Sampling from High-Dimensional Probability Distributions: An Introduction to Markov Chain Monte Carlo Methods

Seminar Computational Aspects of Machine Learning

Maximilian Mozes
Faculty of Informatics
Technical University of Munich
Email: mozes@cs.tum.edu

Abstract— **Statistical learning techniques require accurate and high-quality representations of data sourced from given probability distributions in order to recognize patterns and predict future observations for a given task. However, in high-dimensional spaces the underlying distributions are often unknown or complex. In this paper, we present the basics of Markov Chain Monte Carlo (MCMC) methods, a class of algorithms used to draw samples from high-dimensional and complex probability distributions. We provide an overview of the required mathematical background from a probabilistic perspective and describe the theoretical concepts of MCMC. We strengthen our fundamental presentation by illustrating two commonly-known sampling algorithms based on MCMC, the Metropolis-Hastings algorithm and the Gibbs sampler. Finally, we focus on the computational concerns associated with both algorithms and provide an overview about recent modifications that led to more efficient and accurate realizations.**

Index Terms— **bayesian inference, statistical sampling, markov chains, monte carlo methods, gibbs sampling, metropolis-hastings algorithm**

I. Introduction

The demand for highly-qualitative observations drawn from a given probability distribution is of particular significance in contemporary research scenarios associated with artificial intelligence and machine learning. Nevertheless, mathematical representations of such phenomena often result in complex and partially unknown models that present a substantial constraint to the evaluative approaches applied on a given amount of data. Technological advancements in recent years provide the computational foundation for conducting such evaluations on complex bodies of evidence. However, even with a sufficient amount of computing power it is sometimes impossible to derive or integrate probabilistic models in an analytical way. Markov Chain Monte Carlo (MCMC) methods provide a combination of statistical and probabilistic models aiming at providing reliable tools for handling high-dimensional and complex probability distributions in an efficient yet accurate way.

The term MCMC firstly came up in the 1950s, when a team around the physicist Nicholas Metropolis published a method to numerically compute state equations of a molecular system in the context of mechanical statistics. The developed method became known as the *Metropolis algorithm* [16]. Later, in 1970, Wilfried Hastings published a more generalized version of the algorithm and it is since then referred to as the *Metropolis-Hastings algorithm* [11]. The other popular sampling method that we will discuss in the remainder of this paper, *Gibbs sampling*, is considered a special case of the Metropolis-Hastings algorithm in that it is based on a similar MCMC approach. The detailed description and differentiation between both algorithms will be discussed in section IV.

MCMC algorithms are used in a variety of contemporary research scenarios including natural and life sciences such as solar physics (e.g. [6]) and evolutionary biology (e.g. [13]) as well as other fields like the computational social sciences (e.g. [19]).

In this work, we study basic representations of MCMC algorithms used for sampling from complex probability distributions. In doing so, we explain the necessary mathematical background by considering the principles of statistical Monte Carlo methods as introduced by Stanislaw Ulam and John von Neumann [18] and the basic properties of probabilistic Markov chains in section II. Afterwards, we combine both methods and derive the concepts of sampling from a given probability distribution with MCMC methods by describing and examining the Metropolis-Hastings algorithm in section III and the Gibbs sampler in section IV. Finally, we focus on the computational perspectives of both techniques in section V and describe optimizations that have been suggested to enhance the computational performance and the efficiency of the discussed algorithms.

II. The MCMC Approximation

When dealing with fairly complex and unwieldy probability distributions, computations such as approximating the distribution's expected value become difficult and time-consuming tasks. MCMC provides a methodology that aims at simplifying such approaches by employing statistical *Monte Carlo methods* that are based on a particular class of stochastic processes, the so-called *Markov chains*. In the following, we provide a theoretical overview of both techniques in order to establish a solid foundation of the basic principles associated with MCMC.

A. Monte Carlo methods

The term "Monte Carlo" was introduced by physicist Nicholas Metropolis after Stanislaw Ulam and John von Neumann firstly published the basic idea behind this class of randomized algorithms in the 1940s [18]. Monte Carlo approximation methods serve the purpose of numerically modelling a quantity of interest using a given set of samples in a computationally efficient way. This is especially helpful in the case of highly complex probability distributions that are difficult to handle in an analytical way.

