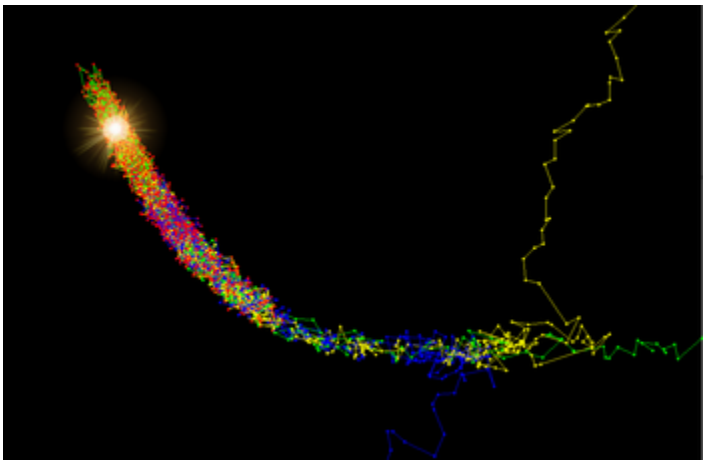
The Metropolis-Hastings Method

De Metropolis-Hastings Methode

Ravish Gangapersad 4627369

August 24, 2020



Thesis Committee Members:
Dr. ir. L.E. Meester
Dr. H.N. Kekkonen

Delft University of Technology
Department of Electrical Engineering, Mathematics and Computer Science

Abstract

In this report, our goal is to find a way to get some information such as the mean out of high dimensional densities. If we want to calculate the mean we need to calculate integrals, which are difficult to do for high dimensional densities. We cannot use the analytical or classical (deterministic) numerical rules for high dimensional problems for which we want to calculate the mean. These methods take a lot of computational time. To solve this problem we introduced the Markov Chain Monte Carlo (MCMC) method, the method samples from the distribution, and with these samples, we can approximate the mean. Then we explain the theory behind these methods and how we can use it. Then we introduced one MCMC method, in particular, the Metropolis-Hastings Algorithm. We explain how this method works and the theory behind it. From this, we see that the method is very easy to implement and can be used to approximate the mean. Then we approximate the mean for some examples using this method.

Contents

1. Introduction

If we make a photo for example of size 900×1500 pixels with a camera but shake our hand by accident the photo which we took will be a little bit blurry. In every photo, we take there will also be a little bit of noise. So our photo will have noise in it and be blurred. Now we want to try making the photo a little less blurry, so we want to make the photo sharper to resemble the reality of which we took the photo. This means we want to find the unknown sharp image θ. Note that the picture which we took is very large so we will have a high-dimensional data set y (blurred image). We would think that this is good because the more data you have the more detail we can have in our pictures (higher resolution). But we will run into computational problems, because of this high dimensional data. Lets now assume that the noise ε in the picture is Gaussian distributed and let us introduce the forward operator A which describes the blurring by smoothing the sharp image θ which we are trying to find. So from all of this, we get the following model

$$y = A\theta + \varepsilon. \tag{1}$$

Now to find θ we will use some terminology of Bayesian statistics. In Bayesian statistics we have a 'prior' density, this describes our knowledge of the unknown before the measurement. Let us note this density as $p(\theta)$ with $\theta \in \mathbb{R}^n$ the parameter we want to estimate. Furthermore, we have the likelihood, lets note this as $L(y|\theta)$ with y the data that we have. Then from the Bayes theorem we get, the 'posterior' distribution $\pi_y(\theta)$ of θ given y is

$$\pi_y(\theta) = \frac{L(y|\theta)p(\theta)}{N} \qquad \text{with } N = \int_{\mathbb{R}^n} L(y|\theta)p(\theta)d\theta, \text{ the normalisation constant.}$$

Note that N is a constant with respect to θ, the unknown that we want to find, that's why we introduce the following, the posterior is proportional to

$$\pi_y(\theta) \propto L(y|\theta)p(\theta).$$

With this we can calculate the posterior mean of function f, this is done in the following way

$$\mathbb{E}[f(X)] = \frac{\int_{\mathbb{R}^n} f(x)\pi_y(x)dx}{\int_{\mathbb{R}^n} \pi_y(x)dx}. \tag{2}$$

If we go back to the example of the picture, we see that this is a Bayesian inverse problem. The prior is our belief of how the picture will look like before we have taken the picture. The prior we choose in our example depends on the specific situation, this means we will choose a prior that is best suited to our example. We assumed that the noise in our image is Gaussian distributed, with this and equation (1) we can calculate the likelihood for our image. The solution that Bayesian statistics gives us for this problem is the posterior distribution. This means we don't know the specific values of each of the pixels but only how they are distributed. But we are interested in finding the best possible (sharp) image, which means we need to find the best possible value for each of the pixels. That's why we use a point estimator such as the mean to find this image. Now the problem that we face is if we want to use the mean as the point estimator then equation (2) tells us that this will be a very high dimensional integral which we need to solve.

We can't calculate these integrals analytically, so we can then try to use numerical methods such as the numerical quadrature rules (a numerical approximation of a definite integral) or we can try using the m-point rules (Simpson's rule) to solve these integrals. With the use of quadrature rules, we face 2 problems. If the dimension of the region on which we integrate is too high we can't use these rules, because we get the so-called curse of dimensionality. This means the cost to compute an approximation with a prescribed accuracy α depends exponentially on the dimension n of the problem. This means we will need a lot of computational time for very high dimensions. Furthermore, we have to know the non zero elements of our probability distribution (support) which can even be the whole of $\mathbb{R}^n$, but this is part of the information that we are looking for. Then we have the m-point rule, this has the drawback that for very high dimensions we will get very big computations that exceed the computational capacity of most computers.

This is where the Markov Chain Monte Carlo (MCMC) method comes in. It would be great if we would have samples from the desired target distribution. Then we could use the sum of these samples to approximate the mean of the target distribution. The MCMC method gives us a practical way to get these samples without much difficulty. We do this by constructing a Markov chain which is our sample so that the distribution of our sample approaches the target distribution

In this report, we will look at one particular MCMC method namely the Metropolis-Hastings (MH) algorithm. The MH method is heavily used in Bayesian statistics. The MH method is a practical way of creating samples that can then be used to approximate the mean of densities. What makes this method useful is that we don't need to know the normalization factor of our density, which is also often extremely difficult to calculate in practice.

The method was first introduced by Metropolis (1953) and later generalized by Hastings (1970). The article of Chib and Greenberg [1] gives us an introduction and explanation of the MH algorithm. The article by Roberts and Rosenthal [2] gives us the convergence results of the MCMC algorithms. In the book of Kaipio and Somersalo [3] we can find an example of an implementation of the MH algorithm. Furthermore, we can find examples of Bayesian inverse problems in this book. As we can see we already have a few articles and books which talk about this algorithm but there are many more. The introduction of MCMC algorithms has changed the way we look at Bayesian inverse problems, by allowing users to sample from posterior distributions of complicated statistical models. So there is a lot of attention being paid to MCMC methods and of which the MH algorithm is one of the most popular.

The rest of the paper is organized as follows. In Section 2 we first introduce the main definitions that we will need throughout this report. Then in Section 3 we introduce the Markov chain with all of the relevant definitions furthermore we talk about the properties of the Markov chain and then introduce and prove one of the main theorems which we will need to create the MH algorithm. In Section 4 we look at the MH algorithm, here we introduce and prove that the MH algorithm satisfies the conditions which are required in the theorems that provide convergence. In Section 5 we look at two example densities on which we implement the MH method. Then we introduce the step size, which is important for the implementation of the MH method. Furthermore, we talk about calculating the mean of these examples. To do this we make use of the different step sizes to look at the differences and then plot and show our results.

2. General definitions

We will start by introducing a few definitions that are frequently used in this report.

Definition 2.1. *The collection $\mathfrak{F}$ of subsets of the sample space Ω is called an event space or σ-algebra if*

1. *$\mathfrak{F}$ is non-empty,*
2. *if $A \in \mathfrak{F}$ then $\Omega \setminus A \in \mathfrak{F}$,*
3. *if $A_1, A_2, \cdots \in \mathfrak{F}$ then $\bigcup_{i=1}^{\infty} A_i \in \mathfrak{F}$.*

Definition 2.2. *The Borel σ-algebra $\mathfrak{B}(\mathbb{R}^n)$ is the smallest σ-algebra containing all open sets in $\mathbb{R}^n$. The sets in $\mathfrak{B}(\mathbb{R}^n)$ are called Borel sets.*

Definition 2.3. *A mapping $\mathbb{P} : \mathfrak{F} \to \mathbb{R}^n$ is called a probability measure on $(\Omega, \mathfrak{F})$ if*

1. *$\mathbb{P}(A) \geq 0$ for $A \in \mathfrak{F}$,*
2. *$\mathbb{P}(\Omega) = 1$ and $\mathbb{P}(\emptyset) = 0$,*
3. *if $A_1, A_2, \ldots$ are disjoint events in $\mathfrak{F}$ (in that $A_i \cap A_j = \emptyset$ whenever $i \neq j$) then*

$$\mathbb{P}\left(\bigcup_{i=1}^{\infty} A_i\right) = \sum_{i=1}^{\infty} \mathbb{P}(A_i).$$

Definition 2.4. *A probability space is a triple $(\Omega, \mathfrak{F}, \mathbb{P})$ of objects such that*

1. *Ω is a non-empty set,*
2. *$\mathfrak{F}$ is an event space of subsets of Ω,*
3. *$\mathbb{P}$ is a probability measure on $(\Omega, \mathfrak{F})$.*

Definition 2.5. *A random variable X on the probability space $(\Omega, \mathfrak{F}, \mathbb{P})$ is a mapping $X : \Omega \to \mathbb{R}^n$ such that for all $A \subseteq \mathbb{R}^n$ we have that $X^{-1}(A) := \{\omega \subseteq \Omega | X(\omega) \subseteq A\} \subseteq \Omega$.*

From here of on we will assume in this report that our σ-algebra $\mathfrak{F} = \mathfrak{B}(\mathbb{R}^n)$

Definition 2.6. *Any function π which satisfies*

$$\pi(x) \geq 0 \text{ for } x \in \mathbb{R}^n,$$

and

$$\int_{\mathbb{R}^n} \pi(x)dx = 1,$$

is the density function of some random variable.

Definition 2.7. *The indicator or characteristic function of a subset A of a set S is a function $\chi_A \colon S \to \{0,1\}$ defined as* $\chi_A(x) := \begin{cases} 1 & \text{if } x \in A, \\ 0 & \text{if } x \notin A. \end{cases}$

Definition 2.8. *The total variation distance between two probability measures $\mu_1(\cdot)$ and $\mu_2(\cdot)$ is:*

$$||\mu_1(\cdot) - \mu_2(\cdot)|| = \sup_A |\mu_1(A) - \mu_2(A)|,$$

where the supremum is taken over every set $A \subseteq \mathfrak{B}(\mathbb{R}^n)$.

Definition 2.9. *A probability measure μ is dominated by π ($\mu << \pi$) if $\pi(A) = 0$ implies $\mu(A) = 0$ for every set $A \subseteq \mathfrak{B}(\mathbb{R}^n)$.*

3. Markov Chain Monte Carlo method

Now if we have a high dimensional (posterior) density we want a way to extract specific information from it, just like in the case of the example of the picture. In statistics, we have different kinds of point estimators. For example, we have the mean and mode of a probability density. The mode is defined as the following:

Definition 3.1. *The mode of a continuous probability distribution is the point at which the probability density function $\pi(x)$ attains a maximum value. We have that*

$$\hat{X}_{mode} = \arg\max_x \pi(x).$$

If π is assumed to be a posterior density π_y then this is called the maximum a posteriori (MAP) estimator. Note that a probability density can attain more than one maximum, so the mode is not necessarily unique. To find the mode of a density we can make use of optimization algorithms. There are a few optimization algorithms available. Most of these algorithms iteratively try to find the maximum or minimum of a function. One of the drawbacks of such algorithms is that the iteration process can take a lot of computational time if we have a complex density. Another drawback is that these algorithms can get stuck at a local maximum, which we aren't interested in. That's why we are going to take a look at a different point estimator, the mean.

