Journal of Data Science **11**(2013), 501-536

# An Empirical Study of an Adaptive Langevin Algorithm for Bounded Target Densities

Christopher H. Mehl
*OMNITEC Solutions, Inc.*

*Abstract*: Markov chain Monte Carlo simulation techniques enable the application of Bayesian methods to a variety of models where the posterior density of interest is too difficult to explore analytically. In practice, however, multivariate posterior densities often have characteristics which make implementation of MCMC methods more difficult. A number of techniques have been explored to help speed the convergence of a Markov chain. This paper presents a new algorithm which employs some of these techniques for cases where the target density is bounded. The algorithm is tested on several known distributions to empirically examine convergence properties. It is then applied to a wildlife disease model to demonstrate real-world applicability.

*Key words*: Adaptive MCMC, controlled MCMC, Langevin algorithms, Metropolis-Hastings algorithms, stochastic approximation, tempered MCMC.

## 1. Introduction

Markov chain Monte Carlo (MCMC) methods have become a common approach to studying posterior distributions for Bayesian statistical models. The relative ease of constructing a Markov chain that theoretically approximates a given target density makes these methods attractive. In practice, however, efficient simulation with Markov chain methods is rarely straightforward and can require ingenuity on the part of the modeller.

Difficulties can arise with complex multivariate target densities, which can contain a number of features that make simulation difficult. For example, when a target distribution has sharp ridges of high probability, a Markov chain can bypass these areas if the average step size, determined by the mean and variance of the proposal distribution, is too large. Second, with high correlation between component variables, sampling each separately is inefficient since the acceptance or rejection of one variable can have a large impact on the acceptance of another.

Finally, with bounded target densities, care must be taken to not propose outside of the support too often, since too many rejected proposals will slow down convergence of the chain.

A variety of techniques are available to alleviate some of the difficulties of sampling from multivariate distributions. Reparameterization–to remove some of the correlation of highly interdependent component variables–or blocking, which uses dependencies between parameters to accelerate convergence, are potential strategies [6, 8]. The blocking method can be especially effective in hierarchical models with spatially structured data [3, 8].

Stochastic dynamics methods are a class of algorithms motivated by stochastic differential equations, which suppress the random walk behaviour of a Markov chain. This has been shown to improve convergence results for high dimensional target densities [7]. These methods are commonly used to simulate physical systems such as quantum molecular dynamics and heat transfer. There are a number of Markov chain methods based on stochastic differential equations, many of which are specifically designed for use in statistical physics, although some have seen broader application [12]. The Langevin algorithm is perhaps the best known, and is based on the discrete simulation of a continuous time diffusion process [6, 12, 13]. Variations on the Langevin algorithm include the tempered Langevin algorithms of [17]; these ideas have been more thoroughly developed as controlled and self-tuning adaptive MCMC (see for example, [1, 16]).

Section 2 of this paper discusses Langevin and tempered Langevin algorithms. Section 3 discusses a variation inspired by the tempered Langevin algorithm for bounded densities, introduced and explored in [11], with a simple heating function specifically designed for target densities with bounded support. Section 4 applies the algorithm to a set of test case densities to compare its performance against other methods. The algorithm is applied to a hierarchical model for Chronic Wasting Disease in Rocky Mountain mule deer to demonstrate its applicability to real-world data. Finally, the results are summarized and discussed in Section 6.

## 2. Langevin and Tempered Langevin Algorithms

Langevin type algorithms are based on the discrete simulation of a continuous time diffusion process [6, 13]. A general diffusion process $X_t = X(t)$ is a continuous time stochastic process defined as the solution to the stochastic differential equation:

$$dX_t = \mu(X_t)dt + \sigma(X_t)dB_t, \tag{2.1}$$

where $B_t$ is standard Brownian motion [10]. In (2.1), $\mu(X_t)$ is a general drift term whose functional form varies depending on the type of diffusion, and $\sigma(X_t)$

is the diffusion term. For simple cases, solutions to (2.1) can be found using the Ito Calculus [4, 10]. If a solution to (2.1) does not exist or if $X_t$ tends to infinity in a finite amount of time, the diffusion process is called explosive [10].

## 2.1 The Langevin Algorithm

For a continuously differentiable target density $f$, if for some real numbers $N, a, b < \infty$, when $(\nabla f(x))^T x \leq a\|x\| + b$, for all $\|x\| > N$, then a diffusion process can be constructed that has $f$ as its stationary distribution [10, 14, 18]. The only non-explosive, reversible diffusion with this property is the Langevin diffusion which has the form:

$$dX_t = \frac{1}{2}\nabla \log(f(X_t))dt + dB_t, \tag{2.2}$$

where $B_t$ is the standard Brownian motion.

The continuous time stochastic process in (2.2) cannot be directly simulated, so a discretized version of (2.2) must be used. The simplest such discretization is the Euler discretization. If $\omega > 0$ is the size of the discretization, then the continuous time diffusion process (2.2) is approximated by a smart random walk, with steps given by:

$$X_{t+1} = X_t + \frac{\omega}{2}\nabla \log f(X_t) + \omega\epsilon_t, \tag{2.3}$$

where $\epsilon_t$ follows a multivariate normal distribution with mean 0 and covariance $I$. The random walk in (2.3) is called "smart" because proposed values are generally in directions of higher probability than the current state. (2.3) can also be written as:

$$X_{t+1} \sim \mathcal{N}(X_t + \frac{\omega}{2}\nabla \log f(X_t), \omega^2 I). \tag{2.4}$$

Other discretization schemes are discussed in [12].

Unfortunately, the behaviour of the continuous time Markov process and the discretized version may be radically different. This is because the Markov chain based on the stochastic process (2.2) samples from the correct distribution only as the discretization size goes to zero [12]. Consequently, there are situations where the random walk given by (2.3) can be transient. Situations leading to transience include those for which the target distribution is insufficiently smooth or erratic, so the gradient experiences large changes in a local area or has a large magnitude. The erratic behaviour of the gradient makes the chain unstable. Many of these problems can be corrected with a Metropolis-Hastings rejection step [2, 6, 12, 13]. This method of correcting for the discretization is called Langevin Monte Carlo [12] or the Metropolis Adjusted Langevin Algorithm (MALA) [18].

The key difference between the Metropolis Adjusted Langevin algorithm and the standard random walk is the MALA suppresses the random walk behaviour of the chain by using the local properties of the target density (i.e., $\nabla \log f(X_t)$) to adjust the proposal distribution. This nudges the chain "uphill" in the direction of a mode, which is a highly desirable property in high dimensional problems where the behaviour of the gradient is smooth [6, 7, 13, 17, 18]. The simulation can be further adjusted for any erratic behaviour of the gradient by using a truncated proposal step to achieve more robust ergodic properties [2, 18]. This truncation comes through the imposition of an upper limit on the step length in the proposal distribution [6, 15]. For example, proposals of the form $Y \sim \mathcal{N}(X_t + \min(b, (\omega\nabla \log f(X_t))/2), \omega^2 I)$ can retain most of the advantages of the Langevin algorithm while removing some of the inherent instability of the gradient. However, for target distributions with insufficiently smooth surfaces, scaling the truncated MALA can be difficult. As an example, if the posterior distribution possesses a narrow region of high probability, surrounded by an area of low probability, such as the Witches Hat distribution [13], then the chain may still overshoot the mode, resulting in the oversampling of low probability regions.

## 2.2 The Tempered Langevin Algorithm

A variation of the Langevin algorithm is the tempered Langevin algorithm, an adaptive MCMC method based on Hamiltonian dynamics [17]. The standard MALA approach uses a fixed variance coefficient ($\omega$ in (2.3) and (2.4)). A more general diffusion process can be used to derive another variation of the Langevin algorithm.

(2.2) is a special case of (2.1), where the drift term is $\mu(X_t) = \nabla \log(f(X_t))/2$, and the diffusion coefficient is $\sigma(X_t) = I$. This leads to the variance term $\omega^2 I$ in the discretization 2.4. If a more general diffusion term is used in 2.1, then the tempered Langevin diffusion is obtained. For an in depth look at the tempered Langevin diffusion see [17].

If the diffusion coefficient is $\sigma(x)$, then the diffusion matrix is given by a positive definite matrix $a(x) = \sigma(x)\sigma^T(x)$. Let $0 \leq d \leq 1/2$. Choosing

$$a(x) = f^{-2d}(x)I = \frac{1}{(f(x))^{2d}}I,$$

and setting the drift term to be

$$\mu(x) = \frac{1-2d}{2}a(x)\nabla \log(f(x))$$

gives the tempered Langevin diffusion, which is the solution to the stochastic differential equation:

$$dX_t = \frac{1-2d}{2} f^{-2d}(X_t)\nabla \log(f(X_t))dt + f^{-d}(X_t) I dB_t. \tag{2.5}$$

The diffusion in (2.5) is sometimes called a heated Markov chain. The two special cases of (2.5) which are important are $d = 0$, which gives the standard Langevin diffusion, and $d = 1/2$, which gives Brownian motion [17].

Heated Markov chains have stronger theoretical convergence properties than unheated chains, since the diffusion matrix acts as an accelerator [17]. The accelerator can alleviate some of the problems encountered with multi-modal target distributions. In areas of low probability, $a(x) = f^{-2d}(x)I$ will be larger, causing the chain to take larger steps in those areas. As the chain enters areas of high probability, $a(x)$ will decrease, leading to a smaller step size.

The discretized version of the diffusion (2.5) is given by the equation:

$$X_{t+1} \sim \mathcal{N}(X_t + \frac{1-2d}{2} a(X_t)\nabla \log f(X_t), a(X_t)\ I). \tag{2.6}$$

Using a rejection step again results in a Metropolis-Hastings algorithm. We can adjust the acceptance probability for the tempered diffusion by setting $\omega^2 = a(X_t)$, which gives:

$$\alpha(X_t, Y_t) = \frac{f(Y_t)}{f(X_t)} \frac{\exp(-\frac{\|X_t - Y_t - \mu(Y_t)\|^2)}{2\sigma(X_t)})}{\exp(-\frac{\|Y_t - X_t - \mu(X_t)\|^2)}{2\sigma(X_t)})}, \tag{2.7}$$

where

$$\mu(x) = \frac{1-2d}{2} a(x)\nabla \log(f(x)),$$

and $\sigma(X_t) = f^{-d}(X_t)$.

The tempered Langevin algorithm, like the standard Langevin algorithm 2.4, is a random walk which makes “smart jumps”. Hence, the chain is likely to propose values in the direction of an area of high probability. Unfortunately, the acceleration term, derived directly from the stochastic differential (2.7), can lead to erratic behaviour. Additionally, the tempered Langevin algorithm relies on a complicated scheme for the proposal variance, which makes it difficult to implement and computationally cumbersome [17]. For target densities with bounded support, additional difficulties arise when the value of the density function is near zero, which causes the acceleration term to be large, leading to an excessive number of proposals outside the support of the target density.

