Simulation Based Nearest Neighbor Entropy Estimation for (Adaptive) MCMC Evaluation

Didier Chauveau* Pierre Vandekerkhove†‡

Abstract

Many recent (including adaptive) MCMC methods are associated in practice to unknown rates of convergence. We propose a simulation-based methodology to estimate MCMC efficiency, grounded on a Kullback divergence criterion requiring an estimate of the entropy of the algorithm successive densities, computed from iid simulated chains. We recently proved in Chauveau and Vandekerkhove (2013) some consistency results in MCMC setup for an entropy estimate based on Monte-Carlo integration of a kernel density estimate based on Györfi and Van Der Meulen (1989). Since this estimate requires some tuning parameters and deteriorates as dimension increases, we investigate here an alternative estimation technique based on Nearest Neighbor (NN) estimates. This approach has been initiated by Kozachenko and Leonenko (1987) but used mostly in univariate situations until recently when entropy estimation has been considered in other fields like neuroscience. We show that in MCMC setup where moderate to large dimensions are common, this estimate seems appealing for both computational and operational considerations, and that the problem inherent to a non neglictible bias arising in high dimension can be overcome. All our algorithms for MCMC simulation and entropy estimation are implemented in an R package taking advantage of recent advances in high performance (parallel) computing.

Key Words: Adaptive MCMC algorithms, Bayesian model, entropy, Kullback divergence, Metropolis-Hastings algorithm, nearest neighbor estimation, nonparametric statistic.

1. Introduction

A Markov Chain Monte Carlo (MCMC) method generates an ergodic Markov chain for which the stationary distribution is a given probability density function (pdf) f. For common Bayesian inference, f is a posterior distribution of the model parameter θ over a state space $\Theta \subseteq \mathbb{R}^d$. This posterior is typically known only up to a multiplicative normalizing constant, and simulation or integration w.r.t. f are approximated by ergodic averages from the chain. The Metropolis-Hastings (MH) algorithm (Hastings, 1970; Metropolis et al., 1953) is one of the most popular algorithm used in MCMC methods. Another commonly used method is the Gibbs sampler introduced by Geman and Geman (1984).

Each step of a MH algorithm at a current position θ^t is based on the generation of the proposed next move from a general *proposal density* $q(\cdot|\theta^t)$. Historically, two popular MH strategies used to be (i) the *Independence Sampler* (MHIS), which uses a proposal distribution independent of the current position, and (ii) the Random Walk MH algorithm (RWMH), for which the proposal is a random perturbation of the current position, most often drawn from a Gaussian distribution with a fixed variance matrix that has to be tuned.

To actually implement a MCMC algorithm, many choices for the proposal density are possible, with the goal of improving mixing and convergence properties of the resulting Markov chain. For instance running a RWMH strategy requires the determination of a

*MAPMO - CNRS UMR 6628 - Fédération Denis Poisson, Université d'Orléans, BP 6759, 45067 Orléans cedex 2, France.

†LAMA - CNRS UMR 8050, Université de Marne-la-Vallée, 5, boulevard Descartes, Cité Descartes - Champs-sur-Marne, 77454 Marne-la-Vallée, France.

‡George W. Woodruff School of Mechanical Engineering, Georgia Institute of Technology, Atlanta, GA 30332-0405, USA.

"good" scaling constant, since the mixing depends dramatically on the variance matrix of the perturbation (Roberts and Rosenthal, 2001). As a consequence, a growing interest in new methods appeared this last decade, which purpose is to optimize in sequence the proposal strategy in MCMC algorithms on the basis of the chain(s) history; see, e.g., Andrieu and Thoms (2008) for a recent survey. These approaches called adaptive Monte Carlo Markov Chains (AMCMC) can be described (not in an entirely general way) as follows: let f be the pdf of interest and suppose that we aim to simulate efficiently from f given a family of Markov kernels $\{P_\vartheta, \vartheta \in \mathcal{E}\}$. This can be done adaptively using a joint process $(\theta^t, \vartheta^t)_{t\geq 0}$ such that the conditional distribution of θ^{t+1} given the information available up to time t is a kernel P_{ϑ^t} where ϑ^t is an Euclidean parameter tuned over time to fit a supposed relevant strategy. Some general sufficient conditions insuring convergence (essentially ergodicity and the strong law of large numbers) of such algorithms have been established by various authors, see Andrieu and Thoms (2008). These conditions are informally based on the two following ideas.

Containment: for any (θ^0, ϑ^0), and any $\varepsilon > 0$, the stochastic process $(M_\varepsilon(\theta^t, \vartheta^t))_{t\geq 0}$ is bounded in probability, where

$$M_\varepsilon(\theta, \vartheta) = \inf\left\{t \geq 1 \ : \ \|P^t_\vartheta(\theta, \cdot) - f(\cdot)\|_{TV} \leq \varepsilon\right\}$$

is the "ε-time to convergence".

Diminishing Adaptation: for any (θ^0, ϑ^0), $\lim_{t\to\infty} D_t = 0$ in $\mathbb{P}_{\theta^0,\vartheta^0}$-probability, where

$$D_t = \sup_{\theta\in\Theta} \|P_{\vartheta^{t+1}}(\theta, \cdot) - P_{\vartheta^t}(\theta, \cdot)\|_{TV},$$

represents the amount of adaptation performed between iterations t and $t + 1$.

