# Directional Metropolis–Hastings algorithms on hyperplanes

Hugo Hammer and Håkon Tjelmeland
Department of Mathematical Sciences
Norwegian University of Science and Technology
Trondheim, Norway

**Abstract**

In this paper we define and study new directional Metropolis–Hastings algorithms that propose states in hyperplanes. Each iteration in directional Metropolis–Hastings algorithms consist of three steps. First a direction is sampled by an auxiliary variable. Then a potential new state is proposed in the subspace defined by this direction and the current state. Lastly, the potential new state is accepted or rejected according to the Metropolis–Hastings acceptance probability. Traditional directional Metropolis–Hastings algorithms define the direction by one vector and so the corresponding subspace in which the potential new state is sampled is a line. In this paper we let the direction be defined by two or more vectors and so the corresponding subspace becomes a hyperplane.

We compare the performance of directional Metropolis–Hastings algorithms defined on hyperplanes with other frequently used Metropolis–Hastings schemes. The experience is that hyperplane algorithms on average produce larger jumps in the sample space and thereby have better mixing properties per iteration. However, with our implementations the hyperplane algorithms is more computation intensive per iteration and so the simpler algorithms in most cases are better when run for the same amount of computer time. An interesting area for future research is therefore to find variants of our directional Metropolis–Hastings algorithms that require less computation time per iteration.

*Keywords:* Angular Gaussian distribution, Directional Metropolis–Hastings, Markov chain Monte Carlo, Hyperplane algorithms, Subspace.

## 1 Introduction

We are often interested in calculating the mean of some function $f(x)$ for $x$ distributed according to a target probability distribution $\pi(\cdot)$. If $x$ is of high dimension and $\pi(\cdot)$ is complex, stochastic simulation is the only viable alternative. There exists a wide range of simulation algorithms for this. If the target distribution $\pi(\cdot)$ is sufficiently complex, the Metropolis–Hastings (MH) algorithm (Metropolis et al., 1953; Hastings, 1970) is the standard choice. In MH algorithms we run a Markov chain which has $\pi(\cdot)$ as its limiting probability distribution. Each new state of the chain is generated in two steps. Letting $x$ denote the current state of the Markov chain, first a potential new state, $y$, is proposed and second the proposal $y$ is accepted with a certain probability, otherwise retaining $x$ as the current state.

In directional MH algorithms the proposal step is done in two parts. First, a direction is sampled via an auxiliary variable. Next, the potential new state is sampled in the subspace defined by the sampled direction and the current state of the Markov chain. Several directional MH algorithms are discussed in the literature: Chen and Schmeiser (1993) generate the direction from an uniform distribution, Roberts and Rosenthal (1998) present an algorithm that moves along a direction centered at the gradient direction evaluated in the current point. Also Eidsvik and Tjelmeland (2006) let the direction depend on the current state of the Markov chain. Gilks et al. (1994), Roberts and Gilks (1994) and Liu and Sabatti (2000) use information from parallel chains to generate proposals along favourable directions.

Goodman and Sokal (1989) and Liu and Sabatti (2000) find what proposal distribution to use in the chosen subspace to obtain unit MH acceptance rate. Eidsvik and Tjelmeland (2006) also study this, but for direction distributions that depend on the current state of the Markov chain.

In all the articles discussed above, the directional MH algorithms use directions that are defined by one vector, and so the corresponding subspace is a line. The focus of this paper is when the direction is defined by two or more vectors, so that the resulting subspace is a (hyper) plane. By allowing ourselves to generate the potential new states in a hyperplane instead of just a line, the setup becomes more flexible and thereby there should be a potential for obtaining algorithms with better mixing properties. We implement this idea by generalising the setup in Eidsvik and Tjelmeland (2006). Similar to what is done in that article our focus is on proposal distributions that give unit MH acceptance rates. We compare the performance of our new directional MH schemes with other frequently used MH algorithms. Our experience is that the hyperplane algorithms on average produce larger jumps in each iteration and thereby have better mixing properties per iteration. In some cases our new algorithms even give negative one iteration correlations, which we were not able to obtain by either the directional MH algorithms using only one vector to define the direction, or the simple random walk and Langevin proposal algorithms. However, with our implementations the hyperplane algorithms require more computation time per iteration and in most cases therefore is outperformed by the simpler algorithms when run for the same amount of computer time. An interesting area for future research is therefore to define variants of the hyperplane algorithms that require less computation time per iteration.

The paper is organised as follows. In Section 2 we give a short introduction to directional MH algorithms and in Section 3 we present details for directional MH algorithms generating proposals in hyperplanes. In Section 4 we study how to define the proposal distributions to obtain unit acceptance rate. In section 5 we present a simulation example comparing the performance of our new directional MH algorithm with other frequently used MH schemes.

## 2 Metropolis–Hastings algorithms

A MH algorithm simulate from a specified target distribution, $\pi(\cdot)$, by simulating a Markov chain for a large number of iterations, where the Markov chain is constructed to have the target distribution $\pi(\cdot)$ as its limiting distribution. In this paper we restrict the attention to continuous target distribution on $\mathbb{R}^n$ and let $\pi(\cdot)$ denote its density with respect to the Lebesgue measure on $\mathbb{R}^n$. We let $x \in \mathbb{R}^n$ denote the current state of the Markov chain.

### 2.1 Parametric Metropolis–Hastings

Parametric MH is a subclass of the MH algorithm where a parametric form is used to generate potential new states (Green, 1995; Waagepetersen and Sørensen, 2001), also known as reversible jump algorithms. The algorithms are originally developed for target distributions defined on sample spaces of varying dimension, but they are equally valid when the dimension of the state vector $x$ is fixed. In the following we describe the procedure for this latter case.

