Journal of Machine Learning Research 1 (2016) 1-48 Submitted 4/00; Published 10/00

Manifold Langevin Monte Carlo with Partial Metric Updates

Theodore Papamarkou THEODORE.PAPAMARKOU@GLASGOW.AC.UK
School of Mathematics and Statistics
University of Glasgow
15 University Gardens
Glasgow, G12 8QW, UK

Eric B. Ford EBF11@PSU.EDU
Center for Astrostatistics
Institute for CyberScience
Center for Exoplanets and Habitable Worlds
Department of Astronomy and Astrophysics
525 Davey Lab
The Pennsylvania State University
University Park, PA 16802, USA

Editor: Leslie Pack Kaelbling

Abstract

Manifold Markov chain Monte Carlo algorithms have been introduced to sample more effectively from challenging target densities exhibiting multiple modes or strong correlations. Such algorithms exploit the local geometry of the parameter space, thus enabling chains to achieve a faster convergence rate when measured in number of steps. However, often acquiring local geometric information increases computational complexity per step to the extent that sampling from high-dimensional targets becomes inefficient in terms of total computational time. This paper proposes a manifold Langevin Monte Carlo framework aimed at balancing the benefits of exploiting local geometry with computational requirements to achieve a high effective sample size for a given computational cost. The suggested strategy regulates the frequency of manifold-based updates via a schedule. An exponentially decaying schedule is put forward that enables more frequent updates of geometric information in early transient phases of the chain, while saving computational time in late stationary phases. Alternatively, a modulo schedule is introduced to allow for infrequent yet recurring geometric updates throughout the sampling course, acting to adaptively update proposals to learn the covariance structure of the parameter space. The average complexity can be manually set for either of these two schedules depending on the need for geometric exploitation posed by the underlying model.

Keywords: Bayesian inference, High-dimensional data analysis, Markov chain Monte Carlo, Langevin Monte Carlo

1. Introduction

The birth of geometric-based Markov chain Monte Carlo (MCMC) dates back to the landmark paper of Duane et al. (1987), which introduced Hamiltonian Monte Carlo (HMC) to unite MCMC with molecular dynamics. Duane et al. (1987) applied HMC in lattice field

theory simulations of quantum chromodynamics. Statistical applications of HMC began with its use in neural network models by Neal (1996).

In the meanwhile, the Metropolis-adjusted Langevin algorithm (MALA) was proposed by Roberts and Rosenthal (1998) to harness Langevin dynamics for proposing new states in MCMC simulations. Both HMC and MALA evaluate the gradient of the target density, so they utilize local geometric flow.

The seminal work of Girolami and Calderhead (2011) raised awareness of the differential geometric foundations of MCMC. Given a vector of parameters $\boldsymbol{\theta} \in \mathbb{R}^{n_\theta}$, Girolami and Calderhead (2011) defines a distance between two probability densities $p(\boldsymbol{\theta})$ and $p(\boldsymbol{\theta} + \delta\boldsymbol{\theta})$ as the quadratic form $\delta\boldsymbol{\theta}^T G(\boldsymbol{\theta})\delta\boldsymbol{\theta}$ for an arbitrary metric $G(\boldsymbol{\theta})$. Thus, the position-specific metric $G(\boldsymbol{\theta})$ induces a Riemann manifold in the space of parameterized probability density functions $\{p(\boldsymbol{\theta}) : \boldsymbol{\theta}\}$. In the context of MCMC, $G(\boldsymbol{\theta})$ depends on the current state $\boldsymbol{\theta}$ of the simulated Markov chain. Girolami and Calderhead (2011) uses $G(\boldsymbol{\theta})$ to define transition kernels that explore the parameter space $\{\boldsymbol{\theta} : \boldsymbol{\theta} \in \mathbb{R}^{n_\theta}\}$ effectively, thereby introducing Riemann manifold Langevin and Hamiltonian Monte Carlo methods.

The differential geometric basis of MCMC has been further researched since the appearance of Girolami and Calderhead (2011). For instance, Byrne and Girolami (2013) developed proposal mechanisms by following geodesic flows over the embedded manifolds of support for the target distribution. Lan et al. (2015) replaced the auxiliary momentum variable appearing in Riemann manifold Hamiltonian Monte Carlo (RMHMC) by velocity to develop a Markov chain Monte Carlo algorithm using Lagrangian dynamics. Lan et al. (2014) defined the so-called wormhole metric, for which distance between modes is shortened, in order to facilitate transitions between modes of multimodal target densities.

Computing the geometric entities of differential geometric MCMC methods creates a performance bottleneck that restricts the applicability of the involved methods. For example, manifold Monte Carlo methods require to calculate the metric tensor $G(\boldsymbol{\theta})$ of choice. Typically, $G(\boldsymbol{\theta})$ is set to be the observed Fisher information matrix, which equals the negative Hessian of the log-target density at a specific point $\boldsymbol{\theta}$. Consequently, the complexity of manifold MCMC algorithms is dominated by Hessian-related computations, either the computation of gradient and Hessian or the inversion of the Hessian.

Girolami and Calderhead (2011) introduced the simplified manifold Metropolis-adjusted Langevin algorithm (SMMALA) that is of the same order but often faster than MMALA and RMHMC in lower-dimensional parameter spaces. The faster speed of SMMALA over MMALA and RMHMC is explained by lower order terms and constant factors appearing in big-oh notation, which are ordinarily omitted but affect runtime in the case of smaller n_θ. For this reason, SMMALA has been employed in conjunction with population MCMC for the Bayesian analysis of mechanistic models based on systems of non-linear differential equations, see Calderhead and Girolami (2011) and Schwentner et al. (2015). Despite the capacity of SMMALA to exploit local geometric information so as to cope with non-linear correlations and modest increase in the dimensionality of the parameter space, in other cases its computational complexity can render the performance inferior to other algorithms such as the Metropolis-adjusted Langevin algorithm (MALA) or adaptive MCMC, see Calderhead et al. (2013).

Various attempts have been made to ameliorate the computational implications of geometric Monte Carlo methods. Along these lines, Lan et al. (2016) used Gaussian processes

to emulate the Hessian matrix and Christoffel symbols associated with the observed Fisher information $G(\boldsymbol{\theta})$. Şimşekli et al. (2016) developed a stochastic quasi-Newton Langevin MCMC algorithm which takes into account the local geometry, while approximating the inverse Hessian by using a limited history of samples and their gradients. Alternatively, Pereyra (2016) used convex analysis and proximal techniques instead of differential calculus in order to construct a Langevin Monte Carlo method for high-dimensional target distributions that are log-concave and possibly not continuously differentiable.

This paper develops a Monte Carlo method that improves on the sampling efficacy of SMMALA, while remaining simple to implement. The proposed method permits adjustments to the metric updates and at the same time has smaller computational cost than SMMALA. Moreover, the suggested algorithm performs exact metric evaluations, avoiding emulation or other approximations of the Hessian of the log-target density.

2. Background

The role of this section is to provide a brief overview of Langevin Monte Carlo and to stage a unified notation, which will be used for introducing the method contributed by this paper.

