (d) Set $\boldsymbol{\theta}^{(t)} = \boldsymbol{\theta}^{(t-1)} + \lambda_t\boldsymbol{\theta'}$.
3. Approximate the integral in Eq. (2) using Eq. (14).

Different hit-and-run algorithms are obtained depending on the proposal function $q(\boldsymbol{\theta})$. For instance, the original *hypersphere directions* (HD) hit-and-run algorithm considered a uniform proposal $q(\boldsymbol{\theta})$ and a uniform target on some bounded region $\Theta \subset \mathbb{R}^{D_\theta}$ [183–185], whereas the *coordinate directions* (CD) hit-and-run randomly chooses one of the D_θ coordinates of the parameter space [187]. Regarding the connections with other methods, the hit-and-run algorithm has similarities both with the MH algorithm and the Gibbs sampler. On the one hand, the hit-and-run algorithm resembles the random walk MH algorithm, but the generated samples are always accepted, since they are drawn from the target[19]. On the other hand, the CD hit-and-run algorithm is equivalent to the random scan Gibbs sampler. However, note that the generic hit-and-run algorithm is more flexible than the Gibbs sampler, since it can choose any arbitrary direction, not only one of the directions corresponding to the different parameters.

Adaptive direction sampling (ADS) A third important family of methods that attempt to improve the convergence speed of the Gibbs sampler is *adaptive direction sampling* (ADS) [189, 190]. The basic idea of ADS is maintaining a set of support points that are constantly updated, with the current support set being used to determine the sampling direction. The general adaptive direction sampler is shown in Algorithm 11.

Algorithm 11 Adaptive direction sampling (ADS).

1. **Initialization:** Choose an initial support set, $\mathcal{S}^{(0)} = \{\boldsymbol{\theta}_1^{(0)}, \ldots, \boldsymbol{\theta}_K^{(0)}\}$ with $K > D_\theta$, the total number of iterations (T), and the burn-in period (T_b).
2. FOR $t = 1, \ldots, T$:
   (a) Draw $i \sim \mathcal{U}(\{1, \ldots, K\})$ and set $\boldsymbol{\theta}_c^{(t)} = \boldsymbol{\theta}_i^{(t-1)}$.
   (b) Select a second vector $\vartheta^{(t)}$ according to some pre-specified scheme, depending on the specific ADS algorithm to be implemented (see below).
   (c) Draw $r' \sim p(r) \propto \pi(\boldsymbol{\theta}_c^{(t)} + r(\vartheta^{(t)} + \lambda_t\boldsymbol{\theta}_c^{(t)})) \times |1 + r\lambda_t|^{D_\theta - 1}$.
   (d) Set $\boldsymbol{\theta}_i^{(t)} = \boldsymbol{\theta}_c^{(t)} + r'(\vartheta^{(t)} + \lambda_t\boldsymbol{\theta}_c^{(t)})$.
3. Approximate the integral in Eq. (2) using Eq. (14).

The procedure shown in Algorithm 11 is very general and many different algorithms can be obtained by considering different choices for $\vartheta^{(t)}$ and λ_t [189]:

- *Snooker algorithm:* Important special case of the general ADS algorithm obtained by setting $\vartheta^{(t)} = \boldsymbol{\theta}_a^{(t)} \sim \mathcal{U}(\mathcal{S}^{(t-1)} \setminus \{\boldsymbol{\theta}_c^{(t)}\})$ and $\lambda_t = -1$. In this specific algorithm, $\boldsymbol{\theta}_a^{(t)}$ sets the direction along which $\boldsymbol{\theta}_c^{(t)}$ is moved in order to obtain the new sample.
- *Parallel ADS:* Obtained by setting $\vartheta^{(t)} = \boldsymbol{\theta}_a^{(t)} - \boldsymbol{\theta}_b^{(t)}$, with $\boldsymbol{\theta}_a^{(t)} \sim \mathcal{U}(\mathcal{S}^{(t-1)} \setminus \{\boldsymbol{\theta}_c^{(t)}\})$ and

[19] Note that drawing samples directly from the target for an arbitrary direction of the parameter space may be unfeasible in many problems. In these cases, the Metropolised hit-and-run sampler proposed in [188] can be used.

$\boldsymbol{\theta}_b^{(t)} \sim \boldsymbol{\mathcal{U}}(\boldsymbol{\mathcal{S}}^{(t-1)} \setminus \{\boldsymbol{\theta}_a^{(t)}, \boldsymbol{\theta}_c^{(t)}\})$, and $\lambda_t = 0$. In this case, the direction for the movement of $\boldsymbol{\theta}_c^{(t)}$ is set by the two auxiliary points drawn from $\boldsymbol{\mathcal{S}}^{(t-1)}$, $\boldsymbol{\theta}_a^{(t)}$ and $\boldsymbol{\theta}_b^{(t)}$.

- *Hit-and-run:* Obtained as a particular case of the general ADS algorithm by setting $\lambda_t = 0$ and $\boldsymbol{\vartheta}^{(t)}$ to some random direction uniformly selected in the parameter space.
- *Random scan Gibbs sampler:* Obtained by setting $\lambda_t = 0$ and $\boldsymbol{\vartheta}^{(t)}$ to be some randomly chosen parameter out of the D_θ available, i.e., $\boldsymbol{\vartheta}^{(t)} = [0, \ldots, 0, \theta_d^{(t)}, 0, \ldots, 0])$.

Subsampling or thinning of Markov chains Finally, let us briefly discuss the issue of thinning or subsampling a Markov chain, which is often used to reduce the correlation among the generated samples of MCMC methods and also serves to decrease the computational/storage burden. Thinning consists of discarding $K-1$ out of every K outputs of the obtained Markov chain (i.e., downsampling or decimating by a factor K in signal processing terminology), thus resulting in the following estimator:

$$\widehat{I}_{M_{\text{thin}}} = \frac{1}{M_{\text{thin}}} \sum_{m=0}^{M_{\text{thin}}-1} g(\boldsymbol{\theta}^{(T_b+1+mK)}), \tag{26}$$

with $M_{\text{thin}} = \left\lfloor \frac{T-(T_b+1)}{K} \right\rfloor$ and $\lfloor \cdot \rfloor$ denoting the integer part approximated from below. It is well-known that the estimator in Eq. (26) has a larger variance than the estimator of Eq. (14), see [191] for a formal proof for reversible Markov chains or [192] for a simpler justification which does not rely on reversibility. Hence, many authors have warned practitioners against subsampling, which should be used only when strictly required due to computation/storage constraints [193]. However, Owen has shown very recently that thinning can actually be advantageous in some cases [194]. Assuming that it costs one unit of time to advance a Markov chain and $\Delta t > 0$ units of time to compute a sampled quantity of interest, he shows that thinning will improve the statistical efficiency (as quantified by the variance of the resulting estimator) when Δt is large and the autocorrelations decay slowly enough. Hence, even when practical restrictions do not apply, signal processing practitioners should check the correlation structure of their problems in order to determine whether thinning can be advantageous or not, and which is the optimal thinning factor (see [194] for further details).

3.2 Adaptive MCMC

The MH algorithm produces samples distributed according to any desired target distribution after the burn-in period by using an arbitrary proposal density that fulfills some mild regularity conditions. However, the choice of the proposal PDF is crucial for the practical operation of the algorithm. In order to sample from a given target distribution efficiently, we must carefully choose the proposal distribution in such a way that we obtain independent-enough samples with a high-enough rate so that we achieve a good approximation of the distribution in a reasonable time (e.g., in minutes or hours instead of hundreds of years). The manual tuning of the proposal distribution can be a tedious task. For that reason, researchers have developed adaptive MCMC methods (e.g., see [195]), which aim at tuning the algorithm's performance automatically, typically by adapting the proposal distribution based on the already seen samples. In the following, we review some of the most relevant adaptive MCMC approaches.

3.2.1 *Parametric approaches*

The history of adaptive MCMC methods can probably be considered to start with the adaptive Metropolis (AM) algorithm [196], where the idea is to adapt the covariance of a random-walk Metropolis algorithm using previous samples drawn by the same algorithm. Let us recall that the random-walk MH algorithm typically uses a Gaussian proposal PDF of the form

$$q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t-1)}) = \boldsymbol{\mathcal{N}}(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t-1)}, \mathbf{C}), \tag{27}$$

where $\mathbf{C}$ is some suitable covariance matrix. The tuning of this algorithm is then reduced to the selection of the covariance matrix $\mathbf{C}$.

One way to approach the tuning problem is to consider an idealized case where we actually know the true covariance $\boldsymbol{\Sigma}$ of the target distribution. It turns out that the optimal covariance matrix is $\mathbf{C}^* = \lambda\boldsymbol{\Sigma}$ under certain idealized conditions. Furthermore, in the Gaussian case, we can compute the optimal $\lambda^* = 2.38^2/D_\theta$ [197]. This result can now be used to adapt the proposal's covariance by replacing the target distribution covariance with its empirical counterpart.

The adaptive Metropolis (AM) [196] uses an adaptation rule where the covariance of the target distribution is estimated via

$$\boldsymbol{\Sigma}_t = \text{Cov}[\boldsymbol{\theta}^{(0)}, \ldots, \boldsymbol{\theta}^{(t-1)}, \boldsymbol{\theta}^{(t)}] + \epsilon\mathbf{I}, \tag{28}$$

where ϵ is a small positive constant which is used to ensure that $\boldsymbol{\Sigma}_t$ is not ill-conditioned. In practice, this amounts to adding an extra step to the MH Algorithm 3, just after the acceptance step, to update the current estimate of the covariance. Furthermore, the adaptation rule can be easily implemented recursively [198]. Numerous modifications and improvements to this rule have been proposed. For example, Gaussian mixture approximations are considered in [199, 200], the combination with early rejection is proposed in [198], and adaptive Kalman filter based covariance estimation is considered in [201].

Because the identity for the optimal λ^* only applies to Gaussian target PDFs, it is often also desirable to adapt the λ-parameter as well [195, 202–204]. In this case, the optimization criterion consists typically in trying to reach "an optimal" acceptance rate of $\alpha^* = 0.234$. Note that this optimal acceptance rate also corresponds to certain idealized conditions, but still provides a good rule of thumb. A typical rule for the adaptation then has the form

$$\log \lambda_t = \log \lambda_{t-1} + \gamma_t \, (\alpha_t - \bar{\alpha}), \tag{29}$$

where γ_t is a suitable gain sequence, α_t is the acceptance probability at the current step, and $\bar{\alpha}$ is the target acceptance rate (e.g., $\bar{\alpha} = \alpha^*$).

The general AM algorithm, including both covariance and acceptance rate adaptation, is shown in Algorithm 12. Comparing Algorithms 3 and 12, we note that the difference simply lies in the introduction of two additional steps (steps 2(d) and 2(e)), where the empirical covariance matrix $\boldsymbol{\Sigma}_t$ and the scale factor λ_t are computed. However, these two simple steps can lead to a substantially improved proposal w.r.t. the initial one and thus to a much better performance of the resulting MH algorithm.

Algorithm 12 General Adaptive Metropolis (AM) Algorithm.

1. **Initialization:** Choose an initial covariance $\boldsymbol{\Sigma}_0$, an initial adaptation parameter λ_0, a target acceptance rate $\bar{\alpha}$, an initial state $\boldsymbol{\theta}^{(0)}$, the total number of iterations (T) and the burn-in period (T_b).
2. FOR $t = 1, \ldots, T$:
   (a) Set $\mathbf{C}_t = \lambda_{t-1} \boldsymbol{\Sigma}_{t-1}$. Draw $\boldsymbol{\theta}' \sim \boldsymbol{\mathcal{N}}(\boldsymbol{\theta}^{(t-1)}, \mathbf{C}_t)$ and $u \sim \boldsymbol{\mathcal{U}}([0, 1))$.
   (b) Compute the acceptance probability:
   $$\alpha_t \equiv \alpha(\boldsymbol{\theta}', \boldsymbol{\theta}^{(t-1)}) = \min\left[1, \frac{\pi(\boldsymbol{\theta}')}{\pi(\boldsymbol{\theta}^{(t-1)})}\right]. \tag{30}$$
   (c) If $u \leq \alpha_t$, accept $\boldsymbol{\theta}'$ and set $\boldsymbol{\theta}^{(t)} = \boldsymbol{\theta}'$. Otherwise (i.e., if $u > \alpha_t$), reject $\boldsymbol{\theta}'$ and set $\boldsymbol{\theta}^{(t)} = \boldsymbol{\theta}^{(t-1)}$.
   (d) Obtain a new estimate of the target's covariance $\boldsymbol{\Sigma}_t$, e.g., by using the estimator of Eq. (28).
   (e) Compute a new λ_t, e.g., by applying the rule of Eq. (29).
3. Approximate the integral in Eq. (2) using Eq. (14).

3.2.2 Non-parametric approaches

Designing parametric adaptive MCMC approaches that attain a good performance directly in high-dimensional parameter spaces can be a very challenging task. For this reason, some authors have concentrated on developing very efficient non-parametric adaptive MH algorithms in low-dimensional parameter spaces (typically one-dimensional spaces). These adaptive schemes can then be used within the Gibbs sampler (see Section 3.1.3) in order to perform estimations in higher dimensional spaces. In this section, we review the most widely known approach, Adaptive Rejection Metropolis Sampling (ARMS), as well as some very recent extensions.

Adaptive rejection Metropolis sampling (ARMS) The adaptive rejection Metropolis sampling (ARMS) technique combines the ARS method and the MH algorithm in order to sample virtually from any univariate PDF [205]. ARMS is summarized in Algorithm 13 for a one-dimensional parameter θ. For multi-dimensional parameter spaces, ARMS can simply be embedded within a Gibbs sampler in a similar way as done in the MH-within-Gibbs algorithm (see Algorithm 7). Essentially, ARMS performs first an RS test, and the rejected samples are used to improve the proposal PDF with the aim of constructing a proposal that becomes closer and closer to the target. Then, the accepted samples from the RS test go through an MH test, where they can still be rejected. This MH step removes the main limitation of rejection sampling approaches: requiring that $Cq_t(\theta) \geq \pi(\theta)$ for some constant C and all the possible values of $\theta \in \Theta$. This allows ARMS to generate samples from a wide variety of target densities, becoming virtually a universal sampler from a theoretical point of view.

The mechanism used to construct the proposal is critical for the good performance of ARMS [205]. Let us consider the set of support points at the mth iteration of the algorithm, $\boldsymbol{\mathcal{S}}^{(m)} = \{\theta_1^{(m)}, \theta_2^{(m)}, \ldots, \theta_{K_m}^{(m)}\}$, and define the intervals $\boldsymbol{\mathcal{I}}_0^{(m)} = (-\infty, \theta_1^{(m)}]$, $\boldsymbol{\mathcal{I}}_j^{(m)} = (\theta_j^{(m)}, \theta_{j+1}^{(m)}]$ for $j = 1, ..., K_m - 1$, and $\boldsymbol{\mathcal{I}}_{K_m}^{(m)} = (\theta_{K_m}^{(m)}, +\infty)$. Moreover, let us denote as $L_{j,j+1}^{(m)}(\theta)$ the line passing through $(\theta_j^{(m)}, V(\theta_j^{(m)}))$ and $(\theta_{j+1}^{(m)}, V(\theta_{j+1}^{(m)}))$, with $V = \log(\pi(\theta))$ and $j = 1, ..., K_m - 1$. Then, a PWL potential function $W_m(x)$ is constructed in ARMS as

$$W_m(\theta) = \begin{cases} L_{1,2}^{(m)}(\theta), & \theta \in \boldsymbol{\mathcal{I}}_0^{(m)}, \\ \max\left\{L_{1,2}^{(m)}(\theta), L_{2,3}^{(m)}(\theta)\right\}, & \theta \in \boldsymbol{\mathcal{I}}_1^{(m)}, \\ \varphi_j^{(m)}(\theta), & \theta \in \boldsymbol{\mathcal{I}}_j, \\ \max\left\{L_{K_m-2,K_m-1}^{(m)}(\theta), L_{K_m-1,K_m}^{(m)}(\theta)\right\}, & \theta \in \boldsymbol{\mathcal{I}}_{K_m-1}^{(m)}, \\ L_{K_m-1,K_m}^{(m)}(\theta), & \theta \in \boldsymbol{\mathcal{I}}_{K_m}^{(m)}, \end{cases} \tag{32}$$

where

$$\varphi_j^{(m)}(\theta) = \max\left\{L_{j,j+1}^{(m)}(\theta), \min\left\{L_{j-1,j}^{(m)}(\theta), L_{j+1,j+2}^{(m)}(\theta)\right\}\right\},$$

and $j = 2, \ldots, K_m - 1$. Hence, the proposal PDF, $q_m(\theta) \propto \exp(W_m(\theta))$, is formed by exponential pieces. However,

Algorithm 13 Adaptive Rejection Metropolis Sampling (ARMS).

1. **Initialization:** Set $t = 0$ (chain's iteration) and $m = 0$ (algorithm's iteration). Choose an initial state ($\boldsymbol{\theta}^{(0)}$), an initial number of support points (K_0), an initial support set $\boldsymbol{\mathcal{S}}^{(0)} = \{\theta_1^{(0)}, \ldots, \theta_{K_0}^{(0)}\}$, the total number of iterations (T) and the burn-in period (T_b).
2. `WHILE` $t \leq T$:
   (a) Build a proposal function, $q_m(\theta)$, given a set of support points $\boldsymbol{\mathcal{S}}^{(m)} = \{\theta_1^{(m)}, \theta_2^{(m)}, \ldots, \theta_{K_m}^{(m)}\}$, according to Eq. (32).
   (b) Draw $\theta' \sim \bar{q}_m(\theta)$ and $u \sim \boldsymbol{\mathcal{U}}([0, 1))$.
   (c) If $u > \frac{\pi(\theta')}{q_m(\theta')}$, then reject θ', update $\boldsymbol{\mathcal{S}}^{(m+1)} = \boldsymbol{\mathcal{S}}^{(m)} \cup \{\theta'\}$, $K_{m+1} = K_m + 1$, and set $m = m + 1$. Go back to step 2(a).
   (d) Otherwise (i.e., if $u \leq \pi(\theta')/q_m(\theta')$), draw $u' \sim \boldsymbol{\mathcal{U}}([0, 1))$ and compute the acceptance probability:

$$\alpha_t \equiv \alpha(\boldsymbol{\theta'}, \boldsymbol{\theta}^{(t-1)}) = \min\left[1, \frac{\pi(\boldsymbol{\theta'}) \min\{\pi(\boldsymbol{\theta}^{(t-1)}), q_m(\boldsymbol{\theta}^{(t-1)})\}}{\pi(\boldsymbol{\theta}^{(t-1)}) \min\{\pi(\boldsymbol{\theta'}), q_m(\boldsymbol{\theta'})\}}\right]. \tag{31}$$

   (e) If $u' \leq \alpha_t$, accept θ' and set $\theta^{(t)} = \theta'$. Otherwise (i.e., if $u' > \alpha_t$), reject θ' and set $\theta^{(t)} = \theta^{(t-1)}$.
   (f) Set $\boldsymbol{\mathcal{S}}^{(m+1)} = \boldsymbol{\mathcal{S}}^{(m)}$, $K_{m+1} = K_m$, $m = m + 1$, $t = t + 1$ and return to step 2(a).
3. Approximate the integral in Eq. (2) using Eq. (14).

note that the number of linear pieces that form the proposal with this construction is larger than K_m in general, since the proposal can be formed by two segments rather than one in some intervals. Hence, the computation of intersection points among these two segments is also required to implement the algorithm. More sophisticated approaches to build $W_m(\theta)$ (e.g., using quadratic segments when possible [206]) have been proposed, but none of them solves the structural problem of ARMS that is briefly described next.

Independent doubly adaptive rejection Metropolis sampling (IA^{2}RMS) Unfortunately, ARMS cannot always guarantee the convergence of the sequence of proposals to the target, since the proposal PDF is only updated when a sample θ' is rejected by the RS test, something that can only happen when $q_m(\theta') > \pi(\theta')$. When a sample is initially accepted by the RS test, as it always happens when $q_m(\theta') > \pi(\theta')$, the proposal is never updated. Thus, the satisfactory performance of ARMS depends on two issues:

1 $W_m(\theta)$ should be constructed in such a way that $W_m(\theta) \geq V(\theta)$ almost everywhere (a.e.), i.e., $q_m(\theta) \geq \pi(\theta)$ a.e., so that support points can be added a.e.
2 The addition of a support point inside an interval must entail a change of the proposal PDF inside other neighbor intervals when building $W_{m+1}(\theta)$. This allows the proposal to improve inside regions where $q_m(\theta) < \pi(\theta)$.

These two conditions can be relaxed by allowing support points to be added inside regions where $q_m(\theta) < \pi(\theta)$ in a controlled way. The independent doubly adaptive rejection Metropolis sampling (IA^{2}RMS) algorithm achieves this task by introducing a second control test that allows rejected samples from the MH stage to be incorporated to the support set with a certain non-null probability [177, 207]. The IA^{2}RMS algorithm is shown in Algorithm 14. Note that the only difference w.r.t. ARMS lies in step 2(f). However, this crucial step allows samples to be added to the support set everywhere regardless of whether the proposal is below or above the target, and guarantees that $q_m(\theta) \to \pi(\theta)$ as $t \to \infty$. Moreover, this allows IA^{2}RMS to effectively decouple the proposal construction from the algorithm's evolution, thus allowing the use of simpler proposals than the one used in ARMS (see [177] for further details).

Finally, let us remark that the mechanism used to accept/reject samples and to build the support set can be generalized further, as shown in [208], where the adaptive procedure is also extended to the framework of multiple-try Metropolis (MTM) algorithms (see Section 5.1.1).

3.2.3 *Convergence of adaptive MCMC algorithms*

Note that, since the adaptation can depend on all past samples in adaptive MCMC algorithms, the Markovian nature of classical MCMC methods is lost. Therefore, ensuring the convergence of these techniques is much more complicated, as it cannot rely on standard tools and usually it has to be analyzed on a case by case basis depending on the method. Furthermore, note that adaptive MCMC methods do not always converge, even if the adaptation itself converges. A cautionary example about this issue is provided by Roberts and Rosenthal in [209]. For these reasons, it is advisable either to follow a finite adaptation policy (i.e., adapting only during a finite number of initial iterations) or to adapt increasingly less often[20]. Indeed, Chimisov et al. [210] have recently analyzed this second scenario, showing that a central limit theorem can be proven for such chains.

[20] In some methods, the adaptation rate is automatically controlled by the algorithm. For instance, this happens in the parametric methods (ARMS, IA^{2}RMS and its variants) described in Section 3.2.2: as new samples are incorporated to the support set, the probability of adding new samples to this set decreases and so the adaptation is performed more and more rarely.

Algorithm 14 Independent Doubly Adaptive Rejection Metropolis Sampling (IA^{2}RMS).

1 **Initialization:** Set $t = 0$ (chain's iteration) and $m = 0$ (algorithm's iteration). Choose an initial state ($\boldsymbol{\theta}^{(0)}$), an initial number of support points (K_0), an initial support set $\boldsymbol{\mathcal{S}}^{(0)} = \{\theta_1^{(0)}, \ldots, \theta_{K_0}^{(0)}\}$, the total number of iterations (T) and the burn-in period (T_b).

2 `WHILE` $t \leq T$:

(a) Build a proposal function, $q_m(\theta)$, given a set of support points $\boldsymbol{\mathcal{S}}^{(m)} = \{\theta_1^{(m)}, \theta_2^{(m)}, \ldots, \theta_{K_m}^{(m)}\}$, using some simple non-parametric approach (see [177] for several possibilities).

(b) Draw $\theta' \sim \bar{q}_m(\theta)$ and $u \sim \boldsymbol{\mathcal{U}}([0,1))$.

(c) If $u > \frac{\pi(\theta')}{q_m(\theta')}$, then reject θ', update $\boldsymbol{\mathcal{S}}^{(m+1)} = \boldsymbol{\mathcal{S}}^{(m)} \cup \{\theta'\}$, $K_{m+1} = K_m + 1$, and set $m = m + 1$. Go back to step 2(a).

(d) Otherwise (i.e., if $u \leq \pi(\theta')/q_m(\theta')$), draw $u' \sim \boldsymbol{\mathcal{U}}([0,1))$ and compute the acceptance probability:

$$\alpha_t \equiv \alpha(\boldsymbol{\theta'}, \boldsymbol{\theta}^{(t-1)}) = \min\left[1, \frac{\pi(\boldsymbol{\theta'}) \min\{\pi(\boldsymbol{\theta}^{(t-1)}), q_m(\boldsymbol{\theta}^{(t-1)})\}}{\pi(\boldsymbol{\theta}^{(t-1)}) \min\{\pi(\boldsymbol{\theta'}), q_m(\boldsymbol{\theta'})\}}\right]. \tag{33}$$

(e) If $u' \leq \alpha_t$, accept θ', setting $\theta^{(t)} = \theta'$ and $\vartheta = \theta^{(t-1)}$. Otherwise (i.e., if $u' > \alpha_t$), reject θ', setting $\theta^{(t)} = \theta^{(t-1)}$ and $\vartheta = \theta'$.

(f) Draw $u'' \sim \boldsymbol{\mathcal{U}}([0,1))$. If $u'' > q_m(\vartheta)/\pi(\vartheta)$, set $\boldsymbol{\mathcal{S}}^{(m+1)} = \boldsymbol{\mathcal{S}}^{(m)} \cup \{\vartheta\}$ and $K_{m+1} = K_m + 1$. Otherwise (i.e., if $u'' \leq q_m(\vartheta)/\pi(\vartheta)$), set $\boldsymbol{\mathcal{S}}^{(m+1)} = \boldsymbol{\mathcal{S}}^{(m)}$ and $K_{m+1} = K_m$.

(g) Set $m = m+1$, $t = t+1$ and return to step 2(a).

3 Approximate the integral in Eq. (2) using Eq. (14).

3.3 Gradient-based techniques

In this section, we consider MCMC methods which use the gradient of the log-posterior, $\nabla \log \pi(\boldsymbol{\theta})$, to enhance the efficiency of the sampling procedure. The intuition is that, by using the gradient, we can form proposal distributions that allow for longer jumps without pushing the acceptance ratio of the method too low.

3.3.1 *Metropolis adjusted Langevin algorithm*

The Metropolis adjusted Langevin algorithm (MALA) is an MH algorithm [211, 212] which uses a stochastic differential equation (SDE) to form the proposal distribution. Let us consider the following Langevin diffusion type of SDE:

$$d\boldsymbol{\theta}(\tau) = \mathbf{f}(\boldsymbol{\theta}(\tau))\, d\tau + d\mathbf{b}(\tau), \tag{34}$$

where $\mathbf{b}(\tau)$ is a D_θ-dimensional Brownian motion. The Fokker–Planck equation giving the probability density $p(\boldsymbol{\theta}, \tau)$ of the diffusion state is

$$\frac{\partial p(\boldsymbol{\theta}, \tau)}{\partial \tau} = -\nabla \cdot [\mathbf{f}(\boldsymbol{\theta})\, p(\boldsymbol{\theta}, \tau)] + \frac{1}{2}\nabla^2 p(\boldsymbol{\theta}, \tau). \tag{35}$$

If we now select the drift of the diffusion as

$$\mathbf{f}(\boldsymbol{\theta}) = \frac{1}{2}\nabla \log \pi(\boldsymbol{\theta}), \tag{36}$$

then the stationary solution $\partial p(\boldsymbol{\theta}, \tau)/\partial \tau = 0$ is given by

$$p(\boldsymbol{\theta}, \tau) = \bar{\pi}(\boldsymbol{\theta}). \tag{37}$$

By starting from a value $\boldsymbol{\theta}^{(0)} \sim \bar{\pi}(\boldsymbol{\theta})$ and solving the SDE for $\tau > 0$ we can "generate" more samples from $\bar{\pi}(\boldsymbol{\theta})$, because the marginal distribution of the SDE solution $\boldsymbol{\theta}^{(t)}$ is $\bar{\pi}(\boldsymbol{\theta})$ for all $t \geq 0$.

MALA uses an SDE of the kind described above as the proposal distribution in an MH algorithm. Unfortunately, we cannot solve or simulate the SDE exactly. Hence, we typically approximate its solution using the Euler–Maruama method [213]:

$$\boldsymbol{\theta}^{(\tau_{n+1})} \approx \boldsymbol{\theta}^{(\tau_n)} + \frac{\Delta\tau}{2}\nabla \log \pi(\boldsymbol{\theta}^{(\tau_n)}) + \sqrt{\Delta\tau}\, \mathbf{z}_n, \tag{38}$$

where $\mathbf{z}_n \sim \boldsymbol{\mathcal{N}}(0, \mathbf{I})$ and $\Delta\tau = \tau_{n+1} - \tau_n$. Nevertheless, let us remark that it would also be possible to use other numerical solution methods for SDEs in MALA as well [213]. Algorithm 15 summarizes the resulting MALA algorithm with one step of the Euler–Maruyama method for the numerical integration.

Algorithm 15 Metropolis adjusted Langevin algorithm (MALA).

1 **Initialization:** Choose an initial state $\boldsymbol{\theta}^{(0)}$, the discretization step $\Delta\tau$, the total number of iterations (T), and the burn-in period (T_b).

2 `FOR` $t = 1, \ldots, T$:

(a) Draw $\mathbf{z} \sim \boldsymbol{\mathcal{N}}(0, \mathbf{I})$, $u \sim \boldsymbol{\mathcal{U}}([0,1))$ and simulate a new sample from the Langevin diffusion:

$$\boldsymbol{\theta'} = \boldsymbol{\theta}^{(t-1)} + \frac{\Delta\tau}{2}\nabla \log \pi(\boldsymbol{\theta}^{(t-1)}) + \sqrt{\Delta\tau}\, \mathbf{z}, \tag{39}$$

(b) Compute the acceptance probability (α_t):

$$\alpha(\boldsymbol{\theta'}, \boldsymbol{\theta}^{(t-1)}) = \min\left[1, \frac{\pi(\boldsymbol{\theta'})\boldsymbol{\mathcal{N}}(\boldsymbol{\theta}^{(t-1)} | \boldsymbol{\theta'} + \frac{\Delta\tau}{2}\nabla \log \pi(\boldsymbol{\theta'}), \Delta\tau \mathbf{I})}{\pi(\boldsymbol{\theta}^{(t-1)})\boldsymbol{\mathcal{N}}(\boldsymbol{\theta'} | \boldsymbol{\theta}^{(t-1)} + \frac{\Delta\tau}{2}\nabla \log \pi(\boldsymbol{\theta}^{(t-1)}), \Delta\tau \mathbf{I})}\right] \tag{40}$$

(c) If $u \leq \alpha_t$, accept $\boldsymbol{\theta'}$ and set $\boldsymbol{\theta}^{(t)} = \boldsymbol{\theta'}$. Otherwise (i.e., if $u > \alpha_t$), reject $\boldsymbol{\theta'}$ and set $\boldsymbol{\theta}^{(t)} = \boldsymbol{\theta}^{(t-1)}$.

3 Approximate the integral in Eq. (2) using Eq. (14).

It would be tempting to use more than one step of the Euler–Maruyama (or other SDE simulation method) in order to improve the proposal distribution. Unfortunately, this is not possible, because with multiple steps the evaluation of the transition density becomes intractable. This limits the proposal to very local moves, and hence the algorithm only provides a small improvement w.r.t. the random walk MH algorithm. A related algorithm which allows for larger jumps (and thus a greater improvement) is the Hamiltonian Monte Carlo (HMC) algorithm which we discuss next.

However, before we describe the HMC algorithm, let us make a short remark regarding the scaling of MALA. The seminar work of Roberts and Rosenthal [214] concluded that the optimal asymptotic AR (i.e., as $D_\theta \to \infty$) for MALA is approximately 0.574. Furthermore, they showed that the proposal variance should scale as $D_\theta^{-1/3}$, and thus $\mathcal{O}(D_\theta^{1/3})$ are required to ensure the algorithm's convergence. Pillai et al. also studied the efficiency of MALA on a natural class of target measures supported on an infinite dimensional Hilbert space, confirming that the optimal scaling AR is 0.574 [160]. Intuitively, this increased optimal AR w.r.t. the random walk MH algorithm (whose optimal AR is around 0.234) is due to the incorporation of additional information about the target into the sampling through the use of the SDE.

3.3.2 Hamiltonian Monte Carlo

The *Hamiltonian Monte Carlo (HMC)* or the *hybrid Monte Carlo (HMC)* method [215, 216], uses a statistical physical simulation of a physical system to form the proposal distribution for the MH algorithm. It is based on considering a particle system with the following Hamiltonian:

$$H(\boldsymbol{\theta}, \boldsymbol{\rho}) = -\log \pi(\boldsymbol{\theta}) + \frac{1}{2}\boldsymbol{\rho}^T \boldsymbol{\rho}, \tag{41}$$

where $\boldsymbol{\theta}$ can be interpreted as the generalized coordinate and $\boldsymbol{\rho}$ is the corresponding momentum. Assuming a suitable temperature, the distribution of the particles is then given by

$$p(\boldsymbol{\theta}, \boldsymbol{\rho}) = \frac{1}{Z}\exp(-H(\boldsymbol{\theta}, \boldsymbol{\rho})) = \bar{\pi}(\boldsymbol{\theta})\mathcal{N}(\boldsymbol{\rho}|\mathbf{0}, \mathbf{I}), \tag{42}$$

which has the target density, $\bar{\pi}(\boldsymbol{\theta})$, as its marginal PDF.

The Hamiltonian equations for the dynamics of the particles in fictious time τ are now given by

$$\frac{d\boldsymbol{\theta}}{d\tau} = \nabla_{\boldsymbol{\rho}} H = \boldsymbol{\rho}, \tag{43a}$$

$$\frac{d\boldsymbol{\rho}}{d\tau} = -\nabla_{\boldsymbol{\theta}} H = \nabla \log \pi(\boldsymbol{\theta}). \tag{43b}$$

The HMC algorithm constructs the proposal distribution by simulating trajectories from the Hamiltonian equations. Because an exact simulation is not possible, we need to use again numerical methods to simulate the trajectories. In order to construct a valid MH algorithm, a symplectic integrator such as the Leapfrog method (e.g., see [217]) needs to be used. Then, we can ensure both the preservation of the volume element as well as the time-reversibility of the simulation, which enables us to correct for the numerical solution inaccuracy by using a single MH acceptance step.

One step of the Leapfrog method for the Hamiltonian equations starting from τ with step size $\Delta\tau$ is given as

$$\widetilde{\boldsymbol{\rho}}^{(\tau+\Delta\tau/2)} = \widetilde{\boldsymbol{\rho}}^{(\tau)} + \frac{\Delta\tau}{2}\nabla \log \pi\left(\widetilde{\boldsymbol{\theta}}^{(\tau)}\right), \tag{44a}$$

$$\widetilde{\boldsymbol{\theta}}^{(\tau+\Delta\tau)} = \widetilde{\boldsymbol{\theta}}^{(\tau)} + \Delta\tau\, \widetilde{\boldsymbol{\rho}}^{(\tau+\Delta\tau/2)}, \tag{44b}$$

$$\widetilde{\boldsymbol{\rho}}^{(\tau+\Delta\tau)} = \widetilde{\boldsymbol{\rho}}^{(\tau+\Delta\tau/2)} + \frac{\Delta\tau}{2}\nabla \log \pi\left(\widetilde{\boldsymbol{\theta}}^{(\tau+\Delta\tau)}\right). \tag{44c}$$

The resulting HMC method is shown in Algorithm 16.

Algorithm 16 Hamiltonian Monte Carlo (HMC) Algorithm.

1 **Initialization:** Choose an initial state $\boldsymbol{\theta}^{(0)}$, the discretization step $\Delta\tau$, the number of integration steps L, the total number of iterations (T), and the burn-in period (T_b).

2 FOR $t = 1, \ldots, T$:

(a) Draw $u \sim \mathcal{U}([0, 1))$, numerically solve the Hamiltonian equations, (43a) and (43b), using L steps of a Leapfrog method (44) starting from $\widetilde{\boldsymbol{\theta}}^{(0)} = \boldsymbol{\theta}^{(t-1)}$ and $\widetilde{\boldsymbol{\rho}}^{(0)} \sim \mathcal{N}(0, \mathbf{I})$, setting $\boldsymbol{\theta}' = \widetilde{\boldsymbol{\theta}}^{(L\Delta\tau)}$ and $\boldsymbol{\rho}' = -\widetilde{\boldsymbol{\rho}}^{(L\Delta\tau)}$.

(b) Compute the acceptance probability:

$$\begin{aligned}\alpha_t &= \alpha(\boldsymbol{\theta}', \boldsymbol{\rho}'; \boldsymbol{\theta}^{(t-1)}, \boldsymbol{\rho}_{t-1}) \\ &= \min\left[1, \exp\left(-H(\boldsymbol{\theta}', \boldsymbol{\rho}') + H(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{\rho}_{t-1})\right)\right]\end{aligned} \tag{45}$$

(c) If $u \leq \alpha_t$, accept $\boldsymbol{\theta}'$ and set $\boldsymbol{\theta}^{(t)} = \boldsymbol{\theta}'$. Otherwise (i.e., if $u > \alpha_t$), reject $\boldsymbol{\theta}'$ and set $\boldsymbol{\theta}^{(t)} = \boldsymbol{\theta}^{(t-1)}$.

3 Approximate the integral in Eq. (2) using Eq. (14).

As discussed, for example, in [212], a single-step of the HMC algorithm is equivalent to MALA, and hence the two methods are closely related. There are numerous improvements and modifications to this basic algorithm—for example, we do not need to randomize the momenta fully at each time step, we can use preconditioning to improve the numerical stability and the mixing rate, and we can adapt the step sizes as well as the number of steps.

Finally, the optimal scaling of the HMC algorithm has been analyzed by Beskos et al. In [218], they prove that it

requires $\mathcal{O}(d^{1/4})$ steps to traverse the state space and that the asymptotically optimal AR for the HMC is 0.651, even higher than the optimal AR for MALA (0.574). This shows that HMC is more efficient than MALA in the incorporation of information about the target into the sampling approach.

3.3.3 Riemann manifold MALA and HMC

A practical challenge in both MALA and HMC methods is that their performance heavily depends on the particular parameterization chosen for $\boldsymbol{\theta}$. Although this problem can be diminished by using a preconditioning matrix to compensate for the uneven scaling of the parameters, its manual selection can be hard. Fortunately, a more general automatic selection procedure is provided by the Riemann manifold Monte Carlo methods that we briefly discuss here.