Note that in Bai et al. (2008) two examples are provided to show that either Diminishing Adaptation or Containment is not necessary for ergodicity of AMCMC, and diminishing Adaptation alone cannot guarantee ergodicity. See also the very simple four-state Markov Chain Example 1 in Rosenthal and Roberts (2007), which illustrates the fact that ergodicity is not an automatic heritage when adapting a Markov Chain from its past.

These various and sometimes experimental algorithmic choices are associated in general to unknown rates of convergence because of the complexity of the kernel, and the difficulty in computing, when available, the theoretical bounds of convergence. For instance, Bai et al. (2010) compare two AMCMC strategies in dimension $d \leq 5$, and Vrugt et al. (2009) compare two AMCMC's against some benchmark in dimension $d = 10$. More recently Fort et al. (2013) define the best interacting ratio for a simple equi-energy type sampler, by minimizing the corresponding limiting variance involved in the Central Limit Theorem (see Fig. 1 in Fort et al. (2013)). There are also recent works proposing tools or methods for MCMC comparisons, showing that these questions are crucial in nowaday MCMC application and research. Thompson (2010) proposes the R package SamplerCompare for comparing several MCMC's differing by a single tuning parameter, using standard evaluation criterion.

In this paper, we propose a methodological approach, and corresponding software tool, only based on Monte Carlo simulation (i.e. not requiring a theoretical study typically MCMC and/or target-specific) with two goals: (i) For MCMC users to easily select a good sampler among possible candidates; (ii) For researchers to better understand which (A)MCMC methods perform best in which circumstances. Let

$$\mathcal{H}(p) := \int p \, \log p = \mathbb{E}_p(\log p) \tag{1}$$

be the differential entropy of a probability density p over Θ, and p^t be the marginal density of the (A)MCMC algorithm at "time" (iteration) t. Our approach is grounded on a criterion

which is the evolution of the Kullback-Leibler divergence between p^t and f,

$$t \mapsto \mathcal{K}(p^t, f) := \int p^t \log\left(\frac{p^t}{f}\right) = \mathcal{H}(p^t) - \int p^t \, \log f.$$

This Kullback "distance" is indeed a natural measure of the algorithm's quality and has strong connections with ergodicity of Markov chains and rates of convergence, see Harremoes and Holst (2007) for recent results. In MCMC setup, Chauveau and Vandekerkhove (2013) showed that if the proposal density of a Metropolis-Hastings algorithm satifies a uniform minoration condition implying its geometric convergence as in Holden (1998), then $\mathcal{K}(p^t, f)$ also decreases geometrically.

In Section 2, we detail our approach which is methodological but relies on estimation techniques that have been proved to be theoretically consistent in simple cases like Gaussian unidimensional RWMH or independent samplers (Chauveau and Vandekerkhove, 2013). Our estimation of $\mathcal{H}(p^t)$ is grounded on the simulation of N *parallel* (iid) chains. Section 3 illustrates the good behavior of our criterion on a synthetic multi-dimensional example. This example also allows us to show the difficulty coming from the curse of dimension in nonparametric statistical estimation. In Section 4 we detail our solution for handling that difficulty, in such a way that our approach being still usable even in large dimension, e.g. for Bayesian models with dozens of parameters.

2. Entropy and Kullback estimation in MCMC context

Recent motivations for entropy estimation in other fields like molecular science appeared recently in the literature (see, e.g. Singh et al., 2003), and are concerned by estimation of $\mathcal{H}(p)$ for multivariate densities p. Most of the estimation techniques proved to be consistent under various conditions are based on iid samples from p. There exists some results about entropy estimation for dependent sequences, but these heavily rely on the mixing properties of these sequences themselves, that are precisely what we want to capture by our simulation-based approach without theoretical investigations concerning mixing properties of the Markov kernel. More importantly, these approaches could be used to estimate $\mathcal{H}(f)$ but cannot estimate $\mathcal{H}(p^t)$ for each t.

2.1 Simulation of iid copies of the (A)MCMC algorithm

Our approach is consequently based on the simulation of N parallel (iid) copies of (A)Markov chains started from a diffuse initial distribution p^0 and using the transition kernel defined by the MCMC strategy under investigation. The N chains started from $\theta_1^0, \ldots, \theta_N^0$ iid$\sim p^0$ are denoted

$$\begin{array}{rcl} \text{chain \# } 1 & : & \theta_1^0 \to \theta_1^1 \to \cdots \to \theta_1^t \sim p^t \to \cdots \\ \vdots & & \vdots \\ \text{chain \# } N & : & \theta_N^0 \to \theta_N^1 \to \cdots \to \theta_N^t \sim p^t \to \cdots \end{array}$$

where "$\to$" indicates the (eventually non-homogeneous) Markov dependence. At "time" (iteration) t, the locations of the N simulated chains $\boldsymbol{\theta}^t = (\theta_1^t, \ldots, \theta_N^t)$ forms a N-sample iid$\sim p^t$.