From a statistical perspective, Langevin Monte Carlo is a special case of Metropolis-Hastings with a normal proposal distribution

$$q(\boldsymbol{\theta}^\star|\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}^\star|\boldsymbol{\mu}(\boldsymbol{\theta}, G, \epsilon), \Sigma(G, \epsilon)), \tag{1}$$

where $\boldsymbol{\theta}^\star$ and $\boldsymbol{\theta}$ denote the respective proposed and current state, G is a positive definite matrix of size $n_\theta \times n_\theta$ and ϵ refers to a tuning parameter known as the integration stepsize. The location $\boldsymbol{\mu}(\boldsymbol{\theta}, G, \epsilon)$ is a function of $\boldsymbol{\theta}$, G and ϵ, whereas the covariance $\Sigma(G, \epsilon)$ of the proposal density depends on G and ϵ. Both $\boldsymbol{\mu}(\boldsymbol{\theta}, G, \epsilon)$ and $\Sigma(G, \epsilon)$ are defined so that the proposed states admit a Langevin diffusion approximated by a first-order Euler discretization. In other words, the Langevin diffusion has a stationary distribution $p(\boldsymbol{\theta})$, so a Markov chain simulated from the Metropolis-Hastings scheme outlined by proposal (1) will converge to the target $p(\boldsymbol{\theta})$. The Metropolis-Hastings acceptance probability is set to its standard form

$$a(\boldsymbol{\theta}, \boldsymbol{\theta}^\star) = \min\left\{\frac{p(\boldsymbol{\theta}^\star)q(\boldsymbol{\theta}|\boldsymbol{\theta}^\star)}{p(\boldsymbol{\theta})q(\boldsymbol{\theta}^\star|\boldsymbol{\theta})}, 1\right\}. \tag{2}$$

The integration stepsize ϵ, also known as drift step, is associated with the first order Euler discretization and significantly affects the rate of state space exploration. If ϵ is selected to be relatively large, many of the proposed candidates will be far from the current state, and are likely to have a low probability of acceptance, so the generated chain will have low acceptance rate. Reducing ϵ will increase the acceptance rate, but the chain will take longer to traverse the state space.

In a Bayesian setting, the target is a possibly unnormalized posterior distribution $p(\boldsymbol{\theta}|\mathbf{y})$, where $\mathbf{y}$ denotes the available data. Replacing $p(\boldsymbol{\theta})$ by $p(\boldsymbol{\theta}|\mathbf{y})$ in (2) makes Langevin Monte Carlo applicable in Bayesian problems.

To fully specify a Langevin Monte Carlo algorithm, the location $\boldsymbol{\mu}(\boldsymbol{\theta}, G, \epsilon)$ and covariance $\Sigma(G, \epsilon)$ of normal proposal (1) need to be defined. In what follows, variations of geometric Langevin Monte Carlo methods are distinguished by their respective proposal location and covariance.

2.1 Metropolis-Adjusted Langevin Algorithm

If G is kept constant, that is if G is a not a function of the current state $\boldsymbol{\theta}$, then the Metropolis-adjusted Langevin algorithm (MALA) arises as

$$\boldsymbol{\mu}(\boldsymbol{\theta}, G, \epsilon) = \boldsymbol{\theta} + \frac{\epsilon^2}{2} G^{-1} \nabla_\theta \log p(\boldsymbol{\theta}), \tag{3}$$

$$\Sigma(G, \epsilon) = \epsilon^2 G^{-1}. \tag{4}$$

G is known as the precondition matrix, see Roberts and Stramer (2002). It is typically set to be the identity matrix $G = I$, in which case MALA is defined in its conventional form

$$\boldsymbol{\mu}(\boldsymbol{\theta}, \epsilon) = \boldsymbol{\theta} + \frac{\epsilon^2}{2} \nabla_\theta \log p(\boldsymbol{\theta}), \ \Sigma(\epsilon) = \epsilon^2 I. \tag{5}$$

MALA uses the gradient flow $\nabla_\theta \log p(\boldsymbol{\theta})$ to make proposals effectively. According to the theoretical analysis of Roberts and Rosenthal (1998), the optimal scaling ϵ has been found to be the value of ϵ which yields a limiting acceptance rate of 57.4% in high-dimensional parametric spaces (as $n_\theta \to \infty$).

2.2 Manifold Langevin Monte Carlo Algorithms

It is possible to incorporate further geometric structure in the form of a position-dependent metric $G(\boldsymbol{\theta})$, as explained by Girolami and Calderhead (2011). The Langevin diffusion is defined on a Riemann manifold endowed by the metric $G(\boldsymbol{\theta})$, and the corresponding Levi-Civita connection specified by the Christoffel symbols of second-kind $\Gamma_{ij}^k(\boldsymbol{\theta})$. The resulting manifold Metropolis-adjusted Langevin algorithm (MMALA) draws candidate states from a normal proposal with location and covariance given by

$$\boldsymbol{\mu}_{(k)}(\boldsymbol{\theta}, G(\boldsymbol{\theta}), \epsilon) = \boldsymbol{\theta} + \frac{\epsilon^2}{2} (G^{-1}(\boldsymbol{\theta}) \nabla_\theta \log p(\boldsymbol{\theta}))_{(k)} - \epsilon^2 \sum_{i,j=1}^{n_\theta} G_{ij}^{-1}(\boldsymbol{\theta}) \Gamma_{ij}^k(\boldsymbol{\theta}), \tag{6}$$

$$\Sigma(G(\boldsymbol{\theta}), \epsilon) = \epsilon^2 G^{-1}(\boldsymbol{\theta}), \tag{7}$$

where $\boldsymbol{\mu}_{(k)}$ refers to the k-th coordinate of proposal location $\boldsymbol{\mu}(\boldsymbol{\theta}, G(\boldsymbol{\theta}), \epsilon)$.

If a manifold of constant curvature is assumed, then the Christoffel symbols vanish, significantly reducing the complexity of calculations needed for complex models. In this case, the proposal location of MMALA simplifies to

$$\boldsymbol{\mu}(\boldsymbol{\theta}, G(\boldsymbol{\theta}), \epsilon) = \boldsymbol{\theta} + \frac{\epsilon^2}{2} G^{-1}(\boldsymbol{\theta}) \nabla_\theta \log p(\boldsymbol{\theta}). \tag{8}$$

The method with location and covariance specified by (8) and (7) is known as simplified Metropolis-adjusted Langeving algorithm (SMMALA). As seen from the location and covariance of their proposal mechanisms, MALA is a special case of SMMALA, which is in turn a special case of MMALA.

The optimal stepsize ϵ for MMALA and SMMALA is empirically suggested by Girolami and Calderhead (2011) to be set so as to obtain an acceptance rate of around 70%; this choice has not been analyzed yet from a theoretical standpoint analogously to the choice of scaling for MALA by Roberts and Rosenthal (1998). Even if the curvature of the target density is not strictly constant, SMMALA can be applied and may be more efficient than MMALA.

2.3 Computational Complexity

Calculations associated with the proposal and target densities determine the incurring computational cost of Langevin Monte Carlo methods. In brief, the main computational requirements include sampling from and evaluating the proposal, as well as evaluating the target and its derivatives.

