Stockholms universitet

An Introduction to Markov Chain Monte Carlo Methods within Bayesian Statistics

Tom Pedersen

Kandidatuppsats i matematisk statistik
Bachelor Thesis in Mathematical Statistics

Kandidatuppsats 2022:11
Matematisk statistik
Juni 2022

www.math.su.se

Matematisk statistik
Matematiska institutionen
Stockholms universitet
106 91 Stockholm

Stockholm University

Mathematical Statistics
Stockholm University
Bachelor Thesis **2022:11**
http://www.math.su.se

An Introduction to Markov Chain Monte Carlo Methods within Bayesian Statistics

Tom Pedersen*

June 2022

Abstract

In this thesis we compared two Markov Chain Monte Carlo algorithms; the Random Walk Metropolis algorithm and the Adaptive Metropolis Algorithm. The latter can be viewed as an extension of the former and our aim was to see if its efficiency is improved by the alteration. To do this, we defined a Bayesian model for logistic regression on a simulated data set. The posterior inference was based on samples from the Random Walk Metropolis and Adaptive Metropolis algorithms. We also introduced methods for assessing convergence of the chains. In our analysis, the Adaptive Metropolis algorithm proved more efficient than the Random Walk Metropolis algorithm.

*Postal address: Mathematical Statistics, Stockholm University, SE-106 91, Sweden. E-mail: pedersentom@me.com. Supervisor: Taras Bodnar, Michael Höhle.

Acknowledgements

I would like to express my gratitude to my supervisors Taras Bodnar and Michael Höhle at the Department of Mathematics at Stockholm University, who guided me throughout this project.

Contents

1 Introduction **4**

2 Logistic regression **6**
2.1 Definition of logistic regression 6
2.2 The likelihood equation for logistic regression 7
2.3 Bayesian logistic regression 8

3 Markov chain Monte Carlo methods **10**
3.1 Monte Carlo integration 10
3.2 Markov chain Monte Carlo 11
3.2.1 Markov chains 11
3.2.2 Metropolis chains 14
3.3 The Random Walk Metropolis Algorithm 15
3.4 The Adaptive Metropolis Algorithm 19

4 Assessing convergence of Markov chains **21**
4.1 Trace plots 21
4.2 The $\widehat{\boldsymbol{R}}$*-statistic* 21
4.3 The Effective Sample Size (ESS) 22
4.4 Rank plots 23

5 Results **24**
5.1 Visualizing the Random Walk and Adaptive Metropolis Algorithm 24
5.2 Comparing the Random Walk and Adaptive Metropolis Algorithm on a simulated data set 26
5.2.1 Simulated data 26
5.2.2 The model 27
5.2.3 MCMC results and diagnostics 28

6 Discussion **31**
6.1 Results 31
6.2 Summary 32

7 References **33**

1 Introduction

Bayesian statistics have found practical use in many areas of research such as the social and behavioural sciences, ecology, genetics and more. The specification of a Bayesian model begins with the choice of a statistical model that incorporate a set of statistical assumptions. The defining characteristics of a Bayesian statistical model is that the observed data and unobserved parameters are given a joint probability distribution with two key components: the *likelihood* and the *prior distribution*, which are combined using *Bayes' theorem* to form the *prior distribution*. The choice of a prior distribution is typically done before observing the data and can be based on previous studies or assumptions about the parameters. Once we've determined a prior distribution and observed the data our knowledge about the parameters are updated by combining the likelihood and prior information using Bayes' theorem, which results in the posterior distribution. The posterior distribution contains all the information about the unobserved parameters given our model and the observed data and is used for statistical inference.

The basis for Bayesian statistics was first described by Reverend Thomas Bayes in an essay on inverse probability in 1763. However, it was not until 50 or so years ago that Bayesian statistics became a viable option for more complex statistical models. The reason for this is that the posterior distribution is typically impossible to express in closed form and hence posterior inference must rely on numerical estimates or simulations of the posterior distribution, which is typically done with the aid of computers.

A class of methods called *Markov chain Monte Carlo* (MCMC) are able to draw samples from any prior distribution through simulation methods based on Markov chain theory. While these methods can produce a sequence of simulated draws that are (eventually) distributed according to the posterior distribution, they may only be guaranteed to do so in theory. Hence, in practice, where we are only able to produce a finite sequence of simulated draws, we have to ask our selves if the generated sequence is long enough to provide reliable estimates of the posterior distribution. This raises the following two questions; is our sequence long enough to provide reliable estimates and how do we construct *efficient* MCMC algorithms such that reliable estimates can be obtained within a reasonable time frame?

The aim of this thesis is to introduce the use of Markov Chain Monte Carlo methods for posterior sampling within a Bayesian model, with a particular focus on comparing two different sampling algorithms; the Random Walk Metropolis (RW) and Adaptive Metropolis (AM) algorithms. Since convergence is only guaranteed in theory (see, for example (Hill and Spall, 2019) for the RW algorithm and (Haario et al., 2001) for the AM algorithm) we make use of several diagnostic tools to assess the convergence of both algorithms. These include *trace plots*, the $\widehat{R}$*-statistic* and the *effective sample size*, which are common di-

agnostic tools for MCMC. We further introduce some improvements proposed by (Vehtari et al., 2021), which advise that *rank plots* could be used in place of trace plots and that some improvements can be made for the $\widehat{R}$*-statistic* and the effective sample size. These diagnostic tools further serve as a metric of efficiency and are used to compare the algorithms. We use both algorithms to sample from the posterior distribution of a logistic regression model which is defined in Section 2. The construction of the RW and AM algorithm is presented in Section 3 together with the necessary theory of Monte Carlo integration and Markov chains. Results are presented in Section 4 and discussed in Section 5. We find that the AM algorithm outperforms the RW algorithm in terms of greater efficiency with regards to the $\widehat{R}$*-statistic* and the effective sample size, and argue that this is because some of the posterior parameters are highly correlated. This serves as an example as to how improvements to MCMC sampling methods can be implemented, yielding more efficient algorithms.

2 Logistic regression

The theory on logistic regression where retrieved from the book written by (Agresti, 2013), chapter 4 and 5.

2.1 Definition of logistic regression

According to (Agresti, 2013) the most important model for categorical response data is *logistic regression*. For observations of a binary response Y and explanatory variables X, logistic regression models the probability π that Y is in either one of the two possible categories as a nonlinear function of X.
We will now define the logistic regression model for data consisting of N observations, a binary response variable Y that assumes the values 0 or 1, and M explanatory variables $\boldsymbol{X}_i = (X_{1i}, X_{2i}, ..., X_{Mi})$. First, define the probability function as

$$\pi(\boldsymbol{X}_i) = P(Y_i = 1|\boldsymbol{X}_i) = 1 - P(Y_i = 0|\boldsymbol{X}_i) \quad \text{for } i = 1, 2, ..., N.$$

Furthermore, let $\boldsymbol{\beta}$ be a $(m+1)$-dimensional vector of regression coefficients such that

$$\boldsymbol{\eta}_i = \boldsymbol{\beta}^T \boldsymbol{X}_i = \beta_0 + \beta_1 X_{1i} + \beta_2 X_{2i} +, ..., +\beta_M X_{Mi}$$

is the linear predictor for observation i. Since π is a probability it must be restrained to the interval $[0, 1]$ on the real line. Hence, if we wish to use the linear predictor in our model we must transform it so that it is restrained to $[0, 1]$ on the real line. The *logistic function* applied to the linear predictor achieves this. Consequently, the logistic regression model is defined as

$$\pi(\boldsymbol{X}_i) = \frac{\exp(\boldsymbol{\eta}_i)}{1 + \exp(\boldsymbol{\eta}_i)} \quad \text{for } i = 1, 2, ..., N. \tag{1}$$

To better understand logistic regression we begin with a definition of the *odds* for a success probability π, which is quoted verbatim from (Agresti, 2013) page 44.

Definition 2.1. For a probability π of success, the *odds* are defined to be

$$\text{odds } \Omega = \pi/(1 - \pi).$$

Now, if we compute the odds for probability (1) and apply the logarithm to both sides we obtain the following,

$$\log \frac{\pi(\boldsymbol{X}_i)}{1 - \pi(\boldsymbol{X}_i)} = \boldsymbol{\beta} \boldsymbol{X}_i,$$

which shows that the *log odds* of $Y = 1$ is a linear function of the explanatory variables.

2.2 The likelihood equation for logistic regression

The theory on the likelihood function (Section 2.2) and Bayesian logistic regression (Section 2.3) is retrieved from the book written by (Held and Bové, 2014).

Definition 2.2. For a statistical model parameterized by $\boldsymbol{\beta}$, the *likelihood function* $L(\boldsymbol{\beta})$ is the joint probability mass or density function of the observed data as a function of $\boldsymbol{\beta}$.