In an experimental framework where when one wants to evaluate a new (A)MCMC algorithm the target f often corresponds to a benchmark example, hence is completely known

(as e.g., in Vrugt et al., 2009). In this case a strongly consistent estimate of $\int p^t \log f$ is given by Monte Carlo integration and the Strong Law of Large Numbers,

$$\hat{p}_N^t(\log f) = \frac{1}{N}\sum_{i=1}^{N} \log f(\theta_i^t), \tag{2}$$

so that estimation of $\mathcal{K}(p^t, f)$ is in turn accessible provided $\mathcal{H}(p^t)$ is. However, if the objective is to evaluate an experimental MCMC method for an actual Bayesian model for which f is a posterior density proportional to the likelihood, say $f(\cdot) \propto \phi(\cdot)$ where the normalization constant is not known, only $\hat{p}_N^t(\log \phi)$ is accessible. We will see that this is not really a flaw since ϕ itself retains all the specificity (shape, modes, tails, ...) of f, and since we are mostly interested in the stabilization in t of $\mathcal{K}(p^t, f)$, not necessarily in knowing its limiting value, as will be detailed in Section 4. In addition, the normalization problem can be eliminated by comparing the MCMC under study to a benchmark MCMC algorithm (e.g., a gaussian RWMH) for the same target f. Indeed, considering two MCMC strategies leading to two sequences of marginal densities, say $(p_1^t)_{t\geq 0}$ and $(p_2^t)_{t\geq 0}$ allows the *difference* of the divergences to be accessible to estimation since

$$D(p_1^t, p_2^t, f) = \mathcal{K}(p_1^t, f) - \mathcal{K}(p_2^t, f) = \mathcal{H}(p_1^t) - \mathcal{H}(p_2^t) + \mathbb{E}_{p_2^t}[\log \phi] - \mathbb{E}_{p_1^t}[\log \phi]. \tag{3}$$

The Kullback criterion is the only usual divergence insuring this property and, in addition to its connection with ergodicity, it motivates our choice. Note also that the Kullback divergence is currently used as a criterion in other simulation approaches, see Douc et al. (2007). The choice of this estimate also has the advantage of avoiding numerical integration in moderate or high dimensional spaces (replaced by Monte Carlo integration), in contrary to other criterion such as the L^1-distance.

For estimating the entropy $\mathcal{H}(p^t)$ a classical, plug-in approach, is to build a nonparametric kernel density estimate of p^t, and to compute the Monte Carlo integration of this estimate. Techniques based on this approach have been suggested by Ahmad and Lin (1989), and studied by many authors under different assumptions (see, e.g., the survey paper Beirlant et al., 1997). Several consistency and asymptotic normality results pertaining to this approach have been proved (see references in Eggermont and LaRiccia, 1999). However, most of these are not suitable to estimate $\mathcal{H}(p^t)$ even in the simplest MH cases, either because they do not apply to multivariate densities, or because they require smoothness conditions that are far too restrictive to be proved for the sequences of densities p^t we have to consider here. Up to our best knowledge, the unique consistency result applicable in this MCMC simulation context is the one proved in Györfi and Van Der Meulen (1989), since it essentially requires a smoothness condition which is in turn implied by a Lipschitz condition. Indeed, for that approach, Chauveau and Vandekerkhove (2013) have proved that adequate smoothness and tail conditions on the "input ingredients" of the MH algorithm (namely p^0, q and f) propagate a Lipschitz condition to the successive marginals p^t, $t = 1, \ldots, n$, so that the sequence of $(\mathcal{H}(p^t))_{t=1,\ldots,n}$ can be consistently estimated. These technical conditions have been proved to hold in simple univariate IS and RWMH cases, but are not meant to be verified in general, since it would require tedious (and often unfeasible) calculations.

2.2 Estimates based on nearest neighbor distances

The plug-in estimate presented above requires the tuning of several parameters: a certain threshold for truncating the data over the tails of p^t, the choice of the kernel and the difficult issue of the appropriate bandwidth matrix, particularly in high dimensions. All these issues

motivated us to find an alternative, and study the behavior of the somehow simpler Nearest Neighbor (NN) estimate from Kozachenko and Leonenko (1987). Based on the sample $\boldsymbol{\theta}^t$, their NN entropy estimate is defined by

$$\hat{\mathcal{H}}_N(p^t) = \frac{1}{N}\sum_{i=1}^{N} \log(\rho_i^d) + \log(N-1) + \log(C_1(d)) + C_E, \tag{4}$$

where $C_E = -\int_0^\infty e^{-t}\log t\, dt \approx 0.5772\ldots$ is the Euler constant, $C_1(d) = \frac{\pi^{d/2}}{\Gamma(d/2+1)}$ and where $\rho_i = \min\{d(\theta_i^t, \theta_j^t), j \in \{1,2,\ldots,N\}, j \neq i\}$ is the (Euclidean) distance from the ith point to its nearest neighbor in the sample $\boldsymbol{\theta}^t$.

Kozachenko and Leonenko (1987) proved the mean square consistency of (4) for any dimension d under mild peak and tail conditions (see also Beirlant et al., 1997). This NN estimate seems more appealing than kernel density estimates in our situation, both from an operational point of view (no tuning parameters like the threshold and bandwidth), and from a computational point of view (the nearest distance can be computed faster than a multivariate kernel density estimate in high dimension). Until recently, this nearest neighbor approach has been used and studied mostly in univariate or bivariate ($d = 2$) situations, like in image processing. One interest of this study is to investigate its behavior in higher dimensions and for MCMC sequences of marginals.