## 3. A Modified Adjusted Langevin Algorithm with Tempered Step Size

The form of the acceleration term in the tempered Langevin algorithm may be cumbersome, but the essential concept of taking larger steps in areas of low probability and smaller steps in areas of high probability is nonetheless useful. Adaptive MCMC methods such as those in [1, 16] achieve this property by adaptively changing the variance of the proposal distribution. The algorithm presented in this paper focuses on adaptively changing the step size. The dynamic step quality can also be captured using a simpler form for the acceleration term.

The main benefit of a Langevin-type algorithm is the use of the gradient in choosing a direction for the proposal value [7, 14, 17, 18]. For a target density with erratic gradient behaviour, however, the magnitude of the proposed value can be extremely large. Proposals with a large magnitude are a problem in a confined parameter space, but the gradient direction still contains useful information. The direction is retained by turning the gradient into a unit vector, which alleviates any potential problems due to the magnitude. Thus, rather than using $\nabla \log f(X_t)$, it is scaled to $d_{\text{step}} = \nabla f(X_t)/\|\nabla f(X_t)\|$, provided $\|\nabla f(X_t)\| \neq 0$. When the gradient is zero, we set $d_{\text{step}} = 0$.

Rather than attempting to find a single step size for all parameters, a dynamic step size is constructed which is qualitatively similar to the tempered Langevin algorithm. The chain is therefore “heated”, so the further the current state is from the mode of the distribution, the larger the step size that should be taken. When the current state is close to the mode, the steps must be smaller.

A expression is constructed which is at a minimum when the current state, $X_t$, is at the mode of the target density. It should get larger the further $X_t$ gets from the mode. For the stability purposes discussed above, this expression uses the log of the posterior density. Let $f$ denote the posterior density function. Let $\hat{x}$ denote the mode of the posterior density. Then for any $X_t$ in the support of $f$, we have $f(X_t) \leq f(\hat{x})$. The logarithm is a monotonic transformation, so this implies $\log(f(X_t)) \leq \log(f(\hat{x}))$. Subtracting $\log(f(X_t))$ from both sides gives $0 \leq \log(f(\hat{x})) - \log(f(X_t))$. However, since the step size in the Markov chain simulation should be non-zero, one is added to both sides. Thus, $1 \leq \log(f(\hat{x})) - \log(f(X_t)) + 1$.

Let $h(\hat{x}, X_t) = \log(f(\hat{x})) - \log(f(X_t)) + 1$, and note $1 = h(\hat{x}, \hat{x}) \leq h(\hat{x}, x)$ for any $x$. For log-concave densities, $h(\hat{x}, X_t)$ increases as the value of $\log(f(X_t))$ gets farther from the log density value at the mode. This acceleration term is bounded from below, resulting in a minimum step size.

Use of the difference between the log densities is useful for situations where the direct use of the density function can lead to numerical instability. For target densities where numerical instability is not an issue, a ratio proportional to $1/f(X_t)$, as in the tempered Langevin algorithm of [17], would also be effective. A candidate that uses the mode is the ratio $f(\hat{x})/f(X_t)$, which is always bounded below by one. Taking the logarithm of both sides of the inequality $1 \leq f(\hat{x})/f(X_t)$ and adding one results in the same expression as above. In practice, the mode in $h(\hat{x}, X_t)$ can be found using any of a host of numerical optimization procedures.

To make the step size appropriate for a given target density, both the proposal variance and the step size function $h(\hat{x}, X_t)$ must be scaled correctly. In the standard MALA, the variance of the proposal has the form $\omega I$, so it is constant for each component, although this is not necessary and it is also possible to vary the proposal variance for different components. A diagonal covariance matrix, $\Sigma$, can be used, with the entries scaled differently for each component. This is a less complicated approach than the construction of the diffusion matrix used for the tempered Langevin algorithm in [17]. The entries of $\Sigma$ must be chosen to ensure values are not proposed outside of the parameter space too often. A good rule of thumb the author has found in practice is to use values for $\Sigma$ that are proportional to the variances of the prior distribution, with the constant of proportionality chosen so that an acceptance rate of 57.4% is achieved, per the heuristic suggested by [15].

In addition to the proposal covariance, the tuning parameter $k$ must be chosen. The tuning parameter $k$ is similar to the discretization size $\omega$ in the MALA algorithm. The parameter $k$ can be chosen by calculating the step size function for points throughout the support of the target density, and choosing the values of $k$ to make $h$ as large or as small as desired. This tuning parameter gives relatively precise control over the minimum step size. Following the heuristic in [15], the tuning parameter should be adjusted to achieve an acceptance rate of 0.574.

The resulting algorithm is a modification of the tempered Langevin algorithm, called the Modified Adjusted Langevin Algorithm with Tempered Step, or MALTS.

**Algorithm 3.1.** (MALTS) Let the current state of the chain be denoted by the $m$-dimensional vector $X_t$ and the target density by $f(x)$. Let $\hat{x}$ denote the mode of the distribution $f$. Choose the $m$ entries of the $m$ by $m$ diagonal matrix $\Sigma$, and tuning parameter $k$. Let $h(\hat{x}, y) = kI(\log(f(\hat{x})) - \log(f(y)) + 1)$.

1. Set $\mu(X_t) = X_t + s_x\vec{d_x}$, where $s_x = kh(\hat{x}, X_t)$ is the step size at $X_t$, and $\vec{d_x} = \nabla \log(f(X_t))/\|\nabla \log(f(X_t))\|$, is the unit vector in the direction of the gradient at $X_t$. If $\nabla \log(f(X_t)) = \vec{0}$, then set $\vec{d_x} = \vec{0}$.
2. Generate $Y \sim \mathcal{N}(\mu(X_t), \Sigma)$.
3. Let $\mu(Y) = Y + s_y\vec{d_y}$ be calculated as is step 1.
4. Let $\alpha(X_t, Y) = \frac{f(Y)}{f(X_t)} \frac{\exp(-0.5*(X_t-\mu(Y))^T\Sigma^{-1}(X_t-\mu(Y)))}{\exp(-0.5*(Y-\mu(X_t))^T\Sigma^{-1}(Y-\mu(X_t)))}$.
5. With probability $\min(\alpha, 1)$, set $X_{t+1} = Y$. Otherwise, set $X_{t+1} = X_t$.

Convergence of Algorithm 3.1 to stationarity can be established by demonstrating it is a special case of the Metropolis-Hastings algorithm. The relationship between the target density and the conditional proposal distribution in the Metropolis-Hastings algorithm must be established. Basic convergence to the target distribution is stated in the following theorem:

**Theorem 3.1.** Let $\mu(x)$ and $\Sigma$ be defined as above in Algorithm 3.1. Then, if the log of target density $f$ has a connected support contained by the set of real numbers, and continuous first partial derivatives throughout its support, the Markov chain produced by 3.1 will converge to the stationary distribution $f$.

**Proof.** The continuous first partial derivatives are necessary to ensure the existence of the gradient throughout the support of the target density. The gradient is used in determining the direction of the proposal value. The unnormalized conditional density is given by:

$$q(x|y) = \exp(-0.5 * (Y - \mu(X_t))^T \Sigma^{-1} (Y - \mu(X_t))). \tag{3.1}$$

This is the kernel of an $m$-dimensional multivariate normal distribution, with support $\mathbb{R}^m$.

Recall a set $A$ is connected if every two points $z_1, z_2$ in the set $A$ can be connected by a piecewise smooth curve entirely contained within $A$. The connected support of the target density is not absolutely necessary, but this condition avoids many mathematical and practical difficulties, so it is assumed here. Let $supp(f)$ denote the support of the target density $f$. If $supp(f) \in \mathbb{R}^m$, then convergence of the Markov chain produced by Algorithm 3.1 to the correct stationary distribution is ensured. See [13] for a proof of the Metropolis-Hastings convergence. □

The relative performance of the MALTS algorithm on several cases of test distributions will be examined in Section 4. Theorem 3.1 ensures basic convergence to stationarity for Algorithm 3.1. However, actual convergence rates for a given target density will depend on the values of the elements of $\Sigma$ and the specific characteristics of the target density. Because the algorithm suppresses the random walk behaviour of the chain by proposing, on average, in an uphill direction, the rate of convergence will generally be better than the standard random walk as the number of dimensions increases [7, 13]. Like the MALA, the MALTS algorithm uses the gradient information to pick a direction for the proposed value. These algorithms are more likely to accept proposed values than a standard random walk, because Langevin type algorithms modify the search by biasing the proposal distribution in favour of candidate states that lie in directions of higher probability.

Note if a target density is flat, as with a uniform distribution, then the gradient vector will be zero. In such a case, the proposal direction will be the zero vector, so the proposal mechanism in Algorithm 3.1 reduces to a standard random walk. With a locally flat density the gradient contains no information about regions of higher probability, but the algorithm will still function. Given the similarities with the tempered Langevin diffusion, the convergence of the chain produced by this algorithm may be better than the unheated Langevin algorithm in (2.4).

MALTS is essentially a random walk with a smart proposal step, which makes use of the gradient information while removing potential instabilities in the gradient. Additionally, the properties of $h(\hat{x}, X_t)$ cause the chain to take larger steps in areas of low probability, and smaller steps in areas of high probability. However, the algorithm requires at least some knowledge of the mode of the target density, which may require the use of computationally intensive optimization procedures before the method can be implemented. Finding the mode of the target distribution can become difficult as the number of dimensions increases. If a poor approximation of the mode is used in the MALTS algorithm, the dynamic step size can bias the proposal in the wrong direction. Let $\tilde{x}$ be a poor approximation to the mode of the target density $f$, and suppose the current state of the chain is $X_t$ is closer to the true mode, then $\log(f(\tilde{x})) - \log(f(X_t)) + 1$ may be negative. This would have the effect of pulling the chain in the wrong direction, away from the areas of higher probability. A corrective measure to this potential problem is to update the mode approximation during the simulation by checking at each iteration if $\log(f(\tilde{x})) < \log(f(X_t))$, and setting $\tilde{x} = X_t$ as the new approximation to the mode if this is true.

As with other Langevin type algorithms, the use of the gradient would make the MALTS algorithm a poor performer with multi-modal target densities. While the algorithm would theoretically converge, it would be expected to mix poorly and be a poor performer in practice, as it would have a tendency to get stuck on one of the modes. This is no different from any other Langevin type algorithm however.

Additionally the algorithm can have a comparatively large number of parameters, i.e., the $m$ entries of the diagonal matrix $\Sigma$ and the constant $k$ for an $m$-dimensional distribution. This added complexity provides the benefit of fine tuning the algorithm for cases where the scale of components in a multivariate density are significantly different, and giving the algorithm more freedom to explore complicated and tightly bounded target densities. It should be noted, however, like the MALA, the MALTS algorithm can be simplified and the number of parameters significantly reduced to require only a single tuning parameter, as $\Sigma$ can be chosen to be an identity matrix.

The complexity of the MALTS algorithm is comparable to the MALA. If the gradient is approximated numerically, then for each iteration, both algorithms require $2m + 1$ evaluations of the density function for an $m$-dimensional target distribution. By contrast, the random walk algorithm requires only one density evaluation per iteration. Each iteration of a standard random walk algorithm will therefore require less time than either MALTS or MALA. The advantage of Algorithm 3.1 or the MALA is that a standard random walk can require an extremely large number of iterations to converge to stationarity [12, 13].