To generate a potential new state $y$ one first samples a $t \in \mathbb{R}^m$ from a distribution $q(t|x)$

and secondly generates the proposal $y$ using the one-to-one transformation

$$\left.\begin{array}{l} y = w_1(x,t) \\ s = w_2(x,t) \end{array}\right\} \Leftrightarrow \left\{\begin{array}{l} x = w_1(y,s) \\ t = w_2(y,s), \end{array}\right. \tag{1}$$

where $w_1 : \mathbb{R}^{n+m} \to \mathbb{R}^n$ and $w_2 : \mathbb{R}^{n+m} \to \mathbb{R}^m$. The proposal $y$ should thereafter be accepted with probability

$$\alpha(y|x) = \min\left\{1, \frac{\pi(y)q(s|y)}{\pi(x)q(t|x)}|J|\right\}, \tag{2}$$

where

$$J = \begin{vmatrix} \frac{\partial w_1(x,t)}{\partial x} & \frac{\partial w_1(x,t)}{\partial t} \\ \frac{\partial w_2(x,t)}{\partial x} & \frac{\partial w_2(x,t)}{\partial t} \end{vmatrix} \tag{3}$$

is the Jacobian determinant for the one-to-one relation (1).

In the following we also use an auxiliary stochastic variable in the parametric MH setup. Letting $\varphi \in \Phi$ denote the auxiliary variable, the procedure is then as follows. First we generate $\varphi$ from some distribution $h(\varphi|x)$. As indicated in the notation the distribution for $\varphi$ may depend on the current state vector $x$. Next we generate $t$ from a distribution $q(t|x,\varphi)$ that also may depend on the auxiliary variable. From $t$ and $\varphi$ we compute the proposal $y$ using the one-to-one relation (1), where now $w_1$ and $w_2$ may be functions also of $\varphi$. Note, however, that the one-to-one relation is still between $(x,t)$ and $(y,s)$, the auxiliary variable $\varphi$ should here be considered as a fixed parameter. Lastly, we accept the proposal $y$ with probability

$$\alpha(y|\varphi,x) = \min\left\{1, \frac{\pi(y)h(\varphi|y)q(s|\varphi,y)}{\pi(x)h(\varphi|x)q(t|\varphi,y)}|J_\varphi|\right\}, \tag{4}$$

where, as indicated in the notation, the Jacobian determinant may be a function of $\varphi$.

## 2.2 Directional Metropolis–Hastings

The class of directional MH algorithms is an important subclass of parametric MH algorithms. In directional MH algorithms potential new states are traditionally generated along a line in the sample space defined by the current state $x$ and an auxiliary variable $\varphi$, where $\varphi$ is either a point or a directional vector in $\mathbb{R}^n$. This situation is considered in Eidsvik and Tjelmeland (2006) and in the following we introduce this setup and discuss the most important experiences for this situation. In the next section we generalise the scheme to allow proposals in a hyperplane.

Following Eidsvik and Tjelmeland (2006), we use $z$ to denote the auxiliary variable whenever it represents a point in $\mathbb{R}^n$, and use $u$ when it represents a direction vector. First consider the case that the line is defined by $x$ and an auxiliary point $z$. An iteration of the MH algorithm then starts by generating $z$ from a distribution $p(z|x)$. Next, we generate a scalar $t$ from some distribution $q(t|z,x)$ and compute the proposal $y$ through the one-to-one relation

$$\left.\begin{array}{l} y = x + t(z-x) \\ s = -\frac{t}{1-t} \end{array}\right\} \Leftrightarrow \left\{\begin{array}{l} x = y + s(z-y) \\ t = -\frac{s}{1-s}. \end{array}\right. \tag{5}$$

Using (4), the acceptance probability becomes

$$\alpha(y|z,x) = \min\left\{1, \frac{\pi(y)p(z|y)q(s|z,y)}{\pi(x)p(z|x)q(t|z,y)}|J_z|\right\}, \tag{6}$$

where the Jacobian determinant is

$$J_z = \left| \begin{array}{cc} \frac{\partial y(x,t)}{\partial x} & \frac{\partial y(x,t)}{\partial t} \\ \frac{\partial s(x,t)}{\partial x} & \frac{\partial s(x,t)}{\partial t} \end{array} \right| = \left| \begin{array}{cc} (1-t)\cdot \mathbf{I}_n & z-x \\ \mathbf{0} & -(1-t)^{-2} \end{array} \right| = -(1-t)^{n-2}, \tag{7}$$

where $I_n$ is the $n$-dimensional identity matrix and $\mathbf{0}$ is a matrix with only zero entries.

Consider next the situation when we use a direction vector $u$ to define the line. To obtain a unique vector $u$ for a given line, Eidsvik and Tjelmeland (2006) require

$$u \in S^n = \{u \in \mathbb{R}^n \setminus \{0\} : \|u\| = 1 \text{ and } u_l > 0 \text{ for } l = \min\{j; u_j \neq 0\}, \}. \tag{8}$$

Each iteration of the algorithm then starts by generating a direction $u$ from a distribution $g(u|x)$. Next, a scalar $t$ is proposed from a distribution $q(t|u,x)$ and the proposal $y$ is found by the one-to-one relation

$$\left. \begin{array}{l} y = x + tu \\ s = -t \end{array} \right\} \Leftrightarrow \left\{ \begin{array}{l} x = y + su \\ t = -s. \end{array} \right. \tag{9}$$

The acceptance probability now becomes

$$\alpha(y|u,x) = \min\left\{1, \frac{\pi(y)g(u|y)q(s|u,y)}{\pi(x)g(u|x)q(t|u,y)}|J_u|\right\}, \tag{10}$$

where

$$J_u = \left| \begin{array}{cc} \frac{\partial y(x,t)}{\partial x} & \frac{\partial y(x,t)}{\partial t} \\ \frac{\partial s(x,t)}{\partial x} & \frac{\partial s(x,t)}{\partial t} \end{array} \right| = \left| \begin{array}{cc} \mathbf{I}_n & u \\ \mathbf{0} & -1 \end{array} \right| = -1. \tag{11}$$

A critical choice in the two MH algorithms defined above is the distribution for the auxiliary variable, $p(z|x)$ or $g(u|x)$. For $p(z|x)$ Eidsvik and Tjelmeland (2006) propose to use a Gaussian approximation to the target distribution. For $g(u|x)$ a uniform distribution in $S^n$ is clearly a possible choice. However, a better choice would be one that is adapted to the target distribution in question and the current state $x$. This can be achieved by first sampling a point $z$ from a $p(z|x)$ and let $u$ be the direction vector that define the same line, i.e.

$$u = \left\{ \begin{array}{ll} \frac{z-x}{||z-x||} & \text{if } \frac{z-x}{||z-x||} \in S^n, \\ -\frac{z-x}{||z-x||} & \text{otherwise.} \end{array} \right. \tag{12}$$

The resulting $g(u|x)$ is then given as

$$g(u|x) = \int_{-\infty}^{\infty} |r|^{n-1} p(x + ru|x) \mathrm{d}r. \tag{13}$$

To be able to compute the corresponding acceptance probability $g(u|x)$ must be analytically available. In general this is not be the case, but when $p(z|x)$ is Gaussian the corresponding $g(u|x)$ is called the angular Gaussian distribution and can easily be evaluated, see e.g. Watson (1983) and Pukkila and Rao (1988).

