FULLY ADAPTIVE GAUSSIAN MIXTURE METROPOLIS-HASTINGS ALGORITHM

David Luengo

Dept. of Circuits and Systems Engineering
Univ. Politécnica de Madrid (Spain)

Luca Martino

Dept. of Signal Theory and Communications
Univ. Carlos III de Madrid (Spain)

ABSTRACT

Markov Chain Monte Carlo methods are widely used in signal processing and communications for statistical inference and stochastic optimization. In this work, we introduce an efficient adaptive Metropolis-Hastings algorithm to draw samples from generic multimodal and multidimensional target distributions. The proposal density is a mixture of Gaussian densities with all parameters (weights, mean vectors and covariance matrices) updated using all the previously generated samples applying simple recursive rules. Numerical results for the one and two-dimensional cases are provided.

Index Terms— Markov Chain Monte Carlo (MCMC), Gaussian mixtures, adaptive Metropolis-Hastings

1. INTRODUCTION

Markov Chain Monte Carlo (MCMC) methods [1, 2] are ubiquitously used for performing inference and solving optimization problems in many scientific fields: statistics, digital communications, machine learning, signal processing, etc. [3, 4, 5, 6]. MCMC approaches are able to generate samples virtually from any target distribution (known up to a normalizing constant) by using a simpler proposal distribution. The basic underlying idea of standard MCMC techniques is producing a Markov chain that converges to the target.

The most famous MCMC technique is the Metropolis-Hastings (MH) algorithm [7, 8, 3]. However, the main drawback of the MH method (and in general of all MCMC methods) is that the correlation among the samples in the Markov chain can be very high when the acceptance rate is low [1, 2, 9]. Correlated samples provide less statistical information and the resulting chain can remain trapped almost indefinitely in a local mode, meaning that convergence can be extremely slow. Therefore, since the correlation depends on the discrepancy between the target and proposal distributions, we would like the proposal to be as close to the target as possible.

Several extensions have been proposed in the literature to speed up the convergence and reduce the so called "burn-in" period. Among them, adaptive MH methods (i.e., MH algorithms with adaptive proposal distributions) are particularly interesting [2, 10]. Indeed, MCMC techniques usually need the selection of several parameters by the user before they can be applied to any particular problem. The use of adaptive proposals overcomes this issue, providing *black-box* algorithms with *self-tuning* capabilities. An adaptive MH technique improves the proposal distribution by learning at least some of its parameters from all the previously generated samples. Unfortunately, an important problem with the adaptation of the proposal is that the Markov property is lost and the invariant distribution of the chain could be disturbed. Hence, adaptive MH algorithms must be carefully designed to avoid this issue.

The adaptive Metropolis (AM), a random walk MH algorithm using an adaptive Gaussian proposal, was introduced in [11]. The covariance matrix of the proposal is updated using recursive empirical estimators applied to the samples generated by the chain. The AM algorithm is an example of a partially adaptive MH approach, since it only updates the covariance of the proposal, whereas the mean of the Gaussian jumps to the current state of the chain at each iteration. An attempt of extending the AM algorithm by using a mixture of Gaussians as a proposal and updating all of its parameters (thus obtaining a fully adaptive MH algorithm) can be found in [12]. However, the resulting algorithm is quite complicated, and the proposal is only updated at some iterations.

In this work, we introduce an independent MH technique where the proposal PDF is an adaptive mixture of Gaussians. All the parameters (weights, means and covariance matrices) of the Gaussians in the mixture are updated using empirical estimators with simple recursive formulas (i.e., our method is fully adaptive). After a training period, the proposal is adapted at every iteration. The resulting AGM-MH algorithm can be used to draw samples from arbitrary multimodal and multidimensional targets, always improving the performance w.r.t. a non-adaptive MH scheme using the initial proposal.

The rest of the paper is organized as follows. Section 2 shows the problem formulation. Section 3 presents the proposed AGM-MH algorithm. Sections 4 and 5 describe efficient parameter update rules and black-box usage. Finally, Sections 6 and 7 show the results and conclude the paper.

*This work has been partly supported by the Spanish government through projects COMONSENS (CSD2008-00010), DEIPRO (TEC2009-14504-C02-01), ALCIT (TEC2012-38800-C03-01), COMPREHENSION (TEC2012-38883-C02-01) and DISSECT (TEC2012-38058-C03-01).