As in the previous section, assume that we have data consisting of N observations and lets further assume that the N binary responses are independent. Then, we may view every outcome y_i of Y_i as a realisation of an independent Bernoulli trial with success probability $\pi_i = \pi(\boldsymbol{X}_i)$, having probability mass function

$$f(y_i) = \pi_i^{y_i}(1-\pi_i)^{1-y_i}.$$

In the frequentist approach to statistical inference, the aim is to infer the values of the parameters in the model from the data. This can be done by maximizing the *likelihood function*, which is the probability mass or density function of the data as a function of the parameters. The values of the parameters that maximizes the likelihood function is called the *maximum likelihood estimate*, and it is the value for which the observed data has the highest probability given our model.
For logistic regression as defined in Section 1, the likelihood function $L(\boldsymbol{\beta})$ is given by

$$L(\boldsymbol{\beta}) = \prod_{i=1}^{N} f(y_i) = \prod_{i=1}^{N} \pi_i^{y_i}(1-\pi_i)^{(1-y_i)}.$$

The log-likelihood function $\ell(\beta)$ becomes a sum of the log-densities, i.e.

$$\begin{aligned}
\ell(\boldsymbol{\beta}) &= \sum_{i=1}^{N} \log\left(\pi_i^{y_i}(1-\pi_i)^{(1-y_i)}\right) \\
&= \sum_{i=1}^{N} y_i \log(\pi_i) + (1-y_i)\log(1-\pi_i) \\
&= \sum_{i=1}^{N} \log(1-\pi_i) + \sum_{i=1}^{N} y_i \log\left(\frac{\pi_i}{1-\pi_i}\right) \\
&= -\sum_{i=1}^{N} \log(1+\exp(\boldsymbol{\beta}_i)) + \sum_{i=1}^{N} y_i \boldsymbol{\beta}\boldsymbol{X}_i \ . \qquad (2)
\end{aligned}$$

According to (Agresti, 2013) page 192, the log-likelihood is used to fit the logistic regression model. In the next section, we will see that the likelihood function plays a central roll in the Bayesian approach to statistical inference as well.

2.3 Bayesian logistic regression

We will now extend logistic regression to a Bayesian model, for which we will need the following theorem.

Definition 2.3. (*Bayes' theorem*)
For any two events A and B with $P(B) > 0$, Bayes' theorem states that

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}.$$

In the Bayesian approach to statistical inference the parameters of the model are viewed as random and hence follow some probability distribution. Before we observe the data, we must select a *prior* distribution for the model parameters. Once the data are taken into account we apply Bayes' theorem in order to get the *posterior distribution*, which contains all available information about the model parameters given the observed data and the prior distribution.

Definition 2.4. *(The posterior distribution)*
Let $\boldsymbol{Y} = \boldsymbol{y}$ denote the vector of an observed realisation from a random vector $\boldsymbol{Y}$ with joint density function $f(\boldsymbol{y}|\boldsymbol{\theta})$, where $\boldsymbol{\theta}$ is a unknown parameter vector with parameter space Θ. By specifying a joint *prior* distribution $\pi(\boldsymbol{\theta})$ for the unknown parameter vector $\boldsymbol{\theta}$, we can compute the joint density function $p(\boldsymbol{\theta}|\boldsymbol{y})$ by an application of Bayes' theorem:

$$p(\boldsymbol{\theta}|\boldsymbol{y}) = \frac{f(\boldsymbol{y}|\boldsymbol{\theta})\pi(\boldsymbol{\theta})}{\int_\Theta f(\boldsymbol{y}|\boldsymbol{\theta})\pi(\boldsymbol{\theta})d\boldsymbol{\theta}} \ . \tag{3}$$

The density function p is called the posterior distribution for $\boldsymbol{\theta}$.

Remark. In the enumerator on the right hand side of the posterior distribution the term $f(\boldsymbol{y}|\boldsymbol{\theta})$ is the joint likelihood $L(\boldsymbol{\theta})$ for the data given $\boldsymbol{\theta}$. In the denominator, we marginalize the likelihood over $\boldsymbol{\theta}$ and, once data has been observed, we're left with some constant that ensures that the posterior density integrates to one. This implies that the joint posterior distribution is proportional to the product of the joint likelihood and joint prior distributions, which we write as $p(\boldsymbol{\theta}) \propto L(\boldsymbol{\theta})\pi(\boldsymbol{\theta})$.

With Bayesian inference we need to select a model that describes the data as well as specifying prior distributions for the unknown parameters of the model. One of the greatest challenges in Bayesian modelling is choosing an appropriate prior. Therefore we resort to using a default prior for logistic regression coefficients as proposed by (Gelman et al., 2014) on page 415. In order to use the default prior, the data are scaled in the following way:

1. Binary inputs are shifted to have a mean of 0 and differ by 1 in their lower and upper conditions;
2. Continuous inputs are set to have a mean of 0 and scaled to have a standard deviation of 0.5.

The default prior for the logistic regression coefficients on the scaled data are then set as independent Cauchy distributions with center 0 and scale 2.5. For the constant term (intercept) we set a Cauchy distribution with center 0 and scale 10.

Definition 2.5. *(Bayesian Logistic Regression with Default Cauchy Priors)* Under the same setting as with logistic regression where the data has been scaled as proposed by (Gelman et al., 2014), the Bayesian model with default Cauchy priors is defined by the following quantities:

1. *The posterior*
$$p(\boldsymbol{\beta}) = \frac{L(\boldsymbol{\beta}) \prod_{j=0}^{M} f_j(\beta_j)}{\int L(\boldsymbol{\beta}) \Pi_{j=0}^{M} f_j(\beta_j)\, d\beta};$$

2. *The likelihood*
$$L(\boldsymbol{\beta}) = \prod_{i=1}^{N} \pi_i^{y_i} (1-\pi_i)^{(1-y_i)};$$

3. *The prior distributions*
$f_0(\beta_0) \sim Cauchy(0, 10)$,
$f_j(\beta_j) \sim Cauchy(0, 2.5)$ for $j = 1, ..., M$.

For computational reasons, we may be interested in the log-posterior, which is

$$\log p(\boldsymbol{\beta}) \propto \ell(\boldsymbol{\beta}) + \sum_{j=0}^{M} \log f_j(\beta_j),$$

where $\ell(\boldsymbol{\beta})$ is equal to equation (2).

For statistical inference we are often interested in the posterior mean $E(\boldsymbol{\beta}|\boldsymbol{y})$, the posterior variance $\text{var}(\boldsymbol{\beta}|\boldsymbol{y})$ and the posterior standard deviation, which is the square root of the posterior variance. Furthermore, if we want to provide an interval for which the parameter of interest is contained with a certain probability, we can provide a *credible interval*. The following definition of a credible interval for a scalar parameter β is quoted essentially verbatim from (Held and Bové, 2014), page 172.

Definition 2.6. *(Credible intervals)*
For a fixed $\gamma \in (0, 1)$, a $\gamma \cdot 100\%$ *credible interval* is defined through two real numbers t_l and t_u, that fulfill

$$\int_{t_l}^{t_u} p(\beta|y) d\beta.$$

The quantity γ is called the *credible level* of the *credible interval* $[t_l, t_u]$.

3 Markov chain Monte Carlo methods

We recall that we have a statistical model parameterized by $\boldsymbol{\beta}$ and wish to conduct statistical inference using the joint posterior distribution

$$p(\boldsymbol{\beta}|\boldsymbol{y}) = \frac{L(\boldsymbol{y}|\boldsymbol{\beta})\pi(\boldsymbol{\beta})}{\int_{\mathcal{X}} L(\boldsymbol{y}|\boldsymbol{\beta})\pi(\boldsymbol{\beta})\, d\boldsymbol{\beta}}.$$

However, direct inference using the posterior distribution is typically not possible because the normalization constant in the denominator is only expressible as an analytically intractable integral. In circumstances where we can not determine the posterior distribution directly we may have to resort to methods of computer simulation. If we are able to generate sufficiently many simulated draws from the posterior these can be used for empirical estimates of the posterior (van de Schoot et al., 2021).

In what follows, we will introduce two simulation methods that only depend on the posterior distribution through the ratio

$$\frac{p(\cdot|\boldsymbol{y})}{p(\cdot|\boldsymbol{y})} = \frac{L(\boldsymbol{y}|\cdot)\pi(\cdot)}{L(\boldsymbol{y}|\cdot)\pi(\cdot)}.$$

Therefor, these methods of simulation allow for indirect posterior inference without the need to determine the normalizing constant. Before we present the methods that allow for posterior sampling in Section 3.2, we will introduce how the samples may be used for estimates of the posterior mean $E(\beta|\boldsymbol{y})$ and posterior variance $\text{Var}(\beta|\boldsymbol{y})$ for any scalar parameter β.

3.1 Monte Carlo integration

In order to compute estimates of posterior quantities like the posterior mean and posterior variance when we rely on samples from the posterior distribution, we may use *Monte Carlo integration*, which will now be defined. The theory on Monte Carlo integration is retrieved from the book (Held and Bové, 2014), chapter 8.