Definition 3.2. *The mean $\hat{X}_{mean}$ of a continuous random variable X with probability density function $\pi(x)$ is*

$$\hat{X}_{mean} = \mathbb{E}(X) = \int_{\mathbb{R}^n} x\pi(x)dx.$$

If π is assumed to be a posterior density π_y then this is called the conditional mean in Bayesian statistics. Now if we look at this definition we see that we have to calculate an integral. This can also be very complicated to do if we have very high dimensional probability densities. The numerical quadrature rules (and the m-point rules) are all methods we can not use to calculate these specific integrals. They work in the following manner let μ be probability measure over $\mathbb{R}^n$ and we want to integrate f with respect to μ. Then define a set of support points $1 \leq j \leq N$ and a set of corresponding weights w_j to get an approximation

$$\int_{\mathbb{R}^n} f(x)\mu(dx) \approx \sum_{j=1}^{N} w_j f(x_j).$$

These methods have a few problems, with one of the biggest problems being that it takes a lot of computational time if we have a high dimensional density. In Section 1 we briefly talked about the curse of dimensionality, this usually refers to what happens if we keep on adding more dimensions. This means it becomes more difficult to attain certain quantities such as the mean because the computational time increases exponentially with each added dimension. Furthermore, we explained that in the numerical quadrature rules (and the m-point rules) we evaluated the probability density at some points to calculate the integrals. This is not a good way of calculating the integrals because we don't know the support of the density. To solve this we can let the density itself give us a set of points (sample) that supports the distribution. To illustrate this better let's introduce a function π, on a state space $\mathbf{S}$, with $\mathbf{S}$ an open subset of $\mathbb{R}^n$, with π possibly unnormalised and satisfying $0 < \int_{\mathbf{S}} \pi < \infty$. From this, we define a probability measure $\Pi(\cdot)$ on $\mathbf{S}$,

$$\Pi(A) = \frac{\int_A \pi(x)dx}{\int_{\mathbf{S}} \pi(x)dx}.$$

We want to estimate the expectation of function $f : \mathbf{S} \to \mathbb{R}^n$ with respect to $\Pi(\cdot)$ we get

$$\mathbb{E}[f(X)] = \frac{\int_{\mathbf{S}} f(x)\pi(x)dx}{\int_{\mathbf{S}} \pi(x)dx}.$$

If π is a complicated function and $\mathbf{S}$ is high-dimensional we will get very difficult to calculate integrals. That's why we introduce the classical Monte Carlo simulations to use for this problem we will simulate $x_1, x_2, \ldots, x_N \sim \Pi(\cdot)$, and then estimate the mean of f with respect to $\Pi(\cdot)$ by

$$\mathbb{E}[f(X)] \approx \frac{1}{N}\sum_{i=1}^{N} f(x_i). \tag{3}$$

Here we directly simulate from $\Pi(\cdot)$, but if the π is very complicated this is very difficult to do. That's why we use the MCMC methods, these algorithms generate a group of samples that can be used for Monte Carlo simulation. In Section 4 we will introduce the MH algorithm one of the most used MCMC methods.

3.1. Markov Chain

Here we are going to introduce the Markov chain.

Definition 3.3. *Let $\mathfrak{B} = \mathfrak{B}(\mathbb{R}^n)$ denote the Borel sets over $\mathbb{R}^n$. A mapping $P : \mathbb{R}^n \times \mathfrak{B} \to [0, 1]$ is called a probability transition kernel, if*

1. *for each $B \in \mathfrak{B}$, the mapping $\mathbb{R}^n \to [0, 1], x \mapsto P(x, B)$ is a measurable function.*
2. *for each $x \in \mathbb{R}^n$, the mapping $\mathfrak{B} \to [0, 1]$, $B \mapsto P(x, B)$ is a probability distribution.*

Now let $x \in \mathbb{R}^n$ and B a measurable set in $\mathfrak{B}$ then $P(x, B) = \mathbb{P}(X_n \in B|X_{n-1} = x)$ is a probability transition kernel.

Definition 3.4. *A discrete time stochastic process is an ordered set $\{X_j\}_{j=1}^{\infty}$ of random variables $X_j \in \mathbb{R}^n$.*

Definition 3.5. *A discrete time process $\mathbf{X} = \{X_0, X_1, X_2, \ldots\}$ is called a Markov chain if and only if the state at time t merely depends on the state at time $t - 1$. More precisely, the transition probabilities*

$$\mathbb{P}[X_t = a_t|X_{t-1} = a_{t-1}, \ldots, X_0 = a_0] = \mathbb{P}[X_t = a_t|X_{t-1} = a_{t-1}],$$

for all values $a_0, a_1, \ldots, a_t$ and all $t \geq 1$

This means that the Markov chains are "memoryless" discrete time processes.

Definition 3.6. *Let μ denote a probability measure over $\mathbb{R}^n$. A time- homogeneous Markov chain with the transition kernel P is a stochastic process $\{X_j\}_{j=1}^{\infty}$ with the properties*

$$\mu_{X_{j+1}}(B_{j+1}|x_1, \ldots, x_j) = \mu_{X_{j+1}}(B_{j+1}|x_j) = P(x_j, B_{j+1}).$$

The first equality means that the probability of $X_{j+1} \in B_{j+1}$ conditioned on observations $X_1 = x_1, \ldots, X_j = x_j$ is equal to the probability conditioned on $X_j = x_j$. This means that for any given time, the conditional probability distribution of the future state of the process, given present and past states, depends only on the present state and not at all on the past states (Markov property). The second equality states that the dependence of adjacent moments do not vary in time, so kernel P does not depend on time j.

3.2. Properties of the Markov chain

We have introduced the Markov chain in the previous section, now we are going to look at the properties of the Markov chain.

Definition 3.7. *Let μ denote a probability measure over $\mathbb{R}^n$. Then the transition kernel that propagates k steps forward in time is defined as the following:*

$$P^{(k)}(x_j, B_{j+k}) = \mu_{X_{j+k}}(B_{j+k}|x_j) = \int_{\mathbb{R}^n} P(x_{j+k-1}, B_{j+k}) P^{(k-1)}(x_j, dx_{j+k-1}), \quad k \geq 2,$$

where $P^1(x_j, B_{j+1}) = P(x_j, B_{j+1})$.

So if we have that μ_{X_j} is the probability distribution of X_j, then the distribution of X_{j+1} is given by

$$\mu_{X_{j+1}}(B_{j+1}|x_j) = \mu_{X_j} P(B_{j+1}) = \int_{\mathbb{R}^n} P(x_j, B_{j+1}) \mu_{X_j}(dx_j).$$

Definition 3.8. *Let μ denote a probability measure over $\mathbb{R}^n$. The measure μ is an invariant measure of $P(x_j, B_{j+1})$ if*

$$\mu P(B) = \int_{\mathbb{R}^n} P(x, B)\mu(dx) = \int_B \mu(dx) = \mu(B),$$

that is the distribution of the random variable X_j, before the time step $j \to j+1$ is the same as the variable X_{j+1} after the step.

This means that if $X_j \sim \mu$ then $X_{j+1} \sim \mu P = \mu$.

Definition 3.9. *We define $\tau_A = \inf\{n \geq 1; X_n \in A\}$ to be the first return time to A.*

Now if we start at point $x \in \mathbb{R}^n$ we note the transition kernel as $\mathbf{P}_x$. If we have a Markov chain with a invariant distribution Π and a set $A \subset \mathfrak{B}(\mathbb{R}^n)$ and suppose that $\mathbf{P}_x(\tau_A < \infty) > 0$, this means there is a chance that we will end up in A in finite time if we begin at point x. (The $\tau_A < \infty$ means that we will eventually return to A.). From this we introduce the following lemma. For every invariant distribution Π almost everywhere and $x \in \mathbb{R}^n$ we can make the stronger statement that $\mathbf{P}_x(\tau_A < \infty) = 1$, this means we will always end up in A in finite time.

Lemma 3.10. *Consider a Markov Chain on $\mathbb{R}^n$, having invariant distribution $\Pi(\cdot)$. Suppose that for some $A \subseteq \mathfrak{B}(\mathbb{R}^n)$, we have $\mathbf{P}_x(\tau_A < \infty) > 0$ for all $x \in \mathbb{R}^n$. Then for all Π a.e $x \in \mathbb{R}^n$, $\mathbf{P}_x(\tau_A < \infty) = 1$.*

Proof. Suppose the contrary that the the conclusion does not hold, i.e. that

$$\Pi\{x \in \mathbb{R}^n : \mathbf{P}_x(\tau_A = \infty) > 0\} > 0. \tag{4}$$

This means that if we start at $x \in \mathbb{R}^n$ we will never end up in A in finite time.

With equation (4), we can find $\delta_1 > 0$ and a subset $B_1 \subseteq \mathfrak{B}(\mathbb{R}^n)$ with $\Pi(B_1) > 0$, such that $\mathbf{P}_x(\tau_A < \infty) \leq 1 - \delta_1$ for all $x \in B_1$. Since $P_x(\tau_A < \infty) > 0$ for all $x \in \mathbb{R}^n$, we can find $l_o \in \mathbb{N}$ and $\delta_2 > 0$ and $B_2 \subseteq B_1$ with $\Pi(B_2) > 0$ and with $P^{l_0}(x, A) \geq \delta_2$ for all $x \in B_2$. Set $\eta = \#\{k \geq 1; X_{kl_0} \in B_2\}$. Then for any $r \in \mathbb{N}$ and $x \in \mathbb{R}^n$, we have $\mathbf{P}_x(\tau_A = \infty,\ \eta = r) \leq (1 - \delta_2)^r$. In particular, $\mathbf{P}_x(\tau_A = \infty,\ \eta = \infty) = 0$. Hence, for $x \in B_2$, we have

$$\begin{aligned}\mathbf{P}_x(\tau_A = \infty,\ \eta < \infty) &= 1 - \mathbf{P}_x(\tau_A = \infty,\ \eta = \infty) - \mathbf{P}_x(\tau_A < \infty) \\ &\geq 1 - 0 - (1 - \delta_1) = \delta_1.\end{aligned}$$

Hence, there is $l \in \mathbb{N}, \delta > 0$, and $B \subseteq B_2$ with $\Pi(B) > 0$, such that

$$\mathbf{P}_x(\tau_A = \infty,\ \sup\{k \geq 1; X_{kl_0} \in B_2\} < l) \geq \delta, \qquad x \in B.$$

Since $B \subseteq B_2$, we have

$$\sup\{k \geq 1; X_{kl_0} \in B_2\} \geq \sup\{k \geq 1; X_{kl_0} \in B\}.$$

So from this we get that

$$\mathbf{P}_x(\tau_A = \infty, \sup\{k \geq 1; X_{kl_0} \in B\} < l) \geq \delta, \qquad x \in B \tag{5}$$

Let B, l, l_0 and δ be as above. Let $L = ll_0$, and $S = \sup\{k \geq 1; X_{kl} \in B\}$, using the convention that $S = -\infty$ if the set $\{k \geq 1; X_{kl} \in B\}$ is empty. Now by using the definition of invariant and equation (5) we get the following computation

$$\begin{aligned}
\int_{x\in\mathbb{R}^n} \mathbf{P}_x[s = r,\ X_{jL} \notin A]\Pi(dx) &= \int_{x\in\mathbb{R}^n}\int_{y\in B} \mathbf{P}_y[S = -\infty,\ X_{(j-r)L} \notin A]P^{rL}(x, dy)\Pi(dx) \\
&= \int_{y\in B}\int_{x\in\mathbb{R}^n} \mathbf{P}_y[S = -\infty,\ X_{(j-r)L} \notin A]\Pi(dx)P^{rL}(x, dy) \\
&= \int_{y\in B} \mathbf{P}_y[S = -\infty,\ X_{(j-r)L} \notin A]\Pi(dy) \\
&\geq \int_{y\in B} \delta\Pi(dy) = \delta\Pi(B).
\end{aligned}$$