The idea of the *Riemann manifold Langevian and Hamiltonian Monte Carlo* methods is to perform the Langevin or Hamiltonian simulations in a suitable Riemann manifold instead of the Euclidean space [212, 219]. Although the idea was already proposed by [219], the introduction of proper symplectic integrators by [212] led to an exact MCMC algorithm. Let us recall that the squared distance between two locations $\boldsymbol{\theta}$ and $\boldsymbol{\theta} + d\boldsymbol{\theta}$ in Euclidean space is given by $d^2 = d\boldsymbol{\theta}^T d\boldsymbol{\theta}$. In the Riemann manifold, the distance is generalized to be $d^2 = d\boldsymbol{\theta}^T \mathbf{G}^{-1}(\boldsymbol{\theta}) d\boldsymbol{\theta}$, where $\mathbf{G}(\boldsymbol{\theta})$ is the metric tensor, which is a positive definite matrix for any given $\boldsymbol{\theta}$. A particularly useful metric tensor in the context of probability distributions is the one arising from information geometry, which is given as

$$\mathbf{G}(\boldsymbol{\theta}) = \mathrm{E}_{\mathbf{y}|\boldsymbol{\theta}}[\nabla \log \pi(\boldsymbol{\theta}|\mathbf{y}) \nabla \log \pi(\boldsymbol{\theta}|\mathbf{y})^T]. \tag{46}$$

We can now modify the MALA method such that the SDE evolves along the Riemann manifold instead of the Euclidean space as follows:

$$d\boldsymbol{\theta}(\tau) = \tilde{\mathbf{f}}(\boldsymbol{\theta}(t))d\tau + d\tilde{\mathbf{b}}(\tau), \tag{47}$$

where

$$\tilde{\mathbf{f}}(\boldsymbol{\theta}) = \frac{1}{2}\mathbf{G}^{-1}(\boldsymbol{\theta}) \nabla \log \pi(\boldsymbol{\theta}), \tag{48a}$$

$$d\tilde{b}_i = |\mathbf{G}(\boldsymbol{\theta})|^{-1/2} \sum_j \frac{\partial}{\partial \theta_j} [\mathbf{G}^{-1}(\boldsymbol{\theta})]_{ij} |\mathbf{G}(\boldsymbol{\theta})|^{-1/2} dt + [\mathbf{G}^{-1/2}(\boldsymbol{\theta}) d\mathbf{b}]_i. \tag{48b}$$

The *Riemann manifold Langevian Monte Carlo (MMALA)* algorithm can now be constructed by replacing the SDE in the basic MALA (see Algorithm 15) with the SDE defined above in Eqs. (48a) and (48b). For further details, the reader is referred to [212].

In the *Riemann manifold Hamiltonian Monte Carlo (RMHMC)* we construct the particle system dynamics in the Riemann manifold. This results in the following Hamiltonian:

$$H(\boldsymbol{\theta}, \boldsymbol{\rho}) = -\log \pi(\boldsymbol{\theta}) + \frac{1}{2}\log |2\pi \mathbf{G}(\boldsymbol{\theta})| + \frac{1}{2}\boldsymbol{\rho}^T \mathbf{G}^{-1}(\boldsymbol{\theta})\boldsymbol{\rho}, \tag{49}$$

and the Hamiltonian equations are now given as

$$\frac{d\boldsymbol{\theta}}{d\tau} = \nabla_{\boldsymbol{\rho}} H = \mathbf{G}^{-1}(\boldsymbol{\theta})\boldsymbol{\rho}, \tag{50a}$$

$$\frac{d\boldsymbol{\rho}}{d\tau} = -\nabla_{\boldsymbol{\theta}} H = \nabla \log \pi(\boldsymbol{\theta}) + \mathbf{h}(\boldsymbol{\theta}), \tag{50b}$$

where the additional term in Eq. (50b) is given by

$$h_i(\boldsymbol{\theta}) = -\frac{1}{2}\mathrm{tr}\left\{\mathbf{G}^{-1}(\boldsymbol{\theta})\frac{\partial \mathbf{G}(\boldsymbol{\theta})}{\partial \theta_i}\right\} + \frac{1}{2}\boldsymbol{\rho}^T \mathbf{G}^{-1}(\boldsymbol{\theta})\frac{\partial \mathbf{G}(\boldsymbol{\theta})}{\partial \theta_i}\mathbf{G}^{-1}(\boldsymbol{\theta})\boldsymbol{\rho}. \tag{51}$$

Although the construction of the RMHMC is analogous to that of the HMC, the selection of integration method requires much more care. The simple Leapfrog method is no longer enough now, and we need to use a more general symplectic integrator. For further details on this issue, see again [212].

3.3.4 Step-size and trajectory-length adaptation methods

Selecting the step size of the Leapfrog method is important for the performance of both MALA- and HMC-based methods. As discussed in [217], fortunately, the step size selection does not have a huge impact on the error in the Hamiltonian provided that it is small enough to make the discrete dynamics stable. For analysis on practical selection of step sizes as well as the lengths of trajectories see [217]. In [220] the authors propose no-U-turn sampler (NUTS) method which approaches the problem by limiting the trajectory to a length where it would change the direction. In NUTS, the step length is adapted using a stochastic optimization method. Some step size estimation methods for Leapfrog and HMC have also been provided in [221, 222]. Finally, the optimal step size scaling with the number of dimensions as well as the optimal acceptance rate of HMC has also been analyzed (e.g., see [223, 224]).

3.3.5 The geometric foundations of HMC and further considerations

Before concluding this section, let us remark that gradient-based algorithms have obtained a wide success in many applications. For instance, in high-dimensional problems, where the probability mass is typically concentrated in a very small portion of the space, they should probably be the first choice for practitioners. Recently, several researchers have developed a rigorous understanding of the reasons for their good performance on

difficult problems, as well as suggestions about their practical application. Intuitively, in high dimensional applications, exploiting the gradient information is crucial to make large jumps away from the current state and, at the same time, ensure that they are highly likely to be accepted. In [225], the authors have suggested that, in order to build a good proposal mechanism in an MH-type method, it is required to define a family of transformations preserving the target measure which form a Lie group on composition, as this ensures that proposals generated are both far away from the previous point and highly likely to be accepted. They also show that HMC emerges naturally when attempting to construct these transformations using ideas from differential geometry [225, 226]. However, it is also important to emphasize that gradient-based methods suffer from some important problems, as noted by Nishimura and Dunson [227]: one is the efficient tuning and/or adaptation of the parameters of the algorithm (which is not an easy task, in general), and another one is their application in multimodal scenarios; see [227, 228] and the discussion in [212] for further information about this issue.

More generally, note that *deterministic procedures* have been included within the sampling algorithms in order to reduce the computational demand of the MC methods and the variance of the resulting estimators. In this section we have explored one widely used possibility: exploiting the gradient information. Another idea, employed in quasi Monte Carlo methods, is using deterministic sequences of points based on the concept of *low-discrepancy* [229].

3.4 Other advanced schemes

In this section, we briefly discuss other important topics and relevant methodologies which are related to the algorithms described in the rest of this work.

3.4.1 Parallel schemes and implementations

A long run of a single chain can remain trapped in a local mode (e.g., when the parameters of the proposal PDF are not well-tuned) or the convergence can be very slow. In some cases (e.g., when the posterior density is multimodal), the use of shorter parallel MCMC chains can be advantageous. Thus, in order to speed up the exploration of the state space, and especially in order to deal with high-dimensional applications, several schemes employing parallel chains have been proposed [230–232], as well as multiple try and interacting schemes (see Section 5.1.1). Several other population-based techniques can also be found in the literature [233–238]. Finally, the use of non-reversible parallel MH algorithms has been also proposed [239, 240].

In addition, the interest in the parallel computation can also be due to other motivations. For instance, several authors have studied the parallelization of MCMC algorithms, that have traditionally been implemented in an iterative non-parallel fashion, in order to reduce their computation time [241]. Furthermore, parallel MCMC schemes are required in *big data* problems, where one possible approach consists of splitting the complete posterior distribution into several partial sub-posteriors [242–245]. Moreover, in the literature there is a great interest in the parallel implementation of MCMC algorithms so that the computation is distributed across a bunch of parallel processors [246–249].

Finally, note that the optimal parallelization strategy for an MCMC algorithm may differ depending on the properties of the parallel system under consideration. For instance, on GPUs parallelizing the individual MCMC steps can yield large performance improvement [250]. On the other hand, on distributed systems a better approach can consist in parallelizing the MCMC algorithm itself [251].

3.4.2 Delayed rejection Metropolis sampling

In Section 5.1, we will describe several multiple try schemes which extend the classical MH method, using (and comparing) different candidates at each iteration. Here, we show an alternative use of different candidates in one iteration of an MH-type method [252]. The idea behind the so called *delayed rejection Metropolis (DRM)* algorithm is the following. In the standard MH algorithm, one sample is proposed at each iteration, $\widetilde{\boldsymbol{\theta}}^{(1)} \sim q_1(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t-1)})$,[21] and accepted with probability

$$\alpha_1(\widetilde{\boldsymbol{\theta}}^{(1)}, \boldsymbol{\theta}^{(t-1)}) = \min\left[1, \frac{\pi(\widetilde{\boldsymbol{\theta}}^{(1)}) q_1(\boldsymbol{\theta}^{(t-1)}|\widetilde{\boldsymbol{\theta}}^{(1)})}{\pi(\widetilde{\boldsymbol{\theta}}^{(t-1)}) q_1(\widetilde{\boldsymbol{\theta}}^{(1)}|\boldsymbol{\theta}^{(t-1)})}\right]. \tag{52}$$

If $\widetilde{\boldsymbol{\theta}}^{(1)}$ is accepted, then $\boldsymbol{\theta}^{(t)} = \widetilde{\boldsymbol{\theta}}^{(1)}$ and the chain is moved forward, as in the standard MH algorithm. However, if $\widetilde{\boldsymbol{\theta}}^{(1)}$ is rejected, the DRM method suggests drawing another sample, $\widetilde{\boldsymbol{\theta}}^{(2)} \sim q_2(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t-1)}, \widetilde{\boldsymbol{\theta}}^{(1)})$, from a different proposal PDF, q_2, which takes into account the previous candidate, $\widetilde{\boldsymbol{\theta}}^{(1)}$, and accepting it with a suitable acceptance probability:

$$\alpha_2^{(t)} \equiv \alpha_2(\widetilde{\boldsymbol{\theta}}^{(2)}, \boldsymbol{\theta}^{(t-1)}) = \min\left[1, \frac{\rho(\widetilde{\boldsymbol{\theta}}^{(2)}, \boldsymbol{\theta}^{(t-1)}|\widetilde{\boldsymbol{\theta}}^{(1)})}{\rho(\boldsymbol{\theta}^{(t-1)}, \widetilde{\boldsymbol{\theta}}^{(2)}|\widetilde{\boldsymbol{\theta}}^{(1)})}\right], \tag{53}$$

[21] Note that the $\widetilde{\boldsymbol{\theta}}^{(1)}$ and $q_1(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t-1)})$ used here play the role of $\boldsymbol{\theta}'$ and $q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t-1)})$, respectively, in Algorithm 3.

where

$$\rho\left(\boldsymbol{\theta}^{(t-1)}, \widetilde{\boldsymbol{\theta}}^{(2)} | \widetilde{\boldsymbol{\theta}}^{(1)}\right) = \pi\left(\boldsymbol{\theta}^{(t-1)}\right) q_1\left(\widetilde{\boldsymbol{\theta}}^{(1)} | \boldsymbol{\theta}^{(t-1)}\right)\left(1-\alpha_1\left(\widetilde{\boldsymbol{\theta}}^{(1)}, \boldsymbol{\theta}^{(t-1)}\right)\right) \pi\left(\widetilde{\boldsymbol{\theta}}^{(2)}\right) q_2\left(\widetilde{\boldsymbol{\theta}}^{(2)} | \boldsymbol{\theta}^{(t-1)}, \widetilde{\boldsymbol{\theta}}^{(1)}\right). \quad (54)$$

The acceptance function of Eq. (53), $\alpha_2(\widetilde{\boldsymbol{\theta}}^{(2)}, \boldsymbol{\theta}^{(t-1)})$, is designed in order to ensure the ergodicity of the chain. Indeed, the kernel of the DRM algorithm satisfies the detailed balance condition [252]. If $\widetilde{\boldsymbol{\theta}}^{(2)}$ is rejected, then we can either set $\boldsymbol{\theta}^{(t)} = \boldsymbol{\theta}^{(t-1)}$ and perform another iteration of the algorithm or continue with the sequential strategy, drawing $\widetilde{\boldsymbol{\theta}}^{(3)} \sim q_3(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t-1)}, \widetilde{\boldsymbol{\theta}}^{(1)}, \widetilde{\boldsymbol{\theta}}^{(2)})$ and testing it with a proper acceptance probability $\alpha_3(\widetilde{\boldsymbol{\theta}}^{(3)}, \boldsymbol{\theta}^{(t-1)})$. The DRM method with two sequential steps is outlined in Algorithm 17. Unlike the multiple try schemes described in Section 5.1, in the DRM sampler the candidates are not compared together at the same time, but each candidate is drawn from a proposal PDF and then tested with an MH-type acceptance probability.

Algorithm 17 Delayed Rejection Metropolis with two acceptance steps.

1 **Initialization:** Choose two proposal functions, $q_1(\boldsymbol{\theta})$ and $q_2(\boldsymbol{\theta})$, an initial state $\boldsymbol{\theta}^{(0)}$, and the total number of iterations (T).

2 FOR $t = 1, \ldots, T$:

(a) Draw $\widetilde{\boldsymbol{\theta}}^{(1)} \sim q_1(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t-1)})$ and $u_1 \sim \mathcal{U}([0,1])$.

(b) Compute the acceptance probability of $\widetilde{\boldsymbol{\theta}}^{(1)}$:

$$\alpha_1^{(t)} \equiv \alpha_1(\widetilde{\boldsymbol{\theta}}^{(1)}, \boldsymbol{\theta}^{(t-1)}) = \min\left[1, \frac{\pi(\widetilde{\boldsymbol{\theta}}^{(1)}) q_1(\boldsymbol{\theta}^{(t-1)}|\widetilde{\boldsymbol{\theta}}^{(1)})}{\pi(\boldsymbol{\theta}^{(t-1)}) q_1(\widetilde{\boldsymbol{\theta}}^{(1)}|\boldsymbol{\theta}^{(t-1)})}\right], \quad (55)$$

(c) If $u_1 \leq \alpha_1^{(t)}$, accept $\widetilde{\boldsymbol{\theta}}^{(1)}$, set $\boldsymbol{\theta}^{(t)} = \widetilde{\boldsymbol{\theta}}^{(1)}$ and go back to step 2(a).

(d) Otherwise (i.e., if $u_1 > \alpha_1^{(t)}$):

i. Draw $\widetilde{\boldsymbol{\theta}}^{(2)} \sim q_2(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t-1)}, \widetilde{\boldsymbol{\theta}}^{(1)})$ and $u_2 \sim \mathcal{U}([0,1))$.

ii. Compute the acceptance probability of $\widetilde{\boldsymbol{\theta}}^{(2)}$:

$$\alpha_2^{(t)} \equiv \alpha_2(\widetilde{\boldsymbol{\theta}}^{(2)}, \boldsymbol{\theta}^{(t-1)}) = \min\left[1, \frac{\rho(\widetilde{\boldsymbol{\theta}}^{(2)}, \boldsymbol{\theta}^{(t-1)}|\widetilde{\boldsymbol{\theta}}^{(1)})}{\rho(\boldsymbol{\theta}^{(t-1)}, \widetilde{\boldsymbol{\theta}}^{(2)}|\widetilde{\boldsymbol{\theta}}^{(1)})}\right],$$

with $\rho(\boldsymbol{\theta}^{(t-1)}, \widetilde{\boldsymbol{\theta}}^{(2)}|\widetilde{\boldsymbol{\theta}}^{(1)})$ given by Eq. (54).

iii. If $u_2 \leq \alpha_2^{(t)}$, accept $\widetilde{\boldsymbol{\theta}}^{(2)}$ and set $\boldsymbol{\theta}^{(t)} = \widetilde{\boldsymbol{\theta}}^{(2)}$. Otherwise (i.e., if $u_2 > \alpha_2^{(t)}$), reject $\widetilde{\boldsymbol{\theta}}^{(2)}$ and set $\boldsymbol{\theta}^{(t)} = \boldsymbol{\theta}^{(t-1)}$.

3 Approximate the integral in Eq. (2) using Eq. (14).

3.4.3 Non-reversible chains

The MCMC techniques described so far fulfill the so-called *detailed balance condition*. In this case, the generated chain is *reversible*. Denoting as $K(\boldsymbol{\theta}'|\boldsymbol{\theta})$ the transition density of a specific MCMC method, the detailed balance condition is given by the Eq. 18 in Section 3.1.2, i.e.,

$$\bar{\pi}(\boldsymbol{\theta}) K(\boldsymbol{\theta}'|\boldsymbol{\theta}) = \bar{\pi}(\boldsymbol{\theta}') K(\boldsymbol{\theta}|\boldsymbol{\theta}'). \quad (56)$$

However, recent studies have shown that *non-reversible* chains can provide better performance [253–258]. Non-reversible chains can be generated using different procedures. For instance, defining a *vorticity density* $\gamma(\boldsymbol{\theta}, \boldsymbol{\theta}')$, such that

$$\gamma(\boldsymbol{\theta}, \boldsymbol{\theta}') = -\gamma(\boldsymbol{\theta}', \boldsymbol{\theta}),$$

$$\int_{\mathbf{A}\times\Theta} \gamma(\boldsymbol{\theta}, \boldsymbol{\theta}') d\boldsymbol{\theta}' d\boldsymbol{\theta} = 0, \qquad \mathbf{A} \in \mathcal{B}(\Theta),$$

where $\mathcal{B}(\Theta)$ is a Borel σ-algebra generated by open sets in Θ and, additionally,

$$\gamma(\boldsymbol{\theta}, \boldsymbol{\theta}') + \pi(\boldsymbol{\theta}') q(\boldsymbol{\theta}|\boldsymbol{\theta}') \geq 0,$$

for all $\boldsymbol{\theta}, \boldsymbol{\theta}' \in \Theta$ for which $\pi(\boldsymbol{\theta}') q(\boldsymbol{\theta}|\boldsymbol{\theta}') \neq 0$. Moreover, in the case that $\pi(\boldsymbol{\theta}') q(\boldsymbol{\theta}|\boldsymbol{\theta}') = 0$, we need $\gamma(\boldsymbol{\theta}, \boldsymbol{\theta}') = 0$. In this scenario, we can run an MH-type algorithm with acceptance probability

$$\alpha = \min\left[1, \frac{\gamma(\boldsymbol{\theta}, \boldsymbol{\theta}') + \pi(\boldsymbol{\theta}') q(\boldsymbol{\theta}|\boldsymbol{\theta}')}{\pi(\boldsymbol{\theta}) q(\boldsymbol{\theta}'|\boldsymbol{\theta})}\right].$$

It is possible to show that the generated chain by the MH method with the α above still has $\bar{\pi}$ as invariant density, although is non-reversible [253, 254].

Many non-reversible MCMC methods have recently been proposed, and some authors have also developed general frameworks to construct different irreversible MC algorithms [259, 260]. However, the amount of improvement provided by these schemes in complex practical problems still remains to be seen.

3.5 MCMC convergence diagnostics

Properly designed MCMC algorithms automatically produce samples from the target distribution after an initial transitory period. However, theoretically determining the length of this transitory period may be very difficult. Hence, during the finite running time of a simulation the Markov chain could fail to converge to its stationary distribution. In this case, the generated samples might not well represent the target PDF and any inference performed using them is bound to produce erroneous results. For

this reason it is important to study the output of the algorithm to determine if the MCMC simulation has properly converged. Methods and good practices for convergence diagnostics can be found, for example, in Chapter 11 of Gelman et al. (2013) [261] or Chapter 6 of Brooks et al. (2011) [216].

3.5.1 General principles of convergence diagnostics

Gelman and Shirley, in Chapter 6 of Brooks et al. (2011) [216], summarize the recommended convergence diagnostics as follows:

1. Run three or more chains in parallel with varying starting points. The starting points can be selected randomly around or within a simpler approximation.
2. The chains should be then split to two halves, where only the second half is retained. Diagnosis methods for between-chain and within-chain analysis can be then used to monitor the mixing of the chain in the second half.
3. After approximate convergence is obtained, the second halves of the chains should be then mixed together to summarize the target distribution. The autocorrelations should not matter at this stage any more.
4. Adaptive Markov chain Monte Carlo (AMCMC) methods can be used for tuning the proposal densities and other properties of the MCMC method. Provided that we always restart the MCMC method after adaptation, any adaptation method can be applied.

Additionally, the algorithm can be debugged by running it on a model with known parameters and checking that the posterior distributions are consistent with the true values.

3.5.2 Between-chain and within-chain diagnostics

Given a set of simulated MCMC samples—such as the second halves of the chains from step 2 in the previous section—it is then possible to investigate whether the samples have converged to the target distribution. Although there are a number of possible approaches for the convergence diagnostics (see, e.g., [262, 263]), the potential scale reduction factor (PSRF) [261, 264] is the diagnostic tool that is often recommended for practical use [216, 261].

Gelman et al. (2013) [261] define the PSRF ($\hat{R}$) as follows. Let us denote the chain consisting of samples from a scalar variable θ as $\theta^{(i,j)}$, where $i = 1, \ldots, M$ is the sample index and $j = 1, \ldots, S$ is the chain index. Compute then B and W, which correspond to the between-chain and within-chain variances:

$$B = \frac{M}{S-1} \sum_{j=1}^{S} (\bar{\theta}^{(\cdot,j)} - \bar{\theta}^{(\cdot,\cdot)})^2, \tag{57a}$$

$$W = \frac{1}{S} \sum_{j=1}^{S} s_j^2 = \frac{1}{S(M-1)} \sum_{j=1}^{S} \sum_{i=1}^{M} (\theta^{(i,j)} - \bar{\theta}^{(\cdot,j)})^2, \tag{57b}$$

where

$$\bar{\theta}^{(\cdot,j)} = \frac{1}{M} \sum_{i=1}^{M} \theta^{(i,j)}, \tag{58a}$$

$$\bar{\theta}^{(\cdot,\cdot)} = \frac{1}{S} \sum_{j=1}^{S} \bar{\theta}^{(\cdot,j)}, \tag{58b}$$

$$s_j^2 = \frac{1}{M-1} \sum_{i=1}^{M} (\theta^{(i,j)} - \bar{\theta}^{(\cdot,j)})^2. \tag{58c}$$

The PSRF can then be defined as [261]

$$\hat{R} = \sqrt{\frac{\widehat{\mathrm{var}}^{+}(\theta)}{W}}, \tag{59}$$

where

$$\widehat{\mathrm{var}}^{+}(\theta) = \frac{M-1}{M} W + \frac{1}{M} B \tag{60}$$

is an estimator for the posterior variance. In the multivariate case the PSRF values can be computed for each dimension separately.

The PSRF value should approach 1 when the convergence occurs. If the value is significantly higher than 1, then convergence has probably not occurred. Although in PSRF we should use multiple independent chains, a single chain can also be analyzed by splitting it into two or more parts and computing the PSRF as if the splits where independent chains.

3.5.3 Effective number of samples

Generally, the MCMC algorithms present a positive correlation among the generated samples. This clearly implies a loss of efficiency with respect to the case of independent samples, i.e., with respect to the classical Monte Carlo approach. Namely, positively correlated samples provide less statistical information than independent samples, meaning that the corresponding estimators will be less efficient in general, i.e., with a higher variance for a given sample size. The concept of effective sample size (ESS) has been introduced to measure this loss of efficiency [105], [208, Section 9.1]. If T is the length of the chain (without removing any burn-in period) and denoting as $\rho(\tau)$ the

autocorrelation at lag τ, the effective sample size for an MCMC algorithm is defined as

$$\text{ESS} = \frac{T}{1 + 2\sum_{\tau=1}^{\infty} \rho(\tau)}. \tag{61}$$

Clearly, positive correlations $\rho(\tau)$ decrease the ESS value, and hence $ESS < T$. If the correlation is zero (i.e., $\rho(\tau) = 0$ for all τ), then $ESS = T$, as in the classical Monte Carlo scheme using independent samples. In a similar fashion, it is also possible to define ESS measures for other Monte Carlo approaches, as shown in Section 4.5.

3.5.4 Other recent approaches

More recently, alternative approaches to the method described in the previous sections have been proposed to measure the convergence of MCMC algorithms. On the one hand, Gorham et al. have introduced a novel family of discrepancy measures based on Stein's method [265]. These measures bound the discrepancy between sample and target expectations over a large class of test functions, and some specific members of this family can be computed by solving a linear program [266]. Finally, using zero mean reproducing kernel theory, several authors have shown that other members of the Stein discrepancy family have a closed-form solution involving the sum of kernel evaluations over pairs of sample points [267–270]. On the other hand, Johndrow et al. have applied computational complexity theory to analyze how the computational efficiency of MCMC algorithms degrades with the problem's size [271]. Their goal is determining whether an MCMC algorithm will perform well in a given context/problem, rather than providing performance bounds which are often very loose to be of any practical use. Note that the aforementioned techniques are still not mature enough for their widespread use in practical applications (especially in high-dimensional problems), but this is a very active research field where we can expect new contributions in coming years.

4 Importance sampling

Importance sampling (IS) is a Monte Carlo methodology that can be used to characterize virtually any target PDF and to approximate its moments. Like any other MC method, IS techniques proceed by drawing samples from one or several proposal PDFs. However, unlike MCMC methods, that accept or discard the drawn samples according to some appropriate test, IS approaches accept all the samples and assign them a weight according to their quality in approximating the desired target distribution. Regarding the number of proposals, IS methods can be divided into classical or standard IS, where a single proposal is used to draw all the samples, and multiple IS (MIS), where a collection of proposals are used to draw the samples. With respect to the temporal evolution of the proposals, IS methods can be classified as non-adaptive or "static," where the proposals are fixed (i.e., their parameters are selected a priori and remain fixed for the whole simulation), and adaptive, where the parameters of the proposals are adapted iteratively in order to better approximate the desired target density. In the following section, we briefly review "static" IS (both using a single and multiple proposals), whereas in Section 4.2 we consider adaptive IS (AIS). Then, some remarks on the convergence of IS and AIS are provided in Section 4.3. This is followed by a discussion on the optimal proposal choice and the variance of the IS estimators in Section 4.4, and the definition of the effective sample size in Section 4.5. Afterwards, the concept of proper weights is introduced in Section 4.6 and this leads naturally to the group importance sampling (GIS) approach described in Section 4.7. Finally, a short introduction to sequential importance sampling (SIS) for the estimation of dynamic parameters is also provided in Section 4.8.

4.1 Standard and multiple importance sampling

4.1.1 Importance sampling with a single proposal

Let us consider a single proposal PDF, $\bar{q}(\boldsymbol{\theta})$, with heavier tails than the target, $\bar{\pi}(\boldsymbol{\theta})$[22]. The proposal is used to draw a set of M IID samples, $\{\boldsymbol{\theta}^{(m)}\}_{m=1}^{M}$ with $\boldsymbol{\theta}^{(m)} \sim \bar{q}(\boldsymbol{\theta})$. An importance weight is then associated to each sample according to

$$w_m = w(\boldsymbol{\theta}_m) = \frac{\pi(\boldsymbol{\theta}_m)}{\bar{q}(\boldsymbol{\theta}_m)}, \quad m = 1, \ldots, M. \tag{62}$$

If the normalizing constant of the target, Z, is known, then we can approximate the targeted integral of Eq. (2) by the so-called *unnormalized estimator*:

$$\widehat{I}_M = \frac{1}{MZ} \sum_{m=1}^{M} w_m g(\boldsymbol{\theta}^{(m)}). \tag{63}$$

Unfortunately, Z is unknown in many practical problems, and we have to resort to the alternative *self-normalized estimator*, which is given by

$$\widetilde{I}_M = \frac{1}{M\widehat{Z}} \sum_{m=1}^{M} w_m g(\boldsymbol{\theta}^{(m)}), \tag{64}$$

where $\widehat{Z} = \frac{1}{M}\sum_{m=1}^{M} w_m$ is an unbiased estimator of Z [104]. Note that both $\widetilde{I}_M$ and $\widehat{I}_M$ are consistent estimators of I, i.e., both $\widetilde{I}_M \to I$ and $\widehat{I}_M \to I$ as $M \to \infty$ (see Section 4.3 for further details). However, their performance (as measured by their variance) is highly related to the discrepancy between $\bar{\pi}(\boldsymbol{\theta})|g(\boldsymbol{\theta})|$ and the proposal

[22] Note that one of the main disadvantages of IS is the fact that the variance of the IS estimator becomes infinite when the tails of the proposal, $\bar{q}(\boldsymbol{\theta})$, decay faster than $\bar{\pi}(\boldsymbol{\theta})^2$ due to the appearance of $\bar{q}(\boldsymbol{\theta})$ in the denominator of the weights in (62) [91]. Therefore, in order prevent this situation, which leads to the complete failure of the IS sampler, a common restriction is selecting a proposal with heavier tails than the target.

$\bar{q}(\boldsymbol{\theta})$ [104]. Indeed, with a properly selected proposal IS methods can provide a lower variance than the direct application of the MC method, whereas a poor selection can easily lead to infinite variance estimators. In practice, since IS approaches are often used to simultaneously approximate several functions $(g_1(\boldsymbol{\theta}), g_2(\boldsymbol{\theta}), \ldots)$, a common approach is simply minimizing the mismatch between the proposal $\bar{q}$ and the target $\bar{\pi}$ [93, Section 3.2].

4.1.2 Importance sampling with multiple proposals

Several reasons justify the use of more than one proposal. On the one hand, the target can be multimodal and therefore it can be better fitted with a mixture of proposals. On the other hand, choosing a single good proposal a priori is usually very difficult, and adaptive processes must be performed (as described in Section 4.2) in order to tailor the proposal to the target. In this situation, the exploration of the parameter space, Θ, is more efficient when multiple proposals, $\{q_n(\boldsymbol{\theta})\}_{n=1}^N$, are available [91, 272]. The use of several proposals is usually known as *multiple importance sampling (MIS)* in the IS literature, and it is a key feature of most state-of-the-art adaptive IS algorithms (e.g., [95, 98, 99, 273, 274]). A generic MIS framework has been recently proposed in [275], where it is shown that several sampling and weighting schemes can be used. For the sake of conciseness, here, we present a single sampling scheme with two different common weighting schemes that evidence the flexibility of MIS. Let us consider a sampling scheme where exactly one sample per proposal (i.e., $M = N$) is drawn,

$$\boldsymbol{\theta}^{(m)} \sim \bar{q}_m(\boldsymbol{\theta}), \qquad m = 1, ..., M. \tag{65}$$

Several proper weighting strategies for this sampling approach have been proposed in the literature (e.g., see [275] and the references therein for a review of different valid weighting schemes), but the two most common ones are the following:

- *Standard MIS (s-MIS)* [95]:

$$w_m = \frac{\pi(\boldsymbol{\theta}^{(m)})}{\bar{q}_m(\boldsymbol{\theta}^{(m)})}, \quad m = 1, \ldots, M; \tag{66}$$

- *Deterministic mixture MIS (DM)* [91]:

$$w_m = \frac{\pi(\boldsymbol{\theta}^{(m)})}{\psi(\boldsymbol{\theta}^{(m)})} = \frac{\pi(\boldsymbol{\theta}^{(m)})}{\frac{1}{N}\sum_{j=1}^N \bar{q}_j(\boldsymbol{\theta}^{(m)})}, \quad m = 1, \ldots, M, \tag{67}$$

  where $\psi(\boldsymbol{\theta}) = \frac{1}{N}\sum_{j=1}^N \bar{q}_j(\boldsymbol{\theta})$ is the mixture PDF, composed of all the proposal PDFs with equal weights in the mixture.

On the one hand, the unnormalized estimator using any of those two sets of weights (i.e., Eq. (66) or Eq. (67)) is consistent and unbiased. On the other hand, the self-normalized estimator is consistent and asymptotically unbiased using both sets of weights. However, the performance of the estimators may differ substantially depending on which set of weights is used. For instance, the performance of the unnormalized estimator $\widehat{I}_M$ with the DM approach is superior (in terms of attaining a reduced variance) w.r.t. the s-MIS approach [275]. Finally, note that both weighting alternatives require the same number of target evaluations, but the DM estimator is computationally more expensive w.r.t. the number of proposal evaluations (N^2 evaluations for the DM weights vs. N evaluations for the s-MIS ones). Several efficient approaches to reduce the variance of the estimators, while limiting the computational complexity, have been proposed [276, 277].

4.2 Adaptive importance sampling

An adaptive importance sampler is an iterative algorithm where the proposals are iteratively improved using either the previously drawn samples or some other independent mechanism. In particular, here, the focus is on the more general case of adaptive MIS, where the set of N proposals $\{\bar{q}_n(\boldsymbol{\theta})\}_{n=1}^N$ are adapted over the subsequent iterations of the algorithm. Introducing the time-step into the notation, the set of available proposals becomes $\{\bar{q}_{n,t}(\boldsymbol{\theta})\}_{n=1}^N$, where t indicates the tth iteration. Therefore, not only the parameters of the proposals can be updated, but even the family of distributions can be changed. For the sake of simplicity, in this review, we only consider location-scale densities, such that each proposal, $\bar{q}_{n,t}$, is completely characterized by a mean vector, $\boldsymbol{\mu}_{n,t}$, and a covariance matrix, $\mathbf{C}_{n,t}$.

Algorithm 18 describes a generic AIS algorithm, where only the location parameters are adapted (i.e., $\mathbf{C}_{n,t} = \mathbf{C}_n$ for all t)[23]. At the tth iteration, M independent samples are drawn from each proposal (step 1 of Algorithm 18), i.e.,

$$\boldsymbol{\theta}_{n,t}^{(m)} \sim q_{n,t}(\boldsymbol{\theta}|\boldsymbol{\mu}_{n,t}, \mathbf{C}_n). \tag{68}$$

Each sample is then associated with an importance weight of the form

$$w_{n,t}^{(m)} = \frac{\pi(\boldsymbol{\theta}_{n,t}^{(m)})}{\Phi_{n,t}(\boldsymbol{\theta}_{n,t}^{(m)})}, \tag{69}$$

for $m = 1, \ldots, M$ and $n = 1, \ldots, N$ (step 2). Note that we use a generic function $\Phi_{n,t}$ at the denominator of the weight. Similarly to the static MIS methodology, different weighting schemes are possible (in the adaptive setup there is an even larger selection of valid weighting schemes). In particular, a basic requirement is choosing

[23] Adapting the scale parameters is dangerous and must be done with a lot of care, since it can lead to ill conditioned proposals that result in estimators with huge variances (potentially infinite). For this reason, many AIS algorithms use multiple proposals with different scales and only adapt their locations. However, several algorithms that adapt the scale parameter have also been proposed [96, 98, 278].

Table 3 Examples of $\Phi_{n,t}$ (function used in the denominator for the estimation) and $\Omega_{n,t}$ (function used in the denominator for the adaptation) that can be found in the literature

Weight den.	**PMC** [95]	AMIS ($N=1$) [98]	APIS [99]
$\Phi_{n,t}(\boldsymbol{\theta})$	$q_{n,t}(\boldsymbol{\theta})$	$\frac{1}{t}\sum_{\tau=1}^{t} q_\tau(\boldsymbol{\theta})$	$\frac{1}{N}\sum_{n=1}^{N} q_{n,t}(\boldsymbol{\theta})$
$\Omega_{n,t}(\boldsymbol{\theta})$	$q_{n,t}(\boldsymbol{\theta})$	$\frac{1}{t}\sum_{\tau=1}^{t} q_\tau(\boldsymbol{\theta})$	$q_{n,t}(\boldsymbol{\theta})$

the set of functions $\{\Phi_{n,t}\}_{n=1,t=1}^{N,T}$ in such a way that the estimator of Eq. (63) is unbiased [106, Section 2.5.4], [273, 275]. Different choices of $\Phi_{n,t}$ available in the literature can be found in Table 3. Finally, the location parameters of the multiple proposals are adapted in step 3. Regarding the adaptation of the location parameter, here, we divide the methods that can be found in the literature into two main groups:

1. Algorithms that employ the previous samples for the adaptation. This approach is summarized in Table 4, and includes the possible application of resampling steps over a subset of these weighted samples [95, 274, 279] or fitting the moments of the proposals [98, 280].
2. Algorithms with independent adaptive procedures (detached from the sampling procedure), e.g., the gradient of $\bar{\pi}$ used in [278, 281], or the MCMC techniques applied to adapt the location parameters in [273, 282–285]. While these approaches usually present a superior performance, they tend to be computationally more expensive. Table 5 summarizes them.

4.3 Convergence of IS and AIS methods

In this section, we briefly discuss the convergence of IS and AIS estimators. We consider both the unnormalized estimator of Eq. (63) and the self-normalized estimator of Eq. (64).

Table 4 Adaptation based on weighted samples

Let us compute the set of IS weights (potentially different from those of Eq. (71)),

$$\rho_{n,t}^{(m)} = \frac{\pi(\boldsymbol{\theta}_{n,t}^{(m)})}{\Omega_{n,t}(\boldsymbol{\theta}_{n,t}^{(m)})},$$

where $\Omega_{n,t}$ are chosen in such a way that they do not jeopardize the consistency of the IS estimators, and they can be equal to $w_{n,t}^{(m)}$ in Eq. (71) or not

(see Table 3). Two different procedures are used in literature:

P1 Apply some resampling strategy to $\{\boldsymbol{\mu}_{n,t-1}\}_{n=1}^{N}$, with probabilities according to the weights $\rho_{n,t}^{(m)}$, to obtain $\{\boldsymbol{\mu}_{n,t}\}_{n=1}^{N}$ [95, 274, 286]. Nonlinear transformations of $\rho_{n,t}^{(m)}$ can also be applied [279].

P2 Build estimators of some moments of π employing $\rho_{n,t}^{(m)}$, and use this information to obtain $\{\boldsymbol{\mu}_{n,t}\}_{n=1}^{N}$ [96, 98, 287].

Table 5 More sophisticated adaptation procedures

P3 Adaptation by using MCMC transitions to obtain $\{\boldsymbol{\mu}_{n,t}\}_{n=1}^{N}$ given $\{\boldsymbol{\mu}_{n,t-1}\}_{n=1}^{N}$, as in [273, 283–286].

P4 Adaptation by using stochastic gradient search of π for moving $\{\boldsymbol{\mu}_{n,t-1}\}_{n=1}^{N}$ to $\{\boldsymbol{\mu}_{n,t}\}_{n=1}^{N}$ [278, 281].