Considerations for choosing the tuning parameters for the MALTS algorithm are the same as those for other methods. For bounded densities, the tuning parameter $k$ in Algorithm 3.1 must be chosen to prevent the chain from frequently proposing values that lay outside the parameter space. For unbounded densities, $k$ and $\Sigma$ must be chosen to ensure sufficient exploration of the parameter space. The dynamic step size $h$ must be also be considered, and the tuning parameter $k$ must balance the behaviour of the step size equation $h$. The variances of the prior distributions are generally easy to calculate and make convenient choices for the entries of the proposal variance matrix $\Sigma$.

Another natural choice for the elements of $\Sigma$ are the preliminary estimates of the marginal posterior variances. Since the MALTS algorithm requires the calculation or approximation of the mode of the target density, this information can be conveniently used in the choice of the tuning parameters. The inverse of the Hessian matrix of the log-posterior density, evaluated at the mode, provides an analytical approximation of the posterior covariance matrix. The values of the approximate covariance matrix can then be used to choose the values of the matrix $\Sigma$. If desired, $\Sigma$ can be diagonal. Using the prior or posterior marginal variances alone may lead to step sizes either too large or too small. Consequently, the magnitude of the entries of the matrix may need to be increased or decreased. This is especially important for bounded densities, since efficient Markov chain samplers should predominantly propose values within the support of the target density. If $h(\hat{x}, x)$ is easy to maximize on the support of the target density, then an upper bound for the step size can be established, to aid in choosing the tuning parameter $k$. A simple alternative is to calculate $h(\hat{x}, x)$ for several points in low probability areas on the support of the target density, and then scale the values of the step matrix to achieve a particular step size.

A common consideration when choosing tuning values for Markov chain simulations is the acceptance rate, i.e., the proportion of proposed moves that are accepted. Low acceptance rates can indicate the step size is too large, since for many densities, a large move is extremely unlikely to be accepted [6, 13]. Smaller step sizes typically lead to larger acceptance rates, since proposed moves extremely close to the current state are very likely to be accepted. As an example, as the step size $\omega$ in the Langevin algorithm 2.3 approaches zero, the discrete Langevin random walk approaches the continuous time Langevin diffusion, in which moves are always accepted. While a high acceptance rate indicates the chain is making more moves, the small step size has the drawback of causing the chain to move about the parameter space very slowly, providing an incomplete picture of the target density. There is some theoretical justification for an acceptance rate in the range $[0.15, 0.5]$. These values are based on limiting acceptance rates for high dimensional densities. If possible, the value of the tuning parameter $k$ should be adjusted to achieve an acceptance rate in this range. Theoretical studies have indicated the optimal acceptance rate for the MALA is near 0.5 [7, 13, 15], so the optimal acceptance rate for the MALTS Algorithm 3.1, may be near 0.5 as well.

## 4. Empirical Results for Known Target Densities

Fundamentally, a Markov chain algorithm should accurately simulate the correct target density, or at least a provide good approximation to it. This can be checked both theoretically, as in Theorem 3.1, and empirically. Once there is confidence a method will produce the correct density, there are questions of performance.

A simple test of accuracy can be performed by using the MALTS algorithm to sample a density with known parameters. The normal distribution truncated to $[0, 1]$, with mean $(0.5, 0.5)^T$ and covariance matrix $0.001 * I$, chosen so that the majority of the density is concentrated around the mode of the distribution, provides an opportunity to test Algorithm 3.1 on a bounded density with known parameters. The distribution was simulated using Algorithm 3.1. Four chains were run, with four initial values chosen to be dispersed about the parameter space. These starting points were: $(0, 0)$, $(0.7, 0.1)$, $(0.1, 0.7)$, and $(0.9, 0.9)$. The proposal variance for the MALTS algorithm was a diagonal matrix with the step size $1 \times 10^{-10}$ for each variable, and the tuning parameter was also $k = 1 \times 10^{-10}$. The results of the simulation are shown in Figure 1. The majority of the points are concentrated around the mode of the target density, providing graphical evidence that the method has sampled the correct density.

Gelman and Rubin's potential scale reduction is a comparison of the variance between runs and within runs, denoted by $\sqrt{\hat{R}}$. This diagnostic is used to assess the convergence of a Markov chain simulation to stationarity, and should decrease to one as the number of iterations increases for all parameters of interest [5]. The potential scale reduction for the truncated normal simulation is calculated using the fours runs of the MALTS algorithm, and the plot of the statistic $\sqrt{\hat{R}}$ versus iterations is shown in Figure 2. The graph shows the potential scale reduction decreasing to one as the number of iterations increases giving evidence the simulated Markov chain is reaching approximate stationarity. A common rule of thumb for the potential scale reduction is that it should be less than 1.2 for each parameter, which occurs within 2000 iterations, indicating this is an acceptable burn-in period. Additionally, the posterior means averaged over all four runs and adjusted for a 2000 iteration burn-in period were 0.5010 for $\theta_1$, and 0.5001 for $\theta_2$. The 95% Bayesian credible intervals averaged over all four simulations are $(0.4925, 0.5110)$ and $(0.4840, 0.5010)$, indicating the algorithm was able to recover the correct parameters.

Having determined the basic accuracy of the MALTS algorithm, the next step was comparing the performance of Algorithm 3.1 with other methods, in particular the standard random walk and the MALA. There are a number of criteria for assessing the performance of a Markov chain simulation, including run time, convergence to stationarity, convergence to independent sampling, and mixing speed. The relative importance of the different criteria is not widely agreed upon in the literature, and for this work the focus was first on the number of iterations an algorithm required to converge to stationarity. Second was how well the algorithm mixes. Third was the amount of time required for the algorithm to converge.

Theoretical results on optimal algorithm performance have centered on two questions: what is the optimal step size, and how well does the algorithm perform at this optimal step size. Theoretical results on these two questions are limited, and have relied on simplifying assumptions such as independence or identically distributed components for an $m$-dimensional target density, examining the limiting behaviour as $m$ diverges to infinity. Under the assumption of independent components, the step size of the standard random walk scales with the dimension as $m^{-1}$, and leads to an limiting acceptance rate of 23% as $m \to \infty$ [7, 13]. By contrast, the Langevin algorithm has a step size which scales as $m^{-1/3}$, and leads to a limiting acceptance rate of 57% [7, 14, 15]. The higher acceptance rate and larger step size indicates Langevin type algorithms will tend to move about the parameter space more than a standard random walk.

There is little theory for more complex scenarios, such as sampling multivariate distributions with correlated components. In such cases empirical methods have been used to compare algorithms [7, 16]. A common approach is to fix the step size and examine the acceptance rates for different algorithms. For a given step size, a higher acceptance rate is an indication an algorithm is making more moves about the parameter space, and consequently mixing better than an algorithm with a lower acceptance rate. The acceptance rate is only a rough measure of how well a chain is mixing, but large differences in acceptance rates would indicate differences in the mixing speed.

To compare the performance of the MALTS, MALA, and standard random walk, a set of experiments were performed comparing both the mixing speed and the convergence to stationarity of each algorithm. The experiments examined a bounded and an unbounded group of distributions. The bounded group of distributions were multivariate normal distributions truncated to the unit cube, while the unbounded group of distributions were highly correlated multivariate normal distributions. These distributions form a reasonable test bed, as they have been used by other authors as prototypical target distributions; see, for example [6, 7]. For each distribution within a group, $80,000$ iterations of a Markov chain simulation were run from five randomly chosen initial values, using the standard random walk, the MALA, and the MALTS algorithm. Each method used multivariate rather than univariate updating, so at each iteration, all of the components were updated. The potential scale reduction was calculated for the means of the first and second components (first order convergence) and for the variances of the first and second components (second order convergence) using the five chains for each simulation method to assess the convergence to stationarity, both in terms of the number of iterations and time. Additionally, acceptance rates were calculated to determine how well the chains produced by each simulation method were mixing.

For the truncated normal distributions, the components were independent, each with mode 0.89. Truncated normal distributions provide a classic case of a bounded density. The variance of the first component was set at $1\times10^{-3}$, and the variance of the remaining components was set at $1\times10^{-5}$. Four sets of simulations of increasing dimension were run within this group. The dimensions were 2, 6, 12, and 36, and the step size parameter was fixed at $1 \times 10^{-5}$ for each method. Acceptance rates for each truncated normal simulation are given in Appendix 1. The results for the simulation show the standard random walk and MALTS algorithms produced comparable acceptance rates for each dimension, indicating both methods were mixing reasonably well. Both the standard random walk and the MALTS algorithm had acceptance rates nearly 5% higher than the MALA for all dimensions, indicating these methods were accepting more proposed moves and exploring more of the parameter space.

Figure 3 shows the plot of the log potential scale reduction versus iteration for each method for the mean first component, with variance $1\times10^{-3}$, and the second component, with variance $1 \times 10^{-5}$, for the 36 dimensional truncated normal simulation. For both components, first order convergence to stationarity for the MALA and MALTS algorithms occurred in fewer iterations than the standard random walk. The MALA converged in fewer iterations than Algorithm 3.1 for the first component, but the two algorithms performance was nearly identical for the second component. The results for second order convergence, shown for the

36 dimensional truncated normal in Figure 4, were similar. For the truncated normal distribution, Algorithm 3.1 provided a middle ground, with convergence to stationarity comparable to the Langevin algorithm and in fewer iterations than a standard random walk, and a mixing speed better than the Langevin algorithm and comparable to the standard random walk.

The amount of time it takes to effectively run an algorithm is another important consideration. For a two dimensional truncated normal simulation, the random walk simulation required an average of $5.0 \times 10^{-4}$ seconds for each iteration, compared to $23.0 \times 10^{-4}$ seconds for the Langevin and MALTS algorithms. Figure 5 shows the plot of the log potential scale reduction versus seconds for each algorithm for the 36 dimensional truncated normal distribution. The longer time per iteration results for the MALTS algorithm resulted in a longer amount of time to reach convergence for the first component. The standard random walk and MALA achieved stationarity in a much shorter amount of time. For the second component, however, the MALTS and MALA clearly reached stationarity faster than the standard random walk. Figure 6 shows the results for second order convergence. The longer per iteration time of $23.0 \times 10^{-4}$ for the MALTS and MALA clearly had a significant impact, but only on the order of seconds. Despite the time advantage of the standard random walk, the fewer iterations required by the derivative using methods (MALA and MALTS) would result in fewer wasted iterations, requiring less memory resources to achieve a usable sample.

For the unbounded multivariate normal distribution, the components were chosen to be highly correlated. Each distribution had a mean of zero and covariance $(1 - \rho)I_k + \rho J_k$, where the matrix $I_k$ is the $k$-dimensional identity, $J_k$ is a matrix of all ones, and $\rho$ is the correlation between parameters. Three levels of correlation were examined, $\rho = 0.879$, $0.970$, and $0.992$, which are the same correlation levels examined in [7]. For each of these correlation values, four sets of simulations were run, with dimensions 2, 6, 12, and 36. Three step size parameters were used for each method, $1 \times 10^{-5}$, $1 \times 10^{-4}$, and $1 \times 10^{-2}$.