One should note that the acceptance probabilities in the two above directional MH algorithms differ. They become different even if the proposal mechanisms are chosen identical by defining $g(u|x)$ indirectly via $p(z|x)$ as discussed above and the proposal distributions along

the line are chosen to be the same. The experience reported in Eidsvik and Tjelmeland (2006) is that the algorithm based on $u$ results in longer jumps and better mixing. Intuitively that can be understood as the acceptance probabilities are conditional on the sampled values of the auxiliary variable and a direction $u$ contains less information than a point $z$. When generalising the above scheme to a situation where the auxiliary variable defines a hyperplane instead of a line our focus is therefore on direction vectors.

### 2.3 Hyperplane directional MH algorithms

We now consider a directional MH algorithm where the potential new state is proposed in a hyperplane of a dimension $k < n$. Thus, $k = 1$ corresponds to the algorithm discussed above. Each iteration of the algorithm then starts by generating $k$ vectors $u_1, \ldots, u_k$ according to a distribution $g(u|x)$, where $u = (u_1, \ldots, u_k)$. Next we propose a $k$-dimensional vector $t = (t_1, \ldots, t_k)^T \in \mathbb{R}^k$ from a distribution $q(t|x, u)$ and compute the proposal $y$ through the one-to-one relation

$$\left.\begin{array}{l} y = x + \sum_{i=1}^k t_i u_i \\ s = -t \end{array}\right\} \Leftrightarrow \left\{\begin{array}{l} x = y + \sum_{i=1}^k s_i u_i \\ t = -s, \end{array}\right. \tag{14}$$

where $s = (s_1, \ldots, s_k)^T \in \mathbb{R}^k$. The corresponding acceptance probability becomes

$$\alpha(y|u, x) = \min\left\{1, \frac{\pi(y)g(u|y)q(s|u, y)}{\pi(x)g(u|x)q(t|u, y)}|J_u|\right\}, \tag{15}$$

where the Jacobian determinant is given by

$$J_u = \left|\begin{array}{cc} \frac{\partial y(x,t)}{\partial x} & \frac{\partial y(x,t)}{\partial t} \\ \frac{\partial s(x,t)}{\partial x} & \frac{\partial s(x,t)}{\partial t} \end{array}\right| = \left|\begin{array}{cc} \mathbf{I}_n & u_1 \ \cdots \ u_k \\ \mathbf{0} & -\mathbf{I}_k \end{array}\right| = (-1)^k. \tag{16}$$

To fully define the simulation algorithm it remains to specify the distribution of the auxiliary variables, $g(u|x)$, and the proposal distribution $q(t|u, x)$. In the following we first discuss how $q(t|u, x)$ should be chosen to give unit acceptance probability.

#### 2.3.1 Proposal $q$ with unit acceptance rate

Following the strategy used in Eidsvik and Tjelmeland (2006) one can easily identify a proposal distribution $q(t|x, u)$ that gives unit acceptance probability, namely

$$q(t|u, x) = c(x, u)\ \pi\left(x + \sum_{i=1}^k t_i u_i\right)\ g\left(u \,\middle|\, x + \sum_{i=1}^k t_i u_i\right), \tag{17}$$

where $c(x, u)$ is a normalising constant given by

$$c(x, u) = \left[\int_{\mathbb{R}^k} \pi\left(x + \sum_{i=1}^k t_i u_i\right)\ g\left(u \,\middle|\, x + \sum_{i=1}^k t_i u_i\right)\ \mathrm{d}t\right]^{-1}. \tag{18}$$

To see that the resulting acceptance probability is equal to one, insert the above expression for $q(t|u, x)$ in (15),

$$\alpha(y|u, x) = \min\left\{\frac{\pi(y)\ g(u|y)\ c(y, u)\ \pi\Big(y + \sum_{i+1}^k s_i u_i\Big)\, g\Big(u \,\Big|\, y + \sum_{i=1}^k s_i u_i\Big)}{\pi(x)\ g(u|x)\ c(x, u)\ \pi\Big(x + \sum_{i=1}^k t_i u_i\Big)\, g\Big(u|x + \sum_{i=1}^k t_i u_i\Big)}\right\}. \tag{19}$$

Using the one-to-one relation between $(x,t)$ and $(y,s)$ in (14) and noting that $c(x,u) = c(y,u)$ we see that everything in the fraction cancels and we get $\alpha(y|u,x) = 1$.

It should be noted that in most cases we will in practice not be able to use the $q(t|u,x)$ defined above because the integral in (17) is not analytically available. However, the above expressions are still of interest as we may construct analytically available approximations to (17) and thereby get acceptance rates close to one. In practice how to do this depend on the properties of the $q(t|u,x)$ in (17), which in turn depends on the choice of $g(u|x)$. Next we therefore discuss the choice of $g(u|x)$.

# 3 Choice of $g(u|x)$

The goal is to choose $g(u|x)$ so that the resulting hyperplanes with high probability goes through high density areas of the target distribution. To obtain this it seems reasonable to draw auxiliary points, $z_1, \ldots, z_k$, from a distribution that resembles the target density and use direction vectors that span the same hyperplane. In the following we discuss two ways to define direction vectors $u_1, \ldots, u_k$ from $z_1, \ldots, z_k$. The first set of direction vectors we denote by $u_1^{(1)}, \ldots, u_k^{(1)}$ and the resulting distribution by $g_1(u^{(1)}|x)$ and use $u_1^{(2)}, \ldots, u_k^{(2)}$ and $g_2(u^{(2)}|x)$ for the corresponding quantities for the second set of direction vectors.