Given N independent and identically distributed (*i.i.d.*) samples $\mathbf{x}_i \sim p(\mathbf{x}\,|\,\mathcal{D})$ from a posterior distribution over a given dataset $\mathcal{D}$, the method aims to approximate a desired quantity of interest ϕ by considering the sample mean, i.e.

$$\lambda = \frac{1}{N}\sum_{i=1}^{N}\phi(\mathbf{x}_i)$$

such that $\lambda \approx \mathbb{E}[\phi(\mathbf{x})\,|\,\mathcal{D}]$ [9]. Such a quantity of interest can be, for example, the marginal posterior

$$p(u\,|\,w) = \int_{\mathcal{V}} p(u,v\,|\,w)\,\mathrm{d}v$$

for an element $(u,v) \in \mathcal{U}\times\mathcal{V}$ of a joint posterior distribution [1]. The justification of this approach is supported by two central properties. First, the individual samples are *i.i.d.* and hence the estimator λ is unbiased such that

$$\mathbb{E}[\lambda] = \frac{1}{N}\sum_{i=1}^{N}\mathbb{E}[\phi(\mathbf{x}_i)\,|\,\mathcal{D}] = \mathbb{E}[\phi(\mathbf{x})\,|\,\mathcal{D}].$$

Second, the *strong law of large numbers* posits that for a given sample with independent and identically distributed observations, the arithmetic mean of our observations converges to the expected value [9]. This theorem, however, is conditioned by the assumption that the variance of our estimator is bounded. But in our case, this characteristic is given as long as the variance of our individual observations is bounded, since

$$\mathrm{Var}(\lambda) = \frac{1}{N^2}\sum_{i=1}^{N}\mathrm{Var}(\phi(\mathbf{x}_i)\,|\,\mathcal{D}) = \frac{1}{N}\mathrm{Var}(\phi(\mathbf{x})\,|\,\mathcal{D}).$$

Nevertheless, in many applications of Monte Carlo methods it is the process of sampling from a given probability distribution itself that represents the most challenging problem. As we will see throughout this paper, MCMC provides a technique for sampling from a given distribution in an efficient yet accurate way by incorporating the probabilistic structures and characteristics entailed in a particular class of stochastic processes called *Markov chains*.

B. Markov chains

Markov chains are statistical tools modelling a stochastic process. In probability theory, a stochastic process is a sequence of random variables $\{X_i\}_{i\geq 0}$ given a fixed number of parameters [17]. Each random variable X_i resides in a given *state space* $E \subseteq \mathbb{R}$ such that $X_i : \Omega \to E$ for a given sample space Ω. The *Markov property* for a given stochastic process says that a random realization of the given state space only depends on the previously drawn random variable such that

$$P(X_i = x_i\,|\,X_0 = x_0, ..., X_{i-1} = x_{i-1})$$
$$= P(X_i = x_i\,|\,X_{i-1} = x_{i-1})$$

holds for all realizations of random variables $X_i = x_i$ in the given state space. A stochastic process that satisfies the Markov property is called *Markov process*. If, in addition, the stochastic process is defined on a countable state space E, we call the process Markov chain [5].

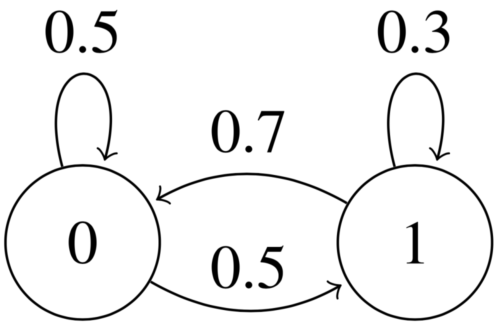

Fig. 1: Illustration of a transition diagram represented by a transition matrix $\mathbf{P} \in \mathbb{R}^{2\times 2}$ modelling a two-state Markov chain.

The transitions between different states of a Markov chain are modelled using a *transition matrix* $\mathbf{P} = \{p_{ij}\}_{i,j\in E}$ with

$$p_{ij} = P(X_n = j\,|\,X_{n-1} = i), \tag{1}$$

where $p_{ij} \geq 0$ and

$$\sum_{j\in E} p_{ij} = 1 \tag{2}$$

for each row i, since the accumulation of probabilities for all transitions from a given state must be equal to 1. A matrix that fulfils property (2) is referred to as *stochastic matrix* [17]. Figure 1 shows an example for a transition graph modelling a Markov chain that is described by

$$\mathbf{P} = \begin{pmatrix} 0.5 & 0.5 \\ 0.7 & 0.3 \end{pmatrix}.$$

The initial distribution $\pi_0^\top \in \mathbb{R}^n$ of a Markov chain (where n represents the number of given states) is an n-dimensional vector denoting the probabilities for each of the chain's states to be the starting point of a process. However, each time step t can be modelled by such a distribution, meaning that $\pi_t^\top$ represents a probability distribution over the Markov chain in the t-th time step. This distribution denotes the probability of each of the chain's states to be the current state of the stochastic process at time step t. We call this distribution *stationary*, if

$$\pi = \pi \cdot \mathbf{P}. \tag{3}$$

This equation implies that a stationary distribution of a Markov chain models the long-term probabilities for the stochastic process to reside in each of the states. In the first state change of a stochastic process, for example, our current probability distribution would be $\pi_1 = \pi_0 \cdot \mathbf{P}$. If our stochastic process possesses a probability distribution such that equation (3) is satisfied, the corresponding Markov chain is called *stationary* [5].