For the step size $1 \times 10^{-5}$, correlation $\rho = 0.879$, there was little difference in the acceptance rates. At this correlation level, each of the methods had an acceptance rate above 95% for each dimension, indicating that at this level of comparison, all of the methods were mixing at approximately the same speed, and accepting roughly the same number of proposals. However, the graphs of the log potential scale reductions by iteration, shown in Figures 7 and 13, show differences in the number of iterations in which the methods converge to stationarity. For the two dimensional simulation, with $\rho = 0.879$, the MALTS algorithm converged to stationarity in far fewer iterations than the Langevin and standard random walk, and as the dimension increases to 36 (Figure 7), the MALTS algorithm still required fewer iterations to converge than the other two algorithms. For second order convergence, shown in Figure 8, the results are similar. The acceptance rates for all of the simulations are given in Appendix 2.

For the two dimensional simulation, the standard random walk required $3.0 \times 10^{-4}$ seconds per iteration, compared to $8.0 \times 10^{-4}$ for the MALA and $9.0 \times 10^{-4}$ for the MALTS algorithm. In spite of the longer time per iteration required by the MALA and MALTS methods, the performance of the three algorithms was similar. The plot of the log potential scale reduction versus time is shown in Figure 9. The graphs for both the first and second parameters are similar, showing little differences between the algorithms. The graphs for second order convergence, shown in Figure 10, show the standard random walk and MALTS methods took less time to converge than the MALA. Increasing the number of dimensions to 36 produced similar results, although with the shorter time per iteration the standard random walk should have had a distinct advantage. The standard random walk required $3.1 \times 10^{-4}$ seconds per iteration, while a single iteration of the MALA required $8.2 \times 10^{-4}$ seconds and the MALTS algorithm required $9.2 \times 10^{-4}$ seconds per iteration. Figures 11 and 12, showing the plots of the log potential scale reduction by time, indicated the three algorithms converged at approximately the same rate. For the second order convergence shown in Figure 12, the differences were barely noticeable.

Increasing the level of correlation had a noticeable effect, as demonstrated in Figures 13 and 14, which show with correlation of 0.992, there was a noticeable effect on the potential scale reduction of the second component for the MALTS and MALA algorithms. The plot of the log potential scale reduction by iteration for the standard random walk did not decay to zero as completely as it did for the other methods. The standard random walk experienced difficulties with second order convergence, although the time advantage of the standard random walk by iteration became more apparent at the higher correlation level. The acceptance rates for each algorithm remained quite high at this step size and correlation level $\rho = 0.992$. As the dimension increased, differences in the acceptance rates became more apparent. Simulations conducted at a correlation of 0.992 for dimensions 64, 144, 200, and 300 highlighted the differences between the algorithms. Figure 15 shows the graphs of the log potential scale reductions for the 200 dimensional normal distribution simulation. As the number of iterations increased, $\log(\sqrt{\hat{R}})$ decreased rapidly for $\theta_1$ for both the MALA and the MALTS algorithms, compared to the standard random walk. For the parameter $\theta_2$, the log potential scale reduction factor for the MALA decreased more rapidly than the standard random walk, and then appeared to experience difficulties, flattening out as the number of iterations increases. The log potential scale reduction factor of $\theta_2$ for the MALTS algorithm also initially decreased faster than the standard random walk, but unlike the MALA, it continued to decrease as the number of iterations increased. By this measure, the MALTS algorithm is clearly reaching stationarity in fewer iterations than the other two methods. The time advantage of the standard random walk disappeared at the higher dimension, as demonstrated in Figure 16.

For the higher dimensional distributions, there was a difference in the acceptance rates as well. As the number of dimensions is increased from 36 to 64, the acceptance rates dropped significantly, from above 95% for each method to 88.7%, 88.0%, and 85.1% for the standard random walk, MALTS, and MALA, respectively. As the number of dimensions increased further, the acceptance rate for the standard random walk decreased, while the acceptance rates for the MALA and MALTS held steady. The increased dimension had less of an impact on the derivative using Langevin type algorithms. Figure 17 shows the plot of the acceptance rate versus dimension. All three simulation methods experienced a noticeable decline in acceptance rates as the dimension increased from 36 to 64, from above 95% to less than 90%. Importantly, as the dimension increased further, the acceptance rate for the standard random walk continued to decline, while the acceptance rates for the MALA and MALTS algorithms remained steady at 85% and 88%, respectively, which indicated the increased dimension had less of an impact on the mixing of the MALA and MALTS algorithms.

The small step size of $1 \times 10^{-5}$ caused nearly every proposal to be accepted. The slightly lower values for the acceptance rates of the MALA and MALTS algorithms were because the chains produced by these methods will on average take larger steps for a fixed step size parameter than the standard random walk. Recall $h(\hat{x}, x)$ in Algorithm 3.1 has a minimum value of one, so that the step size parameter of $1 \times 10^{-5}$ is the smallest step size proposed for MALTS. A similar characteristic exists for the MALA, where the actual step size depends on the magnitude of the gradient at the current state. Increasing the step size by a factor of ten to $1 \times 10^{-4}$ resulted in a decrease in the acceptance rates, given in Appendix 2. The standard random walk had a higher acceptance rate than the MALA and MALTS, but the three methods were within 3% of each other, indicating they were mixing at about the same rate. The number of iterations required for convergence, shown in Figure 18, was again less for the MALA and MALTS algorithms than for the standard random walk, although the differences were slight. The plot of the log potential scale reduction versus time in Figure 19, shows that at this step size the three methods were equally fast. Both the mixing speed and the speed of convergence were nearly equal for the step size of $1 \times 10^{-4}$.

For the sake of comparison, the step size was increased to $1 \times 10^{-2}$. The larger step size results in a decrease in the acceptance rate of about 10% for each method. Figure 20 shows the plot of the log potential scale reduction versus number of iterations for the mean and variance of parameter $\theta_1$. The graphs show the MALTS and MALA methods achieved first and second order convergence in fewer iterations than the standard random walk. The time plots in Figure 21 show the three methods took about the same amount of time to reach first order convergence, while the MALA and MALTS methods achieved second order convergence faster than the standard random walk. Thus the derivative using methods (MALA and MALTS) converged to stationarity at the same rate or faster than the standard random walk, and with fewer wasted iterations.

These experiments provided an examination of the comparative mixing and convergence properties of the standard random walk, Metropolis adjusted Langevin algorithm, and Modified adjusted Langevin algorithm with tempered step size. The results here indicate Algorithm 3.1 will perform at least as well as the MALA and standard random walk, and for some target densities can outperform these methods. These experiments used numerical approximations of the gradient in the MALA and MALTS algorithms, which required log posterior evaluations and slowed down the algorithm. Use of the analytical gradient would speed up the algorithms, reducing the amount of time needed for each iteration.

## 5. Application of MALTS to a Model for Chronic Wasting Disease

Having tested Algorithm 3.1 on several test cases, we next examine its performance simulating the posterior distribution from a spatio-temporal model for Chronic Wasting Disease in mule deer in the state of Colorado, introduced in [9]. Chronic Wasting Disease (CWD) is a fatal, transmissible spongiform encephalopathy found in North American mule deer, white-tailed deer, and elk. Data for the model consisted of CWD prevalence data for each DAU collected by the Colorado Division of Wildlife over a 27 year period, from 1976 to 2002. For a description and references on CWD see [9, 11].

The CWD model in [9] is derived from a differential equation for the dynamics of the prevalence, which is the proportion of animals infected with the disease. The model is extensively described in [9] and will be only briefly outlined here. The discretized model is given by:

$$\boldsymbol{p}_{t+1} = \alpha(\delta\mathbf{1} - \boldsymbol{p}_t) \odot \boldsymbol{p}_t \odot (\mathbf{1} - \boldsymbol{p}_t) + W\boldsymbol{p}_t. \tag{5.1}$$

where $\boldsymbol{p}_t$ is the vector of prevalence values, $W$ is a spatial mixing matrix that represents disease migration from area $j$ to area $i$, and $\odot$ is the term-by-term Hadamard matrix product.

The parameter $\alpha$ represents a kind of acceleration for the disease prevalence, and $\delta$ is the long term proportion of infected deer that the system can sustain, playing a role similar to that of the carrying capacity in the standard logistic equation for population growth. The structure of the matrix $W$ is based on the

Colorado Division of Wildlife's map of their deer data analysis units, or DAU's. Let $W_{ij}$ denote an element of the stochastic transition matrix $W$. For parsimony, fix $W_{ij} = \gamma$ if DAU$_i$ and DAU$_j$ are first-order neighbors with a "significant" proportion of boundary that touch. If DAU$_i$ and DAU$_j$ have less of boundary that touch, or are almost touching, then $W_{ij} = \omega < \gamma$. For DAUs that do not share a boundary and hence are not neighbors, $W_{ij} = 0$. The diagonal values of $W$ are fixed so that row sums of $W$ are equal to 1. Thus, the $W_{ij}$ can be loosely interpreted as the average proportion of deer that migrate from DAU$_i$ to DAU$_j$ each year. $\alpha$, $\delta$, $\gamma$, and $\omega$ are the dynamic parameters in the model.

In addition to the four dynamic parameters, there are the initial conditions–DAU's where the disease was present at the beginning of the time period in 1976. Through a combination of prior knowledge and model comparison via Bayes' factors, the particular model chosen in [9] was that with dynamic parameters given by:

$$\boldsymbol{\eta} = (\alpha, \delta, \gamma).$$

The initial conditions were given by the following:

$$\boldsymbol{p}_0 = (p_{10,0}, p_{4,0}, p_{5,0}),$$

so that $\boldsymbol{\theta} = (\boldsymbol{\eta}, \boldsymbol{p}_0)$. It was necessary to find the posterior distribution of $[\boldsymbol{\theta}|\boldsymbol{Y}]$, where $\boldsymbol{Y}$ denotes the data. Because this distribution does not have a convenient form, computational methods were needed to simulate the posterior distribution.

The posterior distribution was originally simulated using an acceptance sampler based on independent beta distributions, the results of which are shown in Figure 22 for the parameters $\alpha$ and $\delta$. The posterior distribution associated with the above model has several distinctive features which made simulation difficult and slow. The contour plot shows the component variables in the model are highly interdependent, the parameter space is bounded, and the distribution has a narrow ridge of high probability.

The acceptance sampler was written in Matlab and run on a machine with a 2.2Ghz processor and 3GB RAM. The sampler generated a random value from the posterior roughly every 3 seconds, and took nearly 24 hours to generate a sample of size 25,000. The long run time was likely due to the complicated nature of the parameter space. The slowness of the acceptance sampler made it impractical for studying possible variations of the model, or for applying to future updated data sets.