2. PROBLEM FORMULATION

Let us assume that we need to draw samples from a (possibly multimodal) generic d-dimensional target probability density function (PDF), $p_o(\mathbf{x})$, with support $\mathcal{D} \subseteq \mathbb{R}^d$. The AM algorithm [11] uses an adaptive random walk MH with a Gaussian proposal, mean equal to the previous state of the chain ($\boldsymbol{\mu} = \mathbf{x}_{t-1}$), and covariance matrix, $\mathbf{C}$, estimated from all previous states, i.e., $q(\mathbf{x}|\mathbf{x}_{t-1}, \mathbf{C}) \propto \mathcal{N}(\mathbf{x}|\mathbf{x}_{t-1}, \mathbf{C})$, where

$$\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \mathbf{C}) \propto \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^\top \mathbf{C}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right) \quad (1)$$

denotes a standard multivariate Gaussian PDF. In order to improve the performance of the AM algorithm, here we consider a mixture of N Gaussians as the proposal PDF, i.e.,

$$q(\mathbf{x}|\mathbf{w}, \boldsymbol{\mu}_{1:N}, \mathbf{C}_{1:N}) = \sum_{i=1}^{N} w_i \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_i, \mathbf{C}_i), \quad (2)$$

where $\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_i, \mathbf{C}_i)$ is given by (1), $\boldsymbol{\mu}_i = [\mu_{i,1}, ..., \mu_{i,d}]^\top$ is the $d \times 1$ mean vector, $\mathbf{C}_i$ is the $d \times d$ positive definite covariance matrix, and $\mathbf{w} = [w_1, ..., w_N]^\top$ are the normalized weights (i.e., $w_1 + ... + w_N = 1$). Moreover, we define $\boldsymbol{\mu}_{1:N} = [\boldsymbol{\mu}_1, ..., \boldsymbol{\mu}_N]$ and $\mathbf{C}_{1:N} = [\mathbf{C}_1, ..., \mathbf{C}_N]$. We note that our approach is a fully adaptive MH algorithm, since (unlike the AM algorithm, which only updates the covariance) all the parameters in the mixture are learnt from all the previously generated samples. The resulting algorithm is very simple, since the adaptation is based on empirical estimators that can be implemented efficiently using recursive formulas.

Since the adaptation could disturb the convergence of the generated chain to the target PDF, we consider the possibility of stopping it at an iteration T_{stop}. Hence, for $t > T_{stop}$ our algorithm is a standard MH with an improved proposal PDF w.r.t. the initial choice, thus providing a better performance and guaranteeing convergence. However, the numerical results described in Section 6 show that the algorithm seems to maintain the correct ergodicity properties, so we always use $T_{stop} = T_{tot}$. A theoretical convergence proof is under development and will be included in future works. Finally, we note that degeneracy problems can appear during the first iterations in the update of the covariance matrices if we have a poor initialization. In order to avoid this issue, we allow the method to use a few iterations ($t = 1, \ldots, T_{train}$) to collect information about the target, assigning the produced state of the chain to the closest Gaussian in the mixture, as in [11].

3. AGM-MH ALGORITHM

The proposed AGM-MH algorithm is described below. First of all, notice that, during the first T_{train} time steps the algorithm simply assigns the current state of the chain, $\mathbf{x}_t$, to a Gaussian among the N in the mixture, according to the minimum Euclidean distance between $\mathbf{x}_t$ and the means $\boldsymbol{\mu}_i^{(t-1)}$, $i = 1, ..., N$. Afterwards, the algorithm updates all the parameters in the mixture until $t = T_{stop}$, when adaptation is stopped. In the description of the algorithm, the parameters are updated using a block procedure, but efficient recursive update formulas can be obtained, as shown in Section 4.