3. Experiments and simulations

All the estimation techniques and MCMC evaluation criterion presented in the previous Sections are based on intensive simulations and computations for which we provide a software tool implemented in the "EntropyMCMC" package for the R statistical software (R Core Team, 2013), taking advantage of recent advances in High Performance Computing, that will be publicly available in a near futur. This package includes some predefined target distributions and standard MCMC samplers, easy definition of additional ones, functions for running simulations, estimating entropy and Kullback divergences, results visualization and sampler comparison. For instance Figs 1 and 2 have been done using simply a default plot() command from this package. The parallel simulations can be done from inside the package, or imported from external files. We first tried our approach on a simple multidimensional, centered Gaussian target density f with unit spherical covariance matrix. These experiments allows us to numerically check the consistency of $\hat{\mathcal{H}}_N(f)$ (and $\hat{\mathcal{K}}_N(p^t, f)$) since the true entropy is known for the Gaussian distribution. These results are not presented here for brevity.

We illustrate our approach on a synthetic but more complex target density: a 3-components mixture of multivariate, d-dimensional Gaussian distributions,

$$f(\boldsymbol{x}) = \sum_{j=1}^{3} \lambda_j\, \mathcal{N}_d(\mu_j, \Sigma_j)(\boldsymbol{x}),$$

where $\mathcal{N}_d(\mu_j, \Sigma_j)(\cdot)$ informally denotes the multivariate Gaussian density. The three weights are set to $\lambda_j = 1/3$, the mean vectors and (spherical) covariance matrices are set to

$$\mu_1 = 0\mathbf{1}_d, \quad \mu_2 = 4\mathbf{1}_d, \quad \mu_3 = -4\mathbf{1}_d, \quad \Sigma_j = j\mathbb{I}_d, \quad j = 1,2,3,$$

where $\mathbf{1}_d$ is the d-column vector of ones and $\mathbb{I}_d$ is the $d \times d$ identity matrix. The advantage of such a synthetic model is that it is defined for any dimension, but the complexity of the target increases with d (the distance between modes increases and the modes get more and

more separated and spiked). Note that here the normalizing constant of f is known, so that theoretically the sequence of marginals p^t from a proper (converging) MCMC satisfies $\mathcal{K}(p^t, f) \to 0$ as $t \to \infty$. We compare several standard, well known MCMC algorithms for recovering this target:

- Two RWMH with Gaussian proposals $\mathcal{N}_d(\theta^t; \sigma^2 \mathbb{I}_d)$ resulting in slow or fast MCMC's depending on the magnitude σ^2 of their (spherical) variance matrix that we set to $\sigma^2 = 1$ (algorithm called RW1 in the sequel) and 4 (called RW4).
- Three Independence Samplers (IS) with Gaussian proposals $\mathcal{N}_d(0\mathbf{1}_d; \sigma^2 \mathbb{I}_d)$ and the choices $\sigma^2 = 2$ (called IS2 in the sequel), $\sigma^2 = 9$ (IS9) and 16 (IS16).

The idea driving the choices above for the tuning parameters of the candidate MCMC's is that we want to compare fast, slow and even non converging MCMC's. For the 2-dimension Gaussian mixture, it is easy to figure out how to obtain a converging RW, like RW1. For the IS's, the practical support of the proposal density, within which each proposed next move lies, is suppose to include the region of interest of the target. This is definitely not the case for $\sigma^2 = 2$ (obviously for $d \leq 2$ where we can plot the target and proposal density), hence IS2 is a non converging algorithm. Larger variances should lead to better algorithms, but it is not easy to tell which value of σ^2 is the best choice. Also, the IS is known to be geometrically fast if the proposal have heavier tails than the target, which is not the case here. Then we just check how these strategies behave in higher dimension, where the 3 components get more and more separated and spiked.

Fig.1: In small dimensions like $d = 2$ (top panel), our criterion delivers the right answer straightforwardly, since there is no noticeable bias in the estimates, even with $N = 500$ chains. All the convergent MCMC's stabilized well before $n = 1000$ iterations. IS2 is not converging and is almost stabilized to a non-zero value in these $n = 1000$ iterations. Similar runs for more iterations show clearly its non convergence, and increasing N up to say 1000 reduces the variance, resulting in easier-to-read diagostics. The bottom panel displays a similar experiment, but now in dimension $d = 10$, and for $N = 5000$ chains to illustrate the variance reduction resulting is smooth curves (so many chains are not needed to get a readable diagnostic). The two RW's converge, but note that RW4 is less performant than RW1 in this higher dimension. IS2 is stabilized away from zero, indicating non convergence. It is hard to tell in 1000 iterations wether IS9 and IS16 will converge at some point, but 1000 iterations are sufficient to tell that these are slow, comparing with the RW's fast stabilizations. Actually running the experiment up to $n = 10,000$ show that IS16 is slow but converging, whereas IS9 convergence is not clear. This is because the proposal "almost covers" the region of interest of the mixture.

Fig.2: As expected, the scale of the number of iterations required to detect stabilization increases with the difficulty associated to the dimension, hence in this figure two simulations in the $d = 20$ dimensional case have been ran up to $n = 10,000$ iterations. The difference between top and bottom panels is just the number of iid chains, $N = 500$ (top) and $N = 10,000$ (bottom). This last number has been chosen intentionnally huge to illustrate the variance reduction, bias difficulty, and the fact that running so many chains is feasible but not needed to obtain a meaningful criterion. Note that this example with $d = 20$, $n = 10,000$ and $N = 500$ requires about about 25mn of CPU time per algorithm on a 12-cores single workstation.