Markov chains are eligible to exhibit a set of different properties, three of which are of particular interest for the topics discussed in this work, i.e. *irreducibility*, *aperiodicity* and *ergodicity*. Irreducibility describes the property that every state of the chain can be reached from any other state. In order to properly define the term aperiodicity, we need to define the period of a state i. The period $d(i)$ of a Markov chain's state i is the highest number $\gamma \in \mathbb{N}$ such that

$$\{n \in \mathbb{N}_0 \,|\, p_{jj}^{(n)} > 0\} \subseteq \{n \cdot \gamma \,|\, n \in \mathbb{N}_0\}. \tag{4}$$

A state i with $\gamma = d(i) = 1$ is called aperiodic [17]. Here, $p_{jj}^{(n)}$ denotes the probability of reaching state j from itself in n state transitions. If $d(i) = 1$ for all states i of a Markov chain, we call the Markov chain aperiodic. A Markov chain that is both irreducible and aperiodic is referred to as an ergodic Markov chain.

Moreover, every ergodic Markov chain has a unique stationary distribution π with $\pi_t \to \pi$ as $t \to \infty$. Given these characteristics of a Markov chain, we can derive a theorem that is of substantial importance for the application of Markov chains in MCMC techniques.

The ergodic theorem. Let $\{X_i\}_{i\geq 0}$ be an ergodic Markov chain with state space E and let π be its corresponding stationary distribution. Furthermore, let $\boldsymbol{\psi} : E \to \mathbb{R}$ be a function such that

$$\mathbb{E}_\pi[\boldsymbol{\psi}(X)] = \sum_{i \in E} \boldsymbol{\psi}(X_i)\pi(X_i) < \infty.$$

Then

$$\lim_{M\to\infty} \frac{1}{M} \sum_{k=1}^{M} \boldsymbol{\psi}\left(X_k\right) = \mathbb{E}_\pi[\boldsymbol{\psi}(X)] \tag{5}$$

with probability 1, where M denotes the amount of given state realizations $\{X_i\}_{i\geq 0}$ at a given time step t. In other words, the arithmetic sample mean of random state variables modelling an ergodic Markov chain converges to the expected value of the chain's stationary distribution. This allows us to treat the individual X_i as independent measures although they exhibit, by definition, a certain dependency relation and are not *i.i.d.*. The particular relevance of this theorem for MCMC will be discussed in the following section.

C. The MCMC technique

Based on both techniques described above we can now conclude and explain the functionality of Markov Chain Monte Carlo methods by combining these statistical approaches for sampling from a given probability distribution.

With MCMC we aim to find a stationary distribution π of a given Markov chain that approximates our desired posterior $p(\mathbf{x} \,|\, \mathcal{D})$. We can then execute this chain in order to draw samples from a complex probability distribution. Using these given samples drawn from the Markov chain we can then use Monte Carlo methods to efficiently approximate our desired quantities.

The combination of these two statistical techniques is embedded in two popular algorithms used to draw samples from a given probability distribution. The first one is referred to as the *Metropolis algorithm*, published by Nicholas Metropolis, Arianna and Marshall Rosenbluth and Augusta and Edward Teller in 1953 [16]. Their paper *Equation of State Calculations by Fast Computing Machines* is publicly known as the first publication describing an MCMC algorithm [18]. The second one is called *Gibbs sampling* and it was firstly introduced by Stuart and Donald Geman in 1984 [7]. In the following, we describe the functionality of both algorithms in the context of MCMC.

III. The Metropolis-Hastings Algorithm

The need for an algorithm capable of obtaining a sequence of random samples from a given probability distribution was firstly made public in the early 1950s. Metropolis et al. (1953) published the initial version of the *Metropolis algorithm* in the context of statistical physics, using the algorithm to generate random configurations of molecular systems [16].

Algorithm 1: Pseudocode for the basic Metropolis algorithm based on its implementation by [1].

```
Select a starting point $\mathbf{x}^{(0)} \in \mathcal{X}^N$
for $i = 0, 1, 2, ..., M-1$ do
    Sample $\mathbf{x}' \sim p(\mathbf{x}' \,|\, \mathbf{x}^{(i)})$
    Sample $u \sim \mathcal{U}(0, 1)$
    if $u < A(\mathbf{x}^{(i)}, \mathbf{x}')$ then
        $\mathbf{x}^{(i+1)} \leftarrow \mathbf{x}'$
    else
        $\mathbf{x}^{(i+1)} \leftarrow \mathbf{x}^{(i)}$
    end
    $V \leftarrow V \cup \{\mathbf{x}^{(i+1)}\}$
end
```

Although their method was applied to such a specific problem, they aimed to describe the technique in a general and universal fashion. Declared one of the ten most influential algorithms for the development of science and engineering in the 20th century [3], cited in [1], the technique is of substantial importance for contemporary statistical and probabilistic learning tasks in industry and research. In its most basic implementation, the algorithm works as follows.

