# Medical Image Retrieval based on the Parallelization of the Cluster Sampling Algorithm

Hesham Arafat Ali
Computer Engineering and Systems Department,
Faculty of Engineering
Mansoura University, Egypt

Salah Attiya
Computer Engineering and Systems Department,
Faculty of Engineering
Mansoura University, Egypt

Ibrahim El-henawy
Computer Science Department,
Faculty of Computers and Informatics
Zagazig University, Egypt

***Abstract*—Cluster sampling algorithm is a scheme for sequential data assimilation developed to handle general non-Gaussian and nonlinear settings. The cluster sampling algorithm can be used to solve a wide spectrum of problems that requires data inversion such as image retrieval, tomography, weather prediction amongst others. This paper develops parallel cluster sampling algorithms, and show that a multi-chain version is embarrassingly parallel, and can be used efficiently for medical image retrieval amongst other applications. Moreover, it presents a detailed complexity analysis of the proposed parallel cluster samplings scheme and discuss their limitations. Numerical experiments are carried out using a synthetic one dimensional example, and a medical image retrieval problem. The experimental results show the accuracy of the cluster sampling algorithm to retrieve the original image from noisy measurements, and uncertain priors. Specifically, the proposed parallel algorithm increases the acceptance rate of the sampler from $45\%$ to $81\%$ with Gaussian proposal kernel, and achieves an improvement of $29\%$ over the optimally-tuned Tikhonov-based solution for image retrieval. The parallel nature of the proposed algorithm makes the it a strong candidate for practical and large scale applications.**

***Keywords*—Bayes' theorem; Hamiltonian Monte-Carlo; Inverse problems; Markov chain Monte-Carlo; Medical image reconstruction; Parallel programming**

## I. Introduction

Signal retrieval from noisy measurements (observations) involves solving an inverse problem. Inverse problems are essential in many fields such as image reconstruction or retrieval, tomography, weather prediction, and other predictions based on space-time models. The solution of inverse problems usually employs a data assimilation (DA) methodology [2], [5], [6], [10] [13], [16]. DA refers to the process of fusing information about a physical system obtained from different sources in order to produces more accurate conclusions about the physical system of concern. Two approaches are widely employed to solve an inverse problem. The first approach is a variational approach that involves solving an optimization problem with a regularized solution. The second approach is the statistical formulation of the DA problem which incorporates a prior distribution that encapsulates the knowledge about the system produced by the model, prior to the incorporation of any other source of information. Given the prior information, a likelihood function, the posterior is formulated as best estimate of the truth. Markov Chain Monte-Carlo (MCMC) is one of the most powerful simulation techniques for sampling a high-dimensional probability distribution, given only it's shape function, without the intrinsic need to the associated scaling factor. HMC sampling filter [3] is an accelerated Markov chain Monte-Carlo (MCMC) algorithm for solving the non-Gaussian sequential DA "filtering problem". This algorithm works by sampling the posterior distribution to produce description of the system state along with associated uncertainty. Specifically, these algorithms follow a Hamiltonian Monte-Carlo (HMC) approach to sample the posterior.

Cluster sampling filters ($\mathcal{C}\ell$HMC, and MC-$\mathcal{C}\ell$HMC) [1] are developed as extension of the Hamiltonian Monte-Carlo (HMC) sampling filter presented in [3] where the true (unknown) prior distribution is approximate using a Gaussian mixture model (GMM). Given the current computational power, it is natural to try to run Monte-Carlo simulations in parallel. However, Markov chains in general have to satisfy the so-called the "*Markovian*" property which makes the chain generation an inherently sequential problem. This restriction is mainly posed by the transition density function used to generate a proposal state given the current state of the chain. Two approaches have gained wide popularity to parallelize MCMC samplers. The parallel-chain approach proceeds by running several chains in parallel from different initial states. The main disadvantage of this approach is that the burn-in stage has to be carried out by all chains independently, which limits the efficiency gained by running the chains on different processors. Another difficulty with this approach is the aggregation of samples generated on different processors such that the combined ensemble correctly represents the mass of the target distribution. The second approach is to parallelize a single chain. The parallel chain approach turns out to be surprisingly effective in practice. Moreover, if sufficient information about the geometry of the target distribution is available, the parallel chains can be guided to sample effectively from the target distribution. The accuracy of $\mathcal{C}\ell$HMC filters to handle nonlinearity in both model dynamics, and observational mapping operator, puts it on the right direction of applicability to practical problems. The cost of serial $\mathcal{C}\ell$HMC is nearly similar to the cost of the original HMC sampling filter, however the MC-$\mathcal{C}\ell$HMC algorithm is naturally parallel. Following a Bayesian approach, $\mathcal{C}\ell$HMC algorithm can be easily modified and applied for image retrieval given noisy image and a probabilistic representation of prior knowledge [8], [16]. This can be very useful in settings where several medical snapshots are collected for the same object, e.g. a tumor, of different resolution or uncertainty levels. Mathematical regularization is amongst the most popular methods for image reconstruction

from noisy sources [7], [12], [17]. Among the regularization methods, the Tikhonov scheme is most popular due to the Gaussianity assumption about data noise, and the easiness to incorporate prior information. Despite simplicity, the performance of this approach is highly influenced by the choice of the regularization parameter. Widely used methodologies for solving the Bayesian image retrieval problem include the algorithms discussed in [9], [15]. In [9], the authors investigate statistical image reconstruction (SIR) with regularization based on the Markov random field (MRF) model. While, regularization approach is popular, it is sensitive to the choice of the regularization parameter.

The main interest here is to develop highly accurate parallel Bayesian sampling algorithms that can be efficiently used for solving large-scale inverse problems, and show that they are suitable for a wide spectrum of applications including medical image retrieval. This work develops parallel cluster sampling algorithms, and shows that a multi-chain version is embarrassingly parallel, and can be used efficiently for medical image retrieval amongst other applications. The approach discussed in this work does not require regularization, and is designed to work in both Gaussian, and non-Gaussian case, where the computational expense is minimized via parallelization. Specifically, this paper focuses on describing the complexity analysis of a specific scenario where the MC-$\mathcal{C}\ell$HMC is parallelized by running several chains in parallel to sample the posterior distribution. The algorithm proceeds by running several Markov chains in parallel such that the number of chains is specified by the the number of components in the mixture model. This paper will focus on the case where an ensemble of states, generated from an unknown prior distribution is available, and the likelihood function relating observations to target states is either a linear or a nonlinear map. The prior distribution is approximated using a Gaussian mixture distribution which parameters are approximated based on the given prior ensemble by running an expectation maximization (EM) algorithm.