One purpose of Fig.2 is precisely to illustrate the bias problem. Indeed, if we look at the top panel and the three IS's only, we conclude that IS16 and IS9 are non or very

slowly converging, and that IS2 quickly stabilizes near (and above) 0. Hence if we were only comparing algorithms for stabilization near 0, we would falsely conclude that IS2 is the preferable algorithm. But we already know that IS2 is not a convergent MCMC in this synthetic example. This bizarre results is due to a bias in $\mathcal{H}(p^t)$ estimation. Looking now at the curves for the RW's, we see that these stabilize on a common negative value, which is theoretically impossible since $\mathcal{K}(p^t, f) \geq 0$. Hence there is a negative bias, more prominent in the top panel with just $N = 500$ chains. Looking at the bottom panel, we see that estimating the entropy from samples of size $N = 10,000$ has the effect of reducing the variance giving more accurate curves, but only slighlty reducing the bias: all the curves are just shifted from a small amount (enough to tell that IS2 is actually not convergent). The difference between these two plots illustrate the fact that the bias reduces dramatically slowly with N, and that since all the stabilization values are biased, we cannot rely on stabilization near 0 to assess convergence or lack of convergence.

There are two reasons allowing us to evaluate the competing algorithms in view of Fig.2 top: (i) the property of the Kullback divergence says that $\mathcal{K}(p^t, f)$ decreases when $p^t \to f$, so that if the bias is of the same order in all the algorithms, the faster decay and smaller stabilization value is associated to the best algorithm; (ii) we have a prior knowledge that the RW's are convergent MCMC's, without knowing their rates of convergence. The two RW's can be viewed here as convergent *benchmarks*, and since they stabilize at the same value, it means that their biases are quite identical. To summarize, we can conclude that RW1 is preferable, RW4 is convergent but slower (probably as a consequence of the more spiked component modes so that its proposal variance is comparatively getting too large), IS2 is non convergent since it stabilizes above the benchmark RW1, and IS9 and IS16 may be convergent but even much slower. This diagnostic comes from the analysis of the top panel only, i.e. $N = 500$ is enough to get a conclusion. The next section details all these methodological questions.

4. How to handle the bias in large dimension

The experiments from Section 3 show that some care shoud be taken in the analysis of the plots of the estimates of $t \mapsto \mathcal{K}(p^t, f)$ delivered by our techniques, particularly in high dimension (say $d > 10$) where the bias becomes visible. This effect of the dimension on the bias has actually been already noticed in recent literature since nowadays applications of entropy estimation in other fields require moderate to high dimensions. Our results are in accordance with, for instance, Stowell and Plumbley (2009) and Sricharan et al. (2013). These studies show that in $\mathcal{H}(p)$ estimation, one can expect the variance to decreases as $\mathcal{O}(N^{-1})$ whereas the bias only decreases as $\mathcal{O}(N^{-1/(d+1)})$, which these authors called a "glacially slow" rate, and this phenomenon occurs both for Kernel density and NN-based estimates. These behaviors has been confirmed in our case using an iid sampler for a known (Gaussian) $\mathcal{H}(f)$. Hence, trying to achieve in practice the asymptotic unbiasedness guaranteed by the theory by "simply" increasing N is hopeless.

In addition to the bias difficulty, there is also the unknown normalizing constant. In our experiment the target f in the Monte-Carlo estimate (2) was entirely known, but in practical situations like Bayesian inference for a parameter θ, f is a posterior distribution only known up to a multiplicative constant, $f(\theta) = C\phi(\theta)$ where $C = (\int \phi(\theta)\, d\theta)^{-1}$ is the (unknown) normalizing constant. Hence what can be actually estimated is

$$\mathcal{K}(p^t, \phi) = \mathcal{H}(p^t) - \mathbb{E}_{p^t}(\log \phi) = \mathcal{K}(p^t, f) + \log C,$$

and $\mathcal{K}(p^t, \phi) \to \log C$ for a converging MCMC. Actually, this relation holds also for the estimates $\hat{\mathcal{K}}_N(p^t, \phi) = \hat{\mathcal{K}}_N(p^t, f) + \log C$. This is why in (3) we noticed that in the (estimate

of the) difference between the Kullback divergences issued from two MCMC strategies with marginal densities p_1^t and p_2^t, the unknown $\log C$ cancels out and $D(p_1^t, p_2^t, f) \to 0$ as $t \to \infty$ if both strategies are converging. In practice for large d, each Kullback estimate is biased. As seen previously this bias, which comes from the estimation of the entropy $\mathcal{H}(p)$ of any d-dimensional pdf p, depends on p and d and will be denoted $\text{bias}_N(p, d)$. To summarize the behavior of the estimates we can sketch their convergence in t, for fixed but large enough N so that the variance becomes neglictible but the bias still remains, which is what is achievable in practice for d large. Informally, for fixed t,

$$\hat{\mathcal{K}}_N(p^t, \phi) \approx \mathcal{K}(p^t, f) + \log C + \text{bias}_N(p^t, d).$$

We assume that $\text{bias}_N(p^t, d) \to \text{bias}_N(f, d)$ when $p^t \to f$, which seems fairly reasonnable and implies that $\text{bias}_N(p^t, d)$ and $\text{bias}_N(f, d)$ are of the same magnitude for p^t and f "close enough". This is needed in order to compare decays of $t \mapsto \hat{\mathcal{K}}(p^t, f)$ before convergence, and is supported by numerical evidence from other experiments we did (not provided here for brevity), and the fact that even a slow MCMC, if converging, should never be too far from its target, leading to similar biases.