1. **Initialization:**
   (a) *Time instants:* Set $t = 0$. Choose an initial state, $\mathbf{x}_0 \in \mathcal{D}$, and positive integers, T_{train}, T_{stop} and T_{tot}, such that $T_{train} < T_{stop} \leq T_{tot}$.
   (b) *Proposal:* Choose the number of Gaussians N, as well the initial settings for $\boldsymbol{\mu}_{1:N}^{(0)} = [\boldsymbol{\mu}_1^{(0)}, ..., \boldsymbol{\mu}_N^{(0)}]$ and $\mathbf{C}_{1:N}^{(0)} = [\mathbf{C}_1^{(0)}, ..., \mathbf{C}_N^{(0)}]$. Set $\mathbf{w}^{(0)} = \frac{1}{N}\mathbf{1}_N$, where $\mathbf{1}_N$ is an $N \times 1$ vector of ones.
   (c) *Auxiliary parameters:* For $i = 1, ..., N$, define $\mathbf{S}_i^{(0)} = [\mathbf{s}_i^{(1)} = \boldsymbol{\mu}_i^{(0)}]$, with $m_i = 1$ denoting the number of columns of $\mathbf{S}_i^{(0)}$. Let ϵ be a small constant value and $\mathbf{I}_d$ an identity matrix.
2. **MH steps:**
   (a) Sample $\mathbf{x}'$ from a mixture of Gaussian PDFs,
   $$\mathbf{x}' \sim q_t(\mathbf{x}|\mathbf{w}^{(t)}, \boldsymbol{\mu}_{1:N}^{(t)}, \mathbf{C}_{1:N}^{(t)}).$$
   (b) Accept $\mathbf{x}_{t+1} = \mathbf{x}'$ with probability
   $$\alpha = \min\left[1, \frac{p(\mathbf{x}')q(\mathbf{x}_t|\mathbf{w}^{(t)}, \boldsymbol{\mu}_{1:N}^{(t)}, \mathbf{C}_{1:N}^{(t)})}{p(\mathbf{x}_t)q(\mathbf{x}'|\mathbf{w}^{(t)}, \boldsymbol{\mu}_{1:N}^{(t)}, \mathbf{C}_{1:N}^{(t)})}\right]. \quad (3)$$
   Otherwise, set $\mathbf{x}_{t+1} = \mathbf{x}_t$.
3. **Update the parameters of the proposal ($t < T_{stop}$):**
   (a) Find the closest Gaussian to $\mathbf{x}_{t+1}$ (w.r.t. the Euclidean distance), i.e., find the index
   $$j = \arg\min_i |\boldsymbol{\mu}_i^{(t)} - \mathbf{x}_{t+1}|^2. \quad (4)$$
   (b) Set $m_j = m_j + 1$ and update (adding a new column) the j-th auxiliary matrix
   $$\mathbf{S}_j^{(t+1)} = [\mathbf{S}_j^{(t)}, \mathbf{s}_j^{(m_j)} = \mathbf{x}_{t+1}], \quad (5)$$
   whereas $\mathbf{S}_i^{(t+1)} = \mathbf{S}_i^{(t)}$, for all $i \neq j$.
   (c) If $t > T_{train}$: update the parameters of the j-th Gaussian,
   $$\boldsymbol{\mu}_j^{(t+1)} = \frac{1}{m_j}\sum_{i=1}^{m_j} \mathbf{s}_j^{(i)}, \quad (6)$$
   and
   $$\mathbf{C}_j^{(t+1)} = \frac{\tilde{\mathbf{S}}_j^{(t+1)}\left[\tilde{\mathbf{S}}_j^{(t+1)}\right]^\top + (m_j - 1)\epsilon\mathbf{I}_d}{m_j - 1}, \quad (7)$$

where $\tilde{\mathbf{S}}_j^{(t+1)} = \mathbf{S}_j^{(t+1)} - \boldsymbol{\mu}_j^{(t+1)} \otimes \mathbf{1}_{m_j}^\top$, with $\otimes$ denoting the Kronecker product [13]. Set $\boldsymbol{\mu}_i^{(t+1)} = \boldsymbol{\mu}_i^{(t)}$ and $\mathbf{C}_i^{(t+1)} = \mathbf{C}_i^{(t)}$, $\forall i \neq j$. Since m_j has been incremented, we also need to update the weights,

$$w_i^{(t+1)} = \frac{m_i}{\sum_{k=1}^N m_k}, \quad i = 1, ..., N, \tag{8}$$

so that $\mathbf{w}^{(t+1)} = [w_1^{(t+1)}, ..., w_N^{(t+1)}]^T$.

4. If $t < T_{tot}$, return to step 2 with $t = t + 1$.

Observe that the proposal PDF is only updated for $t > T_{train}$. Moreover, note that the matrices $\mathbf{S}_i^{(t)}$ and $\tilde{\mathbf{S}}_i^{(t)}$ have dimension $d \times m_i$ for any $i \in \{1, ..., N\}$, so that $\mathbf{C}_i^{(t)}$ always has dimension $d \times d$. The term $\epsilon \mathbf{I}_d$ is used to avoid numerical problems (the matrix $\mathbf{C}_i^{(t)}$ must be positive definite), as in [11].