Assume we have available a Gaussian approximation to the target distribution $\pi(\cdot)$, and denote this by $\tilde{\pi}(\cdot)$. Thus, letting $z_1, \ldots, z_k$ be independent realisations from $\tilde{\pi}(\cdot)$, we may define corresponding direction vectors $u_1^{(1)}, \ldots, u_k^{(1)}$ by applying (12) to each $z_i$ separately. Referring to our discussion in Section 2.2, the resulting $g_1(u^{(1)}|x)$ clearly becomes a product of angular Gaussian densities.

Notice that even though each $u_i^{(1)}$ is uniquely defined by the corresponding $z_i$, the set $u_1^{(1)}, \ldots, u_k^{(1)}$ is not uniquely defined from the set $z_1, \ldots, z_k$. Other direction vectors spanning the same hyperplane may be defined by taking linear combinations of $u_1^{(1)}, \ldots, u_k^{(1)}$. Thus, as we discuss in the last paragraph of Section 2.2, we should expect to get an algorithm with better mixing properties by in stead using a set of direction vectors, $u_1^{(2)}, \ldots, u_k^{(2)}$, that is uniquely defined from $z_1, \ldots, z_k$. We obtain this by setting

$$\begin{aligned} u_1^{(2)} &= [1, 0, \ldots, 0, u_{1,k+1}, \ldots, u_{1,n}]^T \\ u_2^{(2)} &= [0, 1, \ldots, 0, u_{2,k+1}, \ldots, u_{2,n}]^T \\ &\vdots \\ u_k^{(2)} &= [0, 0, \ldots, 1, u_{k,k+1}, \ldots, u_{k,n}]^T. \end{aligned} \tag{20}$$

Thus, $u_1^{(2)}, \ldots, u_k^{(2)}$ are defined by $k(n-k)$ variables and these can by found from $z_1, \ldots, z_k$ by solving

$$z_i = x + \sum_{j=1}^{k} c_{ij} u_j^{(2)} \quad \text{for } i = 1, \ldots, k, \tag{21}$$

with respect to $u^{(2)} = \{u_{ij}, i = 1, \ldots, k, j = k+1, \ldots, n\}$ and $c = \{c_{ij}, i, j = 1, \ldots, k\}$. The resulting distribution for $u^{(2)}$, $g_2(u^{(2)}|x)$, is found by first finding the joint distribution for $u^{(2)}$ and $c$, $g_2(u^{(2)}, c|x)$, and thereafter marginalise this over $c$,

$$g_2\left(u^{(2)}|x\right) = \int g_2\left(u^{(2)}, c \middle| x\right) dc. \tag{22}$$

For $n = 2$ and $k = 1$ this integral can be solved analytically and $u_{1,2}$, which is then the only variable in $u^{(2)}$, has a Cauchy distribution. For $n > 2$ we have not been able to solve the integral analytically, but simulations show that $g_2(u^{(2)}|x)$ has heavy tails also for $n > 2$. As $g_2(u^{(2)}|x)$ is not analytically available for $n > 2$ it can not be used, but an approximation to it may be used. To construct such an approximation we note that both the conditional marginal distribution for $c$ given $x$, and the conditional distribution for $u^{(2)}$ given $c$ and $x$ are Gaussians. Thus, it is natural to factorize the integrand in (22) as a product of these two distributions,

$$g_2(u^{(2)}, c|x) = g_2(u^{(2)}|c, x)g_2(c|x). \tag{23}$$

Note, however, that $u^{(2)}$ and $c$ given $x$ are not jointly Gaussian. Inserting this in (22) and performing a linear substitution for $c$ so that the new variable, denoted by $\tilde{c}$, gets a $k^2$-variate standard Gaussian distribution, we have

$$g_2(u^{(2)}|x) = \int g_2(u^{(2)}|\tilde{c}, x)\mathrm{N}(\tilde{c})\mathrm{d}\tilde{c}, \tag{24}$$

where $\mathrm{N}(\cdot)$ is the density of a standard Gaussian distribution. A natural approximation is therefore to sample $\tilde{c}_1, \ldots, \tilde{c}_N$ independently from the standard Gaussian distribution and use

$$\widehat{g}_2(u^{(2)}|x) = \frac{1}{N}\sum_{i=1}^{N} g_2(u^{(2)}|\tilde{c}_i, x). \tag{25}$$

Clearly, $\widehat{g}_2(u^{(2)}|x)$ is easy to sample from and it is straight forward to evaluate the corresponding density. We will therefore use this as our second distribution for $u$. We note in passing that taking $N = 1$ corresponds to using $z_1, \ldots, z_k$ to define the hyperplane. Clearly, in practice one should use a value for $N$ much larger than this.

# 4 Appearance of the unit acceptance rate proposal distribution

For $k = 1$ Eidsvik and Tjelmeland (2006) show that the proposal distribution that gives unit acceptance rate, equation (17), typically becomes a bi-modal distribution with one mode close to $x$ and one far away from $x$. Thus, using this proposal distribution enables large jumps in the sample space by proposing values to the mode away from the current state $x$. In this section we study how this generalises for $k > 1$. As discussed in Section 2.3.1 we can not actually use (17) as our proposal distribution, but this is still of interest for two reasons. It shows what the potential is for using a proposal distribution that is an approximation to (17) and it may generate ideas for how to construct good approximations to (17).

To study the appearance of (17) for $k > 1$ we consider a simplified variant of a Bayesian model that in Rabben et al. (2008) is used for non-linear inversion of seismic data. The variable of interest, $x \in \mathbb{R}^n$, is assumed to have a Gaussian prior distribution with mean $m$ and covariance matrix $S$. Given $x$, the data $d$ is assumed Gaussian with mean value $ax \odot x + bx$ and covariance matrix $\Sigma$, where $a$ and $b$ are scalar (known) parameters and $\odot$ denotes elementwise multiplication. Thus, the posterior distribution for $x$ given $d$ becomes non-Gaussian for $a \neq 0$. Our simplification relative to the model used in Rabben et al. (2008) is that we here consider $n = 10$, and in Section 5 $n = 30$, whereas in the seismic inversion application much larger values of $n$ are of interest.