Definition 3.1. *(Monte Carlo integration)*
If we have simulated S independent draws $\beta^{(1)}, ..., \beta^{(S)}$ from the posterior distribution $p(\beta|\boldsymbol{y})$ of interest, then, for any suitable function g,

$$\widehat{E}(g(\beta)|\boldsymbol{y}) = \frac{1}{S}\sum_{s=1}^{S} g(\beta^{(s)}) \tag{4}$$

is the the *Monte Carlo estimate* of $E(g(\beta)|\boldsymbol{y})$ obtained through *Monte Carlo integration*.

Remark. If we set $g(x) = x$ in equation (4) we obtain the Monte Carlo estimate of the posterior mean and with $g(x) = x^2$ we obtain the Monte Carlo estimate

of $E(\beta^2|\boldsymbol{y})$. These estimates can be used to obtain the Monte Carlo estimate of the posterior variance, $\widehat{\text{Var}}(\beta|\boldsymbol{y}) = \widehat{E}(\beta^2|\boldsymbol{y}) - \widehat{E}(\beta|\boldsymbol{y})^2$.

With the law of large numbers, it can be shown that the Monte Carlo estimate of $E(g(\beta)|\boldsymbol{y})$ is a simulation-consistent, meaning that the estimate converges to $E(g(\beta)|\boldsymbol{y})$ as $S \to \infty$. Simulation-consistency may hold even if the samples $\beta^{(1)}, ..., \beta^{(S)}$ are not independent, but the accuracy of the Monte Carlo estimate is reduced if the samples are positively correlated.

Furthermore, if we want to use our samples to provide a symmetric 95% credible interval for any scalar estimand β, we may use the 2.5% and 97.5% quantiles of the ordered set of simulated draws (van de Schoot et al., 2021).

3.2 Markov chain Monte Carlo

There exist several ways to simulate draws from the posterior distribution but we will focus on a method that belongs to a class of techniques known as *Markov chain Monte Carlo*, abbreviated as MCMC. In order to use MCMC for posterior inference we construct a *Markov chain*, a sequence of random variables, such that the sequence will (at least eventually) represent draws from the posterior distribution. The defining characteristic of a Markov chain is the *Markov property*, which states that the conditional probability by which the sequence moves to the next state only depends on the current state of the sequence.

This section is mainly concerned with how to construct a Markov chain such that its *stationary distribution* is the posterior of interest. Since these methods are applicable outside of Bayesian statistics it is common to say that we want to sample from some *target distribution*, and henceforth we will use posterior- and target distribution interchangeably. Certain Markov chains will have a stationary distribution, meaning that its distribution remains the same as the sequence progresses. Our aim is to construct a Markov chain such that its stationary distribution is the target distribution. In order to do this we will consider *Metropolis chains*, which are Markov chains modified to have the target distribution as its stationary distribution. We begin by defining Markov chains and further present the theory necessary to prove that a Metropolis chain has the correct stationary distribution. The theory in Section 3.2.1 and 3.2.2 were retrieved from (Ross, 2019) and (Levin et al., 2017).

3.2.1 Markov chains

A *stochastic process* $\{X_t, t \in T\}$ is a sequence of random variables with indices from a *index set* T which takes on values in a set $\mathcal{X}$. The set $\mathcal{X}$ is called the *state space* of the process and the index set T is commonly interpreted as time. Hence, we say that the process is in state $x \in \mathcal{X}$ at time t whenever $X_t = x$.

We will consider a discrete-time process where $T = \{0, 1, 2, ...\}$ and where the state space $\mathcal{X}$ is finite.

A *Markov chain* is a discrete-time stochastic process that moves along the state space as follows: if the process is currently in state $x \in \mathcal{X}$ there exists some fixed probability $P(x, y)$ that it will make a one-step transition to state $y \in \mathcal{X}$ that only depends on the current state x. This is known as the *Markov property* and it may formally be expressed as

$$\begin{aligned} P(x, y) &= P(X_{t+1} = y | X_t = x, X_{t-1} = x_{t-1}, ..., X_1 = x_1, X_0 = x_0) \\ &= P(X_{t+1} = y | X_t = x), \end{aligned}$$

for all states $x, y \in \mathcal{X}$ and for all $t \geq 0$. We will refer to $P(x, y)$ as the *one-step probabilities* of the Markov chain.

For any Markov chain with a finite state space $\mathcal{X}$ we may construct a $|\mathcal{X}| \times |\mathcal{X}|$-dimensional matrix $\mathbf{P}$ with entries $\mathbf{P}(x, y) = P(x, y)$ for all $x, y \in \mathcal{X}$. The matrix $\mathbf{P}$ is called the *one-step transition matrix* and is constructed such that each row represents a probability distribution. Particularly, if the process is in state x at time t, row x in $\mathbf{P}$ is the probability distribution for X_t. Consequently, obeying the axioms of probability, each row in $\mathbf{P}$ must sum to one.

Now we will show that for any $t \in T$, the distribution of X_t can be found through matrix multiplication. If we let π_t denote the distribution of X_t we have that

$$\pi_t(x) = P(X_t = x) \quad \text{for all } x \in \mathcal{X}.$$

By conditioning on all previous states at time $t - 1$, we see that

$$\begin{aligned} \pi_t(y) &= P(X_t = y) \\ &= \sum_{x \in \mathcal{X}} P(X_t = y | X_{t-1} = x) P(X_{t-1} = x) \\ &= \sum_{x \in \mathcal{X}} P(X_{t-1} = x) \mathbf{P}(x, y) \\ &= \sum_{x \in \mathcal{X}} \pi_{t-1}(x) \mathbf{P}(x, y) \quad \text{for all } y \in \mathcal{X}, \end{aligned}$$

where the third equality follows from the Markov property. The preceding equation can be rewritten in vector form as

$$\pi_t = \pi_{t-1} \mathbf{P} \quad \text{for } t \geq 1.$$

Now that we have introduced Markov chains we will show how we can adapt

its transition matrix $\mathbf{P}$ such that the sequence generated by the Markov chain converges to any probability distribution π on $\mathcal{X}$. We begin with the following definition.

Definition 3.2. *(The limiting distribution)*
For a Markov chain $\{X_t, t \geq 0\}$ with a finite state space $\mathcal{X}$, the probability distribution π is called the *limiting distribution* of the Markov chain if

$$\pi_t(y) = \lim_{t \to \infty} P(X_t = y | X_0 = x)$$

for all $x, y \in \mathcal{X}$.

Definition 3.2 is concerned with the long-term behaviours of the Markov chain. We give the remark that limiting distributions do not always exists, but when they do, they are equal to the stationary distribution of the chain, which is defined next.

Definition 3.3. *(The stationary distribution)*
For a Markov chain with finite state space $\mathcal{X}$ and transition matrix $\mathbf{P}$, any probability distribution π on $\mathcal{X}$ that satisfies

$$\pi = \pi \mathbf{P}$$

is *stationary* for $\mathbf{P}$.

Now that we have defined the stationary distribution, we are ready to give the following important theorem which will be used to prove the result of the next section.

Theorem 1. *Consider a Markov chain with a finite state space $\mathcal{X}$, transition matrix $\mathbf{P}$ and let π be any probability distribution on $\mathcal{X}$. If π satisfy the following equations,*

$$\pi(x)\mathbf{P}(x, y) = \pi(y)\mathbf{P}(y, x), \tag{5}$$

for all $x, y \in \mathcal{X}$, then π is stationary for $\mathbf{P}$.

Proof. We sum both sides of equation 5 over all y in $\mathcal{X}$, and using that each row in $\mathbf{P}$ must sum to one, we obtain

$$\sum_{y \in \mathcal{X}} \pi(x)\mathbf{P}(x, y) = \sum_{y \in \mathcal{X}} \pi(y)\mathbf{P}(y, x) = \pi(x).$$

□

The equations (5) are known as the *detailed balance equations*.

3.2.2 Metropolis chains

Suppose that we have a Markov chain with finite state space $\mathcal{X}$ and wish to construct a transition matrix $\mathbf{P}$ such that it has a stationary distribution equal to some target distribution π on $\mathcal{X}$. To do so, let $\mathbf{Q}$ be a symmetric transition matrix on $\mathcal{X}$ satisfying that $\mathbf{Q}(x, y) = \mathbf{Q}(y, x)$ for all $x, y \in \mathcal{X}$. Now, we shall modify $\mathbf{Q}$ such that we obtain a Markov chain with stationary distribution π.