First of all, let us consider a fixed set of proposals, $q_n(\boldsymbol{\theta})$ for $n = 1, \ldots, N$. Then, it can be easily shown that the unnormalized estimator of Eq. (63) is an unbiased estimator of the desired integral, i.e., $\mathbb{E}(\widehat{I}_M) = I$ for any value of M. On the other hand, the strong law of large numbers guarantees that the self-normalized estimator is asymptotically unbiased, i.e., $\widetilde{I}_M \to I$ a.s. as $M \to \infty$. These two results hold, regardless of whether the s-MIS or the DM-MIS weights are used, as long as $q(\boldsymbol{\theta}) > 0$ whenever $\pi(\boldsymbol{\theta}) > 0$ [288]. Furthermore, under some additional mild regularity conditions (see [288]), the following CLTs can be established:

$$\frac{\widehat{I}_M - I}{\sqrt{\widehat{V}_M}} \xrightarrow{d} \mathcal{N}(0,1), \tag{75a}$$

$$\frac{\widetilde{I}_M - I}{\sqrt{\widetilde{V}_M}} \xrightarrow{d} \mathcal{N}(0,1), \tag{75b}$$

where

$$\widehat{V}_M = \frac{1}{M}\mathbb{E}_{\bar{\pi}}((g(\boldsymbol{\theta}) - I)^2 w(\boldsymbol{\theta})), \tag{76a}$$

$$\widetilde{V}_M = \frac{1}{M}\mathbb{E}_{\bar{\pi}}((g(\boldsymbol{\theta}) - I)^2 \bar{w}(\boldsymbol{\theta})), \tag{76b}$$

$w(\boldsymbol{\theta})$ is the weighting function used to construct $\widehat{I}_M$ and $\bar{w}(\boldsymbol{\theta})$ is its normalized counterpart, which is used in the formulation of $\widetilde{I}_M$. Hence, $\widehat{I}_M \xrightarrow{d} \mathcal{N}(I, \widehat{V}_M)$ and $\widetilde{I}_M \xrightarrow{d} \mathcal{N}(I, \widetilde{V}_M)$ as $M \to \infty$. Note that, even though the convergence of both estimators to the desired integral for any proper weighting scheme is ensured, the differences in convergence rate can be quite large (e.g., see [275] for variance proofs and a discussion on this issue).

Now, let us briefly consider adaptive IS schemes, where the proposals are iteratively updated. First of all, note that the previous results also hold for AIS methods. However, a second question that arises in this case is the convergence of the estimators as the proposals are adapted. This issue is tackled by Oh and Berger in [287], where they analyze the estimator obtained by aggregating weighted samples produced through several consecutive iterations using different proposal PDFs. More precisely, they consider the estimator in Eq. (74a) and prove, under fairly general conditions, that $\widetilde{I}_{MNT} \to I$ a.s. and $\widetilde{I}_{MNT} \xrightarrow{d} \mathcal{N}(I, \widetilde{V}_{MNT})$ (with the decay of $\widetilde{V}_{MNT}$ proportional to $\frac{1}{MNT}$, the optimal Monte Carlo approximation rate) as $MNT \to \infty$. See [287] for further details.

Algorithm 18 Generic Adaptive Population Importance Sampler.

1. **Initialization:** Choose the number of proposals (N), the number of samples drawn per proposal and iteration (M), the initial proposals, $q_{n,0}(\boldsymbol{\theta}|\mu_{n,0}, \mathbf{C}_n)$ for $n = 1, \ldots, N$ with appropriate values of $\mu_{n,0}$ and $\mathbf{C}_n$, and the total number of iterations (T).
2. FOR $t = 1, \ldots, T$:
   (a) **Sampling:** Draw M samples $\boldsymbol{\theta}_{n,t}^{(1)}, \ldots, \boldsymbol{\theta}_{n,t}^{(M)}$ from each of the N proposal PDFs in the population $\{\bar{q}_1, \ldots, \bar{q}_N\}$, i.e.,
   $$\boldsymbol{\theta}_{n,t}^{(m)} \sim q_{n,t}(\boldsymbol{\theta}|\boldsymbol{\mu}_{n,t}, \mathbf{C}_n), \tag{70}$$
   for $m = 1, \ldots, M$.
   (b) **Weighting:** Weight the samples, $\{\boldsymbol{\theta}_{n,t}^{(m)}\}_{n=1}^N$, using
   $$w_{n,t}^{(m)} = \frac{\pi(\boldsymbol{\theta}_{n,t}^{(m)})}{\Phi_{n,t}(\boldsymbol{\theta}_{n,t}^{(m)})} \tag{71}$$
   (c) **Adaptation of the means:** Apply some suitable procedure to update the mean vectors,
   $$\{\boldsymbol{\mu}_{n,t-1}\}_{n=1}^N \longrightarrow \{\boldsymbol{\mu}_{n,t}\}_{n=1}^N, \tag{72}$$
   without jeopardizing the consistency of the IS estimators.
3. **Output:** Approximate the integral in Eq. (2) using either the unnormalized estimator when the normalizing constant is known,
   $$\widehat{I}_{MNT} = \frac{1}{MNTZ} \sum_{t=1}^{T} \sum_{n=1}^{N} \sum_{m=1}^{M} w_{n,t}^{(m)} g(\boldsymbol{\theta}_{n,t}^{(m)}), \tag{73}$$
   or the self-normalizing estimator when the normalizing constant is unknown,
   $$\widetilde{I}_{MNT} = \frac{1}{MNT\hat{Z}} \sum_{t=1}^{T} \sum_{n=1}^{N} \sum_{m=1}^{M} w_{n,t}^{(m)} g(\boldsymbol{\theta}_{n,t}^{(m)}), \tag{74a}$$
   $$\widehat{Z} = \frac{1}{MNT} \sum_{t=1}^{T} \sum_{n=1}^{N} \sum_{m=1}^{M} w_{n,t}^{(m)}. \tag{74b}$$

4.4 Variance of the IS estimators and optimal proposal

In this section, we analyze the variance of the IS estimators, briefly discussing which is the optimal proposal in terms of variance minimization. Assume first that Z is known. Recalling that $I = \int_\Theta g(\boldsymbol{\theta})\bar{\pi}(\boldsymbol{\theta})d\boldsymbol{\theta}$, the variance of the IS estimator $\widehat{I}_M$ in Eq. (63) is $\widehat{V}_M = \mathrm{var}_q[\widehat{I}_M] = \frac{\sigma_q^2}{M}$, where

$$\begin{aligned}\sigma_q^2 &= \int_\Theta \left(\frac{g(\boldsymbol{\theta})\bar{\pi}(\boldsymbol{\theta})}{\bar{q}(\boldsymbol{\theta})}\right)^2 \bar{q}(\boldsymbol{\theta})d\boldsymbol{\theta} - I^2, \\ &= \int_\Theta \frac{\left(g(\boldsymbol{\theta})\bar{\pi}(\boldsymbol{\theta})\right)^2}{\bar{q}(\boldsymbol{\theta})}d\boldsymbol{\theta} - I^2 = \int_\Theta \frac{\left(g(\boldsymbol{\theta})\bar{\pi}(\boldsymbol{\theta}) - I\bar{q}(\boldsymbol{\theta})\right)^2}{\bar{q}(\boldsymbol{\theta})}d\boldsymbol{\theta},\end{aligned} \tag{77}$$

where we have used that

$$\begin{aligned}&\int_\Theta \frac{\left(g(\boldsymbol{\theta})\bar{\pi}(\boldsymbol{\theta}) - I\bar{q}(\boldsymbol{\theta})\right)^2}{\bar{q}(\boldsymbol{\theta})}d\boldsymbol{\theta} \\ &= \int_\Theta \frac{\left(g(\boldsymbol{\theta})\bar{\pi}(\boldsymbol{\theta})\right)^2 + I^2q^2(\boldsymbol{\theta}) - 2Ig(\boldsymbol{\theta})\bar{\pi}(\boldsymbol{\theta})\bar{q}(\boldsymbol{\theta})}{\bar{q}(\boldsymbol{\theta})}d\boldsymbol{\theta} \\ &= \int_\Theta \frac{\left(g(\boldsymbol{\theta})\bar{\pi}(\boldsymbol{\theta})\right)^2}{\bar{q}(\boldsymbol{\theta})}d\boldsymbol{\theta} + I^2 - 2I^2, \\ &= \int_\Theta \frac{\left(g(\boldsymbol{\theta})\bar{\pi}(\boldsymbol{\theta})\right)^2}{\bar{q}(\boldsymbol{\theta})}d\boldsymbol{\theta} - I^2.\end{aligned} \tag{78}$$

For a specific function $g(\boldsymbol{\theta})$, the optimal proposal PDF is

$$q_{opt}(\boldsymbol{\theta}) = \frac{|g(\boldsymbol{\theta})|\bar{\pi}(\boldsymbol{\theta})}{I} = \frac{g(\boldsymbol{\theta})\bar{\pi}(\boldsymbol{\theta})}{\int_\Theta g(\boldsymbol{\theta})\bar{\pi}(\boldsymbol{\theta})d\boldsymbol{\theta}} \propto |g(\boldsymbol{\theta})|\bar{\pi}(\boldsymbol{\theta}). \tag{79}$$

However, in many applications practitioners are not interested in estimating a specific integral I, but in approximating the measure of $\bar{\pi}$. In this case, an appropriate choice for the proposal is $\bar{q}(\boldsymbol{\theta}) = \bar{\pi}(\boldsymbol{\theta}) \propto \pi(\boldsymbol{\theta})$, which leads to $w_n = \frac{1}{Z}$ and $\bar{w}_n = \frac{1}{N}$ for all $n = 1, \ldots, N$, i.e., we come back to the original Monte Carlo scheme described in Section 2.1. Furthermore, the variance of the random variable $w(\boldsymbol{\theta}) = \frac{\pi(\boldsymbol{\theta})}{\bar{q}(\boldsymbol{\theta})}$, with $\boldsymbol{\theta} \sim \bar{q}(\boldsymbol{\theta})$, is given by

$$\begin{aligned}\mathrm{var}_q\left[w(\boldsymbol{\theta})\right] &= \int_\Theta \left(\frac{\pi(\boldsymbol{\theta})}{\bar{q}(\boldsymbol{\theta})}\right)^2 \bar{q}(\boldsymbol{\theta})d\boldsymbol{\theta} - \left(\int_\Theta \left(\frac{\pi(\boldsymbol{\theta})}{\bar{q}(\boldsymbol{\theta})}\right)\bar{q}(\boldsymbol{\theta})d\boldsymbol{\theta}\right)^2, \\ &= \int_\Theta \frac{\pi^2(\boldsymbol{\theta})}{\bar{q}(\boldsymbol{\theta})}d\boldsymbol{\theta} - Z^2, \\ &= \int_\Theta \frac{(\pi(\boldsymbol{\theta}) - Z\bar{q}(\boldsymbol{\theta}))^2}{\bar{q}(\boldsymbol{\theta})}d\boldsymbol{\theta} \\ &= Z^2 \int_\Theta \frac{(\bar{\pi}(\boldsymbol{\theta}) - \bar{q}(\boldsymbol{\theta}))^2}{\bar{q}(\boldsymbol{\theta})}d\boldsymbol{\theta} = Z^2\chi^2(\bar{\pi}, \bar{q}),\end{aligned} \tag{80}$$

where we have used $\bar{\pi}(\boldsymbol{\theta}) = \frac{1}{Z}\pi(\boldsymbol{\theta})$ in the last step of the derivation, and $\chi^2(\bar{\pi}, \bar{q}) = \int_\Theta \frac{(\bar{\pi}(\boldsymbol{\theta}) - \bar{q}(\boldsymbol{\theta}))^2}{\bar{q}(\boldsymbol{\theta})}d\boldsymbol{\theta}$ is the Pearson divergence between $\bar{\pi}$ and $\bar{q}$ [289]. Finally, the variance of $\widehat{Z} = \frac{1}{M}\sum_{m=1}^M w(\boldsymbol{\theta}^{(m)})$ is $\mathrm{var}_q[\widehat{Z}] = \frac{Z^2}{M^2}\chi^2(\bar{\pi}, \bar{q})$.

4.5 Effective sample size

Let us denote in this section the standard Monte Carlo estimator as

$$\bar{I}_M = \frac{1}{M}\sum_{m=1}^{M} g(\boldsymbol{\theta}^{(m)}), \tag{81}$$

where the samples $\boldsymbol{\theta}^{(1)},\dots,\boldsymbol{\theta}^{(M)}$ are directly drawn from $\bar{\pi}(\boldsymbol{\theta})$. Moreover, let us define the normalized IS weights,

$$\bar{w}_m = \frac{1}{M\widehat{Z}} w_m = \frac{w_m}{\sum_{i=1}^{M} w_i}, \quad m = 1,\dots,M, \tag{82}$$

then the self-normalized IS estimator can be written as $\widetilde{I}_M = \sum_{m=1}^{M} \bar{w}_m g(\boldsymbol{\theta}^{(m)})$. In general, the estimator $\widetilde{I}_M$ is less efficient than $\bar{I}_M$ in Eq. (81). In several applications of importance sampling, it is required to measure this loss in efficiency, when $\widetilde{I}_M$ is used instead of $\bar{I}_M$. The idea is to define the ESS as the ratio of the variances of the estimators [290],

$$\mathrm{ESS} = M\frac{\mathrm{var}_{\bar{\pi}}[\widehat{I}]}{\mathrm{var}_{\bar{q}}[\widetilde{I}]}. \tag{83}$$

The ESS value represents the number of samples from $\bar{\pi}$ required to obtain a Monte Carlo estimator $\widehat{I}$ with the same efficiency of the IS estimator $\widetilde{I}$, considering $\bar{q}$ as the proposal density. Finding a useful expression of ESS derived analytically from the theoretical definition above is not straightforward. Different derivations proceed by using several approximations and assumptions to yield an expression which is useful from a practical point of view [290, 291], [292, Chapter 11], [293, Chapter 4]. A well-known ESS approximation, widely used in the literature [106, 292, 293], is

$$\widehat{\mathrm{ESS}} = \frac{1}{\sum_{m=1}^{M} \bar{w}_m^2}. \tag{84}$$

An interesting property of the $\widehat{\mathrm{ESS}}$ in (84) is that $1 \le \widehat{\mathrm{ESS}} \le M$. Although Eq. (84) is often considered a suitable approximation of the theoretical ESS definition, its derivation [290, 293, 294],[289, Section 3] contains several approximations and strong assumptions [295]. As a consequence, $\widehat{\mathrm{ESS}}$ can differ substantially from the original definition of the ESS in many scenarios. In [295], different alternative approximations are discussed. For instance,

$$\widehat{\mathrm{ESS}} = \frac{1}{\max \bar{w}_m} \tag{85}$$

results again in $1 \le \widehat{\mathrm{ESS}} \le M$: the minimum is obtained when all the samples have zero weight except only one, whereas the maximum is reached when all the weights are equal to $\bar{w}_m = \frac{1}{M}$ [295]. Other related discussions and results can be found in [296–298].

4.6 Proper weighting

Although widely adopted, the standard IS weights in Eq. (62) are not the unique possibility. The definition of a *properly weighted sample* can be extended as suggested in [104, Section 14.2], [106, Section 2.5.4] and in [275]. More specifically, given a set of samples, they are properly weighted with respect to the target $\bar{\pi}$ if, for any integrable function g,

$$E_Q[\, w(\boldsymbol{\theta}^{(m)}) g(\boldsymbol{\theta}^{(m)})] = cE_{\bar{\pi}}[\, g(\boldsymbol{\theta}^{(m)})], \quad \forall m \in \{1,\dots,M\}, \tag{86}$$

where $c > 0$ is a constant value, independent from the index m, and the expectation of the left hand side is performed w.r.t. to the joint PDF of $w(\boldsymbol{\theta})$ and $\boldsymbol{\theta}$, i.e., $Q(\boldsymbol{\theta}, w)$. Thus, in order to obtain consistent estimators, one has to design a joint PDF $Q(\boldsymbol{\theta}, w)$ which guarantees that the restriction of Eq. (86) is fulfilled. An example is provided below.

4.6.1 *Proper weighting of a resampled particle*

Let us consider the particle approximation of the measure of $\bar{\pi}$ obtained by the IS approach drawing M IID particles $\boldsymbol{\theta}^{(m)} \sim q(\boldsymbol{\theta})$,

$$\widehat{\pi}(\boldsymbol{\theta}|\boldsymbol{\theta}_{1:M}) = \frac{1}{M\widehat{Z}}\sum_{m=1}^{M} w(\boldsymbol{\theta}^{(m)})\delta(\boldsymbol{\theta} - \boldsymbol{\theta}^{(m)}), \tag{87}$$

where $w(\boldsymbol{\theta}^{(m)}) = w_m = \frac{\pi(\boldsymbol{\theta}^{(m)})}{q(\boldsymbol{\theta}^{(m)})}$ and $\delta(\boldsymbol{\theta})$ is the Dirac delta function. Therefore, given the set of weighted samples $\{\boldsymbol{\theta}^{(m)}, w_m\}_{m=1}^{M}$, a resampled particle $\widetilde{\boldsymbol{\theta}}$ is a sample drawn from $\{\theta^{(m)}\}_{m=1}^{M}$ according to the probability mass $\frac{w_m}{M\widehat{Z}}$, i.e.,

$$\widetilde{\boldsymbol{\theta}} \sim \widehat{\pi}(\boldsymbol{\theta}|\boldsymbol{\theta}_{1:M}). \tag{88}$$

Let us denote the joint PDF $\widetilde{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}_{1:M}) = \widehat{\pi}(\boldsymbol{\theta}|\boldsymbol{\theta}_{1:M})\left[\prod_{i=1}^{M} q(\boldsymbol{\theta}^{(i)})\right]$. The marginal PDF $\widetilde{q}(\boldsymbol{\theta})$ of a resampled particle $\widetilde{\boldsymbol{\theta}}$, integrating out $\boldsymbol{\theta}_{1:M}$, i.e., $\widetilde{\boldsymbol{\theta}}' \sim \widetilde{q}(\boldsymbol{\theta})$, is

$$\begin{aligned}
\widetilde{q}(\boldsymbol{\theta}) &= \int_{\mathcal{X}^M} \widetilde{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}_{1:M}) d\boldsymbol{\theta}_{1:M} \\
&= \int_{\mathcal{X}^M} \widehat{\pi}(\boldsymbol{\theta}|\boldsymbol{\theta}_{1:M}) \left[\prod_{i=1}^{M} q(\boldsymbol{\theta}^{(i)})\right] d\boldsymbol{\theta}_{1:M}, \\
&= \pi(\boldsymbol{\theta}) \sum_{j=1}^{M} \left(\int_{\mathcal{X}^{M-1}} \frac{1}{M\widehat{Z}} \left[\prod_{\substack{i=1 \\ i\neq j}}^{M} q(\boldsymbol{\theta}^{(i)})\right] d\boldsymbol{\theta}_{\neg j} \right), \\
&= \pi(\boldsymbol{\theta}) \int_{\mathcal{X}^{M-1}} \frac{1}{\widehat{Z}} \left[\prod_{\substack{i=1 \\ i\neq j}}^{M} q(\boldsymbol{\theta}^{(i)})\right] d\boldsymbol{\theta}_{\neg j},
\end{aligned} \tag{89}$$

and the standard IS weight of a resampled particle $\widetilde{\boldsymbol{\theta}}$ is

$$w(\widetilde{\boldsymbol{\theta}}) = \frac{\pi(\widetilde{\boldsymbol{\theta}})}{\widetilde{q}(\widetilde{\boldsymbol{\theta}})}. \tag{90}$$

However, usually $\widetilde{q}(\boldsymbol{\theta})$ in Eq. (89) cannot be evaluated, and thus the standard IS weight cannot be computed [299–302] [273, App. C1] [232, App. B3]. An alternative is

to use Liu's definition of proper weighting in Eq. (86) and look for a weight function $\rho(\widetilde{\boldsymbol{\theta}}) = \rho(\widetilde{\boldsymbol{\theta}}, \boldsymbol{\theta}_{1:M})$ such that

$$E_{\widetilde{Q}(\boldsymbol{\theta},\boldsymbol{\theta}_{1:M})}[\,\rho(\boldsymbol{\theta},\boldsymbol{\theta}_{1:M})h(\boldsymbol{\theta})] = cE_{\bar{\pi}}[\,h(\boldsymbol{\theta})]\,, \tag{91}$$

where $\widetilde{Q}(\boldsymbol{\theta},\boldsymbol{\theta}_{1:M}) = \widehat{\pi}(\boldsymbol{\theta}|\boldsymbol{\theta}_{1:M})\left[\prod_{i=1}^{M} q(\boldsymbol{\theta}^{(i)})\right]$. A suitable choice is

$$\rho(\widetilde{\boldsymbol{\theta}},\boldsymbol{\theta}_{1:M}) = \widehat{Z} = \frac{1}{M}\sum_{i=1}^{M} w(\boldsymbol{\theta}^{(i)}), \tag{92}$$

since it holds in Eq. (91). For the proof and further discussions, see [301, 303, 304]. The proper weighting of a resampled particle is used in several Monte Carlo approaches, like the group IS described in Section 4.7.

4.7 Group importance sampling

Here, we use the results of the previous section to assign one single weighted sample to a set of weighted samples to summarize all the statistical information. Let us consider L sets of weighted samples, $\boldsymbol{\mathcal{S}}_1 = \{\boldsymbol{\theta}_1^{(m)}, w_{1,m}\}_{m=1}^{M_1}$, $\boldsymbol{\mathcal{S}}_2 = \{\boldsymbol{\theta}_2^{(m)}, w_{2,m}\}_{m=1}^{M_2}$, ..., $\boldsymbol{\mathcal{S}}_L = \{\boldsymbol{\theta}_L^{(m)}, w_{L,m}\}_{m=1}^{M_L}$, where $\boldsymbol{\theta}_\ell^{(m)} \sim q_\ell(\boldsymbol{\theta})$, i.e., a different proposal PDF can be used to generate each set $\boldsymbol{\mathcal{S}}_\ell$ and in general $M_i \neq M_j$ for all $i \neq j$, $i,j \in \{1,...,L\}$.

In different Monte Carlo applications, it is convenient (and often required) to compress the statistical information contained in all these sets by using a summary sample, $\widetilde{\boldsymbol{\theta}}_\ell$, and summary weight, W_ℓ, $\ell = 1,\ldots,L$, in such a way that

$$\widetilde{I}_L = \frac{1}{\sum_{j=1}^{L} W_j}\sum_{\ell=1}^{L} W_\ell g(\widetilde{\boldsymbol{\theta}}_\ell) \tag{93}$$

is still a consistent estimator of I, for a generic integrable function $g(\boldsymbol{\theta})$ [303]. Thus, although the compression is lossy, we still have a suitable particle approximation by the set of weighted samples $\{\widetilde{\boldsymbol{\theta}}_\ell, W_\ell\}_{\ell=1}^{L}$ of the target $\bar{\pi}$, as shown in the following. Let us denote the IS of the mth sample in the ℓth group as $w_{\ell,m} = \frac{\pi(\theta_{\ell,m})}{q_m(\theta_{\ell,m})}$, and $\widehat{Z}_\ell = \frac{1}{M_\ell}\sum_{m=1}^{M_\ell} w_{\ell,m}$. Then, it is possible to show that, with the choice

$$\widetilde{\boldsymbol{\theta}}_\ell \sim \frac{1}{\sum_{i=1}^{M_\ell} w_{i,\ell}}\sum_{m=1}^{M_\ell} w_{\ell,m}\delta\left(\boldsymbol{\theta} - \boldsymbol{\theta}_\ell^{(m)}\right) \tag{94}$$

and

$$W_\ell = M_\ell \widehat{Z}_\ell, \tag{95}$$

then $\widetilde{I}_L$ in Eq. (93) is a consistent estimator of I. Note that $\widetilde{\boldsymbol{\theta}}_\ell$ is a resampled particle within the ℓth group and W_ℓ takes into account the proper weight of a resampled particle, $\widehat{Z}_\ell$, and the number of samples in the ℓth set, M_ℓ. Let us consider the normalized weights $\bar{w}_{\ell,m} = \frac{w_{m,\ell}}{\sum_{i=1}^{M_\ell} w_{i,\ell}}$. Since $\bar{w}_{\ell,m} = \frac{w_{\ell,m}}{M_\ell \widehat{Z}_\ell} = \frac{w_{\ell,m}}{W_\ell}$, the unnormalized weight of the particle $\boldsymbol{\theta}_\ell^{(m)}$ can be expressed as $w_{\ell,m} = W_\ell \bar{w}_{\ell,m}$. This confirms that after a particle is resampled according to $\bar{w}_{\ell,m}$, in order to represent the ℓth group of M_ℓ weighted samples, it must be weighted as W_ℓ. The idea of a summary sample/weight has been implicitly used in different SMC schemes proposed in literature, for instance, for the communication among parallel particle filters [305–307], and in the particle island methods [297, 308, 309]. GIS also appears indirectly in particle filtering for model selection [304, 310, 311], and in the so-called Nested Sequential Monte Carlo techniques [302, 312, 313]. For further observations and applications of GIS see [301, 303].

4.8 Sequential importance sampling (SIS)

In this section, we describe the sequential importance sampling (SIS) scheme. In some applications, the parameters of interest $\boldsymbol{\theta}$ can be split in two disjoint groups, $\boldsymbol{\theta} = [\mathbf{x}, \boldsymbol{\lambda}]$, where the first one, $\mathbf{x}$, is related to a dynamical system (for instance, $\mathbf{x}$ can be the hidden state in a state-space model) and the other, $\boldsymbol{\lambda}$, is a static parameter (for instance, an unknown parameter of the model). The strategies for making inference about $\mathbf{x}$ and $\boldsymbol{\lambda}$ should take into account the different nature of the two parameters. In the previous sections, we have considered $\boldsymbol{\theta} = \boldsymbol{\lambda}$. In Section 5.2.2, we tackle the general case $\boldsymbol{\theta} = [\mathbf{x}, \boldsymbol{\lambda}]$, whereas here we address the case $\boldsymbol{\theta} = \mathbf{x}$. Namely, we assume that the variable of interest is a dynamical variable, i.e., $\boldsymbol{\theta} = \mathbf{x} = x_{1:D} = [x_1 \ldots, x_D]^\top$ with $x_d \in \mathbb{R}$ for the sake of simplicity, and the target can be factorized as

$$\bar{\pi}(\mathbf{x}) \propto \pi(\mathbf{x}) = \gamma_1(x_1)\prod_{d=2}^{D}\gamma_d(x_d|x_{d-1}). \tag{96}$$

Given a proposal $q(\mathbf{x}) = q_1(x_1)\prod_{d=2}^{D} q_d(x_d|x_{d-1})$, and a sample $\mathbf{x}^{(m)} = x_{1:D}^{(m)} \sim q(\mathbf{x})$ with $x_d^{(m)} \sim q_d(x_d|x_{d-1}^{(m)})$, we assign the importance weight

$$w(\mathbf{x}^{(m)}) = w_D^{(m)} = \frac{\pi(\mathbf{x}^{(m)})}{q(\mathbf{x}^{(m)})} = \frac{\gamma_1(x_1^{(m)})\gamma_2(x_2^{(m)}|x_1^{(m)})\cdots\gamma_D(x_D^{(m)}|x_{D-1}^{(m)})}{q_1(x_1^{(m)})q_2(x_2^{(m)}|x_1^{(m)})\cdots q_D(x_D^{(m)}|x_{D-1}^{(m)})}. \tag{97}$$

The weight above can be computed efficiently by following a recursive procedure to compute the importance weights: starting with $w_1^{(m)} = \frac{\pi(x_1^{(m)})}{q(x_1^{(m)})}$ and then obtaining

$$w_d^{(m)} = w_{d-1}^{(m)}\beta_d^{(m)} = \prod_{j=1}^{d}\beta_j^{(m)}, \qquad d = 1,\ldots,D, \tag{98}$$

where

$$\beta_1^{(m)} = w_1^{(m)} \tag{99a}$$

$$\beta_d^{(m)} = \frac{\gamma_d(x_d^{(m)}|x_{d-1}^{(m)})}{q_d(x_d^{(m)}|x_{d-1}^{(m)})}, \tag{99b}$$

4.8.1 Sequential importance resampling (SIR)

Sequential importance resampling, a.k.a., standard particle filtering, is a SIS scheme where resampling steps are incorporated during the recursion, as shown in Algorithm 19 [304, 314–316]. Resampling consists in drawing particles from the current cloud according to the normalized importance weights. In general, the resampling steps are applied only in certain iterations in order to avoid the path degeneration, taking into account an ESS approximation, such as $\widehat{\text{ESS}} = \frac{1}{\sum_{m=1}^{M}(\bar{w}_d^{(m)})^2}$ or $\widehat{\text{ESS}} = \frac{1}{\max \bar{w}_d^{(m)}}$ with $\bar{w}_d^{(m)} = \frac{w_d^{(m)}}{\sum_{i=1}^{M} w_d^{(i)}}$ [295]. If $\frac{1}{M}\widehat{\text{ESS}}$ is smaller than a pre-established threshold $\eta \in [0,1]$, the particles are resampled. Thus, the condition for the adaptive resampling can be expressed as $\widehat{\text{ESS}} < \eta M$. When $\eta = 1$, the resampling is applied at each iteration and in this case SIR is often called bootstrap particle filter [314, 315]. If $\eta = 0$, no resampling steps are applied, and we have the SIS method described above. Consider the Algorithm 19. Let us define

$$\widehat{Z}_d = \frac{1}{M}\sum_{m=1}^{M} w_d^{(m)} = \frac{1}{M}\sum_{m=1}^{M}\prod_{j=1}^{d}\beta_j^{(m)}, \tag{100}$$

where we have used the recursion for the weights in Alg. 19. Note that in Algorithm 19 we have employed a proper weighting for resampling particles (see Section 4.6.1 and [301]),

$$w_d^{(1)} = w_d^{(2)} = \ldots = w_d^{(M)} = \widehat{Z}_d. \tag{101}$$

In many works regarding particle filtering it is noted that the *unnormalized* weights of the resampled particles, $w_d^{(1)} = w_d^{(2)} = \ldots = w_d^{(M)}$, but a specific value is not given. if a different value $c \neq \widehat{Z}_d$ is employed, i.e., $w_d^{(1)} = \ldots = w_d^{(M)} = c$, the algorithm is still valid (if the resampling is applied considering *all* the particles), *but* the weight recursion loses some statistical meaning. In the case of the standard SIR scheme, i.e., when the resampling is performed considering *all* the M particles, the *normalized* weights of the resampled particles are

$$\bar{w}_d^{(1)} = \bar{w}_d^{(2)} = \ldots = \bar{w}_d^{(M)} = \frac{1}{M},$$

for any possible choice of c. Moreover, people usually employs a different marginal likelihood estimator

$$\widehat{Z}_d^{\mathbf{alt}} = \prod_{j=1}^{d}\left[\sum_{m=1}^{M}\bar{w}_{j-1}^{(m)}\beta_j^{(m)}\right], \tag{102}$$

which involves only the normalized weights, $\bar{w}_d^{(m)}$, instead of the unnormalized ones, $w_d^{(m)}$. Hence, this is a suitable and consistent estimator, in this scenario. However, the standard marginal likelihood estimator

$$\widehat{Z} = \widehat{Z}_D = \frac{1}{M}\sum_{m=1}^{M} w_D^{(M)} = \sum_{m=1}^{M}\prod_{j=1}^{D}\beta_j^{(m)}, \tag{103}$$

is consistent only if a proper weighting after resampling is used [301, 303, 304]. Moreover, if the resampling is performed considering *only* a subset of the particles of cardinality $R < M$ (instead over all the M particles), the proper weighting is strictly needed.

Algorithm 19 Sequential importance resampling (SIR).

1 **Initialization:** Choose the number of particles (M), the initial particles $x_0^{(m)}$ for $m = 1, \ldots, M$, an ESS approximation, and a constant value $\eta \in [0,1]$.

2 FOR $d = 1, \ldots, D$:

(a) **Propagation:** Draw $x_d^{(m)} \sim q_d(x_d|x_{d-1}^{(m)})$, for $m = 1, \ldots, M$.

(b) **Weighting:** Compute the weights

$$w_d^{(m)} = w_{d-1}^{(m)}\beta_d^{(m)} = \prod_{j=1}^{d}\beta_j^{(m)}, \qquad m = 1, \ldots, M, \tag{104}$$

where $\beta_d^{(m)} = \frac{\gamma_d(x_d^{(m)}|x_{d-1}^{(m)})}{q_d(x_d^{(m)}|x_{d-1}^{(m)})}$.

(c) IF $\widehat{ESS} < \eta M$ THEN:

i. **Resampling:** Resample M times within the set $\{x_{d-1}^{(m)}\}_{m=1}^{M}$ according to the probabilities $\bar{w}_d^{(m)} = \frac{w_d^{(m)}}{\sum_{j=1}^{M} w_d^{(j)}}$, obtaining M resampled particles $\{\bar{x}_d^{(m)}\}_{m=1}^{M}$. Then, set $x_d^{(m)} = \bar{x}_d^{(m)}$, for $m = 1, \ldots, M$.

ii. **Proper weighting:** Compute $\widehat{Z}_d = \frac{1}{M}\sum_{m=1}^{M} w_d^{(m)}$ and set $w_d^{(m)} = \widehat{Z}_d$ for $m = 1, \ldots, M$.

3 Return $\{\mathbf{x}^{(m)} = x_{1:D}^{(m)}, w^{(m)} = w_D^{(m)}\}_{m=1}^{M}$.

4.8.2 Conditional particle filter

The conditional particle filter (CPF) is a modification of the particle filter algorithm which takes a reference state sequence $\mathbf{x}^* = x_{1:D}^*$ as input [317, 318]. Namely, the CPF is a standard particle filter (e.g., the SIR in Algorithm 19) setting as the first particle $x_{1:D}^{(1)} = x_{1:D}^*$, the reference path. Hence, the implementation of the CPF algorithm is exactly like a standard particle filter, except for the following two points:

1. The first path $x_{1:D}^{(1)}$ is not sampled, i.e., it is not randomly generated but fixed in advance. Indeed, each component of the first path $x_{1:D}^{(1)}$ is copied from the reference path $x_{1:D}^{*}$.
2. In the resampling step, the first particle is guaranteed to survive.

Considering a resampling step at each iteration (just for the sake of simplicity), the CPF is outlined in Algorithm 20. It is important to remark that the procedure (a) picking a reference path $\mathbf{x}^*$, (b) running the CPF, (c) picking a path $\mathbf{x}' = x'_{1:D}$ by resampling once with probabilities proportional to the final weights w_m for $m = 1, \ldots, M$, and (d) repeating from (a) considering $\mathbf{x}^* = \mathbf{x}'$, leaves invariant the target density $\bar{\pi}(\mathbf{x})$. Indeed, this procedure virtually coincides with the Ensemble MCMC method that will be described in Section 5.1.4 (see [232, Appendix C] for a proof). For this reason, the CPF method is often applied within sophisticated MCMC techniques, called Particle Gibbs algorithms (see Section 5.2.3 for further details). The CPF provides a particle approximation of the target measure given the reference path $\mathbf{x}'$, i.e.,

$$\widehat{\pi}(\mathbf{x}|\mathbf{x}') = \sum_{m=1}^{M} \bar{w}_D^{(m)} \delta(\mathbf{x} - \mathbf{x}^{(m)}). \tag{105}$$

Finally, note that we have considered a CPF method that is slightly different from the technique proposed in [317]. Indeed, here we have described the CPF version given in [318, 319].

Algorithm 20 Conditional particle filter (CPF).

1 **Initialization:** Determine the reference path $x_{1:D}^{*}$. Choose the number of particles (M) and other $M-1$ initial particles $x_0^{(m)}$ for $m = 2, \ldots, M$.

2 FOR $d = 1, \ldots, D$:

(a) **Propagation:** Set $x_d^{(1)} = x_d^*$ and draw $x_d^{(m)} \sim q_d(x_d|x_{d-1}^{(m)})$, for $m = 2, \ldots, M$.

(b) **Weighting:** Compute the weights

$$w_d^{(m)} = w_{d-1}^{(m)} \beta_d^{(m)} = \prod_{j=1}^{d} \beta_j^{(m)}, \qquad m = 1, \ldots, M, \tag{106}$$

where $\beta_d^{(m)} = \frac{\gamma_d(x_d^{(m)}|x_{d-1}^{(m)})}{q_d(x_d^{(m)}|x_{d-1}^{(m)})}$.

(c) **Conditional Resampling:**

i. Set $\bar{x}_d^{(1)} = x_d^{(1)}$.

ii. Resample $M-1$ times within the set of M samples $\{x_{d-1}^{(m)}\}_{m=1}^{M}$ according to the probabilities $\bar{w}_d^{(m)} = \frac{w_d^{(m)}}{\sum_{j=1}^{M} w_d^{(j)}}$, $m = 1, \ldots, M$, obtaining $M-1$ resampled particles $\{\bar{x}_d^{(m)}\}_{m=2}^{M}$.

iii. Set $x_d^{(m)} = \bar{x}_d^{(m)}$, for $m = 1, \ldots, M$.

(d) **Proper weighting:** Compute $\widehat{Z}_d = \frac{1}{M} \sum_{m=1}^{M} w_d^{(m)}$ and set $w_d^{(m)} = \widehat{Z}_d$ for all $m = 1, \ldots, M$ [301,303]

3 Return $\{\mathbf{x}^{(m)} = x_{1:D}^{(m)}, w^{(m)} = w_D^{(m)}\}_{m=1}^{M}$.

5 MC-within-MCMC methods

In this section, we describe several MCMC techniques that use other inner MC estimators at each iteration[24]. The resulting hybrid methods are still MCMC algorithms, since they rely on a Markov chain to sample from the target PDF, but they require these inner MC techniques for different reasons. They can be divided into two classes. The methods in the first class (see Sections 5.1 and 5.2) use IS and a resampling step to generate better candidates for the MH acceptance test. The methods in the second class need some inner MC algorithm (either IS or MCMC) to obtain unbiased estimators of the likelihood function (see Sections 5.3 and 5.4). There is a connection between these two classes of algorithms which is apparent between the methods in Sections 5.2.2 and 5.3. We also split the first class in two sub-families. In the first one (Section 5.1), we describe the MCMC techniques that propose multiple candidates at each iteration and work in a batch way (i.e., directly in the entire space of $\boldsymbol{\theta}$). The methods contained in the second sub-family (Section 5.2) also generate several candidates at each iteration, but they assume that a factorization of the target density is available. This assumption allows the sequential generation of the candidates (via particle filtering, for instance).

[24] Note that we have already described an MC-within-MCMC method in Section 3.1.4: the MH-within-Gibbs algorithm.

5.1 MCMC with multiple candidates for the estimation of a static parameter

In the MH algorithm, at each iteration, one new sample, $\boldsymbol{\theta}'$, is generated and tested w.r.t. the previous state, $\boldsymbol{\theta}^{(t-1)}$, by using the acceptance probability $\alpha_t = \alpha(\boldsymbol{\theta}', \boldsymbol{\theta}^{(t-1)})$. Other generalized MH schemes generate several candidates at each iteration to be tested as the new possible state with the aim of increasing the acceptance rate of candidate samples. In all these schemes, an extended acceptance probability, α_t, has to be properly designed in order to guarantee the ergodicity of the chain. Below we describe the most important examples of this kind of generalized MH algorithms [122]. Furthermore, most of these

techniques use an AIS approximation of the target density (see Section 4.2) in order to improve the proposal procedure within an MH-type algorithm. Namely, they build an IS approximation adaptively and then draw one sample from this approximation (resampling step). Finally, the selected sample is compared with the previous state of the chain, $\boldsymbol{\theta}^{(t-1)}$, according to a suitable generalized acceptance probability α_t.