The Metropolis algorithm. Let $\mathcal{X}^N$ be a state space and $q(\mathbf{x})$ be the probability mass or density function of a given N-dimensional probability distribution on $\mathcal{X}^N$ from which we want to generate samples. Algorithm 1 describes the procedure for drawing n samples from a distribution $\pi(\mathbf{x})$ that is proportional to $q(\mathbf{x})$. $\pi(\mathbf{x})$ is called *target distribution*. The drawn samples are successively being added to the set of generated samples V. Here, $p(\mathbf{x}' \,|\, \mathbf{x}^{(i)})$ describes a *proposal* *distribution* that is dependent on the current sample value $\mathbf{x}^{(i)}$ and $M = |V|$ denotes the number of individual samples the algorithm generates. This proposal distribution is also referred to as *random walk proposal* [1] and it models the underlying stochastic process that is described by a Markov chain such that $p(\mathbf{x}' \,|\, \mathbf{x}^{(i)})$ denotes the transition from state $\mathbf{x}^{(i)}$ to state $\mathbf{x}'$. In the terms introduced in section II-B, this proposal distribution embodies the generalisation of p_{ij} (see equation (1)). This implies that the algorithm models a proposal distribution over time using a Markov chain, and eventually the stationary distribution of our Markov chain models the probability distribution $\pi(\mathbf{x})$ [11]. The ergodic theorem provides evidence that in this case the chain approximates our probability distribution as the number of iterations (which equals the number of state realizations of the Markov chain) approaches infinity ($M \to \infty$, see equation (5)). Hastings (1970) suggests a general technique to construct an appropriate transition matrix with $\pi(\mathbf{x})$ as its stationary distribution [11]. However, in the remainder of this paper we will not discuss this approach in detail.

In its most basic implementation, the algorithm requires the proposal distribution to be symmetric, i.e. $p(\mathbf{x}' \,|\, \mathbf{x}^{(i)}) = p(\mathbf{x}^{(i)} \,|\, \mathbf{x}')$. The corresponding *acceptance ratio* is then given as

$$A(\mathbf{x}', \mathbf{x}^{(i)}) := \min\left\{1, \frac{\pi(\mathbf{x}')}{\pi(\mathbf{x}^{(i)})}\right\}.$$

The intuition behind this acceptance ratio is that if the probability of our proposed value $\mathbf{x}'$ is higher than that of $\mathbf{x}^{(i)}$, the corresponding acceptance ratio takes its highest possible value, i.e. 1. Otherwise, the ratio takes a value in [0,1). As described in the algorithm, we use a random sample value u drawn from a uniform distribution $\mathcal{U}(0,1)$ and evaluate it against $A(\mathbf{x}', \mathbf{x}^{(i)})$. Hence, the Metropolis algorithm does not reject proposed sample values $\mathbf{x}'$ whose probability is higher than that of the previously drawn sample value $\mathbf{x}^{(i)}$ (as in this case $A(\mathbf{x}', \mathbf{x}^{(i)}) \geq u$). But if $A(\mathbf{x}', \mathbf{x}^{(i)}) < 1$, then the algorithm randomly decides whether or not to accept the proposed value $\mathbf{x}'$.

The symmetry constraint of our proposal distribution, however, is not required in the more general case. In 1970, Wilfried Hastings proposed a more general realization of the Metropolis algorithm by adjusting the acceptance ratio such that the proposal distribution $p(\mathbf{x}' \,|\, \mathbf{x}^{(i)})$ is not required to be symmetric anymore [11]. Thus,

$$A(\mathbf{x}', \mathbf{x}^{(i)}) := \min\left\{1, \frac{\pi(\mathbf{x}')p(\mathbf{x}^{(i)} \,|\, \mathbf{x}')}{\pi(\mathbf{x}^{(i)})p(\mathbf{x}' \,|\, \mathbf{x}^{(i)})}\right\}.$$

This adjustment is known as the *Hastings correction* [17] and one can clearly see the difference between the original and the more recent approach. If the proposal distribution is symmetric, however, the Hastings correction degenerates to the original acceptance ratio. But if this is not the case, it takes into account the ratio $p(\mathbf{x}^{(i)} \,|\, \mathbf{x}')/p(\mathbf{x}' \,|\, \mathbf{x}^{(i)})$ representing the relative difference between the probabilities of the previous and the current sample values conditioned on each other.