Section II reviews the general iterative and Bayesian frameworks for inverse problems and image reconstruction. Section III formulates the problem, and reviews the $\mathcal{C}\ell$HMC filter formulation. Section IV discusses opportunities for parallelization of $\mathcal{C}\ell$HMC. Section V presents a detailed complexity analysis of the proposed parallel version of $\mathcal{C}\ell$HMC filter. Numerical results are presented in Section VI. Conclusions and future works are drawn in Section VII

## II. Iterative and Bayesian Image Reconstruction

As mentioned in Section I, one of the most popular iterative reconstruction algorithms is Regularization-based algorithms. For the sake of completeness, This paper reviews the Tikhonov regularization approach [14], [17], then it presents the Bayesian formulation. The Tikhonov regularization approach involves solving the following optimization problem:

$$\mathbf{x}^{\mathrm{a}} = \min_{\mathbf{x}} T(\mathbf{x}) = \|\mathcal{H}(\mathbf{x}) - \mathbf{y}\|^2_{R^{-1}} + \alpha\|\mathbf{x}\|^2_{\mathbf{C}}, \tag{1}$$

where $\alpha$ is the regularization parameter, and $\mathbf{C}$ is the regularization matrix and it can be chosen in many clever ways. Here $\mathcal{H}$ is an observation operator that maps the model space to the observation space. If the target state is directly observed then $\mathcal{H} = \mathbf{I}$, where $\mathbf{I}$ is the identity operator. The weighted norm in Equation (1) is described as follows:

$$\|\mathbf{c} - \mathbf{d}\|^2_M = (\mathbf{c} - \mathbf{d})^{\mathbf{T}}\mathbf{M}(\mathbf{c} - \mathbf{d}) \tag{2}$$

The traditional approach to regularization is the variational formulation in which equation (1) is minimized w.r.t $\mathbf{x}$. Usually, derivative-based iterative minimization algorithms are employed to solve the problem described by (1). The derivative of the objective function $T(\mathcal{H}(\mathbf{x}))$ w.r.t the parameter $]x$ is given by:

$$\nabla_{\mathbf{x}} T(\mathcal{H}\mathbf{x}) = [\partial\mathcal{H}(\mathbf{x})]^* \, \mathbf{R}^{-1}(\mathcal{H}(\mathbf{x}) - \mathbf{y}) + \alpha\mathbf{C}\mathbf{x}, \tag{3}$$

where $[\partial\mathcal{H}]^*$ is the adjoint of the derivative, e.g. the Jacobian-transpose, of the observation operator $\mathcal{H}$. In the case of a linear observation operator this is simply the transpose of the observation operator.

In the statistical approach the underlying state $\mathbf{x}$ based on a formulation constructed using Bayesian theory is inferred, where the goal is to represent the state as a random variable which distribution is of interest. Assume $\mathcal{P}^{\mathrm{b}}(\mathbf{x})$ is a prior probability density function (PDF) representing prior knowledge about the state $\mathbf{x}$. Assume also that $\mathcal{P}(\mathbf{y}|\mathbf{x})$ is the data likelihood function that describes the observational error distribution. Using Bayes' theorem, the probability of the state $\mathbf{x}$ given the collected measurements is characterized by the posterior distribution:

$$\mathcal{P}(\mathbf{x}|\mathbf{y}) \propto \mathcal{P}(\mathbf{y}|\mathbf{x})\mathcal{P}(\mathbf{x})\,. \tag{4}$$

A common practice is to assume that the prior distribution is, generally speaking, a multivariate normal (Gaussian) distribution centered around a a first-guess state, e.g. a fore-casted state $\mathbf{x}^{\mathrm{b}}$, and have a predefined covariance structure $\mathbf{C}$, i.e. $\mathbf{x} \sim \mathcal{N}(\mathbf{x}^{\mathrm{b}}, C)$.

For Gaussian priors and consequently Gaussian posteriors, the variational approach corresponds to finding the maximum a posteriori probability (MAP) estimate of the posterior PDF. The MAP is the maximizer of the posterior PDF, or equivalently, the minimizer of its negative logarithm $-\log(\mathcal{P}(\mathbf{x}|\mathbf{y}))$. Following Tikhonov regularization approach (1), and assuming Gaussian noise, the likelihood function reads:

$$\mathcal{P}(\mathbf{y}|\mathbf{x}) \propto \exp\left(-\frac{1}{2}\|\mathcal{H}(\mathbf{x}) - \mathbf{y}\|^2_{\mathbf{R}^{-1}}\right). \tag{5}$$

Without loss of generality, assuming $\mathbf{x} \sim \mathbf{N}(0, \mathbf{C})$, the prior would be on the form

$$\mathcal{P}(\mathbf{x}) \propto \exp\left(-\frac{1}{2}\|\mathbf{x}\|^2_{\mathbf{B}}\right), \tag{6}$$

where $\mathbf{B}$ is the precision matrix, i.e. the inverse of the covariance $\mathbf{B} = \mathbf{C}^{-1}$. The MAP estimator in this formulation is the minimizer of

$$-\log\mathcal{P}(\mathbf{x}|\mathbf{y}) \propto \frac{1}{2}\|\mathcal{H}(\mathbf{x}) - \mathbf{y}\|^2_{\mathbf{R}^{-1}} + \frac{1}{2}\|\mathbf{x}\|^2_{\mathbf{B}}\,. \tag{7}$$

This shows the equivalence between Tikhonov regularization approach with the Bayesian formulation in the Gaussian linear settings.

In the Bayesian approach, once the posterior is constructed, a sampling mechanism is usually employed to estimate all the desired statistics of the posterior PDF, such as the posterior mean $\mathbb{E}(\mathbf{x}|\mathbf{y})$ that can be used as a reliable estimate of the state given the data. Moreover, the generated ensemble can be used to estimate the posterior covariance that can be used as a proxy prior error covariance for future applications of the inverse problem. Sampling the posterior PDF is usually carried out following a Monte-Carlo approach. The most powerful Monte-Carlo sampling methodology is the general family Markov-Chain Monte Carlo (MCMC) samplers. Sampling high dimensional distribution however is a very expensive process, and requires parallel efficient implementation to be considered practical. As explained in the next Section, MCMC is not limited to Gaussian or linear settings, and can be very efficient if implemented in parallel.

## III. Cluster Sampling Filter

Let $\mathbf{x} \in \mathbb{R}^{\mathrm{N_{var}}}$ is a discretized approximation of the true state of the model, for example the intensities of an image pixels.

The prior distribution $\mathcal{P}^{\mathrm{b}}(\mathbf{x})$ encapsulates the knowledge about the system state before additional information is incorporated. The likelihood function $\mathcal{P}(\mathbf{y}|\mathbf{x})$ quantifies the deviation of the prediction of model observations from the collected measurements $\mathbf{y} \in \mathbb{R}^{\mathrm{N_{obs}}}$, where $\mathrm{N_{obs}} \leq \mathrm{N_{var}}$.

From Bayes' theorem, the posterior distribution $\mathcal{P}^{\mathrm{a}}(\mathbf{x})$ reads:

$$\mathcal{P}^{\mathrm{a}}(\mathbf{x}) = \mathcal{P}(\mathbf{x}|\mathbf{y}) = \frac{\mathcal{P}(\mathbf{y}|\mathbf{x})\mathcal{P}^{\mathrm{b}}(\mathbf{x})}{\mathcal{P}(\mathbf{y})} \propto \mathcal{P}(\mathbf{y}|\mathbf{x})\mathcal{P}^{\mathrm{b}}(\mathbf{x})\,, \tag{8}$$

where $\mathcal{P}^{\mathrm{b}}(\mathbf{x})$ is the prior distribution, $\mathcal{P}(\mathbf{y}|\mathbf{x})$ is the likelihood function. $\mathcal{P}(\mathbf{y})$ acts as a scaling factor and is ignored in in the MCMC context.

Assuming the prior distribution is approximated by a Gaussian Mixture distribution, the prior takes the form:

$$\begin{aligned}\mathcal{P}^{\mathrm{b}}(\mathbf{x}) &= \sum_{i=1}^{\mathrm{N_c}} \tau_i\, \mathcal{N}(\mu_i,\, \Sigma_i) \\ &= \sum_{i=1}^{\mathrm{N_c}} \tau_i\, \frac{(2\pi)^{-\frac{\mathrm{N_{var}}}{2}}}{\sqrt{|\Sigma_i|}}\, \exp\left(-\frac{1}{2}\|\mathbf{x}-\mu_i\|^2_{\Sigma_i^{-1}}\right),\end{aligned} \tag{9}$$

where the weight $\tau_i$ quantifies the probability that an ensemble member $\mathbf{x}[e]$ belongs to the $i^{th}$ component, and $(\mu_i,\, \Sigma_i)$ are the mean and the covariance matrix associated with the $i^{th}$ component of the mixture model. Here $\mathbf{x} \in \mathrm{R}^{\mathrm{N_{var}}}$, where $\mathrm{N_{var}}$ the dimension of the target state space.