Ordinarily, the proposal used in Langevin Monte Carlo is set to be a multivariate normal distribution $q(\boldsymbol{\theta}^\star|\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}^\star|\boldsymbol{\mu}(\boldsymbol{\theta}, G, \epsilon), \epsilon^2 G^{-1}(\boldsymbol{\theta}))$, as seen in (1), (4) and (7). To propose a state $\boldsymbol{\theta}^\star$, the proposal is typically simulated in the Cholesky approach by letting

$$\boldsymbol{\theta}^\star = \boldsymbol{\mu}(\boldsymbol{\theta}, G, \epsilon) + \epsilon \left(\sqrt{G^{-1}(\boldsymbol{\theta})}\right)' \tau \tag{9}$$

where $\sqrt{G^{-1}(\boldsymbol{\theta})}$ satisfies the Cholesky decomposition $\left(\sqrt{G^{-1}(\boldsymbol{\theta})}\right)' \sqrt{G^{-1}(\boldsymbol{\theta})} = G^{-1}(\boldsymbol{\theta})$ and $\tau \sim \mathcal{N}(\mathbf{0}, I)$, Chib and Greenberg (1995). So, sampling from the proposal has a complexity of $\mathcal{O}(2n_\theta^3)$, since it requires the inversion of metric $G(\boldsymbol{\theta})$ and the Cholesky decomposition of $G^{-1}(\boldsymbol{\theta})$, each of which are $\mathcal{O}(n_\theta^3)$ operations.

To calculate the Metropolis-Hastings acceptance ratio (2) appearing in SMMALA and MMALA, it is required to evaluate the normal proposal twice. As it has become apparent, a proposal evaluation has a complexity of $\mathcal{O}(n_\theta^3)$ due to the proposal covariance $\epsilon^2 G^{-1}(\boldsymbol{\theta})$ entailing the inversion of $G(\boldsymbol{\theta})$.

So, SMMALA and MMALA necessitate computing the two matrix inverses $G^{-1}(\boldsymbol{\theta})$ and $G^{-1}(\boldsymbol{\theta}^\star)$ in order to evaluate the proposals involved in the acceptance probability (2), and one Cholesky decomposition $\sqrt{G^{-1}(\boldsymbol{\theta})}$ to sample from the proposal. Consequently, the cost of computations associated with the normal proposal is of order $\mathcal{O}(3n_\theta^3)$ for each of these two samplers.

Owing to the works of Davie and Stothers (2013), Williams (2011) and Gall (2014), optimized versions of the Coppersmith-Winograd algorithm allow to perform matrix multiplication and therefore matrix inversion in $\mathcal{O}(n_\theta^{2.373})$ time. Hence, optimized implementations of SMMALA and MMALA can each run their proposal-related computations in $\mathcal{O}(n_\theta^3 + 2n_\theta^{2.373})$ and time.

MALA, as opposed to SMMALA and MMALA, relies on a constant preconditioning matrix G, therefore G^{-1} and $\sqrt{G^{-1}}$ are evaluated once and cached at the beginning of the simulation avoiding the $\mathcal{O}(2n_\theta^3)$ penalty. Since the preconditioning matrix G, its inverse G^{-1} and the Cholesky decomposition $\sqrt{G^{-1}}$ are cached upon initialization for MALA, the quadratic form $(\boldsymbol{\theta}^\star - \boldsymbol{\theta})' G^{-1} (\boldsymbol{\theta}^\star - \boldsymbol{\theta})$ in the normal proposal density dictates a time complexity of $\mathcal{O}(n_\theta^2)$ for proposal-related computations in MALA. If G is set to be the identity matrix, then $(\boldsymbol{\theta}^\star - \boldsymbol{\theta})' G^{-1} (\boldsymbol{\theta}^\star - \boldsymbol{\theta})$ reduces to the inner product $\langle \boldsymbol{\theta}^\star - \boldsymbol{\theta}, \boldsymbol{\theta}^\star - \boldsymbol{\theta} \rangle$ and the complexity of calculations associate wit the proposal in MALA is of order $\mathcal{O}(n_\theta)$.

If a log-target has a complexity of order $\mathcal{O}(f(n_\theta))$, where f is a function of n_θ, then the complexity orders for computations associated with the log-target in MALA, SMMALA and MMALA grow as $\mathcal{O}(f(n_\theta)n_\theta)$, $\mathcal{O}(f(n_\theta)n_\theta^2)$ and $\mathcal{O}(f(n_\theta)n_\theta^3)$, respectively. These incurring costs for MALA, SMMALA and MMALA arise from the respective evaluations of log-target gradient, Hessian and Hessian derivatives.

Adding up the cost of computations associated with the proposal and the target yields the order of complexity for Langevin Monte Carlo. So, it follows that MALA, SMMALA and

MMALA run in $\mathcal{O}(f(n_\theta)n_\theta + n_\theta^2)$, $\mathcal{O}(f(n_\theta)n_\theta^2 + n_\theta^3 + 2n_\theta^{2.373})$ and $\mathcal{O}(f(n_\theta)n_\theta^3 + n_\theta^3 + 2n_\theta^{2.373})$ time, respectively.

For simple models, the order of complexity of the log-target is not greater than that of the proposal, so it is assumed that the log-target scales as $\mathcal{O}(1)$. Thus, for simple models, the respective complexity bounds for MALA, SMMALA and MMALA reduce to $\mathcal{O}(n_\theta^2)$, $\mathcal{O}(n_\theta^2)$ and $\mathcal{O}(n_\theta^3)$ after dropping scaling factors and lower-order terms. The complexity order of the typical MALA with identity preconditioning matrix further reduces to $\mathcal{O}(n_\theta)$ for a simple model.

On the other hand, computationally expensive models imply $\mathcal{O}(f(n_\theta)) >> \mathcal{O}(n_\theta)$. For such models, the cost of computations implicating the log-target is much higher than the cost of proposal-related calculations. In other words, if the log-target is of high complexity, then derivative computations supersede linear algebra calculations, and this is why the computational cost of a manifold MCMC algorithm tends to be reported as a function of the order of derivatives appearing in the algorithm. For instance, the complexity of SMMALA, which scales as $\mathcal{O}(f(n_\theta)n_\theta^2 + n_\theta^3 + 2n_\theta^{2.373})$, can be simply written as $\mathcal{O}((f(n_\theta)n_\theta^2)$ for a computationally intensive model.

A typical example of a computationally expensive model is a system of non-linear ordinary differential equations (ODEs), where each log-target calculation requires solving the ODE system numerically. It is then expected that the log-target and its derivative evaluations will dominate the cost of Langevin Monte Carlo simulations according to the discussed complexity bounds.

A detailed account of the time complexity of SMMALA has been given to highlight the main sources of computational load involved in geometric updates. In what follows, scaling factors and constants will be suppressed in big-$\mathcal{O}$ notation following common practice.

Combining the best of two worlds, an effective and efficient MCMC algorithm would exploit geometric information in time as close as possible to the linear time of MALA. One potential approach towards this end would be to approximate the expensive components of geometric computation, such as $G^{-1}(\boldsymbol{\theta})$ and $\sqrt{G^{-1}(\boldsymbol{\theta})}$, in the spirit of Şimşekli et al. (2016). An alternative route, taken by the present paper, is to design a hybrid sampling method that switches between expensive geometrically-informed and cheaper non-geometric proposal mechanisms. The latter idea is motivated by the objective to exploit exact geometric updates whenever these occur, therefore not rendering each Monte Carlo iteration approximate.