The central idea behind the Metropolis chain is to sequentially generate a sequence of random variables $Y_0, Y_1, Y_2, ...$ such that $P(Y_{t+1} = y_{t+1}|Y_t = y_t) = \mathbf{Q}(y_t, y_{t+1})$ for all $t \geq 0$ and $y_t, y_{t+1} \in \mathcal{X}$, forming a Markov chain with transition matrix $\mathbf{Q}$ and state space $\mathcal{X}$, and at each state correct the chain such that its stationary distribution is the target distribution π. In order for the corrected chain to have the correct stationary distribution we introduce a new transition matrix $\mathbf{P}$ on $\mathcal{X}$, defined as

$$\mathbf{P}(x, y) = \begin{cases} \mathbf{Q}(x, y)\alpha(x, y) & \text{if } y \neq x \\ 1 - \sum_{z:z\neq x} \mathbf{Q}(x, z)\alpha(x, z) & \text{if } y = x \end{cases}, \tag{6}$$

$$\text{where } \alpha(x, y) = \min\left(1, \tfrac{\pi(y)}{\pi(x)}\right),$$

for all $x, y \in \mathcal{X}$. Since the transition probabilities defined by $\mathbf{P}$ only depend on the current state x, the Markov property is preserved. That it also has stationary distribution π is confirmed by the following proposition.

Proposition 1. *Given a Markov chain with symmetric transition matrix* $\mathbf{Q}$ *and finite state space* $\mathcal{X}$*, and for any distribution* π *on* $\mathcal{X}$*, the transition matrix* $\mathbf{P}$ *as in* (6) *has stationary distribution* π*.*

Proof. The proposition holds if $\mathbf{P}$ and π satisfy the *detailed balance equations* (5). To show this we have to consider the two cases where either

$$\pi(y) \leq \pi(x) \quad \text{or} \quad \pi(y) > \pi(x).$$

We begin with the former case. If $x \neq y$ and $\pi(y) \leq \pi(x)$ then $\alpha(x, y) = \pi(y)/\pi(x)$ and $\alpha(y, x) = 1$ by definition. Using that $\mathbf{Q}$ is symmetric, it follows that

$$\begin{aligned} \pi(x)\mathbf{P}(x, y) &= \pi(x)\mathbf{Q}(x, y)\alpha(x, y) \\ &= \pi(x)\mathbf{Q}(y, x)\tfrac{\pi(y)}{\pi(x)} \\ &= \pi(y)\mathbf{Q}(y, x)\alpha(y, x) = \pi(y)\mathbf{P}(y, x). \end{aligned}$$

The second case is shown analogously and is presented for completeness. If $x \neq y$ and $\pi(y) > \pi(x)$ then $\alpha(x, y) = 1$ and $\alpha(y, x) = \pi(x)/\pi(y)$. Again, using that $\mathbf{Q}$ is symmetric, it follows that

$$\begin{aligned}\pi(x)\mathbf{P}(x,y) &= \pi(x)\mathbf{Q}(x,y)\alpha(x,y) \\ &= \pi(x)\mathbf{Q}(y,x)\tfrac{q(y)}{q(y)} \\ &= \pi(y)\mathbf{Q}(y,x)\alpha(y,x) = \pi(y)\mathbf{P}(y,x).\end{aligned}$$

Since π and $\mathbf{P}$ satisfy the detailed balance equations it follows that π is the stationary distribution of $\mathbf{P}$. □

According to (Ross, 2019), page 262, a sufficient condition that the stationary distribution of the Metropolis chain also is the limiting distribution is that $\mathbf{P}(x,x) > 0$ for some x. Hence, if a sufficiently long chain is generated where $\mathbf{P}$ satisfies the previous condition, it will correspond to simulated draws of the target distribution.

3.3 The Random Walk Metropolis Algorithm

The method of sampling from a distribution π on a finite state space $\mathcal{X}$ with Metropolis chains can be extended to the continuous case where the underlying Markov chain is replaced with a *Markov process* on a continuous state space Θ, the transition matrix is replaced by a *transition kernel*, and the target distribution p can be *any* probability density function on Θ. For a comprehensive review on the theory of convergence and stationarity in the continuous case, we refer the reader to (Hill and Spall, 2019).

We will now present how the previous method of sampling from a target distribution can be used within Bayesian analysis to sample from a posterior distribution that cannot be determined analytically. The methods we will present are the *Random Walk Metropolis* (RW) and *Adaptive Metropolis* (AM) algorithms. Remember that our aim is to derive inference about a statistical model based on the posterior distribution p of the model parameters $\boldsymbol{\beta}$, using the likelihood function and a prior distribution on the parameters.

The Random Walk Metropolis algorithm is presented with the notation consistent with (Gelman et al., 2014) page 278. Suppose that we are interested in generating samples from the joint posterior distribution

$$p(\boldsymbol{\beta}|\boldsymbol{y}) = \frac{L(\boldsymbol{y}|\boldsymbol{\beta})f(\boldsymbol{\beta})}{\int_\Theta L(\boldsymbol{y})f(\boldsymbol{\beta})\, d\boldsymbol{\beta}} \tag{7}$$

where $L(\boldsymbol{y}|\boldsymbol{\beta})$ is the joint likelihood-function and $f(\boldsymbol{\beta})$ is the joint prior distribution function for $\boldsymbol{\beta}$.
For a given starting value $\boldsymbol{\beta}_0$ the RW algorithm proceeds by iteratively proposing a new value for $\boldsymbol{\beta}^*$ at step t, which is drawn from a symmetric proposal distribution $q(\cdot|\boldsymbol{\beta}^{(t-1)})$. The proposal is either accepted or rejected based on the ratio r of the following densities

$$r = \frac{p(\boldsymbol{\beta}^*)|\boldsymbol{y})}{p(\boldsymbol{\beta}^{(t-1)}|\boldsymbol{y})}.$$

If the proposed value $\boldsymbol{\beta}^*$ is in a higher density region of the posterior distribution than the current value $\boldsymbol{\beta}^{(t-1)}$ the ratio r will be greater than 1 and the proposal is accepted. In this case, we will always move to a region with a higher posterior density. If the ratio is less than 1, implying that $\boldsymbol{\beta}^{(t-1)}$ is in a higher density region than the proposal, the proposed value $\boldsymbol{\beta}^*$ is either accepted with probability r or rejected with probability $1 - r$. If the proposal is accepted the chain moves to a region with a lower posterior density. If the proposal is rejected, we set $\boldsymbol{\beta}^{(t)} = \boldsymbol{\beta}^{(t-1)}$.

Since r is the ratio of the posterior distribution, the normalizing constant $1/\int_\Theta L(\boldsymbol{y}|)f(\boldsymbol{\beta})\, d\boldsymbol{\beta}$ will cancel, from which it follows that

$$r = \frac{p(\boldsymbol{\beta}^*|\boldsymbol{y})}{p(\boldsymbol{\beta}^{(t-1)}|\boldsymbol{y})} = \frac{L(\boldsymbol{\beta}^*|\boldsymbol{y}) \cdot f(\boldsymbol{\beta}^*)}{L(\boldsymbol{\beta}^{(t-1)}|\boldsymbol{y}) \cdot f(\boldsymbol{\beta}^{(t-1)}))}.$$

From this it follows that for a sufficiently large value of t such that the chain has reached its stationary distribution, we can draw samples from the posterior distribution by only knowing the likelihood-function and the prior distribution function. We summarise the steps in the RW algorithm below.

The Random Walk Metropolis Algorithm

For $(t = 1, ..., T)$, given a starting value $\boldsymbol{\beta}^{(0)}$, repeat:

1. Draw a *proposal* $\boldsymbol{\beta}^*$ from $q(\cdot|\boldsymbol{\beta}^{(t-1)})$

2. Compute the ratio

$$r = \frac{L(\boldsymbol{\beta}^*) \cdot f(\boldsymbol{\beta}^*)}{L(\boldsymbol{\beta}^{(t-1)}) \cdot f(\boldsymbol{\beta}^{(t-1)}))}$$

3. Set

$$\boldsymbol{\beta}^{(t)} = \begin{cases} \boldsymbol{\beta}^* \text{ with probability } \min(1, r) \\ \boldsymbol{\beta}^{(t-1)} \text{ otherwise.} \end{cases}$$

One problem that arises is how to select the the proposal distribution q. In (Gelman et al., 2014) page 296, it is suggested that a common approach is to set

$$q(\boldsymbol{\beta}^*|\boldsymbol{\beta}^{(t-1)}) = N(\boldsymbol{\beta}^*|\boldsymbol{\beta}^{(t-1)}, \sigma\Sigma), \tag{8}$$

i.e. a multivariate Gaussian distribution with mean equal to $\boldsymbol{\beta}^{(t-1)}$ and covariance matrix Σ scaled by a tuning parameter σ. The proposal distribution is hence centered around the current value $\boldsymbol{\beta}^{(t-1)}$ and fulfills the symmetric condition on q.
With q as in (8) the specification of the proposal distribution comes down to specification of Σ and σ. One way to determine these parameters are to consider the proportion of jumps that are accepted. In (Gelman et al., 2014), page 296, it is suggested that an optimal acceptance rate for the Metropolis algorithm with a Gaussian proposal distribution centered at the current value of sequence is about 0.44 in one dimension and around 0.23 in higher dimensions.