5.1.1 Multiple-try Metropolis (MTM)

The *multiple-try Metropolis* (MTM) algorithms are examples of this class of methods [122, 320–324]. In this case, N samples (a.k.a. "tries" or "candidates") are drawn from the proposal PDF, one of them is selected according to some suitable weights, and the selected candidate is finally accepted or rejected according to a generalized probability function α_t.

The standard MTM scheme is shown in Algorithm 21. For the sake of simplicity, we have considered the use of the standard importance weights $w(\boldsymbol{\theta}) = \frac{\pi(\boldsymbol{\theta})}{q(\boldsymbol{\theta})}$ (see Section 4.2), but other more sophisticated alternatives are also possible [320, 321, 325]. In its general form, when the proposal depends on the previous state of the chain, $q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t-1)})$, MTM requires the generation of $N-1$ auxiliary samples, $\mathbf{v}^{(1)}, \ldots, \mathbf{v}^{(N-1)}$, which are employed in the computation of the acceptance probability α_t. These samples are required in order to guarantee the ergodicity of the underlying Markov chain. Indeed, it can be proved the resulting MTM kernel satisfies the detailed balance condition, implying that the chain is reversible.

Note that, for $N = 1$, we have $\widetilde{\boldsymbol{\theta}}^{(j)} = \widetilde{\boldsymbol{\theta}}^{(1)}$, $\mathbf{v}^{(1)} = \boldsymbol{\theta}^{(t-1)}$, and the acceptance probability of the MTM method becomes

$$\begin{aligned}\alpha_t = \alpha(\widetilde{\boldsymbol{\theta}}^{(1)}, \boldsymbol{\theta}^{(t-1)}) &= \min\left[1, \frac{w(\widetilde{\boldsymbol{\theta}}^{(1)})}{w(\mathbf{v}^{(1)})}\right], \\ &= \min\left[1, \frac{w(\widetilde{\boldsymbol{\theta}}^{(1)})}{w(\boldsymbol{\theta}^{(t-1)})}\right], \\ &= \min\left[1, \frac{\pi(\widetilde{\boldsymbol{\theta}}^{(1)}) q(\boldsymbol{\theta}^{(t-1)}|\widetilde{\boldsymbol{\theta}}^{(1)})}{\pi(\boldsymbol{\theta}^{(t-1)}) q(\widetilde{\boldsymbol{\theta}}^{(1)}|\boldsymbol{\theta}^{(t-1)})}\right],\end{aligned} \tag{111}$$

which is the acceptance probability of the classical MH technique shown in Algorithm 3 with $\widetilde{\boldsymbol{\theta}}^{(1)}$ playing the role of $\boldsymbol{\theta}'$. Several variants of the standard MTM method shown in Algorithm 21 have been studied. For instance, some authors have considered the use of correlated tries or different proposal PDFs [231, 321].

5.1.2 Independent multiple-try Metropolis (I-MTM) schemes

The MTM method described in Algorithm 21 requires drawing $2N-1$ samples at each iteration (N candidates and $N-1$ auxiliary samples) and only $N-1$ of those samples are used in the acceptance probability function. The generation of the auxiliary points,

$$\mathbf{v}^{(1)}, \ldots, \mathbf{v}^{(j-1)}, \mathbf{v}^{(j+1)}, \ldots, \mathbf{v}^{(N)} \sim q(\boldsymbol{\theta}|\widetilde{\boldsymbol{\theta}}^{(j)}),$$

can be avoided if the proposal PDF is independent from the previous state, i.e., $q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t-1)}) = q(\boldsymbol{\theta})$. In this case, we should draw $N-1$ samples again from $q(\boldsymbol{\theta})$ at step 2(d) of Algorithm 21. However, since we have already drawn N samples from $q(\boldsymbol{\theta})$ at step 2(a) of Algorithm 21, we can set

Algorithm 21 Multiple Try Metropolis (MTM) method.

1 **Initialization:** Choose a proposal function $q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t-1)})$, an initial state $\boldsymbol{\theta}^{(0)}$, the total number of iterations (T), and the number of tries (N).

2 FOR $t = 1, \ldots, T$:

(a) Draw $\widetilde{\boldsymbol{\theta}}^{(1)}, \widetilde{\boldsymbol{\theta}}^{(2)}, \ldots, \widetilde{\boldsymbol{\theta}}^{(N)} \sim q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t-1)})$.

(b) Compute the importance weights:

$$w(\widetilde{\boldsymbol{\theta}}^{(n)}) = \frac{\pi(\widetilde{\boldsymbol{\theta}}^{(n)})}{q(\widetilde{\boldsymbol{\theta}}^{(n)}|\boldsymbol{\theta}^{(t-1)})}, \quad \text{for} \quad n = 1, \ldots, N. \tag{107}$$

(c) Select one sample $\widetilde{\boldsymbol{\theta}}^{(j)} \in \{\widetilde{\boldsymbol{\theta}}^{(1)}, \ldots, \widetilde{\boldsymbol{\theta}}^{(N)}\}$, according to the probability mass function

$$\bar{w}_n = \frac{w(\widetilde{\boldsymbol{\theta}}^{(n)})}{\sum_{i=1}^{N} w(\widetilde{\boldsymbol{\theta}}^{(i)})}. \tag{108}$$

(d) Draw $N-1$ auxiliary samples $\mathbf{v}^{(1)}, \ldots, \mathbf{v}^{(j-1)}, \mathbf{v}^{(j+1)}, \ldots, \mathbf{v}^{(N)} \sim q(\boldsymbol{\theta}|\widetilde{\boldsymbol{\theta}}^{(j)})$, and set $\mathbf{v}^{(j)} = \boldsymbol{\theta}^{(t-1)}$.

(e) Compute the weights of the auxiliary samples,

$$w(\mathbf{v}^{(n)}) = \frac{\pi(\mathbf{v}^{(n)})}{q(\mathbf{v}^{(n)}|\widetilde{\boldsymbol{\theta}}^{(j)})}, \quad \text{for} \quad n = 1, \ldots, N, \tag{109}$$

and the acceptance probability of $\widetilde{\boldsymbol{\theta}}^{(j)}$:

$$\alpha_t \equiv \alpha(\widetilde{\boldsymbol{\theta}}^{(j)}, \boldsymbol{\theta}^{(t-1)}) = \min\left[1, \frac{\sum_{n=1}^{N} w(\widetilde{\boldsymbol{\theta}}^{(n)})}{\sum_{n=1}^{N} w(\mathbf{v}^{(n)})}\right], \tag{110}$$

(f) Draw $u \sim \mathcal{U}([0,1))$. If $u \leq \alpha_t$, accept $\widetilde{\boldsymbol{\theta}}^{(j)}$ and set $\boldsymbol{\theta}^{(t)} = \widetilde{\boldsymbol{\theta}}^{(j)}$. Otherwise (i.e., if $u > \alpha_t$), reject $\widetilde{\boldsymbol{\theta}}^{(j)}$ and set $\boldsymbol{\theta}^{(t)} = \boldsymbol{\theta}^{(t-1)}$.

3 Approximate the integral in Eq. (2) using Eq. (14).

$$\begin{aligned}\mathbf{v}^{(1)} &= \widetilde{\boldsymbol{\theta}}^{(1)}, \ldots, \mathbf{v}^{(j-1)} = \widetilde{\boldsymbol{\theta}}^{(j-1)}, \mathbf{v}^{(j)} = \boldsymbol{\theta}^{(t-1)}, \mathbf{v}^{(j+1)} \\ &= \widetilde{\boldsymbol{\theta}}^{(j+1)} \ldots \mathbf{v}^{(N-1)} = \widetilde{\boldsymbol{\theta}}^{(N)},\end{aligned} \tag{112}$$

without jeopardizing the ergodicity of the chain. Hence, we can avoid step 2(d) in Algorithm 21, and the acceptance probability becomes

$$\alpha_t \equiv \alpha(\widetilde{\boldsymbol{\theta}}^{(j)}, \boldsymbol{\theta}^{(t-1)}) = \min\left[1, \frac{w(\widetilde{\boldsymbol{\theta}}^{(j)}) + \sum_{n=1, n\neq j}^{N} w(\widetilde{\boldsymbol{\theta}}^{(n)})}{w(\boldsymbol{\theta}^{(t-1)}) + \sum_{n=1, n\neq j}^{N} w(\widetilde{\boldsymbol{\theta}}^{(n)})}\right]. \tag{113}$$

The I-MTM technique is shown in Algorithm 22. Note that Eq. (113) can be expressed alternatively as

$$\alpha(\widetilde{\boldsymbol{\theta}}^{(j)}, \boldsymbol{\theta}^{(t-1)}) = \min\left[1, \frac{\widehat{Z}_1}{\widehat{Z}_2}\right], \tag{114}$$

where we have denoted

$$\widehat{Z}_1 = \frac{1}{N}\sum_{n=1}^{N} w(\widetilde{\boldsymbol{\theta}}^{(n)}), \tag{115a}$$

$$\widehat{Z}_2 = \frac{1}{N}\left[w(\boldsymbol{\theta}^{(t-1)}) + \sum_{\substack{n=1 \\ n\neq j}}^{N} w(\widetilde{\boldsymbol{\theta}}^{(n)})\right]. \tag{115b}$$

From the IS theory (see Section 4), we know that both $\widehat{Z}_1$ and $\widehat{Z}_2$ are unbiased estimators of the normalizing constant (a.k.a, partition function or marginal likelihood) of the target, Z. Moreover, Eq. (115b) suggests that other more sophisticated unbiased estimators of Z could be used without jeopardizing the ergodicity of the I-MTM algorithm. For instance, instead of recycling the samples generated in the same iteration as the auxiliary points in Eq. (112), we could reuse samples generated in the previous iteration. This alternative version of the I-MTM method (I-MTM2) is described in Algorithm 23. The I-MTM2 method is related to the well-known particle Metropolis-Hastings (PMH) algorithm [317] (see [303, 326] for further considerations). The ergodicity of I-MTM2 is thus ensured, since it can be interpreted as a PMH algorithm where no resampling is applied (implying that the resulting candidates are independent from each other).

5.1.3 Group Metropolis sampling

The auxiliary weighted samples in the previous I-MTM schemes (i.e., the $N-1$ samples drawn at each iteration which are not selected for comparison with the previous state $\boldsymbol{\theta}^{(t-1)}$) can be recycled in order to provide a final Monte Carlo estimator [301, 303]. This leads to Algorithm 24, known as group Metropolis sampling (GMS). GMS can be considered an extension (for $N > 1$ candidates) of the algorithm described in [327], where the authors show how to recycle and include the samples rejected in one run of a standard MH method (i.e., $N = 1$ in this case) into a unique consistent estimator. GMS yields a sequence of sets of weighted samples, $\boldsymbol{\mathcal{S}}_t = \{\boldsymbol{\theta}^{(t,n)}, \rho^{(t,n)}\}_{n=1}^{N}$ for $t = 1, \ldots, T$, where we have denoted as $\rho^{(t,n)}$ the importance weights assigned to the samples $\boldsymbol{\theta}^{(t,n)}$. All the samples are then employed to obtain a joint particle approximation of the target. This approximation can then be used to compute any desired moment of the target PDF as

$$\begin{aligned}\widehat{I}_{N(T-T_b)} &= \frac{1}{T-T_b}\sum_{t=T_b+1}^{T}\sum_{n=1}^{N}\frac{\rho^{(t,n)}}{\sum_{i=1}^{N}\rho^{(t,i)}} g(\boldsymbol{\theta}^{(t,n)}) \\ &= \frac{1}{T-T_b}\sum_{t=T_b+1}^{T}\widehat{I}_N^{(t)},\end{aligned} \tag{121}$$

Algorithm 22 Independent Multiple Try Metropolis (I-MTM).

1 **Initialization:** Choose a proposal function $q(\boldsymbol{\theta})$, an initial state $\boldsymbol{\theta}^{(0)}$, the total number of iterations (T), and the number of tries (N).

2 FOR $t = 1, \ldots, T$:

(a) Draw $\widetilde{\boldsymbol{\theta}}^{(1)}, \widetilde{\boldsymbol{\theta}}^{(2)}, \ldots, \widetilde{\boldsymbol{\theta}}^{(N)} \sim q(\boldsymbol{\theta})$.

(b) Compute the importance weights:

$$w(\widetilde{\boldsymbol{\theta}}^{(n)}) = \frac{\pi(\widetilde{\boldsymbol{\theta}}^{(n)})}{q(\widetilde{\boldsymbol{\theta}}^{(n)})}, \quad \text{for} \quad n = 1, \ldots, N. \tag{116}$$

(c) Select one sample $\widetilde{\boldsymbol{\theta}}^{(j)} \in \{\widetilde{\boldsymbol{\theta}}^{(1)}, \ldots, \widetilde{\boldsymbol{\theta}}^{(N)}\}$, according to the following probability mass function:

$$\bar{w}_n = \frac{w(\widetilde{\boldsymbol{\theta}}^{(n)})}{\sum_{i=1}^{N} w(\widetilde{\boldsymbol{\theta}}^{(i)})}.$$

(d) Compute the acceptance probability of $\widetilde{\boldsymbol{\theta}}^{(j)}$:

$$\alpha_t \equiv \alpha(\widetilde{\boldsymbol{\theta}}^{(j)}, \boldsymbol{\theta}^{(t-1)}) = \min\left[1, \frac{w(\widetilde{\boldsymbol{\theta}}^{(j)}) + \sum_{n=1, n\neq j}^{N} w(\widetilde{\boldsymbol{\theta}}^{(n)})}{w(\boldsymbol{\theta}^{(t-1)}) + \sum_{n=1, n\neq j}^{N} w(\widetilde{\boldsymbol{\theta}}^{(n)})}\right], \tag{117a}$$

$$= \min\left[1, \frac{\widehat{Z}_1}{\widehat{Z}_2}\right], \tag{117b}$$

where $\widehat{Z}_1$ and $\widehat{Z}_2$ are given by (115a) and (115b), respectively.

(e) Draw $u \sim \boldsymbol{\mathcal{U}}([0,1))$. If $u \leq \alpha_t$, accept $\widetilde{\boldsymbol{\theta}}^{(j)}$ and set $\boldsymbol{\theta}^{(t)} = \widetilde{\boldsymbol{\theta}}^{(j)}$. Otherwise (i.e., if $u > \alpha_t$), reject $\widetilde{\boldsymbol{\theta}}^{(j)}$ and set $\boldsymbol{\theta}^{(t)} = \boldsymbol{\theta}^{(t-1)}$.

3 Approximate the integral in Eq. (2) using Eq. (14).

Algorithm 23 Alternative version of I-MTM method (I-MTM2)

1 **Initialization:** Choose a proposal function $q(\boldsymbol{\theta})$, an initial state $\boldsymbol{\theta}^{(0)}$, an initial estimate of the normalizing constant of the target $\widehat{Z}_0$, the total number of iterations (T), and the number of tries (N).

2 FOR $t = 1, \ldots, T$:

(a) Draw $\widetilde{\boldsymbol{\theta}}^{(1)}, \widetilde{\boldsymbol{\theta}}^{(2)}, \ldots, \widetilde{\boldsymbol{\theta}}^{(N)} \sim q(\boldsymbol{\theta})$.

(b) Compute the importance weights:

$$w(\widetilde{\boldsymbol{\theta}}^{(n)}) = \frac{\pi(\widetilde{\boldsymbol{\theta}}^{(n)})}{q(\widetilde{\boldsymbol{\theta}}^{(n)})}, \quad \text{for} \quad n = 1, \ldots, N. \tag{118}$$

(c) Select one sample $\widetilde{\boldsymbol{\theta}}^{(j)} \in \{\widetilde{\boldsymbol{\theta}}^{(1)}, \ldots, \widetilde{\boldsymbol{\theta}}^{(N)}\}$, according to the following probability mass function:

$$\bar{w}_n = \frac{1}{N\widehat{Z}'} w(\widetilde{\boldsymbol{\theta}}^{(n)}), \tag{119a}$$

$$\widehat{Z}' = \frac{1}{N}\sum_{i=1}^{N} w(\widetilde{\boldsymbol{\theta}}^{(i)}). \tag{119b}$$

(d) Compute the acceptance probability of $\widetilde{\boldsymbol{\theta}}^{(j)}$:

$$\alpha_t \equiv \alpha(\widetilde{\boldsymbol{\theta}}^{(j)}, \boldsymbol{\theta}^{(t-1)}) = \min\left[1, \frac{\widehat{Z}'}{\widehat{Z}_{t-1}}\right]. \tag{120}$$

(e) Draw $u \sim \mathcal{U}([0,1))$. If $u \leq \alpha_t$, accept $\widetilde{\boldsymbol{\theta}}^{(j)}$, setting $\boldsymbol{\theta}^{(t)} = \widetilde{\boldsymbol{\theta}}^{(j)}$ and $\widehat{Z}_t = \widehat{Z}'$. Otherwise (i.e., if $u > \alpha_t$), reject $\widetilde{\boldsymbol{\theta}}^{(j)}$, setting $\boldsymbol{\theta}^{(t)} = \boldsymbol{\theta}^{(t-1)}$ and $\widehat{Z}_t = \widehat{Z}_{t-1}$.

3 Approximate the integral in Eq. (2) using Eq. (14).

where $\widehat{I}_N^{(t)} = \sum_{n=1}^{N} \frac{\rho^{(t,n)}}{\sum_{i=1}^{N}\rho^{(t,i)}} g(\boldsymbol{\theta}^{(t,n)})$ and T_b is the burn-in period, as usual.

GMS is related to the MTM schemes previously described [321, 326], even though no resampling steps are applied at each iteration in GMS. Nevertheless, we can recover an MTM chain from the GMS output by applying one resampling step when $\boldsymbol{\mathcal{S}}_t \neq \boldsymbol{\mathcal{S}}_{t-1}$, i.e.,

$$\boldsymbol{\theta}^{(t)} = \begin{cases} \boldsymbol{\theta}' \sim \sum_{n=1}^{N} \frac{\rho^{(t,n)}}{\sum_{i=1}^{N}\rho^{(t,i)}} \delta(\boldsymbol{\theta} - \boldsymbol{\theta}^{(t,n)}), & \text{if} \quad \boldsymbol{\mathcal{S}}_t \neq \boldsymbol{\mathcal{S}}_{t-1}, \\ \boldsymbol{\theta}^{(t-1)}, & \text{if} \quad \boldsymbol{\mathcal{S}}_t = \boldsymbol{\mathcal{S}}_{t-1}, \end{cases} \tag{125}$$

for $t = 1, \ldots, T$. More specifically, $\{\boldsymbol{\theta}^{(t)}\}_{t=1}^{T}$ is equivalent to the Markov chain obtained in one run of the I-MTM2 technique shown in Algorithm 23. GMS can also be interpreted as an iterative importance sampling scheme, where an IS approximation using N samples is built at each iteration and compared with the previous IS approximation. This procedure is iterated T times, and all the accepted IS estimators, $\widehat{I}_N^{(t)}$, are finally combined to provide a unique global approximation using $N(T - T_b)$ samples. Note that the temporal combination of the IS estimators is obtained dynamically by the random repetitions due to the rejections in the MH test. Therefore, the complete procedure for weighting the samples generated by GMS can be interpreted as the composition of two weighting schemes: (a) by an importance sampling approach building $\{\rho^{(t,n)}\}_{n=1}^{N}$ and (b) by the possible random repetitions due to the rejections in the MH test.

Algorithm 24 Group Metropolis Sampling (GMS).

1 **Initialization:** Choose a proposal function $q(\boldsymbol{\theta})$, an initial state $\boldsymbol{\theta}^{(0)}$, an initial estimate of the normalizing constant of the target $\widehat{Z}_0$, the total number of iterations (T), and the number of tries (N).

2 FOR $t = 1, \ldots, T$:

(a) Draw $\widetilde{\boldsymbol{\theta}}^{(1)}, \widetilde{\boldsymbol{\theta}}^{(2)}, \ldots, \widetilde{\boldsymbol{\theta}}^{(N)} \sim q(\boldsymbol{\theta})$.

(b) Compute the importance weights:

$$w(\widetilde{\boldsymbol{\theta}}^{(n)}) = \frac{\pi(\widetilde{\boldsymbol{\theta}}^{(n)})}{q(\widetilde{\boldsymbol{\theta}}^{(n)})}, \quad \text{for} \quad n = 1, \ldots, N. \tag{122}$$

Define $\boldsymbol{\mathcal{S}}' = \{\widetilde{\boldsymbol{\theta}}^{(n)}, w(\widetilde{\boldsymbol{\theta}}^{(n)})\}_{n=1}^{N}$ and compute $\widehat{Z}' = \frac{1}{N}\sum_{n=1}^{N} w(\widetilde{\boldsymbol{\theta}}^{(n)})$.

(c) Compute the acceptance probability:

$$\alpha_t \equiv \alpha(\boldsymbol{\mathcal{S}}', \boldsymbol{\mathcal{S}}_{t-1}) = \min\left[1, \frac{\widehat{Z}'}{\widehat{Z}_{t-1}}\right]. \tag{123}$$

(d) Draw $u \sim \mathcal{U}([0,1))$. If $u \leq \alpha_t$, accept $\boldsymbol{\mathcal{S}}'$, setting $\widehat{Z}_t = \widehat{Z}'$ and

$$\boldsymbol{\mathcal{S}}_t = \left\{\boldsymbol{\theta}^{(t,n)} = \widetilde{\boldsymbol{\theta}}^{(n)}, \rho^{(t,n)} = w(\widetilde{\boldsymbol{\theta}}^{(n)})\right\}_{n=1}^{N}. \tag{124}$$

Otherwise (i.e., if $u > \alpha_t$), reject $\boldsymbol{\mathcal{S}}'$, setting $\widehat{Z}_t = \widehat{Z}_{t-1}$ and $\boldsymbol{\mathcal{S}}_t = \boldsymbol{\mathcal{S}}_{t-1}$.

3 Approximate the integral in Eq. (2) using Eq. (121).

5.1.4 Ensemble MCMC

Another alternative procedure, called ensemble MCMC and involving several tries at each iteration, has been proposed in [232, 241, 328]. In this section, we present the simplest version, which employs a proposal PDF,

$q(\boldsymbol{\theta})$, independent of the previous state of the chain. At each iteration, the ensemble MCMC method (summarized in Algorithm 25) generates N new samples, $\widetilde{\boldsymbol{\theta}}^{(1)}, \widetilde{\boldsymbol{\theta}}^{(2)}, \ldots, \widetilde{\boldsymbol{\theta}}^{(N)}$ and then draws the new state $\boldsymbol{\theta}^{(t)}$ from a set of $N+1$ samples that includes the previous state, $\{\widetilde{\boldsymbol{\theta}}^{(1)}, \ldots, \widetilde{\boldsymbol{\theta}}^{(N)}, \widetilde{\boldsymbol{\theta}}^{(N+1)} = \boldsymbol{\theta}^{(t-1)}\}$, according to the following probabilities:

$$\bar{w}_j = \frac{w(\widetilde{\boldsymbol{\theta}}^{(j)})}{\sum_{i=1}^{N} w(\widetilde{\boldsymbol{\theta}}^{(i)}) + w(\boldsymbol{\theta}^{(t-1)})}, \qquad j = 1, \ldots, N+1, \tag{126}$$

where $w(\boldsymbol{\theta}) = \frac{\pi(\boldsymbol{\theta})}{q(\boldsymbol{\theta})}$ denotes again the standard IS weight. Note that, for $N = 1$, Eq. (126) becomes

$$\begin{aligned} \bar{w}_j &= \frac{w(\widetilde{\boldsymbol{\theta}}^{(j)})}{w(\widetilde{\boldsymbol{\theta}}^{(i)}) + w(\boldsymbol{\theta}^{(t-1)})} \\ &= \frac{\frac{\pi(\widetilde{\boldsymbol{\theta}}^{(j)})}{q(\widetilde{\boldsymbol{\theta}}^{(j)})}}{\frac{\pi(\widetilde{\boldsymbol{\theta}}^{(j)})}{q(\boldsymbol{\theta}^{(j)})} + \frac{\pi(\boldsymbol{\theta}^{(t-1)})}{q(\boldsymbol{\theta}^{(t-1)})}} \\ &= \frac{\pi(\widetilde{\boldsymbol{\theta}}^{(j)}) q(\boldsymbol{\theta}^{(t-1)})}{\pi(\widetilde{\boldsymbol{\theta}}^{(j)}) q(\boldsymbol{\theta}^{(t-1)}) + \pi(\boldsymbol{\theta}^{(t-1)}) q(\widetilde{\boldsymbol{\theta}}^{(j)})}, \end{aligned} \tag{127}$$

which is Barker's acceptance function, as given by Eq. (12b), with an independent proposal density and $\widetilde{\boldsymbol{\theta}}^{(j)}$ playing the role of $\boldsymbol{\theta}'$ in (12b). See [232, Appendix C] for a proof of the ergodicity.

Algorithm 25 Ensemble MCMC with an independent proposal PDF.

1 **Initialization:** Choose a proposal function $q(\boldsymbol{\theta})$, an initial state $\boldsymbol{\theta}^{(0)}$, the total number of iterations (T), and the number of tries (N).

2 FOR $t = 1, \ldots, T$:

(a) Draw $\widetilde{\boldsymbol{\theta}}^{(1)}, \widetilde{\boldsymbol{\theta}}^{(2)}, \ldots, \widetilde{\boldsymbol{\theta}}^{(N)} \sim q(\boldsymbol{\theta})$.

(b) Compute the importance weights:

$$w(\widetilde{\boldsymbol{\theta}}^{(n)}) = \frac{\pi(\widetilde{\boldsymbol{\theta}}^{(n)})}{q(\widetilde{\boldsymbol{\theta}}^{(n)})}, \quad \text{for} \quad n = 1, \ldots, N. \tag{128}$$

(c) Select one sample $\widetilde{\boldsymbol{\theta}}^{(j)} \in \{\widetilde{\boldsymbol{\theta}}^{(1)}, \ldots, \widetilde{\boldsymbol{\theta}}^{(N)}, \widetilde{\boldsymbol{\theta}}^{(N+1)} = \boldsymbol{\theta}^{(t-1)}\}$, according to the probability mass function

$$\bar{w}_j = \frac{w(\widetilde{\boldsymbol{\theta}}^{(j)})}{\sum_{i=1}^{N} w(\widetilde{\boldsymbol{\theta}}^{(i)}) + w(\boldsymbol{\theta}^{(t-1)})}. \tag{129}$$

Set $\boldsymbol{\theta}^{(t)} = \widetilde{\boldsymbol{\theta}}^{(j)}$.

3 Approximate the integral in Eq. (2) using Eq. (14).

5.2 MCMC with multiple candidates for the estimation of a dynamic parameter

In this section, we consider that the parameter of interest to be estimated (or at least part of it) is a dynamical variable, such as the state in a state-space model. In Section 5.2.1, the parameter of interest consists of a dynamical variable $\mathbf{x}$, i.e., $\boldsymbol{\theta} = \mathbf{x}$. In Section 5.2.2, we consider the more general scenario where the parameter of interest is formed by both a dynamical variable $\mathbf{x}$ and static variable $\boldsymbol{\lambda}$, i.e., $\boldsymbol{\theta} = [\mathbf{x}, \boldsymbol{\lambda}]^\top$.

5.2.1 *Particle Metropolis-Hastings (PMH) algorithms*

Let us assume that the variable of interest is a dynamical variable, i.e., $\boldsymbol{\theta} = \mathbf{x} = x_{1:D} = [x_1 \ldots, x_D]^\top$. This is the case of inferring a hidden state in state-space model, for instance. More generally, let assume that we are able to factorize the target density as

$$\bar{\pi}(\mathbf{x}) \propto \pi(\mathbf{x}) = \gamma_1(x_1) \prod_{d=2}^{D} \gamma_d(x_d | x_{d-1}). \tag{130}$$

The particle Metropolis-Hastings (PMH) method [317–319, 329] is an efficient MCMC technique, proposed independently from the MTM algorithm, specifically designed for being applied in this framework. Indeed, we can take advantage of the factorization of the target PDF and consider a proposal PDF decomposed in the same fashion

$$q(\mathbf{x}) = q_1(x_1) \prod_{d=2}^{D} q_d(x_d | x_{d-1}).$$

Then, as in a batch IS scheme, given an nth sample $\mathbf{x}^{(n)} = x_{1:D}^{(n)} \sim q(\mathbf{x})$ with $x_d^{(n)} \sim q_d(x_d | x_{d-1}^{(n)})$, we assign the importance weight

$$\begin{aligned} w(\mathbf{x}^{(n)}) = w_D^{(n)} &= \frac{\pi(\mathbf{x}^{(n)})}{q(\mathbf{x}^{(n)})} \\ &= \frac{\gamma_1(x_1^{(n)}) \gamma_2(x_2^{(n)} | x_1^{(n)}) \cdots \gamma_D(x_D^{(n)} | x_{D-1}^{(n)})}{q_1(x_1^{(n)}) q_2(x_2^{(n)} | x_1^{(n)}) \cdots q_D(x_D^{(n)} | x_{D-1}^{(n)})}. \end{aligned} \tag{131}$$

The structure above suggests the use of a sequential approach. Thus, PMH uses an SIR approach (see Section 4.8) to provide the particle approximation $\widehat{\pi}(\mathbf{x} | \mathbf{x}^{(1:N)}) = \sum_{i=1}^{N} \bar{w}_D^{(i)} \delta(\mathbf{x} - \mathbf{x}^{(i)})$, where $\bar{w}_D^{(i)} = \frac{w_D^{(i)}}{\sum_{n=1}^{N} w_D^{(n)}}$ and $w_D^{(i)} = w(\mathbf{x}^{(i)})$ is given by Eq. (131). Then, one particle is drawn from this approximation, i.e., with a probability proportional to the corresponding normalized weight.

Estimation of the marginal likelihood Z SIR combines the SIS approach with the application of resampling procedures. In SIR, a consistent estimator of Z is given by

$$\widetilde{Z} = \prod_{d=1}^{D} \left[\sum_{n=1}^{N} \bar{w}_{d-1}^{(n)} \beta_d^{(n)} \right], \tag{132}$$

where

$$\bar{w}_{d-1}^{(i)} = \frac{w_{d-1}^{(i)}}{\sum_{n=1}^{N} w_{d-1}^{(n)}}.$$

Due to the application of the resampling, in SIR the standard estimator

$$\widehat{Z} = \frac{1}{N} \sum_{n=1}^{N} w_D^{(n)} = \frac{1}{N} \sum_{n=1}^{N} w(\mathbf{x}^{(n)}), \tag{133}$$

is a possible alternative *only if* a proper weighting of the resampled particles is applied [301, 303]. If a proper weighting of a resampled particle is employed, both $\widetilde{Z}$ and $\widehat{Z}$ are equivalent estimators of Z [301, 303, 326]. Without the use of resampling steps (i.e., in SIS), $\widetilde{Z}$ and $\widehat{Z}$ are also equivalent estimators [303].

The complete description of PMH is provided in Algorithm 26 considering the use of $\widetilde{Z}$. At each iteration, a particle filter is run to obtain an approximation of the measure of the target with N weighted samples. Then, a sample among the N weighted particles is chosen by applying a single resampling step. This selected sample is then accepted or rejected as the next state of the chain according to an MH-type acceptance probability, which involves two estimators of the marginal likelihood Z.

Relationship between MTM and PMH schemes A simple look at I-MTM2 and PMH shows that they are closely related [326]. Indeed, the structure of the two algorithms coincides. The main difference lies in the fact that the candidates in PMH are generated sequentially using an SIR scheme. If no resampling steps are applied, then I-MTM2 and PMH are *exactly* the same algorithm, with candidates being drawn either in a *batch* setting or in a *sequential* way. Indeed, both PMH and I-MTM2 can be interpreted as a standard MH method with an independent proposal PDF and a proper weighting of a resampled particle [301, 303]. See [122, 303, 326] for further discussions on this issue.

5.2.2 *Particle marginal Metropolis-Hastings (PMMH) method*

Assume now that the variable of interest is formed by both dynamical and static variables, i.e., $\boldsymbol{\theta} = [\mathbf{x}, \boldsymbol{\lambda}]^\top$.

Algorithm 26 Particle Metropolis-Hastings (PMH).

1 **Initialization:** Choose a initial state $\mathbf{x}_0$ and obtain an initial estimation $\widetilde{Z}_0 \approx Z$.

2 For $t = 1, \ldots, T$:

(a) Employ an SIR approach to draw N particles and weight them, $\{\mathbf{x}^{(i)}, w_D^{(i)}\}_{i=1}^N$, i.e., sequentially obtain a particle approximation $\widehat{\pi}(\mathbf{x}) = \sum_{i=1}^N \bar{w}_D^{(i)} \delta(\mathbf{x} - \mathbf{x}^{(i)})$ where $\mathbf{x}^{(i)} = [x_1^{(i)}, \ldots, x_D^{(i)}]^\top$. Furthermore, also obtain $\widetilde{Z}^*$ as in Eq. (132).

(b) Draw $\mathbf{x}^* \sim \widehat{\pi}(\mathbf{x}|\mathbf{x}^{(1:N)})$, i.e., choose a particle $\mathbf{x}^* = \{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)}\}$ with probability $\bar{w}_D^{(i)}$, $i = 1, ..., N$.

(c) Set $\mathbf{x}_t = \mathbf{x}^*$ and $\widetilde{Z}_t = \widetilde{Z}^*$ with probability

$$\alpha = \min \left[1, \frac{\widetilde{Z}^*}{\widetilde{Z}_{t-1}} \right], \tag{134}$$

otherwise set $\mathbf{x}_t = \mathbf{x}_{t-1}$ and $\widetilde{Z}_t = \widetilde{Z}_{t-1}$.

3 **Return:** $\{\mathbf{x}_t\}_{t=1}^T$ with $\mathbf{x}_t = [x_{1,t}, \ldots, x_{D,t}]^\top$.

For instance, this is the case of inferring both the hidden state $\mathbf{x}$ in state-space model and the static parameters $\boldsymbol{\lambda}$ of the model. The particle marginal Metropolis-Hastings (PMMH) technique is an extension of PMH which addresses this problem [317, 319, 329].

Let us consider $\mathbf{x} = x_{1:D} = [x_1, x_2, \ldots, x_D] \in \mathbb{R}^{d_x}$, and an additional model parameter $\boldsymbol{\lambda} \in \mathbb{R}^{d_\lambda}$ to be inferred as well ($\boldsymbol{\theta} = [\mathbf{x}, \boldsymbol{\lambda}]^\top \in \mathbb{R}^D$, with $D = d_x + d_\lambda$). Assuming a prior PDF $g_\lambda(\boldsymbol{\lambda})$ over $\boldsymbol{\lambda}$, and a factorized complete posterior PDF $\bar{\pi}(\boldsymbol{\theta}) = \bar{\pi}(\mathbf{x}, \boldsymbol{\lambda})$,

$$\bar{\pi}(\mathbf{x}, \boldsymbol{\lambda}) \propto \pi(\mathbf{x}, \boldsymbol{\lambda}) = g_\lambda(\boldsymbol{\lambda}) \pi(\mathbf{x}|\boldsymbol{\lambda}), \tag{135}$$

where $\pi(\mathbf{x}|\boldsymbol{\lambda}) = \gamma_1(x_1|\boldsymbol{\lambda}) \prod_{d=2}^D \gamma_d(x_d|x_{1:d-1}, \boldsymbol{\lambda})$. For a specific value of $\boldsymbol{\lambda}$, we can use a $\boldsymbol{\lambda}$ and we can use a particle filter approach, obtaining the approximation $\widehat{\pi}(\mathbf{x}|\boldsymbol{\lambda}) = \sum_{n=1}^N \bar{w}_D^{(n)} \delta(\mathbf{x} - \mathbf{x}^{(n)})$ and the estimator $\widetilde{Z}(\boldsymbol{\lambda})$, as described above. The PMMH technique is then summarized in Algorithm 27. The PDF $q_\lambda(\boldsymbol{\lambda}|\boldsymbol{\lambda}_{t-1})$ denotes the proposal density for generating possible values of $\boldsymbol{\lambda}$. Observe that, with the specific choice $q_\lambda(\boldsymbol{\lambda}|\boldsymbol{\lambda}_{t-1}) = g_\lambda(\boldsymbol{\lambda})$, the acceptance function becomes

$$\alpha = \min \left[1, \frac{\widetilde{Z}(\boldsymbol{\lambda}^*)}{\widetilde{Z}(\boldsymbol{\lambda}_{t-1})} \right]. \tag{136}$$

Note also that PMMH w.r.t. to $\boldsymbol{\lambda}$ can be interpreted as MH method where the posterior cannot be evaluated pointwise. Indeed, $\widetilde{Z}(\boldsymbol{\lambda})$ approximates the marginal likelihood $p(\mathbf{y}|\boldsymbol{\lambda})$, i.e., it can also be interpreted as a special case of the pseudo-marginal approach described below [330].

Algorithm 27 Particle Marginal MH (PMMH).

1 **Initialization:** Choose the initial states $\mathbf{x}_0$, $\boldsymbol{\lambda}_0$, and an initial approximation $\widetilde{Z}_0(\boldsymbol{\lambda}) \approx Z(\boldsymbol{\lambda}) \approx p(\mathbf{y}|\boldsymbol{\lambda})$.

2 FOR $t = 1, \ldots, T$:

(a) Draw $\boldsymbol{\lambda}^* \sim q_\lambda(\boldsymbol{\lambda}|\boldsymbol{\lambda}_{t-1})$.

(b) Given $\boldsymbol{\lambda}^*$, run a particle filter obtaining $\widehat{\pi}(\mathbf{x}|\boldsymbol{\lambda}^*) = \sum_{n=1}^{N} \bar{w}_D^{(n)} \delta(\mathbf{x} - \mathbf{x}^{(n)})$ and $\widetilde{Z}(\boldsymbol{\lambda}^*)$, as in Eq. (132).

(c) Draw $\mathbf{x}^* \sim \widehat{\pi}(\mathbf{x}|\boldsymbol{\lambda}^*, \mathbf{x}^{(1:N)})$, i.e., choose a particle $\mathbf{x}^* = \{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)}\}$ with probability $\bar{w}_D^{(i)}$, $i = 1, ..., N$.

(d) Set $\boldsymbol{\lambda}_t = \boldsymbol{\lambda}^*$ and $\mathbf{x}_t = \mathbf{x}^*$ with probability

$$\alpha = \min\left[1, \frac{\widetilde{Z}(\boldsymbol{\lambda}^*) g_\lambda(\boldsymbol{\lambda}^*) q_\lambda(\boldsymbol{\lambda}_{t-1}|\boldsymbol{\lambda}^*)}{\widetilde{Z}(\boldsymbol{\lambda}_{t-1}) g_\lambda(\boldsymbol{\lambda}_{t-1}) q_\lambda(\boldsymbol{\lambda}^*|\boldsymbol{\lambda}_{t-1})}\right]. \tag{137}$$

Otherwise, set $\boldsymbol{\lambda}_t = \boldsymbol{\lambda}^*$ and $\mathbf{x}_t = \mathbf{x}_{t-1}$.