This shows that for all integers $1 \leq r \leq j$,

$$\int_{x\in\mathbb{R}^n} \mathbf{P}_x[S = r, X_{jL} \notin A]\Pi(dx) \geq \delta\Pi(B). \tag{6}$$

Now using the definition of invariant and equation (6) we have that for any $j \in \mathbb{N}$,

$$\begin{aligned}
\Pi(A^C) &= \int_{x\in\mathbb{R}^n} P^{jL}(x, A^C)\Pi(dx) = \int_{x\in\mathbb{R}^n} \mathbf{P}_x[X_{jL} \notin A]\Pi(dx) \\
&\geq \sum_{r=1}^{j} \int_{x\in\mathbb{R}^n} \mathbf{P}_x[S = r, X_{jL} \notin A]\Pi(dx) \geq \sum_{r=1}^{j} \delta\Pi(B) = j\delta\Pi(B).
\end{aligned}$$

For $j > \frac{1}{\delta\Pi(B)}$, this gives $\Pi(A^C) > 1$, which is impossible. This gives a contradiction, and hence completes the proof of Lemma 3.10. □

3.3. Convergence of the Markov Chain

Here we are going to introduce a few theorems which are very important for the MCMC methods. These theorems give us the result that the Markov chains converge to the invariant distribution, given it satisfies certain conditions.

Definition 3.11. *Let μ denote a probability measure over $\mathbb{R}^n$. Then the transition kernel P is irreducible (with respect to μ) if for each $x \in \mathbb{R}^n$ and $B \in \mathfrak{B}$(borel set) with $\mu(B) > 0$ there exists an integer k such that $\mathbf{P}^{(k)}(x, B) > 0$.*

This means that the transition kernel P can visit every set of positive measure (we can reach every point in our measure probability distribution). This is the same as writing $\mathbf{P_x}(\tau_B < \infty) > 0$.

Definition 3.12. *Let P be an irreducible kernel. We say that P is periodic if, for some integer $m \geq 2$, there is a set of disjoint nonempty sets $\{E_1, \ldots, E_m\} \subset \mathbb{R}^n$ such that for all $j = 1, \ldots, m$ and all $x \in E_j$, $P(x, E_{j+1(\mod m)}) = 1$.*

This means that the periodic transition kernel gives us a Markov chain that is in an infinite periodic loop.

Definition 3.13. *A kernel P is an aperiodic kernel if it is not periodic.*

We see that the aperiodic kernel does not give us an infinite periodic loop of Markov chains.

Theorem 3.14. *Let μ be a probability measure in $\mathbb{R}^n$ and $\{X_j\}$ a time-homogeneous Markov chain with a transition kernel P. Assume further that μ is an invariant measure of the transition kernel P, and that P is irreducible and aperiodic. Then for all μ a.e $x \in \mathbb{R}^n$*

$$\lim_{N\to\infty} ||P^{(N)}(x,\cdot) - \mu(\cdot)|| = 0.$$

In particular, $\lim_{N\to\infty} P^N(x,B) = \mu(B)$ for all $B \in \mathfrak{B}$ with $\mathfrak{B}$ the Borel sets over $\mathbb{R}^n$.

Theorem 3.15. *Let μ, X and P be as in Theorem 3.14. Then for $f \in L^1(\mu(dx))$,*

$$\lim_{N\to\infty} \frac{1}{N}\sum_{j=1}^{N} f(X_j) = \int_{\mathbb{R}^n} f(x)\mu(dx),$$

almost certainly. This is called the ergodicity property.

We will prove Theorem 3.14, but to do this we will first need to introduce some lemmas. The proof of Theorem 3.15 can be found at [4].

Now we are going to introduce the concept of coupling. We are going to do this because the proof becomes a lot less difficult and technical than the other proofs that do not make use of this.

Lemma 3.16. *Suppose we have two random variables X and Y, defined jointly on $\mathbb{R}^n$. If we write $\mu(X)$ and $\mu(Y)$ for their respective probability distributions, then we can write*

$$\begin{aligned}
||\mu(X) - \mu(Y)|| &= \sup_A |\mathbb{P}(X \in A) - \mathbb{P}(Y \in A)| \\
&= \sup_A |\mathbb{P}(X \in A, X = Y) + \mathbb{P}(X \in A, X \neq Y) \\
&\quad - \mathbb{P}(Y \in A, Y = X) - \mathbb{P}(Y \in A, Y \neq X)| \\
&= \sup_A |\mathbb{P}(X \in A, X \neq Y) - \mathbb{P}(Y \in A, Y \neq X)| \\
&\quad \leq \mathbb{P}(X \neq Y)
\end{aligned}$$

That is, the total variation distance between the two random variables is bounded by the probability that they are unequal.

Definition 3.17. *A subset $C \subseteq \mathbb{R}^n$ is small (or, (n_0, ε, μ)-small) if there exists a positive integer $n_0, \varepsilon > 0$, and a probability measure $\mu(\cdot)$ on $\mathbb{R}^n$ such that the following minorisation condition holds:*

$$P^{n_0}(x,\cdot) \geq \varepsilon\mu(\cdot) \quad x \in C,$$

i.e. $P^{n_0}(x,A) \geq \varepsilon\mu(A)$ for all $x \in C$ and all measurable $A \subseteq \mathbb{R}^n$.

Now suppose we have a small set C. We will implement a coupling construction on this set. The construction has the following idea, we run two copies $\{X_n\}$ and $\{X'_n\}$ of the Markov chain, with each marginally following the updating rules $P(x,\cdot)$. Then their joint construction (using C) gives them the highest probability possible of becoming equal to each other. Note that one of the copies X'_n starts with the invariant measure Π.

THE COUPLING CONSTRUCTION.

1 Start with $X_0 = x$ and $X_0' \sim \Pi(\cdot)$, and $n = 0$, and repeat the following loop forever. **Beginning of Loop**. Given X_n and X_n':

2 If $X_n = X_n'$, choose $X_{n+1} \sim P(X_n, \cdot)$, and replace n by $n+1$

3 Else, if $(X_n, X_n') \in C \times C$, then:

4 (a) with probability $\varepsilon > 0$, choose $X_{n+n_0} = X_{n+n_0}' \sim \mu(\cdot)$.

5 (b) else, with probability $1-\varepsilon$, conditionally independently choose

$$X_{n+n_0} \sim \frac{1}{1-\varepsilon}[P^{n_0}(X_n, \cdot) - \varepsilon\mu(\cdot)], \quad X_{n+n_0}' \sim \frac{1}{1-\varepsilon}[P^{n_0}(X_n', \cdot) - \varepsilon\mu(\cdot)].$$

In the case of $n_0 > 1$, for completeness go back and construct $X_{n+1}, \ldots, X_{n+n_0-1}$ from their correct conditional distributions given X_n and X_{n+n_0}, and similarly (and conditionally independently) construct $X_{n+1}', \ldots, X_{n+n_0-1}'$ from their correct conditional distributions given X_n' and X_{n+n_0}'. In any case, replace n by $n+n_0$.

6 Else, conditionally indpendently choose $X_{n+1} \sim P(X_n, \cdot)$ and $X_{n+1}' \sim P(X_n', \cdot)$, and replace n by $n+1$.

7 **Then return to Beginning of Loop.**

We then use Lemma 3.16 to find bounds. For example if we are in small set C we have probability ε of at least making $X_n \sim P^n(x, \cdot)$ and $X_n' \sim \Pi(\cdot)$ equal. From this we would get that $\mathbb{P}(X_n \neq X_n') \leq (1-\varepsilon)$. This could then be used to bound $||P^n(x, \cdot) - \Pi||$.

Theorem 3.18. *Every irreducible Markov chain, with probability measure Π, on the Borel sets over $\mathbb{R}^n$, contains a small set $C \subseteq \mathfrak{B}(\mathbb{R}^n)$ with $\Pi(C) > 0$. (In fact, each $B \subseteq \mathfrak{B}(\mathbb{R}^n)$ with $\Pi(B) > 0$ in turn contains a small set $C \subseteq B$ with $\Pi(C) > 0$). Furthermore, the minorisation measure $\mu(\cdot)$ may be taken to satisfy $\mu(C) > 0$.*

This theorem ties everything together it brings Markov chains, the coupling construction and small sets into one theorem. To see the proof of Theorem 3.18 see [5].

Lemma 3.19. *Consider an aperiodic Markov chain on $\mathbb{R}^n$, with invariant distribution $\Pi(\cdot)$. Let $\mu(\cdot)$ be any probability measure on $\mathbb{R}^n$. Assume that $\mu(\cdot) << \Pi(\cdot)$, and that for all $x \in \mathbb{R}^n$, there is $n = n(x) \in \mathbb{N}$ and $\delta = \delta(x) > 0$ such that $P^n(x, \cdot) \geq \delta\mu(\cdot)$ (for example, this always holds if $\mu(\cdot)$ is a minorisation measure for a small set which is reachable from all states). Let $T = \{n \geq 1; \exists \delta_n > 0 \text{ s.t. } \int P^n(x, \cdot)\mu(dx) \geq \delta_n \mu(\cdot)\}$, and assume that T is non-empty. Then there is $n_* \in \mathbb{N}$ with $T \supset \{n_*, n_*+1, n_*+2, \ldots\}$.*

For the proof of Lemma 3.19 see [2] Lemma 35. Now with all of the lemmas and definitions above we will prove Theorem 3.14.

Proof of Theorem 3.14. We want to show that the pair X_n, Y_n ends in $C \times C$ with C a small set, infinitely many times, which means that the pair has infinitely many times to couple with probability $\varepsilon > 0$ so it will do so with probability 1. Let C be a small set as in Theorem 3.18 and consider the coupling construction $\{(X_n, Y_n)\}$. Let $G \subseteq \mathbb{R}^n \times \mathbb{R}^n$ be the set of (x, y) for which $\mathbf{P}_{(x,y)}(\exists n \geq 1; X_n = Y_n) = 1$. From the coupling construction, we see that if $(X_0, Y_0) \equiv (x, Y_0) \in G$, then $\lim_{n\to\infty} \mathbf{P}[X_n = Y_n] = 1$, so that $\lim_{n\to\infty} ||P^n(x, \cdot) - \Pi(\cdot)|| = 0$, proving Theorem 3.14. Hence it suffices to show that for all Π-a.e. $x \in \mathbb{R}^n$, we have that $\mathbf{P}[(x, Y_0) \in G] = 1$.