Assuming the observation errors are characterized by a Gaussian distribution $\mathcal{N}(0, \mathbf{R})$, the likelihood reads:

$$\mathcal{P}(\mathbf{y}|\mathbf{x}) = \frac{(2\pi)^{-\frac{\mathrm{m}}{2}}}{\sqrt{|\mathbf{R}|}}\, \exp\left(-\frac{1}{2}\|\mathcal{H}(\mathbf{x})-\mathbf{y}\|^2_{\mathbf{R}^{-1}}\right), \tag{10}$$

where $\mathbf{R} \in \mathrm{R}^{\mathrm{m}\times\mathrm{m}}$ is the observation error covariance matrix, and $\mathcal{H} : \mathrm{R}^{\mathrm{N_{var}}} \rightarrow \mathrm{R}^{\mathrm{m}}$ is the observation operator that maps the state space to the observation space. Here $\mathbf{y} \in \mathrm{R}^{\mathrm{m}}$ is the observation vector.

From Equations (8), (9), (10), the posterior takes the form:

$$\begin{aligned}\mathcal{P}^{\mathrm{a}}(\mathbf{x}) \propto \exp\left(-\frac{1}{2}\|\mathcal{H}(\mathbf{x})-\mathbf{y}\|^2_{\mathbf{R}^{-1}}\right) \\ \sum_{i=1}^{\mathrm{N_c}} \frac{\tau_i}{\sqrt{|\Sigma_i|}}\, \exp\left(-\frac{1}{2}\|\mathbf{x}-\mu_i\|^2_{\Sigma_i^{-1}}\right)\end{aligned} \tag{11}$$

This mixture distribution may not correspond to a Gaussian mixture in general if the observation operator is a nonlinear map.

The negative-logarithm (negative-log) of the posterior distribution kernel (11) is required by the HMC sampling algorithm. Specifically, the posterior negative-log is viewed as the potential energy function in the extended Hamiltonian phase space. The posterior negative-log is given by:

$$\begin{aligned}\mathcal{J}(\mathbf{x}) = \frac{1}{2}\|\mathcal{H}(\mathbf{x})-\mathbf{y}\|^2_{\mathbf{R}^{-1}} \\ -\log\left(\sum_{i=1}^{\mathrm{N_c}} \frac{\tau_i}{\sqrt{|\Sigma_i|}}\, \exp\left(-\frac{1}{2}\|\mathbf{x}-\mu_i\|^2_{\Sigma_i^{-1}}\right)\right)\end{aligned} \tag{12}$$

The derivative of the posterior negative-log reads:

$$\begin{aligned}\nabla_{\mathbf{x}}\mathcal{J}(\mathbf{x}) &= \mathbf{H}^T\mathbf{R}^{-1}(\mathcal{H}(\mathbf{x})-\mathbf{y}) \\ &+ \frac{\sum_{i=1}^{\mathrm{N_c}} \frac{\tau_i}{\sqrt{|\Sigma_i|}} \left(\exp\left(-\frac{1}{2}\|\mathbf{x}-\mu_i\|^2_{\Sigma_i^{-1}}\right)\right) \Sigma_i^{-1}(\mathbf{x}-\mu_i)}{\sum_{i=1}^{\mathrm{N_c}} \frac{\tau_i}{\sqrt{|\Sigma_i|}}\, \exp\left(-\frac{1}{2}\|\mathbf{x}-\mu_i\|^2_{\Sigma_i^{-1}}\right)}\end{aligned} \tag{13}$$

In this work, the posterior is sampled distribution (11) following a parallel chains approach given only ensemble of states generated from the prior distribution.

## IV. Parallelization of $\mathcal{C}\ell$HMC Filter

In this section, a brief review of the MCMC sampling algorithm is presented, and discuss opportunities of running $\mathcal{C}\ell$HMC filter on parallel architecture. The parallelism of $\mathcal{C}\ell$HMC filter even if the Hamiltonian system is replaced with a Gaussian proposal kernel build around the forecast also discussed.

### A. Markov Chain Monte-Carlo (MCMC)

MCMC is a sampling scheme capable of producing ensembles from an arbitrary distribution given it's shape function, without the need for the associated scaling factor. The choice of the proposal kernel has the greatest influence on the performance of the sampler. Here, two proposal kernels are chosen; a) a Gaussian density function centered around the current state of the chain, b) Hamiltonian Monte Carlo (HMC). The standard MCMC sampler is described in Algorithm 1.

The standard MCMC algorithm1 generally suffers from random walk behavior, slow convergence to the target density, low acceptance rate, and slow space exploration. Moreover, the generated samples are highly correlated when the vanilla MCMC algorithm is used. Many of these problems can be addressed by using Hybrid Monte Carlo (HMC). HMC uses a Hamiltonian system which plays the role of the proposal density.

**Algorithm 1** MCMC algorithm to sample from $\pi(\mathbf{x})$.

**Input:** An initial state for the chain ($\mathbf{x}^0$), and the proposal kernel $q$
**Input:** An ensemble of states from the posterior distribution $\propto \pi(\mathbf{x})$
**while** No sufficient samples collected **do**
  Given current state $\mathbf{x}^k$, use $q$ to propose $x'$
  Calculate the acceptance probability (MH): $a^k = \min\left(1\,,\, \frac{\pi(x')q(x',x^k)}{\pi(x^k)q(x^k,x')}\right)$
  Sample a uniform r.v. $u^k \sim \mathcal{U}(0,1)$
  **if** $a^k > u^k$ **then**
    Accept the proposal: $x^{k+1} = x'$
  **else**
    Reject the proposal: $x^{k+1} = x^k$

### B. The multi-chain MCMC algorithm (MC-MCMC)

Although the traditional MCMC algorithm is inherently serial, there are several modifications that can be made to allow for parallelization. In the proposed approach, instead of constructing a single long Markov chain to produces $\mathrm{N_{ens}}$ samples, generate several shorter Markov chains are generated and divide the ensemble size $\mathrm{N_{ens}}$ over these chains. The constructed chains can run in parallel to sample different regions of the target distribution independently. The parallel (MC-MCMC) sampler starts by running an Expectation Maximization step to build a Gaussian Mixture Model (GMM) approximation of the prior distribution. Once the GMM is constructed on the root processor, the GMM information is broadcasted to all the working nodes. Of course, if the number of processors is exactly the same as the number of components, each node is assigned one chain. If the number of processors is less than the number of components/chains, several chains can be assigned to each processor, e.g. based on the local ensemble sizes or simply in a round-robin fashion. Caution has to be exercised to maintain a balanced load. Once all chains have generated their assigned local samples, the ensembles are gathered to the root processor and returned as output. Of course parallel output can be considered to reduce the communication overhead. The steps of the proposed MC-MCMC scheme is detailed in Algorithm 2.

By running a Markov chain starting at each component of the mixture distribution, the proposed algorithm navigates all modes of the posterior distribution, and covers all regions of high probability.

While parallelization of HMC itself can be considered for further increase in the performance, it has been avoided for clarity and to simplify the idea. The Master-Slave parallel pattern is elected to be used, where a master core sends the information required for creating the individual chains.