3 **Return:** $\{\mathbf{x}_t\}_{t=1}^T$ and $\{\boldsymbol{\lambda}_t\}_{t=1}^T$.

5.2.3 *Particle Gibbs algorithm*

Note that, in order to draw from $\bar{\pi}(\mathbf{x}, \boldsymbol{\lambda}) \propto \pi(\mathbf{x}, \boldsymbol{\lambda}) = \pi(x_{1:D}, \boldsymbol{\lambda})$ in Eq. (135), we could use a simple Gibbs sampling approach: draw first from the conditional PDF $\boldsymbol{\lambda}' \sim \bar{\pi}(\boldsymbol{\lambda}|\mathbf{x}')$ given a reference path $\mathbf{x}'$, and then sample a new path from the other conditional $\mathbf{x}'' \sim \bar{\pi}(\mathbf{x}|\boldsymbol{\lambda}')$. This procedure continues iteratively, drawing $\boldsymbol{\lambda}'' \sim \bar{\pi}(\boldsymbol{\lambda}|\mathbf{x}'')$ and $\mathbf{x}''' \sim \bar{\pi}(\mathbf{x}|\boldsymbol{\lambda}'')$, in a Gibbs sampling fashion. We can draw approximately the paths $\mathbf{x} = x_{1:D}$ from the conditional PDF $\bar{\pi}(\mathbf{x}|\boldsymbol{\lambda})$ by running a particle filter and then resampling once within the cloud of paths, as described in the sections above (exactly as in PMMH). However, note that is this procedure does not take into account the previous path $\mathbf{x}_{t-1}$ in order to generate the next sample $\mathbf{x}_t$, but only the $\boldsymbol{\lambda}_{t-1}$. Namely, $\mathbf{x}''$ is drawn from $\bar{\pi}(\mathbf{x}|\boldsymbol{\lambda}')$ that does not depend on $\mathbf{x}'$. The *particle Gibbs* (PG) technique is an extension of the simple Gibbs approach previously described that also considers the last sample generated, $\mathbf{x}_{t-1}$, to draw the next path $\mathbf{x}_t$ [317–319, 329][25]. Algorithm 28 summarizes the PG algorithm, which is guaranteed to generate a Markov chain with $\bar{\pi}(\mathbf{x}, \boldsymbol{\lambda})$ as invariant density [317–319].

5.3 Pseudo-marginal MCMC methods

There are numerous applications where the target density $\bar{\pi}$ is not available in closed form and cannot be evaluated pointwise exactly but only approximately. For instance, in some situations we can evaluate the joint target $\bar{\pi}(\boldsymbol{\lambda}, \mathbf{x})$,

[25] Related ideas about taking into account the previous path have been also discussed in [326].

Algorithm 28 Particle Gibbs (PG).

1 **Initialization:** Choose the initial states $\mathbf{x}_0$, $\boldsymbol{\lambda}_0$, the number of particles N and the total number of Gibbs iterations T.

2 FOR $t = 1, \ldots, T$:

(a) Run the Conditional Particle Filter (CPF) described in Section 4.8.2 (only $N-1$ particles are randomly generated), given $\boldsymbol{\lambda}_{t-1}$ and the reference path $\mathbf{x}_{t-1}$. Thus, we obtain $\widehat{\pi}(\mathbf{x}|\boldsymbol{\lambda}_{t-1}, \mathbf{x}_{t-1}) = \sum_{n=1}^{N} \bar{w}_D^{(n)} \delta(\mathbf{x} - \mathbf{x}^{(n)})$.

(b) Draw $\mathbf{x}_t \sim \widehat{\pi}(\mathbf{x}|\boldsymbol{\lambda}_{t-1}, \mathbf{x}_{t-1})$, i.e., choose a particle $\mathbf{x}^* = \{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)}\}$ with probability $\bar{w}_D^{(i)}$, $i = 1, ..., N$.

(c) Draw $\boldsymbol{\lambda}_t \sim \bar{\pi}(\boldsymbol{\lambda}|\mathbf{x}_t)$, as in a standard Block Gibbs sampling [48,49]

3 **Return:** $\{\mathbf{x}_t\}_{t=1}^T$ and $\{\boldsymbol{\lambda}_t\}_{t=1}^T$.

but we are actually only interested on the marginal target PDF, $\bar{\pi}(\boldsymbol{\lambda}) = \int_{\mathcal{X}} \bar{\pi}(\boldsymbol{\lambda}, \mathbf{x}) d\mathbf{x}$. If we cannot compute this integral, we cannot evaluate $\bar{\pi}(\boldsymbol{\lambda})$. One simple possibility is to run an MCMC algorithm in the extended space $[\boldsymbol{\lambda}, \mathbf{x}]$ and then consider only the first component. However, this approach can be very inefficient in many cases. An alternative is to run an MCMC algorithm in the subspace of $\boldsymbol{\lambda}$, addressing $\bar{\pi}(\boldsymbol{\lambda})$ but using an unbiased estimator $\widehat{\pi}(\boldsymbol{\lambda})$ of $\bar{\pi}(\boldsymbol{\lambda})$. This unbiased estimator can be provided by another Monte Carlo method. This is exactly the case of the PMMH algorithm described in Algorithm 27. Note that, if we are interested only in making inference about $\boldsymbol{\lambda}$, then the variable $\mathbf{x}$ can be considered integrated out using a Monte Carlo approximation [317].

In other related scenarios, the likelihood function $\ell(\mathbf{y}|\boldsymbol{\theta})$ cannot be evaluated and, fixing a generic value $\boldsymbol{\theta}$, an *unbiased* estimator $\widehat{\ell}(\mathbf{y}|\boldsymbol{\theta})$ of the probability $\ell(\mathbf{y}|\boldsymbol{\theta})$ is available, i.e.,

$$E\left[\widehat{\ell}(\mathbf{y}|\boldsymbol{\theta})\right] = \ell(\mathbf{y}|\boldsymbol{\theta}), \qquad \forall \boldsymbol{\theta} \in \Theta. \tag{138}$$

Note that this estimator must be unbiased and valid for all possible values of $\boldsymbol{\theta} \in \Theta$. If $\widehat{\ell}(\mathbf{y}|\boldsymbol{\theta})$ is available, then different Monte Carlo algorithms, such as MCMC techniques, can be applied considering the approximated posterior density [330]

$$\widehat{\pi}(\boldsymbol{\theta}) = \widehat{\pi}(\boldsymbol{\theta}|\mathbf{y}) \propto \widehat{\ell}(\mathbf{y}|\boldsymbol{\theta}) p_0(\boldsymbol{\theta}), \tag{139}$$

where $p_0(\boldsymbol{\theta})$ represents the prior PDF. Since the IS method is often used to provide the unbiased estimator $\widehat{\ell}(\mathbf{y}|\boldsymbol{\theta})$ [330], usually we have IS-within-MCMC algorithms in the pseudo-marginal setup. The generic pseudo-marginal MH method is summarized in Algorithm 29. This method is also known in the literature as *group independence MH* (GIMH) and a variant of this method is called *Monte Carlo*

within Metropolis (MCWM) [330]. They differ in the estimator, $\widehat{\pi}(\boldsymbol{\theta}^{(t-1)})$, used in the denominator of the acceptance probability α: in MCWM, $\widehat{\pi}(\boldsymbol{\theta}^{(t-1)})$ is recomputed at each iteration, whereas in GIMH the value estimated in the previous iteration is recycled (as in Algorithm 29).

Algorithm 29 Generic Pseudo-Marginal MH method

1 **Initialization:** Choose a proposal function $q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t-1)})$, an initial state $\boldsymbol{\theta}^{(0)}$, the total number of iterations (T), and the burn-in period (T_b).
2 FOR $t = 1, \ldots, T$:
   (a) Draw $\boldsymbol{\theta}' \sim q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t-1)})$.
   (b) Build an unbiased estimator $\widehat{\ell}(\mathbf{y}|\boldsymbol{\theta}')$ of the likelihood function $\ell(\mathbf{y}|\boldsymbol{\theta}')$ and $\widehat{\pi}(\boldsymbol{\theta}') \propto \widehat{\ell}(\mathbf{y}|\boldsymbol{\theta}')p_0(\boldsymbol{\theta}')$.
   (c) Set $\boldsymbol{\theta}^{(t)} = \boldsymbol{\theta}'$ with probability,
$$\alpha = \min\left[1, \frac{\widehat{\pi}(\boldsymbol{\theta}')q(\boldsymbol{\theta}^{(t-1)}|\boldsymbol{\theta}')}{\widehat{\pi}(\boldsymbol{\theta}^{(t-1)})q(\boldsymbol{\theta}'|\boldsymbol{\theta}^{(t-1)})}\right], \quad (140)$$
   otherwise, with probability $1-\alpha$, set $\boldsymbol{\theta}^{(t)} = \boldsymbol{\theta}^{(t-1)}$.
3 **Return:** $\boldsymbol{\theta}^{(t)}$ for $t = 1, \ldots, T$.

In the following subsections, we describe four different frameworks where the pseudo-marginal approach is either required or indirectly used. However, before describing potential applications of the pseudo-marginal approach, let us remark that the variance of the unbiased estimator used needs to be small in order to obtain a useful output. Otherwise, pseudo-marginal methods can result in very slowly-mixing chains even if they converge asymptotically in the limit. This emphasizes the importance of ensuring the geometric convergence of any MCMC algorithm to guarantee that it converges with a non-arbitrarily-slow rate.

5.3.1 Latent variable models

In latent variable models, the likelihood is often only available as an intractable integral

$$\ell(\mathbf{y}|\boldsymbol{\theta}) = \int_{\mathcal{Z}} \psi(\mathbf{y}, \mathbf{z}|\boldsymbol{\theta})d\mathbf{z},$$

and hence $\pi(\boldsymbol{\theta}|\mathbf{y}) \propto p_0(\boldsymbol{\theta})\left[\int_{\mathcal{Z}} \psi(\mathbf{y}, \mathbf{z}|\boldsymbol{\theta})d\mathbf{z}\right]$, which is also intractable. The simplest solution is to apply an MCMC algorithm for generating vectors $[\boldsymbol{\theta}', \mathbf{z}']$ from the joint target PDF, $\pi(\boldsymbol{\theta}, \mathbf{z}|\mathbf{y})$, and then considering only the first component of the drawn vectors [104]. More generally, an approximation of the integral $\int_{\mathcal{Z}} \psi(\mathbf{y}, \mathbf{z}|\boldsymbol{\theta})d\mathbf{z}$ is required. In some cases, this can be obtained using another Monte Carlo technique such as the IS technique.

5.3.2 Doubly-intractable likelihoods

Another scenario where the posterior PDF cannot be completely evaluated is the case of the so-called "doubly-intractable" likelihood functions. In this situation, a "portion" of the likelihood is unknown or cannot be evaluated, e.g.,

$$\ell(\mathbf{y}|\boldsymbol{\theta}) = \frac{1}{C(\boldsymbol{\theta})}\phi(\mathbf{y}|\boldsymbol{\theta}), \quad (141)$$

where $\phi(\mathbf{y}|\boldsymbol{\theta})$ can be evaluated, but

$$C(\boldsymbol{\theta}) = \int_{\Theta} \phi(\mathbf{y}|\boldsymbol{\theta})d\mathbf{y} \quad (142)$$

is unknown. Hence, the value $C(\boldsymbol{\theta})$ must be approximated [331]. A first algorithm for handling this kind of distributions was proposed in [332]. As an example, The single variable exchange (SVE) algorithm is described in Algorithm 30 (see [331]). If we denote as $\mathbf{y}_{\mathtt{true}}$ the actual observed data, the posterior PDF is

$$\begin{aligned}\bar{\pi}(\boldsymbol{\theta}|\mathbf{y}_{\mathtt{true}}) \propto \pi(\boldsymbol{\theta}|\mathbf{y}_{\mathtt{true}}) &= \ell(\mathbf{y}_{\mathtt{true}}|\boldsymbol{\theta})p_0(\boldsymbol{\theta}),\\ &= \frac{1}{C(\boldsymbol{\theta})}\phi(\mathbf{y}_{\mathtt{true}}|\boldsymbol{\theta})p_0(\boldsymbol{\theta}).\end{aligned} \quad (143)$$

Note that, if we are able to draw samples $\mathbf{y}^{(k)} \sim \ell(\mathbf{y}|\boldsymbol{\theta}) \propto \phi(\mathbf{y}|\boldsymbol{\theta})$ for $k = 1, ..., L$, then we can approximate the constant $C(\boldsymbol{\theta})$ via Monte Carlo approximation, i.e.,

$$C(\boldsymbol{\theta}) = \int_{\Theta} \phi(\mathbf{y}|\boldsymbol{\theta})d\mathbf{y} \approx \frac{1}{L}\sum_{k=1}^{L} \phi(\mathbf{y}^{(k)}|\boldsymbol{\theta}). \quad (144)$$

If we are able to draw from $\ell(\mathbf{y}|\boldsymbol{\theta})$, we can use the IS method, i.e., $\mathbf{y}^{(k)} \sim q_y(\mathbf{y})$ and then we have

$$C(\boldsymbol{\theta}) = \int_{\Theta} \phi(\mathbf{y}|\boldsymbol{\theta})d\mathbf{y} \approx \frac{1}{L}\sum_{k=1}^{L} \frac{\phi(\mathbf{y}^{(k)}|\boldsymbol{\theta})}{q_y(\mathbf{y}^{(k)})}. \quad (145)$$

For the sake of simplicity, let us assume that we are able to draw from $\ell(\mathbf{y}|\boldsymbol{\theta})$. Moreover, we set $L = 1$ and denote $\mathbf{y}' = \mathbf{y}^{(1)} \sim \ell(\mathbf{y}|\boldsymbol{\theta})$. Hence, we have $C(\boldsymbol{\theta}) \approx \phi(\mathbf{y}'|\boldsymbol{\theta})$. Then, we can write the approximate posterior function as

$$\begin{aligned}\pi(\boldsymbol{\theta}|\mathbf{y}_{\mathtt{true}}) &\approx \widehat{\pi}(\boldsymbol{\theta}|\mathbf{y}_{\mathtt{true}}, \mathbf{y}')\\ &= \frac{1}{\phi(\mathbf{y}'|\boldsymbol{\theta})}\phi(\mathbf{y}_{\mathtt{true}}|\boldsymbol{\theta})p_0(\boldsymbol{\theta}), \qquad \mathbf{y}' \sim \ell(\mathbf{y}|\boldsymbol{\theta}).\end{aligned} \quad (146)$$

The SVE algorithm is an MH method with the target function $\widehat{\pi}(\boldsymbol{\theta}|\mathbf{y}_{\mathtt{true}}, \mathbf{y}')$. Note that

$$\frac{C(\boldsymbol{\theta}^{(t-1)})}{C(\boldsymbol{\theta}')} \approx \frac{\phi(\mathbf{y}^{(t-1)}|\boldsymbol{\theta}^{(t-1)})}{\phi(\mathbf{y}'|\boldsymbol{\theta}')},$$

where $\mathbf{y}^{(t-1)} \sim \ell(\mathbf{y}|\boldsymbol{\theta}^{(t-1)})$ and $\mathbf{y}' \sim \ell(\mathbf{y}|\boldsymbol{\theta}')$.

Algorithm 30 Single Variable Exchange (SVE) algorithm.

1 **Initialization:** Choose a proposal function $q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t-1)})$, an initial state $\boldsymbol{\theta}^{(0)}$, the total number of iterations (T), and the burn-in period (T_b).

2 FOR $t = 1, \ldots, T$:

(a) Draw $\boldsymbol{\theta}' \sim q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t-1)})$.

(b) Draw $\mathbf{y}' \sim \ell(\mathbf{y}|\boldsymbol{\theta}') \propto \phi(\mathbf{y}|\boldsymbol{\theta}')$.

(c) Set $\boldsymbol{\theta}^{(t)} = \boldsymbol{\theta}'$ and $\mathbf{y}^{(t)} = \mathbf{y}'$ with probability,

$$\alpha = \min\left[1, \frac{\widehat{\pi}(\boldsymbol{\theta}'|\mathbf{y}_{\mathtt{true}}, \mathbf{y}')q(\boldsymbol{\theta}^{(t-1)}|\boldsymbol{\theta}')}{\widehat{\pi}(\boldsymbol{\theta}^{(t-1)}|\mathbf{y}_{\mathtt{true}}, \mathbf{y}^{(t-1)})q(\boldsymbol{\theta}'|\boldsymbol{\theta}^{(t-1)})}\right],$$

$$= \min\left[1, \frac{\phi(\mathbf{y}_{\mathtt{true}}|\boldsymbol{\theta}')}{\phi(\mathbf{y}_{\mathtt{true}}|\boldsymbol{\theta}^{(t-1)})} \frac{p_0(\boldsymbol{\theta}')}{p_0(\boldsymbol{\theta}^{(t-1)})} \frac{q(\boldsymbol{\theta}^{(t-1)}|\boldsymbol{\theta}')}{q(\boldsymbol{\theta}'|\boldsymbol{\theta}^{(t-1)})} \frac{\phi(\mathbf{y}^{(t-1)}|\boldsymbol{\theta}^{(t-1)})}{\phi(\mathbf{y}'|\boldsymbol{\theta}')}\right]$$

otherwise, with probability $1-\alpha$, set $\boldsymbol{\theta}^{(t)} = \boldsymbol{\theta}^{(t-1)}$ and $\mathbf{y}^{(t)} = \mathbf{y}^{(t-1)}$.

3 **Return:** $\boldsymbol{\theta}^{(t)}$ for $t = 1, \ldots, T$.

5.3.3 Approximate Bayesian computation (ABC)

In many applications, the likelihood cannot be evaluated for several different reasons: (a) it is to costly and/or (b) it is unknown analytically. However, in some of these scenarios it is possible to generate artificial data according to the likelihood, i.e., we can simulate synthetic data from the observation model [333–335]. Namely, in the Approximate Bayesian Computation (ABC) framework, we can draw samples $[\boldsymbol{\theta}', \mathbf{y}']$ from the joint target density,

$$p(\boldsymbol{\theta}, \mathbf{y}) = \ell(\mathbf{y}|\boldsymbol{\theta})p_0(\boldsymbol{\theta}),$$

with the following procedure:

1. Draw $\boldsymbol{\theta}' \sim p_0(\boldsymbol{\theta})$ (i.e., draw $\boldsymbol{\theta}'$ from the prior).
2. Draw $\mathbf{y}' \sim \ell(\mathbf{y}|\boldsymbol{\theta}')$ (i.e., draw $\mathbf{y}'$ from the observation model given $\boldsymbol{\theta}'$).

However, we are interested in having samples from the posterior density,

$$\bar{\pi}(\boldsymbol{\theta}|\mathbf{y}_{\mathtt{true}}) \propto p(\boldsymbol{\theta}, \mathbf{y}_{\mathtt{true}}), \tag{147}$$

where $\mathbf{y}_{\mathtt{true}}$ represents the actual observed data. To solve this issue, the underlying idea in ABC is to apply Monte Carlo techniques considering the generalized posterior PDF,

$$\bar{\pi}_\epsilon(\boldsymbol{\theta}, \mathbf{y}|\mathbf{y}_{\mathtt{true}}) \propto \pi_\epsilon(\boldsymbol{\theta}, \mathbf{y}|\mathbf{y}_{\mathtt{true}}) = h_\epsilon(||\mathbf{y} - \mathbf{y}_{\mathtt{true}}||)\ell(\mathbf{y}|\boldsymbol{\theta})p_0(\boldsymbol{\theta}), \tag{148}$$

where $||\cdot||$ denotes a norm, and $h_\epsilon(\xi) \in [0, 1]$ is a weighting function defined for $\xi \geq 0$ (with a parameter ϵ) which satisfies the following conditions: the maximum value is reached at 0 (i.e., $h_\epsilon(0) > h_\epsilon(\xi)$ for any $\xi > 0$), and the two following limits must be fulfilled, $\lim_{\epsilon \to 0} h_\epsilon(\xi) = \delta(\xi)$ and $\lim_{\epsilon \to \infty} h_\epsilon(\xi) = 0$. For instance, one possible choice is

$$h_\epsilon(||\mathbf{y} - \mathbf{y}_{\mathtt{true}}||) = \exp\left[-\frac{||\mathbf{y} - \mathbf{y}_{\mathtt{true}}||^2}{2\epsilon^2}\right], \tag{149}$$

whereas another common alternative is

$$h_\epsilon(||\mathbf{y} - \mathbf{y}_{\mathtt{true}}||) = \begin{cases} 1 & \text{if} \quad ||\mathbf{y} - \mathbf{y}_{\mathtt{true}}|| \leq \epsilon, \\ 0 & \text{if} \quad ||\mathbf{y} - \mathbf{y}_{\mathtt{true}}|| > \epsilon. \end{cases} \tag{150}$$

Considering the weighting function of Eq. (150), it is straightforward to see that, as $\epsilon \to 0$, then the generalized target $\bar{\pi}_\epsilon(\boldsymbol{\theta}, \mathbf{y}|\mathbf{y}_{\mathtt{true}})$ becomes more and more similar to $\bar{\pi}(\boldsymbol{\theta}|\mathbf{y}_{\mathtt{true}})$, and indeed

$$\lim_{\epsilon \to 0} \bar{\pi}_\epsilon(\boldsymbol{\theta}, \mathbf{y}|\mathbf{y}_{\mathtt{true}}) = \bar{\pi}(\boldsymbol{\theta}|\mathbf{y}_{\mathtt{true}}). \tag{151}$$

A Metropolis-Hastings ABC (MH-ABC) algorithm addressing the target density $\bar{\pi}_\epsilon(\boldsymbol{\theta}, \mathbf{y}|\mathbf{y}_{\mathtt{true}})$ defined in Eq. (148) with weighting function defined in Eq. (150), can be described as in Algorithm 31. Note that the extended proposal PDF in this case is

$$q_e(\boldsymbol{\theta}, \mathbf{y}|\boldsymbol{\theta}^{(t-1)}) = \ell(\mathbf{y}|\boldsymbol{\theta})q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t-1)}). \tag{152}$$

Drawing a $[\boldsymbol{\theta}', \mathbf{y}'] \sim q_e(\boldsymbol{\theta}, \mathbf{y}|\boldsymbol{\theta}^{(t-1)})$, the acceptance probability of the MH method in this case is

$$\alpha = \min\left[1, \frac{\pi_\epsilon(\boldsymbol{\theta}', \mathbf{y}'|\mathbf{y}_{\mathtt{true}})q_e(\boldsymbol{\theta}^{(t-1)}, \mathbf{y}^{(t-1)}|\boldsymbol{\theta}')}{\pi_\epsilon(\boldsymbol{\theta}^{(t-1)}, \mathbf{y}^{(t-1)}|\mathbf{y}_{\mathtt{true}})q_e(\boldsymbol{\theta}', \mathbf{y}'|\boldsymbol{\theta}^{(t-1)})}\right]. \tag{153}$$

Then, replacing the expressions of π_e an q_e, we have

$$\alpha = \min\left[1, \frac{h_\epsilon(||\mathbf{y}' - \mathbf{y}_{\mathtt{true}}||)\ell(\mathbf{y}'|\boldsymbol{\theta}')p_0(\boldsymbol{\theta}')\ell(\mathbf{y}^{(t-1)}|\boldsymbol{\theta}^{(t-1)})q(\boldsymbol{\theta}^{(t-1)}|\boldsymbol{\theta}')}{h_\epsilon(||\mathbf{y}^{(t-1)} - \mathbf{y}_{\mathtt{true}}||)\ell(\mathbf{y}^{(t-1)}|\boldsymbol{\theta}^{(t-1)})p_0(\boldsymbol{\theta}^{(t-1)})\ell(\mathbf{y}'|\boldsymbol{\theta}')q(\boldsymbol{\theta}'|\boldsymbol{\theta}^{(t-1)})}\right],$$

$$= \min\left[1, \frac{h_\epsilon(||\mathbf{y}' - \mathbf{y}_{\mathtt{true}}||)p_0(\boldsymbol{\theta}')q(\boldsymbol{\theta}^{(t-1)}|\boldsymbol{\theta}')}{h_\epsilon(||\mathbf{y}^{(t-1)} - \mathbf{y}_{\mathtt{true}}||)p_0(\boldsymbol{\theta}^{(t-1)})q(\boldsymbol{\theta}'|\boldsymbol{\theta}^{(t-1)})}\right].$$

It is important to remark that in the previous expression we do not need to evaluate the likelihood function. Finally, note that, if h_ϵ is given by Eq. (150), in order to avoid zeros in the denominator, the acceptance test involving the probability α can be split in two parts [334, 335],

$$\alpha = \begin{cases} \min\left[1, \frac{p_0(\boldsymbol{\theta}')q(\boldsymbol{\theta}^{(t-1)}|\boldsymbol{\theta}')}{p_0(\boldsymbol{\theta}^{(t-1)})q(\boldsymbol{\theta}'|\boldsymbol{\theta}^{(t-1)})}\right] & \text{if} \quad ||\mathbf{y} - \mathbf{y}_{\mathtt{true}}|| \leq \epsilon, \\ 0 & \text{if} \quad ||\mathbf{y} - \mathbf{y}_{\mathtt{true}}|| > \epsilon. \end{cases} \tag{154}$$

Algorithm 31 uses the acceptance probability in Eq. (154) [333–335].

5.3.4 Big data context

The ABC method completely avoids the evaluation of the likelihood. As a counterpart, ABC requires the ability of drawing artificial data from the observation model. Clearly, ABC fits very well in applications where evaluating the likelihood is expensive. The likelihood function can be costly due to the complexity of the model or

Algorithm 31 Metropolis-Hastings ABC (MH-ABC) algorithm

1 **Initialization:** Choose a proposal function $q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t-1)})$, an initial state $\boldsymbol{\theta}^{(0)}$, the total number of iterations (T), and the burn-in period (T_b).
2 FOR $t = 1, \ldots, T$:

(a) Draw $\boldsymbol{\theta}' \sim q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t-1)})$.
(b) Draw $\mathbf{y}' \sim \ell(\mathbf{y}|\boldsymbol{\theta}')$.
(c) If $||\mathbf{y} - \mathbf{y}_{\mathtt{true}}|| > \epsilon$, set $\boldsymbol{\theta}^{(t)} = \boldsymbol{\theta}^{(t-1)}$ and $\mathbf{y}^{(t)} = \mathbf{y}^{(t-1)}$.
(d) If $||\mathbf{y} - \mathbf{y}_{\mathtt{true}}|| \leq \epsilon$, then:

- Set $\boldsymbol{\theta}^{(t)} = \boldsymbol{\theta}'$ and $\mathbf{y}^{(t)} = \mathbf{y}'$ with probability,

$$\alpha = \min\left[1, \frac{p_0(\boldsymbol{\theta}')q(\boldsymbol{\theta}^{(t-1)}|\boldsymbol{\theta}')}{p_0(\boldsymbol{\theta}^{(t-1)})q(\boldsymbol{\theta}'|\boldsymbol{\theta}^{(t-1)})}\right]. \tag{155}$$

- Otherwise, with probability $1-\alpha$, set $\boldsymbol{\theta}^{(t)} = \boldsymbol{\theta}^{(t-1)}$ and $\mathbf{y}^{(t)} = \mathbf{y}^{(t-1)}$.

(e) **Return:** $\boldsymbol{\theta}^{(t)}$ for $t = 1, \ldots, T$.

because the size of the full dataset prohibits many evaluations of the likelihood. Specific methodologies have been designed for this second scenario, i.e., when a big number of data is available. All these techniques consider a cheaper likelihood function including only a subset of data at each iteration. One possible strategy, often known as *adaptive subsampling*, consists in computing the approximate acceptance probability α of the standard MH method, obtained considering only a random subset of data. Namely, an approximate implementation of the MH test is performed in some suitable way, in order to guarantee that the performance of the resulting technique is not jeopardized (e.g., the total variation distance between the perturbed invariant distribution and the desired target distribution is controlled) [336–339]. Other methods are based on the so-called *delayed acceptance* approach: divide the acceptance MH test into several parts involving likelihood functions with an increasing number of data [336, 337, 339]. A related strategy, called *early rejection*, was proposed in [340]. However, in the early rejection MCMC technique the acceptance of a new proposed state still requires the evaluation of the full-likelihood, whereas the rejection may require only the evaluation of a partial likelihood based on a subset of data[26]. Another simple approach consists in dividing the full dataset into mini-batches, running different parallel Monte Carlo algorithms and combining all the partial estimators to obtain the global one [242, 243, 341–344].

[26] Despite their denominations, this kind of methods (called "delayed acceptance" or "early rejection") are not directly related to the delayed rejection MH algorithm described in Section 3.4.2, which always considers the complete likelihood.

5.4 Noisy and approximate likelihood methods

5.4.1 Noisy MCMC methods

The aforementioned methods can be grouped in the unique framework shown in Algorithm 32 (e.g., see [345–347]). Let us assume that we are not able to evaluate the standard acceptance probability function of the MH method,

$$\alpha = \min\left[1, \rho(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{\theta}')\right],$$

where $\rho(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{\theta}') = \frac{\pi(\boldsymbol{\theta}')q(\boldsymbol{\theta}^{(t-1)}|\boldsymbol{\theta}')}{\pi(\boldsymbol{\theta}^{(t-1)})q(\boldsymbol{\theta}'|\boldsymbol{\theta}^{(t-1)})}$. Then, an approximation

$$\widehat{\alpha} = \min\left[1, \widehat{\rho}(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{\theta}', \mathbf{y}')\right],$$

where $\mathbf{y}' \sim \ell(\mathbf{y}|\boldsymbol{\theta}')$, can be used [346]. It is possible to show that, if $\widehat{\rho}(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{\theta}', \mathbf{y})$ fulfills the following condition,

$$\int_{\mathcal{Y}} \left|\widehat{\rho}(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{\theta}', \mathbf{y}) - \rho(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{\theta}')\right| \ell(\mathbf{y}|\boldsymbol{\theta}')d\mathbf{y} \leq \delta(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{\theta}'),$$
$$\forall \boldsymbol{\theta}^{(t-1)}, \boldsymbol{\theta}' \in \Theta$$

the stationary density of the generated chain will approximate the desired posterior PDF [346]. Namely, $\widehat{\rho}(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{\theta}', \mathbf{y}')$ with $\mathbf{y}' \sim \ell(\mathbf{y}|\boldsymbol{\theta}')$ is a randomized version of $\rho(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{\theta}')$, and it is reasonable to require $\left|\widehat{\rho}(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{\theta}', \mathbf{y}) - \rho(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{\theta}')\right|$ be small in order to obtain a useful approximation [346, 347].

Algorithm 32 Noisy MH method

1 **Initialization:** Choose a proposal function $q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t-1)})$, an initial state $\boldsymbol{\theta}^{(0)}$, the total number of iterations (T), and the burn-in period (T_b).
2 FOR $t = 1, \ldots, T$:

(a) Draw $\boldsymbol{\theta}' \sim q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t-1)})$.
(b) Draw $\mathbf{y}' \sim \ell(\mathbf{y}|\boldsymbol{\theta}')$.
(c) Build an estimator $\widehat{\rho}(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{\theta}', \mathbf{y}')$ of the standard acceptance probability of the MH method, $\rho(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{\theta}')$.
(d) Set $\boldsymbol{\theta}^{(t)} = \boldsymbol{\theta}'$ with probability

$$\widehat{\alpha} = \min\left[1, \widehat{\rho}(\boldsymbol{\theta}^{(t-1)}, \boldsymbol{\theta}', \mathbf{y}')\right],$$

Otherwise, with probability $1 - \widehat{\alpha}$, set $\boldsymbol{\theta}^{(t)} = \boldsymbol{\theta}^{(t-1)}$.

3 **Return:** $\boldsymbol{\theta}^{(t)}$ for $t = 1, \ldots, T$.

5.4.2 Approximate likelihood methods

In state-space models, and especially in models involving non-linear stochastic differential equations, the (marginal) likelihood $\ell(\mathbf{y}|\boldsymbol{\theta})$ often cannot be evaluated exactly, but we may have a simple approximation $\widehat{\ell}(\mathbf{y}|\boldsymbol{\theta})$ available. For example, in non-linear state-space models we might have a non-linear Kalman filter-based Gaussian approximation of the system [348], which also provides us with an approximation of the likelihood.

As discussed above, if $\widehat{\ell}(\mathbf{y}|\boldsymbol{\theta})$ is unbiased in the sense of Eq. (138), then using the corresponding posterior distribution (139) in an MH algorithm leads to a valid algorithm for sampling the parameters. In the case of non-linear Kalman filter approximations (like extended, unscented, or cubature Kalman filters) the estimate is not unbiased, but this has not prevented researchers from using them. Indeed, several researchers have shown that Kalman filters can provide good approximations of the true posterior distribution in non-linear discrete-time state-space models [348], as well as in non-linear models involving stochastic differential equations [349, 350].

5.4.3 Analysis of noisy/approximate likelihood methods

Note that, if we have a Markov chain $\mathcal{M}$, and another Markov chain $\mathcal{M}'$ close to $\mathcal{M}$ in some sense, the stationary distribution π' of $\mathcal{M}'$ need not exist, and if it does, it need not be close to the stationary distribution of $\mathcal{M}$. Consequently, studying noisy/approximate MCMC methods is a rather delicate task. In this sense, it is worth mentioning the work of Johndrow et al. [351], which includes a general perturbation bound for uniformly ergodic chains, as well as Negrea and Rosenthal's work [352], which presents a more complicated bound for geometrically ergodic chains.

6 Numerical simulations

In this section, we present several examples where the performance of many of the previously described algorithms is evaluated. We start with two simple examples (univariate and bivariate Gaussians), where the true estimators can be computed analytically, and thus we can gauge exactly the performance of the different methods. Then, we address a challenging problem that appears in several scientific fields: the estimation of the parameters of a chaotic system. Finally, we also tackle two classical signal processing problems: localization in a wireless sensor network and a spectral analysis example.

6.1 Illustrative example for adaptive MCMC algorithms

For the sake of simplicity, in this first example we consider a univariate target density which is a mixture of Gaussian PDFs. More specifically, the target PDF is formed by M Gaussians, i.e.,

$$\bar{\pi}(\theta) = \frac{1}{M}\sum_{i=1}^{M} \mathcal{N}(\theta|\eta_i, \rho_i^2), \tag{156}$$

with variances $\rho_i^2 = 4$ for all $i = 1, ..., M$. We consider three different cases, with $M \in \{2, 3, 6\}$. The means are $\eta_1 = -10$ and $\eta_2 = 10$ for $M = 2$; $\eta_1 = -10$, $\eta_2 = 0$, and $\eta_3 = 10$ for $M = 3$; $\eta_1 = -15$, $\eta_2 = -10$, $\eta_3 = -5$, $\eta_4 = 5$, $\eta_5 = 10$, and $\eta_6 = 15$ for $M = 6$.

We test the adaptive Metropolis (AM) scheme, that uses an adaptive random walk Gaussian proposal [196], $q_t(\theta|\theta^{(t-1)}, \sigma_t^2) = \mathcal{N}(\theta|\theta^{(t-1)}, \sigma_t^2)$, and the Adaptive Gaussian Mixture Metropolis-Hastings (AGM-MH) algorithm of [200], that uses the following proposal PDF:

$$q_t(\theta|\mu_{1,t}, \ldots, \mu_{N,t}, \sigma_{t,1}^2, \ldots, \sigma_{t,N}^2) = \sum_{n=1}^{N} w_n^{(t)} \varphi_i(\theta|\mu_i^{(t)}, \sigma_{i,t}^2), \tag{157}$$

formed by N Gaussian which are independent from the previous state of the chain, i.e., $\varphi_i(\theta|\mu_i^{(t)}, \sigma_{i,t}^2) = \mathcal{N}(\theta|\mu_i^{(t)}, \sigma_{i,t}^2)$. Moreover, we also compare the correlations obtained by the adaptive MH schemes with those obtained using a non-adaptive standard MH algorithm with a random walk proposal PDF. In AGM-MH, we set $N = M$ Gaussians and each initial mean is chosen uniformly in $[-20, 20]$. The initial variances and weights are set as $\sigma_{i,0}^2 = 10$ and $w_{i,0} = 1/N$ for all $i = 1, ..., N$. The same initialization of the variance is employed for the single component of AM: $\sigma_0^2 = 10$. The goal of this example is to show that performing Monte Carlo estimation on multi-modal targets without specialized algorithms (like adiabatic MC [353]) is challenging, but can still be tackled by properly designed adaptive algorithms with mixture proposals.

We perform $T = T_{\text{tot}} = 5000$ iterations of the chain, setting $T_{\text{train}} = 200$ (the number of iterations for the initial training period) and $T_{\text{stop}} = T_{\text{tot}}$ (i.e., the adaptation is never stopped) for the AGM-MH algorithm (see [200] for a detailed description of these two parameters). The initial state of the chain is randomly chosen as $\theta^{(0)} \sim \mathcal{N}(\theta|0, 1)$ in all cases. Then, we use all the generated samples (i.e., $T_b = 0$) to estimate the normalizing constant of the target. Table 6 shows the mean squared error (MSE), averaged over 1000 independent runs, for the AM and AGM-MH

Table 6 Mean squared error (MSE) for the univariate Gaussian target in Section 6.1

Algorithm	$M = 2$	$M = 3$	$M = 6$
AM	2×10^{-2}	2×10^{-2}	6×10^{-3}
AGM-MH	1.6×10^{-4}	1.1×10^{-4}	2×10^{-5}

Table 7 Normalized auto-correlation (at lag one) for the univariate Gaussian target in Section 6.1

Algorithm	$M = 2$	$M = 3$	$M = 6$
Standard MH	0.81	0.72	0.46
AM	0.33	0.26	0.20
AGM-MH	0.13	0.14	0.16

algorithms in the estimation of the expected value of the target PDF, whereas Table 7 shows the auto-correlation (at lag one) of AM, AGM-MH, and a standard MH algorithm without adaptation. Note the improvement, both in terms or MSE and auto-correlation, attained by both of the adaptive MH algorithms (especially by AGM-MH) even in this simple example. Finally, Fig. 1 depicts the averaged values of the acceptance probability, α_t, of the AGM-MH algorithm as function of t and for different values of M. Note the increase in the averaged values of α_t for $t > T_{\text{train}}$ as a result of the adaptation.