4. EFFICIENT RECURSIVE UPDATE OF THE PARAMETERS

To update the parameters of the selected (j-th) Gaussian PDF in the mixture, we can use recursive expressions. Indeed, recalling that, in step 3b of the algorithm, $m_j = m_j + 1$ has already been updated and $\mathbf{s}_j^{(m_j)} = \mathbf{x}_{t+1}$, Eq. (6) becomes

$$\boldsymbol{\mu}_j^{(t+1)} = \frac{1}{m_j}\mathbf{x}_{t+1} + \frac{m_j - 1}{m_j}\boldsymbol{\mu}_j^{(t)}, \tag{9}$$

and (7) can be rewritten as

$$\mathbf{C}_j^{(t+1)} = \frac{m_j - 2}{m_j - 1}\mathbf{C}_j^{(t)} + \frac{1}{m_j - 1}\left[\frac{(\mathbf{x}_{t+1} - \boldsymbol{\mu}_j^{(t+1)})(\mathbf{x}_{t+1} - \boldsymbol{\mu}_j^{(t+1)})^T}{m_j} + \epsilon \mathbf{I}_d\right]. \tag{10}$$

Finally, note that

$$\sum_{k=1}^N m_k = t + 1 + N,$$

so that

$$w_i^{(t+1)} = \frac{m_i}{t + N + 1}, \quad i = 1, ..., N. \tag{11}$$

for $t > T_{train}$. In this way, the novel technique becomes computationally efficient even for high dimensional problems.

5. AGM-MH AS BLACK-BOX METHOD

The AGM-MH method shows sensitive dependence on the initial conditions. If some prior information about the target is available, it can be used to choose the initial parameters. If no prior informations is available, the AGM-MH can be applied as black-box algorithm in the following way:

- Use a large number of Gaussians, N (typically N must increase as the dimension, d, increases).
- Select randomly the means $\boldsymbol{\mu}_{1:N}^{(0)}$ in order to cover as much as possible of the the target's domain, $\mathcal{D} \subseteq \mathbb{R}^d$.
- Choose diagonal covariance matrices, $\mathbf{C}_i^{(0)} = \sigma_i^2 \mathbf{I}_d$ ($i = 1, \ldots, N$), with a large value of σ_i^2 in order to be able to explore the domain of interest, $\mathcal{D} \subseteq \mathbb{R}^d$, during the training period, $t \leq T_{train}$.
- The parameter T_{train} should increase as the problem's dimensions, d, increases. Numerical results suggest that $T_{train} = 100d$ can be a suitable choice, although for very complicated target distributions a higher value of T_{train} may be needed.

Finally, we remark that the use of a huge number of Gaussians does not involve computational problems, since the weights of the irrelevant Gaussians quickly tend to zero. Hence, the computational cost is controlled by the adaptation, so that only the Gaussians located close to high probability regions survive. The useless Gaussian PDFs, located far away from the modes of the target, are virtually discarded.

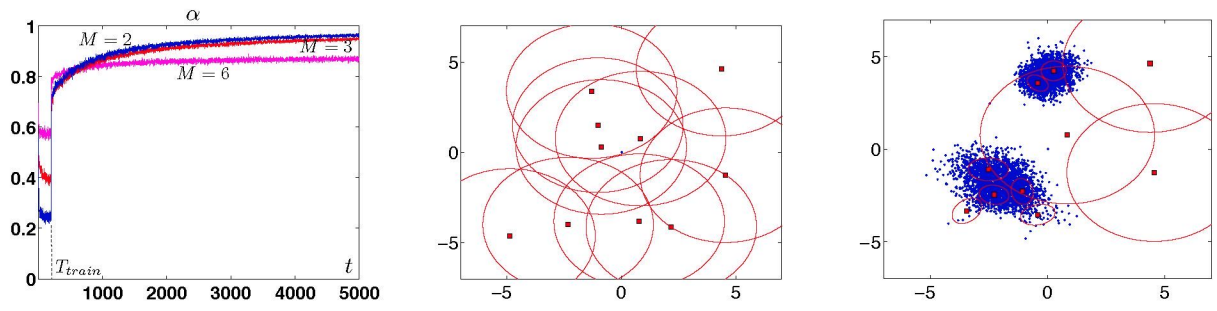
6. SIMULATIONS

6.1. Example 1

In this toy example, we apply the AGM-MH method to draw samples from a univariate bimodal target PDF defined as

$$\begin{aligned} p_o(x) \propto p(x) &= \exp\left(-\frac{(x^2 - 4)^2}{4}\right) \\ &= \exp\left(-\frac{x^4 - 4x^2 + 16}{4}\right), \end{aligned} \tag{12}$$

