An R package using HPC for entropy estimation and MCMC evaluation

D. Chauveau[a] and **P. Vandekerkhove**[b]

[a]MAPMO - Fédération Denis Poisson
Université d'Orléans et CNRS UMR 7349
BP 6759, 45067 Orléans cedex 2
didier.chauveau@univ-orleans.fr

[b]LAMA - CNRS UMR 8050
Université de Marne-la-Vallée
77454 Marne-la-Vallée cedex 2
Pierre.Vandekerkhove@univ-mlv.fr

Mots clefs : Entropy estimation, High Performance Computing, Kullback divergence, MCMC algorithms, Nonparametric statistics, Rmpi library.

Many recent (including adaptive) MCMC methods are associated in practice to unknown rates of convergence, leading to difficulties in assessing performance of specific MCMC samplers. Comparison or evaluation of MCMC samplers is now a challenge addressed by various approaches (see, e.g., the recent SamplerCompare package [6]). Let f be a d-dimensional target density of a MCMC algorithm, and p^t the marginal density of the algorithm at "time" (iteration) t. In Chauveau and Vandekerkhove [1], we have first proposed to evaluate a MCMC sampler efficiency from a Kullback divergence criterion,

$$\mathcal{K}(p^t, f) = \int p^t \log\left(\frac{p^t}{f}\right) = \mathcal{H}(p^t) - \mathbb{E}_{p^t}[\log f].$$

where $\mathcal{H}(p) := \mathbb{E}_p[\log p] = \int p \log p$ is the differential entropy of a probability density p over $\mathbb{R}^d$. We have introduced a simulation-based methodology allowing to estimate the entropy of the algorithm successive densities, $\mathcal{H}(p^t)$, based on the "parallel " simulation of N iid copies of (eventually Markov) chains at step t, resulting in a N-sample $\boldsymbol{X}^t$ iid$\sim p^t$, for $t \geq 1$. These simulations are first used to estimate $\mathbb{E}_{p^t}[\log f]$ (or more generally an estimate $\propto \mathbb{E}_{p^t}[\log f]$ if the normalizing constant of f is unknown) via standard Monte Carlo integration. The sample $\boldsymbol{X}^t$ is also used to compute an estimate of $\mathcal{H}(p^t)$, and we have proved in [1] some consistency results in this MCMC context for an entropy estimate based on Monte-Carlo integration of a kernel density estimate introduced by Györfi and Van Der Meulen [3]. Unfortunately, this estimate deteriorates as dimension increase, and require some parameters (like, e.g., the kernel bandwidth) whose tuning is challenging in practice.
We investigate here an alternative strategy based on Nearest Neighbor (NN) estimates of differential entropy, initiated by Kozachenko and Leonenko [4],

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

where $C_E = -\int_0^\infty e^{-t}\log t\, dt$ is the Euler constant, $C_1(d) = \frac{\pi^{d/2}}{\Gamma(d/2+1)}$ and where ρ_i is the nearest neighbor (Euclidean) distance from the ith point to the other points in the sample $\boldsymbol{X}^t$.

Kozachenko and Leonenko [4] proved, under mild conditions a mean square consistency of $\hat{\mathcal{H}}_N(p^t)$ for any dimension d. However, apparently, this NN approach has been used and studied mostly in univariate or bivariate ($d = 2$) situations (e.g., in image processing). We show that, in MCMC setup where moderate to large dimensions are common, this estimate seems more promising than kernel density estimates, both from an operational point of view (no tuning parameters like the bandwidth), and from a computational point of view (the nearest distance can be computed faster than a multivariate kernel density estimate in high dimension). Entropy estimation is also considered in other fields, and recent researchs extend the NN idea to a k-th nearest distance estimate (see, e.g., Singh et al. [5]), that we plan to investigate as well.
The computational burden required by our method can be heavy (depending on the dimension, kernel complexity, number of iid chains). We thus implement all our algorithms (iid MCMC simulation plus entropy and Kullback estimation) in the R package EntropyMCMC, which takes advantage of recent advances of High Performance Computing in R. This package can use MCMC output from samplers and target distributions implemented in other packages, such as, e.g., SamplerCompare [6]. The end user can also run its own MCMC inside the package by just providing R definitions for its target and, e.g., the proposal for standard Hastings-Metropolis or Independence samplers. Several functions are written with appropriate C and R code for running it on multicore computers, network of workstations or actual clusters, using e.g., the Rmpi library. We illustrate its usage for studying the behavior of the NN estimate in MCMC setup and moderate to large dimensions, using the cluster *Centre de Calcul Scientifique en région Centre* (`http://cascimodot.fdpoisson.fr/?q=ccsc`).

References

[1] Chauveau, D. and Vandekerkhove, P. (2012). Smoothness of Metropolis-Hastings algorithm and application to entropy estimation. *ESAIM: Probability and Statistics*.

[2] et Modélisation Orléans Tours, C. S. (2011). Centre de calcul scientifique en région centre.

[3] 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.

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

[5] 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.

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