This example shows that a classical adaptive algorithm (AM) fails, as clearly shown by the large MSE and auto-correlation values, whereas a properly designed adaptive MC method (AGM-MH) with an adequate proposal can attain very good results: an MSE two orders of magnitude lower and an auto-correlation up to 2.5 times smaller. The performance of the random walk MH largely depends on the variance of the proposal, which should be optimized in order to attain a 25–40% average acceptance rate, as discussed earlier. Note that this can be easily achieved in this simple example but becomes a much more challenging task for more complex problems. Therefore, properly

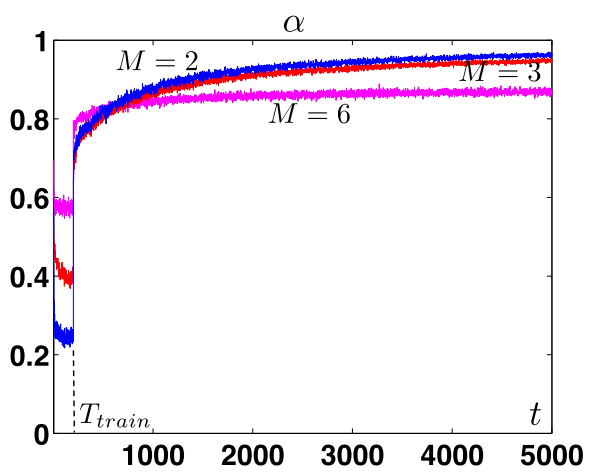


Fig. 1 Averaged values of the acceptance probability, α_t, as function of the iteration index t and different values of $M \in \{2, 3, 6\}$, for the univariate Gaussian target in Section 6.1, using the AGM-MH algorithm. For $t > T_{\text{train}}$, α_t grows as a result of the adaptation of the proposal, which becomes closer to the target

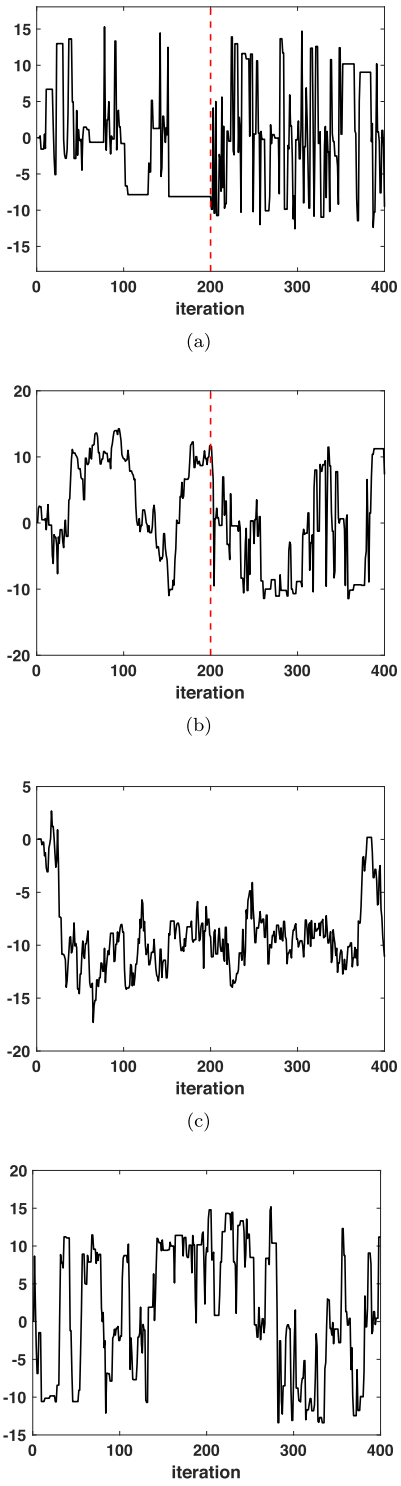


Fig. 2 Trace plots for the univariate Gaussian target in Section 6.1 with $M = 3$. **a** AGM-MH algorithm. **b** AM algorithm. The red dashed line in (**a**) and (**b**) marks the beginning of the adaptation period. **c** Random walk MH with $\sigma = 2$. **d** Random walk MH with $\sigma = 5$

designed adaptive algorithms should be preferred when applicable.

In order to further illustrate the behavior of the three considered algorithms (RWMH, AM, and AGM-MH), Fig. 2 shows typical trace plots for $M = 3$ of the two adaptive techniques (AGM-MH and AM), as well as the RWMH algorithm with two different values of σ. From Fig. 2a, we see that the chain's state for AGM-MH constantly switches to locations around the three modes of the target (placed at $\eta_1 = -10$, $\eta_2 = 0$, and $\eta_3 = 10$ for $M = 3$), showing that the chain is frequently exploring all the modes. Then, Fig. 2b shows that the chain attained by AM also explores the three modes, but the jumps from one mode to another occur less frequently. Finally, Fig. 2c, d show that the performance of the RWMH algorithm critically depends on the variance: for $\sigma = 2$ the resulting chain in the example completely fails to explore one of the modes, whereas for $\sigma = 5$ all the three modes are properly covered.

Table 8 MSE in the estimation of $E(\boldsymbol{\theta})$, keeping the total number of evaluations of the target fixed to $L = KNT = 2 \cdot 10^5$ in all algorithms, for the bivariate target in Section 6.2

$\boldsymbol{L = NKT = 2 \cdot 10^5}$							
$\boldsymbol{N}$	**Algorithm**	$\sigma = \mathbf{1}$	$\sigma = \mathbf{2}$	$\sigma = \mathbf{5}$	$\sigma = \mathbf{10}$	$\sigma = \mathbf{20}$	$\sigma = \mathbf{70}$
5		92.80	38.71	12.65	0.38	0.047	37.44
100	Standard PMC [95]	75.17	59.42	14.24	0.25	0.028	0.18
$5 \cdot 10^4$		68.29	37.44	7.01	0.25	0.033	0.17
	DM-PMC ($K = 1$)	72.48	36.21	5.34	0.036	0.029	0.21
	GR-PMC ($K = 2$)	69.41	26.23	3.09	0.022	0.028	0.17
	LR-PMC ($K = 2$)	2.68	0.007	0.010	0.018	0.102	32.88
	GR-PMC ($K = 5$)	67.04	17.44	0.11	0.013	0.023	0.15
	LR-PMC ($K = 5$)	8.04	0.012	0.008	0.016	0.027	2.00
	GR-PMC ($K = 20$)	61.58	15.13	0.42	0.012	0.024	0.14
	LR-PMC ($K = 20$)	9.51	1.16	0.011	0.013	0.023	0.22
100	GR-PMC ($K = 100$)	64.94	12.50	0.08	0.015	0.026	0.18
	LR-PMC ($K = 100$)	9.60	1.21	0.022	0.015	0.026	0.20
	GR-PMC ($K = 500$)	58.49	9.63	0.08	0.014	0.024	0.16
	LR-PMC ($K = 500$)	14.79	6.72	0.10	0.010	0.024	0.20
100	M-PMC [96]	71.39	81.33	18.14	0.058	0.031	0.14
10		84.14	81.68	6.49	0.76	0.024	4.60
100	SMC [286]	77.00	76.5	15.98	0.79	0.068	0.86
$5 \cdot 10^4$		69.08	51.29	20.48	0.22	0.038	0.68
	DM-SMC ($K = 1$)	70.95	42.40	1.91	0.039	0.027	0.19
	GR-SMC ($K = 5$)	66.64	41.54	0.16	0.015	0.024	0.19
100	LR-SMC ($K = 5$)	8.16	2.32	0.007	0.015	0.027	2.19
	GR-SMC ($K = 20$)	65.48	37.91	0.10	0.013	0.025	0.19
	LR-SMC ($K = 20$)	8.88	4.15	0.010	0.014	0.026	0.20
	APIS ($T = 100$)	0.0318	0.0011	0.0054	**0.0129**	**0.0211**	0.1794
	APIS ($T = 50$)	0.0144	0.0007	0.0051	0.0131	**0.0221**	0.1772
100	APIS ($T = 20$)	0.0401	0.0006	**0.0047**	0.0136	**0.0245**	0.1732
	APIS ($T = 5$)	**0.0008**	0.0005	0.0064	0.0149	0.0270	0.2076
	APIS ($T = 2$)	0.0017	0.0116	0.0103	0.0182	0.0387	0.1844
1	AMIS (best)	112.70	107.85	44.93	0.7404	**0.0121**	**0.0141**
	AMIS (worst)	115.62	111.83	70.62	9.43	0.0871	18.62

The best results for each value of σ are highlighted in bold-face

6.2 Illustrative example for adaptive importance sampling

As a second simple example, let us consider a multimodal target PDF consisting of a mixture of five bivariate Gaussians, i.e.,

$$\bar{\pi}(\boldsymbol{\theta}) = \frac{1}{5}\sum_{i=1}^{5}\mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\nu}_i, \boldsymbol{\Sigma}_i), \tag{158}$$

with $\boldsymbol{\theta} \in \mathbb{R}^2$, means $\boldsymbol{\nu}_1 = [-10, -10]^\top$, $\boldsymbol{\nu}_2 = [0, 16]^\top$, $\boldsymbol{\nu}_3 = [13, 8]^\top$, $\boldsymbol{\nu}_4 = [-9, 7]^\top$ and $\boldsymbol{\nu}_5 = [14, -14]^\top$, and covariances $\boldsymbol{\Sigma}_1 = [2,\ 0.6; 0.6,\ 1]$, $\boldsymbol{\Sigma}_2 = [2,\ -0.4; -0.4,\ 2]$, $\boldsymbol{\Sigma}_3 = [2,\ 0.8; 0.8,\ 2]$, $\boldsymbol{\Sigma}_4 = [3,\ 0; 0,\ 0.5]$, and $\boldsymbol{\Sigma}_5 = [2,\ -0.1; -0.1,\ 2]$. Since we can analytically compute the moments of the target, this example is very useful to validate the performance of different Monte Carlo techniques. In particular, we consider the computation of the mean of the target, $E(\boldsymbol{\theta}) = [1.6, 1.4]^\top$, and the normalizing constant, $Z = 1$. We use the MSE (averaged over both components in the computation of $E(\boldsymbol{\theta})$) as the figure of merit of the different estimators.

For simplicity, we use again Gaussian proposal densities for all the MC methods. The proposals are "poorly" initialized on purpose in order to test the robustness and the adaptation capabilities of the methods. More specifically, the location parameters of the proposals are initialized uniformly within the $[-4, 4] \times [-4, 4]$ square, i.e., $\boldsymbol{\mu}_i^{(1)} \sim \mathcal{U}([-4, 4] \times [-4, 4])$ for $i = 1, \ldots, N$. Note that none of the modes of the target falls within the initialization square. We test all the alternative methods using the same isotropic covariance matrices for all the Gaussian proposals, $\mathbf{C}_i = \sigma^2 \mathbf{I}_2$ with $\sigma \in \{1, 2, 5, 10, 20, 70\}$. All the results have been averaged over 500 independent experiments, where the computational cost of the different techniques (in terms of the total number of evaluations of the target distribution, which is usually the most costly step in practice) is fixed to $L = KNT$[27]. We compare the following schemes:

- *Standard PMC [95]:* The standard PMC algorithm proposed in [95] with $N = 100$ proposals and $T = 2000$ iterations. The total number of samples drawn is $L = NT = 2 \cdot 10^5$.
- *M-PMC [96]:* The M-PMC algorithm proposed in [96] with $N = 100$ proposals, $M = 100$ samples per iteration, and $T = 2000$ iterations. The total number of samples drawn is $L = MT = 2 \cdot 10^5$.
- *SMC [286]:* A sequential Monte Carlo (SMC) scheme combining resampling and MCMC steps. More precisely, we consider MH steps as forward reversible kernels. In this example, we do not employ a

[27] Note that $L = KNT$ also corresponds to the total number of samples generated in all the schemes.

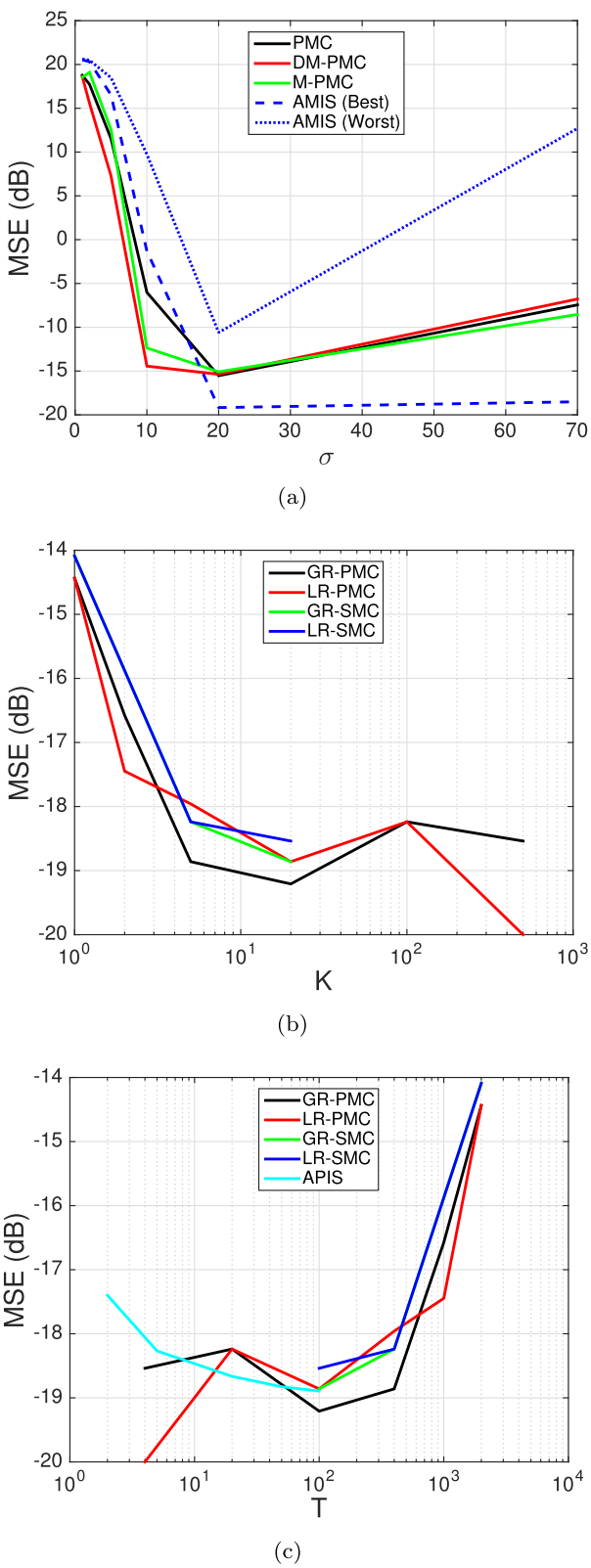


Fig. 3 Example of Section 6.2. **a** MSE of several algorithms as a function of σ for $N = 100$. **b** MSE as a function of K for $\sigma = 10$. **c** MSE as a function of T for $\sigma = 10$

Table 9 MSEs in estimation of R and Ω using FUSS and MH inside a Gibbs sampler, with $\delta = 10^{-3}$, $K = 10$, and $N_G = 50$, for the example of Section 6.3

		$\lambda = 0.001$	$\lambda = 0.005$	$\lambda = 0.01$	$\lambda = 0.05$	$\lambda = 0.08$	$\lambda = 0.10$
FUSS	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

The observed sequence, $\mathbf{z}_{1:T}$, is generated with $R = 3.7$, $\Omega = 0.4$, and $N = 20$ and different values of λ

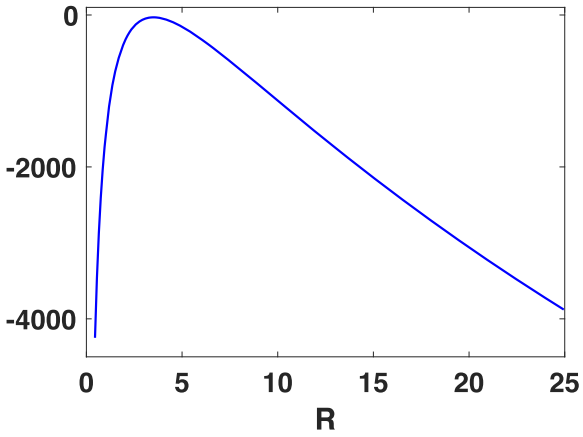
sequence of tempered target PDFs, i.e., we consider always the true target density. The proposal PDFs for the MH kernels coincide with the Gaussian proposals employed in the propagation resampling steps, with the scale parameters $\mathbf{C}_i$ of the other tested methods. Due to the application of the MH steps, in this case, $L > 2 \cdot 10^5$.

- *K-PMC [274]:* The standard PMC scheme using $N = 100$ proposals, but drawing $K > 1$ samples per proposal at each iteration and performing global resampling (GR). In order to keep the total number of samples constant, the number of iterations of the algorithm is now $T = 2 \cdot 10^5/(KN)$.
- *DM-PMC [274]:* The standard PMC using the weights of Eq. (67) (i.e., the mixture of all proposals at each iteration), $N = 100$ proposals, $T = 2000$ iterations, and drawing $K = 1$ samples per proposal (i.e., $M = N = 100$ samples per iteration). The total number of samples drawn is again $L = MT = 2 \cdot 10^5$.
- *GR-PMC [274]:* The standard PMC scheme with multiple samples per proposal (K), weights computed as in DM-PMC, and global resampling (GR). We use $N = 100$ proposals and $T = L/(KN)$ iterations with $L = 2 \cdot 10^5$ again. In particular, we test the values $K \in \{2, 5, 20, 100, 500\}$, and thus $T \in \{1000, 400, 100, 20, 4\}$.
- *LR-PMC [274]:* The standard PMC scheme with multiple samples per proposal (K) and local resampling (LR). All the parameters are selected as in the GR-PMC scheme.
- *Improved SMC [274, 286]:* The SMC scheme with the improvements proposed in those two papers. In all cases, we use the importance weights as in DM-PMC (deterministic mixture of the proposals at each iteration), and we try the GR-SMC and LR-SMC variants. We test $K \in \{5, 20\}$
- *APIS [99]:* The adaptive population importance sampling (APIS) scheme with $N = 100$ proposals and $T = 2000$ iterations. The IS weights are again the spatial deterministic mixture weights.
- *AMIS [98]:* The adaptive multiple importance sampling (AMIS) algorithm, which uses a single proposal, drawing K samples per iteration and running for T iterations. We use values of K and T such that $L = KT = 2 \cdot 10^5$, for a fair comparison. Specifically, we have run different simulations using $K \in \{500, 1000, 2000, 5000\}$ and, as a consequence, $T \in \{40, 20, 10, 4\}$. Since the samples are reweighted using the whole set of past temporal proposals in the denominator (i.e., a sort of temporal deterministic mixture), AMIS becomes more costly when T increases. In Table 8, we show the best and worst performance for each value of σ.

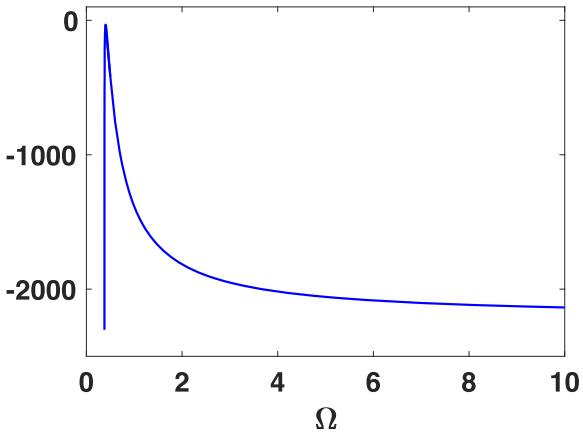
Table 8 shows the full results for the MSE in the estimation of $E(\boldsymbol{\theta})$ averaged over both components, whereas Fig. 3 graphically displays some selected cases. We can see that the compared schemes outperform the standard PMC for any value of σ. In general, the local resampling (LR-PMC) works better than the global resampling (GR-PMC). APIS obtains a good performance for several intermediate values of σ, while AMIS behaves well with large values of σ. Moreover, we note that the optimum value of K in GR-PMC and LR-PMC depends on the value of σ, the scale parameter of the proposals: for small values of σ (e.g., $\sigma = 1$ or $\sigma = 2$) small values of K lead to better performance, whereas a larger value of K (and thus less iterations T) can be used for larger values of σ (e.g., $\sigma = 10$ or $\sigma = 20$).

6.3 Parameter estimation in a chaotic system

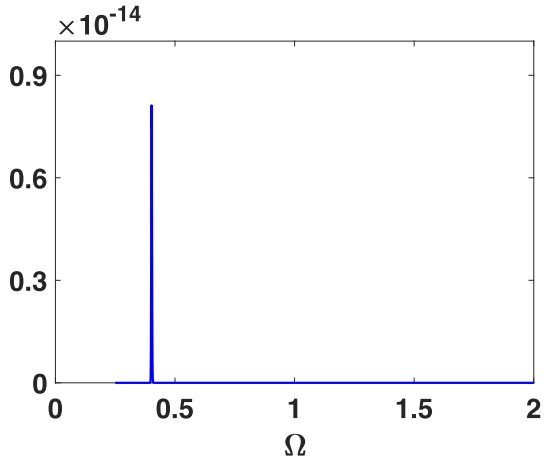
In this numerical experiment, we address the estimation of the parameters of a chaotic system, which is considered a very challenging problem in the literature [354, 355], since the resulting PDFs typically present very sharp full-conditionals. This type of systems is often utilized for modeling the evolution of population sizes, for instance in ecology [354]. Let us consider a logistic map [356] perturbed by multiplicative noise,



(a)

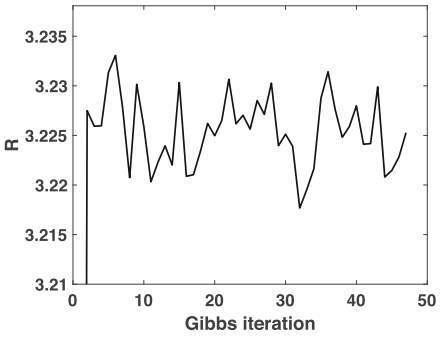


(b)

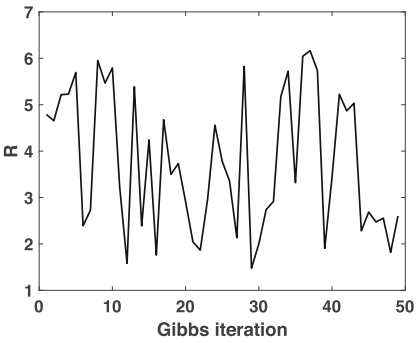


(c)

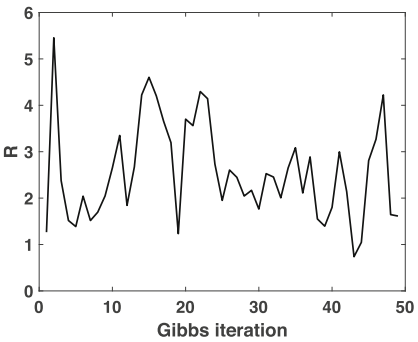
Fig. 4 Example of Section 6.3. **a**, **b** Examples of (unnormalized) conditional log-PDFs with $\lambda = 0.1$ and considering $N = 20$ observations. **a** Fixing $\Omega = 4$. **b** Fixing $R = 0.7$. **c** The (unormalized) conditional PDF corresponding to **b**



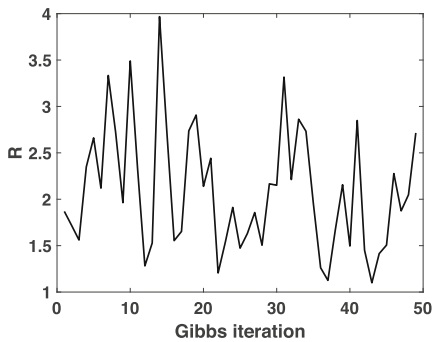
(a)



(b)



(c)



(d)

Fig. 5 Trace plots for R in the example of Section 6.3. **a** FUSS-within-Gibbs. **b**–**d** MH-within-Gibbs for $\sigma = 0.2$ (**b**), $\sigma = 1$ (**c**) and $\sigma = 2$

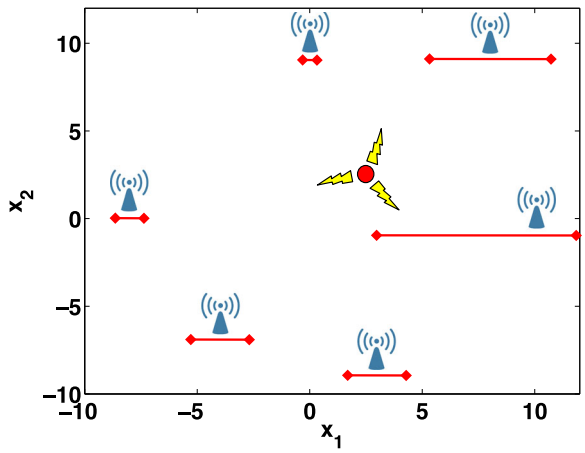


Fig. 6 Location of the sensors and the target for the example in Section 6.4

$$z_{t+1} = R\left[\, z_t \left(1 - \frac{z_t}{\Omega}\right)\right] \exp(\epsilon_t), \tag{159}$$

with $\epsilon_t \sim \mathcal{N}(0, \lambda^2)$, $z_1 \sim \mathcal{U}([\,0, 1\,])$, and unknown parameters $R > 0$ and $\Omega > 0$. Let us assume that a sequence $\mathbf{z}_{1:T} = [z_1, \ldots, z_T]$ is observed and, for the sake of simplicity, that λ is known. Under these circumstances, the likelihood function is given by

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

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,

$$\bar{\pi}(\boldsymbol{\theta}|\mathbf{z}_{1:T}) = \bar{\pi}(R, \Omega|\mathbf{z}_{1:T}) \propto p(\mathbf{z}_{1:T}|R, \Omega), \tag{160}$$

which corresponds to the minimum mean squared error (MMSE) estimate of the parameters. Note that the parameter vector to be inferred in this example is $\boldsymbol{\theta} = [\theta_1 = R, \theta_2 = \Omega]$.

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 [354]. Then, we apply FUSS-within-Gibbs [179] (with $\delta = 10^{-3}$, $K = 10$ and an initial grid $\widetilde{\mathcal{S}}_M = \{10^{-4}, 2 \cdot 10^{-4}, \ldots, 20\}$), using only $N_G = 50$ iterations of the Gibbs sampler. We also consider an MH-within-Gibbs approach with a random walk proposal, $\bar{q}(\theta_i^{(t)}|\theta_i^{(t-1)}) \propto \exp\left(\frac{-(\theta_i^{(t)} - \theta_i^{(t-1)})^2}{2\sigma_p^2}\right)$, with $i \in \{1, 2\}$, and two different values of $\sigma_p \in \{1, 2\}$. The initial states of the chains are chosen randomly from $\mathcal{U}([\,1, 5\,])$ and $\mathcal{U}([\,0.38, 1.5\,])$, respectively. In order to compare the performance of both approaches, we also perform an approximate computation of the true value of the mean via an expensive deterministic numerical integration procedure.

The results, averaged over 1000 independent runs, are shown in Table 9. It can be clearly seen that FUSS-within-Gibbs 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. Figure 4a, b provide two examples of conditional log-PDFs, whereas Fig. 4c shows the "sharp" conditional density corresponding to Fig. 4b. This PDF resembles a delta function: even using sophisticated adaptive techniques it is difficult to recognize the mode of this kind of target PDF. However, by constructing a proposal which is adapted to the full conditionals using the FUSS algorithm, very good results can be obtained even in this extreme case.

Finally, Fig. 5 shows the trace plots for this example using the FUSS and MH algorithms, both within the Gibbs sampler, for the parameter R[28]. On the one hand, note the small variance of the chain's state around the true value of the target R in Fig.5a when using the FUSS algorithm. Let us remark that the conditional distribution of R in this example is univariate and with a very narrow peak, so having all the samples concentrated around the true value of R is the desired behaviour. On the other hand, the variance of the chain's state is much larger when using MH-within-Gibbs, Fig. 5b–d, and the mean value is not equal to the true value of R (especially when σ increases). This explains the poor performance shown in Table 9.

6.4 Localization in WSN and tuning of the network

In this second practical example, we consider the problem of localizing a target in $\mathbb{R}^2$ using range-only measurements in a wireless sensor network (WSN) [357, 358]. We assume that the measurements are contaminated by noise with an unknown power, which can be different for each sensor. This situation is common in several practical scenarios. The noise perturbation of each of the sensors can vary with the time and depends on the location of the sensor (due to manufacturing defects, obstacles in the reception, different physical environmental conditions, etc.). More specifically, let us denote the target position using the random vector $\mathbf{Z} = [Z_1, Z_2]^\top$. The position of the target is then a specific realization $\mathbf{z}$. The range measurements are obtained from $N_S = 6$ sensors located at $\mathbf{h}_1 = [3, -8]^\top$, $\mathbf{h}_2 = [8, 10]^\top$, $\mathbf{h}_3 = [-4, -6]^\top$,

[28] The behaviour of the trace plots for the other parameters (not shown) is similar.

Table 10 Results of the GMS algorithm for the example in Section 6.4

MSE	1.30	1.24	1.22	1.21	1.22	**1.19**	1.31	**1.44**
N	10	20	50	100	200	500	1000	2000
T	1000	500	200	100	50	20	10	5
E	$MT = 10^4$							
MSE range	**Min MSE= 1.19** ——— **Max MSE= 1.44**							

$\mathbf{h}_4 = [-8, 1]^\top$, $\mathbf{h}_5 = [10, 0]^\top$ and $\mathbf{h}_6 = [0, 10]^\top$, as shown in Fig. 6.

The observation model is

$$Y_j = 20 \log\left(||\mathbf{z} - \mathbf{h}_j||\right) + B_j, \quad j = 1, \ldots, N_S, \tag{161}$$

where the B_j are independent Gaussian random variables with PDFs $\mathcal{N}(b_j; 0, \lambda_j^2)$ for $j = 1, \ldots, N_S$. We use $\boldsymbol{\lambda} = [\lambda_1, \ldots, \lambda_{N_S}]$ to denote the vector of standard deviations. Given the position of the target, $\mathbf{z}^* = [z_1^* = 2.5, z_2^* = 2.5]^\top$, and setting $\boldsymbol{\lambda}^* = [\lambda_1^* = 1, \lambda_2^* = 2, \lambda_3^* = 1, \lambda_4^* = 0.5, \lambda_5^* = 3, \lambda_6^* = 0.2]$, we generate $N_O = 20$ observations from each sensor according to the model in Eq. (161). Then, we finally obtain a measurement matrix $\mathbf{Y} = [y_{k,1}, \ldots, y_{k,N_S}] \in \mathbb{R}^{D_y}$, where $D_y = N_O N_S = 120$ for $k = 1, \ldots, N_O$. We consider a uniform prior $\mathcal{U}(\mathcal{R}_z)$ over the position $[z_1, z_2]^\top$ with $\mathcal{R}_z = [-30 \times 30]^2$, and a uniform prior over λ_j, so that $\boldsymbol{\lambda}$ has prior $\mathcal{U}(\mathcal{R}_\lambda)$ with $\mathcal{R}_\lambda = [0, 20]^{N_S}$. Thus, the posterior PDF is

$$\begin{aligned}\bar{\pi}(\boldsymbol{\theta}|\mathbf{Y}) &= \bar{\pi}(\mathbf{z}, \boldsymbol{\lambda}|\mathbf{Y}) \\ &= \left[\prod_{k=1}^{N_O}\prod_{j=1}^{N_S} \frac{1}{\sqrt{2\pi\lambda_j^2}} \exp\left(-\frac{1}{2\lambda_j^2}(y_{k,j} + 10\log\left(||\mathbf{z} - \mathbf{h}_j||\right)^2\right)\right] \mathbb{I}_z(\mathcal{R}_z)\mathbb{I}_\lambda(\mathcal{R}_\lambda),\end{aligned} \tag{162}$$

where $\boldsymbol{\theta} = [\mathbf{z}, \boldsymbol{\lambda}]^\top$ is the parameter vector to be inferred, of dimension $D_\theta = N_S + 2 = 8$, and $\mathbb{I}_c(\mathcal{R})$ is an indicator function: $\mathbb{I}_c(\mathcal{R}) = 1$ if $c \in \mathcal{R}$, $\mathbb{I}_c(\mathcal{R}) = 0$ otherwise.

Our goal is computing the minimum mean square error (MMSE) estimator, i.e., the expected value of the posterior $\bar{\pi}(\boldsymbol{\theta}|\mathbf{Y}) = \bar{\pi}(\mathbf{z}, \boldsymbol{\lambda}|\mathbf{Y})$. Since the MMSE estimator cannot be computed analytically, we apply Monte Carlo methods to approximate it. We compare the GMS algorithm, the corresponding MTM scheme, the AMIS technique, and N parallel MH chains with a random walk proposal PDF. For all of them we consider Gaussian proposal densities. For GMS and MTM, we set $q_t(\boldsymbol{\theta}|\boldsymbol{\mu}_{n,t}, \sigma^2\mathbf{I}) = \mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\mu}_t, \sigma^2\mathbf{I})$ where $\boldsymbol{\mu}_t$ is adapted by considering the empirical mean of the generated samples after a training period, $t \geq 0.2T$ [200], $\boldsymbol{\mu}_0 \sim \mathcal{U}([1, 5]^{D_\theta})$ and $\sigma = 1$. For AMIS, we have $q_t(\boldsymbol{\theta}|\boldsymbol{\mu}_t, \mathbf{C}_t) = \mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\mu}_t, \mathbf{C}_t)$, where $\boldsymbol{\mu}_t$ is as previously described (with $\boldsymbol{\mu}_0 \sim \mathcal{U}([1, 5]^{D_\theta})$) and $\mathbf{C}_t$ is also adapted using the empirical covariance matrix, starting with $\mathbf{C}_0 = 4\mathbf{I}$. We also test the use of N parallel MH chains (including the case $N = 1$, which corresponds to a single chain), with a Gaussian random-walk proposal PDF, $q_n(\boldsymbol{\mu}_{n,t}|\boldsymbol{\mu}_{n,t-1}, \sigma^2\mathbf{I}) = \mathcal{N}(\boldsymbol{\mu}_{n,t}|\boldsymbol{\mu}_{n,t-1}, \sigma^2\mathbf{I})$, and $\boldsymbol{\mu}_{n,0} \sim \mathcal{U}([1, 5]^D)$ for all n and $\sigma = 1$.

We fix the total number of evaluations of the posterior density as $E = MT = 10^4$. Note that the evaluation of the posterior is usually the most costly step in MC algorithms (AMIS has the additional cost of re-weighting all the samples at each iteration according to the deterministic mixture procedure [98]). Let us recall that T denotes the total number of iterations and M the number of samples drawn from each proposal at each iteration. We consider $\boldsymbol{\theta}^* = [\mathbf{z}^*, \boldsymbol{\lambda}^*]^\top$ as the ground-truth and compute the MSE in the estimation obtained with the different algorithms. The results, averaged over 500 independent runs, are provided in Tables 10, 11, and 12, as well as Fig. 7. Note that GMS outperforms AMIS for each a pair $\{M, T\}$ (keeping $E = MT = 10^4$ fixed) and also provides smaller MSE values than N parallel MH chains (the case $N = 1$ corresponds to a single longer chain). Figure 6b shows the MSE versus N for GMS and the corresponding MTM method. This figure confirms again the advantage of recycling the samples in an MTM scheme.

6.5 Spectral analysis

Many problems in science and engineering require dealing with a noisy multi-sinusoidal signal, whose general form is given by

$$y_c(\tau) = A_0 + \sum_{i=1}^{D_\theta} A_i \cos(2\pi f_i \tau + \phi_i) + r(\tau), \quad \tau \in \mathbb{R},$$

Table 11 Results of the AMIS method for the example in Section 6.4

MSE	1.58	1.57	1.53	1.48	1.42	**1.29**	1.48	**1.71**
N	10	20	50	100	200	500	1000	2000
T	1000	500	200	100	50	20	10	5
E	$NT = 10^4$							
MSE range	**Min MSE= 1.29** ——— **Max MSE= 1.71**							

Table 12 Results of N parallel MH chains with a random-walk proposal PDF for the example in Section 6.4

MSE	1.42	**1.31**	1.44	2.32	2.73	**3.21**	3.18	3.15
N	1	5	10	50	100	500	1000	2000
T	10^4	2000	1000	200	100	20	10	5
E	$NT = 10^4$							
MSE range	**Min MSE= 1.31**	———	**Max MSE=3.21**					

where A_0 is a constant term, D_θ is the number of sinusoids, $\{A_i\}_{i=1}^{D_\theta}$ is the set of amplitudes, $\{2\pi f_i\}_{i=1}^{D_\theta}$ are the frequencies, $\{\phi_i\}_{i=1}^{D_\theta}$ their phases, and $r(\tau)$ is an additive white Gaussian noise (AWGN) term. The estimation of the parameters of this signal is required by many applications in signal processing [359, 360], in control (where a multi-harmonic disturbance is often encountered in industrial plants) [361, 362] or in digital communications (where multiple narrowband interferers can be roughly modeled as sinusoidal signals) [363, 364]. Let us assume that we have L equispaced samples from $y_c(\tau)$, obtained discretizing $y_c(\tau)$ with a period $T_s < \frac{\pi}{\max_{1\le i\le D_\theta} 2\pi f_i}$ (in order to fulfill the sampling theorem [365]):

$$y[k] = A_0 + \sum_{i=1}^{D_\theta} A_i \cos(\Omega_i k + \phi_i) + r[k], \quad k = 1, \ldots, L,$$

where $y[k] = y_c(kT_s)$ for $k = 0, 1, \ldots, L-1$, $\Omega_i = 2\pi f_i T_s$ for $i = 1, \ldots, D_\theta$, and $r[k] \sim \mathcal{N}(0, \sigma_w^2)$. We apply parallel MH algorithms to provide an accurate estimate of the set of unknown frequencies, $\{\Omega_i\}_{i=1}^{D_\theta}$ or merely $\{f_i\}_{i=1}^{D_\theta}$. In order to maintain the notation used throughout the paper, we denote the vector of frequencies to be inferred as $\boldsymbol{\theta} \in \mathbb{R}^{D_\theta}$. Thus, considering the hyper-rectangular domain $\Theta = \left[0, \frac{1}{2}\right]^{D_\theta}$ (it is straightforward to note the periodicity outside Θ), and a uniform prior on Θ, the posterior distribution given K data is $\bar{\pi}(\boldsymbol{\theta}) \propto \exp(-V(\boldsymbol{\theta}))$, where

$$V(\theta_1, \ldots, \theta_{D_\theta}) = \frac{1}{2\sigma_w^2} \sum_{k=1}^{L} \left(y[k] - A_0 - \sum_{i=1}^{D_\theta} A_i \cos(\theta_i k + \phi_i) \right)^2 \mathbb{I}_\Theta(\boldsymbol{\theta}),$$

and we have used $\mathbb{I}_\Theta(\boldsymbol{\theta})$ to denote the indicator function such that $\mathbb{I}_\Theta(\boldsymbol{\theta}) = 1$ if $\boldsymbol{\theta} \in \Theta$ and $\mathbb{I}_\Theta(\boldsymbol{\theta}) = 0$ if $\boldsymbol{\theta} \notin \Theta$. Moreover, for the sake of simplicity we have also assumed that S and σ_w^2 are known. , we set $A_0 = 0$, $A_i = A = 1$ and $\phi_i = 0$[29]. Note that the problem is symmetric with respect to the hyperplane $\theta_1 = \theta_2 = \ldots = \theta_{D_\theta}$ (and, in general, multimodal). Bidimensional examples of $V(\boldsymbol{\theta}) = \log \pi(\boldsymbol{\theta})$ are depicted in Fig. 8. We apply the OMCMC method [232], where N parallel interacting MH chains are used, comparing it with N independent parallel MH chains (IPCs). The proposal densities are all Gaussian random-walks proposal PDFs with diagonal covariance matrices $\mathbf{C} = \sigma^2 \mathbf{I}$.