For a MCMC strategy with marginals $p^t \to g$, we then have informally (i.e. forgetting as above the randomness and variance associated to the N chains) when $t \to \infty$,

$$\hat{\mathcal{K}}_N(p^t, \phi) \to \mathcal{K}(g, f) + \log C + \text{bias}_N(g, d). \tag{5}$$

For comparing two converging strategies $p_j^t \to f, j = 1, 2$, a possibility is to plot the difference $\hat{D}_N(p_1^t, p_2^t, f)$ which delivers the correct answer since the sign and slope of its decay towards 0 indicates the best strategy. But if one (or both) strategies are not converging, $D_N(p_1^t, p_2^t, f)$ can stabilizes at any value including 0 leading to wrong conclusion.

To be guarded against such situations we suggest a simpler method only using plots of $\hat{\mathcal{K}}_N(p^t, \phi)$'s: add to the competing strategies a "benchmark MCMC" with marginals p_B^t known to converge, i.e. for which $\hat{\mathcal{K}}_N(p_B^t, \phi) \to \log C + \text{bias}_N(f, d)$. Comparing decays and stabilization of the other challenging strategies against this benchmark stabilization value taking account for the bias delivers the correct answer (as we did in Section 3): any converging strategy stabilizes to the benchmark level, and any non converging MCMC stabilizes above that since $\mathcal{K}(g, f) > 0$. Hence this criterion provides both performance comparisons and MCMC convergence assessment. The only exception to that could be a very special situation where $\mathcal{K}(g, f) + \text{bias}_N(g, d) = \text{bias}_N(f, d)$.

5. Discussion

We have proposed a methodological approach to evaluate MCMC efficiency and control of convergence on the basis of intensive simulation only. The diagnostic is based on a practical, easy-to-understand graphical criterion. To evacuate the difficulty induced by the bias in high dimensions we have introduced a benchmark convergent MCMC which indicates when stabilization means convergence.

Since our method requires intensive simulations that may be computationally demanding, all our algorithms have been implemented in a package called EntropyMCMC for the R statistical software (R Core Team, 2013) that will be publicly available in a near futur. This package takes advantage of recent advances in parallel, High Performance Computing (HPC) using the Rmpi package (Yu, 2012). All the examples shown in this paper have been ran with this package on multicore workstations and the regional cluster CCSC[1].

[1] *Centre de Calcul Scientifique en région Centre*

These simulations (or part of it) from the best sampler are recyclable after comparisons, or can be re-used in the fly for statistical inference.

Recent researchs extend the NN idea to a k-th nearest neighbor distance estimate (Singh et al., 2003), see also Wang and Kulkarni (2009). There are some hope that these extensions together with recent computing strategies for computing approximate k-NN estimates reduce the bias in entropy estimation. However, it also brings back a tuning parameter (how to choose k) that plays somehow the role of the bandwidth in the kernel density estimation approach. Theoretically, NN-estimates consistency require mild peak and tail conditions as mentionned Section 2.2, like $\int |\log p(\theta)|^{1+\varepsilon} p(\theta)\, d\theta < \infty$ for the $1-$NN case (see Kozachenko and Leonenko, 1987). We have assumed here that this condition is reasonable enough to be satisfied for MCMC successive densities. Indeed, as detailed in Chauveau and Vandekerkhove (2013), the crucial point for MCMC successive densities seems to be smoothness more than tail conditions, one reason being that a MH kernel is a building block of many MCMC, including adaptive ones, and this kernel has a point mass at the chain's current position. Propagation of some tail conditions are already proved in Chauveau and Vandekerkhove (2013). All our experiments indicate numerical evidence of consistency, nevertheless it would be interesting to determine which minimal initial conditions on the algorithm imply propagation of the $1-$NN tail condition to the sequence of p^t's. All these considerations are perspectives for futur work.

References

Ahmad, I. A. and Lin, P. E. (1989). A nonparametric estimation of the entropy for absolutely continuous distributions. *IEEE Trans. Inform. Theory*, 36:688–692.

Andrieu, C. and Moulines, E. (2006). On the ergodicity properties of some adaptive markov chain monte carlo algorithms. *Ann. Appl. Prob.*, 16:1426–1505.

Andrieu, C. and Thoms, J. (2008). A tutorial on adaptive MCMC. *Stat. Comput.*, 18:343–373.

Bai, Y., Craiu, R. V., and Di Narzo, A. F. (2010). Divide and conquer: A mixture-based approach to regional adaptation for mcmc. *J. Comp. Graph. Stat.*, pages 1–17.

Bai, Y., Roberts, G. O., and Rosenthal, J. S. (2008). On the containment condition for adaptive markov chain monte carlo algorithms. Technical report, Dept. Satist. Univ. Toronto.

Beirlant, J., Dudewicz, E. J., Györfi, L., and van der Meulen, E. C. (1997). Nonparametric entropy estimation, an overview. *Int. J. Math. Stat. Sci.*, 6:17–39.

Chauveau, D. and Vandekerkhove, P. (2013). Smoothness of Metropolis-Hastings algorithm and application to entropy estimation. *ESAIM: Probability and Statistics*, 17:419–431.