3. Partial Manifold Langevin Metric Updates

3.1 Partial Manifold Langevin Metric Updates in MALA

Sections 2.1 and 2.2 provide a unified formulation of MALA and SMMALA. More specifically, both samplers constitute variations of a Metropolis-Hastings algorithm with a normal proposal given by (1) and acceptance probability defined by (2). The MALA and SMMALA normal proposals differ in their locations and covariances, which are specified by (3)-(4) and (8)-(7) respectively. From an algorithmic perspective, the natural way to switch between MALA and SMMALA updates is to variably rely on the locations and covariances of their associated candidate-generating kernels.

Both MALA and SMMALA learn from the local geometry of the underlying manifold. MALA resorts to gradient flow computations, while SMMALA further enriches its geometric information by incorporating the position-specific metric $G(\boldsymbol{\theta})$ to the updating mechanism. So, strictly speaking, a hybrid algorithm that switches between MALA and SMMALA proposals combines two geometric MCMC methods with varying levels of geometric information and of computational cost. To keep the resulting complexity as low as possible, the bulk of transitions should be driven by MALA, allowing intervening SMMALA updates to occur just frequently enough to boost sampling effectiveness.

A first question is how to pick from the MALA and SMMALA proposals at each MCMC iteration. One possibility is to make a random choice via a Bernoulli trial. At the i-th iteration, draw a sample $b(i) \sim \text{Bernoulli}(p(i))$ with probability $p(i)$ of taking a SMMALA step. If $b(i) = 0$ then propose a new state $\boldsymbol{\theta}^\star$ sampled from the MALA proposal, whereas if $b(i) = 1$ then propose $\boldsymbol{\theta}^\star$ drawn from the SMMALA proposal.

Secondly, there needs to be a way of selecting the probabilities $p(i)$ as a function of the current iteration $i \in \{1, 2, \cdots, n_m\}$, where n_m is the total number of MCMC steps. In other words, a schedule is needed for the probabilities $p(i)$ of choosing between MALA and SMMALA updates. One possible choice of $p(i)$ is an exponentially decaying schedule

$$p(i) = \exp\left(-a(i-1)/n_m\right), \tag{10}$$

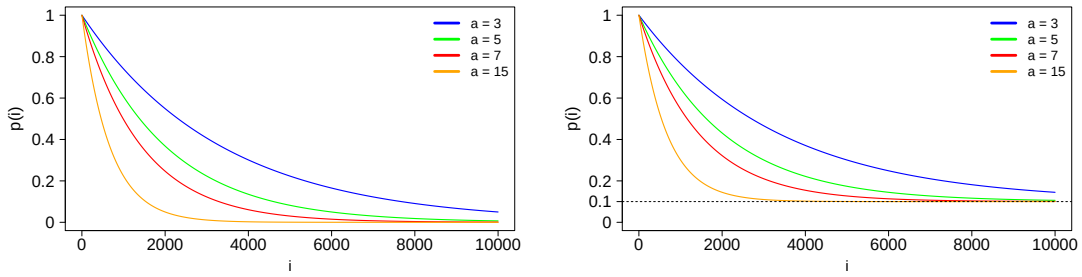
where a denotes a constant tuning factor that determines the decay rate of $p(i)$. Schedule (10) makes SMMALA updates more probable in early transient iterations of the chain, while it makes MALA updates increasingly more likely in late stationary iterations. Hence, as the chain converges, less geometric information is acquired to reduce the computational load. Figure 1a displays the schedule of Bernoulli probabilities for varying decaying rates. This schedule is likely to be efficient for problems well approximated by a single metric, even if an appropriate metric is not known a priori.

For other models, the curvature of the target density may change significantly over the region of parameter space with significant probability density. In such cases, rare updates to the metric after many iterations would lead to inefficient proposals. For such problems, it would be desirable to maintain a certain probability b of making SMMALA proposals in late Monte Carlo iterations instead of eliminating this probability. Schedule (10) can be modified as

$$p(i) = (1-b)\exp\left(-a(i-1)/n_m\right) + b, \tag{11}$$

where the additional tuning factor b sets the asymptotic probability of using the SMMALA proposal mechanism in late MCMC iterations. Obviously (10) arises as a special case of (11) by setting $b = 0$ in the latter equation. Figure 1b visualizes schedule (11) for varying decaying rates and for $b = 0.1$, which corresponds to a 10% probability of picking SMMALA proposals as the simulation progresses towards its end.

A third and last question to be addressed in order to fully construct the hybrid sampler is how to set the preconditioning matrix for MALA proposals. Perhaps the simplest option would be to utlilize an identity preconditioning matrix $G = I$ corresponding to MALA proposal densities of the form $\mathcal{N}(\boldsymbol{\theta} + \frac{\epsilon^2}{2}\nabla_\theta \log p(\boldsymbol{\theta}), \epsilon^2 I)$, thus adhering to the most commonly used version of MALA. A more sophisticated algorithm is proposed in this paper that reuses the most recent metric $G(\boldsymbol{\theta}^0)$ computed at the latest state $\boldsymbol{\theta}^0$ obtained via a SMMALA



(a) $b = 0$ (b) $b = 0.1$

Figure 1: Exponentially decaying schedule of probabilities $p(i)$ of making SMMALA proposals as a function of i-th MCMC iteration. The tuning factor b sets the asymptotic probability of selecting the SMMALA proposal mechanism over its MALA counterpart. On the other hand, the tuning factor a determines the rate of decay of $p(i)$ with larger a corresponding to more rapid decay.

update. This offers the benefits of making use of geometric information at no additional cost. The improvement in computational efficiency comes from caching and reusing $G(\boldsymbol{\theta}^0)$, $G^{-1}(\boldsymbol{\theta}^0)$ and $\sqrt{G^{-1}(\boldsymbol{\theta}^0)}$ for the intervening MALA steps between successive SMMALA updates.

The resulting algorithm is abbreviated as ALSMMALA. The AL prefix implies that the bulk of states are drawn from a MALA proposal mechanism and the SMMALA suffix indicates that the local geometry is exploited via more expensive SMMALA updates. Algorithm 1 provides pseudocode to summarize ALSMMALA for some schedule $p(i)$.

3.2 Partial Manifold Langevin Metric Updates in Adaptive Metropolis

The focal idea of this paper extends beyond ALSMMALA. In principle, it is suggested to allow the proposal mechanism to vary between MCMC iterations. For example, most states can be sampled from a cheaper candidate-generating kernel, whereas fewer states can be drawn from a more costly proposal that uses local geometric information.

The candidate-generating kernel that contributes the majority of states to a hybrid sampler with a dual proposal mechanism impacts the statistical properties of the sampler. For instance, Roberts and Tweedie (1996) and Livingstone and Girolami (2014) have shown that if a target density has tails heavier than exponential or lighter than Gaussian, then a MALA proposal kernel does not yield a geometrically ergodic Markov chain. For targets with such tail behaviour, ALSMMALA is not anticipated to admit geometric ergodicity either.