which has two modes at $x = \pm 2$. We set the number of Gaussians in the proposal PDF to $N = 2$, with $w_i^{(0)} = 0.5$ and $\sigma_i^2 = 10$ for $i = 1, 2$. The two initial means are chosen randomly with uniform distributions in $[-4, 0]$ and $[0, 4]$ respectively, i.e., $\mu_1^{(0)} \sim \mathcal{U}([-4, 0])$ and $\mu_2^{(0)} \sim \mathcal{U}([0, 4])$. We perform $T_{tot} = 5000$ iterations of the chain, setting $T_{train} = 200$ and $T_{stop} = T_{tot}$ (i.e., the adaptation is never stopped). The initial state is randomly chosen as $x_0 \sim \mathcal{N}(x|0, 1)$. We use *all* the generated samples to estimate the mean of the target (which is equal to zero, since $p(x)$ is symmetric). The mean square error (MSE) of the estimation (averaged over 2000 runs) is $15 \cdot 10^{-4}$, whereas the estimated linear correlation between contiguous samples is 0.18. Without using any adaptation (i.e., using a standard MH) the correlation is ≈ 0.78. Hence, we can observe that the correlation decreases using the AGM-MH algorithm. The final averaged locations of the means of the two Gaussians in the proposal are $\mu_1^{(T_{tot})} \approx -1.88$ and $\mu_2^{(T_{tot})} \approx 1.88$, with weights $w_i^{(T_{tot})} \approx 0.5$ and variances $\sigma_i^2 \approx 0.16$ for $i = 1, 2$.



(a) (b) (c)

Fig. 1. **(a)** Averaged values of α as a function of the iteration index, t, for $M = 2, 3, 6$ in Example 2. For $t > T_{train}$, α increases as the proposal becomes closer to the target. **(b)** Initial configuration of the means (squares) and the covariance matrices (ellipses) in Example 3. **(c)** Final configuration (red) and drawn samples (blue), at $t = T_{tot} = 7000$, for Example 3.

6.2. Example 2

For the sake of simplicity, we consider again a univariate target density. However, now we consider that the target distribution is itself a mixture of Gaussian PDFs. Specifically, the target PDF is formed by M Gaussians, i.e.,

$$p_o(x) \propto p(x) = \sum_{i=1}^{M} a_i \mathcal{N}(x|\eta_i, \rho_i^2), \tag{13}$$

where the weights are $a_i = 1/M$ and the variances $\rho_i^2 = 4$ for $i = 1, ..., M$. We consider three different cases with $M \in \{2, 3, 6\}$. The means are, for each case:

- $\underline{M = 2}$: $\eta_1 = -10$ and $\eta_2 = 10$.
- $\underline{M = 3}$: $\eta_1 = -10$, $\eta_2 = 0$ and $\eta_3 = 10$.
- $\underline{M = 6}$: $\eta_1 = -15$, $\eta_2 = -10$, $\eta_3 = -5$, $\eta_4 = 5$, $\eta_5 = 10$ and $\eta_6 = 15$.

In the proposal we also use $N = M$ Gaussians, with the initial means chosen uniformly in $[-20, 20]$, the initial variances set to $\sigma_i^2 = 10$ and the weights to $w_i^{(0)} = 1/N$ for all $i = 1, ..., N$. As in the first example, we perform $T_{tot} = 5000$ iterations of the chain, setting $T_{train} = 200$ and $T_{stop} = T_{tot}$ (i.e., the adaptation is never stopped), and the initial state of the chain is randomly chosen as $x_0 \sim \mathcal{N}(x|0, 1)$. We use all the generated samples to estimate the normalizing constant of the target. The mean square errors (MSE) of the estimations (averaged over 1000 runs) are $1.6 \cdot 10^{-4}$, $1.1 \cdot 10^{-4}$ and $2 \cdot 10^{-5}$ for $M = 2, 3, 6$ respectively. The resulting correlations are 0.13, 0.14 and 0.16 for $M = 2, 3, 6$. In comparison, with a standard MH (i.e., without adaptation) the correlations are 0.81, 0.72 and 0.46 for $M = 2, 3, 6$. Thus, we remark again that the adaptation provided by the AGM-MH algorithm reduces considerably the correlation among the generated samples. Finally, Figure 1(a) depicts the averaged value of the acceptance function, α in (3), as a function of t for different values of M. Note that, for $t > T_{train}$, the averaged values of α increase as the proposal becomes closer to the target.