Our target distribution of interest is the posterior distribution $\pi(x|d) \propto \pi(d|x)\pi(x)$. Our simulation algorithm requires that we have available a Gaussian approximation to $\pi(x|d)$ and this is easily obtained by approximating the likelihood mean in each dimension, $ax_i^2 + bx_i$ by the corresponding linear Taylor approximation around the prior mean, $m_i$. We let $\tilde{\pi}(x|d)$ denote the resulting Gaussian posterior approximation. In the following we use $b = 1$, $m = d = (1, \ldots, 1)^T$, $S_{ij} = \exp\{-(i-j)^2\}$ and $\Sigma_{ij} = \exp\{-|i-j|\}$, and consider three different values for $a$, 0, 0.1 and 0.3. Clearly, for $a = 0$ the two posterior distributions $\pi$ and $\tilde{\pi}$ are identical.

The appearance of the proposal distribution (17) that gives unit acceptance probability depends on our choice of distribution for the auxiliary variable $u$. We consider the two alternatives defined in Section 3, $g_1(u^{(1)}|x)$ and $g_2(u^{(2)}|x)$. As discussed in Section 3, the $g_2(u^{(2)}|x)$ is not analytically available, so here we use the approximation in (25) with a very large value for $N$ so that the result is indistinguishable from $g_2(u^{(2)}|x)$. To map the appearance of (17) we first run until convergence a directional MH algorithm with $g_1(u^{(1)}|x)$ as distribution for the auxiliary variable. At an arbitrary iteration after convergence we stop the simulation, generate a $u^{(1)}$ and the corresponding $u^{(2)}$ and map the resulting two proposal distributions,

$$q_j\left(t|u^{(j)}, x\right) = c\left(x, u^{(j)}\right)\pi\left(x + \sum_{i=1}^{k} t_i u_i^{(j)}\right)\, g_j\left(u^{(j)}\,\middle|\, x + \sum_{i=1}^{k} t_i u_i^{(j)}\right) \tag{26}$$

for $j = 1, 2$. The two $q_1(t|u^{(1)}, x)$ and $q_2(t|u^{(2)}, x)$ are not directly comparable as the same vector $t$ give different proposed values $y$. To fix this we in stead choose an orthonormal basis for the hyperplane spanned by $x$ and $u^{(1)}$ (or $u^{(2)}$), $v_1, \ldots, v_k$ and consider the two distributions relative to this basis, i.e.

$$r_j\left(t\,\middle|\, u^{(j)}, v_1, \ldots, v_k, x\right) \propto \pi\left(x + \sum_{i=1}^{k} t_i v_i\right)\, g_1\left(u^{(j)}\,\middle|\, x + \sum_{i=1}^{k} t_i v_i\right) \tag{27}$$

for $j = 1, 2$. It should be noted that by transforming to an orthonormal basis gives that the Euclidean norm $\|t\|$ is equal to the Euclidean distance between $x$ and the proposed value $y$ in $\mathbb{R}^n$. In Figure 1 we show results for $k = 2$ when $a = 0$ (left column) and $a = 0.3$ (right column). Thus, the target distribution is Gaussian in the left column and non-Gaussian in the right column. The upper row shows the target distribution $\pi(x + t_1 v_1 + t_2 v_2)$ for $t = (t_1, t_2)^T$, $t_1, t_2 \in (-7, 5, 7.5)$. For the same values of $t$ the middle row shows $r_1(t|u^{(1)}, v_1, v_2, x)$ and the lower row $r_2(t|u^{(2)}, v_1, v_2, x)$. The current state $x$ is located in the center in all six plots. We see that in the lower row we get an **O**-shaped distribution. Comparing the upper and lower rows for $a = 0$ we observe that $r_2(t|u^{(2)}, v_1, v_2, x)$ seem to follow the contours of the target distribution. Intuitively it seems reasonable that proposals to other parts of the target distribution with almost the same density as $x$ should often be accepted with high probability. We observe this property also for $a = 0.3$, but here $r_2(t|u^{(2)}, v_1, v_2, x)$ is not symmetrical. Note that the results in the lower row is in accordance with the observation made in Eidsvik and Tjelmeland (2006), namely that the proposal distributions for $k = 1$ typically have two modes, one on each side of a high density area of the target distribution. Here we have a high density area of the target distribution in the center of the **O**-shaped distribution. For $a = 0$, middle row, we see that the two largest modes are placed on each side of a high density area of the target distribution, one close to $x$ and one on the other side of the high density area of the target distribution. Again this is accordance with Eidsvik and Tjelmeland (2006).