Let G be as above, let $G_x = \{y \in \mathbb{R}^n; (x, y) \in G\}$ for $x \in \mathbb{R}^n$, and let $\bar{G} = \{x \in \mathbb{R}^n; \Pi(G_x) = 1\}$. So we want to prove that $\Pi(\bar{G}) = 1$. Indeed since $\mu(C) > 0$ by Theorem 3.18 it follows from Lemma 3.19 that, from any $(x, y) \in \mathbb{R}^n \times \mathbb{R}^n$, the joint chain has positive probability of eventually hitting $C \times C$. It then follows by applying Lemma 3.10 to the joint chain, that the joint chain will return to $C \times C$ with probability 1 from $(\Pi \times \Pi)$-a.e $(x, y) \notin C \times C$. Once the joint chain reaches $C \times C$, the chain could leave $C \times C$ without coupling they will by Lemma 3.10 again return to $C \times C$ with probability 1. Hence, the joint chain will repeatedly return to $C \times C$ with probability 1, until such time as $X_n = Y_n$. And by the coupling construction, each time the joint chain is in $C \times C$, it has probability $\geq \varepsilon$ of then forcing $X_n = Y_n$. Hence, eventually we will have $X_n = Y_n$, thus proving that $(\Pi \times \Pi)(G) = 1$. Now, if we had $\Pi(\bar{G}) < 1$, then we would have

$$(\Pi \times \Pi)(G^C) = \int_{\mathbb{R}^n} \Pi(G_x^C)\Pi(dx) = \int_{\bar{G}^C} [1 - \Pi(G_x)]\Pi(dx) > 0,$$

contradicting the fact that $(\Pi \times \Pi)(G) = 1$. □

We will not prove Theorem 3.15 in this report. From the theorems above we see, that we need to create an invariant, aperiodic, and irreducible transition kernel P and then draw a sequence of sample points from P. Then we will be able to calculate the integrals using the Monte Carlo integration method.

4. The Metropolis-Hastings Algorithm

The Metropolis-Hastings (MH) Algorithm is an MCMC method, this method is mostly used to approximate multidimensional distributions and to determine the integral of a distribution. This method can also be used for one-dimensional distributions but there are easier MCMC methods that we can use for these distributions. To construct this algorithm we need a few ingredients. First, let μ be the target measure in $\mathbb{R}^n$ and let μ be absolutely continuous with respect to the Lebesgue measure so that there exists π such that $\mu(dx) = \pi(x)dx$. Then we want to create a transition kernel $P(x, B)$ with $B \in \mathfrak{B}$ (Borel set) such that μ is its invariant measure. Let's assume P as any transition kernel, then if we apply P on a point $x \in \mathbb{R}^n$ then the kernel either goes to another point $y \in \mathbb{R}^n$ or stays in its place. So then we can split our kernel $P(x, B)$ into two parts in the following manner

$$P(x, B) = \int_B K(x, y)dy + r(x)\chi_B(x), \tag{7}$$

with χ_B the characteristic function of the set $B \in \mathfrak{B}$. The $K(x, y) \geq 0$ gives us the probability of moving from point x to the point y. Note that this K is not necessarily a density, because K does not always integrate to one. The $r(x) \geq 0$ tells us the probability of the chain staying in its place at x. We use the characteristic function χ_B because if $x \notin B$, then the MH algorithm goes to B only through a move. Then we have the condition that

$$P(x, \mathbb{R}^n) = \mathbb{P}(X_n \in \mathbb{R}^n | X_{n-1} = x) = 1,$$

this is always true, because $\{X_j\}_{j=1}^{\infty} \in \mathbb{R}^n$ and we know that P is a probability measure so it means that this must be equal to 1. Then we make use of characteristic function $\chi_{\mathbb{R}^n}$. We know that $x \in \mathbb{R}^n$, so this means that $\chi_{\mathbb{R}^n} = 1$. From these two conditions we get the following

$$\begin{aligned} P(x, \mathbb{R}^n) = \int_{\mathbb{R}^n} K(x, y)dy + r(x)\chi_{\mathbb{R}^n}(x) \implies \\ 1 = \int_{\mathbb{R}^n} K(x, y)dy + r(x) \implies \\ r(x) = 1 - \int_{\mathbb{R}^n} K(x, y)dy. \end{aligned} \tag{8}$$

If we look at Theorem 3.14 we see that we must have an invariant measure $\mu(dx)(= \pi(x)dx)$ of P in order for the kernel P to satisfy this theorem, we must have the following identity

$$\begin{aligned} \mu P(B) = \int_{\mathbb{R}^n} P(x, B)\pi(x)dx &= \int_{\mathbb{R}^n} \left(\left[\int_B K(x, y)dy \right] + r(x)\chi_B(x) \right) \pi(x)dx \\ &= \int_{\mathbb{R}^n} \left[\int_B K(x, y)dy \right] \pi(x)dx + \int_{\mathbb{R}^n} r(x)\chi_B(x)\pi(x)dx \\ &= \int_B \left[\int_{\mathbb{R}^n} K(x, y)\pi(x)dx \right] dy + \int_B r(x)\pi(x)dx \\ &= \int_B \left[\int_{\mathbb{R}^n} K(x, y)\pi(x)dx \right] dy + \int_B r(y)\pi(y)dy \\ &= \int_B \left(\left[\int_{\mathbb{R}^n} K(x, y)\pi(x)dx \right] + r(y)\pi(y) \right) dy \\ &= \int_B \pi(y)dy. \end{aligned}$$

For the above to be true for all $B \in \mathfrak{B}$ we assume

$$\int_{\mathbb{R}^n} K(x, y)\pi(x)dx + r(y)\pi(y) = \pi(y) \implies \int_{\mathbb{R}^n} K(x, y)\pi(x)dx = (1 - r(y))\,\pi(y) \tag{9}$$

Now if we use equation (8) we see that

$$r(y) = 1 - \int_{\mathbb{R}^n} K(y,x)dx.$$

Then if we fill this in equation (9) we get the following

$$(1 - r(y))\,\pi(y) = \left(\int_{\mathbb{R}^n} K(y,x)dx\right)\pi(y) = \int_{\mathbb{R}^n} \pi(y)K(y,x)dx.$$

With all of these ingredients we get the following condition

$$\int_{\mathbb{R}^n} \pi(y)K(y,x)dx = \int_{\mathbb{R}^n} \pi(x)K(x,y)dy. \tag{10}$$

This is called the balance equation. In practice equation (10) is difficult to work with, because we have to work with a lot of integrals. To avoid this problem we make a stronger assumption

$$\pi(y)K(y,x) = \pi(x)K(x,y),$$

which is called the detailed balance equation. To construct the Metropolis-Hastings Algorithm we assume that the transition kernel K satisfies the detailed balance condition. To do this we first introduce the function $q : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}_+$ with the property $\int q(x,y)dy = 1$. This q is called the proposal distribution or candidate-generating kernel. With this function q we define transition kernel,

$$Q(x,A) = \int_A q(x,y)dy.$$

If q satisfies the detailed balance equation we can simply set $K(x,y) = q(x,y)$ and $r(x) = 0$ and we are done. If that is not the case we define

$$K(x,y) = \alpha(x,y)q(x,y).$$

This α is a correction term which must be determined and is also referred to as the probability of move. Let's assume there are $x, y \in \mathbb{R}^n$ which don't satisfy the detailed balance equation, so we assume that we have the following inequality

$$\pi(y)q(y,x) < \pi(x)q(x,y). \tag{11}$$

This means that we move from x to y too often and from y to x less frequently. This means we want to choose an α so that we can get the following expression

$$\pi(y)\alpha(y,x)q(y,x) = \pi(x)\alpha(x,y)q(x,y),$$

this is achieved if we fix

$$\alpha(y,x) = 1 \text{ and } \alpha(x,y) = \frac{\pi(y)q(y,x)}{\pi(x)q(x,y)} < 1. \tag{12}$$

Now we reverse the x and y see that we must have that

$$\alpha(x,y) = 1 \text{ and } \alpha(y,x) = \frac{\pi(x)q(x,y)}{\pi(y)q(y,x)} < 1. \tag{13}$$

Now if we look at the $\alpha(x,y)$ in expressions (12) and (13) we see that we can define

$$\alpha(x,y) = \min\left(1, \frac{\pi(y)q(y,x)}{\pi(x)q(x,y)}\right)$$

. Now if we have a normalisation constant in our probability distribution π, then we see that in the way we have defined our α this constant vanishes. Now if we take everything we have defined and fill this in equation (7) we get the following

$$P(x,B) = \int_B \alpha(x,y)q(x,y)dy + \left(1 - \int_{\mathbb{R}^n} \alpha(x,y)q(x,y)dy\right)\chi_B(x).$$

This P is called the Metropolis-Hastings kernel. This transition kernel satisfies all of the properties needed for Theorem 3.14 that we discussed in section 2, so the MH method converges asymptotically. Now with this kernel we get the following implementation of the MH algorithm.

Implementation of the Metropolis-Hastings Algorithm.

1 First we choose an initial value x_0.
2 Then we iterate for i $= 1, \ldots, n_{draws}$ the following:
3 First we simulate y from the candidate (proposal) distribution $q(x_{i-1}, y)$.
4 Then we compute the acceptance probability

$$\alpha = \min\left\{\frac{\pi(y)q(y, x_{i-1})}{\pi(x_{i-1})q(x_{i-1}, y)}, 1\right\},$$

with target density $\pi(x)$.
5 We simulate $U \sim \mathcal{U}(0, 1)$, and check the following:
6 If $u \leq \alpha$, then accept. So $x_i = y$.
7 And if $u > \alpha$, then reject. So $x_i = x_{i-1}$.

If we look into the implementation of the algorithm we see that in the α there is a $q(y, x_{i-1})$ and $q(x_{i-1}, y)$. If we choose $q(x_{i-1}, y)$ symmetric, we will have

$$q(x_{i-1}, y) = q(y, x_{i-1}).$$

From this we get the following simplification

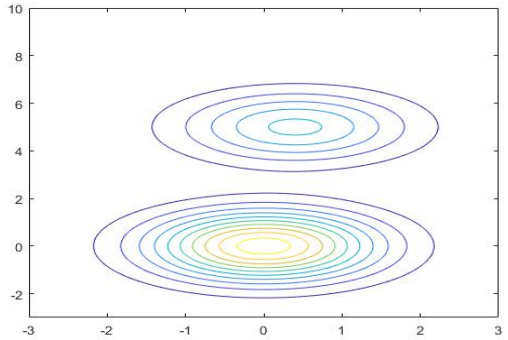
$$\alpha(x, y) = \min\left\{\frac{\pi(y)q(y, x_{i-1})}{\pi(x_{i-1})q(x_{i-1}, y)}, 1\right\} = \min\left\{\frac{\pi(y)}{\pi(x_{i-1})}, 1\right\}. \tag{14}$$

5. Examples

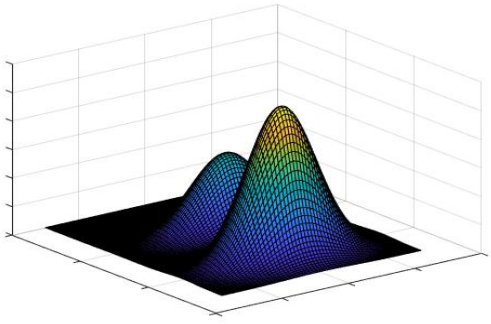
We will now consider two examples, we will use the MH method to sample from them. Then we will approximate the mean by using the samples and look a bit into how the algorithm depends on the candidate distribution q and a so-called step size γ. Note that we will be working in $\mathbb{R}^2$, so $x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$.

Example 1. We have the target density $\pi_1(x) \propto \frac{1}{3}\phi(x, \theta_1, \Sigma) + \frac{2}{3}\phi(x, \theta_2, \Sigma)$, with

$$\phi(x, \theta, \Sigma) = \frac{1}{\sqrt{2\pi \det(\Sigma)}} \exp\left(-\frac{(x-\theta)^T\Sigma^{-1}(x-\theta)}{2}\right),$$
$$\theta_1 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \theta_2 = \begin{pmatrix} 0.4 \\ 5 \end{pmatrix} \text{ and } \Sigma = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.$$



(a) Contour plot of target density $\pi_1(x)$.



(b) The 3-D plot of target density $\pi_1(x)$.

Figure 1: Here we plot the 3-D image and contour plot of π_1 side by side.

Then we will implement and look at the psuedo code of the MH algorithm on this example with the candidate distribution

$$q(x,y) \propto \exp\left(-\frac{1}{2\gamma^2}||x-y||^2\right).$$

If we look at q we see that it is Gaussian distributed with mean 0 and that it is symmetric. Furthermore the γ tells us how wide the q is.