**Algorithm 2** MC-MCMC parallel sampling algorithm.

**Input:** Ensemble of prior states.
**Output:** An ensemble of states from the posterior Mixture distribution (11).
Run an EM algorithm (possibly in parallel) to build a GMM approximation of the prior distribution the given ensemble
**if** EM is run in parallel **then**
  GatherAll GMM information to all processors
**else**
  Broadcast GMM information to all processors
$\mathrm{N_c}$ chains are assigned to the available processors
The local ensemble size (sample size per chain) can be specified for example based on the weight of the prior weight of the corresponding component, multiplied by the likelihood of the mean of that component
Every chain is initialized to the mean of the corresponding component in the prior distribution
The parameters of the proposal kernel, e.g. covariance of the Gaussian kernel, or the mass matrix associated with HMC sampler, are set locally based on the statistics obtained from the prior ensemble under the corresponding component in the prior mixture.
After each chain collects it's assigned sample size, Gather the ensembles generated by all nodes, and possibly weight them according to the importance of each component

## V. Complexity Analysis

This section provides a detailed theoretical discussion of the computational cost of the proposed parallel algorithm.

### A. Cost of sampling a Gaussian $\mathcal{N}(\mu, \Sigma)$

A scalar normal distribution can be sampled using many accurate algorithms such as the Mersenne Twister, Box-Muller transform, Marsaglia polar method, and Ziggurat algorithm. The least expensive is Ziggurat algorithm, where a typical value produced only requires the generation of one random floating-point value and one random table index, followed by one table lookup, one multiply operation and one comparison. The cost of generating an $\mathrm{N_{var}}-$dimensional standard normal random vector $\mathbf{x} \in \mathbb{R}^{\mathrm{N_{var}}}$ is $O(\mathrm{N_{var}})$. To generate a multivariate normal (MVN) random vector *Ziggurat algorithm* from a general Gaussian distribution $\mathcal{N}(\mu, \Sigma)$ the following steps are required:

1) Factorization of the covariance matrix $\Sigma$, to generate $\Sigma^{\frac{1}{2}}$, e.g. using Cholesky decomposition
2) Draw a standard normal random vector $\mathbf{y} \in \mathcal{N}(\mathbf{0}\,\mathbf{I})$,
3) Scaling: $\mathbf{x} = \Sigma^{\frac{1}{2}}\mathbf{y} + \mu$

If the covariance matrix $\Sigma$ is diagonal, the factorization costs $O(\mathrm{N_{var}})$, while the cost of Cholesky decomposition in general is $O(\mathrm{N_{var}^3})$. If the covariance matrix $\Sigma$ is diagonal, the cost of the scaling step is $O(\mathrm{N_{var}})$, otherwise the scaling cost is $O(\mathrm{N_{var}^2})$.

### B. Cost of MCMC with a Gaussian proposal density

Assuming independent observations, evaluating the posterior PDF at a given state requires the evaluation of $\mathrm{N_c} + 1$ matrix-vector products. The cost of evaluating the posterior PDF is $O\left((\mathrm{N_c} + 1)\,\mathrm{N_{var}}\right) = O(\mathrm{N_{var}})$ if the covariance matrices of the components the GMM prior are diagonal, otherwise, the cost is $O(\mathrm{N_{var}^2})$.

*a) Cost of one MCMC step::* In the presence of a Gaussian kernel, each step of the chain construction requires the following:

1) One MVN random vector drawn from the Gaussian kernel,
2) Two function (posterior PDF) evaluations,
3) One draw from a uniform random distribution $\mathcal{U}(0,1)$,
4) One scalar comparison in the Metropolis-Hastings step.

The cost of one MCMC step can be summarized as follows:

$$\begin{cases} O(\mathrm{N}_{\mathrm{var}}^2) + O(\mathrm{N}_{\mathrm{var}}^2) + O(1) = O(\mathrm{N}_{\mathrm{var}}^2) & \text{full GMM covariances, and full Gaussian kernel} \\ O(\mathrm{N}_{\mathrm{var}}^2) + O(\mathrm{N}_{\mathrm{var}}) + O(1) = O(\mathrm{N}_{\mathrm{var}}^2) & \text{full GMM covariances, and diagonal Gaussian kernel} \\ O(\mathrm{N}_{\mathrm{var}}) + O(\mathrm{N}_{\mathrm{var}}^2) + O(1) = O(\mathrm{N}_{\mathrm{var}}^2) & \text{diagonal GMM covariances, and full Gaussian kernel} \\ O(\mathrm{N}_{\mathrm{var}}) + O(\mathrm{N}_{\mathrm{var}}) + O(1) = O(\mathrm{N}_{\mathrm{var}}) & \text{diagonal GMM covariances, and diagonal Gaussian kernel} \end{cases}$$

To find the total cost of MCMC sampling algorithm, it is needed to evaluate the total number of steps in the Markov chain. If the ensemble size is $\mathrm{N}_{\mathrm{ens}}$, it is needed to construct a chain of length $b_s + m_s \times \mathrm{N}_{\mathrm{ens}}$, where $b_s$ are burn-in steps carried out to achieve convergence to the stationary distribution, and $m_s$ are mixing steps introduced to improve the sampler mixing and reduce the correlation between selected ensembles

*b) Total cost of MCMC sampling::* The total cost of a serial MCMC with with a Gaussian proposal density reads:

$$T_s = \begin{cases} O\Big((b_s + m_s\,\mathrm{N}_{\mathrm{ens}})\,\mathrm{N}_{\mathrm{var}}\Big) & \text{diagonal prior covariances, and diagonal Gaussian kernel} \\ O\Big((b_s + m_s\,\mathrm{N}_{\mathrm{ens}})\,\mathrm{N}_{\mathrm{var}}^2\Big) & \text{otherwise} \end{cases}$$

## C. Cost of HMC

The HMC sampler requires the evaluation of the gradient of the negative-log of the target distribution.

*a) Cost of one HMC step::* In the case of HMC, each step of the chain construction requires the following:

1) One draw of a momentum $\mathbf{p} \sim \mathcal{N}(0, \mathbf{M})$ costs $O(\mathrm{N}_{\mathrm{var}})$ with diagonal $M$.
2) Forward propagation of the pair $(\mathbf{p}, \mathbf{x})$.

The cost of the second step is dominated by the cost of evaluating a Jacobian-vector product $O(\mathrm{N}_{\mathrm{var}}^2)$. With step parameters $T = m \times h$, the cost is $O(m\,\mathrm{N}_{\mathrm{var}}^2)$, where $m$ is the number of steps in the Hamiltonian trajectory. Evaluating loss of energy requires evaluating the negative-log of the posterior shape function. Again, the cost of evaluating the posterior PDF is $O\left((\mathrm{N}_{\mathrm{c}} + 1)\,\mathrm{N}_{\mathrm{var}}\right) = O(\mathrm{N}_{\mathrm{var}})$ if the covariance matrices of the components the GMM prior are diagonal, otherwise, the cost is $O(\mathrm{N}_{\mathrm{var}}^2)$.

For Hamiltonian Monte Carlo (HMC), Assuming a diagonal mass matrix $\mathbf{M}$ the cost of one step of the chain is

$$\begin{cases} O(\mathrm{N}_{\mathrm{var}}) + O(m\,\mathrm{N}_{\mathrm{var}}^2) + O(\mathrm{N}_{\mathrm{var}}) + O(1) = O(m\,\mathrm{N}_{\mathrm{var}}^2) & \text{diagonal prior covariances} \\ O(\mathrm{N}_{\mathrm{var}}) + O(m\,\mathrm{N}_{\mathrm{var}}^2) + O(\mathrm{N}_{\mathrm{var}}^2) + O(1) = O(m\,\mathrm{N}_{\mathrm{var}}^2) & \text{non-diagonal prior covariances} \end{cases} \tag{14}$$

*b) Total cost of HMC sampling::* Following the discussion above, the cost of serial HMC sampler is $T_s = O\Big((b_s + m_s\,\mathrm{N}_{\mathrm{ens}})\; m\,\mathrm{N}_{\mathrm{var}}^2\Big)$.