The features of the posterior which made it difficult to simulate with an acceptance sampler mad0e it a good potential real data case for Algorithm 3.1. Attempts were made to apply the standard random walk, the MALA, and a tempered MALA to the CWD model, but each of these experienced serious difficulties, and extensive efforts to scale them appropriately did not yield success.

None of the algorithms were able to converge to stationarity, even after more than 300,000 iterations.

The MALTS algorithm by contrast was able to accurately and quickly reproduce the posterior density in [9]. 300,000 iterations on the same machine used for the acceptance sampler took approximately 40 minutes, a significant improvement over the acceptance sampler. Calculation of the potential scale reduction showed that the Markov chain produced by Algorithm 3.1 converged to stationarity. Visual inspection of the graphs of the potential scale reduction for all of the model parameters, shown in Figure 23, indicates stationarity occurs around the 8000th iteration.

To ensure that the chain converged to the correct distribution, the marginal distributions produced by the Markov chain sampler were compared to the marginal distributions produced by the acceptance sampler. The comparisons of the marginal distributions of the density produced by Algorithm 3.1 to those produced by the acceptance sampler were done by means of the Kolmogorov-Smirnov test, which indicated the two densities were the same. Figure 24 shows the distinctive banana shape for the $\alpha$ by $\delta$ contour plot, and provides further evidence of the MALTS algorithm's success in producing the target distribution. After the burn-in period seen in the lower left corner of Figure 24, the contour plots looks identical to Figure 22. The visual evidence coupled with the Kolmogorov-Smirnov results on the marginals shows Algorithm 3.1 was successful in producing the correct posterior distribution.

Because of the speed of Algorithm 3.1, multiple runs can be combined to better approximate an independent sample [5, 6]. Six runs of 300,000 iterations were run from six different initial values, and the last 50,000 iterates from each run were combined into a single sample for posterior inference. Posterior Monte Carlo estimates and 95% Bayesian credible intervals are shown in Table 1; all the values have been multiplied by 100. These estimates are in line with those found from the acceptance sampler in [9].

Table 1: Posterior estimates and 95% credible intervals for the CWD model produced by the MALTS algorithm; values have been multiplied by 100

| Parameter | Lower Limit | Upper Limit | Mean | Median | StdDev |
|---|---|---|---|---|---|
| $p_{10,0}$ | 1.73 | 6.34 | 3.58 | 3.41 | 1.28 |
| $p_{4,0}$ | 0.86 | 3.71 | 1.92 | 1.78 | 0.88 |
| $p_{5,0}$ | 0.08 | 0.78 | 0.34 | 0.30 | 0.28 |
| $\alpha$ | 5.90 | 20.76 | 12.27 | 11.96 | 3.76 |
| $\delta$ | 11.32 | 21.00 | 15.24 | 14.96 | 2.55 |
| $\gamma$ | 0.63 | 1.10 | 0.85 | 0.84 | 0.12 |

## 6. Conclusion

Using existing Markov chain simulation methods based on stochastic dynamic processes as a starting point, a strategy and simulation approach were developed which shares similarities with tempered and adaptive Langevin algorithms. While not as elegantly derived from stochastic differential equations as the tempered and adaptive Langevin algorithms of [1, 17], the MALTS algorithm is intended to be effective for simulating distribution with a bounded parameter spaces and zones of high probability. In empirical comparisons with the standard random walk and MALA Markov chain algorithms, the MALTS algorithm proves to be just as effective, and in some cases more so. The MALTS algorithm has been applied with success to a Bayesian model for Chronic Wasting Disease in Rocky Mountain mule deer that is described in [9], showing that it can be used for real as well as contrived data.

While the empirical evidence is positive, the theoretical properties of MALTS still need to be rigorously studied. The essential convergence of the algorithm to the correct stationary distribution was established, but theoretical results on convergence properties are necessary to more fully understand the algorithm. Theoretical guidelines for the use of MALTS will help to determine what kinds of target densities are best suited to the new algorithm. Nonetheless, the results in this paper provide another tool for Markov Chain Monte Carlo simulation.

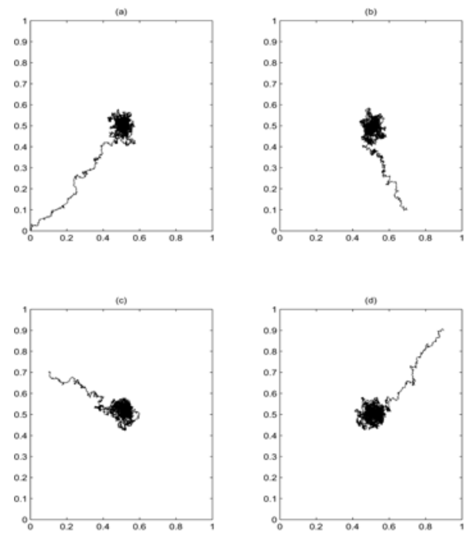


Figure 1: Pairwise plots of the mean parameters $\theta_1$ and $\theta_2$ for the truncated normal distribution with true mean $(0.5, 0.5)^T$ and true covariance $0.001*I$. (a) shows $8,000$ iterations for the initial value $(0,0)^T$. (b) shows $8,000$ iterations for the starting value $(0, 0.001)^T$. (c) shows $8,000$ iterations for the starting value $(0.3, 0.4)^T$. (d) shows $8,000$ iterations for the starting value $(0.1, 0.8)^T$

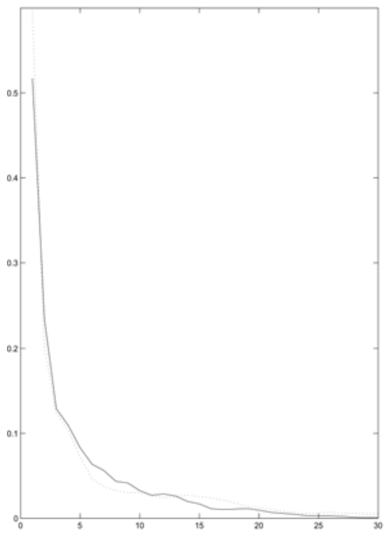


Figure 2: Plots of the log potential scale reductions by thousand iterations for the mean parameters $\theta_1$ and $\theta_2$ for the MALTS simulation of the truncated normal distribution. The solid line shows the results for $\theta_1$, the dotted line shows the results for $\theta_2$

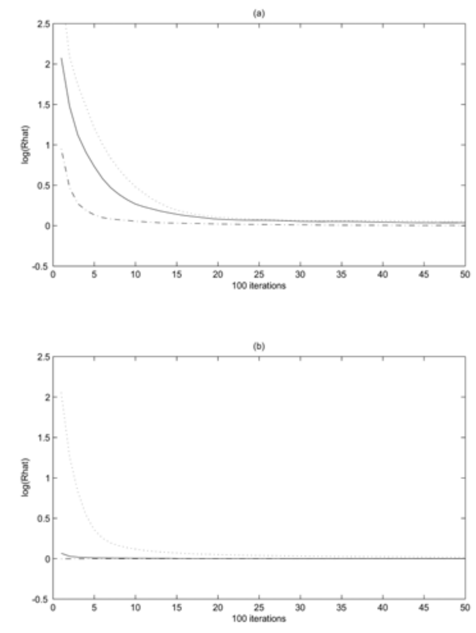


Figure 3: Plots of the log potential scale reductions by hundred iterations for the parameters $\theta_1$ and $\theta_2$ for the the simulation of the 36 dimensional truncated normal distribution. The solid line is the MALTS algorithm, the dash-dot is the Langevin algorithm, and the dotted line is the standard random walk. (a) shows the results for $\theta_1$. (b) shows the results for $\theta_2$

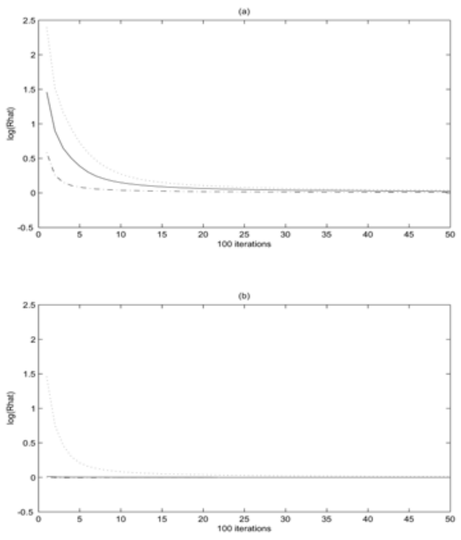


Figure 4: Plots of the log potential scale reductions by hundred iterations for the variance of the parameters $\theta_1$ and $\theta_2$ for the the simulation of the 36 dimensional truncated normal distribution. The solid line is the MALTS algorithm, the dash-dot is the Langevin algorithm, and the dotted line is the standard random walk. (a) shows the results for $\theta_1$. (b) shows the results for $\theta_2$

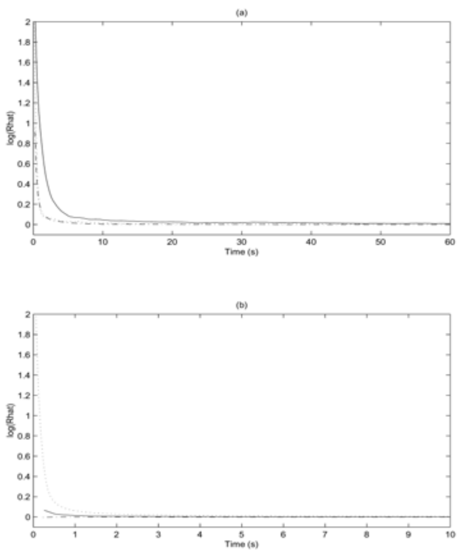


Figure 5: Plots of the log potential scale reductions by time for the parameters $\theta_1$ and $\theta_2$ for the the simulation of the 36 dimensional truncated normal distribution. The solid line is the MALTS algorithm, the dash-dot is the Langevin algorithm, and the dotted line is the standard random walk. (a) shows the results for $\theta_1$. (b) shows the results for $\theta_2$

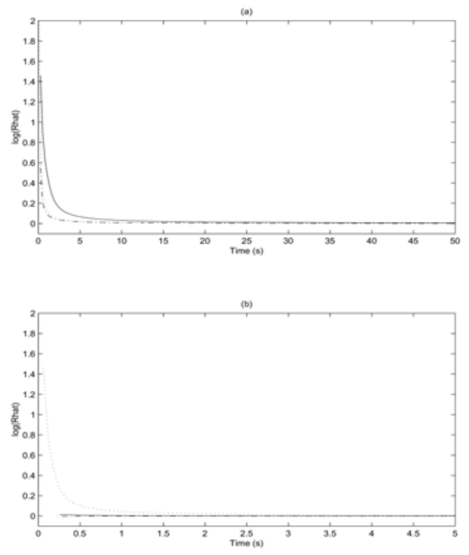


Figure 6: Plots of the log potential scale reductions by time for the variance of the parameters $\theta_1$ and $\theta_2$ for the simulation of the 36 dimensional truncated normal distribution. The solid line is the MALTS algorithm, the dash-dot is the Langevin algorithm, and the dotted line is the standard random walk. (a) shows the results for $\theta_1$. (b) shows the results for $\theta_2$