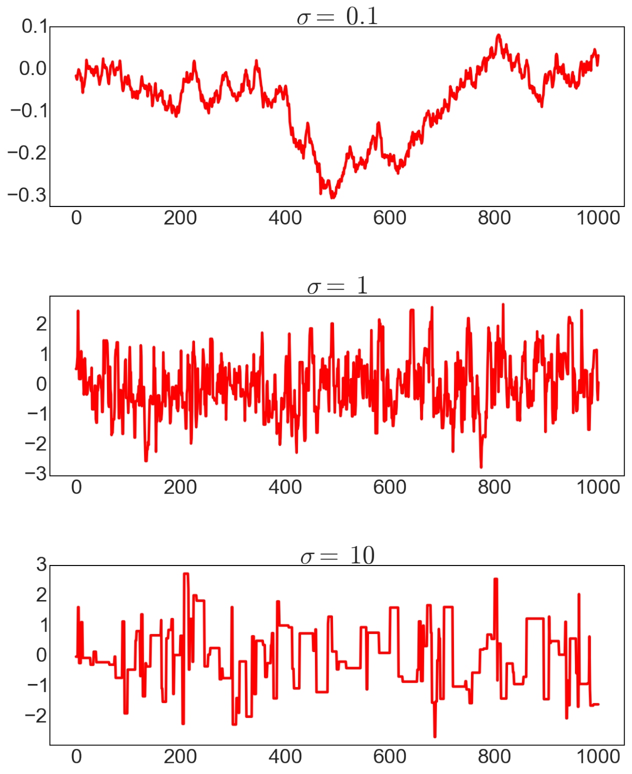

Fig. 2: Sampling output of a random walk Metropolis-Hastings algorithm with $n = 1000$ iterations for different variances σ^2. The target distribution is $\pi(x) \sim \mathcal{N}(0,1)$. The x-axis denotes the number of iterations, and the y-axis represents the sample values.

Hastings' contributions to the Metropolis algorithm and its corresponding generalization led to the **Metropolis-Hastings (MH) algorithm**, which is a common term of the described algorithm in the contemporary literature.

$A(\mathbf{x}', \mathbf{x}^{(i)})$ clearly illustrates that it suffices for the algorithm to sample from $q(\mathbf{x})$ using a proportional target distribution $\pi(\mathbf{x})$. The acceptance ratio only takes into consideration the ratio between different values of $\pi(\mathbf{x})$. But as we know that $\pi(\mathbf{x}) \propto q(\mathbf{x})$, we can write $q(\mathbf{x}) = (1/z) \cdot \pi(\mathbf{x})$ for $z \in \mathbb{R}$. The factor z, however, cancels out when considering the acceptance ratio $A(\mathbf{x}', \mathbf{x}^{(i)})$ with respect to $\pi(\mathbf{x})$ such that it equals the ratio with respect to $q(\mathbf{x})$.

The performance of the MH algorithm is strongly dependent on the choice of the proposal distribution. Wide distributions, for example, result in high rejection rates of the algorithm which again leads to high correlations. If, in contrast, the distribution is too narrow, it might appear that only one mode of $\pi(\mathbf{x})$ is considered [1]. A common proposal distribution to choose for the MH algorithm is the Gaussian distribution centered at the previous sample value $\mathbf{x}^{(i)}$ such that $p(\mathbf{x}' \,|\, \mathbf{x}^{(i)}) \sim \mathcal{N}(\mathbf{x}' \,|\, \mathbf{x}^{(i)}, \mathbf{\Sigma})$ for a given covariance matrix $\mathbf{\Sigma}$. When utilizing this proposal distribution the method is referred to as the *random walk* MH algorithm [17]. The algorithm's performance then strongly depends on σ. Figure 2 shows the performance of a random walk using different variances. The proposal distribution is $\mathcal{N}(x^{(i)}, \sigma^2)$. For small values of σ (i.e. $\sigma = 0.1$), one can see that the samples lie in a very limited range, such that every drawn sample x satisfies $-0.3 < x < 0.1$. On the other hand, if we choose large values for the standard deviation (here $\sigma = 10$) one can see that the algorithm accepts relatively few proposals. In this particular experiment, the acceptance rate is $134/1000 = 0.134$, whereas for $\sigma = 1$ the acceptance rate is 0.702 (and 0.975 for $\sigma = 0.1$, respectively).

IV. Gibbs Sampling

Gibbs sampling can be considered as a similar approach to the previously discussed Metropolis-Hastings algorithm as it also employs stochastic processes and is reliant on the discussed ergodic theorem to successfully draw samples from a given distribution [1]. In its basic implementation, the algorithm aims to draw samples from a given joint probability distribution $\pi(\mathbf{x}_1, ..., \mathbf{x}_M)$ by sampling from the conditionals of some distribution $q(\mathbf{x}_1, ..., \mathbf{x}_M)$, i.e.

$$q\left(\mathbf{x}_j^{(t)} \,|\, \mathbf{x}_1^{(t)}, \mathbf{x}_2^{(t)}, ..., \mathbf{x}_{j-1}^{(t)}, \mathbf{x}_{j+1}^{(t)}, ..., \mathbf{x}_M^{(t)}\right),$$

where $\mathbf{x}_j \in \mathcal{X}^N$, $j \in \{1, ..., M\}$ and t denotes the t-th iteration of our sampling process [4]. Similarly to the MH algorithm, Gibbs sampling utilizes an underlying Markov chain to model the approximation process to the target distribution. In detail, the Markov chain's stationary distribution represents the joint target distribution.