Psuedo code of MH for the function $\pi_1(\mathbf{x})$.

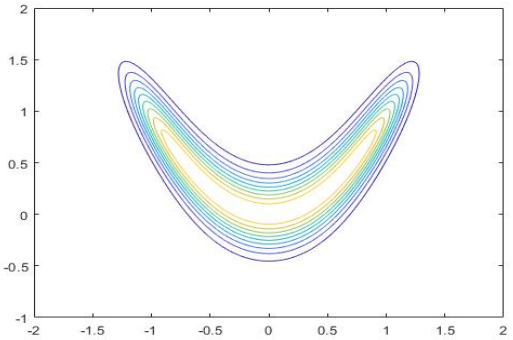
Pick initial value x_0. Set $x = x_0$
For $k = 2 : K$ do
    Calculate $\pi_1(x)$
    Draw $W \sim N(0, \gamma^2 I)$, set $y = x + w$
    Calculate $\pi_1(y)$
    Calculate $\alpha(x,y) = \min\left(1, \frac{\pi_1(y)}{\pi_1(x)}\right)$
    Draw $U \sim \mathcal{U}([0,1])$
    if $u < \alpha(x,y)$
        Accept: Set $x = y,\ x_k = x,$ else
        Reject: Set $x_k = x,$
    end
end

In the code we see a K this is the total number of iterations we want to do, in our case K = 5000. In the pseudo-code above we also see that there is a γ, this γ works as a step size in the MH algorithm. The MH algorithm depends on the step size γ through q, because the variance of our q is equal to γ. Now if we take the γ very large, this means the multivariate normal distribution from which we take our w will give us a high chance that the candidate y that we choose will be further away from the previous value that the algorithm accepted. This is true because the value which was accepted by the MH algorithm, is now the mean of the multivariate normal distribution from which we will now sample our next candidate, so if you have a high variance you will automatically have a high chance of picking a candidate that is further away from the mean and this mean is the latest accepted draw of the MH algorithm. And if we take γ very small the opposite is true, then we have a high chance of picking a candidate that is close to the latest accepted draw.

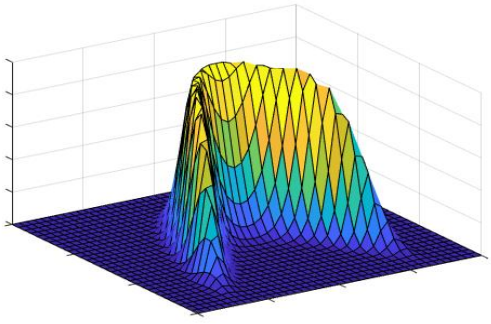
Example 2. We will now use the pseudo-code to sample from the target density

$$\pi_2(x) = \pi_2(x_1, x_2) \propto \exp\left(-10(x_1^2 - x_2)^2 - \left(x_2 - \frac{1}{4}\right)^4\right),$$

with the same candidate density q.



(a) Contour plot of target density $\pi_2(x)$.



(b) The 3-D plot of target density $\pi_2(x)$.

Figure 2: Here we plot the 3-D image and contour plot of π_2 side by side.

Now that we have introduced the two examples above, we will study Examples 1 and 2 by plotting the accepted draws of the MH algorithm in the contour plot of each example. To do this we first choose an initial value for each example, which we then use for the MH method. Furthermore we will choose different step sizes and apply the MH algorithm on target distribution $\pi_1(x)$ and $\pi_2(x)$.

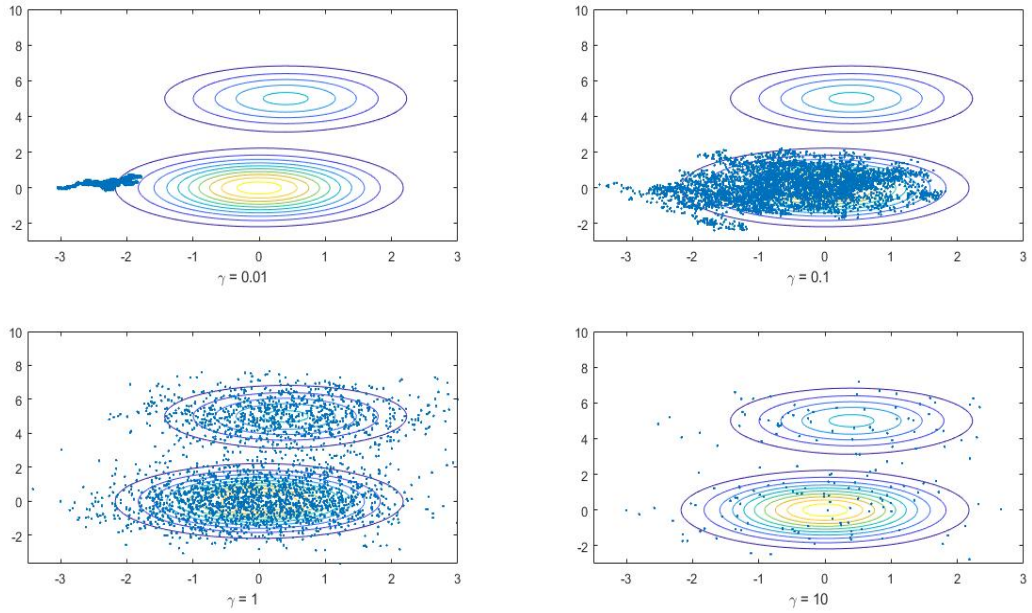


Figure 3: The outcome of the MH algorithm with different step sizes for target density $\pi_1(x)$ with initial value $x_0 = \begin{pmatrix} -3 \\ 0 \end{pmatrix}$ and K = 5000.

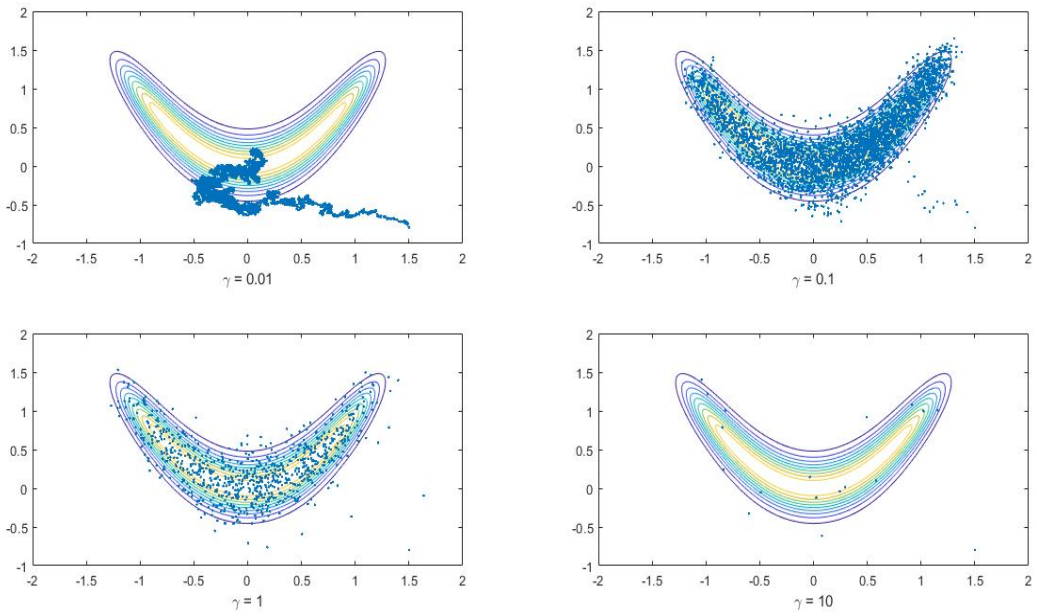


Figure 4: The outcome of the MH algorithm with different step sizes for target density $\pi_2(x)$ with initial value $x_0 = \begin{pmatrix} 1.5 \\ 0.8 \end{pmatrix}$ and K = 5000.

If we look at Figures 3 and 4 we see that the subplot with $\gamma = 0.01$ and $\gamma = 10$ gives us very bad coverage of the the target distribution. The second subplot with $\gamma = 0.1$, has the same problem here we see that in Figure 3 one of the peaks is not covered and in Figure 4 that the hands are not equally covered. Now in Figure 3 and 4 we see that at $\gamma = 1$ the whole target distribution is covered, so we can say this seems to be the best choice. Furthermore we see that there is a tail in the subplot for $\gamma = 0.01$ and $\gamma = 0.1$ in Figures 3 and 4, starting from the initial value x_0. This happens because the MH method needs a few iteration before it arrives at the region of high probability of the desired target distribution. We want to get rid of this tail, because the tail skews the results of the calculation of the mean in the direction of the initial value, because we have more weight there. We do a process called burn-in. Which states that we just remove a few of the accepted (or repeated) draws at the beginning of the MH method. So in our example we removed the first 500 draws. But this value is dependent on each individual target distribution and step size (So it's not set in stone).

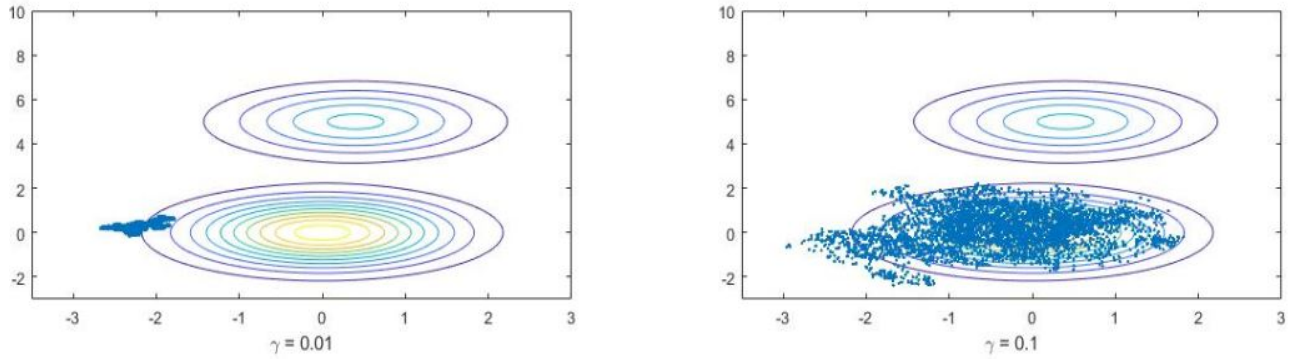


Figure 5: The outcome of the MH algorithm with different step sizes for target density $\pi_1(x)$ after burn-in.

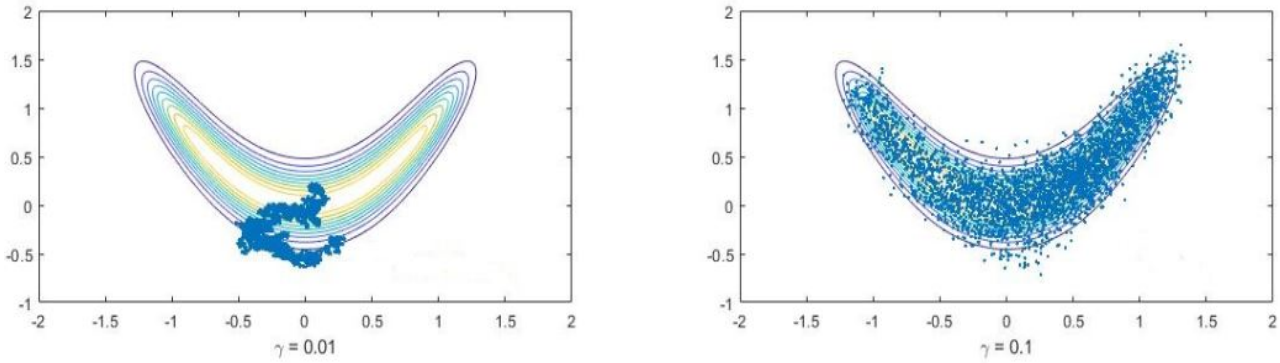


Figure 6: The outcome of the MH algorithm with different step sizes for target density $\pi_2(x)$ after burn-in.