## D. Cost of MC-MCMC sampling

The serial complexity $T_s$ of MC-MCMC is summarized as follows:

$$T_s = \begin{cases} O\left((b_s + m_s\,\mathrm{N}_{\mathrm{ens}})\,\mathrm{N}_{\mathrm{var}}\right); & \text{diagonal or spherical covariances of GMM, and proposal density} \\ O\left((b_s + m_s\,\mathrm{N}_{\mathrm{ens}})\,\mathrm{N}_{\mathrm{var}}^2\right); & \text{otherwise} \end{cases} \Big\} \text{Gaussian proposal}$$
$$\Big\} \text{Hybrid Monte Carlo}$$

(15)

*a) Parallel cost::* The parallel complexity can be studied clearly under a simplified (ideal) assumption, where each chain is to sample $\frac{\mathrm{N}_{\mathrm{ens}}}{\mathrm{N}_{\mathrm{c}}}$ ensemble points. In this case, since there are $\mathrm{N}_{\mathrm{c}}$ chains the parallel cost (discarding the communication cost) of MC-MCMC is given by:

$$T_p = \begin{cases} O\left(\frac{\mathrm{N}_{\mathrm{c}}}{p}\left(b_s + m_s\,\frac{\mathrm{N}_{\mathrm{ens}}}{\mathrm{N}_{\mathrm{c}}}\right)\mathrm{N}_{\mathrm{var}}\right); & \text{diagonal, or spherical covariances of GMM, and proposal density} \\ O\left(\frac{\mathrm{N}_{\mathrm{c}}}{p}\left(b_s + m_s\,\frac{\mathrm{N}_{\mathrm{ens}}}{\mathrm{N}_{\mathrm{c}}}\right)\mathrm{N}_{\mathrm{var}}^2\right); & \text{otherwise} \end{cases} \Big\} \text{Gaussian proposal}$$
$$\Big\} \text{Hybrid Monte Carlo}$$

(16)

*b) Speedup::* The speedup of MC-MCMC is given by:

$$S = \frac{T_s}{T_p} = \frac{(b_s + m_s\,\mathrm{N}_{\mathrm{ens}})}{\frac{\mathrm{N}_{\mathrm{c}}}{p}\left(b_s + m_s\,\frac{\mathrm{N}_{\mathrm{ens}}}{\mathrm{N}_{\mathrm{c}}}\right)} = \frac{p\,(b_s + m_s\,\mathrm{N}_{\mathrm{ens}})}{\mathrm{N}_{\mathrm{c}}\left(b_s + m_s\,\frac{\mathrm{N}_{\mathrm{ens}}}{\mathrm{N}_{\mathrm{c}}}\right)} \tag{17}$$

*c) Parallel efficiency::* The parallel efficiency of MC-MCMC is given by:

$$E = \frac{S}{p} = \frac{(b_s + m_s\,\mathrm{N}_{\mathrm{ens}})}{\mathrm{N}_{\mathrm{c}}\left(b_s + m_s\,\frac{\mathrm{N}_{\mathrm{ens}}}{\mathrm{N}_{\mathrm{c}}}\right)} \tag{18}$$

If the burn-in stage is discard , i.e. set $b_s = 0$, the speedup, and the parallel efficiency simplify to:

$$S = \begin{cases} \mathbf{p} & ; \quad p \le \mathrm{N}_{\mathrm{c}} \\ \mathrm{N}_{\mathrm{c}} & ; \quad p > \mathrm{N}_{\mathrm{c}} \end{cases}, \text{and,} \qquad E = \frac{S}{p} = \begin{cases} 1 & ; \quad p \le \mathrm{N}_{\mathrm{c}} \\ \frac{nc}{p} & ; \quad p > \mathrm{N}_{\mathrm{c}} \end{cases}.$$

It follows that both speedup, and parallel efficiency are independent from the state space dimension $\mathrm{N}_{\mathrm{var}}$.

*d) Communication overhead::* Assuming serial GMM run on the root node, the cost of broadcasting GMM information to all nodes is the cost of broadcasting, the means, the covariance and/or precision matrices, and the weights.

Assuming a linear communication model [11], and assuming that $t_s$, and $t_w$ are the startup, and the per-word transfer times respectively, the cost of broadcasting GMM information to $p$ nodes is given by:

$$\begin{cases} \Big(t_s + t_w\left((2\mathrm{N}_{\mathrm{var}} + 1)\,\mathrm{N}_{\mathrm{c}}\right)\Big)\log(p)\,; & \text{diagonal, or spherical GMM covariances} \\ \Big(t_s + t_w\left(\left(\frac{\mathrm{N}_{\mathrm{var}}^2}{\mathrm{N}_{\mathrm{c}}} + \mathrm{N}_{\mathrm{var}} + 1\right)\mathrm{N}_{\mathrm{c}}\right)\Big)\log(p)\,; & \text{tied GMM covariances} \\ \Big(t_s + t_w\left((\mathrm{N}_{\mathrm{var}}^2 + \mathrm{N}_{\mathrm{var}} + 1)\,\mathrm{N}_{\mathrm{c}}\right)\Big)\log(p)\,; & \text{full GMM covariances} \end{cases} \tag{19}$$

After sampling in parallel, the collected ensembles are gathered on the root node, at a cost $(t_s + t_w\,(\mathrm{N}_{\mathrm{ens}} \times \mathrm{N}_{\mathrm{var}}))\log(p)$. Consequently, the total communication cost reads:

$$\begin{cases} \Big(2t_s + t_w\left[\left(\left(\frac{\mathrm{N}_{\mathrm{ens}}}{\mathrm{N}_{\mathrm{c}}} + 2\right)\mathrm{N}_{\mathrm{var}} + 1\right)\mathrm{N}_{\mathrm{c}}\right]\Big)\log(p)\,; & \text{diagonal, or spherical GMM covariances} \\ \Big(2t_s + t_w\left[\left(\left(\frac{\mathrm{N}_{\mathrm{ens}} + \mathrm{N}_{\mathrm{var}}}{\mathrm{N}_{\mathrm{c}}} + 1\right)\mathrm{N}_{\mathrm{var}} + 1\right)\mathrm{N}_{\mathrm{c}}\right]\Big)\log(p)\,; & \text{tied GMM covariances} \\ \Big(2t_s + t_w\left[\left(\frac{\mathrm{N}_{\mathrm{ens}} \times \mathrm{N}_{\mathrm{var}}}{\mathrm{N}_{\mathrm{c}}} + \mathrm{N}_{\mathrm{var}}^2 + \mathrm{N}_{\mathrm{var}} + 1\right)\mathrm{N}_{\mathrm{c}}\right]\Big)\log(p)\,; & \text{full GMM covariances} \end{cases} \tag{20}$$