Depending on the target density, different proposal mechanisms might need to be combined to attain a hybrid sampler with the desired statistical properties and sampling efficiency. In order to define a viable complement to ALSMMALA and to further explore how

Algorithm 1 ALSMMALA

$\boldsymbol{\theta}^{\circ} = \boldsymbol{\theta}$

for $k = 1$ to n_m **do**
  Sample $b \sim \text{Bernoulli}(p(i))$

  Sample $u \sim \mathcal{U}(0,1)$

  if $b = 0$ **then** ▷ Use MALA proposal
    Sample $\boldsymbol{\theta}^{*} \sim \mathcal{N}(\boldsymbol{\mu}(\boldsymbol{\theta}, G(\boldsymbol{\theta}^{\circ}), \epsilon), \epsilon^2 G^{-1}(\boldsymbol{\theta}^{\circ}))$

    if $u < \dfrac{p(\boldsymbol{\theta}^{*})\mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\mu}(\boldsymbol{\theta}^{*}, G(\boldsymbol{\theta}^{\circ}), \epsilon), \epsilon^2 G^{-1}(\boldsymbol{\theta}^{\circ}))}{p(\boldsymbol{\theta})\mathcal{N}(\boldsymbol{\theta}^{*}|\boldsymbol{\mu}(\boldsymbol{\theta}, G(\boldsymbol{\theta}^{\circ}), \epsilon), \epsilon^2 G^{-1}(\boldsymbol{\theta}^{\circ}))}$ **then**
      $\boldsymbol{\theta} = \boldsymbol{\theta}^{*}$
    end if
  else if $b = 1$ **then** ▷ Use SMMALA proposal
    Sample $\boldsymbol{\theta}^{*} \sim \mathcal{N}(\boldsymbol{\mu}(\boldsymbol{\theta}, G(\boldsymbol{\theta}), \epsilon), \epsilon^2 G^{-1}(\boldsymbol{\theta}))$

    if $u < \dfrac{p(\boldsymbol{\theta}^{*})\mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\mu}(\boldsymbol{\theta}^{*}, G(\boldsymbol{\theta}^{*}), \epsilon), \epsilon^2 G^{-1}(\boldsymbol{\theta}^{*}))}{p(\boldsymbol{\theta})\mathcal{N}(\boldsymbol{\theta}^{*}|\boldsymbol{\mu}(\boldsymbol{\theta}, G(\boldsymbol{\theta}), \epsilon), \epsilon^2 G^{-1}(\boldsymbol{\theta}))}$ **then**
      $\boldsymbol{\theta} = \boldsymbol{\theta}^{*}$
    end if

    $\boldsymbol{\theta}^{\circ} = \boldsymbol{\theta}$
  end if
end for

to combine heterogeneous proposal mechanisms under a single MCMC algorithm, one more hybrid sampler with partial geometric updates will be constructed.

The new algorithm will sample most of its states from a proposal kernel inspired by the adaptive Metropolis (AM) kernel of Haario et al. (2001), while it will draw fewer states from the SMMALA proposal kernel. Given its candidate-generating constituents, the new algorithm will be called AMSMMALA. The examples of Section 4 will demonstrate the relative performance of ALSMMALA and AMSMMALA for different target densities.

Initially, a concise account of the adaptive Metropolis algorithm of Haario et al. (2001) is given. Consider a hitherto simulated chain $(\boldsymbol{\theta}_0, \boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_k)$, denoted by $\boldsymbol{\theta}_{0:k}$. It is noted that $\boldsymbol{\theta}_i \in \mathbb{R}^{n_\theta}$ refers to the i-th n_θ-dimensional state of the chain, to be distinguished from the parenthesized subscript $\boldsymbol{\theta}_{(i)} \in \mathbb{R}$, which represents the i-th scalar coordinate of some state $\boldsymbol{\theta} \in \mathbb{R}^{n_\theta}$. A candidate state $\boldsymbol{\theta}^{\star}$ is sampled from the normal density $\mathcal{N}(\boldsymbol{\theta}_k, \hat{C}(\boldsymbol{\theta}_k|\boldsymbol{\theta}_{0:k-1}))$. This proposal density is commonly centred on the last state $\boldsymbol{\theta}_k$ and its covariance matrix $\text{Cov}(\boldsymbol{\theta}_k)$ is estimated adaptively by

$$\hat{C}(\boldsymbol{\theta}_k|\boldsymbol{\theta}_{0:k-1}) = s_{n_\theta}\widehat{\text{Cov}}(\boldsymbol{\theta}_k|\boldsymbol{\theta}_{0:k-1}) + s_{n_\theta}\lambda I, \tag{12}$$

where $\widehat{\mathrm{Cov}}(\boldsymbol{\theta}_k|\boldsymbol{\theta}_{0:k-1})$ is the empirical covariance matrix

$$\widehat{\mathrm{Cov}}(\boldsymbol{\theta}_k|\boldsymbol{\theta}_{0:k-1}) = \frac{1}{k-1}\left(\sum_{i=0}^{k-1}\boldsymbol{\theta}_i\boldsymbol{\theta}_i^T - k\bar{\boldsymbol{\theta}}_{k-1}\bar{\boldsymbol{\theta}}_{k-1}^T\right). \tag{13}$$

$\bar{\boldsymbol{\theta}}_{k-1}$ denotes the sample mean of $\boldsymbol{\theta}_{0:k-1}$. The tuning parameter s_{n_θ} depends only on the dimension n_θ of the parameter space and is used for scaling the empirical covariance. Furthermore, $\lambda > 0$ is another tuning constant, which is chosen to be very small and is needed for preventing the estimator $\hat{C}(\boldsymbol{\theta}_k|\boldsymbol{\theta}_{0:k-1})$ from becoming degenerate for larger k. The candidate state $\boldsymbol{\theta}^\star$ is accepted with probability

$$a(\boldsymbol{\theta}_k, \boldsymbol{\theta}^\star) = \min\left\{\frac{p(\boldsymbol{\theta}^\star)}{p(\boldsymbol{\theta}_k)}, 1\right\}. \tag{14}$$

Since the proposal kernel $\mathcal{N}(\boldsymbol{\theta}^\star|\boldsymbol{\theta}_k, \hat{C}(\boldsymbol{\theta}_k|\boldsymbol{\theta}_{0:k-1}))$ is symmetric, it does not appear in the acceptance ratio (14).