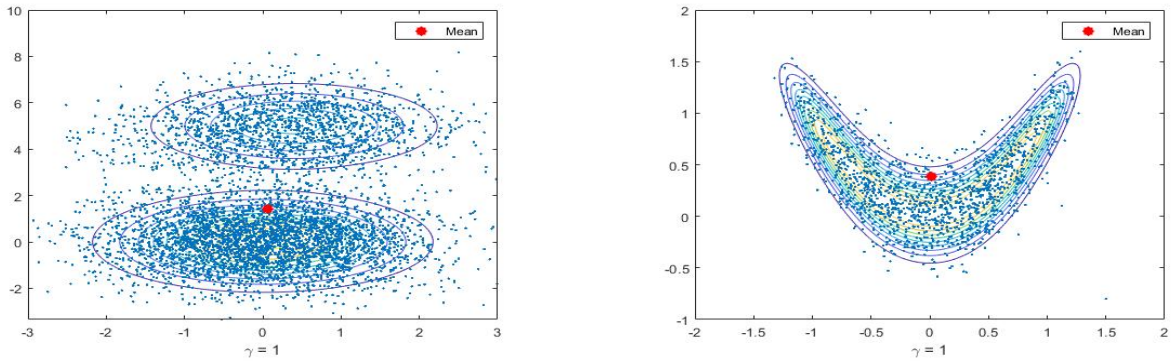
Now we have cut off the beginning of the chain to remove the burn-in phase of the chain and have gotten Figure 5 and 6. We see that the tail has been removed in all of the figures except in Figure 5a. We took a burn-in length of 500, for all of the other figures this was sufficient but for Figure 5a this was not. This can be attributed to the fact that the step size $\gamma = 0.01$ that we have chosen performs very bad. This has only been done for step sizes $\gamma = 0.1$ and $\gamma = 0.01$ because the other step sizes don't have a tail.

5.1. Finding the mean

Now we want to calculate the mean of a general example with target density $\pi(x)$. We know if we want to calculate the mean for our example we have the following formula with $X \sim \Pi(\cdot)$

$$\mathbb{E}(X) = \int_{\mathbb{R}^2} x\pi_1(x)dx.$$

Here we will show how we calculate the mean with the MH method for target distribution $\pi(x)$. We can approximate this integral with the mean of the sum of the draws $\left(\frac{1}{K}\sum_{k=1}^{K} x_k\right)$, because of Theorem 3.15. From Figure 3 and 4 we see that for step size $\gamma = 1$ the MH method makes a good approximation of the target distribution. That's why we will use $\gamma = 1$ as our step size to calculate the mean.



(a) Mean of target density $\pi_1(x)$.

(b) Mean of target density $\pi_2(x)$.

Figure 7: The mean for Example 1 and 2 with K = 5000 and $\gamma = 1$, calculated using method 2.

In Figure 7 we see a graphical representation of the mean of the MH method for $\gamma = 1$ for both target densities. We used two methods to find the mean. The first method we used to find the mean, was to run the code a few times while taking a larger number of iterations K. In our case we ran the code 5 times with $K = 100000$ and found that for the target density $\pi_1(x)$ the mean is varying in the x_1-coordinate between $[0.0907, 0.1235]$ and in the x_2-coordinate is varying between $[1.5121, 1.9311]$. Furthermore for target density $\pi_2(x)$ we did the same and found that the mean in the x_1-coordinate is varying between $[-0.0035, 0.0182]$ and in the x_2-coordinate is

varying between $[0.3817, 0.3913]$. The second method we used to get the mean was to run the code 100 times with K = 5000 and then take the average of the means that we got. After doing this we found for the target density $\pi_1(x)$ the mean in the x_1-direction was 0.1251 and in the x_2-direction 1.6341 and for target density $\pi_2(x)$ the mean in the x_1-direction was 0.0006 and in the x_2-direction 0.3849. Now we see that both methods are very close to each other, the values of Method 2 are in the intervals of Method 1 except for 0.1251. Both methods are very use full because they give you different ways of estimating the mean.

5.2. Contraction of the algorithm

We know that the MH method asymptotically converges to the target density, seen in Theorem 3.14 and 3.15, but we also want to get a good approximation on a finite sample. If the step size is too big we will reject almost everything, so we won't get a good approximation of our target density (it moves around inefficiently). But if our step size is too small we will accept almost everything and will need a lot of iteration before we can get a good approximation of our target density. This would also mean that the computational time will increase significantly which is precisely what we don't want. Furthermore, the accepted points are mostly packed around one specific part of the distribution, which is not very good, because we get poor coverage of the target distribution. This means if we want to calculate the mean it would give us a bad approximation, because all the points are clustered together and not spread evenly. So we want to find the best possible step size, such that we can get a relatively good amount of acceptance and with this a good approximation of our target density.

If we now look at Figure 3, we get the following acceptance percentage for each individual step size. For step size $\gamma = 0.01$ we get an 99% acceptance rate, at $\gamma = 0.1$ we get an 98% acceptance rate, at $\gamma = 1$ we have an acceptance rate of 56% and at $\gamma = 10$ we have an acceptance rate of 3%. And if we look at Figure 4 the acceptance ratio at $\gamma = 0.01$ is 96%, at $\gamma = 0.1$ we get an 80% acceptance rate, at $\gamma = 1$ we have an acceptance rate of 16% and at $\gamma = 10$ we have an acceptance rate of 0.2%

Furthermore we see in Figures 3 and 4, that for $\gamma = 0.01$ the MH method needs a lot of iteration before it arrives at the density. For $\gamma = 0.1$, we see that it covers a part of the support of the density, but not everything. At $\gamma = 1$, we see that the MH method covers the whole density. And for $\gamma = 10$ in figure 4 we see that it covers the target density good, but the problem is more that there aren't many points, which is what we want.

We see that the step size γ has a lot of influence on the MH method. To get a better understanding we will make trace plots of our parameters and look if it makes sense what is happening.

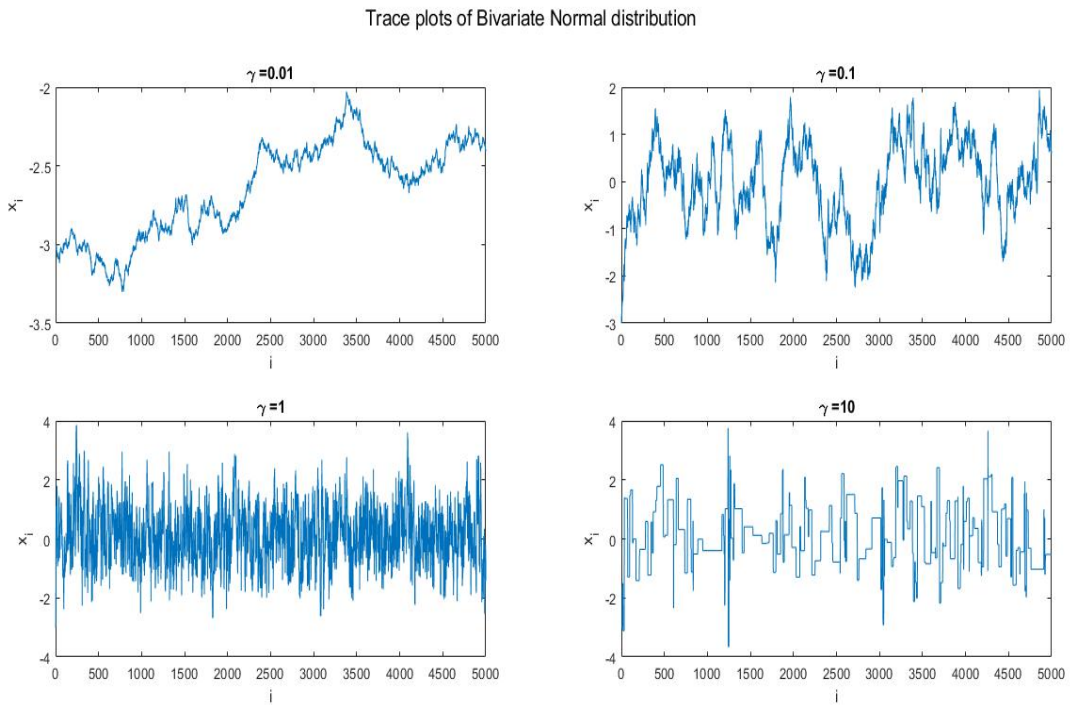


Figure 8: Trace plots with different step sizes of the target density $\pi_1(x)$ of the first component x_1.

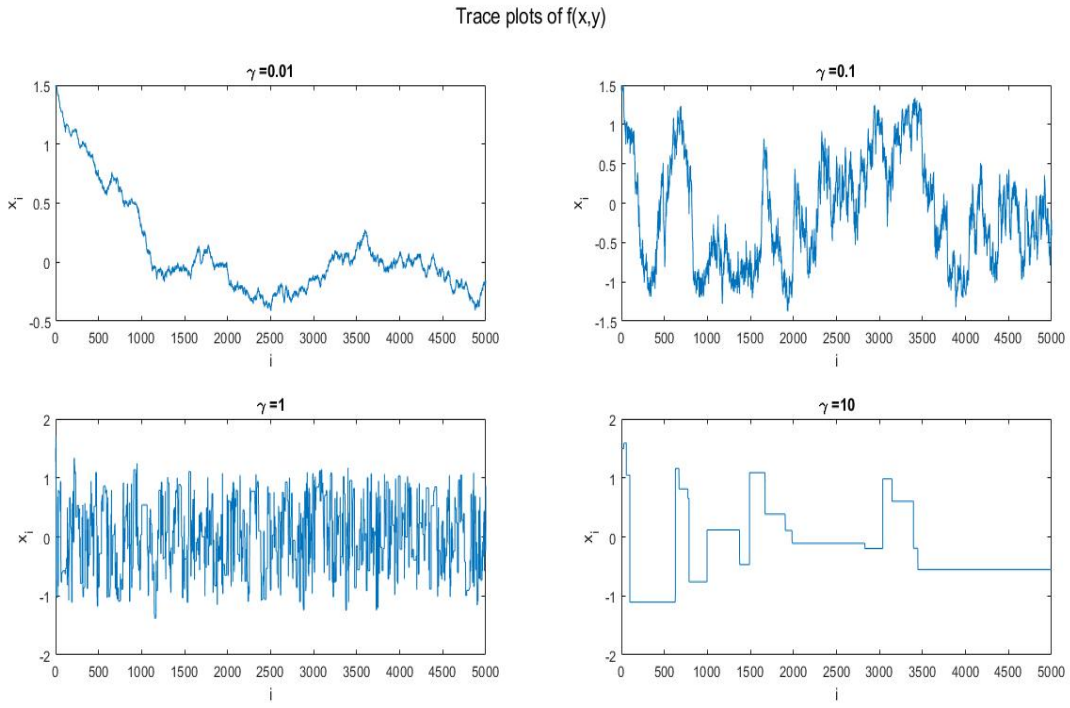


Figure 9: Trace plots with different step sizes of the target density $\pi_2(x)$ of the first component x_1.