When implementing the Random Walk Metropolis algorithm we utilize several concepts presented in Appendix C.3 in (Gelman et al., 2014). Firstly, we work on the log-scale to avoid computational underflow (or overflow) when multiplying many factors. This entails specifying the unnormalized posterior as the sum of the log-likelihood and the log-prior density function. Furthermore, this means that the ratio r of the two densities will be computed as the difference of the log-densities. Step 3 is implemented by generating a point u from the uniform distribution $U(0, 1)$ and comparing it to the ratio r.

Pseudo-code for implementing the Random Walk Metropolis algorithm is presented below.

symbol	**description**
α	the acceptance rate
β	the parameter we wish to sample
σ	a the tuning parameter that scales the proposal distribution
T	the number of iterations
p	the unnormalized posterior distribution
C_t	the estimated covariance matrix at iteration t

Table 1: Description of the symbols used in Algorithm 1 and 2.

Algorithm 1: Random Walk Metropolis

$\alpha \leftarrow 0$
$\beta^{(1)} \leftarrow \beta_0$
for $t = 2, 3, ..., T$ **do**
    - generate a proposal $\beta^* \sim N(\beta^{(t-1)}, \sigma I)$
    - compute $r = \log(p(\beta^*)) - \log(p(\beta^{(t-1)}))$
    - generate a random number $u \sim U(0, 1)$
    if $u \leq \min(1, \exp(r))$ **then**
        $\beta^{(t)} \leftarrow \beta^*$
        $\alpha \leftarrow \alpha + 1$
        else
            $\beta^{(t)} \leftarrow \beta^{(t-1)}$

$\alpha \leftarrow \alpha / T$
return β, α

3.4 The Adaptive Metropolis Algorithm

According to (Haario et al., 2001) it is essential to choose an effective proposal distribution for the Random Walk Metropolis algorithm. In the previous section we argued that the effectiveness of the Random Walk Metropolis algorithm is highly effected by the acceptance rate, which is tuned by scaling the proposal distributions appropriately. Rather than tuning the algorithm *ad hoc* by running it several times until an optimal acceptance rate has been achieved, one may consider an algorithm that iteratively uses the accumulated samples to approximate the covariance structure of the target distribution and adjust the proposal distribution accordingly. One such algorithm is the Adaptive Metropolis Algorithm which was introduced by (Haario et al., 2001). Other than adjusting the proposal distribution the Adaptive Metropolis algorithm is identical to the Random Walk Metropolis algorithm.

For the proposal distribution $q(\cdot|\boldsymbol{X}_1, ..., \boldsymbol{X}_{t-1})$ in the adaptive Metropolis algorithm, (Haario et al., 2001) proposes a multivariate normal distribution centered at the current point $\boldsymbol{X}_{t-1}$ and covariance matrix

$$C_t = \begin{cases} C_0, & t = t_0 \\ s_d(\text{cov}(\boldsymbol{X}_1, ..., \boldsymbol{X}_{t-1}) + \varepsilon I_d), & t > t_0 \end{cases}.$$

Here, s_d is a scaling parameter that only depends on the dimension d of the parameter $\boldsymbol{\beta}$ of interest, $\varepsilon > 0$ is a constant that may be chosen very small and I_d is the d-dimensional identity matrix. The quantity εI_d is added to ensure that C_t will not become singular (Haario et al. (2001)). Notice that we let the covariance of the proposal distribution to depend on all the previous samples up until time t. This allows the proposal distribution to adapt to the covariate structure of the target distribution. One caveat is that the proposal distribution depends on the whole history of the chain up until $t-1$ and hence the Markov property is lost. Even without the Markov property, the chain as defined by the Adaptive Metropolis algorithm is shown to have the correct stationary distribution by (Haario et al., 2001).

For computational efficiency, (Haario et al., 2001) suggest using the following recursive calculation of the covariance matrix C_t,

$$C_{t+1} = \frac{t-1}{t} C_t + \frac{s_d}{t}(t\bar{\boldsymbol{X}}_{t-1}\bar{\boldsymbol{X}}_{t-1}^T - (t+1)\bar{\boldsymbol{X}}_t\bar{\boldsymbol{X}}_t^T + \boldsymbol{X}_t\boldsymbol{X}_t^T + \varepsilon I_d), \quad (9)$$

where $\bar{\boldsymbol{X}}_t$ denotes the mean of accumulated samples up until time t and $\bar{\boldsymbol{X}}_t$ is the vector transpose of $\bar{\boldsymbol{X}}_t$. Pseudo-code for the Adaptive Metropolis algorithm is presented below.

Algorithm 2: Adaptive Metropolis

$\alpha \leftarrow 0$
$\beta^{(1)} \leftarrow \beta_0$
for $t = 2, 3, ..., T$ **do**
    - compute covariance matrix C_t using $\beta^{(1)}, ..., \beta^{(t-1)}$
    - generate a proposal $\beta^* \sim N(\beta^{(t-1)}, C_{t-1})$
    - compute $r = \log(p(\beta^*)) - \log(p(\beta^{(t-1)}))$
    - generate a random number $u \sim U(0, 1)$
    if $u \leq \min(1, \exp(r))$ **then**
        $\beta^{(t)} \leftarrow \beta^*$
        $\alpha \leftarrow \alpha + 1$
        else
            $\beta^{(t)} \leftarrow \beta^{(t-1)}$

$\alpha \leftarrow \alpha / T$
return β, α

4 Assessing convergence of Markov chains

A concept common to chains generated by Markov chain Monte Carlo methods are how well they are *mixing*, which refers to how well the chain is exploring the support of the target distribution and thus how representative the samples are. When we initialize a chain it may take many iterations before the chain has reached its stationary distribution. Samples that were drawn during this initial phase are often disregarded since they don't represent the target distribution, and keeping with the conventions of (Gelman et al., 2014) we say that these samples represent the *warm-up period*.

In the previous section we introduced two MCMC algorithms whose stationary distributions can be shown to converge to the target distribution. However, in practical application we will only have a finite amount of samples for which the theoretical results may not hold. In the article written by (van de Schoot et al., 2021) three commonly used methods for assessing convergence of Markov chains are presented. These are *trace plots*, the $\widehat{R}$*-statistic* and the *effective sample size*. However, as Vats and Jones noted on page 701 in (Vehtari et al., 2021): "Diagnostics based on the simulated values cannot *prove* that the simulation is providing representative samples, and the best we can hope for is that it indicates when a problems has occurred."

A common approach to check for poor mixing is to run multiple independent chains with various starting points and check whether their distribution is similar. From now on we assume that we have ran m independent chains and that the samples form the warm-up period have been disregarded.

4.1 Trace plots

With trace plots we visualize the path of the chain by plotting the simulated values on the y-axis against the iteration number on the x-axis. For m independent chains drawn in the same plot, the trace plot can be used as a qualitative diagnostic regarding poor mixing both within and between chains.

4.2 The $\widehat{R}$-*statistic*

The $\widehat{R}$*-statistic* is a quantitative measure of how well the chains are mixing. Following the definition in (Gelman et al., 2014), the m chains are halved giving a total of $M = 2m$ chains. Suppose that each chain contains N samples and let β_{ij} denote the ith sample from the Mth chain ($i = 1, ..., n; j = 1, .., M$) of a scalar estimand β.
Now, let

$$\bar{\beta}_{.j} = \frac{1}{N}\sum_{i=1}^{N}\beta_{ij} \quad \text{and} \quad \bar{\beta}_{..} = \frac{1}{M}\sum_{j=1}^{M}\beta_{.j}$$

denote the *within-sequence means* and *total (pooled) mean* of any scalar estimand β, respectively. Then we may express the *between-chain variances* B and the *within-chain variances* W as

$$B = \frac{N}{M-1}\sum_{j=1}^{M}(\bar{\beta}_{\cdot j} - \bar{\beta}_{\cdot\cdot})^2 \quad \text{and} \quad W = \frac{1}{N-1}\sum_{i=1}^{N}(\beta_{ij} - \bar{\beta}_{\cdot j})^2.$$

An estimate for the posterior variance $\text{var}(\beta|y)$ is given by

$$\widehat{\text{var}}^{+}(\beta|y) = \frac{n-1}{n}W + \frac{1}{n}B,$$

which is unbiased under stationarity (Gelman et al., 2014), page 284. The $\widehat{R}$*-statistic* is then defined as

$$\widehat{R} = \sqrt{\frac{\widehat{\text{var}}^{+}(\beta|y)}{W}}. \tag{10}$$

At convergence the within and between chain variances should be equal and, consequently, $\widehat{R} \approx 1$ indicates that the chains have converged. In (Vehtari et al., 2021) the authors present what they call the *improved-*$\widehat{R}$, which is defined in the same way as $\widehat{R}$ but computed on the rank normalized samples. This new version of $\widehat{R}$ is defined even when either the posterior mean, variance or both are undefined. The authors assert that a value of $\widehat{R}$ greater than 1.01 may be a sign of poor mixing.