To enable switching betweeen AM and SMMALA transitions, the covariance matrices of the two involved proposals must be brought together under a common framework. If a SMMALA proposal $\boldsymbol{\theta}^\star$ is made at the k-th MCMC iteration, the local covariance at the current state $\boldsymbol{\theta}_k$ is evaluated exactly as $\mathrm{Cov}(\boldsymbol{\theta}_k) = \epsilon^2 G^{-1}(\boldsymbol{\theta}_k)$. If on the other hand a state $\boldsymbol{\theta}^\star$ is proposed via adjusted Metropolis at the k-th iteration, the metric $G(\boldsymbol{\theta}_k)$ at the current state $\boldsymbol{\theta}_k$ may or may not be known depending on whether a SMMALA or AM step was taken at the previous iteration; if a SMMALA step was taken at the previous iteration, then $G^{-1}(\boldsymbol{\theta}_k)$ is available, otherwise an empirical estimator $\widehat{G^{-1}}(\boldsymbol{\theta}_k|\boldsymbol{\theta}_{0:k-1})$ of $G^{-1}(\boldsymbol{\theta}_k)$ must be obtained. Setting the diffusion drift step ϵ temporarily aside, it becomes clear that $G^{-1}(\boldsymbol{\theta}_k)$ is either evaluated exactly by SMMALA or it is estimated by the empirical covariance $\widehat{G^{-1}}(\boldsymbol{\theta}_k|\boldsymbol{\theta}_{0:k-1}) = \widehat{\mathrm{Cov}}(\boldsymbol{\theta}_k|\boldsymbol{\theta}_{0:k-1})$ as given by (13).

As long as SMMALA steps are embedded throughout the AMSMMALA simulation, the AM estimator $\widehat{G^{-1}}(\boldsymbol{\theta}_k|\boldsymbol{\theta}_{0:k-1})$ is corrected regularly by SMMALA exact inverse metric evaluations. Put another way, recurring geometric updates prevent the empirical covariance of AM from becoming degenerate. Hence, the $s_{n_\theta}\lambda I$ term in (12) is omitted by setting $\lambda = 0$. Moreover, the tuning parameter s_{n_θ} of AM is set to $s_{n_\theta} = \epsilon^2$ in (12) so that $\widehat{G^{-1}}(\boldsymbol{\theta}_k|\boldsymbol{\theta}_{0:k-1})$ and $G^{-1}(\boldsymbol{\theta}_k)$ are coherently scaled by the same factor in AM and SMMALA updates respectively. Setting $\lambda = 0$ and $s_{n_\theta} = \epsilon^2$ in (12) defines the empirical covariance estimator $\hat{C}(\boldsymbol{\theta}_k|\boldsymbol{\theta}_{0:k-1}) = \epsilon^2\widehat{G^{-1}}(\boldsymbol{\theta}_k|\boldsymbol{\theta}_{0:k-1})$ for the AM proposal density at the k-th step of AMSMMALA, where $\widehat{G^{-1}}(\boldsymbol{\theta}_k|\boldsymbol{\theta}_{0:k-1})$ is given by (13).

To ensure that the adaptive Metropolis covariance estimator receives exact geometric updates regularly, a $\bmod(a)$ scheduler can be employed. If the k-th MCMC iteration is a multiple of a tuning constant a, then a SMMALA update is attempted, else an AM proposal is made.

It follows from (13) that the empirical inverse metric $\widehat{G^{-1}}(\boldsymbol{\theta}_k|\boldsymbol{\theta}_{0:k-1})$, shortly denoted as $\widehat{G^{-1}}(\boldsymbol{\theta}_k)$, at the k-th AMSMMALA iteration calculates recursively as

$$(k-1)\widehat{G^{-1}}(\boldsymbol{\theta}_k) = (k-2)M(\boldsymbol{\theta}_{k-1}) + \boldsymbol{\theta}_{k-1}\boldsymbol{\theta}_{k-1}^T - k\bar{\boldsymbol{\theta}}_{k-1}\bar{\boldsymbol{\theta}}_{k-1}^T + (k-1)\bar{\boldsymbol{\theta}}_{k-2}\bar{\boldsymbol{\theta}}_{k-2}^T. \tag{15}$$

The sample means in (15) are also calculable recursively according to

$$k\bar{\boldsymbol{\theta}}_k = (k-1)\bar{\boldsymbol{\theta}}_{k-1} + \boldsymbol{\theta}_k. \tag{16}$$

Apparently, (15) and (16) help compute the empirical covariance of AMSMMALA faster than relying on (13).

The AM proposal kernel $\mathcal{N}(\boldsymbol{\theta}_k, \epsilon^2 M(\boldsymbol{\theta}_k))$ at the k-th iteration of AMSMMALA has covariance that is either known exactly from the previous SMMALA step or it is estimated from the previous AM step, therefore

$$M(\boldsymbol{\theta}_k) = \begin{cases} G^{-1}(\boldsymbol{\theta}_k), & \text{if } k-1 \text{ step taken by SMMALA} \\ \widehat{G^{-1}}(\boldsymbol{\theta}_k), & \text{if } k-1 \text{ step taken by AM} \end{cases}. \tag{17}$$

Algorithm 2 summarizes AMSMMALA with the help of pseudocode. The empirical covariance $\widehat{G^{-1}}(\boldsymbol{\theta}_k)$ in Algorithm 2 is computed via (15).

The AM steps of AMSMMALA waive the cost of proposal evaluations thanks to the acceptance probability (14) and do not require computing log-target derivatives. However, sampling a candidate state from the AM proposal kernel creates a complexity bottleneck of $\mathcal{O}(n_\theta^3)$ due to the associated Cholesky decomposition of the empirical covariance. The recursive formula (15) comes to rescue, as it allows to replace the Cholesky factorization by two rank one updates and one rank one downdate, thus reducing the complexity of AM updates from $\mathcal{O}(n_\theta^3)$ to $\mathcal{O}(3n_\theta^2)$. Gill et al. (1974) and Seeger (2004) elaborate on low rank updates for Cholesky decompositions.

AMSMMALA yields two more advantages over AM apart from exploiting local geometric information. An initial covariance matrix $\hat{C}_0 = \epsilon^2\widehat{G^{-1}}(\boldsymbol{\theta}_0)$ is needed for starting up the recursion in (15). Haario et al. (2001) propose to choose $\hat{C}_0$ on the basis of prior knowledge. If prior elicitation for $\hat{C}_0$ turns to be non-trivial, AMSMMALA gives the option to initialize $C_0 = \epsilon^2 G^{-1}(\boldsymbol{\theta}_0)$ by a SMMALA update. Additionally, the recurring corrections of the empirical covariance via exact SMMALA updates make AMSMMALA available to target densities of non-bounded support, extending this way the range of applicability of the original AM algorithm by Haario et al. (2001).

3.3 Choice of Update Schedule

The two proposal mechanisms along with the schedule for picking between them comprise the three main building blocks of any resulting hybrid sampler. ALSMMALA and AMSMMALA have manifested ways of combining cheap with more expensive geometric updates to construct a new MCMC algorithm. The underlying proposals set some limitations in the choice of schedule. AMSMMALA for example needs to receive exact metric updates throughout the course of simulation to correct the empirical covariance used in adaptive Metropolis steps. Despite any inherent limitations posed by the proposal mechanisms, there is room for experimentation when choosing a schedule.