*e) Total parallel cost::* It follows immediately from the discussion above, that the total parallel cost of MC-MCMC sampling algorithm simplifies to:

$$pT_p = \begin{cases} \begin{cases} \left(2t_s + t_w\left[\left(\left(\frac{\mathrm{N_{ens}}}{\mathrm{N_c}}+2\right)\mathrm{N_{var}}+1\right)\mathrm{N_c}\right]\right)p\log(p) + O\left(\mathrm{N_c}\left(b_s + m_s\frac{\mathrm{N_{ens}}}{\mathrm{N_c}}\right)\mathrm{N_{var}}\right); & \text{diagonal, or spherical covariances of GMM, and proposal density} \\ \left(2t_s + t_w\left[\left(\left(\frac{\mathrm{N_{ens}}+2\mathrm{N_{ens}}}{\mathrm{N_c}}+1\right)\mathrm{N_{var}}+1\right)\mathrm{N_c}\right]\right)p\log(p) + O\left(\mathrm{N_c}\left(b_s + m_s\frac{\mathrm{N_{ens}}}{\mathrm{N_c}}\right)\mathrm{N_{var}^2}\right); & \text{tied covariances of GMM} \\ \left(2t_s + t_w\left[\left(\frac{\mathrm{N_{ens}}\times\mathrm{N_{ens}}}{\mathrm{N_c}}+\mathrm{N_{var}^2}+\mathrm{N_{var}}+1\right)\mathrm{N_c}\right]\right)p\log(p) + O\left(\mathrm{N_c}\left(b_s + m_s\frac{\mathrm{N_{ens}}}{\mathrm{N_c}}\right)\mathrm{N_{var}^2}\right); & \text{otherwise} \end{cases} \text{Gaussian proposal} \\ O\left(\mathrm{N_c}\left(b_s + m_s\frac{\mathrm{N_{ens}}}{\mathrm{N_c}}\right)m\,\mathrm{N_{var}^2}\right) + \begin{cases} \left(2t_s + t_w\left[\left(\left(\frac{\mathrm{N_{ens}}}{\mathrm{N_c}}+2\right)\mathrm{N_{var}}+1\right)\mathrm{N_c}\right]\right)p\log(p); & \text{diagonal, or spherical GMM covariances} \\ \left(2t_s + t_w\left[\left(\left(\frac{\mathrm{N_{ens}}+2\mathrm{N_{ens}}}{\mathrm{N_c}}+1\right)\mathrm{N_{var}}+1\right)\mathrm{N_c}\right]\right)p\log(p); & \text{tied GMM covariances} \\ \left(2t_s + t_w\left[\left(\frac{\mathrm{N_{ens}}\times\mathrm{N_{ens}}}{\mathrm{N_c}}+\mathrm{N_{var}^2}+\mathrm{N_{var}}+1\right)\mathrm{N_c}\right]\right)p\log(p); & \text{full GMM covariances} \end{cases} \text{HMC} \end{cases} \tag{21}$$

*f) Total overhead::* The total overhead function ($T_o = PT_p - T_s$) of MC-MCMC reads:

$$T_o = \begin{cases} \begin{cases} \left(2t_s + t_w\left[\left(\left(\frac{\mathrm{N_{ens}}}{\mathrm{N_c}}+2\right)\mathrm{N_{var}}+1\right)\mathrm{N_c}\right]\right)p\log(p) + O\left(\left(\mathrm{N_c}-1\right)b_s\,\mathrm{N_{var}}\right); & \text{diagonal, or spherical covariances of GMM, and proposal density} \\ \left(2t_s + t_w\left[\left(\left(\frac{\mathrm{N_{ens}}+2\mathrm{N_{ens}}}{\mathrm{N_c}}+1\right)\mathrm{N_{var}}+1\right)\mathrm{N_c}\right]\right)p\log(p) + O\left(\left(\mathrm{N_c}-1\right)b_s\,\mathrm{N_{var}^2}\right); & \text{tied covariances of GMM} \\ \left(2t_s + t_w\left[\left(\frac{\mathrm{N_{ens}}\times\mathrm{N_{ens}}}{\mathrm{N_c}}+\mathrm{N_{var}^2}+\mathrm{N_{var}}+1\right)\mathrm{N_c}\right]\right)p\log(p) + O\left(\left(\mathrm{N_c}-1\right)b_s\,\mathrm{N_{var}^2}\right); & \text{otherwise} \end{cases} \text{Gaussian proposal} \\ O\left(\left(\mathrm{N_c}-1\right)b_s\,m\,\mathrm{N_{var}^2}\right) + \begin{cases} \left(2t_s + t_w\left[\left(\left(\frac{\mathrm{N_{ens}}}{\mathrm{N_c}}+2\right)\mathrm{N_{var}}+1\right)\mathrm{N_c}\right]\right)p\log(p); & \text{diagonal, or spherical GMM covariances} \\ \left(2t_s + t_w\left[\left(\left(\frac{\mathrm{N_{ens}}+2\mathrm{N_{ens}}}{\mathrm{N_c}}+1\right)\mathrm{N_{var}}+1\right)\mathrm{N_c}\right]\right)p\log(p); & \text{tied GMM covariances} \\ \left(2t_s + t_w\left[\left(\frac{\mathrm{N_{ens}}\times\mathrm{N_{ens}}}{\mathrm{N_c}}+\mathrm{N_{var}^2}+\mathrm{N_{var}}+1\right)\mathrm{N_c}\right]\right)p\log(p); & \text{full GMM covariances} \end{cases} \text{HMC} \end{cases} \tag{22}$$

*g) Isoefficiency::* Assuming the burn-in stage is discarded, i.e. set $b_s = 0$. With $\mathrm{N_{ens}} \geq 2\mathrm{N_c}$, the isoefficiency function $W(p) = \frac{E}{1-E}T_o = kT_o$ simplifies to (the dominant terms):

$$W(W,p) = \begin{cases} k\,t_w\,\mathrm{N_{ens}}\,\mathrm{N_{var}}\,p\,\log(p); & \text{diagonal, or spherical GMM covariances, and HMC, or Gaussian proposal with diagonal covariance} \\ k\,t_w\,\left(\mathrm{N_{ens}}+\mathrm{N_{var}}\right)\mathrm{N_{var}}\,p\,\log(p); & \text{tied covariances of GMM, and HMC, or Gaussian proposal} \\ k\,t_w\,\left(\mathrm{N_{ens}}+\mathrm{N_{var}}\,\mathrm{N_c}\right)\mathrm{N_{var}}\,p\,\log(p); & \text{full covariances of GMM, and HMC, or Gaussian proposal} \end{cases} \tag{23}$$

## VI. Numerical Experiments and Performance Analysis

This section presents numerical experiments to assess the complexity analysis provided in Section V. As mentioned above, the speedup, and parallel efficiency are independent of problem dimensionality. The computational cost and the performance are discussed using one dimensional examples. The GMM distribution is fitted to the ensemble using Python code borrowed from the PYTHON DA testing suite DATeS [4].

To execute the Markov chains in parallel, the MPI4PY package is used, which provides bindings of the Message Passing Interface (MPI) standard for the PYTHON programming language, allowing any PYTHON program to exploit multiple processors.

Numerical experiments are presented to assess the complexity analysis provided in Section V. Specifically, numerical results are shown for the parallel $\mathcal{C}\ell$HMC flavors discussed in this paper using one dimensional synthetic example, and a two dimensional image retrieval experiment.