Douc, R., Guillin, A., Marin, J., and Robert, C. (2007). Convergence of adaptive mixtures of importance sampling schemes. *Ann. Statist.*, 35(1):420–448.

Eggermont, P. P. B. and LaRiccia, V. N. (1999). Best asymptotic normality of the kernel density entropy estimator for smooth densities. *IEEE trans. Inform. Theory*, 45(4):1321–1326.

Fort, G., Moulines, E., Priouret, P., and Vandekerkhove, P. (2013). A central limit theorem for adaptive and interacting markov chains. *Bernoulli*, To appear.

Geman, S. and Geman, D. (1984). Stochastic relaxation, Gibbs distributions and the bayesian restoration of images. *IEEE Trans. Pattern Anal. Mach. Intell.*, 6:721–741.

Györfi, L. and Van Der Meulen, E. C. (1989). An entropy estimate based on a kernel density estimation. *Colloquia Mathematica societatis János Bolyai 57, Limit Theorems in Probability and Statistics Pécs*, pages 229–240.

Harremoes, P. and Holst, K. K. (2007). Convergence of Markov chains in information divergence. Technical report, Center for Mathematics and Computer Science, Amsterdam.

Hastings, W. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57:97–109.

Holden, L. (1998). Geometric convergence of the Metropolis-Hastings simulation algorithm. *Statistics and Probabilitiy Letters*, 39.

Kozachenko, L. and Leonenko, N. N. (1987). Sample estimate of entropy of a random vector. *Problems of Information Transmission*, 23:95–101.

Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A., and Teller, E. (1953). Equations of state calculations by fast computing machines. *J. Chem. Phys.*, 21:1087–1092.

R Core Team (2013). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Roberts, G. and Rosenthal, J. (2001). Optimal scaling for various Metropolis-Hastings algorithms. *Statistical Science*, 16:351–367.

Rosenthal, J. S. and Roberts, G. O. (2007). Coupling and ergodicity of adaptive mcmc. *J. Appl. Prob*, 44(458–475).

Singh, H., Misra, N., Hnizdo, V., Fedorowicz, A., and Demchuk, E. (2003). Nearest neighbor estimate of entropy. *American Journal of Mathematical and Management Sciences*, 23(3):301–321.

Sricharan, K., Wei, D., and Hero III, A. O. (2013). Ensemble estimators for multivariate entropy estimation. http://arxiv.org/abs/1203.5829v3.

Stowell, D. and Plumbley, M. D. (2009). Fast multidimensional entropy estimation by k-d partitioning. *IEEE Signal Processing Letters*, 16(6):537–540.

Thompson, M. (2010). *SamplerCompare: A framework for comparing the performance of MCMC samplers*. R package version 1.0.1.

Vrugt, J. A., Braak, C., Diks, C., Robinson, B. A., Hyman, J. M., and Higdon, D. (2009). Accelerating markov chain monte carlo simulation by differential evolution with self-adaptive randomized subspace sampling. *International Journal of Nonlinear Sciences & Numerical Simulation*, 10(3):271–288.

Wang, Q. and Kulkarni, S. R. (2009). Divergence estimation for multidimensional densities via k-nearest-neighbor distances. *IEEE Transactions on Information Theory*, 55(5):2392–2405.

Yu, H. (2012). *Rmpi: Interface (Wrapper) to MPI (Message-Passing Interface)*.