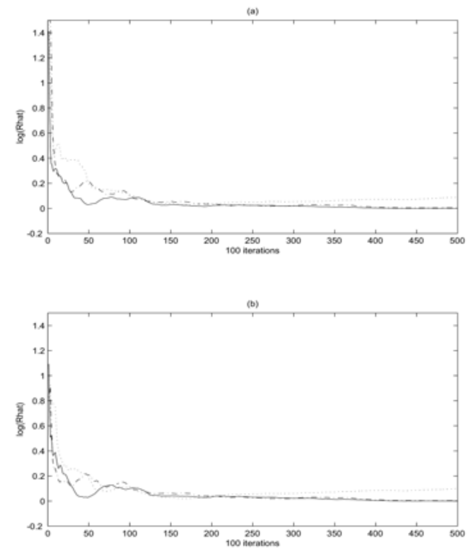


Figure 7: Plots of the log potential scale reductions by hundred iterations for the parameters $\theta_1$ and $\theta_2$ for the simulation of the 36 dimensional normal distribution, with correlation $\rho = 0.879$ and step size $1 \times 10^{-5}$. The solid line is the MALTS algorithm, the dash-dot is the Langevin algorithm, and the dotted line is the standard random walk. (a) shows the results for $\theta_1$. (b) shows the results for $\theta_2$

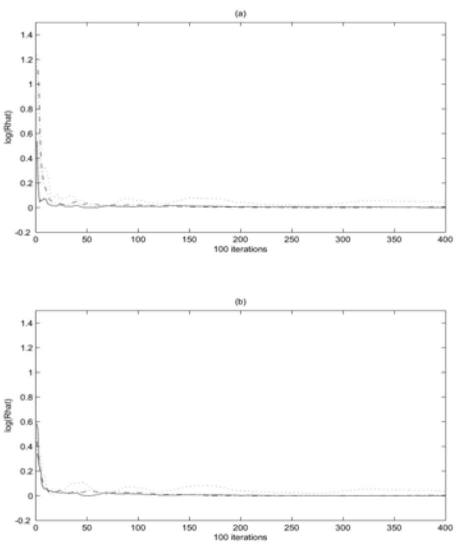


Figure 8: Plots of the log potential scale reductions by hundred iterations for the variance of the parameters $\theta_1$ and $\theta_2$ for the the simulation of the 36 dimensional normal distribution, with correlation $\rho = 0.879$ and step size $1 \times 10^{-5}$. The solid line is the MALTS algorithm, the dash-dot is the Langevin algorithm, and the dotted line is the standard random walk. (a) shows the results for $\theta_1$. (b) shows the results for $\theta_2$

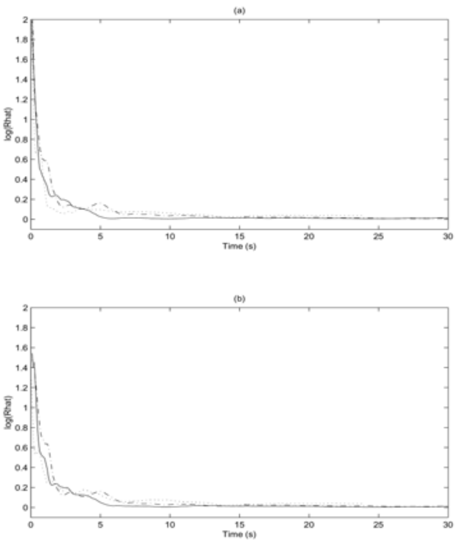


Figure 9: Plots of the log potential scale reductions by time for the parameters $\theta_1$ and $\theta_2$ for the simulation of the two dimensional normal distribution, with correlation $\rho = 0.879$ and step size $1 \times 10^{-5}$. The solid line is the MALTS algorithm, the dash-dot is the Langevin algorithm, and the dotted line is the standard random walk. (a) shows the results for $\theta_1$. (b) shows the results for $\theta_2$

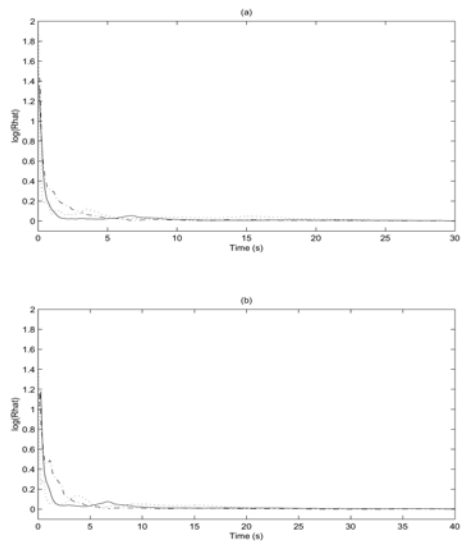


Figure 10: Plots of the log potential scale reductions by time for the variance of the parameters $\theta_1$ and $\theta_2$ for the simulation of the two dimensional normal distribution, with correlation $\rho = 0.879$ and step size $1 \times 10^{-5}$. The solid line is the MALTS algorithm, the dash-dot is the Langevin algorithm, and the dotted line is the standard random walk. (a) shows the results for $\theta_1$. (b) shows the results for $\theta_2$

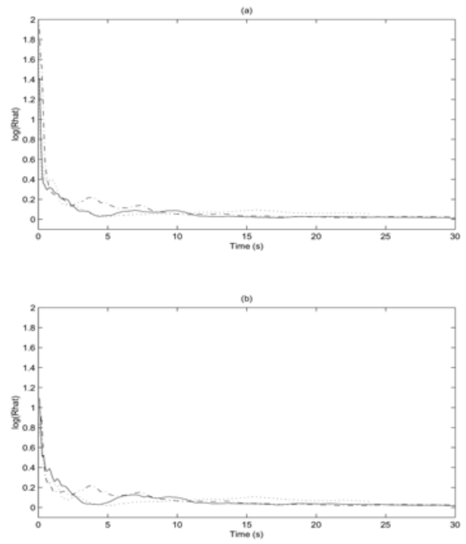


Figure 11: Plots of the log potential scale reductions by time for the parameters $\theta_1$ and $\theta_2$ for the simulation of the 36 dimensional normal distribution, with correlation $\rho = 0.879$ and step size $1 \times 10^{-5}$. The solid line is the MALTS algorithm, the dash-dot is the Langevin algorithm, and the dotted line is the standard random walk. (a) shows the results for $\theta_1$. (b) shows the results for $\theta_2$

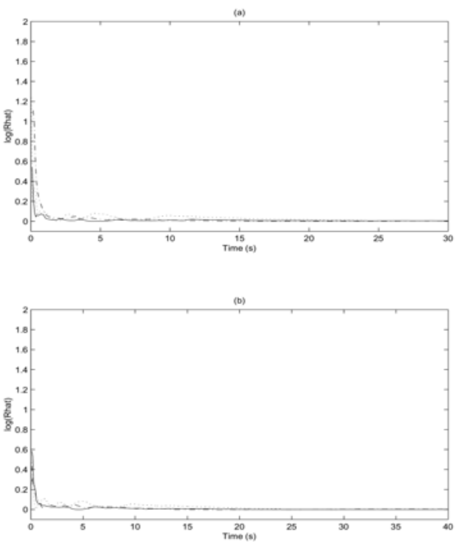


Figure 12: Plots of the log potential scale reductions by time for the variance of the parameters $\theta_1$ and $\theta_2$ for the simulation of the 36 dimensional normal distribution, with correlation $\rho = 0.879$ and step size $1 \times 10^{-5}$. The solid line is the MALTS algorithm, the dash-dot is the Langevin algorithm, and the dotted line is the standard random walk. (a) shows the results for $\theta_1$. (b) shows the results for $\theta_2$

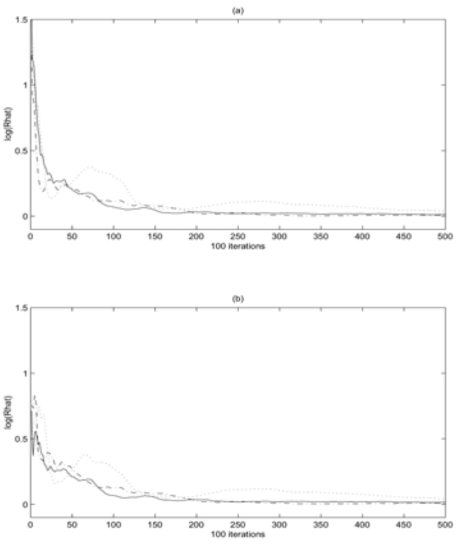
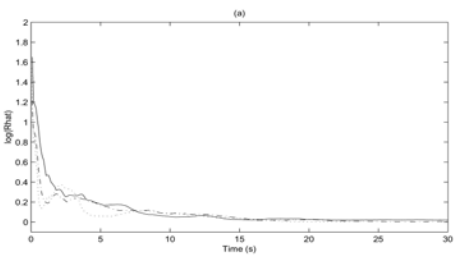


Figure 13: Plots of the log potential scale reductions by hundred iterations for the parameters $\theta_1$ and $\theta_2$ for the simulation of the 36 dimensional normal distribution, with correlation $\rho = 0.992$ and step size $1 \times 10^{-5}$. The solid line is the MALTS algorithm, the dash-dot is the Langevin algorithm, and the dotted line is the standard random walk. (a) shows the results for $\theta_1$. (b) shows the results for $\theta_2$



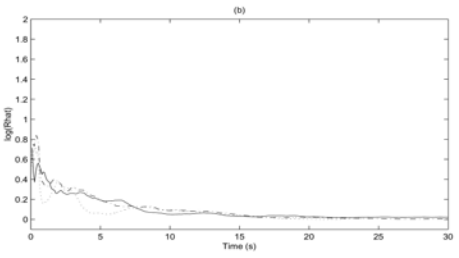
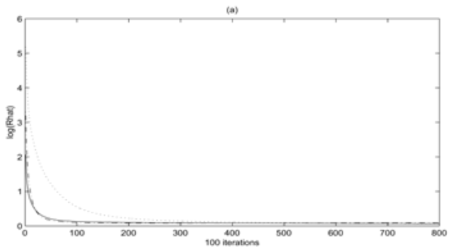


Figure 14: Plots of the log potential scale reductions by time for the parameters $\theta_1$ and $\theta_2$ for the simulation of the 36 dimensional normal distribution, with correlation $\rho = 0.992$ and step size $1 \times 10^{-5}$. The solid line is the MALTS algorithm, the dash-dot is the Langevin algorithm, and the dotted line is the standard random walk. (a) shows the results for $\theta_1$. (b) shows the results for $\theta_2$