4.3 The Effective Sample Size (ESS)

As noted in Section 3.1, the Monte Carlo estimates are less efficient when computed on samples with positive correlation. With the *effective sample size* we try to estimate how many independent draws our dependent draws correspond to. For the effective sample size $\hat{n}_{\text{eff}}$ we will use the definition in (Gelman et al., 2014), page 286.
For m chains with n samples each, representing the draws of some scalar parameter β, the *effective sample size* $\hat{n}_{\text{eff}}$ is defined as

$$\hat{n}_{\text{eff}} = \frac{mn}{1 + 2\Sigma_{t=1}^{T}\hat{\rho}_t}, \tag{11}$$

where $\hat{\rho}_t$ is the estimated autocorrelations at lag t and T is the first odd positive integer for which $\hat{\rho}_{T+1} - \hat{\rho}_{T+2}$ is negative. As noted by (Vehtari et al., 2021), the effective sample size in Definition 11 should only be thought of as being the approximate number of independent samples for the bulk of the distribution. Furthermore, the authors recommend that effective sample size should be at least 100 per chains used.

4.4 Rank plots

Rank plots where suggested by (Vehtari et al., 2021) as an improvement to trace plots and are histogram of the sample ranks rather than their actual value. The ranks are computed on the pooled sample of all chains, and are thereafter displayed in histograms where the sample ranks are again grouped by which chain they belong to. Similarity between all rank plots indicate that the chains have mixed well.

5 Results

5.1 Visualizing the Random Walk and Adaptive Metropolis Algorithm

In this section we will use the Random Walk Metropolis Algorithm as well as the Adaptive Metropolis Hastings Algorithm to sample from a known target distribution. As mentioned in Section 3.3 the efficiency of the Random Walk Metropolis Algorithm is directly determined by the acceptance rate, which is tuned by modifying the scale of the proposal distribution.
In both examples the target distribution is a bivariate normal distribution with mean vector μ and covariance matrix Σ set to

$$\mu = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad \text{and} \quad \Sigma = \begin{pmatrix} 1 & 0.8 \\ 0.8 & 1 \end{pmatrix}. \tag{12}$$

The Random Walk Metropolis Algorithm was implemented in **R** by following the pseudo-code of Algorithm 1. The proposal distribution was set to a bivariate normal distribution centered at the previous draw, with independent components and scaled by a tuning parameter σ.

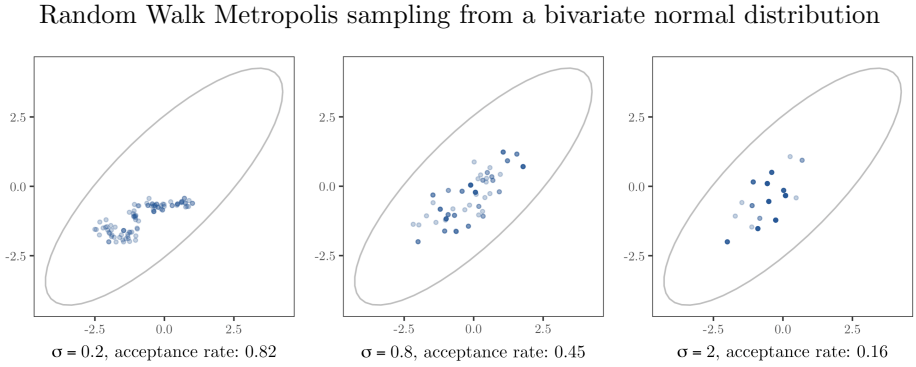

Figure 1: The Random walk Metropolis was used to draw samples from a know bivariate normal distribution with parameters as in (12). The ellipses are the 95% probability regions of the target distribution. A total of 100 samples were drawn with three different settings of the tuning parameter σ.

Figure 1 visualizes how the random walk Metropolis algorithm behaves for different settings of the tuning parameter σ. With a high acceptance rate *(left)* the algorithm explores the target distribution through small steps, yielding inefficient sampling. With an acceptance rate of 0.45 *(middle)*, close to the *optimal* rate of 0.44, the algorithm explores the target distribution more efficiently. If the scale of the proposal distribution is too high *(right)* many proposals will be within a low density region of the target distribution and hence are unlikely to be accepted, thus yielding inefficient sampling.
Now we use the Adaptive Metropolis algorithm to sample from the same target

distribution as in the previous example. The shape of the proposal distribution is shown for three stages of the chain, giving some insight into how the algorithm adapts to the covariate structure of the target distribution.

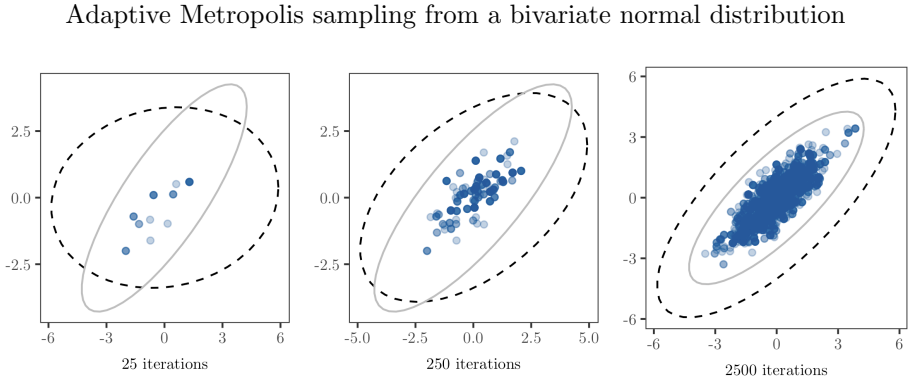

Figure 2: The AM algorithm was used to draw samples (blue points) from a know target distribution (a bivariate normal distribution with highly correlated components). The solid ellipse is the 95% probability region of the target distribution. The dashed ellipse is the 95% probability region of the proposal distribution. As the algorithm progresses, the proposal distribution adapts to the covariate structure of the target distribution. At 2500 iterations, the covariance of the proposal distribution is visually proportional to that of the target distribution.

5.2 Comparing the Random Walk and Adaptive Metropolis Algorithm on a simulated data set

5.2.1 Simulated data

In this section we will compare the RW and AM algorithms on the simulated data set from *Default* (Witten et al., 2013) which is available in the **R** package ISLR. The data contains 10.000 rows. Every row correspond to a customer, and for each customer the following 4 variables are available:

- *default:* a factor with levels "Yes" and "No", indicating whether the customer defaulted on their loan,
- *student:* a factor with levels "Yes" and "No", indicating whether the customer is a student or not,
- *balance:* the average balance the customer has remaining after making their monthly payment,
- *income:* income of customer.

The proportion of students were computed to be $(\text{"Yes"}, \text{"No"}) = (0.706, 0.294)$. Our aim is to employ logistic regression with default Cauchy priors as presented in Section 2.3, and the data are processed accordingly. Ten observation selected at random from the original data set are displayed in the table below, together with the corresponding observations in the processed data set.

Original data

default	*student*	*income*	*balance*
No	No	729.53	44361.63
No	Yes	817.18	12106.13
Yes	Yes	1487.00	17854.40
Yes	Yes	2205.80	14271.49
No	No	1073.55	31767.14
Yes	Yes	1774.69	20359.51
No	No	529.25	35704.49
Yes	No	1889.60	48956.17
Yes	Yes	1899.39	20655.20
No	No	785.66	38463.50

Processed data

default	*student*	*income*	*balance*
0	-0.29	-0.11	0.41
0	0.71	-0.02	-0.80
1	0.71	0.67	-0.59
1	0.71	1.42	-0.72
0	-0.29	0.25	-0.07
1	0.71	0.97	-0.49
0	-0.29	-0.32	0.08
1	-0.29	1.09	0.58
1	0.71	1.10	-0.48
0	-0.29	-0.05	0.19

5.2.2 The model

Using the definitions provided in Section 2, our model is defined as follows. We have $N = 10000$ observations and $M = 4$ explanatory variables. Letting $i = 1, ..., N$ denote the ith observation, the probability π is

$$\pi_i = \mathbf{P}(default_i = 1) = 1 - \mathbf{P}(default_i = 0) \quad \text{for } i = 1, ..., 10000. \tag{13}$$

The linear predictor $\boldsymbol{\eta}$ of explanatory variables and regression coefficients $\boldsymbol{\beta}$ is

$$\boldsymbol{\eta}_i = \beta_0 + \beta_1 \times student_i + \beta_2 \times balance_i + \beta_3 \times income_i,$$

and the probability π_i is hence given by

$$\pi_i = \frac{\exp(\boldsymbol{\eta}_i)}{1 + \exp(\boldsymbol{\eta}_i)} \quad \text{for } i = 1, ..., 10000.$$

The likelihood function is

$$L(\boldsymbol{\beta}) = \prod_{i=1}^{10000} \pi_i^{default_i}(1 - \pi_i)^{(1-default_i)}. \tag{14}$$