### A. One-dimensional example:

Following the strategy described in [1], starting with a synthetic prior ensemble generated from a GMM with $\mathrm{N_c} = 5$. A GMM approximation of the true prior probability distribution is constructed using the EM algorithm. The model selection criterion used here is AIC. The parameters of the true GMM prior are:

$$\begin{aligned} &\{(\tau_i;\,\mu_i,\sigma_i^2)\}_{i=1,\ldots,5} = \{(0.09;\,-6.0,0.20),\\ &(0.19;\,-2.5,0.28),\,(0.09;\,0.0,0.08),\\ &(0.28;\,2,5,0.24),\,(0.15;\,6.0,0.28),\\ &(0.15;\,6.5,0.08),\,(0.03;\,7.5,0.12),\\ &(0.02;\,8.0,0.04)\,\}\,. \end{aligned} \tag{24}$$

Assume the observation errors follow Gaussian distribution with zero mean, and variance 2.2. Assuming a synthetic observation $\mathbf{y} = -1.0$, the observation likelihood function given by:

$$\mathcal{P}(\mathbf{y}|\mathbf{x}) = \frac{1}{\sqrt{2.2}\sqrt{2\pi}}\,\exp\left(-\frac{1}{2}\,\frac{(\mathbf{x}-\mathbf{y})^2}{2.2}\right). \tag{25}$$

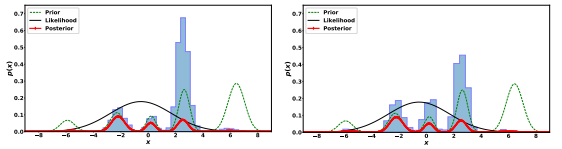
The generated GMM approximation of the prior has $\mathrm{N_c} = 7$ and the following parameters:

$$\begin{aligned} &\{(\tau_i;\,\mu_i,\sigma_i^2)\}_{i=1,\ldots,7} = \{(0.111;\,-5.78,0.123),\\ &(0.177;\,-2.49,0.223),\,(0.045;\,-1.49,0.001),\\ &(0.065;\,0.12,0.061),\,(0.146;\,2.05,0.032),\\ &(0.225;\,2.78,0.148),\,(0.231;\,6.12,0.164)\,\}\,. \end{aligned} \tag{26}$$

Figure 1 shows the results of sampling a one dimensional posterior (11) with seven components. Since this paper is mainly interested in asymptotic behavior of the sampler, and to avoid the effect of sampling error, the ensemble size here is set to 1000. Despite the good mass distribution resulting from the use of a Gaussian density with the serial MCMC sampler 1a, the acceptance rate is $\approx 45\%$ resulting in a large amount of wasted calculations. On the other hand, the parallel MC-MCMC with Gaussian proposal kernel 1b improves the acceptance rate to $\approx 81\%$. The acceptance rate is increased due to the local adjustment of the sampler hyper-parameters based on the local ensemble under the corresponding prior component in the mixture.

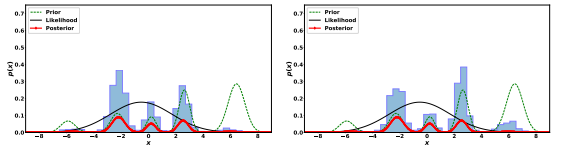
While the Gaussian-based MCMC sampler represents an acceptable mass distribution, it suffers from random walk behavior leading to the demonstrated low acceptance rate. One the other hand HMC sampling results in general in high acceptance rate. Unfortunately, as explained by the results in Figure 1c is unable to sample all probability modes. The acceptance rate in the serial $\mathcal{C}\ell$HMC sampler here is $96\%$. By running $\mathcal{C}\ell$HMC sampling methodology in parallel, the acceptance rate drops to $81\%$ (which is still very high). However, the mass distribution is much better than the serial case. This is supported by results in Figure 1d compared to Figure 1c.

Figure 2 shows the CPU time and speedup results of the clustering sampling algorithms with Gaussian and HMC proposal mechanisms. As suggested by the analysis in Section V, the CPU time becomes flat once the number of processors reaches 7, the number of components in the mixture.




(a) Gaussian Proposal, Serial MCMC

(b) Gaussian Proposal, Parallel MCMC




(c) HMC Proposal, Serial MCMC

(d) HMC Proposal, Parallel MCMC

Figure 1. Sampling results of a one dimensional posterior with seven components. The results of the serial MCMC and MC-MCMC with two proposal mechanisms are shown. The mode (serial vs. parallel) and the proposals are shown under each panel. The ensemble size here is 1000

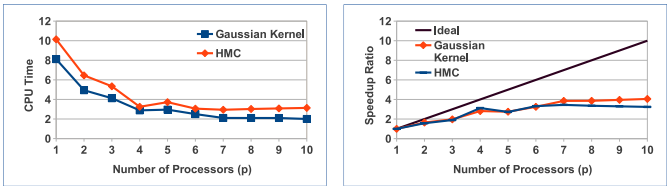



Figure 2. Sampling results of a one dimensional posterior with four components. The CPU time, and speedup results of MC-MCMC with two proposal mechanisms are plotted. The ensemble size here is 1000

The parallel efficiency results are shown in Figure 3. The numerical results shown here suggest that, running the

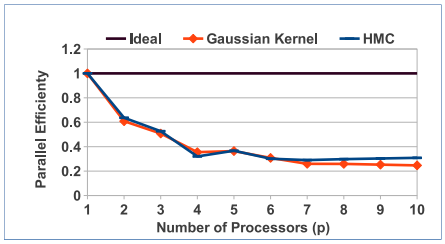



Figure 3. Sampling results of a one dimensional posterior with four components. The parallel efficiency results of MC-MCMC with two proposal mechanisms are plotted. The ensemble size here is 1000

clustering filters in parallel not only results in computational saving, but also can potentially increase the sampling accuracy. While the discussed parallelization of the sampler reduces the computational time, the numerical results suggest that more parallelization effort is needed in order to achieve higher efficiency. In the current settings, the chains are assigned to processes in a round-robin fashion. The performance of the sampler can be greatly enhanced if the a smart scheduler is used such that the parallel chains are assigned to processors based on the sample size per chain.

### B. Two-dimensional example

The parallel $\mathcal{C}\ell$HMC sampling algorithm as a tool for statistical medical image retrieval is tested here and employing a non-linear Gaussian convolution filter as a forward operator $\mathcal{H}$. The convolution filter is applied to a two-dimensional image resulting in a blurred image, then Gaussian noise is added to collect synthetic measured/observed image. Here, the vector $\mathbf{x}$ represents intensities of the image pixels arranged in a column vector. The observation noise level is set to be $9\%$ of the average intensity of the original image. This formulation clearly results in a nonlinear inverse problem that can be challenging for traditional approaches such as Tikhonov regularization. For , the Jacobian of the convoluted image with respect to the intensities of the image pixels is found to be a Toeplitz matrix.

The goal here is to retrieve the original image given the noisy measurement, and a sample drawn from the probability distribution from which the blurred image is drawn. This is is a relevant problem description in many cases, where several low resolution or blurred images are taken along with the collected measurement.

Figure 4 shows the original (true) image, the blurred image constructed by the convolution filter, and the noisy image, i.e. the blurred image with additive noise.

(a) Original Image. (b) Blurred Image. (c) Data: Blurred Image with added noise.

Figure 4. Inputs to the sampling filter. Original image $\mathbf{x}$, blurred image $\mathcal{H}(\mathbf{x})$, and noisy image $\mathbf{y}$.

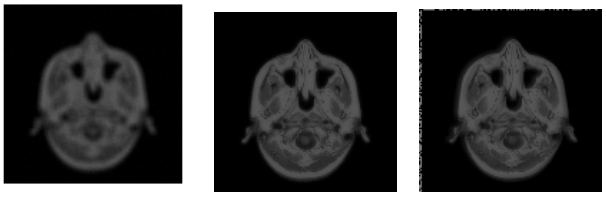
To create a synthetic non-Gaussian sample, $50$ images was sampled from a Gaussian distribution centered around the blurred image with variances equal to $8\%$ of the average intensity of the original image. To create a synthetic prior sample, $\mathrm{N_{ens}} = 30$ uniformly random distributed images have been selected from the generated Gaussian sample. This procedure is guaranteed to result in a non-Gaussian prior, and is powerful to test the GMM prior assumption.