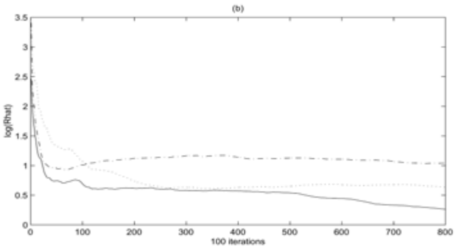


Figure 15: Plots of the log potential scale reductions by hundred iterations for the parameters $\theta_1$ and $\theta_2$ for the simulation of the 200 dimensional normal distribution, with correlation $\rho = 0.992$ and step size $1 \times 10^{-5}$. The solid line is the MALTS algorithm, the dash-dot is the Langevin algorithm, and the dotted line is the standard random walk. (a) shows the results for $\theta_1$. (b) shows the results for $\theta_2$

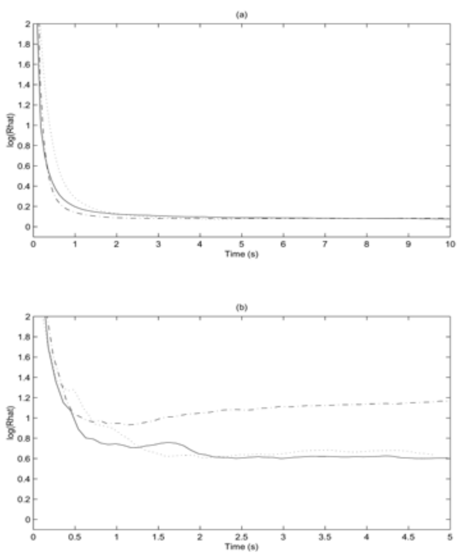


Figure 16: Plots of the log potential scale reductions by time for the parameters $\theta_1$ and $\theta_2$ for the simulation of the 200 dimensional normal distribution, with correlation $\rho = 0.992$ and step size $1 \times 10^{-5}$. The solid line is the MALTS algorithm, the dash-dot is the Langevin algorithm, and the dotted line is the standard random walk. (a) shows the results for $\theta_1$. (b) shows the results for $\theta_2$

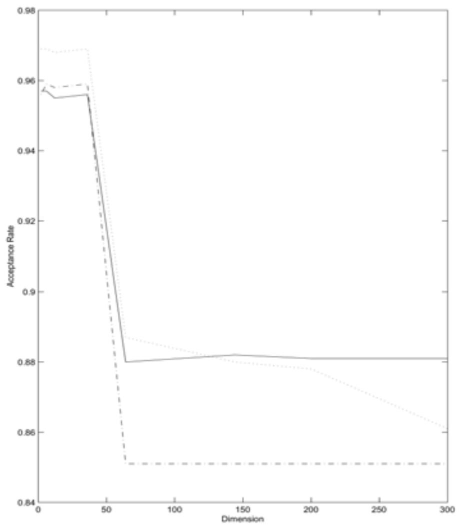


Figure 17: Plots of the acceptance rates by dimension for the multivariate normal simulation with correlation $\rho = 0.992$ and step size $1 \times 10^{-5}$. The solid line is the MALTS algorithm, the dash-dot is the Langevin algorithm, and the dotted line is the standard random walk

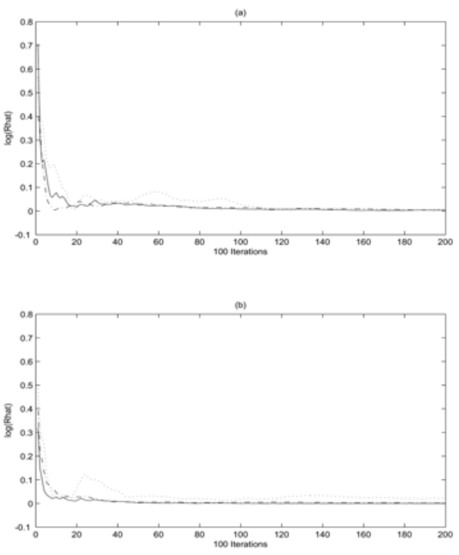


Figure 18: Plots of the log potential scale reductions by hundred iterations for the parameters $\theta_1$ and $\theta_2$ for the simulation of the 36 dimensional normal distribution, with correlation $\rho = 0.992$, and step size $1 \times 10^{-4}$. The solid line is the MALTS algorithm, the dash-dot is the Langevin algorithm, and the dotted line is the standard random walk. (a) shows the results for $\theta_1$. (b) shows the results for $\theta_2$

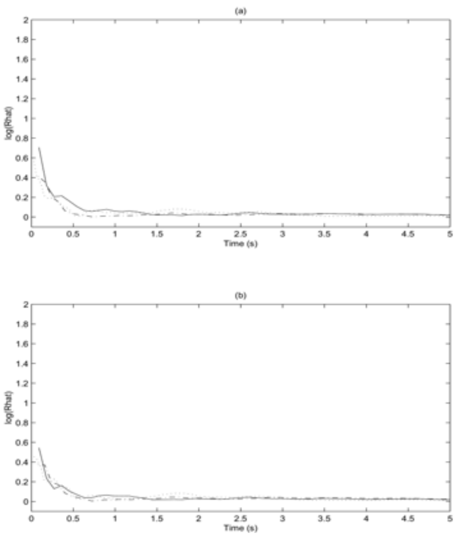


Figure 19: Plots of the log potential scale reductions by time for the parameters $\theta_1$ and $\theta_2$ for the simulation of the 36 dimensional normal distribution, with correlation $\rho = 0.992$, and step size $1 \times 10^{-4}$. The solid line is the MALTS algorithm, the dash-dot is the Langevin algorithm, and the dotted line is the standard random walk. (a) shows the results for $\theta_1$. (b) shows the results for $\theta_2$

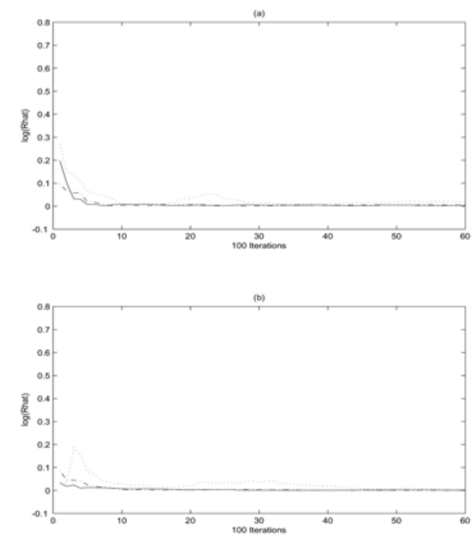


Figure 20: Plots of the log potential scale reductions by hundred iterations for the parameters $\theta_1$ and $\theta_2$ for the simulation of the 36 dimensional normal distribution, with correlation $\rho = 0.992$, and step size $1 \times 10^{-2}$. The solid line is the MALTS algorithm, the dash-dot is the Langevin algorithm, and the dotted line is the standard random walk. (a) shows the results for $\theta_1$. (b) shows the results for the variance of $\theta_1$

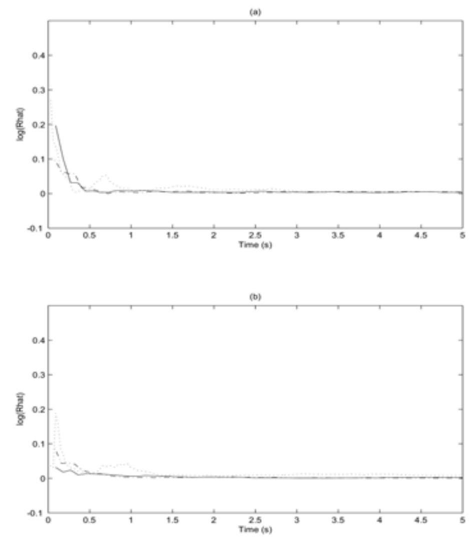


Figure 21: Plots of the log potential scale reductions by time for the parameters $\theta_1$ and $\theta_2$ for the simulation of the 36 dimensional normal distribution, with correlation $\rho = 0.992$, and step size $1 \times 10^{-2}$. The solid line is the MALTS algorithm, the dash-dot is the Langevin algorithm, and the dotted line is the standard random walk. (a) shows the results for $\theta_1$. (b) shows the results for the variance of $\theta_1$

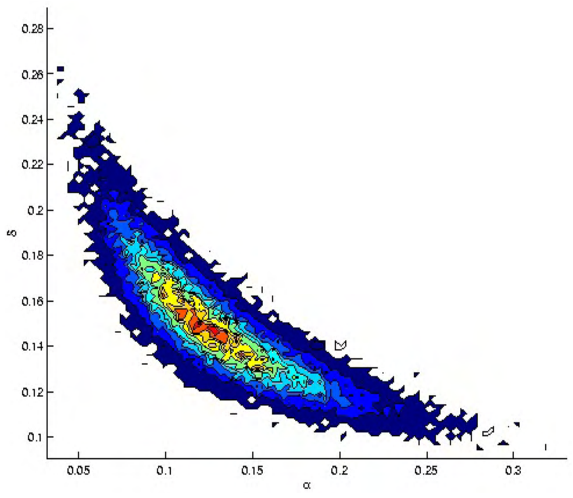


Figure 22: A contour plot of $\delta$ against $\alpha$. Lighter shaded regions represent higher density areas. Most of the values occur along a banana shaped ridge of high probability

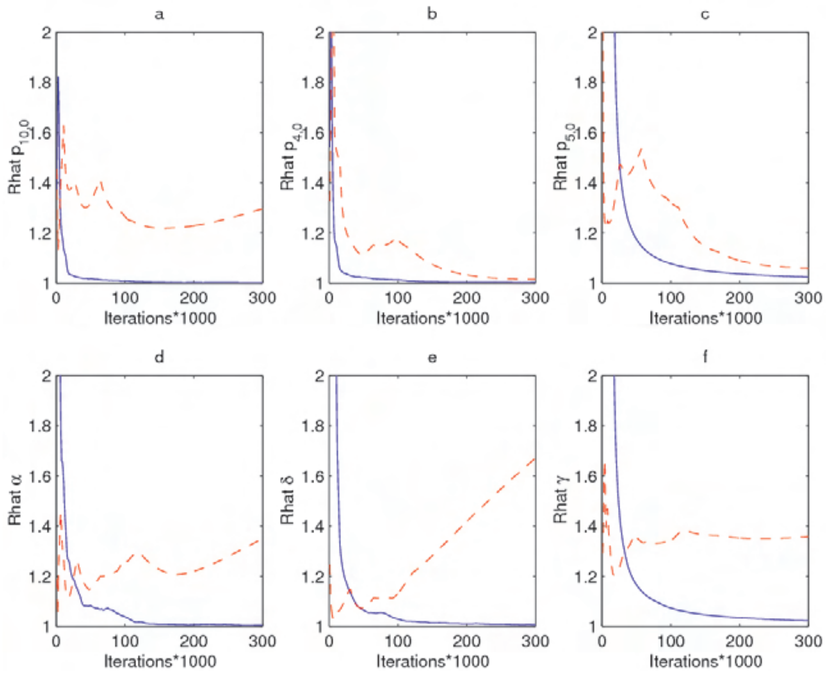


Figure 23: Plots of the potential scale reduction by iteration for each parameter in the CWD model. The dashed line shows the results for a standard random walk; the solid line shows the results for the MALTS algorithm. The results illustrate both the poor performance of the standard random walk and the convergence of MALTS

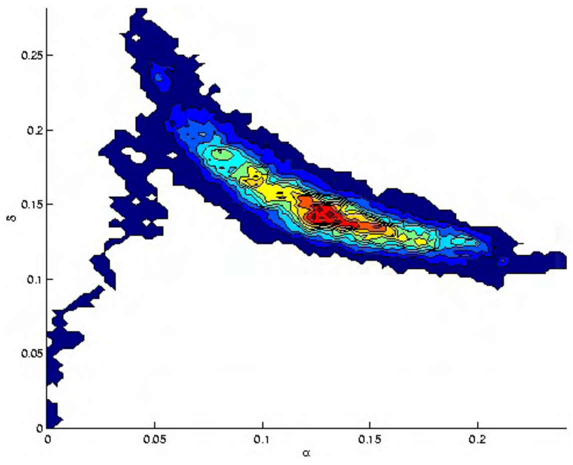


Figure 24: A contour plot of $\delta$ against $\alpha$ produced by the MALTS algorithm. Lighter coloured regions represent higher density areas. The algorithm was able to reproduce the distinctive banana shaped ridge of the posterior

## Appendix 1: Truncated Normal Acceptance Rates

The standard random walk, Metropolis adjusted Langevin algorithm (MALA), and Modified adjusted Langevin algorithm with Tempered step size (MALTS) were used to simulate multivariate normal distributions of increasing dimension, truncated to the unit cube. The dimensions were 2, 6, 12, and 36. The step size was fixed for each method at $1 \times 10^{-5}$.