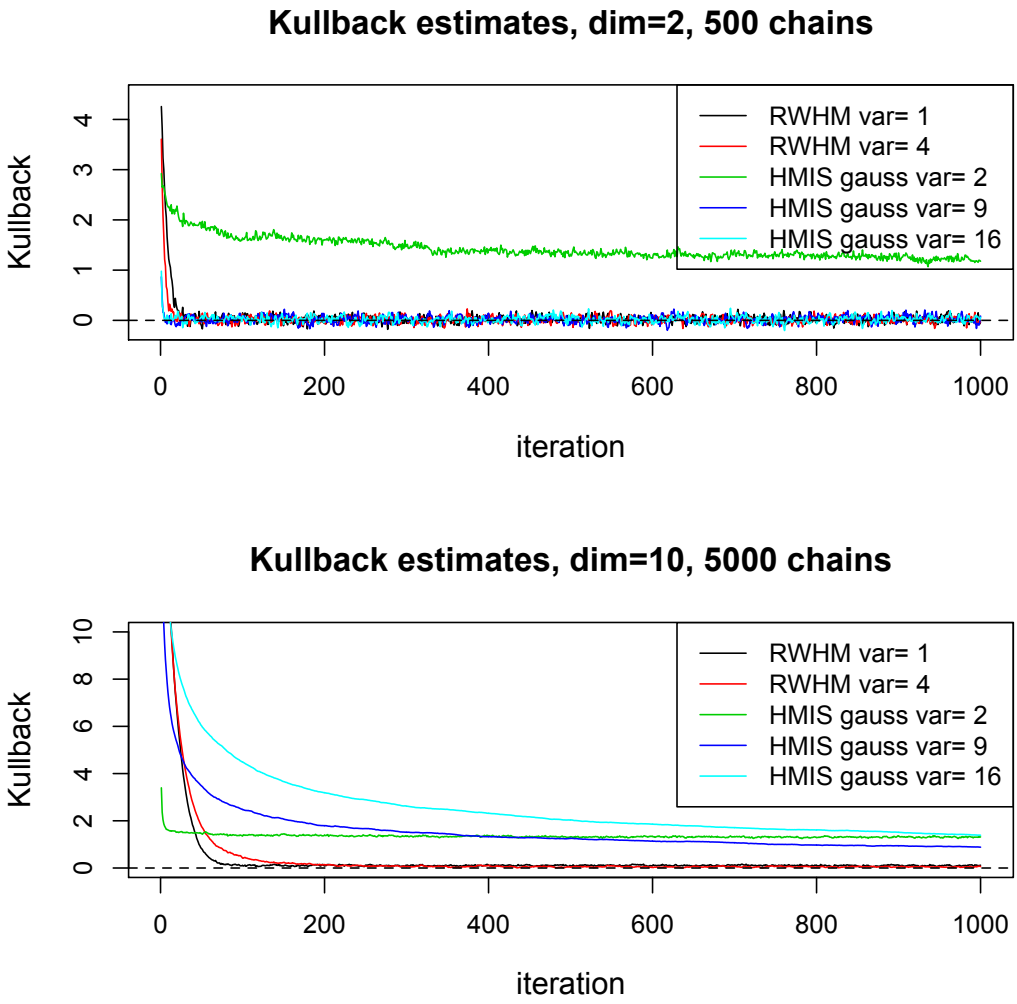

Figure 1: *Kullback estimates* $t \mapsto \hat{K}_N(p^t, f)$ *for* $n = 1000$ *iterations of the 5 MCMC strategies RW1 (black), RW4 (red), IS2 (green), IS9 (blue), IS16 (light blue). Top:* $d = 2$ *and* $N = 500$ *chains; bottom:* $d = 10$ *and* $N = 5000$ *chains.*

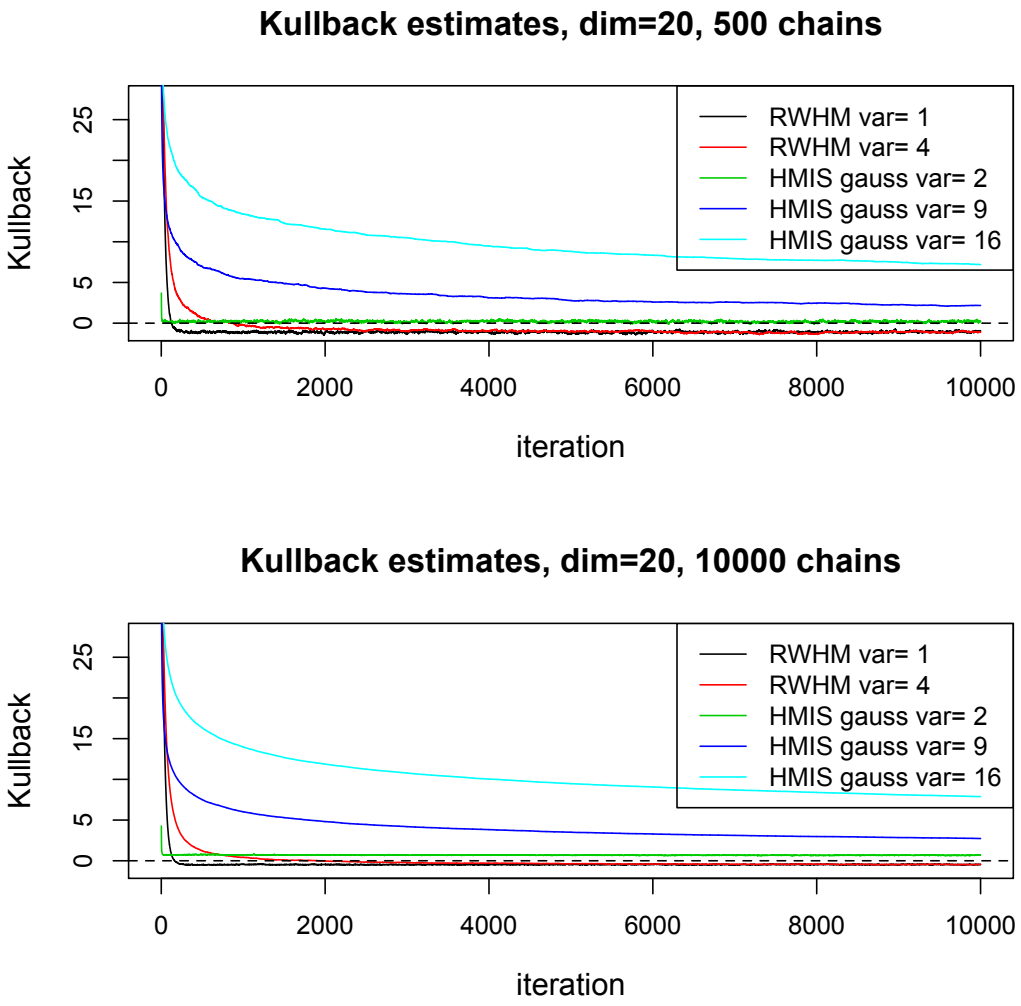

Figure 2: *Kullback estimates* $t \mapsto \hat{K}_N(p^t, f)$ *for* $d = 20$ *and* $n = 10,000$ *iterations of the 5 MCMC strategies RW1 (black), RW4 (red), IS2 (green), IS9 (blue), IS16 (light blue). Top:* $N = 500$ *chains; bottom:* $N = 10,000$ *chains.*