The log-likelihood function is

$$\ell(\boldsymbol{\beta}) = \sum_{i=1}^{10000} default_i \cdot \boldsymbol{\eta}_i - \sum_{i=1}^{10000} \log(1 + \exp(\boldsymbol{\eta}_i)).$$

With the likelihood as in (14), the posterior distribution is

$$p(\boldsymbol{\beta}) = \frac{L(\boldsymbol{\beta}) \prod_{j=0}^{M} f_j(\beta_j)}{\int L(\boldsymbol{\beta}) \Pi_{j=0}^{M} f_j(\beta_j)\, d\boldsymbol{\beta}},$$

where the prior distributions are

$$f(\beta_0) \sim Cauchy(0, 10) \quad \text{and} \quad \beta_j \sim Cauchy(0, 2.5) \text{ for } j = 1, 2, 3.$$

5.2.3 MCMC results and diagnostics

In this section we present the results from the Bayesian model in Section 5.2.2 using the Random Walk Metropolis and Adaptive Metropolis algorithms to sample the posterior distributions. In both cases we used four independent chains with varying starting points for each parameter. The simulations ran for 10.000 iterations each and the initial 2000 samples were disregarded as warm-up. Model summaries are presented in Tables 2 and 3. Each summary contains the estimated posterior means and standard errors, an estimated 95% credible interval as well as the improved-$\widehat{R}$ statistics and bulk effective sample sizes.

	mean	se	CI (95%)	$\widehat{R}$	bulk-ESS
intercept	-6.156	0.182	[-6.513, -5.799]	1.0039	474
student	-0.646	0.227	[-1.089, -0.209]	1.0035	491
balance	5.532	0.215	[5.117, 5.949]	1.0048	465
income	0.076	0.210	[-0.334, 0.470]	1.0046	518

Table 2: Model summary (Random Walk Metropolis)

	mean	se	CI (95%)	$\widehat{R}$	bulk-ESS
intercept	-6.162	0.191	[-6.544, -5.796]	1.0025	1914
student	-0.639	0.230	[-1.095, -0.205]	1.0026	1837
balance	5.538	0.225	[5.109, 5.988]	1.0033	1965
income	0.085	0.215	[-0.349, 0.495]	1.0020	1830

Table 3: Model summary (Adaptive Metropolis)

Figure 3 show pairwise plots, histograms of marginal posterior draws and correlation between the samples for each parameter. The graphical diagnostics presented in Figure 4 and 5 include trace plots, autocorrelation plots and rank plots. The trace plots show the path of each chain which are distinguished by different shades of blue. For rank plots the parameter with the lowest bulk effective sample size is shown. In Figure 6 we plot the bulk effective sample size and improved-$\widehat{R}$ for an increasing numbers of iterations. Both plots represents these measures for the parameter *balance*, since it had the over all lowest bulk effective sample size.

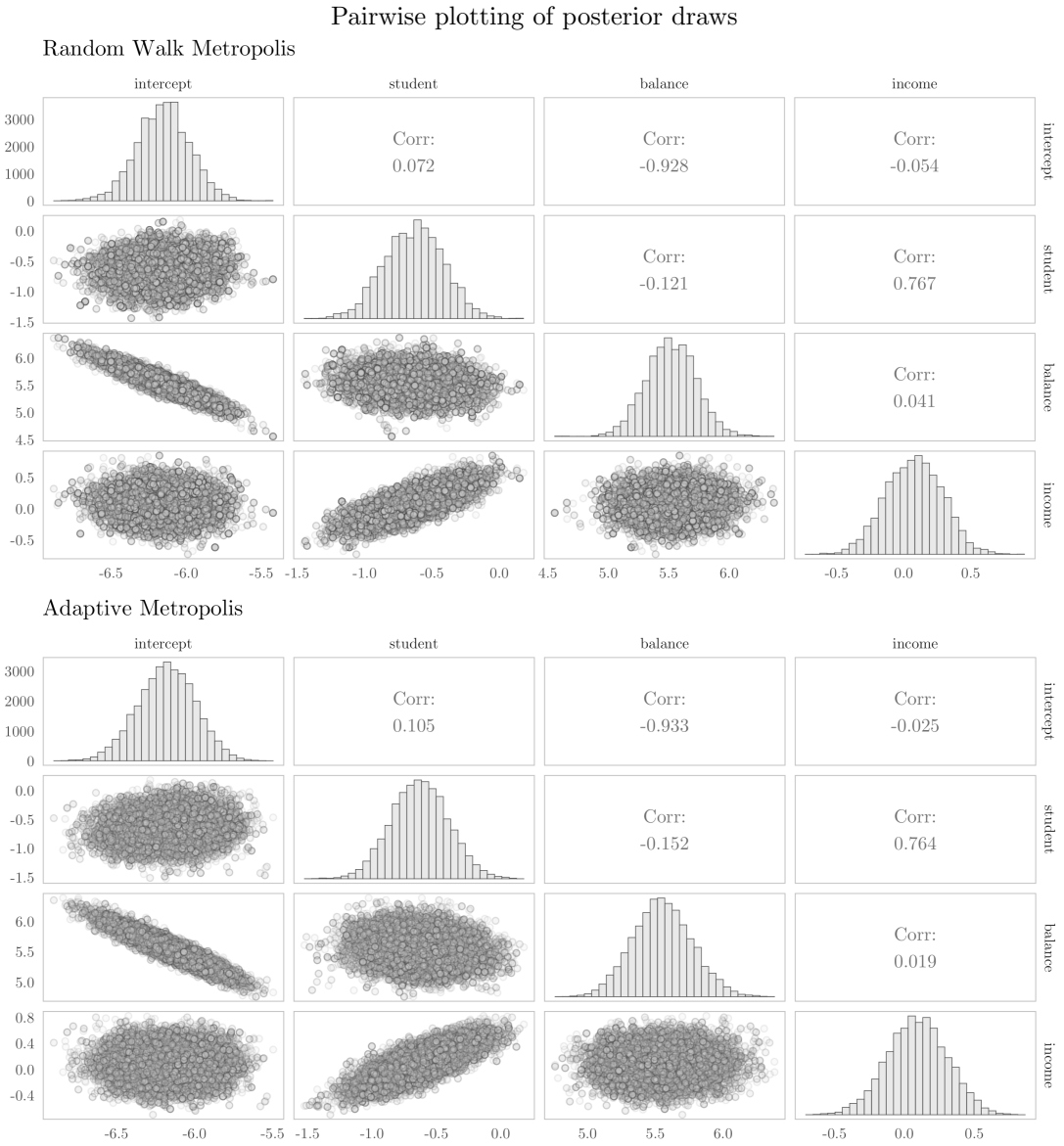

Figure 3: Pairwise plots of the posterior draws obtained from the Random Walk Metropolis (*upper*) and Adaptive Metropolis (*lower*) algorithms. The diagonal display the marginal posterior draws for the parameters, and the upper part shows their correlation.

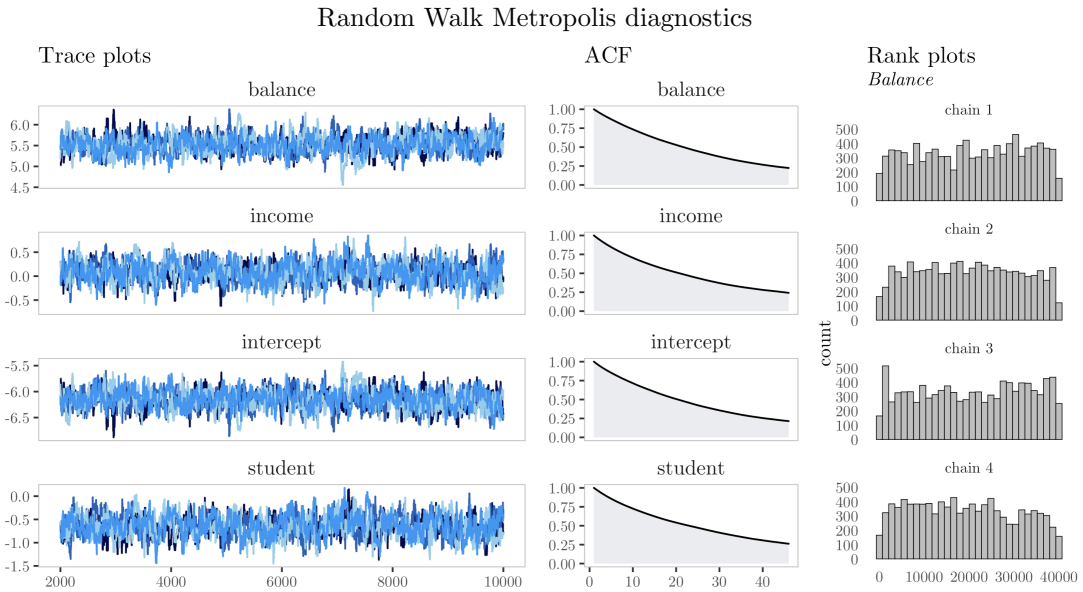

Figure 4: Graphical diagnostics of the Random Walk Metropolis algorithm. ‖ The left column displays the trace plots for every parameter, for which four chains each were used. ‖ The middle column shows the auto correlation plots for each parameter. ‖ The right column shows the rank plots for each parameter.