Algorithm 2: Illustration of the Gibbs sampling algorithm as described in [10].

```
1 Initialize starting values $\mathbf{x}_j^{(0)}$
2 for $i = 1, 2, ..., K$ do
3     for $j = 1, 2, ..., M$ do
4         $\mathbf{x}_j^{(i)} \sim q\left(\mathbf{x}_j^{(i)} \,|\, \mathbf{x}_1^{(i)}, ..., \mathbf{x}_{j-1}^{(i)}, \mathbf{x}_{j+1}^{(i-1)}, ..., \mathbf{x}_M^{(i-1)}\right)$
5     end
6 end
```

Algorithm 2 illustrates that in each iteration i, the method samples from the conditional probability distribution of the previous and the current iteration. That means, if we consider the first variable $\mathbf{x}_1^{(i)}$, we base the sampling solely on the sampled values $\mathbf{x}_j^{(i-1)}$ with $j \in \{2, ..., M\}$ of the previous iteration. For $j > 1$, however, we take into consideration all sample values $\mathbf{x}_k^{(i)}$ with $k < j$ of the current iteration and all previously drawn sample values $\mathbf{x}_l^{(i-1)}$ with $l \in \{j+1, ..., M\}$ of iteration $(i-1)$.

This procedure is continued until the drawn samples $\mathbf{x}_1^{(i)}, \mathbf{x}_2^{(i)}, ..., \mathbf{x}_M^{(i)}$ converge to certain values and the process stabilizes [10]. Here, the number of iterations K depends on the pace with which the process stabilizes and is therefore unknown before execution. The algorithm models a Markov chain that continues to draw new random variables from a given state space $\mathcal{X}^N$ until it converges to the target distribution $\pi(\mathbf{x}_1, ..., \mathbf{x}_M)$ of our given probability model. In other words, the Markov chain converges to its stationary distribution π, which equals our desired distribution. This

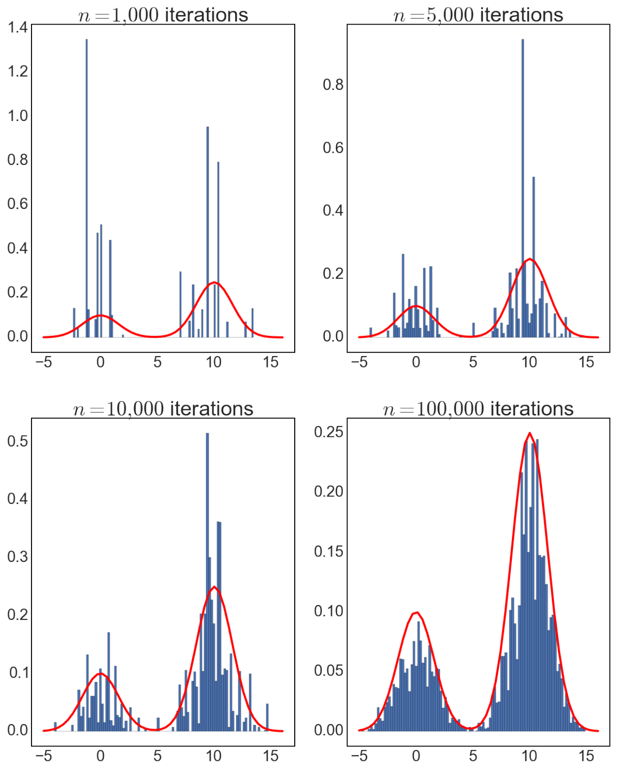

Fig. 3: Sample distribution of a random walk Metropolis-Hastings run with proposal distribution $p(x' \,|\, x^{(i)}) \sim \mathcal{N}(x^{(i)}, 200^2)$ for different iterations n. The target distribution (red line) is $\pi(x) \propto \frac{1}{10}\exp\left(-\frac{1}{5}x^2\right) + \frac{1}{4}\exp\left(-\frac{1}{5}(x-10)^2\right)$. The x-axis represents the values sampled from $\pi(x)$ and the y-axis denotes their corresponding normalized frequencies in the sample set.

guarantees that our final set of random variables is a sample from $\pi(\mathbf{x}_1, ..., \mathbf{x}_M)$ [10].