In Figure 8 and 9 we have plotted the first 5000 accepted and possibly repeated draws of the first component x_1 of our MH method. Note that to get these trace plots we did another simulation of our algorithm so it does

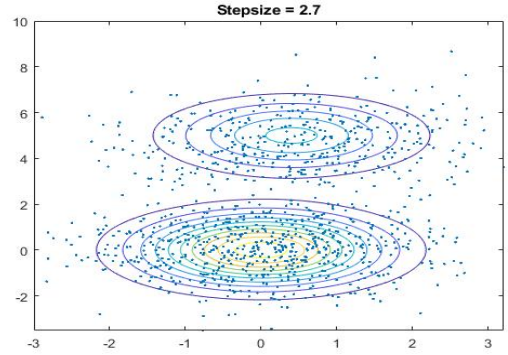
not precisely correspond to Figure 3 and 4. If we look at these figures we see for $\gamma = 0.01$ that the MH method goes rather slow through the parameter space, so it will accept a lot but won't cover a large part of the target density and here we also have a very high correlation .This means our candidate density is too narrow.

When $\gamma = 0.1$ we see that the method goes faster than for $\gamma = 0.01$, but it still moves very slowly through the parameter space. This means we still have a very high acceptance rate and a very high correlation.

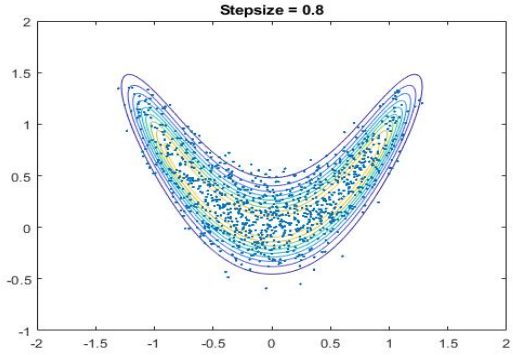
For $\gamma = 1$ in Figure 8 and 9 we see that the method goes very smoothly through the parameter space. This means we have a reasonable amount of accepted draws and reasonable correlation in our draws.

When $\gamma = 10$ the method gets stuck at a few values, so this means we reject a lot and we also have a very high correlation in the draws. Furthermore, we also get very few (separate) points, which we really want. So our candidate density is too wide.

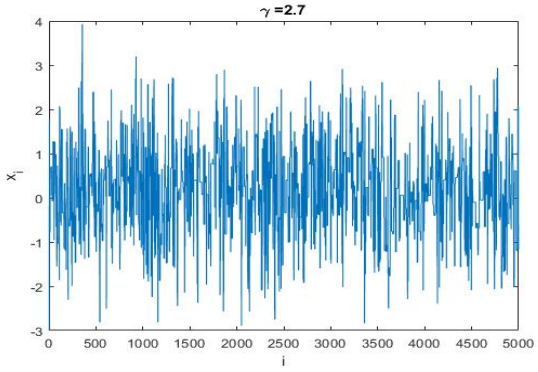
If we look at Figure 8 and 9, it would seem that $\gamma = 1$ would be the best choice, but we still want to know if we can do better. From articles [1] and [6] we find that at an 23% acceptance percentage we will have an accurate description of a target and candidate density which is Gaussian distributed. Furthermore we will also get reasonably low correlation in the draws. Note that this percentage is optimal if the dimension goes to infinity. In our example we have dimension equal to 2, but we still want to see if it gives us good coverage and whether the results we get are very far off, of what we already have. Furthermore we must note that this percentage is attained only for Gaussian target and candidate densities. This means that for Example 2 this percentage might not work so well. If we take this percentage in to consideration then we must take a step size $\gamma = 2.7$ for the target density $\pi_1(x)$ and a $\gamma = 0.8$ for target density $\pi_2(x)$. Then we get the following trace plots and plots for the outcome of the MH method.



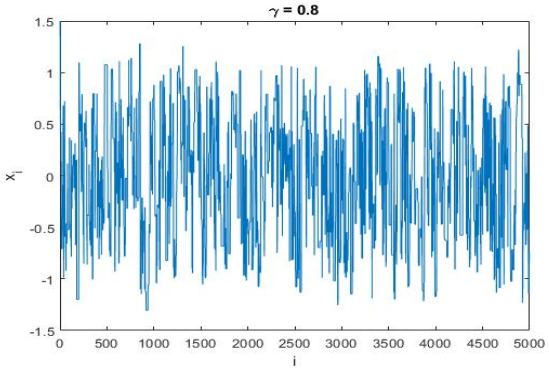
(a) Target density $\pi_1(x)$.



(b) Target density $\pi_2(x)$.



(c) Trace plot of target density $\pi_1(x)$.



(d) Trace plot of target density π_2.

Figure 10: Outcome of the MH method for target density $\pi_1(x)$ and $\pi_2(x)$.

In Figure 10 we see that the first component goes smoothly through the parameter space in both cases. In Figures 3 and 4 we saw that the best choice for γ was $\gamma = 1$ for both target distributions. So we see that we were not very far from the optimal γ. This means that if the trace plot looks like the one we saw at $\gamma = 1$ in Figure 8 and 9, then the chosen γ is very accurate.

6. Conclusion

Our objective was to find a way to easily calculate the mean of high dimensional densities by circumventing the problems of integration, so we then found the MH method which gave us samples from the distribution, which could then be used to approximate the mean. Theorem 3.14 shows that the Markov chains converge to the invariant distribution and Theorem 3.15 shows us how we can approximate integrals with a sum. We then use these theorems to construct the MH algorithm, this method gives us a practical way of approximating integrals using the samples. Then we introduced two examples, on which we used the MH algorithm to approximate their respective mean. We used two ways of calculating the mean with the MH method. The first method we used was by running our code 5 times while taking a larger number of iterations $K = 100000$. The second method we used was by running the code 100 times with $K = 5000$ and then taking the average of the results. Both methods gave us very similar approximations of the mean of the example target distributions.

If we have a look at the MH method we see that the candidate distribution q that we used in the examples has variance γ, this γ is also referred to as the step size. If we take the wrong step size the MH algorithms perform poorly. This can be seen in both examples. If we take a very big step size we will have a very low acceptance rate while if our step size is too small we will accept almost everything and will need a lot of iteration before we can get a good approximation of our target density.

The trace plots can give us an indication if we have chosen the right step size. In the trace plots, we can see how the first component x_1 from the sample behaves in its sample space. If the γ is too small it goes very slow through the sample space. Whereas if the γ is too large we will see that the same value is repeated constantly so here we see that it get stuck at certain values. Then we took a look at the 23% acceptance rate which we found in article [1] and was further expanded on in article [6]. This acceptance rate is the most ideal in the case of a Gaussian target, proposal density and the dimensions going to infinity. In our examples, we used a Gaussian proposal density. Furthermore we have only 2 dimensions but we still wanted to look if this accepted percentage gave us very different results. For Example 1 the we got a step size around $\gamma = 0.8$ and for Example 2 around $\gamma = 2.7$. These results showed us that they were not really far of our $\gamma = 1$, which we stated to be the best step size for our examples.

Further, we can also conclude that a high acceptance percentage does not necessarily mean we will get a good approximation of the target density, this is easily seen in the graphs we get for our examples. Small γ's give us high acceptance ratio's but do not really cover the whole area well of the example target distribution.

Now if we look at the correlation we see by choosing the wrong step size we will attain very high correlations. We want this to be as small as possible because a high auto-correlation means we will have to have more draws which means we will need more computational time. But we must stress that the MH method is not only dependent on the step size. If we take a wrong initial value for example it could take a very long time before the MH gives us an accurate description of our target density. This would mean that we would have a very high burn-in. That's why we should always try to find an initial value that is in the region of high probability of the target density. The choice of the candidate distribution also plays a role in our examples, we chose the Gaussian distribution, but there are also other possibilities that would have a lot of influence on the MH algorithm. So from all this, we can conclude that the MH algorithm can be used to efficiently sample from the example distributions, and with these samples, we get a good approximation of the mean.

7. Appendix

7.1. Code of the Multivariate Gaussian

The implementation of the MH method is done in a function at the end of this section.

```
% Multivariate Guassian

%Plots of the MH on g without burn-in
K = 5000; % Number of iterations

a = figure;
h1 = 0.1;
x = (-3:h1:3);
y = (-3:h1:10);
subplot(2,2,1)
[X,Y,Z,~] = g(x,y);
contour(X,Y,Z)
hold on;
[~,~,x1,acce1,~,~] = mp2(0.01,K);
b1 = x1;
a1_x = autocorr(x1(:,1),1);
plot(x1(:,1),x1(:,2),'.')
xlim([-3.5, 3])
xlabel('\gamma = 0.01');
hold on;
subplot(2,2,2)
[X,Y,Z,~] = g(x,y);
contour(X,Y,Z)
hold on;
[~,~,x1,acce2,~,~] = mp2(0.1,K);
m2_x =  mean(x1(:,1));
a2_x = autocorr(x1(:,1),1);
b2 = x1;
plot(x1(:,1),x1(:,2),'.')
xlim([-3.5, 3])
xlabel('\gamma = 0.1');
hold on;
subplot(2,2,3)
[X,Y,Z,~] = g(x,y);
contour(X,Y,Z)
hold on;
[~,~,x1,~,~,~] = mp2(1,K);
m3_x = mean(x1(:,1));
b3 = x1;
plot(x1(:,1),x1(:,2),'.')
xlim([-3.5, 3])
xlabel('\gamma = 1');
hold on;
subplot(2,2,4)
[X,Y,Z,~] = g(x,y);
contour(X,Y,Z)
hold on;
[~,~,x1,acce4,~,~] = mp2(10,K);
m4_x = mean(x1(:,1));
b4 = x1;
```

```
plot(x1(:,1),x1(:,2),'.')
xlim([-3.5, 3])
xlabel('\gamma = 10');
hold on;
figure(a)

%Plots of the MH on g with burn-in
K = 5000;
h1 = 0.1;
x = (-3:h1:3);
y = (-3:h1:10);
s = 500; %Burn-in
a44 = figure;
subplot(2,2,1)
[X,Y,Z,~] = g(x,y);
contour(X,Y,Z)
hold on;
plot(b1(s:end,1),b1(s:end,2),'.')
xlim([-3.5, 3])
xlabel('\gamma = 0.01');
hold on;
subplot(2,2,2)
[X,Y,Z,~] = g(x,y);
contour(X,Y,Z)
hold on;
plot(b2(s:end,1),b2(s:end,2),'.')
xlim([-3.5, 3])
xlabel('\gamma = 0.1');
hold on;
subplot(2,2,3)
[X,Y,Z,~] = g(x,y);
contour(X,Y,Z)
hold on;
plot(b3(s:end,1),b3(s:end,2),'.')
xlim([-3.5, 3])
xlabel('\gamma = 1');
hold on;
subplot(2,2,4)
[X,Y,Z,~] = g(x,y);
contour(X,Y,Z)
xlim([-3 3])
hold on;
plot(b4(s:end,1),b4(s:end,2),'.')
xlim([-3.5, 3])
xlabel('\gamma = 10');
hold on;
figure(a44)

%Plots of the MH on g with the mean at given stepsize.

a222 = figure;
K = 5000;
h1= 0.1;
x = -3:h1:3;
y = (-3:h1:10);
```

```
[X,Y,Z,~] = g(x,y);
contour(X,Y,Z)
hold on;
[~,~,x1,acce3,m_x,m_y] = mp2(1,K);
m3_x = mean(x1(:,1));
b3 = x1;
e = 500;
plot(x1(e:end,1),x1(e:end,2),'.')
xlim([-3 3])
xlabel('\gamma = 1');
hold on;
p1 = plot(m_x,m_y,'r*','linewidth',6);
legend(p1,'Mean')
figure(a222)

%% Calculating the mean by running the code 100 times
K = 5000;
for i = 1:100
    [~,~,~,~,m_x(i),m_y(i)] = mp2(1,K);
end
m_xx = mean(m_x);
m_yy = mean(m_y);
%% Calculating the mean by taking very high number of iterations
K=100000;
[~,~,~,~,m_x,m_y] = mp2(1,K);
%% Find the stepsize which has acceptance probability in the region of (23 +
    epsilon)%
ac = 0;
h = 0.05;
K = 5000;
eps = 0.7;
while abs(ac-23) >= eps
 [Xb1, Xb2, x,ac,~,~] = mp2(h,K);
  h = 0.05 + h;
end

q3 =figure;
h1 = 0.1;
x1 = (-3:h1:3);
y = (-3:h1:10);
[X,Y,Z,~] = g(x1,y);
contour(X,Y,Z)
hold on;
plot(Xb1,Xb2,'.')
title('Stepsize = '+string(h))
figure(q3)

%Traceplot of the ideal stepsize for g
d1 =figure;
plot(x(1:K,1))
xlabel('i')
ylabel('x_i')
title('\gamma ='+string(h))
figure(d1)