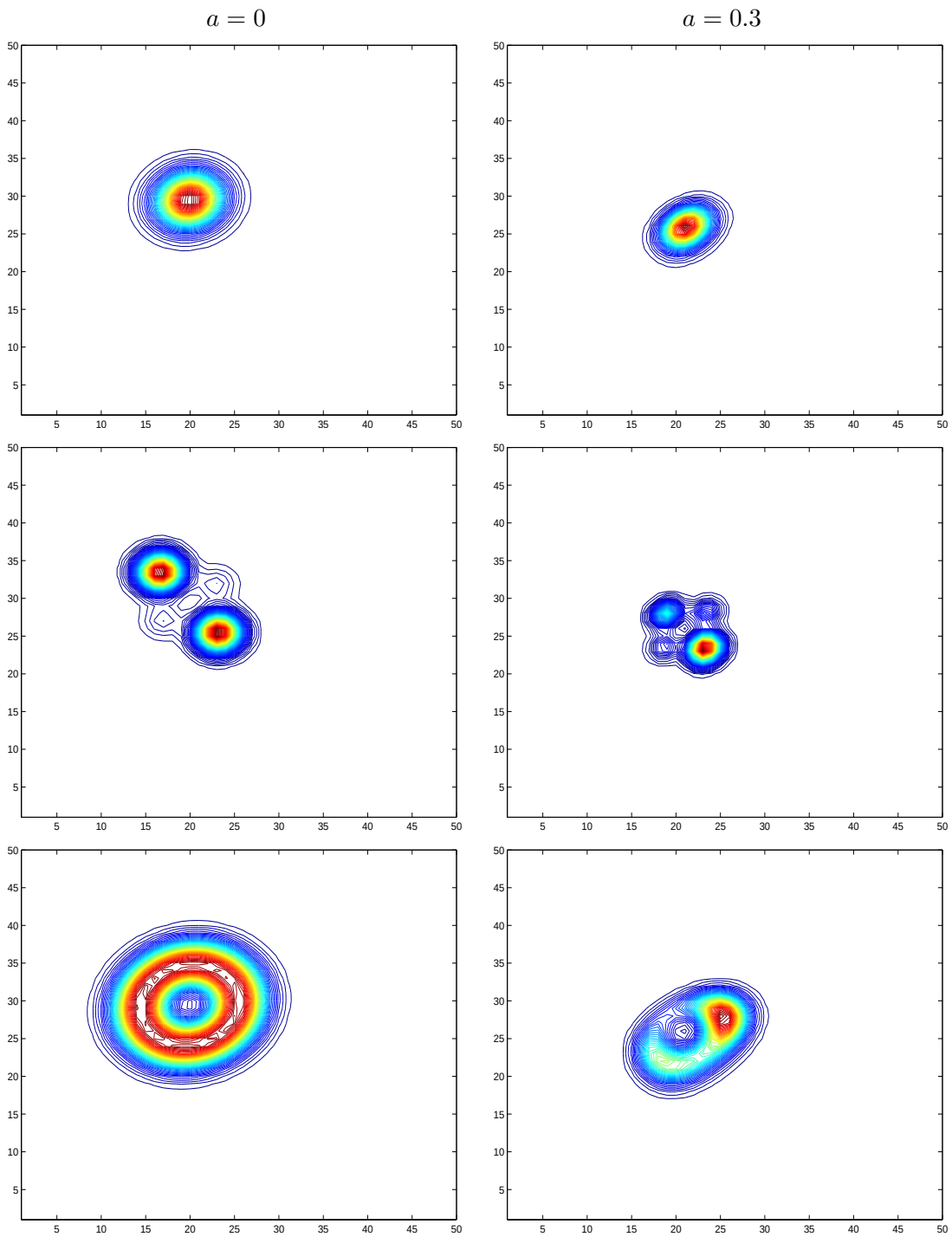


Figure 1: Results for the Bayesian simulation example with $a = 0$ (left column) and $a = 0.3$ (right column). The upper row shows the target distribution $\pi(x+t_1v_1+t_2v_2)$ for $t = (t_1, t_2)^T$, $t_1, t_2 \in (-7, 5, 7.5)$. For the same values of $t$ the middle row shows $r_1(t|u^{(1)}, v_1, v_2, x)$ and the lower row $r_2(t|u^{(2)}, v_1, v_2, x)$. The current state $x$ is in the center, $(25, 25)$, in all of the panels.

Comparing the middle and lower rows for $a = 0$, we observe that the two largest modes in the middle row are parts of the **O**-shaped distribution. This seem to be a general behaviour. We observe the same for $a = 0.3$. Thus, long jumps are possible with both $g_1(u|x)$ and $g_2(u|x)$, but we are loosing something in *where* we are able to propose potential new states by using $u^{(1)}$ based on a non-unique basis relative to using $u^{(2)}$ that is based on a unique basis.

# 5 Bayesian simulation example

In this Section we analyse the behaviour of the algorithm based on $g_1(u^{(1)}|x)$ introduced above and compare the results with simpler MH algorithms. In turns out that the simulations necessary to calculate a satisfying approximation to (22) by using (25) is too expensive to be of any practical use. We again consider the Bayesian model for non-linear seismic inversion defined in Section 5, but now let $n = 30$.

## 5.1 Approximating the unit acceptance rate proposal distribution

As the proposal distribution (17) is not analytically available we need to construct an approximation to it and use this as our proposal distribution. From the discussion in Section 4 we know that the proposal distribution (17) typically contains two large modes, one close to $t = 0$ and one further away from $t = 0$. Our strategy is to approximate (17) by a mixture of two Gaussian distributions.

To locate the large mode close to $t = 0$ we start a deterministic minimisation algorithm at $t = 0$, searching for a local minimum for $V(t) = -\ln\{q_1(t|u^{(1)}, x)\}$. Let $\mu_0$ denote the location of local minimum found and let $\Sigma_0 = (\nabla^2 V(\mu_0))^{-1}$ denote the inverse of the Hessian matrix evaluated at $\mu_0$. We use $\mu_0$ and $\Sigma_0$ as the mean vector and covariance matrix, respectively, in the first mixture component. We want to use a similar procedure to approximate the second large mode in (17), but then need to use another starting value for $t$ in the minimisation algorithm. From the results discussed in Section 4 we know that the maximal value for $\pi(x + \sum_{i=1}^k t_i u_i)$ (as function of $t$) is located close to the line going through the two larger local maxima of $q_1(t|u^{(1)}, x)$. Moreover, the maximal value for $\pi(x + \sum_{i=1}^k t_i u_i)$ is located between the two large local maxima of $q_1(t|u^{(1)}, x)$. A natural initial value for the second minimisation run is therefore $t = \mu_0 + 2(\mu_\pi - \mu_0)$, where $\mu_\pi$ is the location of the local maximum of $\pi(x + \sum_{i=1}^k t_i u_i)$. Letting $\mu_1$ denote the location of the resulting second local minimum found for $V(t)$, and letting $\Sigma_1 = (\nabla^2 V(\mu_1))^{-1}$ denote the corresponding inverse Hessian matrix, our approximation to $q_1(t|u^{(1)}, x)$ is

$$\widehat{q}_1(t|u^{(1)}, x) = \frac{q_1(\mu_0|u^{(1)}, x)\mathrm{N}(t; \mu_0, \Sigma_0) + R \cdot q_1(\mu_1|u^{(1)}, x)\mathrm{N}(t; \mu_1, \Sigma_1)}{q_1(\mu_0|u^{(1)}, x) + R \cdot q_1(\mu_1|u^{(1)}, x)} \tag{28}$$

where $\mathrm{N}(t; \mu, \Sigma)$ is the density of the Gaussian distribution with mean vector $\mu$ and covariance matrix $\Sigma$ evaluated at $t$, and $R$ is a scalar constant that can make proposals to the mode far away from the current state more or less likely. The results in Roberts et al. (1997) and Roberts and Rosenthal (1998) indicate that a value of $R > 1$ should be preferable. For $a = 0$ simulations show that the acceptance rate is close to one for all choices of $R$. In this case it is therefore best always to propose the mode away from the current state, i.e. use a large value for $R$. For $a \neq 0$ the situation is more complicated. In all the simulations of the directional MH algorithm we have tuned the parameter $R$ to the optimal value in terms of mean jump

length. It turns out that high acceptance rates, over 50%, works best. Before presenting further simulation results for the proposal distribution just defined we give a few remarks.