This approach has two requirements for $q(\mathbf{x}_1, ..., \mathbf{x}_M)$ that need to be satisfied for the algorithm to successfully generate the samples. First, $q(\mathbf{x}_1, ..., \mathbf{x}_M) > 0$ needs to be satisfied for the Markov chain to converge. Second, $\pi(\mathbf{x}_1, ..., \mathbf{x}_M)$ has to be the equilibrium of $q(\mathbf{x}_1, ..., \mathbf{x}_M)$, meaning that we always converge to $\pi(\mathbf{x}_1, ..., \mathbf{x}_M)$, independent of the initial random variables. Apart from that there are no requirements given for the algorithm to succeed such that any distribution $q(\mathbf{x}_1, ..., \mathbf{x}_M)$ that satisfies the above properties can be employed [2].

V. Computational Concerns and Optimizations

The descriptions of the Metropolis-Hastings algorithm and the Gibbs sampler clearly illustrate that both algorithms come at a certain computational cost. However, previous research conducted on the optimization of both methods suggests promising optimization procedures that aim at reducing the computational costs whilst maintaining the degree of performance accuracy of each algorithm. In this section, we focus on individual optimization procedures for the Metropolis-Hastings algorithm and the Gibbs sampler that address the above concerns.

A. *Burn-in*

In the context of MCMC the term *burn-in* describes an optimization approach for increasing the quality of sampled data points. As we have seen in section II, MCMC methods start with an arbitrary sample and converge to the Markov chain's stationary distribution over time. However, the quality of our sampled elements increases with more iterations as the chain approximates its stationary distribution. Burn-in describes a concept to ignore the first n samples of an MCMC method in order to increase the overall quality of sampled data points [8]. This approach obviously comes with an additional computational cost as the sampling algorithm requires a greater number of iterations. Furthermore, it is hard to evaluate the exact amount of iterations n after which the MCMC algorithm has burned-in. In fact, this problem represents one of the central disadvantages associated with MCMC [17].

B. *Runtime and accuracy*

It makes intuitive sense that the accuracy of an MCMC method strongly depends on the amount of samples drawn from the posterior distribution. Figure 3 indicates that a larger amount of iterations results in a more accurate approximation of the desired posterior distribution. When computing a quantity of interest based on the given sample data it is important to consider the necessary number of drawn elements for a reasonable approximation. On the other hand, although a large amount of samples might result in an accurate approximation of the desired probability distribution, huge sample sets in turn might drastically increase the computational costs associated with the corresponding numerical computation of a desired quantity.

C. *Collapsed Gibbs sampling*

A widely-known optimization approach for the Gibbs sampling algorithm is referred to as *collapsed Gibbs sampling*. Unlike the general approach, this approach aims at integrating out irrelevant parameter variables of the given conditional probability distributions in order to reduce the computational cost of an iteration. Suppose, for example, we want to draw samples from a conditional distribution $q(A \mid B, C)$. If the representation of our given distribution allows to sample A directly from $q(A \mid B)$ and B directly from $q(B \mid A)$, then we can integrate out C in our model and continue with a sparser probability distribution. Then, after convergence of our model, we can draw C from our actual target distribution $\pi(C \mid A, B)$ [15]. This approach is also referred to as *Rao-Blackwellisation* and is based on the following theorem.

Theorem (Rao-Blackwell): For two dependent n-dimensional random variables Γ and Ω,

$$\mathrm{Var}_{\Gamma,\Omega}(\phi(\Gamma,\Omega)) \geq \mathrm{Var}_{\Gamma}(\mathbb{E}_{\Omega}[\phi(\Gamma,\Omega \mid \Gamma)]) \qquad (6)$$

holds with a given scalar function $\phi : \Gamma \times \Omega \to \mathbb{R}$. In other words, when integrating out a parameter Ω, the resulting variance of the model's estimate will never be higher than the estimate's original variance. This theorem therefore guarantees increased efficiency with a collapsed Gibbs sampling method over the general approach [17].

D. *Blocking Gibbs sampling*

As discussed in section IV, the Gibbs sampler draws a sample from one single variable at a time (this is called *single site updating* (e.g. [17])). Contrary to this approach, *blocking Gibbs sampling* creates blocks of random variables and samples from these blocks in a simultaneous fashion [14]. This method therefore represents another approach of reducing the computational complexity of the algorithm by tackling the increased costs of single site updating. The general concept is as follows. Let

$$\mathcal{V} := \{\mathbf{v}_j \mid j \in \{1, ..., n\}\}$$

be the set of all variables of the given probability distribution. The algorithm divides $\mathcal{V}$ into k subsets E_i with $i \in \{1, ..., k\}$ such that $\bigcup_{i=1}^{k} E_i = \mathcal{V}$. These subsets do not necessarily have to be disjoint. Furthermore, we define $A_i := \mathcal{V} \backslash E_i$ as the complementary set for the i-th subset. The blocking Gibbs sampler then draws samples from all subsets E_i conditioned on their corresponding configuration A_i. Once the algorithm visited all subsets E_i, one iteration of the sampler has been completed. The fact that the union of all subsets E_i equals the total set of variables $\mathcal{V}$ implies that for each variable $\mathbf{v}_j$ at least one sample has been drawn from the given distribution. It might, however, be the case that for some variables multiple samples were drawn (as the subsets do not have to be disjoint).