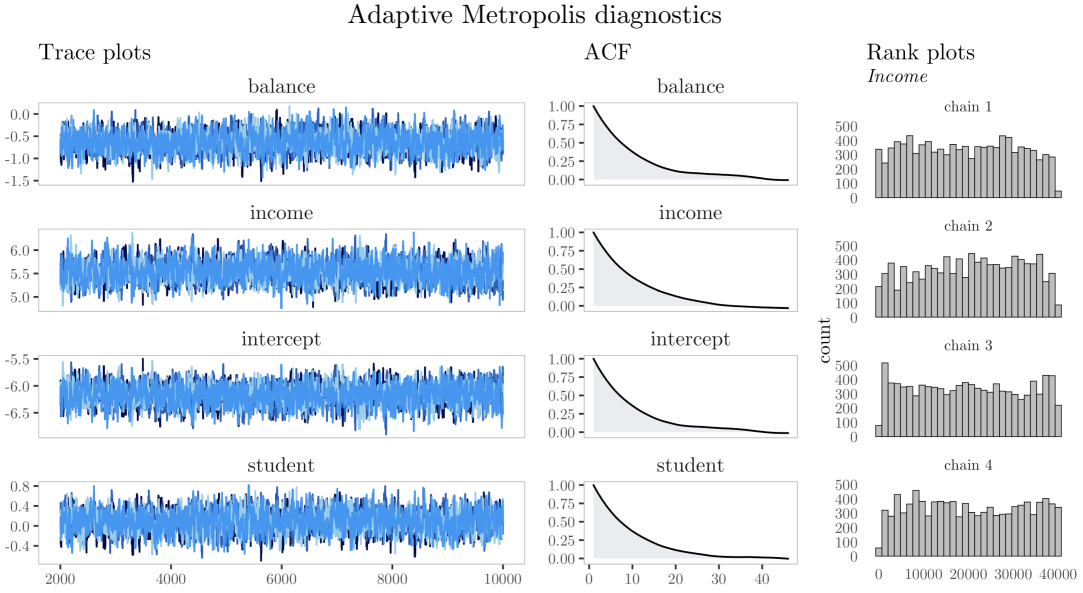

Figure 5: Graphical diagnostics of the Adaptive Metropolis algorithm. ‖ The left column displays the trace plots for every parameter, for which four chains each were used. ‖ The middle column shows the auto correlation plots for each parameter. ‖ The right column shows the rank plots for each parameter.

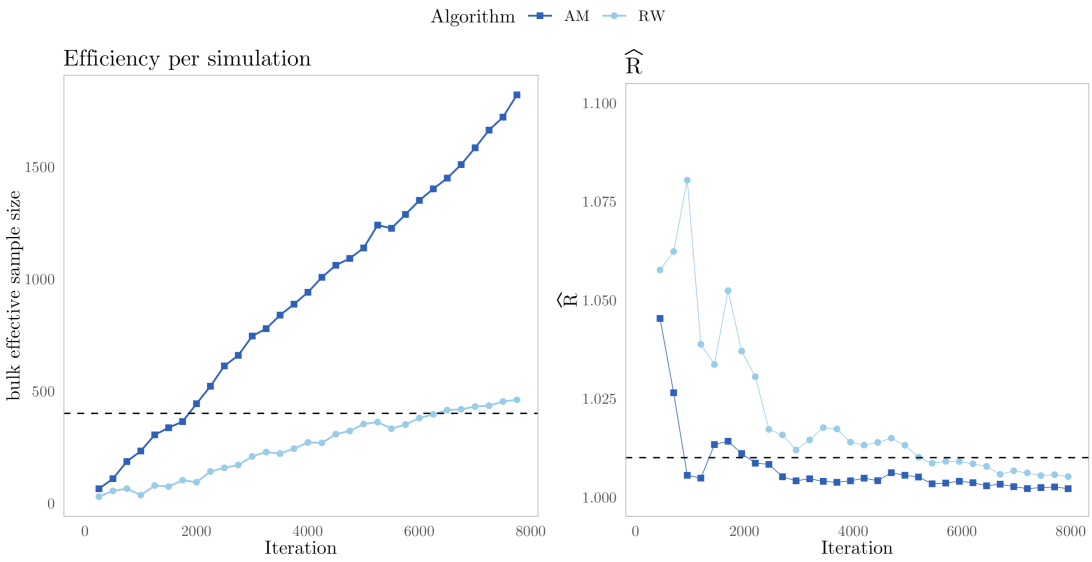

Figure 6: The bulk effective sample size and *improved*-$\widehat{R}$ were computed for an increasing number of iterations. The dashed lines represents the thresholds of 400 (100 per chain) for bulk effective sample size and $\widehat{R} = 1.01$.

6 Discussion

6.1 Results

Both algorithms have yielded similar posterior estimates. The most notable difference in the summaries of Table 2 and 3 are the bulk effective sample sizes. While both algorithms have estimates that exceed the threshold of 400, the bulk-ESS is far greater for the AM algorithm across all parameters. If we observe the autocorrelation plots in Figure 4 and 5 we see that the draws from the RW algorithm suffers from high autocorrelation even at high lags while the AM algorithm fares better. To see why, we turn to the pairwise plots in Figure 3. Here, its clear that the parameters *student* and *balance* are highly correlated with an estimated correlation of 0.77. This is presumably the reason as to why the AM algorithm is more efficient than the RW algorithm in this particular case, since the former adapts to the covariate structure of the posterior distribution. Since the RW algorithm was tuned to have an efficient acceptance rate we may have had to set the scale in such a way that the proposal distribution does not propose too many values in a low density region of the posterior. For a posterior with a complicated covariate structure, this can yield the same behaviour as in Figure 1 where the algorithm moves slowly through the target distribution. This in turn may explain the high autocorrelation observed in the samples from the RW algorithm. Finally, if we look at the evolution of the bulk effective sample sizes and improved-$\widehat{R}$ in Figure 6 we see that the AM algorithm reached the

respective thresholds much faster than the RW algorithm. This means that the AM algorithm yields more effective Monte Carlo estimates for a lower number of iterations than the RW algorithm, and in some sense is more efficient.

6.2 Summary

In this thesis we have compared two Markov Chain Monte Carlo algorithms; the Random Walk Metropolis algorithm and the Adaptive Metropolis Algorithm. The latter can be viewed as an extension of the former and our aim was to see if its efficiency is improved by the alteration. To do this, we defined a Bayesian model for logistic regression on a simulated data set. The posterior inference was based on samples from the Random Walk Metropolis and Adaptive Metropolis algorithms. We also introduced methods for assessing convergence of the chains. In our analysis, the Adaptive metropolis algorithm proved more efficient than the Random Walk Metropolis algorithm.

7 References

Agresti, A. (2014). *Categorical Data Analysis*. Hoboken Wiley.

Gelman, A., Carlin, J.B., Hal Steven Stern, Dunson, D.B., Aki Vehtari and Rubin, D.B. (2014). *Bayesian data analysis*. Boca Raton: Crc Press.

Haario, H., Saksman, E. and Tamminen, J. (2001). An Adaptive Metropolis Algorithm. *Bernoulli*, 7(2), p.223. doi:10.2307/3318737.

Held, L. and Daniel Sabanés Bové (2014). *Applied Statistical Inference : Likelihood and Bayes.* Berlin, Heidelberg: Springer Berlin Heidelberg.

Hill, S.D. and Spall, J.C. (2019). Stationarity and Convergence of the Metropolis-Hastings Algorithm: Insights into Theoretical Aspects. *IEEE Control Systems*, 39(1), pp.56–67. doi:10.1109/mcs.2018.2876959.

Levin, D., Peres, Y., Wilmer, E., Propp, J. and Wilson, D., 2017. Markov chains and mixing times. 2nd ed. Providence: American mathematical society.

Ross, S., 2019. *Introduction to Probability Models* (Twelfth Edition). [S.l.]: Academic Press.

van de Schoot, R., Depaoli, S., King, R. et al. Bayesian statistics and modelling. Nat Rev Methods Primers 1, 1 (2021). https://doi.org/10.1038/s43586-020-00001-2

Vehtari, A., Gelman, A., Simpson, D., Carpenter, B. and Bürkner, P.-C. (2021). Rank-Normalization, Folding, and Localization: An Improved $\widehat{R}$ for Assessing Convergence of MCMC. *Bayesian Analysis*. doi:10.1214/20-ba1221.

James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013) *An Introduction to Statistical Learning with applications in R*, https://www.statlearning.com, Springer-Verlag, New York