For instance, a geometric distribution Geometric(p) could be used for regulating the occurrence of metric updates in AMSMMALA, where p is a tuning parameter giving the probability of making a SMMALA proposal. Setting $p = 1/(1+a)$ yields an expected number of $(1-p)/p = a$ AM steps before a new SMMALA proposal is made. This way,

Algorithm 2 AMSMMALA

for $k = 1$ to n_m **do**
  if $k \equiv 0$ (mod (a)) **then**
    $b_k = 1$
  else
    $b_k = 0$
  end if

  Sample $u \sim \mathcal{U}(0, 1)$

  if $b_k = 0$ **then** ▷ Use adaptive Metropolis proposal
    if $b_{k-1} = 1$ **then**
      $M(\boldsymbol{\theta}_k) = G^{-1}(\boldsymbol{\theta}_k)$
    else if $b_{k-1} = 0$ **then**
      $M(\boldsymbol{\theta}_k) = \widehat{G^{-1}}(\boldsymbol{\theta}_k)$
    end if

    Sample $\boldsymbol{\theta^*} \sim \mathcal{N}(\boldsymbol{\theta}_k, \epsilon^2 M(\boldsymbol{\theta}_k))$

    $r = \dfrac{p(\boldsymbol{\theta}^*)}{p(\boldsymbol{\theta}_k)}$
  else if $b_k = 1$ **then** ▷ Use SMMALA proposal
    Sample $\boldsymbol{\theta^*} \sim \mathcal{N}(\boldsymbol{\mu}(\boldsymbol{\theta}_k, G(\boldsymbol{\theta}_k), \epsilon), \epsilon^2 G^{-1}(\boldsymbol{\theta}_k))$

    $r = \dfrac{p(\boldsymbol{\theta}^*)\mathcal{N}(\boldsymbol{\theta}_k | \boldsymbol{\mu}(\boldsymbol{\theta}^*, G(\boldsymbol{\theta}^*), \epsilon), \epsilon^2 G^{-1}(\boldsymbol{\theta}^*))}{p(\boldsymbol{\theta}_k)\mathcal{N}(\boldsymbol{\theta}^* | \boldsymbol{\mu}(\boldsymbol{\theta}_k, G(\boldsymbol{\theta}_k), \epsilon), \epsilon^2 G^{-1}(\boldsymbol{\theta}_k))}$
  end if

  if $u < r$ **then**
    $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}^*$
  else
    $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k$
  end if
end for

the Geometric$(1/(1 + a))$ schedule is a stochastic analogue to the deterministic mod (a) schedule of Algorithm 2.

Other samplers with dual proposal mechanism, such as ALSMMALA, do not need regular geometric steps, thence it makes sense to exploit local geometric information mostly in the early transient phases of the simulated chain and to reduce computational load via a cheaper proposal in late stationary phases. A Bernoulli(p(i)) distribution comes in handy, leaving space for experimentation with the choice of schedule for $p(i)$. The probability $p(i)$ plays a role analogous to temperature in the cooling schedule of simulated annealing. This analogy brings up a prolific body of literature to open up possibilities for scheduling the cooling of probability $p(i)$ of geometric updates.

Cooling	$p(i)$
Exponential	$(1-b)\exp(-a(i-1)/n_m)+b$
Linear	$\dfrac{1-b}{1+a(i-1)/n_m}+b$
Quadratic	$\dfrac{1-b}{1+a((i-1)/n_m)^2}+b$
Logarithmic	$\dfrac{(1-b)}{1+a\log(1+(i-1)/n_m)}+b$

Table 1: Different types of cooling schedule for probability $p(i),\ i \in \{1, 2, \ldots, n_m\}$, of making a geometric proposal as a function of the i-th MCMC iteration. Such cooling for $p(i)$ is plausible if the dual proposal scheme does not require regular geometric updates throughout the simulation.

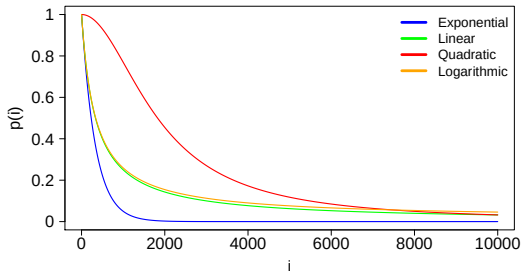


Figure 2: Visualization of four cooling schedule types of Table 1 for probability $p(i)$ of making a geometric proposal as a function of the i-th MCMC iteration. The displayed schedules have been generated using $a = 30$ and $b = 0$.

Table 1 suggests four cooling schedules for $p(i)$, which have been acquired empirically and are variants of temperature schedules previously discussed by Hajek (1987) and Nourani and Andresen (1998). Figure 2 displays instances of the four schedules of Table 1.

3.4 Expected Complexity

The concept of complexity carries three distinct meanings in the context of MCMC. Firstly, MCMC methods need to be tuned so as to achieve a balance between proposing large enough jumps and ensuring that a reasonable proportion of jumps are accepted. By way of illustration, MALA attains its optimal acceptance rate of 0.574 as $n_\theta \to \infty$ by setting its drift step ϵ to be in the vicinity of $n_\theta^{-1/3}$. Because of this, it is said that the algorithmic efficiency of MALA scales $\mathcal{O}(n_\theta^{-1/3})$ as the number n_θ of parameters increases.

Secondly, the quality of MCMC algorithms is assessed by estimating the effective sample size (ESS) of their simulated chains. The ESS of a chain is interpreted as the number of samples in the chain bearing the same amount of variance as the one found in n_m independent samples.

A third criterion for assessing MCMC methods is their computational time. This criterion corresponds to the ordinary concept of algorithmic complexity as it entails a count of numerical operations performed by an MCMC algorithm. To give an example, the computational complexity of MALA with an identity preconditioning matrix for a simple model is of order $\mathcal{O}(n_\theta)$, as explained in Section 2.3.

Of these three indicators of algorithmic complexity, ESS and computational runtime are the ones typically used for understanding the range of applicability of MCMC methods. To get a single-number summary, the ratio of ESS over runtime is usually employed. In the present paper, the computational time of ALSMMALA and AMSMMALA are derived theoretically, while both samplers are empirically assessed in Section 4 via their ESS and CPU runtime.

Proposition 1 *Let A_1 and A_2 be two MCMC algorithms whose proposal mechanisms have computational complexities c_1 and c_2, respectively. A composite MCMC algorithm A conducts a Bernoulli trial $B(i) \sim$ Bernoulli$(p(i))$, $i \in \{1, 2, \ldots, n_m\}$, to take its i-th step using the proposal kernel of algorithm A_1 with probability $p(i)$, or using the proposal mechanism of A_2 with probability $1 - p(i)$. The mean complexity of A is given by*

$$\mathcal{O}\left(\frac{\sum_{i=1}^{n_m} p(i)}{n_m} c_1 + \frac{n_m - \sum_{i=1}^{n_m} p(i)}{n_m} c_2\right). \tag{18}$$

Proof The expected number of invocations of the proposal kernel associated with algorithm A_1 equals

$$E\left(\sum_{i=1}^{n_m} B(i)\right) = \sum_{i=1}^{n_m} E(B(i)) = \sum_{i=1}^{n_m} p(i),$$

whence the conclusion follows directly. ■

Proposition 2 *If an exponentially decaying schedule $p(i)$, as described by* (10)*, is used for regulating the choice of proposal kernel at each step of algorithm A, then the expected complexity of A becomes*

$$\mathcal{O}\left(\frac{1 - \exp(-a)}{n_m(1 - \exp(-a/n_m))} c_1 + \left(1 - \frac{1 - \exp(-a)}{n_m(1 - \exp(-a/n_m))}\right) c_2\right). \tag{19}$$