The complexity of this approach substantially depends on the size and the amount k of the subsets A_i. The convergence rate increases with a larger size of the subsets as this leads to a maximum overlap of variables $\mathbf{v}_j$, thus causing each variable to be sampled more often during an individual iteration. This desire contrasts with the fact that an increasing size of the subsets results in a higher complexity. Jensen and Kjaerulff (1995) address this problem by presenting an approach for identifying optimal subset sizes using a method based on junction tree algorithms. However, in the course of this paper we do not discuss this approach in further detail and refer to their work for a detailed explanation [14].

VI. Conclusion

This work presented an overview of MCMC, a widely-known set of algorithms that are utilized for sampling problems associated with high-dimensional probability distributions. We described the basic theoretical concepts associated with these methods by deriving MCMC from statistical Monte Carlo methods and stochastic processes represented by Markov chains. We then illustrated the concepts of MCMC in a more detailed fashion by providing two commonly-known examples, the Metropolis-Hastings algorithm and the Gibbs sampler. Finally, we focused on the computational concerns associated with this set of sampling methods and presented existing approaches aiming at reducing the computational costs and complexity of these algorithms whilst maintaining their degree of probabilistic accuracy.

References

[1] C. Andrieu, N. de Freitas, A. Doucet and M. I. Jordan, 2003. *An Introduction to MCMC for Machine Learning*. Machine Learning, Vol. 50, 5-43.

[2] D. Barber. *Bayesian Reasoning and Machine Learning*. Cambridge University Press. 2012.

[3] I. Beichl and F. Sullivan, 2000. *The Metropolis algorithm.* Computing in Science and Engineering, Vol. 2, No. 1, 65-69.

[4] C. M. Bishop. *Pattern Recognition and Machine Learning.* Springer. 2006.

[5] P. Bremaud. *Markov Chains - Gibbs Fields, Monte Carlo Simulation, and Queues*. Springer. 1999.

[6] P. B. Demorest, T. Pennucci, S. M. Ransom, M. S. E. Roberts and J. W. T. Hessels, 2010. *A two-solar-mass neutron star measured using Shapiro delay*. Nature, Vol. 467, No. 7319, 1081-1083.

[7] S. Geman and D. Geman, 1984. *Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images*. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 6, No. 6, 721-741.

[8] C. J. Geyer., 2011. *Introduction to Markov Chain Monte Carlo*. In *Handbook of Markov Chain Monte Carlo* (pp. 3-48). Chapman and Hall/CRC. 2011.

[9] I. Goodfellow, Y. Bengio and A. Courville. *Deep Learning*. MIT Press. 2016.

[10] T. Hastie., R. Tibshirani and J. Friedman. *The Elements of Statistical Learning - Data Mining, Inference, and Prediction*. Second Edition. Springer. 2009.

[11] W. K. Hastings, 1970. *Monte Carlo Sampling Methods Using Markov Chains and Their Applications*. Biometrika, Vol. 57, No. 1, 97-109.

[12] D. M. Higdon, 1998. *Auxiliary Variable Methods for Markov Chain Monte Carlo with Applications*. Journal of the American Statistical Association, Vol. 93, No. 442, 585-595.

[13] J. P. Huelsenbeck, F. Ronquist, R. Nielsen and J. P. Bollback, 2001. *Bayesian Inference of Phylogeny and Its Impact on Evolutionary Biology*. Science, Vol. 295, No. 5550, 2310-2314.

[14] C. S. Jensen and U. Kjaerluf, 1995. *Blocking Gibbs sampling in very large probabilistic expert systems*. International Journal of Human-Computer Studies, Vol. 42, No. 6, 647-666.

[15] J. S. Liu, 1994. *The Collapsed Gibbs Sampler in Bayesian Computations with Applications to a Gene Regulation Problem*. Journal of the American Statistical Association, Vol. 89, No. 427, 958-966.

[16] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller and E. Teller, 1953. *Equation of State Calculations by Fast Computing Machines*. The Journal of Chemical Physics, Vol. 21, No. 6.

[17] K. Murphy. *Machine Learning - A probabilistic perspective*. MIT Press, Cambridge, Massachusetts, London, England. 2012.

[18] C. Robert and G. Casella, 2011. *A Short History of Markov Chain Monte Carlo: Subjective Recollections from Incomplete Data*. Statistical Science, Vol. 26, No. 1, 102-115.

[19] T. Snijders, 2002. *Markov chain Monte Carlo estimation of exponential random graph models*. Journal of Social Structure, Vol. 3, No. 2, 1-40.