The $\mathcal{C}\ell$HMC sampling algorithm with both Gaussian and Hamiltonian kernels performed similarly. The acceptance rates however were quite different. Specifically, the rejection rate in the case of Gaussian proposal was $56\%$ resulting in wasted computations, while HMC rejection rate was $7\%$.

The initial state of each chain is chosen as the mean of the corresponding component in the prior mixture, and the parameters of the Hamiltonian system are tuned empirically to give acceptable acceptance rate. 30 sample members are collected from the posterior distribution.

The mean and median of posterior samples collected using the parallel $\mathcal{C}\ell$HMC sampling algorithm are shown in Figure 5 (panels 5b, and 5c). For the sake of comparison to one of the

most popular and widely used approaches, the results obtained using Tikhonov regularization approach with regularization parameter are shown as they optimally tuned following an L-curve approach. The Tikhonov-regularized solution is shown in panel 5a of Figure 5.



(a) Tihonov regularized solution (b) $\mathcal{C}\ell$HMC: sample mean (c) $\mathcal{C}\ell$HMC: sample mean

Figure 5. Inverse problem solution using Tihonov regularization and the parallel $\mathcal{C}\ell$HMC posterior sample. Mean and Median, of the collected 30 sample members using parallel $\mathcal{C}\ell$HMC. The method and statistic used are shown under each panel.

By comparing results in Figure 5, to the blurred (prior) image 4b and the noisy image 4c, one can see that the posterior samples produce statistics those are closer to the original image. Moreover, the retrieved results using the non-Gaussian $\mathcal{C}\ell$HMC (serial or parallel) algorithm is much better than the solution obtained by the traditional Tikhonov regularization approach. The relative error of the mean obtained using the parallel $\mathcal{C}\ell$HMC sampling algorithm is $0.01663$, while the relative error of the regularized solution is $0.021447$. The relative error is defined as $\frac{\|\mathbf{x}-\mathbf{x}^{\text{true}}\|}{\|\mathbf{x}^{\text{true}}\|}$ , where $\mathbf{x}$ is the retrieved image, and $\mathbf{x}^{\text{true}}$ is the true image. The proposed parallel algorithms achieve an improvement of $29\%$ over the optimally tuned Tikhonov-based solution. These results show the capability and accuracy of the $\mathcal{C}\ell$HMC sampling algorithm in non-Gaussian settings.

While the blurred image has been shown in Figure 4, it is of utmost importance to highlight the fact that the formulation and the sampler is unaware of this blurred image. This is mainly due to the fact that that has uniformly sampled the Gaussian sample centered around this blurred image.

## VII. Conclusions and Future Work

This work developed parallel cluster sampling algorithms for solving Bayesian inverse problems. Specifically, a set of parallel sampling algorithms, based on the non-Gaussian cluster non-Gaussian sampling filter ($\mathcal{C}\ell$HMC), have been proposed. The presented methodologies can be efficiently used for solving various large-scale inverse problems including medical image retrieval from noisy observations. Detailed complexity analysis of the proposed algorithms (MC-$\mathcal{C}\ell$HMC) with mixture model representation of the prior information has been introduced. Generally speaking, aside from parallelization, the parallel versions of the algorithm result in higher acceptance rates. Specifically, the parallel $\mathcal{C}\ell$HMC increased the acceptance rate of the sampler from $45\%$ to $81\%$ with Gaussian proposal kernel, leading to massive saving of computations. The proposed sampling algorithms have also achieved an improvement of approximately $29\%$ over the optimally-tuned Tikhonov-based solution for image retrieval. The algorithm can run significantly faster than the serial sampler in ideal settings, where the load is equally distributed among all working nodes. In future work, we will investigate the possibility of parallelizing other components of the sampling algorithm such as the likelihood function and the proposal mechanisms. For example a parallel version of EM can be considered to construct the GMM approximation to the prior distribution. In the case of HMC, the symplectic integrator can also be parallelized. Methods for parallelizing a single chain can be considered for an additional level of parallelization.

## References

[1] Ahmed Attia, Azam Moosavi, and Adrian Sandu. Cluster sampling filters for non-gaussian data assimilation. *arXiv preprint arXiv:1607.03592*, 2016.

[2] Ahmed Attia, Vishwas Rao, and Adrian Sandu. A hybrid monte-carlo sampling smoother for four dimensional data assimilation. *International Journal for Numerical Methods in Fluids*, 2016. fld.4259.

[3] Ahmed Attia and Adrian Sandu. A hybrid Monte Carlo sampling filter for non-gaussian data assimilation. *AIMS Geosciences*, 1(geosci-01-00041):41–78, 2015.

[4] Ahmed Attia and Adrian Sandu. Dates: A highly-extensible data assimilation testing suite. *arXiv preprint arXiv:1704.05594*, 2017.

[5] Ahmed Attia, Razvan Stefanescu, and Adrian Sandu. The reduced-order hybrid monte carlo sampling smoother. *International Journal for Numerical Methods in Fluids*, 2016. fld.4255.

[6] Bing Bai and Evren Asma. *PET Image Reconstruction: Methodology and Quantitative Accuracy*. Springer International Publishing, Cham, 2017.

[7] Jinchao Feng, Shudong Jiang, Yan Zhao, Junqing Xu, Scott C Davis, Brian W Pogue, and Keith Paulsen. Direct soft prior regularization in nir spectral tomography from mri-contrast and distance-constraints, for segmentation-free reconstruction. In *Optical Tomography and Spectroscopy*, pages JM3A–4. Optical Society of America, 2016.

[8] Dorota Glowacka, Yee Whye Teh, and John Shawe-Taylor. Image retrieval with a bayesian model of relevance feedback. *arXiv preprint arXiv:1603.09522*, 2016.

[9] Tsung Hsiao, Anand Rangarajan, and Gene Gindi. Bayesian image reconstruction for transmission tomography using deterministic annealing. *Journal of Electronic Imaging*, 12(1):7–16, 2003.

[10] E. Kalnay. *Atmospheric modeling, data assimilation and predictability*. Cambridge University Press, 2002.

[11] Vipin Kumar, Ananth Grama, Anshul Gupta, and George Karypis. *Introduction to parallel computing: design and analysis of algorithms*, volume 400. Benjamin/Cummings Redwood City, CA, 1994.

[12] Xueyan Liu and Dong Peng. Regularized iterative weighted filtered back-projection for few-view data photoacoustic imaging. *Computational and Mathematical Methods in Medicine*, 2016, 2016.

[13] Yosra Marnissi, Yuling Zheng, Emilie Chouzenoux, and Jean-Christophe Pesquet. A variational bayesian approach for restoring data corrupted with non-gaussian noise. *arXiv preprint arXiv:1610.07519*, 2016.

[14] Igor Peterlik, Radovan Jirik, Nicole Ruiter, and Jirı Jan. Regularized image reconstruction for ultrasound attenuation transmission tomography. *Radioengineering*, 2008.

[15] Vassilios Stathopoulos and Joemon M Jose. Bayesian probabilistic models for image retrieval. In *WAPA*, pages 41–47, 2011.

[16] Jenni Tick, Aki Pulkkinen, and Tanja Tarvainen. Image reconstruction with uncertainty quantification in photoacoustic tomography. *The Journal of the Acoustical Society of America*, 139(4):1951–1961, 2016.

[17] Leslie Ying, Dan Xu, and Z-P Liang. On tikhonov regularization for image reconstruction in parallel mri. In *Engineering in Medicine and Biology Society, 2004. IEMBS'04. 26th Annual International Conference of the IEEE*, volume 1, pages 1056–1059. IEEE, 2004.