*Remark* 1. Simulation show that for $R = 1$ the algorithm results in acceptance rates over 95% for $k = 1, 2, 3, 4$, so the approximation in (28) is good.

*Remark* 2. If we use a quasi-Newton optimisation algorithm to find $\mu_0$ and $\mu_1$, the algorithm in addition to finding the minimum also build up an approximation to the Hessian matrix that can be used as $\Sigma_0$ and $\Sigma_1$.

*Remark* 3. We motivate our approximation $\widehat{q}_1(t|u^{(1)}, x)$ from the results in Section 4, that is for $k = 2$. However, we have done similar studies for $k = 3$, and also there $q_1(t|u^{(1)}, x)$ typically seems to contain two large modes.

### 5.2 Performance of the directional MH algorithm for $k \geq 1$

To evaluate the performance of the directional MH algorithm with proposal distribution $\widehat{q}_1(t|u^{(1)}, x)$ we run the algorithm for $k = 1, 2, 3, 4$ and compare the results with the results from the simpler random walk (RW) and Langevin (L) proposal algorithms. We tuned the parameters to get close to the optimal acceptance rates of 0.23 and 0.57, respectively, calculated in Roberts et al. (1997) and Roberts and Rosenthal (1998). The results are summarised in Figure 2 and Table 1. In Figure 2 we give estimated autocorrelation functions for the various algorithms considered. The left column shows the autocorrelation as a function of iterations, whereas in the right column the autocorrelation is shown as function of number of target density evaluations. The three rows show results for $a = 0$, $a = 0.1$ and $a = 0.3$, respectively. Table 1 gives some key attributes for the runs. For $a = 0$ we see that the directional MH works better then both RW and L, both when measured in number of iterations and in number of target density evaluations. For $k > 1$ we even get negative lag one correlations. For $a = 0.1$ the directional MH algorithms are better when measured in number of iterations, but when evaluated in terms of the number of target density evaluations the Langevin algorithm is clearly preferable. For $a = 0.1$ the directional MH algorithm with $k = 1$ and RW performs equally good in terms of the number of target density evaluations. The directional MH with $k > 1$ are less favourable in this case. Note in Table 1 that using $k > 1$ does not generate much longer jumps than $k = 1$. As $k > 1$ requires more target density evaluations than $k = 1$, the $k > 1$ is less attractive in this case. For $a = 0.3$ the qualitative performances of the various algorithms are similar to with $a = 0.1$, but the differences are larger.

## 6 Closing remarks

Directional MH algorithms are generalised to propose new states in a hyperplane in this article. The hyperplane is specified by the current state and auxiliary variables. We consider two types of auxiliary variables, one is uniquely given by the hyperplane and the other is not. Consistent with the findings in Eidsvik and Tjelmeland (2006), our analysis indicates that it is preferable to use the auxiliary variables that is uniquely given by the hyperplane. However, we are not able to do the analytical analysis necessary to implement the MH algorithm in this case. Therefore we instead consider the MH algorithm corresponding to the auxiliary variables that are not uniquely defined by the hyperplane and compare its performance with frequently used simpler alternatives in a simulation example. The experience is that hyperplane algorithms on average produce larger jumps in the sample space and thereby better mixing properties per