Table 2: Acceptance rates for the multivariate truncated normal distribution

| Method | Dim = 2 | Dim = 6 | Dim = 12 | Dim = 36 |
|---|---|---|---|---|
| Std. RW | 0.699 | 0.699 | 0.699 | 0.698 |
| MALA | 0.661 | 0.659 | 0.660 | 0.658 |
| MALTS | 0.700 | 0.700 | 0.701 | 0.697 |

## Appendix 2: MV Normal Acceptance Rates

The standard random walk, Metropolis adjusted Langevin algorithm (MALA), and Modified adjusted Langevin algorithm with Tempered step size (MALTS) were used to simulate multivariate normal distributions of increasing dimension and correlation. The dimensions were 2, 6, 12, and 36, and the levels of correlation were 0.879, 0.970, and 0.992. Additional simulations were run at correlation 0.992 with dimensions 64, 144, 200, and 300. The step size was fixed for each method at $1 \times 10^{-5}$, $1 \times 10^{-4}$, and $1 \times 10^{-2}$.

Table 3: Acceptance rates for each dimension for the multivariate normal distribution with correlation $\rho = 0.879$. The step size is $1 \times 10^{-5}$

| Method | Dim = 2 | Dim = 6 | Dim = 12 | Dim = 36 |
|---|---|---|---|---|
| Std. RW | 0.967 | 0.969 | 0.969 | 0.968 |
| MALA | 0.959 | 0.959 | 0.958 | 0.958 |
| MALTS | 0.955 | 0.955 | 0.956 | 0.956 |

Table 4: Acceptance rates for each dimension for the multivariate normal distribution with correlation $\rho = 0.879$. The step size is $1 \times 10^{-4}$

| Method | Dim = 2 | Dim = 6 | Dim = 12 | Dim = 36 |
|---|---|---|---|---|
| Std. RW | 0.904 | 0.902 | 0.902 | 0.903 |
| MALA | 0.871 | 0.871 | 0.871 | 0.871 |
| MALTS | 0.873 | 0.872 | 0.872 | 0.873 |

Table 5: Acceptance rates for each dimension for the multivariate normal distribution with correlation $\rho = 0.879$. The step size is $1 \times 10^{-2}$

| Method | Dim = 2 | Dim = 6 | Dim = 12 | Dim = 36 |
|---|---|---|---|---|
| Std. RW | 0.809 | 0.809 | 0.808 | 0.809 |
| MALA | 0.758 | 0.758 | 0.759 | 0.756 |
| MALTS | 0.775 | 0.774 | 0.773 | 0.774 |

Table 6: Acceptance rates for each dimension for the multivariate normal distribution with correlation $\rho = 0.970$. The step size is $1 \times 10^{-5}$

| Method | Dim = 2 | Dim = 6 | Dim = 12 | Dim = 36 |
|---|---|---|---|---|
| Std. RW | 0.997 | 0.969 | 0.969 | 0.969 |
| MALA | 0.995 | 0.958 | 0.958 | 0.959 |
| MALTS | 0.995 | 0.956 | 0.958 | 0.958 |

Table 7: Acceptance rates for each dimension for the multivariate normal distribution with correlation $\rho = 0.970$. The step size is $1 \times 10^{-4}$

| Method | Dim = 2 | Dim = 6 | Dim = 12 | Dim = 36 |
|---|---|---|---|---|
| Std. RW | 0.902 | 0.901 | 0.901 | 0.902 |
| MALA | 0.870 | 0.871 | 0.871 | 0.870 |
| MALTS | 0.872 | 0.873 | 0.872 | 0.873 |

Table 8: Acceptance rates for each dimension for the multivariate normal distribution with correlation $\rho = 0.970$. The step size is $1 \times 10^{-2}$

| Method | Dim = 2 | Dim = 6 | Dim = 12 | Dim = 36 |
|---|---|---|---|---|
| Std. RW | 0.809 | 0.807 | 0.809 | 0.808 |
| MALA | 0.756 | 0.758 | 0.758 | 0.757 |
| MALTS | 0.773 | 0.774 | 0.774 | 0.774 |

Table 9: Acceptance rates for each dimension for the multivariate normal distribution with correlation $\rho = 0.992$. The steps size is $1 \times 10^{-5}$

| Method | Dim = 2 | Dim = 6 | Dim = 12 | Dim = 36 |
|---|---|---|---|---|
| Std. RW | 0.969 | 0.969 | 0.968 | 0.969 |
| MALA | 0.956 | 0.959 | 0.958 | 0.959 |
| MALTS | 0.957 | 0.957 | 0.955 | 0.956 |

Table 10: Acceptance rates for each dimension for the multivariate normal distribution with correlation $\rho = 0.992$. The steps size is $1 \times 10^{-5}$

| Method | Dim = 64 | Dim = 144 | Dim = 200 | Dim = 300 |
|---|---|---|---|---|
| Std. RW | 0.887 | 0.880 | 0.878 | 0.861 |
| MALA | 0.851 | 0.851 | 0.851 | 0.851 |
| MALTS | 0.880 | 0.882 | 0.881 | 0.881 |

Table 11: Acceptance rates for each dimension for the multivariate normal distribution with correlation $\rho = 0.992$. The step size is $1 \times 10^{-4}$

| Method | Dim = 2 | Dim = 6 | Dim = 12 | Dim = 36 |
|---|---|---|---|---|
| Std. RW | 0.903 | 0.902 | 0.901 | 0.902 |
| MALA | 0.873 | 0.871 | 0.871 | 0.871 |
| MALTS | 0.872 | 0.873 | 0.873 | 0.873 |

Table 12: Acceptance rates for each dimension for the multivariate normal distribution with correlation $\rho = 0.992$. The step size is $1 \times 10^{-2}$

| Method | Dim = 2 | Dim = 6 | Dim = 12 | Dim = 36 |
|---|---|---|---|---|
| Std. RW | 0.810 | 0.808 | 0.808 | 0.808 |
| MALA | 0.759 | 0.758 | 0.758 | 0.757 |
| MALTS | 0.774 | 0.774 | 0.774 | 0.774 |

## References

[1] Atchade, Y. (2006). An adaptive version for the Metropolis adjusted Langevin algorithm with a truncated drift. *Methodology and Computing in Applied Probability* **8**, 235-254.

[2] Besag, J., Green, P. J., Higdon, D. and Mengerson, K. (1995). Bayesian computation and stochastic systems (with discussion). *Statistical Science* **10**, 3-66.

[3] Besag, J., York, J. and Mollie, A. (1991). Bayesian image restoration, with two applications in spatial statistics (with discussion). *Annals of the Institute of Statistical Mathematics* **43**, 1-59.

[4] Chung, K. L. and Williams, R. J. (1990). *Introduction to Stochastic Integration*, 2nd edition. Birkhäuser, Boston, Massachusetts.

[5] Gelman, A., Carlin, J. B., Stern, H. S. and Rubin, D. B. (1995). *Bayesian Data Analysis.* Texts in Statistical Science. Chapman and Hall/CRC Press, Boca Raton, Florida.

[6] Gilks, W. R., Richardson, S. and Spiegelhalter, D. J. (1999). *Markov Chain Monte Carlo in Practice*: *Interdisciplinary Statistics.* Chapman and Hall/ CRC Press, New York.

[7] Gustafson, P., Macnab, Y. C. and Wen, S. (2004). On the value of derivative evaluations and random walk suppression in Markov Chain Monte Carlo algorithms. *Statistics and Computing* **14**, 23-38.

[8] Haran, M., Hodges, J. S. and Carlin, B. P. (2003). Accelerating computation in Markov random field models for spatial data via structured MCMC. *Journal of Computational and Graphical Statistics* **12**, 249-264.

[9] Johns, C. J. and Mehl, C. H. (2006). A dynamic spatial model for chronic wasting disease in Colorado. *Journal of Data Science* **4**, 21-37.

[10] Øsendal, B. (1998). *Stochastic Differential Equations*: *An Introduction with Applications*, 5th edition. Springer, Milan.

[11] Mehl, C. H. (2004). *Bayesian Hierarchical Modeling and Markov Chain Simulation for Chronic Wasting Disease.* Ph.D. Dissertation, University of Colorado, Denver, Colorado.

[12] Neal, R. M. (1993). *Probabilistic Inference Using Markov Chain Monte Carlo Methods.* Ph.D. Dissertation, University of Toronto, Toronto.

[13] Robert, C. P. and Casella, G. (1999). *Monte Carlo Statistical Methods.* Springer, New York.

[14] Roberts, G. O. and Rosenthal, J. S. (1996). Quantitative bounds for convergence rates of continuous time Markov processes. *Electronic Journal of Probability* **1**, 1-21.

[15] 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.

[16] Roberts, G. O. and Rosenthal, J. S. (2009). Examples of adaptive MCMC. *Journal of Computational and Graphical Statistics* **18**, 349-367.

[17] Roberts, G. O. and Stramer, O. (2002). Tempered Langevin diffusions and algorithms. Department of Statistics and Actuarial Science Technical Report no. 314, University of Iowa, Iowa.

[18] Skare, O., Benth, F. E. and Frigessi, A. (2000). Smoothed Langevin proposals in Metropolis-Hastings algorithms. *Statistics and Probability Letters* **49**, 345-354.

Received June 11, 2012; accepted February 21, 2013.

Christopher H. Mehl
Senior Mathematician
OMNITEC Solutions, Inc.
6701 Democracy Blvd, Suite 300. Bethesda, Maryland 20817, USA
cmehl@omnitecinc.com