6.3. Example 3

In this example, our goal is drawing samples from a bivariate target PDF using the AGM-MH algorithm as a black-box technique. Just for simplicity, we also consider a bivariate mixture of $M = 2$ Gaussians as target distribution, with weights $a_1 = a_2 = 0.5$, means $\boldsymbol{\eta}_1 = [-2, -2]^\top$ and $\boldsymbol{\eta}_2 = [0, 4]^\top$, and covariance matrices $\boldsymbol{\Sigma}_1 = [0.3,\ 0.1; 0.1,\ 0.3]$ and $\boldsymbol{\Sigma}_2 = [0.8,\ -0.3; -0.3,\ 0.8]$. We set $T_{tot} = 7000$, $T_{train} = 200$ and $T_{stop} = T_{tot}$ (i.e., the adaptation is never stopped). On the one hand, we use for the proposal $N = 2$ Gaussian PDFs with $w_i^{(0)} = 0.5$ and $\mathbf{C}_i^{(0)} = 10\mathbf{I}_d$ for $i = 1, 2$. The means are selected uniformly, $\boldsymbol{\mu}_1 \sim \mathcal{U}([-5, 5] \times [0, 5])$ and $\boldsymbol{\mu}_2 \sim \mathcal{U}([-5, 5] \times [-5, 0])$. In this case, all the parameters of the mixture in the proposal always converge to the corresponding values of the mixture in the target PDF. On the other hand, we also consider the case with $N = 10$. We set $w_i^{(0)} = 0.1$, $\mathbf{C}_i^{(0)} = 10\mathbf{I}_d$ and $\boldsymbol{\mu}_i \sim \mathcal{U}([-5, 5] \times [-5, 5])$ for $i = 1, \ldots, 10$. In this situation, the AGM-MH algorithm improves the initial proposal PDF, updating the parameters of the Gaussians placed in good locations, whereas the weights of the unhelpful Gaussians decrease quickly to zero and their parameters remain invariant, as shown in Fig. 1.

7. DISCUSSION

We have proposed a novel adaptive independent MH algorithm (AGM-MH) to draw samples from arbitrary multimodal and multidimensional targets. AGM-MH builds on the work of [11], extending it by using a Gaussian mixture proposal and updating also the means and the weights of the Gaussians. Compared to a previous extension provided by [12], our approach is more efficient, updating the proposal at every iteration instead of only at a fixed number of iterations.

References

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

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

[3] C. P. Robert and G. Casella, *Monte Carlo Statistical Methods*, Springer, 2004.

[4] Xiaodong Wang, Rong Chen, and Jun S. Liu, "Monte Carlo Bayesian signal processing for wireless communications," *Journal of VLSI Signal Processing*, vol. 30, pp. 89–105, 2002.

[5] Christophe Andrieu, Nando De Freitas, Arnaud Doucet, and Michael I. Jordan, "An introduction to MCMC for machine learning," *Machine Learning*, vol. 50, pp. 5–43, 2003.

[6] W. J. Fitzgerald, "Markov chain Monte Carlo methods with applications to signal processing," *Signal Processing*, vol. 81, no. 1, pp. 3–18, January 2001.

[7] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller, "Equations of state calculations by fast computing machines," *Journal of Chemical Physics*, vol. 21, pp. 1087–1091, 1953.

[8] W. K. Hastings, "Monte Carlo sampling methods using Markov chains and their applications," *Biometrika*, vol. 57, no. 1, pp. 97–109, 1970.

[9] L. Martino and J. Míguez, "Generalized rejection sampling schemes and applications in signal processing," *Signal Processing*, vol. 90, no. 11, pp. 2981–2995, November 2010.

[10] L. Martino, J. Read, and D. Luengo, "Improved adaptive rejection Metropolis sampling algorithms," *arXiv:1205.5494*, May 2012.

[11] H. Haario, E. Saksman, and J. Tamminen, "An adaptive Metropolis algorithm," *Bernoulli*, vol. 7, no. 2, pp. 223–242, April 2001.

[12] P. Giordani and R. Kohn, "Adaptive independent Metropolis-Hastings by fast estimation of mixtures of normals," *Journal of Computational and Graphical Statistics*, vol. 19, no. 2, pp. 243–259, September 2010.

[13] Charles F. Van Loan, "The ubiquitous Kronecker product," *Journal of Computational and Applied Mathematics*, vol. 123, pp. 85–100, 2000.