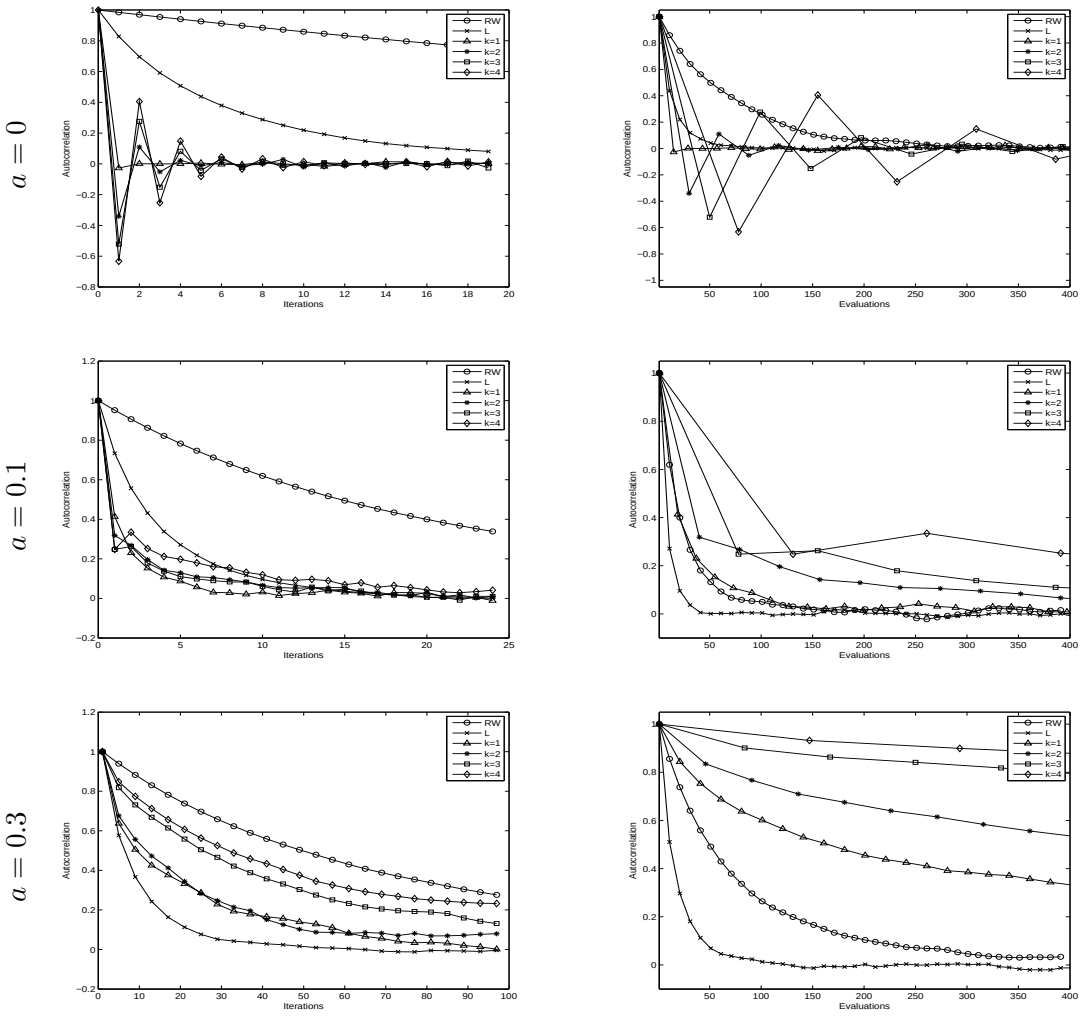


Figure 2: Estimated autocorrelation functions for random walk and Langevin proposal MH algorithms, and for our directional MH algorithm with $k = 1, 2, 3, 4$. Left column: Autocorrelation as a function of iteration. Right column: Autocorrelation as a function of number of target evaluations. The upper, middle and lower rows shows results for $a = 0$, $a = 0.1$ and $a = 0.3$, respectively.

Table 1: Attributes for the random walk and Langevin proposal MH algorithms, and for the directional MH algorithm for $k = 1, 2, 3, 4$

| | Mean number of target evaluations per iteration | Mean acceptance rates | Mean jump length | Evaluations per independent sample |
|---|---|---|---|---|
| $\underline{a = 0:}$ | | | | |
| RW | 1 | 0.23 | 0.33 | 300 |
| L | 2 | 0.54 | 1.61 | 75 |
| $k = 1$ | 14 | 0.99 | 5.5 | 30 |
| $k = 2$ | 29 | 0.99 | 6.4 | 150 |
| $k = 3$ | 49 | 0.99 | 6.8 | 300 |
| $k = 4$ | 77 | 0.99 | 7.1 | 750 |
| $\underline{a = 0.1:}$ | | | | |
| RW | 1 | 0.25 | 0.31 | 200 |
| L | 2 | 0.58 | 1.1 | 50 |
| $k = 1$ | 18 | 0.61 | 3.0 | 150 |
| $k = 2$ | 39 | 0.58 | 3.2 | 600 |
| $k = 3$ | 77 | 0.56 | 3.3 | 1200 |
| $k = 4$ | 130 | 0.56 | 3.3 | 2500 |
| $\underline{a = 0.3:}$ | | | | |
| RW | 1 | 0.25 | 0.28 | 400 |
| L | 2 | 0.58 | 0.96 | 100 |
| $k = 1$ | 20 | 0.51 | 1.0 | 2000 |
| $k = 2$ | 45 | 0.64 | 1.0 | 2700 |
| $k = 3$ | 83 | 0.73 | 0.94 | $15 \cdot 10^3$ |
| $k = 4$ | 146 | 0.80 | 0.90 | $600 \cdot 10^3$ |

iteration. However, in our implementation the hyperplane algorithms is more computation intensive per iteration and so the simpler algorithms in most cases are preferable when run for the same amount of computation time. An interesting area for future research is therefore to find variants of our directional Metropolis–Hastings algorithm that require less computation time per iteration.

## References


Chen, M. and Schmeiser, B. (1993). Performance of the Gibbs, hit-and-run and Metropolis samplers, *Journal of computational and graphical statistics* **2**: 251–272.

Eidsvik, J. and Tjelmeland, H. (2006). On directional Metropolis–Hastings algorithms, *Statistics and Computing* **16**: 93–106.

Gilks, W. R., Roberts, G. O. and George, E. I. (1994). Adaptive directional sampling, *The Statistician* **43**: 179–189.

Goodman, J. and Sokal, A. D. (1989). Multigrid Monte Carlo method. Conceptual foundations, *Physical Review D* **40**: 2035–2072.

Green, P. (1995). Reversible jump Markov chain Monte Carlo computations and Bayesian model determination, *Biometrica* **57**: 97–109.

Hastings, W. (1970). Monte Carlo sampling using Markov chains and their applications, *Biometrica* **57**: 97–109.

Liu, J. S. and Sabatti, C. (2000). Generalized Gibbs sampler and multigrid Monte Carlo for Bayesian computation, *Biometrika* **87**: 353–369.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H. and Teller, E. (1953). Equation of state calculations by fast computing machines, *Journal of Chemical Physics* **21**: 1087–1092.

Pukkila, T. M. and Rao, C. R. (1988). Pattern recognition based on scale invariant discriminant functions, *Information sciences* **45**: 379–389.

Rabben, T. E., Tjelmeland, H. and Ursin, B. (2008). Nonlinear Bayesian joint inversion of seismic reflection coefficients, *Geophysical Journal International* p. To appear.

Roberts, G. O., Gelman, A. and Gilks, W. R. (1997). Weak convergence and optimal scaling of random walk Metropolis algorithms, *Annals of Applied Probability* **7**: 110–120.

Roberts, G. O. and Gilks, W. R. (1994). Convergence of adaptive directional sampling, *Journal of multivariate analysis* **49**: 287–298.

Roberts, G. O. and Rosenthal, J. S. (1998). Optimal scaling of discrete approximations to Langevin diffusions, *Journal of the Royal Statistical Society. Series B* **60**: 255–268.

Waagepetersen, R. and Sørensen, D. (2001). A tutorial on reversible jump MCMC with a view toward application in QTL-mapping, *International Statistical Review* **69**: 49–61.

Watson, G. S. (1983). *Statistics on spheres*, Wiley-Interscience.