[29] Let us remark that the estimation of all these parameters would make the inference harder, but can be easily incorporated into our algorithm.

We set $\mathbf{f} = [f_1 = 0.1, f_2 = 0.3]^\top$ and generate $L = 10$ synthetic data from the model. Moreover, we set the total number of target evaluations for OMCMC to $E_T = M(N+1) \in \{2730, 5450, 10.9 \cdot 10^3\}$. For a fair comparison, we consider N independent parallel chains (IPCs) choosing T such that $E'_T = NT$ is equal to E_T, i.e., $E'_T = E_T$. We test different values of $\sigma \in [0.05, 0.5]$ and $N \in \{2, 5, 10\}$. We test several combinations of the number of chains (N) and epochs (M for OMCMC and T for IPCs), always keeping E_T fixed. The relative error (RE) in the estimation, averaged over 500 independent runs, is shown in Fig. 9. We can observe that O-MCMC (solid line) outperforms IPCs (dashed line), attaining lower REs. The performance becomes similar as the computational effort E_T grows, since the state space in the first experiment, $\Theta = [0, \frac{1}{2}]^2$, is small enough to allow for an exhaustive exploration of Θ by independent chains.

Finally, Fig. 10 shows two typical examples of trace plots for the estimation of frequency $f_2 = \frac{1}{3}$, as in Fig. 8a. In both cases, we use the OMCMC-MTM algorithm with $T_v = 2$ vertical steps of an MH algorithm, $T_h = 1$ horizontal steps of the MTM algorithm, and $E_T = 700$ target evaluations. Note the fast convergence of the algorithm to frequencies close to the true one. This is a particularly good result,

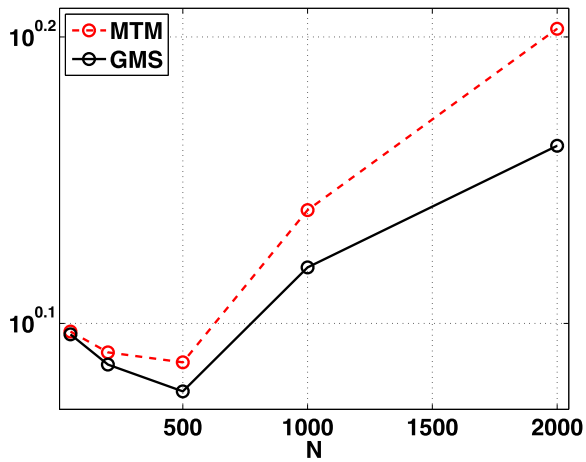
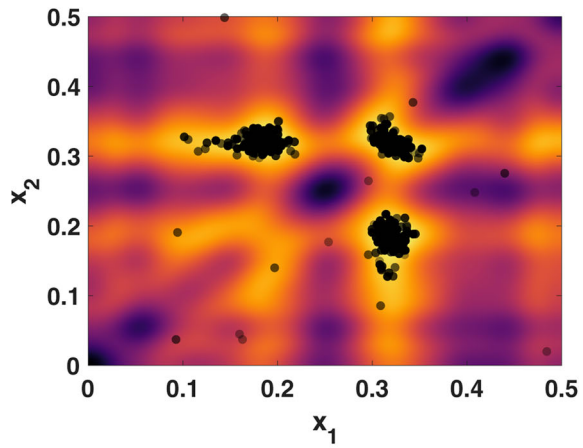
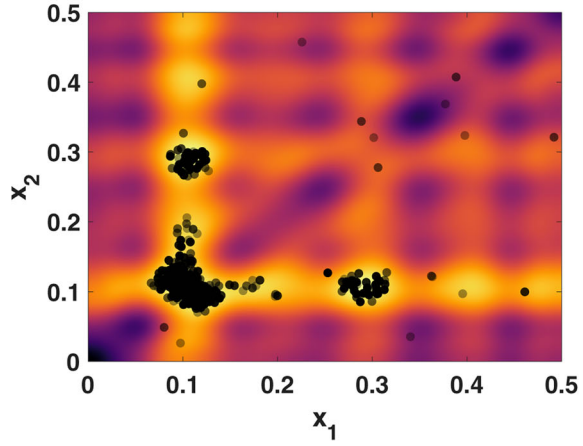


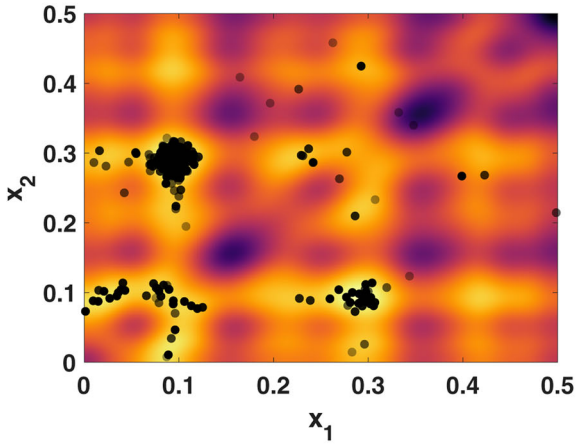
Fig. 7 MSE (log-scale) versus the number of candidates, $N \in \{50, 200, 500, 1000, 2000\}$, obtained by GMS and the corresponding MTM algorithm, for the example in Section 6.4. The total number of evaluations of the posterior PDF is fixed to $E = MT = 10^4$, so that $T \in \{200, 50, 20, 10, 5\}$



(a)

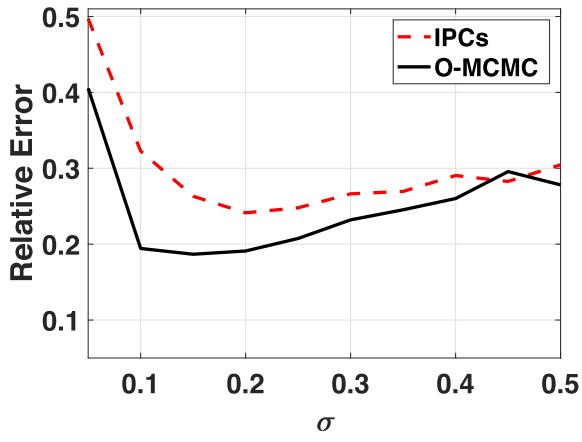


(b)

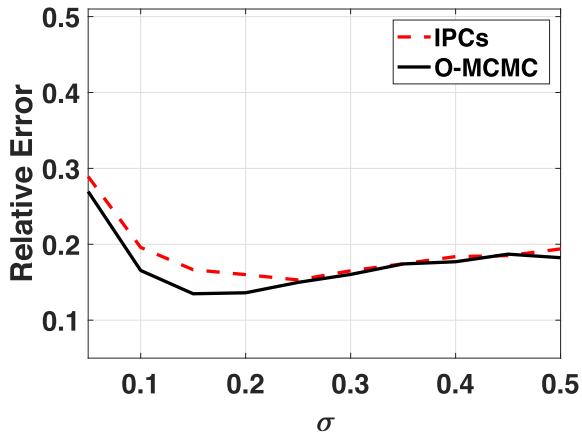


(c)

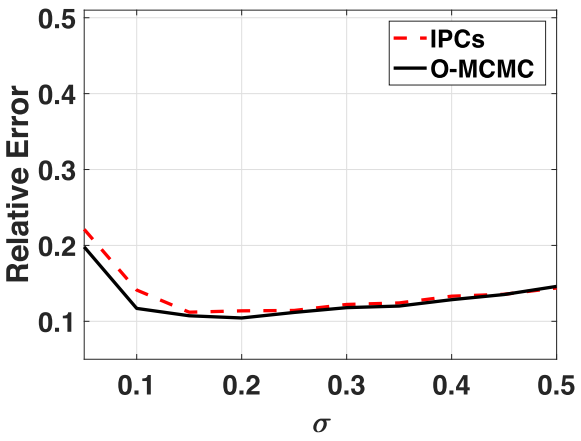
Fig. 8 Several examples of the function $V(\boldsymbol{\theta}) = \log \pi(\boldsymbol{\theta})$ with $D_\theta = 2$, $L = 10$, given different realizations of the measurements $y[1], \ldots, y[K]$. Black dotted points shows all the states generated throughout an O-MCMC run ($N = 10$ and $T = 500$)



(a)

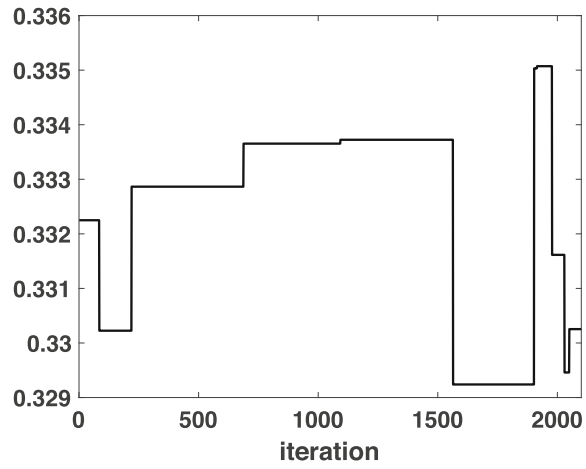


(b)

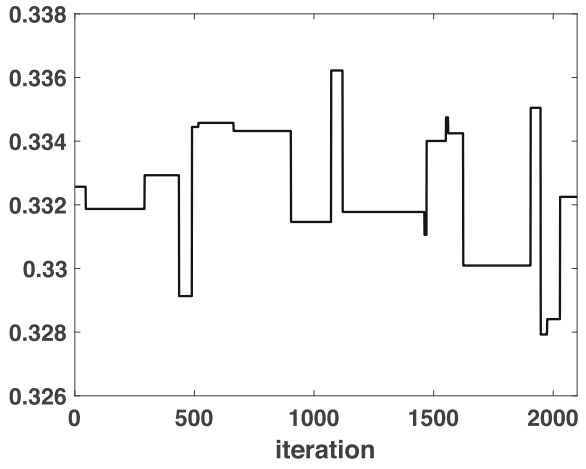


(c)

Fig. 9 Relative error (averaged over 500 runs) for OMCMC (solid line) and IPCs (dashed line) with different computational effort E_T



(a)



(b)

Fig. 10 Two typical trace plots for $f_2 = \frac{1}{3}$ in the example of Section 6.5 using the OMCMC-MTM algorithm with $T_v = 2$ vertical steps of an MH algorithm, $T_h = 1$ horizontal steps of the MTM algorithm, and $E_T = 700$ target evaluations

since the peaks of the target PDF in this case were very narrow.

7 Conclusion

In this paper, we have performed a review of Monte Carlo (MC) methods for the estimation of static parameters in statistical signal processing problems. MC methods are simulation-based techniques that are extensively used nowadays to perform approximate inference when analytical estimators cannot be computed, as it happens in many real-world signal processing applications. We have concentrated on the description of some of the most relevant methods available in the literature, rather than focusing on specific applications. Many different algorithms are provided throughout the text in a clear and unified format, so that signal processing practitioners can directly apply them in their specific problems.

In order to make the paper as self-contained as possible, we have started from scratch, describing first the MC method altogether with its convergence properties. Markov chain Monte Carlo (MCMC) techniques are considered first, starting with three classical MC methods (the Metropolis-Hastings (MH) algorithm, the Gibbs sampler, and MH-within-Gibbs) that are widely used by signal processing practitioners and can be considered as the basic building blocks of more recent approaches. Then, we detail several advanced MCMC algorithms, focusing on adaptive MCMC schemes (both using parametric and non-parametric proposals) and MCMC methods with multiple candidates. Although the focus of the paper is on MCMC methods, a brief description of importance sampling (IS) and adaptive importance sampling (AIS) methods is also included for the sake of completeness.

Two simple problems (where the analytical estimators can be computed and used to evaluate the performance of several MC methods), a challenging example that appears in several scientific fields (the estimation of the parameters of a chaotic system), and two classical signal processing applications (localization in a wireless sensor network and the spectral analysis of multiple sinusoids) are used to test many of the algorithms described.

Finally, let us remark that Monte Carlo methods can result in infinite variance estimators if not properly applied. As a cautionary note, let us mention Newton and Raftery's weighted likelihood bootstrap [366]. Although their approach leads to asymptotically unbiased estimators, it is also well-known that the variance of these estimators is infinite and thus practitioners may end up with estimated values which are very far away from the correct ones.

Acknowledgements

The authors gratefully acknowledge the support for their research: Ministerio de Economía y Competitividad (MINECO) of Spain under the TEC2015-64835-C3-3-R MIMOD-PLC project; Ministerio de Educación, Cultura y Deporte of Spain under CAS15/00350 grant; Universidad Politécnica de Madrid through a mobility grant for a short visit to Stony Brook University (D. Luengo); MINECO of Spain through Red de Excelencia KERMES TEC2016-81900-REDT (D. Luengo and L. Martino); the National Science Foundation under Award CCF-1617986 (M. F. Bugallo); the European Research Council (ERC) through the ERC Consolidator Grant SEDAL ERC-2014-CoG 647423 (L. Martino); the Academy of Finland project 266940 (S. Särkkä); the French National Research Agency through PISCES project ANR-17-CE40-0031-01 (V. Elvira); and from the French-American Fulbright Commission through the Fulbright scholar fellowship (V. Elvira). They also want to thank the reviewers for their many helpful comments that have contributed to substantially improve the quality of the paper.

Authors' contributions

David Luengo wrote the first version of Sections 1, 2, and 3.1. Simo Särkkä took care of part of Section 3. Luca Martino took care of part of Sections 3 and several parts of Sections 4 and 5. Víctor Elvira and Mónica Bugallo were in charge of the first part of Section 4. The numerical simulations in Section 5 were performed by Víctor Elvira and Luca Martino. Finally, all the authors revised the whole paper and contributed to all the sections. The authors read and approved the final manuscript.

Competing interests
The authors declare that they have no competing interests.

Author details
[1]Universidad Politécnica de Madrid, ETSIST, C/Nikola Tesla, s/n, 28031 Madrid, Spain. [2]Universitat de València, Valencia, Spain. [3]Universidad Carlos III de Madrid, Valencia, Spain. [4]Stony Brook University, Stony Brook (NY), USA. [5]CRIStAL (UMR 9189), IMT Lille Douai, Université de Lille, Lille, France. [6]Aalto University, Helsinki, Finland.

Received: 9 July 2017 Accepted: 10 March 2020
Published online: 29 May 2020

References

1. G. Casella, R. L. Berger, *Statistical Inference*. (Duxbury, Pacific Grove, CA (USA), 2002)
2. L. L. Scharf, *Statistical Signal Processing*. (Addison-Wesley, Reading, MA (USA), 1991)
3. S. M. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory*. (Prentice Hall, Upper Saddle River, NJ (USA), 1993)
4. H. L. Van Trees, *Detection, Estimation, and Modulation Theory, Part I: Detection, Estimation, and Linear Modulation Theory*. (Wiley, West Sussex (UK), 2004)
5. S. J. Godsill, P. J. W. Rayner, *Digital Audio Restoration, 1st edn.* (Springer, London (UK), 1998)
6. S. S. Haykin, *Blind Deconvolution*. (Prentice Hall, Englewood Cliffs, NJ (USA), 1994)
7. Z. Ding, Y. Li, *Blind Equalization and Identification*. (Marcel Dekker, New York, NY (USA), 2001)
8. P. Comon, C. Jutten, *Handbook of Blind Source Separation: Independent Component Analysis and Applications*. (Academic Press, Oxford (UK), 2010)
9. F. Zhao, L. J. Guibas, *Wireless Sensor Networks: An Information Processing Approach*. (Morgan Kaufmann, Amsterdam (The Netherlands), 2004)
10. A. Swami, Q. Zhao, Y.-W. Hong, L. Tong, *Wireless Sensor Networks: Signal Processing and Communications*. (John Wiley & Sons, West Sussex (UK), 2007)
11. J. M. Bernardo, A. F. M. Smith, *Bayesian theory*. (IOP Publishing, 2001)
12. M. Evans, T. Swartz, Methods for approximating integrals in statistics with special emphasis on Bayesian integration problems. Stat. Sci., 254–272 (1995). https://doi.org/10.1214/ss/1177009938
13. D. G. Tzikas, A. C. Likas, N. P. Galatsanos, The variational approximation for Bayesian inference. IEEE Signal Process. Mag. **25**(6), 131–146 (2008)
14. C. W. Fox, S. J. Roberts, A tutorial on variational Bayesian inference. Artif. Intell. Rev. **38**(2), 85–95 (2012)
15. R. Martin, *The St. Petersburg Paradox*. (Stanford Encyclopedia of Philosophy, 2011)
16. C.omte.d.e.B.uffon. Georges-Louis Leclerc, *Solutions des problémes qui regardoient le jeu du franc carreau*. (Histoire de l'Acadèmie Royale des Sciences de Paris, 1733), pp. 43–45
17. C.omte.d.e.B.uffon. Georges-Louis Leclerc, Essai d'arithmétique morale. Supplèment á l'Histoire Naturelle. **4**, 46–148 (1777)
18. P. S. de Laplace, *Théorie Analytique des Probabilités, vol. 7*. (Courcier, Paris (France), 1820)
19. A. De Morgan, *A Budget of Paradoxes*. (Longmans Green, Chicago, IL (USA), 1872)
20. A. Hall, On an experimental determination of π. Messenger Math. **2**, 113–4 (1873)
21. M. Lazzarini, Un'applicazione del calcolo della probabilità. Periodico di Matematico. **2**, 140–143 (1902)
22. H. Riedwyl, Rudolf Wolf's contribution to the Buffon needle problem (an early Monte Carlo experiment) and application of least squares. Am. Stat. **44**(2), 138–139 (1990)
23. T. Gridgeman, Geometric probability and the number π. Scr. Mathematika. **25**(3), 183–195 (1960)
24. T. H. O'Beirne, *Puzzles and Paradoxes*. (Oxford University Press, New York, NY (USA), 1965)
25. L. Badger, Lazzarini's lucky approximation of π. Math. Mag. **67**(2), 83–91 (1994)
26. S. M. Stigler, Stochastic simulation in the nineteenth century. Stat. Sci. **6**(1), 89–97 (1991)
27. W. S. (.Student") Gosset, The probable error of a mean. Biometrika, 1–25 (1908). https://doi.org/10.2307/2331554
28. W. S. (.Student") Gosset, Probable error of a correlation coefficient. Biometrika, 302–310 (1908). https://doi.org/10.2307/2331474
29. H. Hotelling, British statistics and statisticians today. J. Am. Stat. Assoc. **25**(170), 186–190 (1930)
30. S. L. Zabell, On Student's 1908 Article "The Probable Error of a Mean". J. Am. Stat. Assoc. **103**(481), 1–7 (2008)
31. L. H. C. Tippett, On the extreme individuals and the range of samples taken from a normal population. Biometrika. **17**(3-4), 364–387 (1925)
32. L. H. C. Tippett, Random sampling numbers (1927)
33. F. Galton, Dice for statistical experiments (1890). https://doi.org/10.1038/042013a0
34. E. Segrè, *From X-rays to Quarks: Modern Physicists and Their Discoveries*. (WH Freeman, San Francisco, CA (USA), 1980)
35. H. L. Anderson, Metropolis, Monte Carlo and the MANIAC. Los Alamos Sci. **14**, 96–107 (1986)
36. N. Metropolis, The beginning of the Monte Carlo method. Los Alamos Sci. **15**, 125–130 (1987)
37. S. I. Gass, A. A. Assad, Model world: tales from the time line—the definition of OR and the origins of Monte Carlo simulation. Interfaces. **35**(5), 429–435 (2005)
38. R. Eckhardt, Stan Ulam, John von Neumann, and the Monte Carlo method. Los Alamos Sci. **15**(131–136), 30 (1987)
39. N. Metropolis, S. Ulam, The Monte Carlo method. J. Am. Stat. Assoc. **44**(247), 335–341 (1949)
40. C. C. Hurd, A note on early Monte Carlo computations and scientific meetings. Ann. Hist. Comput. **7**(2), 141–155 (1985)
41. J. von Neumann, Various techniques used in connection with random digits. Natl. Bur. Stand. **12**, 36–38 (1951)
42. D. H. Lehmer, in *Proceedings of the 2nd Symposium on Large-Scale Digital Calculating Machinery*. Mathematical methods in large-scale computing units (Harvard University Press, 1951), pp. 141–146
43. N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, E. Teller, Equation of state calculations by fast computing machines. J. Chem. Phys. **21**(6), 1087–1092 (1953)
44. W. K. Hastings, Monte Carlo sampling methods using Markov chains and their applications. Biometrika. **57**(1), 97–109 (1970)
45. A. A. Barker, Monte Carlo calculation of the radial distribution functions for a proton–electron plasma. Aust. J. Phys. **18**, 119–133 (1965)
46. P. H. Peskun, Optimum Monte-Carlo sampling using Markov chains. Biometrika. **60**(3), 607–612 (1973)
47. D. B. Hitchcock, A history of the Metropolis–Hastings algorithm. Am. Stat. **57**(4), 254–257 (2003)
48. S. Geman, D. Geman, Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. IEEE Trans. Pattern Anal. Mach. Intell. **6**, 721–741 (1984). https://doi.org/10.1016/b978-0-08-051581-6.50057-x
49. A. E. Gelfand, A. F. M. Smith, Sampling-based approaches to calculating marginal densities. J. Am. Stat. Assoc. **85**(410), 398–409 (1990)
50. A. Gelman, in *Computing Science and Statistics: Proceedings of the 24th Symposium on the Interface*. Iterative and non-iterative simulation algorithms, (1992), pp. 433–438
51. L. Tierney, Markov chains for exploring posterior distributions. Ann. Stat., 1701–1728 (1994). https://doi.org/10.1214/aos/1176325750
52. G. Casella, E. I. George, Explaining the Gibbs sampler. Am. Stat. **46**(3), 167–174 (1992)
53. S. Chib, E. Greenberg, Understanding the Metropolis-Hastings algorithm. Am. Stat. **49**(4), 327–335 (1995)
54. W. R. Gilks, D. G. Clayton, D. J. Spiegelhalter, N. G. Best, A. J. McNeil, Modelling complexity: applications of Gibbs sampling in medicine. J. R. Stat. Soc. Ser. B (Methodol.), 39–52 (1993)
55. J. Geweke, *Bayesian comparison of econometric models. Technical report, Working Paper*. (Federal Reserve Bank of Minneapolis, Minnesota, 1994)
56. A. Gelman, D. B. Rubin, Markov chain Monte Carlo methods in biostatistics. Stat. Methods Med. Res. **5**(4), 339–355 (1996)
57. B. Mau, M. A. Newton, B. Larget, Bayesian phylogenetic inference via Markov chain Monte Carlo methods. Biometrics. **55**(1), 1–12 (1999)
58. J. Dongarra, F. Sullivan, Guest editors' introduction: the top 10 algorithms. Comput. Sci. Eng. **2**(1), 22–23 (2000)

59. T.-H. Li, in *Proceedings of the 27th Asilomar Conference on Signals, Systems and Computers*. Blind deconvolution of discrete-valued signals, (1993), pp. 1240–1244. https://doi.org/10.1109/acssc.1993.342374
60. R. Chen, T.-H. Li, Blind restoration of linearly degraded discrete signals by Gibbs sampling. IEEE Trans. Signal Process. **43**(10), 2410–2413 (1995)
61. A. Doucet, P. Duvaut, in *Proceedings of the 8th European Signal Processing Conference (EUSIPCO)*. Fully Bayesian analysis of hidden Markov models, (1996), pp. 1–4
62. T. Clapp, S. Godsill, in *Proceedings of the IEE Colloquium on Adaptive Signal Processing for Mobile Communication Systems*. Bayesian blind deconvolution for mobile communications, (1997), pp. 1–9. https://doi.org/10.1049/ic:19971315
63. O. Cappé, A. Doucet, M. Lavielle, E. Moulines, Simulation-based methods for blind maximum-likelihood filter identification. Signal Process. **73**(1), 3–25 (1999)
64. J. J. K. Ó Ruanaidh, W. J. Fitzgerald, Interpolation of missing samples for audio restoration. IEE Electron. Lett. **30**(8), 622–623 (1994)
65. S. J. Godsill, P. J. W. Rayner, in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), vol. 2*. Robust noise reduction for speech and audio signals, (1996), pp. 625–628. https://doi.org/10.1109/icassp.1996.543198
66. S. J. Godsill, Bayesian enhancement of speech and audio signals which can be modelled as ARMA processes. Int. Stat. Rev. **65**(1), 1–21 (1997)
67. C. A. Bouman, K. Sauer, S. S. Saquib, in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), vol. 5*. Tractable models and efficient algorithms for Bayesian tomography, (1995), pp. 2907–2910. https://doi.org/10.1109/icassp.1995.479453
68. S. S. Saquib, C. A. Bouman, K. Sauer, ML parameter estimation for Markov random fields with applications to Bayesian tomography. IEEE Trans. Image Process. **7**(7), 1029–1044 (1998)
69. J. J. Rajan, W. Rayner, Parameter estimation of time-varying autoregressive models using the Gibbs sampler. IEE Electron. Lett. **31**(13), 1035–1036 (1995)
70. J. J. Rajan, P. J. W. Rayner, S. J. Godsill, Bayesian approach to parameter estimation and interpolation of time-varying autoregressive processes using the Gibbs sampler. IEE Proc.-Vis. Image Signal Process. **144**(4), 249–256 (1997)
71. W. J. Fitzgerald, Markov chain Monte Carlo methods with applications to signal processing. Signal Process. **81**(1), 3–18 (2001)
72. P. M. Djuric, S. J. Goodsill, Guest editorial special issue on Monte Carlo methods for statistical signal processing. IEEE Trans. Signal Process. **50**(2), 173–173 (2002)
73. X. Wang, R. Chen, J. S. Liu, Monte Carlo Bayesian signal processing for wireless communications. J. VLSI Signal Process. **30**(1), 89–105 (2002)
74. C. Andrieu, N. De Freitas, A. Doucet, M. I. Jordan, An introduction to MCMC for machine learning. Mach. Learn. **50**(1), 5–43 (2003)
75. A. Doucet, X. Wang, Monte Carlo methods for signal processing: a review in the statistical signal processing context. IEEE Signal Process. Mag. **22**(6), 152–170 (2005)
76. H. Kahn, Random sampling (Monte Carlo) techniques in neutron attenuation problems. Nucleonics. **6**(5), 27–37 (1950)
77. H. Kahn, A. W. Marshall, Methods of reducing sample size in Monte Carlo computations. J. Oper. Res. Soc. Am. **1**(5), 263–278 (1953)
78. W. A. Moy, Sampling techniques for increasing the efficiency of simulations of queuing systems. PhD thesis (1965)
79. K. P. N. Murthy, R. Indira, Analytical results of variance reduction characteristics of biased Monte Carlo for deep-penetration problems. Nucl. Sci. Eng. **92**(3), 482–487 (1986)
80. T. E. Booth, A Monte Carlo learning/biasing experiment with intelligent random numbers. Nucl. Sci. Eng. **92**(3), 465–481 (1986)
81. B. R. Davis, An improved importance sampling method for digital communication system simulations. IEEE Trans. Commun. **34**(7), 715–719 (1986)
82. P. H. Hahn, M. C. Jeruchim, Developments in the theory and application of importance sampling. IEEE Trans. Commun. **35**(7), 706–714 (1987)
83. A. E. Conway, A. Goyal, in *Proceedings of the 17th Symposium on Fault-Tolerant Computing*. Monte Carlo simulation of computer system availability/reliability models, (Pittsburg, Pennsylvania (USA), 1986), pp. 230–235
84. T. C. Hesterberg, in *Proceedings of the Statistical Computing Section, American Statistical Association 1987 Meeting*. Importance sampling in multivariate problems, (1987), pp. 412–417
85. T. Kloek, H. K. V. Dijk, Bayesian estimates of equation system parameters: an application of integration by Monte Carlo. Econometrica J. Econometric Soc., 1–19 (1978). https://doi.org/10.2307/1913641
86. L. Stewart, Multiparameter univariate Bayesian analysis. J. Am. Stat. Assoc. **74**(367), 684–693 (1979)
87. T. C. Hesterberg, in *Computing and Graphics in Statistics, vol. 36*, ed. by A. Buja, P. Tukey. Importance sampling for Bayesian estimation (Springer, New York, NY (USA), 1991), p. 63
88. T. C. Hesterberg, *Weighted Average Importance Sampling and Defensive Mixture Distributions*. (Stanford University. Division of Biostatistics, Stanford, CA (USA), 1991)
89. T. C. Hesterberg, Weighted average importance sampling and defensive mixture distributions. Technometrics. **37**(2), 185–194 (1995)
90. E. Veach, L. J. Guibas, in *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*. Optimally combining sampling techniques for Monte Carlo rendering, (1995), pp. 419–428. https://doi.org/10.1145/218380.218498
91. A. Owen, Y. Zhou, Safe and effective importance sampling. J. Am. Stat. Assoc. **95**(449), 135–143 (2000)
92. N. J. Gordon, D. J. Salmond, A. F. M. Smith, in *IEE Proceedings F (Radar and Signal Processing), vol. 140*. Novel approach to nonlinear/non-Gaussian Bayesian state estimation, (1993), pp. 107–113. https://doi.org/10.1049/ip-f-2.1993.0015
93. A. Doucet, A. M. Johansen, A tutorial on particle filtering and smoothing: Fifteen years later. Handb. Nonlinear Filtering. **12**(656-704), 3 (2009)
94. Y. Iba, Population Monte Carlo algorithms. Trans. Japan. Soc. Artif. Intell. **16**(2), 279–286 (2001)
95. O. Cappé, A. Guillin, J. M. Marin, C. P. Robert, Population Monte Carlo. J. Comput. Graphical Stat. **13**(4), 907–929 (2004)
96. O. Cappé, R. Douc, A. Guillin, J. M. Marin, C. P. Robert, Adaptive importance sampling in general mixture classes. Stat. Comput. **18**, 447–459 (2008)
97. V. Elvira, L. Martino, D. Luengo, M. F. Bugallo, Improving population Monte Carlo: alternative weighting and resampling schemes. Signal Process. **131**, 77–91 (2017)
98. J. M. Cornuet, J. M. Marin, A. Mira, C. P. Robert, Adaptive multiple importance sampling. Scand. J. Stat. **39**(4), 798–812 (2012)
99. L. Martino, V. Elvira, D. Luengo, J. Corander, An adaptive population importance sampler: Learning from the uncertainty. IEEE Trans. Signal Process. **63**(16), 4422–4437 (2015)
100. C. Andrieu, G. O. Roberts, et al., The pseudo-marginal approach for efficient monte carlo computations. Ann. Stat. **37**(2), 697–725 (2009)
101. J.-M. Marin, P. Pudlo, C. P. Robert, R. J. Ryder, Approximate Bayesian computational methods. Stat. Comput., 1–14 (2012)
102. M. Sunnåker, A. G. Busetto, E. Numminen, J. Corander, M. Foll, C. Dessimoz, Approximate Bayesian computation. PLoS Comput. Biol. **9**(1), 1002803 (2013)
103. J. Zhu, J. Chen, W. Hu, Big learning with Bayesian methods (2014). arXiv:1411.6370
104. C. P. Robert, G. Casella, *Monte Carlo Statistical Methods*. (Springer, New York, NY (USA), 2004)
105. D. Gamerman, H. F. Lopes, *Markov Chain Monte Carlo: Stochastic Simulation for Bayesian Inference*. (Chapman and Hall/CRC, Boca Raton, FL (USA), 2006)
106. J. S. Liu, *Monte Carlo Strategies in Scientific Computing*. (Springer, New York, NY (USA), 2004)
107. F. Liang, C. Liu, R. Carroll, *Advanced Markov Chain Monte Carlo Methods: Learning from Past Samples, vol. 714*. (Wiley, West Sussex (UK), 2011)
108. S. Brooks, A. Gelman, G. Jones, X.-L. Meng, *Handbook of Markov Chain Monte Carlo*. (CRC press, 2011)
109. D. P. Kroese, T. Taimre, Z. I. Botev, *Handbook of Monte Carlo Methods, vol. 706*. (Wiley, Hoboken, NJ (USA), 2013)
110. B. Ristic, S. Arulampalam, N. Gordon, *Beyond the Kalman Filter: Particle Filters for Tracking Applications*. (Artech House, Boston, MA (USA), 2003)
111. B. Ristic, *Particle Filters for Random Set Models, vol. 798*. (Springer, New York, NY (USA), 2013)

112. J. V. Candy, *Bayesian Signal Processing: Classical, Modern, and Particle Filtering Methods, vol. 54*. (Wiley, Hoboken, NJ (USA), 2016)
113. A. Doucet, N. De Freitas, N. Gordon, *Sequential Monte Carlo Methods in Practice*. (Springer, New York, NY (USA), 2001), pp. 3–14
114. S. Särkkä, *Bayesian Filtering and Smoothing, vol. 3*. (Cambridge University Press, Cambridge (UK), 2013)
115. L. Devroye. Non-uniform random variate generation (Springer, New York, NY (USA), 1986)
116. J. Dagpunar, *Principles of Random Variate Generation*. (Clarendon Press (Oxford and New York), New York, 1988)
117. W. Hörmann, J. Leydold, G. Derflinger, *Automatic Nonuniform Random Variate Generation*. (Springer, New York, NY (USA), 2003)
118. J. E. Gentle, *Random Number Generation and Monte Carlo Methods*. (Springer, New York, NY (USA), 2004)
119. L. Martino, D. Luengo, J. Míguez, *Independent Random Sampling Methods*. (Springer, Germany, 2018)
120. P. M. Djurić, in *12th European Signal Processing Conference (EUSIPCO)*. Monte Carlo methods for signal processing: recent advances (IEEE, 2004), pp. 853–860
121. P. J. Green, K. Łatuszyński, M. Pereyra, C. P. Robert, Bayesian computation: a summary of the current state, and samples backwards and forwards. Stat. Comput. **25**(4), 835–862 (2015)
122. L. Martino, A review of multiple try MCMC algorithms for signal processing. Digital Signal Process. (2018). https://doi.org/10.1016/j.dsp.2018.01.004
123. C. P. Robert, V. Elvira, N. Tawn, C. Wu, Accelerating mcmc algorithms (2018). arXiv:1804.02719
124. M. F. Bugallo, L. Martino, J. Corander, Adaptive importance sampling in signal processing. Digit. Signal Process. **47**, 36–49 (2015)
125. L. Martino, V. Elvira, D. Luengo, F. Louzada, Adaptive population importance samplers: a general perspective. IEEE Sensor Array Multichannel Signal Process. Work. (SAM), 1–5 (2016). https://doi.org/10.1109/sam.2016.7569668
126. M. F. Bugallo, V. Elvira, L. Martino, D. Luengo, J. Míguez, P. Djuric, Adaptive importance sampling. The past, the present, and the future. IEEE Signal Process. Mag. **34**(4), 60–79 (2017)
127. D. J. Lunn, A. Thomas, N. Best, D. Spiegelhalter, Winbugs-a bayesian modelling framework: concepts, structure, and extensibility. Stat. Comput. **10**(4), 325–337 (2000)
128. D. Lunn, D. Spiegelhalter, A. Thomas, N. Best, The bugs project: evolution, critique and future directions. Stat. Med. **28**(25), 3049–3067 (2009)
129. D. Lunn, C. Jackson, N. Best, A. Thomas, D. Spiegelhalter, *The BUGS Book: A Practical Introduction to Bayesian Analysis*. (CRC press, Boca Raton, FL (USA), 2012)
130. K. Plummer, in *Proceedings of DSC, vol. 2*. Jags: A program for analysis of bayesian graphical models using gibbs sampling, (2003), pp. 1–1
131. A. Gelman, D. Lee, J. Guo, Stan: A probabilistic programming language for bayesian inference and optimization. J. Educ. Behav. Stat. **40**(5), 530–543 (2015)
132. B. Carpenter, A. Gelman, M. D. Hoffman, D. Lee, B. Goodrich, M. Betancourt, M. Brubaker, J. Guo, P. Li, A. Riddell, Stan: A probabilistic programming language. J. Stat. Softw. **76**(1) (2017)
133. M. F. Bugallo, L. Martino, J. Corander, Adaptive importance sampling in signal processing. Digit. Signal Process. **47**, 36–49 (2015)
134. J. H. Ahrens, U. Dieter, Computer methods for sampling from gamma, beta, Poisson and binomial distributions. Computing. **12**, 223–246 (1974)
135. P. Damien, S. G. Walker, Sampling truncated normal, beta, and gamma densities. J. Comput. Graph. Stat. **10**(2), 206–215 (2001)
136. R. C. H. Cheng, The generation of gamma variables with non-integral shape parameter. J. R. Stat. Soc. Ser. C (Appl. Stat.) **26**, 71–75 (1977)
137. N. C. Beaulieu, C. Cheng, Efficient Nakagami-*m* fading channel simulation. IEEE Trans. Veh. Technol. **54**(2), 413–424 (2005)
138. D. Luengo, L. Martino, Almost rejectionless sampling from Nakagami-*m* distributions ($m \geq 1$). IET Electron. Lett. **48**(24), 1559–1561 (2012)
139. Q. M. Zhu, X. Y. Dang, D. Z. Xu, X. M. Chen, Highly efficient rejection method for generating Nakagami-*m* sequences. IET Electron. Lett. **47**(19), 1100–1101 (2011)
140. L. Martino, D. Luengo, Extremely efficient acceptance-rejection method for simulating uncorrelated Nakagami fading channels. (To appear) Commun. Stat. - Simul. Comput., 1–20 (2018). https://doi.org/10.1080/03610918.2018.1423694
141. W. R. Gilks, P. Wild, Adaptive rejection sampling for Gibbs sampling. Appl. Stat. **41**(2), 337–348 (1992)
142. C. Botts, W. Hörmann, J. Leydold, Transformed density rejection with inflection points. Stat. Comput. **23**, 251–260 (2013)
143. L. Martino, J. Míguez, in *Proc. of the 34th IEEE ICASSP*. A novel rejection sampling scheme for posterior probability distributions, (2009). https://doi.org/10.1109/icassp.2009.4960235
144. W. Hörmann, A rejection technique for sampling from T-concave distributions. ACM Trans. Math. Softw. **21**(2), 182–193 (1995)
145. L. Martino, J. Míguez, Generalized rejection sampling schemes and applications in signal processing. Signal Process. **90**(11), 2981–2995 (2010)
146. D. Görür, Y. W. Teh, Concave convex adaptive rejection sampling. J. Comput. Graph. Stat. **20**(3), 670–691 (2011)
147. L. Martino, J. Míguez, A generalization of the adaptive rejection sampling algorithm. Stat. Comput. **21**(4), 633–647 (2011)
148. M. Evans, T. Swartz, Random variate generation using concavity properties of transformed densities. J. Comput. Graph. Stat. **7**(4), 514–528 (1998)
149. L. Martino, F. Louzada, Adaptive rejection sampling with fixed number of nodes. (To appear) Commun. Stat. Simul. Comput. (2017). https://doi.org/10.1080/03610918.2017.1395039
150. L. Martino, Parsimonious adaptive rejection sampling. IET Electron. Lett. **53**(16), 1115–1117 (2017)
151. W. Hörmann, A universal generator for bivariate log-concave distributions. Computing. **52**, 89–96 (1995)
152. L. Martino, V. Elvira, Metropolis Sampling (2017)
153. M. K. Cowles, B. P. Carlin, Markov chain Monte Carlo convergence diagnostics: a comparative review. J. Am. Stat. Assoc. **91**(434), 883–904 (1996)
154. G. O. Roberts, A. Gelman, W. R. Gilks, et al., Weak convergence and optimal scaling of random walk metropolis algorithms. Ann. Appl. Probab. **7**(1), 110–120 (1997)
155. A. Gelman, G. O. Roberts, W. R. Gilks, *et al*, Efficient metropolis jumping rules. Bayesian Stat. **5**(599-608), 42 (1996)
156. M. Bedard, J. Rosenthal, Optimal scaling of metropolis algorithms: is 0.234 as robust as is believed? Technical report (2007)
157. M. Bedard, Optimal acceptance rates for metropolis algorithms: moving beyond 0.234. Stoch. Process. Appl. **118**(12), 2198–2222 (2008)
158. M. Bédard, Efficient sampling using metropolis algorithms: applications of optimal scaling results. J. Comput. Graph. Stat. **17**(2), 312–332 (2008)
159. A. Beskos, G. Roberts, A. Stuart, et al., Optimal scalings for local Metropolis–Hastings chains on nonproduct targets in high dimensions. Ann. Appl. Probab. **19**(3), 863–898 (2009)
160. N. S. Pillai, A. M. Stuart, A. H. Thiéry, et al., Optimal scaling and diffusion limits for the langevin algorithm in high dimensions. Ann. Appl. Probab. **22**(6), 2320–2356 (2012)
161. Y. Guan, R. Fleißner, P. Joyce, S. M. Krone, Markov chain Monte Carlo in small worlds. Stat. Comput. **16**(2), 193–202 (2006)
162. Y. Guan, S. M. Krone, Small-world MCMC and convergence to multi-modal distributions: From slow mixing to fast mixing. Ann. Appl. Probab. **17**(1), 284–304 (2007)
163. K. L. Mengersen, R. L. Tweedie, Rates of convergence of the Hastings and Metropolis algorithms. Ann. Stat. **24**(1), 101–121 (1996)
164. G. O. Roberts, R. L. Tweedie, Geometric convergence and central limit theorems for multidimensional Hastings and Metropolis algorithms. Biometrika. **83**(1), 95–110 (1996)
165. S. Kirkpatrick, C. D. Gelatt, M. P. Vecchi, Optimization by simulated annealing. Science. **220**(4598), 671–680 (1983)
166. G. O. Roberts, J. S. Rosenthal, Surprising convergence properties of some simple gibbs samplers under various scans. Int. J. Stat. Probab. **5**(1), 51 (2015)
167. J. S. Liu, *Metropolized Gibbs sampler: an improvement. Technical report*. (Department of Statistics, Stanford University, 1996)
168. L. Martino, V. Elvira, G. Camps-Valls, The Recycling Gibbs Sampler for efficient learning. Digit. Signal Process. **74**, 1–13 (2018)
169. M. J. Schervish, B. P. Carlin, On the convergence of successive substitution sampling. J. Comput. Graph. Stat. **1**(2), 111–127 (1992)
170. J. S. Liu, The collapsed Gibbs sampler in Bayesian computations with applications to a gene regulation problem. J. Am. Stat. Assoc. **89**(427), 958–966 (1994)