```

```
%% Traceplots of the MH on g
d = figure;
subplot(2,2,1)
h = 0.01;
[~,~,x] = mp2(h,K);
plot(x(1:K,1))
xlim ([0 K]);
title('\gamma ='+string(h))
xlabel('i')
ylabel('x_i')
subplot(2,2,2)
h = 0.1;
[~,~,x] = mp2(h,K);
plot(x(1:K,1))
xlim ([0 K]);
title('\gamma ='+string(h))
xlabel('i')
ylabel('x_i')
subplot(2,2,3)
h = 1;
[~,~,x] = mp2(h,K);
plot(x(1:K,1))
xlim ([0 K]);
title('\gamma ='+string(h))
xlabel('i')
ylabel('x_i')
subplot(2,2,4)
h = 10;
[~,~,x] = mp2(h,K);
plot(x(1:K,1))
xlim ([0 K]);
title('\gamma ='+string(h))
xlabel('i')
ylabel('x_i')
suptitle('Trace plots of Bivariate Normal distribution')
figure(d)



%% correlation

[Xb1 ,Xb2,x] = mp2(10,K);
Xb3=x(1:100);
autoc = autocorr(Xb3,1);

%% The 3-D plot of g
abc = figure;
h1 = 0.1;
x = (-3:h1:3);
y = (-3:h1:10);
[X,Y,Z,D] = g(x,y);
surf(X,Y,Z)
set(gca,'xticklabel',[])
set(gca,'yticklabel',[])
```

```
set(gca,'zticklabel',[])
figure(abc)



%% All of the functions that we need

function [X,Y,Z,D] = g(x,y) %Bivariate normal distribution
[X,Y] =meshgrid(x,y);
mu1 = [0 0];
sigma1 = eye(2);
mu2 = [0.4 5];
sigma2 = [1 0; 0 1];
ratio = 2./3;
A = [X(:) Y(:)];
D = ratio*mvnpdf(A,mu1,sigma1)+(1-ratio)*mvnpdf(A,mu2,sigma2) ;
Z = reshape(D,length(y),length(x));
end


%MH method for 2D (or more with a few tweaks)
function [Xb1, Xb2, x,ac,m_x,m_y] = mp2(h,K)
x = zeros(K,2); %vector in which we will store the accepted values
x(1,:)=[-3; 0]; %initial value
ac = 0;% counter accepted draws
for i = 2:K
    mu = [0 0];
    sigma = eye(2);
    [~,~,~,px] = g(x(i-1,1),x(i-1,2));
    w= mvnrnd(mu,(h^2)*sigma);
    y = x(i-1,:) + w;

    [~,~,~,py] = g(y(1),y(2));
    alpha = min(1,py/px );

    mu = unifrnd(0,1,1);
     if mu < alpha
         x(i,:) = y;
         ac = ac +1;
     else
        x(i,:) =x(i-1,:);
     end
end
ac = (ac/K)*100; % accepted percentage
bg = K*0.1; % Burn-in
Xb1 = x(bg:end,1); % first component
Xb2 = x(bg:end,2);% second component
m_x = mean(x(:,1)); % mean of first component
m_y = mean(x(:,2)); % mean of second component
end
```

7.2. Code of the Horse Shoe

The implementation of the MH method is done in a function at the end of this section.

```
% Horse Shoe
```

```
clear all;
K = 5000; % Number of iterations

%Traceplots of the MH on f (horse shoe)
d = figure;
subplot(2,2,1)
[~,~,x,acc1] = mp(0.01,K);
plot(x(:,1))
xlim ([0 5000]);
title('\gamma =0.01')
xlabel('i')
ylabel('x_i')
subplot(2,2,2)
[~,~,x,acc2] = mp(0.1,K);
plot(x(:,1))
xlim ([0 5000]);
title('\gamma =0.1')
xlabel('i')
ylabel('x_i')
subplot(2,2,3)
[~,~,x,acc3] = mp(1,K);
plot(x(:,1))
xlim ([0 5000]);
title('\gamma =1')
xlabel('i')
ylabel('x_i')
subplot(2,2,4)
[~,~,x,acc4] = mp(10,K);
plot(x(:,1))
xlim ([0 5000]);
title('\gamma =10')
xlabel('i')
ylabel('x_i')
suptitle('Trace plots of f(x,y)')
figure(d)

% Contour plot of f
a2 = figure;
h1 = 0.01;
x1 = -2:h1:2;
y1 = -1:h1:2;
[X,Y] =meshgrid(x1,y1);
Z = exp(-10*(X.^2-Y).^2-(Y-0.25).^4);
contour(X,Y,Z)
title('The contour plot of f(x,y)')
figure(a2)


%Plots of the MH on f without burn-in
a = figure;
subplot(2,2,1)
h1 = 0.01;
x1 = -2:h1:2;
y1 = -1:h1:2;
[X,Y] =meshgrid(x1,y1);
```

```
Z = exp(-10*(X.^2-Y).^2-(Y-0.25).^4);
contour(X,Y,Z)
hold on;
[~,~,x1] = mp(0.01,K);
plot(x1(:,1),x1(:,2),'.')
xlabel('\gamma = 0.01');
subplot(2,2,2)
contour(X,Y,Z)
hold on;
[~,~,x2] = mp(0.1,K);
plot(x2(:,1),x2(:,2),'.')
xlabel('\gamma = 0.1');
subplot(2,2,3)
contour(X,Y,Z)
hold on;
[~,~,x3] = mp(1,K);
plot(x3(:,1),x3(:,2),'.')
xlabel('\gamma = 1');
subplot(2,2,4)
contour(X,Y,Z)
hold on;
[~,~,x4] = mp(10,K);
plot(x4(:,1),x4(:,2),'.')
xlabel('\gamma = 10');
figure(a)


s = 500; %Burn-in
a = figure;
subplot(2,2,1)
contour(X,Y,Z)
hold on;
plot(x1(s:end,1),x1(s:end,2),'.')
xlabel('\gamma = 0.01');
subplot(2,2,2)
contour(X,Y,Z)
hold on;
plot(x2(s:end,1),x2(s:end,2),'.')
xlabel('\gamma = 0.1');
subplot(2,2,3)
contour(X,Y,Z)
hold on;
plot(x3(s:end,1),x3(s:end,2),'.')
xlabel('\gamma = 1');
subplot(2,2,4)
contour(X,Y,Z)
hold on;

plot(x4(s:end,1),x4(s:end,2),'.')
xlabel('\gamma = 10');
figure(a)


%% Correlation
h=1;
```

```
[Xb1 ,~] = mp(1,K);
Xb3=Xb1(1:500);
autoc = autocorr(Xb3,h);

%% Plot of the MH on f with the mean at given stepsize.
a2 = figure;
h1 = 0.01;
x1 = -2:h1:2;
y1 = -1:h1:2;
[X,Y] =meshgrid(x1,y1);
Z = exp(-10*(X.^2-Y).^2-(Y-0.25).^4);
contour(X,Y,Z)
hold on;
[Xb1,Xb2,~,~,mx,my] = mp(1,K);
plot(Xb1,Xb2,'.')
hold on;
p1 = plot(mx,my,'r*','linewidth',6);
xlim([-2 2])
ylim([-1 2])
legend(p1,'Mean')
xlabel('\gamma = 1');
figure(a2)
%% Calculating the mean by taking very high number of iterations
K = 10000;
[~,~,~,~,mx,my] = mp(1,K);
%% Calculating the mean by running the code 100 times
K = 5000;
for i = 1:100
    [~,~,~,~,mx(i),my(i)] = mp(1,K);
end
m_xx= mean(mx);
m_yy = mean(my);

%% Find the stepsize which has acceptance probability in the region of (23 +
    epsilon)%
ac = 0;
h = 0.05;
K = 5000;
eps = 1;
while abs(ac-23) >= eps
 [Xb1,Xb2,x,ac] = mp(h,K);
  h = 0.05 + h;
end

q3 =figure;
h1 = 0.01;
x1 = -2:h1:2;
y1 = -1:h1:2;
[X,Y] =meshgrid(x1,y1);
Z = exp(-10*(X.^2-Y).^2-(Y-0.25).^4);
contour(X,Y,Z)
hold on;
plot(Xb1,Xb2,'.')
title('Stepsize = '+string(h))
figure(q3)
```

```
166
167 q4 = figure;
168 plot(x(1:K,1))
169 xlim ([0 5000]);
170 title('\gamma = '+string(h))
171 xlabel('i')
172 ylabel('x_i')
173 figure(q4)
174 %% 3D-plot of the Horse shoe (f)
175 abc = figure;
176 h1 = 0.1;
177 x1 = -2:h1:2;
178 y1 = -1:h1:2;
179 [X,Y] =meshgrid(x1,y1);
180 Z = exp(-10*(X.^2-Y).^2-(Y-0.25).^4);
181 surf(X,Y,Z)
182 set(gca,'xticklabel',[])
183 set(gca,'yticklabel',[])
184 set(gca,'zticklabel',[])
185 figure(abc)
186 %% The MH method on the horse shoe
187 function [Xb1, Xb2, x,ac,mx,my] = mp(h,K)
188 x = zeros(K,2);
189 x(1,:)=[1.5; -0.8];
190 ac = 0;
191 for i = 2:K
192    mu = [0 0];
193    sigma = eye(2);
194    px = exp(-10*(x(i-1,1).^2-x(i-1,2)).^2-(x(i-1,2)-0.25).^4);
195    w= mvnrnd(mu,(h^2)*sigma);
196    y = x(i-1,:) + w;
197    py = exp(-10*(y(1).^2-y(2)).^2-(y(2)-0.25).^4);
198    alpha = min(1,py/px );
199    mu = unifrnd(0,1,1);
200     if mu < alpha
201         x(i,:) = y;
202         ac = ac +1;
203     else
204        x(i,:) =x(i-1,:);
205     end
206 end
207 ac = (ac/K)*100; % accepted percentage
208 bg = K*0.1; % Burn-in
209 Xb1 = x(bg:end,1); % first component
210 Xb2 = x(bg:end,2);% second component
211 mx = mean(x(:,1)); % mean of first component
212 my = mean(x(:,2)); % mean of second component
213 end
```

References

[1] Siddhartha Chib; Edward Greenberg. Understanding the metropolis-hastings algorithm. *The American Statistician*, 49(4):327–335, 1995.

[2] Gareth O. Roberts; Jeffrey S. Rosenthal. General state space markov chains and mcmc algorithms. *Probability Surveys*, 1:20–71, 2004.

[3] Jari Kaipio and Erkki Somersalo. *Statistical and Computational Inverse Problems*. Springer Science+Business Media, Inc, 233 Spring Street, New York, NY 10013, USA, 2004.

[4] George Casella Christian P. Robert. *Monte Carlo Statistical Methods*. Springer Science+Business Media, Inc, 233 Spring Street, New York, NY 10013, USA, 2004.

[5] J.S. Rosenthal. A review of asymptotic convergence for general state space markov chains. *Far East J. Theor. Stat.*, 5:37–50, 2001.

[6] G. O. Roberts, A. Gelman, and W. R. Gilks. Weak convergence and optimal scaling of random walk metropolis algorithms, 1994.