171. A. Terenin, M. Magnusson, L. Jonsson, D. Draper, Polya urn latent Dirichlet allocation: a doubly sparse massively parallel sampler. IEEE Trans. Pattern Anal. Mach. Intell. (2018). https://doi.org/10.1109/tpami.2018.2832641
172. S. L. Zeger, M. R. Karim, Generalized linear models with random effects; a Gibbs sampling approach. J. Am. Stat. Assoc. **86**(413), 79–86 (1991)
173. C. Ritter, M. A. Tanner, *The griddy Gibbs sampler. Technical report*. (Department of Statistics, University of Wisconsin, 1991)
174. C. Ritter, M. A. Tanner, Facilitating the Gibbs sampler: the Gibbs stopper and the griddy-Gibbs sampler. J. Am. Stat. Assoc. **87**(419), 861–868 (1992)
175. J. Geweke, H. Tanizaki, On Markov chain Monte Carlo methods for nonlinear and non-Gaussian state-space models. Commun. Stat. Simul. Comput. **28**(4), 867–894 (1999)
176. J. Geweke, H. Tanizaki, Bayesian estimation of state-space models using the Metropolis-Hastings algorithm within Gibbs sampling. Comput. Stat. Data Anal. **37**(2), 151–170 (2001)
177. L. Martino, J. Read, D. Luengo, Independent doubly adaptive rejection Metropolis sampling within Gibbs sampling. IEEE Trans. Signal Process. **63**(12), 3123–3138 (2015)
178. P. Neal, G. Roberts, et al., Optimal scaling for partially updating mcmc algorithms. Ann. Appl. Probab. **16**(2), 475–515 (2006)
179. L. Martino, H. Yang, D. Luengo, J. Kanniainen, J. Corander, A fast universal self-tuned sampler within Gibbs sampling. Digit. Signal Process. **47**, 68–83 (2015)
180. M. A. Tanner, W. H. Wong, The calculation of posterior distributions by data augmentation. J. Am. Stat. Assoc. **82**(398), 528–540 (1987)
181. A. P. Dempster, N. M. Laird, D. B. Rubin, Maximum likelihood from incomplete data via the EM algorithm. J. R. Stat. Soc. Ser. B (Methodol.), 1–38 (1977)
182. R. M. Neal, Slice sampling. Ann. Stat. **31**(3), 705–767 (2003)
183. A. Boneh, A. Golan, in *Proceedings of the 3rd European Congress on Operations Research (EURO III)*. Constraints' redundancy and feasible region boundedness by random feasible point generator (RFPG), (Amsterdam (Netherland), 1979)
184. R. L. Smith, in *Proceedings of the ORSA/TIMS Conference*. Monte Carlo techniques for generating random feasible solutions to mathematical programs, (Washington DC (USA), 1980)
185. R. L. Smith, Efficient Monte Carlo procedures for generating points uniformly distributed over bounded regions. Oper. Res. **32**(6), 1296–1308 (1984)
186. H. C. Berbee, C. G. Boender, A. H. Rinnooy Ran, C. Scheffer, R. L. Smith, J. Telgen, Hit-and-run algorithms for the identification of nonredundant linear inequalities. Math. Programm. **37**(2), 184–207 (1987)
187. D. E. Kaufman, R. L. Smith, Direction choice for accelerated convergence in hit-and-run sampling. Oper. Res. **46**(1), 84–95 (1998)
188. M.-H. Chen, B. Schmeiser, Performance of the Gibbs, hit-and-run, and Metropolis samplers. J. Comput. Graph. Stat. **2**(3), 251–272 (1993)
189. W. R. Gilks, G. O. Roberts, E. I. George, Adaptive direction sampling. Statistician, 179–189 (1994)
190. G. O. Roberts, W. R. Gilks, Convergence of adaptive direction sampling. J. Multivar. Anal. **49**(2), 287–298 (1994)
191. C. J. Geyer, Practical Markov chain Monte Carlo. Stat. Sci., 473–483 (1992)
192. S. N. MacEachern, L. M. Berliner, Subsampling the Gibbs sampler. Am. Stat. **48**(3), 188–190 (1994)
193. W. A. Link, M. J. Eaton, On thinning of chains in MCMC. Methods Ecol. Evol. **3**(1), 112–115 (2012)
194. A. B. Owen, Statistically efficient thinning of a Markov chain sampler (2015). arXiv:1510.07727
195. C. Andrieu, J. Thoms, A tutorial on adaptive MCMC. Stat. Comput. **18**(4), 343–373 (2008)
196. H. Haario, E. Saksman, J. Tamminen, An adaptive Metropolis algorithm. Bernoulli. **7**(2), 223–242 (2001)
197. A. Gelman, G. O. Roberts, W. R. Gilks, et al., Efficient Metropolis jumping rules. Bayesian Stat. **5**(599-608), 42 (1996)
198. H. Haario, M. Laine, A. Mira, E. Saksman, DRAM: efficient adaptive MCMC. Stat. Comput. **16**(4), 339–354 (2006)
199. P. Giordani, R. Kohn, Adaptive independent Metropolis–Hastings by fast estimation of mixtures of normals. J. Comput. Graph. Stat. **19**(2), 243–259 (2010)
200. D. Luengo, L. Martino, in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Fully adaptive Gaussian mixture Metropolis-Hastings algorithm, (2013), pp. 6148–6152. https://doi.org/10.1109/icassp.2013.6638846
201. I. S. Mbalawata, S. Särkkä, M. Vihola, H. Haario, Adaptive Metropolis algorithm using variational Bayesian adaptive Kalman filter. Comput. Stat. Data Anal. **83**, 101–115 (2015)
202. Y. Atchadé, G. Fort, Limit theorems for some adaptive MCMC algorithms with subgeometric kernels. Bernoulli. **16**(1), 116–154 (2010)
203. M. Vihola, On the stability and ergodicity of adaptive scaling Metropolis algorithms. Stoch. Process. Appl. **121**(12), 2839–2860 (2011)
204. M. Vihola, Robust adaptive Metropolis algorithm with coerced acceptance rate. Stat. Comput. **22**(5), 997–1008 (2012)
205. W. R. Gilks, N. G. Best, K. K. C. Tan, Adaptive Rejection Metropolis Sampling within Gibbs Sampling. Appl. Stat. **44**(4), 455–472 (1995)
206. R. Meyer, B. Cai, F. Perron, Adaptive rejection Metropolis sampling using Lagrange interpolation polynomials of degree 2. Comput. Stat. Data Anal. **52**(7), 3408–3423 (2008)
207. L. Martino, J. Read, D. Luengo, in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Independent doubly adaptive rejection Metropolis sampling, (2014), pp. 7998–8002. https://doi.org/10.1109/icassp.2014.6855158
208. L. Martino, R. Casarin, F. Leisen, D. Luengo, Adaptive Independent Sticky MCMC algorithms. (to appear) EURASIP J. Adv. Signal Process. (2017)
209. G. O. Roberts, J. S. Rosenthal, Coupling and ergodicity of adaptive Markov chain Monte Carlo algorithms. J. Appl. Probab. **44**(2), 458–475 (2007)
210. C. Chimisov, K. Latuszynski, G. Roberts, Air Markov chain Monte Carlo. Technical report, 46 (2018). https://arxiv.org/abs/1801.09309. Accessed 28 Jan 2018
211. G. O. Roberts, O. Stramer, Langevin diffusions and Metropolis-Hastings algorithms. Methodol. Comput. Appl. Probab. **4**, 337–357 (2002)
212. M. Girolami, B. Calderhead, Riemann manifold Langevin and Hamiltonian Monte Carlo methods. J. R. Stat. Soc. Ser. B (Stat. Methodol.) **73**(2), 123–214 (2011)
213. P. E. Kloeden, E. Platen, *Numerical Solution of Stochastic Differential Equations*. (Springer, New York, NY (USA), 1999)
214. G. O. Roberts, J. S. Rosenthal, Optimal scaling of discrete approximations to langevin diffusions. J. R. Stat. Soc. Ser. B (Stat. Methodol.) **60**(1), 255–268 (1998)
215. S. Duane, A. D. Kennedy, B. J. Pendleton, D. Roweth, Hybrid Monte Carlo. Phys. Lett. B. **195**(2), 216–222 (1987)
216. S. Brooks, A. Gelman, G. Jones, X.-L. Meng, *Handbook of Markov Chain Monte Carlo*. (CRC Press, Boca Raton, FL (USA), 2011)
217. R. M. Neal, in *Handbook of Markov Chain Monte Carlo, Chap. 5*. MCMC using Hamiltonian dynamics (CRC Press, Boca Raton, FL (USA), 2011)
218. A. Beskos, N. Pillai, G. Roberts, J.-M. Sanz-Serna, A. Stuart, et al., Optimal tuning of the hybrid monte carlo algorithm. Bernoulli. **19**(5A), 1501–1534 (2013)
219. M. Zlochin, Y. Baram, Manifold stochastic dynamics for Bayesian learning. Neural Comput. **13**(11), 2549–2572 (2001)
220. M. D. Hoffman, A. Gelman, The No-U-turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo. J. Mach. Learn. Res. **15**(1), 1593–1623 (2014)
221. L. Chen, Z. Qin, J. S. Liu, in *Proceedings of the International Symposium on Bayesian Analysis (ISBA)*. Exploring hybrid Monte Carlo in Bayesian computation, (2000)
222. T. Holder, B. Leimkuhler, S. Reich, Explicit variable step-size and time-reversible integration. Appl. Numer. Math. **39**(3), 367–377 (2001)
223. A. Beskos, N. Pillai, G. Roberts, J.-M. Sanz-Serna, A. Stuart, Optimal tuning of the hybrid Monte Carlo algorithm. Bernoulli. **19**(5A), 1501–1534 (2013)
224. M. Betancourt, S. Byrne, M. Girolami, Optimizing the integrator step size for Hamiltonian Monte Carlo (2014). arXiv:1411.6669
225. M. Betancourt, S. Byrne, S. Livingstone, M. Girolami, The geometric foundations of Hamiltonian Monte Carlo (2014). arXiv:1410.5110
226. M. Betancourt, A conceptual introduction to Hamiltonian Monte Carlo (2017). arXiv:1701.02434
227. A. Nishimura, D. Dunson, Geometrically Tempered Hamiltonian Monte Carlo, 1–44 (2016). arXiv:1604.00872
228. S. Lan, J. Streets, B. Shahbaba, in *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*. Wormhole Hamiltonian Monte Carlo, (2014), pp. 1–30
229. H. Niederreiter, *Random Number Generation and Quasi-Monte Carlo Methods*. (Society for Industrial Mathematics, Philadelphia, PA (USA), 1992)

230. R. Craiu, J. Rosenthal, C. Yang, Learn from thy neighbor: parallel-chains and regional adaptive MCMC. J. Am. Stat. Assoc. **104**(448), 1454–1466 (2009)
231. R. Casarin, R. V. Craiu, F. Leisen, Interacting multiple try algorithms with different proposal distributions. Stat. Comput. **23**, 185–200 (2013)
232. L. Martino, V. Elvira, D. Luengo, J. Corander, F. Louzada, Orthogonal parallel MCMC methods for sampling and optimization. Digit. Signal Process. **58**, 64–84 (2016)
233. G. Altekar, S. Dwarkadas, J. P. Huelsenbeck, F. Ronquist, Parallel metropolis coupled Markov Chain Monte Carlo for bayesian phylogenetic inference. Bioinformatics. **20**(3), 407–415 (2004)
234. C. J. Geyer, Markov Chain Monte Carlo maximum likelihood. Comput. Sci. Stat. Proceedings of the 23rd Symposium on the Interface, 156–163 (1991)
235. L. Martino, V. Elvira, D. Luengo, A. Artes, J. Corander, in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. Smelly parallel MCMC chains, (2015). https://doi.org/10.1109/icassp.2015.7178736
236. L. Martino, V. Elvira, D. Luengo, F. Louzada, in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. Parallel Metropolis chains with cooperative adaptation, (2016). https://doi.org/10.1109/icassp.2016.7472423
237. C. J. Geyer, E. A. Thompson, Annealing Markov Chain Monte Carlo with applications to ancestral inference. J. Am. Stat. Assoc. **90**, 909–920 (1995)
238. A. Jasra, D. A. Stephens, C. C. Holmes, On population-based simulation for static inference. Stat. Comput. **17**(3), 263–279 (2007)
239. J. Corander, M. Gyllenberg, T. Koski, Bayesian model learning based on a parallel MCMC strategy. Stat. Comput. **16**, 355–362 (2006)
240. J. Corander, M. Ekdahl, T. Koski, Parallel interacting MCMC for learning of topologies of graphical models. Data Min. Knowl. Discov. **17**(3), 431–456 (2008)
241. B. Calderhead, A general construction for parallelizing Metropolis-Hastings algorithms. Proc. Natl. Acad. Sci. U.S.A. (PNAS). **111**(49), 17408–17413 (2014)
242. R. Bardenet, A. Doucet, C. Holmes, On Markov chain Monte Carlo methods for tall data (2015). arXiv:1505.02827
243. W. Neiswanger, C. Wang, E. Xing, Asymptotically exact, embarrassingly parallel MCMC (2013). arXiv:1311.4780
244. X. Wang, D. B. Dunson, Parallelizing MCMC via Weierstrass sampler (2014). arXiv:1312.4605v2
245. X. Wang, F. Guo, K. A. Heller, D. B. Dunson, Parallelizing MCMC with random partition trees (2015). arXiv:1311.4780
246. A. E. Brockwell, Parallel Markov chain Monte Carlo simulation by pre-fetching. J. Comput. Graph. Stat. **15**(1), 246–261 (2006)
247. J. P. Huelsenbeck, F. Ronquist, et al., Mrbayes: Bayesian inference of phylogenetic trees. Bioinformatics. **17**(8), 754–755 (2001)
248. J. S. Rosenthal, Parallel computing and Monte Carlo algorithms. Far East J. Theor. Stat. **4**(2), 207–236 (2000)
249. I. Strid, Efficient parallelisation of Metropolis-Hastings algorithms using a prefetching approach. Comput, Stat. Data Anal. **54**(11), 2814–2835 (2010)
250. A. Terenin, S. Dong, D. Draper, GPU-accelerated Gibbs sampling: a case study of the Horseshoe Probit model. Stat. Comput. **29**(2), 301–310 (2019)
251. A. Terenin, D. Simpson, D. Draper, Asynchronous Gibbs sampling (2018). arXiv:1509.08999v5
252. L. Tierney, A. Mira, Some adaptive Monte Carlo methods for Bayesian inference. Stat. Med. **18**, 2507–2515 (1999)
253. J. Bierkens, Non-reversible Metropolis-Hastings. Stat. Comput. **26**(6), 1213–1228 (2016)
254. R. M. Neal, *Improving Asymptotic Variance of MCMC Estimators: Non-reversible Chains Are Better. Technical report, No. 0406*. (Department of Statistics, University of Toronto, 2004), pp. 1–25
255. H. C. M. Fernandes, M. Weigel, Non-reversible Monte Carlo simulations of spin models. Comput. Phys. Commun. **182**(9), 1856–1859 (2011)
256. K. S. Turitsyn, M. Chertkov, M. Vucelja, Irreversible Monte Carlo algorithms for efficient sampling. Phys. D Nonlinear Phenom. **240**(4), 410–414 (2011)
257. R. D. Schram, G. T. Barkema, Monte carlo methods beyond detailed balance. Phys. A Stat. Mech. Appl. **418**, 88–93 (2015)
258. M. Vucelja, Lifting — a non-reversible Markov Chain Carlo algorithm, 1–14 (2014). arXiv:1412.8762
259. K. S. Turitsyn, M. Chertkov, M. Vucelja, Irreversible Monte Carlo algorithms for efficient sampling. Phys. D Nonlinear Phenom. **240**(4-5), 410–414 (2011)
260. R. D. Schram, G. T. Barkema, Monte Carlo methods beyond detailed balance. Phys. A Stat. Mech. Appl. **418**, 88–93 (2015)
261. A. Gelman, J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, D. B. Rubin, *Bayesian Data Analysis, 3rd edn.* (Chapman and Hall/CRC, Boca Raton, FL (USA), 2013)
262. S. P. Brooks, A. Gelman, General methods for monitoring convergence of iterative simulations. J. Comput. Graph. Stat. **7**(4), 434–455 (1998)
263. S. Brooks, P. Giudici, A. Philippe, Nonparametric convergence assessment for MCMC model selection. J. Comput. Graph. Stat. **12**(1), 1–22 (2003)
264. A. Gelman, D. B. Rubin, Inference from iterative simulation using multiple sequences. Stat. Sci. **7**(4), 457–472 (1992)
265. J. Gorham, L. Mackey, in *Advances in Neural Information Processing Systems (NIPS)*. Measuring sample quality with Stein's method, (2015), pp. 226–234
266. J. Gorham, A. B. Duncan, S. J. Vollmer, L. Mackey, Measuring sample quality with diffusions (2018). arXiv:1611.06972v6
267. K. Chwialkowski, H. Strathmann, A. Gretton, in *International Conference on Machine Learning (ICML)*. A kernel test of goodness of fit, (2016). JMLR: Workshop and Conference Proceedings
268. Q. Liu, J. Lee, M. Jordan, in *International Conference on Machine Learning (ICML)*. A kernelized Stein discrepancy for goodness-of-fit tests, (2016), pp. 276–284
269. C. J. Oates, M. Girolami, N. Chopin, Control functionals for Monte Carlo integration. J. R. Stat. Soc. Ser. B (Stat. Methodol.) **79**(3), 695–718 (2017)
270. J. Gorham, L. Mackey, in *Proceedings of the 34th International Conference on Machine Learning (ICML)*. Measuring sample quality with kernels, (2017), pp. 1292–1301. JMLR. org
271. J. E. Johndrow, A. Smith, N. Pillai, D. B. Dunson, MCMC for imbalanced categorical data. J. Am. Stat. Assoc., 1–10 (2018). https://doi.org/10.1080/01621459.2018.1505626
272. E. Veach, L. Guibas, in *SIGGRAPH 1995 Proceedings*. Optimally combining sampling techniques for Monte Carlo rendering, (1995), pp. 419–428
273. L. Martino, V. Elvira, D. Luengo, J. Corander, Layered Adaptive Importance Sampling. Stat. Comput. **27**(3), 599–623 (2017)
274. V. Elvira, L. Martino, D. Luengo, M. F. Bugallo, Improving Population Monte Carlo: Alternative weighting and resampling schemes. Signal Process. **131**(12), 77–91 (2017)
275. V. Elvira, L. Martino, D. Luengo, M. F. Bugallo, Generalized multiple importance sampling (2015). arXiv:1511.03095
276. V. Elvira, L. Martino, D. Luengo, M. F. Bugallo, Efficient multiple importance sampling estimators. IEEE Signal Process. Lett. **22**(10), 1757–1761 (2015)
277. V. Elvira, L. Martino, D. Luengo, M. F. Bugallo, Heretical multiple importance sampling. IEEE Signal Process. Lett. **23**(10), 1474–1478 (2016)
278. V. Elvira, L. Martino, D. Luengo, J. Corander, in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. A gradient adaptive population importance sampler, (2015), pp. 4075–4079. https://doi.org/10.1109/icassp.2015.7178737
279. E. Koblents, J. Miguez, A population Monte Carlo scheme with transformed weights and its application to stochastic kinetic models. Stat. Comput. (2014). https://doi.org/10.1007/s11222-013-9440-2
280. L. Martino, V. Elvira, D. Luengo, J. Corander, *An adaptive population importance sampler*, (2014), pp. 8088–8092. https://doi.org/10.1109/icassp.2014.6855166
281. I. Schuster, Gradient importance sampling, 313–316 (2015). arXiv:1507.05781
282. L. Martino, V. Elvira, D. Luengo, J. Corander, in *Interdisciplinary Bayesian Statistics, vol. 118*, ed. by A. Polpo, F. Louzada, L. L. R. Rifo, J. R. Stern, and M. Lauretto. MCMC-driven adaptive multiple importance sampling (Chap. 8, New York, NY (USA), 2015), pp. 97–109
283. A. L. Nguyen, Regenerative Markov chain importance sampling. Commun. Stat. Simul. Comput. (2015). https://doi.org/10.1080/03610918.2015.1043383
284. Z. I. Botev, P. L'Ecuyer, B. Tuffin, Markov chain importance sampling with applications to rare event probability estimation. Stat. Comput. **23**, 271–285 (2013)
285. X. Yuan, Z. Lu, C. Z. Yue, A novel adaptive importance sampling algorithm based on Markov chain and low-discrepancy sequence. Aerosp. Sci. Technol. **29**, 253–261 (2013)

286. P. del Moral, A. Doucet, A. Jasra, Sequential Monte Carlo samplers. J. R. Stat. Soc. Ser. B (Stat. Methodol.) **68**(3), 411–436 (2006)
287. M.-S. Oh, J. O. Berger, Adaptive importance sampling in Monte Carlo integration. J. Stat. Comput. Simul. **41**(3-4), 143–168 (1992)
288. J. Geweke, Bayesian inference in econometric models using Monte Carlo integration. Econometrica J. Econometric Soc., 1317–1339 (1989). https://doi.org/10.2307/1913710
289. R. Chen, Another look at rejection sampling through importance sampling. Stat. Probab. Lett. **72**, 277–283 (2005)
290. A. Kong, *A note on importance sampling using standardized weights, vol. 348*. (Dept. of Statistics, Tech. Rep, University of Chicago, 1992)
291. A. Kong, J. S. Liu, W. H. Wong, Sequential imputations and Bayesian missing data problems. J. Am. Stat. Assoc. **89**(425), 278–288 (1994)
292. (A. Doucet, N. de Freitas, N. Gordon, eds.), *Sequential Monte Carlo Methods in Practice*. (Springer, New York, 2001)
293. C. P. Robert, G. Casella, *Introducing Monte Carlo Methods With R*. (Springer, New York, NY (USA), 2010)
294. C. Robert, Effective Sample Size. Christian P. Robert's blog (2010). https://xianblog.wordpress.com/2010/09/24/effective-sample-size/. Accessed 22 Apr 2020
295. L. Martino, V. Elvira, F. Louzada, Effective sample size for importance sampling based on discrepancy measures. Signal Process. **131**, 386–401 (2017)
296. S. Agapiou, O. Papaspiliopoulos, D. Sanz-Alonso, A. M. Stuart, Importance sampling: computational complexity and intrinsic dimension (2015). arXiv:1511.06196
297. N. Whiteley, A. Lee, K. Heine, On the role of interaction in sequential Monte Carlo algorithms. Bernoulli. **22**(1), 494–529 (2016)
298. J. H. Huggins, D. M. Roy, Convergence of sequential Monte Carlo based sampling methods (2015). arXiv:1503.00966
299. R. Lamberti, Y. Petetin, F. Septier, F. Desbouvries, in *IEEE Statistical Signal Processing Workshop (SSP)*. An improved sir-based sequential monte carlo algorithm, (2016), pp. 1–5. https://doi.org/10.1109/ssp.2016.7551745
300. R. Lamberti, Y. Petetin, F. Desbouvries, F. Septier, Independent resampling sequential monte carlo algorithms. IEEE Trans. Signal Process. **65**(20), 5318–5333 (2017)
301. L. Martino, V. Elvira, F. Louzada, Weighting a resampled particle in Sequential Monte Carlo. IEEE Stat. Signal Process. Work. (SSP). **122**, 1–5 (2016)
302. C. A. Naesseth, F. Lindsten, T. B. Schon, Nested sequential Monte Carlo methods. Proc. Int. Conf. Mach. Learn. **37**, 1–10 (2015)
303. L. Martino, V. Elvira, G. Camps-Valls, Group importance sampling for particle filtering and MCMC. Digit. Signal Process. **82**, 133–151 (2018)
304. L. Martino, J. Read, V. Elvira, F. Louzada, Cooperative parallel particle filters for on-line model selection and applications to urban mobility. Digit. Signal Process. **60**, 172–185 (2017)
305. M. Bolić, P. M. Djurić, S. Hong, Resampling algorithms and architectures for distributed particle filters. IEEE Trans. Signal Process. **53**(7), 2442–2450 (2005)
306. J. Míguez, M. A. Vázquez, A proof of uniform convergence over time for a distributed particle filter. Signal Process. **122**, 152–163 (2016)
307. J. Read, K. Achutegui, J. Míguez, A distributed particle filter for nonlinear tracking in wireless sensor networks. Signal Process. **98**, 121–134 (2014)
308. C. Vergé, C. Dubarry, P. D. Moral, E. Moulines, On parallel implementation of sequential Monte Carlo methods: the island particle model. Stat. Comput. **25**(2), 243–260 (2015)
309. C. Vergé, P. D. Moral, E. Moulines, J. Olsson, Convergence properties of weighted particle islands with application to the double bootstrap algorithm, 1–39 (2014). arXiv:1410.4231
310. C. C. Drovandi, J. McGree, A. N. Pettitt, A sequential Monte Carlo algorithm to incorporate model uncertainty in Bayesian sequential design. J. Comput. Graph. Stat. **23**(1), 3–24 (2014)
311. I. Urteaga, M. F. Bugallo, P. M. Djurić, in *2016 IEEE Statistical Signal Processing Workshop (SSP)*. Sequential Monte Carlo methods under model uncertainty, (2016), pp. 1–5
312. C. A. Naesseth, F. Lindsten, T. B. Schon, High-dimensional filtering using nested sequential Monte Carlo, 1–48 (2016). arXiv:1612.09162
313. R. B. Stern, A statistical contribution to historical linguistics. Phd Thesis (2015)
314. P. M. Djurić, J. H. Kotecha, J. Zhang, Y. Huang, T. Ghirmai, M. F. Bugallo, J. Míguez, Particle filtering. IEEE Signal Process. Mag. **20**(5), 19–38 (2003)
315. A. Doucet, A. M. Johansen, A tutorial on particle filtering and smoothing: fifteen years later. Technical report (2008)
316. K. Achutegui, L. Martino, J. Rodas, C. J. Escudero, J. Miguez, in *IEEE Intelligent Control and Control Applications (CCA)*. A multi-model particle filtering algorithm for indoor tracking of mobile terminals using RSS data, (2009), pp. 1702–1707. https://doi.org/10.1109/cca.2009.5280960
317. C. Andrieu, A. Doucet, R. Holenstein, Particle Markov chain Monte Carlo methods. J. R. Stat. Soc. B. **72**(3), 269–342 (2010)
318. M. I. Lindsten, M. Jordan, T. B. Schon, Particle Gibbs with ancestor sampling. J. Mach. Learn. Res. **15**(1), 2145–2184 (2014)
319. J. Kokkala, *Particle and Sigma-point Methods for State and Parameter Estimation in Nonlinear Dynamic Systems. Phd Thesis*. (Department of Computer Science, Aalto University; Supervisor S, Sarkka, 2016), pp. 1–72
320. J. S. Liu, F. Liang, W. H. Wong, The multiple-try method and local optimization in Metropolis sampling. J. Am. Stat. Assoc. **95**(449), 121–134 (2000)
321. L. Martino, J. Read, On the flexibility of the design of multiple try Metropolis schemes. Comput. Stat. **28**(6), 2797–2823 (2013)
322. L. Martino, V. P. D. Olmo, J. Read, A multi-point Metropolis scheme with generic weight functions. Stat. Probab. Lett. **82**(7), 1445–1453 (2012)
323. M. Bédard, R. Douc, E. Mouline, Scaling analysis of multiple-try MCMC methods. Stoch. Process. Appl. **122**, 758–786 (2012)
324. R. V. Craiu, C. Lemieux, Acceleration of the multiple try Metropolis algorithm using antithetic and stratified sampling. Stat. Comput. **17**(2), 109–120 (2007)
325. L. Martino, F. Louzada, Issues in the mltiple try Metropolis mixing. Comput. Stat. **32**(1), 239–252 (2017)
326. L. Martino, F. Leisen, J. Corander, On multiple try schemes and the particle Metropolis-Hastings algorithm (2014). viXra:1409.0051
327. G. Casella, C. P. Robert, Rao-Blackwellisation of sampling schemes. Biometrika. **83**(1), 81–94 (1996)
328. R. Neal, MCMC using ensembles of states for problems with fast and slow variables such as Gaussian process regression (2011). arXiv:1101.0387
329. J. Kokkala, S. Sarkka, Combining particle MCMC with Rao-Blackwellized Monte Carlo data association for parameter estimation in multiple target tracking. Digit. Signal Process. **47**, 84–95 (2015)
330. C. Andrieu, G. O. Roberts, The pseudo-marginal approach for efficient monte carlo computations. Ann. Stat. **37**(2), 697–725 (2009)
331. I. Murray, Z. Ghahramani, D. MacKay, *MCMC for doubly-intractable distributions*, (2006), pp. 1–8
332. J. Moller, A. Pettit, R. Reeves, K. Bertheksen, An efficient Markov Chain Monte Carlo method for distributions with intractable normalising constants. Biometrika. **93**, 451–458 (2006)
333. P. Marjoram, J. Molitor, V. Plagnol, S. Tavare, Markov Chain Monte Carlo without likelihoods. Proc. Natl. Acad. Sci. U.S.A. **100**, 15324–15328 (2003)
334. M. A. Beaumont, Estimation of population growth or decline in genetically monitored populations. Genetics. **164**, 1139–1160 (2003)
335. J. M. Marin, P. Pudlo, C. P. Robert, R. J. Ryder, Approximate Bayesian computational methods. Stat. Comput. **22**(6), 1167–1180 (2006)
336. R. Bardenet, A. Doucet, C. Holmes, in *Proceedings of the 31st International Conference on Machine Learning*. Towards scaling up Markov Chain Monte Carlo: an adaptive subsampling approach, (2014), pp. 1–9
337. A. Korattikara, Y. Chen, M. Welling, in *Proceedings of the 31st International Conference on Machine Learning*. Austerity in MCMC land: Cutting the Metropolis-Hastings budget, (2014), pp. 681–688
338. J. A. Christen, C. Fox, Markov Chain Monte Carlo using an approximation. J. Comput. Graph. Stat. **14**, 795–810 (2005)
339. C. Sherlock, G. A, D. A. Henderson, Adaptive, delayed-acceptance MCMC for targets with expensive likelihoods. J. Comput. Graph. Stat. **26**(2), 434–444 (2017)
340. A. Solonen, P. Ollinaho, M. Laine, H. Haario, J. Tamminen, H. Jarvinen, Efficient MCMC for climate model parameter estimation: parallel adaptive chains and Early Rejection. Bayesian Anal. **7**(3), 715–736 (2012)
341. D. Luengo, L. Martino, V. Elvira, M. Bugallo, in *IEEE 6th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*. Bias correction for distributed Bayesian estimators (IEEE, 2015), pp. 253–256. https://doi.org/10.1109/camsap.2015.7383784
342. D. Maclaurin, R. P. Adams, Firefly Monte Carlo: exact MCMC with subsets of data (2014). arXiv:1403.5693
343. D. Luengo, L. Martino, V. Elvira, M. F. Bugallo, Efficient linear fusion of partial estimators. Digit. Signal Process. **78**, 265–283 (2018)

344. S. L. Scott, A. W. Blocker, F. V. Bonassi, H. A. Chipman, E. I. George, R. E. McCulloch, in *EFaBBayes 250th Conference, vol. 16*. Bayes and big data: the consensus Monte Carlo algorithm, (2013)
345. G. K. Nicholls, C. Fox, A. M. Watt, Coupled MCMC with a randomized acceptance probability, 1–20 (2012). arXiv:1205.6857
346. P. Alquier, N. Friel, R. Everitt, A. Boland, Noisy Monte Carlo: convergence of Markov chains with approximate transition kernels. Stat. Comput. **26**(2), 29–47 (2016)
347. F. J. Medina-Aguayo, A. Lee, G. O. Roberts, Stability of noisy Metropolis-Hastings. Stat. Comput. **26**(6), 1187–1211 (2016)
348. S. Särkkä, *Bayesian Filtering and Smoothing*. (Cambridge University Press, New York, 2013)
349. I. S. Mbalawata, S. Särkkä, H. Haario, Parameter estimation in stochastic differential equations with markov chain monte carlo and non-linear kalman filtering. Comput. Stat. **28**(3), 1195–1223 (2013)
350. S. Särkkä, J. Hartikainen, I. S. Mbalawata, H. Haario, Posterior inference on parameters of stochastic differential equations via non-linear Gaussian filtering and adaptive MCMC. Stat. Comput. **25**(2), 427–437 (2015)
351. J. E. Johndrow, J. C. Mattingly, S. Mukherjee, D. Dunson, Optimal approximating Markov chains for Bayesian inference (2015). arXiv:1508.03387
352. J. Negrea, J. S. Rosenthal, Error bounds for approximations of geometrically ergodic Markov chains (2017). arXiv:1702.07441
353. M. Betancourt, Adiabatic Monte Carlo (2014). arXiv:1405.3489
354. C. T. Perretti, S. B. Munch, G. Sugihara, Model-free forecasting outperforms the correct mechanistic model for simulated and experimental data. Proc. Natl. Acad. Sci. (PNAS). **110**(13), 5253–5257 (2013)
355. C. T. Perretti, S. B. Munch, G. Sugihara, Reply to Hartig and Dormann: the true model myth. Proc. Natl. Acad. Sci. (PNAS). **110**(42), 3976–3977 (2013)
356. A. Boyarsky, P. Góra, *Law of Chaos*. (Birkhöuser, Boston, MA (USA), 1997)
357. A. T. Ihler, J. W. Fisher, R. L. Moses, A. S. Willsky, Nonparametric belief propagation for self-localization of sensor networks. IEEE J. Sel. Areas Commun. **23**(4), 809–819 (2005)
358. A. M. Ali, S. Asgari, T. C. Collier, M. Allen, L. Girod, R. E. Hudson, K. Yao, C. E. Taylor, D. T. Blumstein, An empirical study of collaborative acoustic source localization. J. Signal Process. Syst. **57**(3), 415–436 (2009)
359. P. Stoica, List of references on spectral line analysis. Signal Process. **31**(3), 329–340 (1993)
360. H. So, K. W. Chan, Y. T. Chan, K. C. Ho, Linear prediction approach for efficient frequency estimation of multiple real sinusoids: algorithms and analyses. Signal Process. IEEE Trans. **53**(7), 2290–2305 (2005)
361. M. Bodson, S. C. Douglas, Adaptive algorithms for the rejection of sinusoidal disturbances with unknown frequency. Automatica. **33**(12), 2213–2221 (1997)
362. A. Bobtsov, A. A. Pyrkin, Cancelation of unknown multiharmonic disturbance for nonlinear plant with input delay. Int. J. Adapt. Control Signal Process. **26**(4), 302–315 (2012)
363. C. Carlemalm, H. V. Poor, A. Logothetis, Suppression of multiple narrowband interferers in a spread-spectrum communication system. IEEE J. Sel. Areas Commun. **18**(8), 1365–1374 (2000)
364. K. D. Rao, M. N. S. Swamy, E. Plotkin, A nonlinear adaptive filter for narrowband interference mitigation in spread spectrum systems. Signal Process. **85**(3), 625–635 (2005)
365. J. G. Proakis, *Digital Communications*. (McGraw-Hill, Singapore, 1995)
366. M. A. Newton, A. E. Raftery, Approximate Bayesian inference with the weighted likelihood bootstrap. J. R. Stat. Soc. Ser.B (Methodol.), 3–48 (1994)

Publisher's Note