Proof Schedule (10) yields the geometric series sum

$$\sum_{i=1}^{n_m} p(i) = \sum_{i=1}^{n_m} \exp(-a(i-1)/n_m) = \frac{1 - \exp(-a)}{1 - \exp(-a/n_m)}. \tag{20}$$

Plugging (20) into (18) gives (19). ■

Lemma 3 *For sufficiently large number n_m of MCMC iterations ($n_m \to \infty$), the mean complexity of an algorithm A with exponentially decaying schedule* (10) *is.*

$$\mathcal{O}\left(\frac{1-\exp(-a)}{a}c_1 + \left(1 - \frac{1-\exp(-a)}{a}\right)c_2\right). \tag{21}$$

Proof It suffices to notice that

$$\lim_{n_m\to\infty} \frac{1-\exp(-a)}{n_m(1-\exp(-a/n_m))} = \frac{1-\exp(-a)}{a}. \tag{22}$$

■

Proposition 4 *The mean complexity of an algorithm A with a* $\mod(a)$ *schedule $p(i)$ is*

$$\mathcal{O}\left(\frac{1}{a}c_1 + \frac{a-1}{a}c_2\right). \tag{23}$$

Proof The proposal mechanism of algorithm c_1 is called n_m/a times out of n_m iterations. Hence, the proportion of proposals made via c_1 is $1/a$, which completes the proof. ■

Lemma 5 *For a log-target with complexity of order $\mathcal{O}(f(n_\theta)) >> \mathcal{O}(n_\theta)$ and for a large number n_m of iterations ($n_m \to \infty$), the expected complexity of ALSMMALA, as presented in Algorithm 1 and equipped with the exponentially decaying schedule defined by* (10)*, is given by*

$$\mathcal{O}\left(\frac{1-\exp(-a)}{a}f(n_\theta)n_\theta^2 + \left(1 - \frac{1-\exp(-a)}{a}\right)f(n_\theta)n_\theta\right). \tag{24}$$

Proof (24) follows from (21) by recalling that the respective complexities of SMMALA and MALA for a log-target with complexity of order $\mathcal{O}(f(n_\theta)) >> \mathcal{O}(n_\theta)$ are $\mathcal{O}(f(n_\theta)n_\theta^2)$ and $\mathcal{O}(f(n_\theta)n_\theta)$. ■

Lemma 6 *For a log-target with complexity of order $\mathcal{O}(f(n_\theta)) >> \mathcal{O}(n_\theta)$, the expected complexity of AMSMMALA, as defined in Algorithm 2, is*

$$\mathcal{O}\left(\frac{1}{a}f(n_\theta)n_\theta^2 + \frac{a-1}{a}f(n_\theta)\right). \tag{25}$$

Proof Setting $c1 = f(n_\theta)n_\theta^2$ and $c_2 = f(n_\theta)$ in (23), which correspond to the complexity orders of SMMALA and AM for a log-target with complexity of order $\mathcal{O}(f(n_\theta)) >> \mathcal{O}(n_\theta)$, gives (25). ■

Lemma 7 *For a log-target with complexity of order $\mathcal{O}(1)$ and for a large number n_m of iterations ($n_m \to \infty$), the expected complexity of ALSMMALA, as presented in Algorithm 1 and equipped with the exponentially decaying schedule defined by* (10)*, is given by*

$$\mathcal{O}\left(\frac{1-\exp(-a)}{a} n_\theta^3 + \left(1 - \frac{1-\exp(-a)}{a}\right) n_\theta^2\right). \tag{26}$$

Proof (26) is derived from (21) by observing that the respective complexities of SMMALA and MALA for a log-target with complexity of order $\mathcal{O}(1)$ are $\mathcal{O}(n_\theta^3)$ and $\mathcal{O}(n_\theta^2)$. ■

Lemma 8 *For a log-target with complexity of order $\mathcal{O}(1)$, the expected complexity of AMSMMALA, as defined in Algorithm 2, is*

$$\mathcal{O}\left(\frac{1}{a} n_\theta^3 + \frac{a-1}{a} n_\theta^2\right). \tag{27}$$

Proof Setting $c1 = n_\theta^3$ and $c_2 = n_\theta^2$ in (23), which represent the complexity orders of SMMALA and AM for a log-target with complexity of order $\mathcal{O}(1)$, yields (27). ■

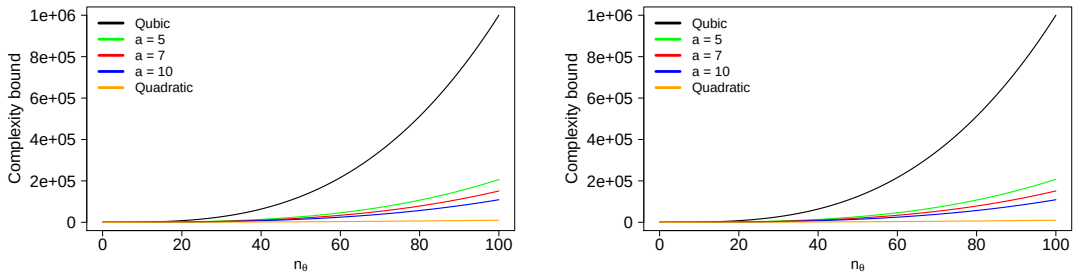
The schedule $p(i)$ for choosing between proposal mechanisms plays a decisive role in the complexity of the resulting sampler as seen from (18). Besides, the decaying rate a of an exponentially decaying schedule (10) and the modulus a of a $\mod(a)$ schedule can be used for adjusting the computational cost according to (21) and (23).

Figure 3 displays the complexity of ALSMMALA with an exponential schedule specified by (10) and of AMSMMALA with a $\mod(a)$ schedule as a function of the parameter space dimension n_θ for computationally cheap models. The curves of Figures 3a and 3b correspond to varying levels of decaying rate a for ALSMMALA and of modulus a for AMSMMALA. In both cases, the tuning factor a determines the trade-off between runtime and amount of geometric computation. Larger values of a reduce the runtime due to fewer geometric updates, whereas smaller values of a increase the complexity as they induce more geometric updates.

3.5 Analytically Intractable Geometric Updates

In practice, challenges in the implementation of manifold MCMC algorithms might raise additional computational implications. In particular, two notoriously recurring issues relate to the Cholesky decomposition of metric $G^{-1}(\boldsymbol{\theta})$ and to the calculation of up to third order derivatives of $G(\boldsymbol{\theta})$.

Various contributing factors, including finite-precision floating point arithmetic, can lead to an indefinite proposal covariance matrix $\epsilon^2 G^{-1}(\boldsymbol{\theta})$. This in turn breaks the Cholesky factorization of $\epsilon^2 G^{-1}(\boldsymbol{\theta})$. Betancourt (2013) introduced the so-called SoftAbs metric, which offers a positive definite approximation of the indefinite proposal covariance. The working SoftAbs solution to the Cholesky factorization problem involves an eigen-decomposition, so it comes with an $\mathcal{O}(n_\theta^3)$ cost. Other research avenues pursuit different positive definite approximations of indefinite matrices. The works of Higham (1988) and Higham (2002) for



(a) ALSMMALA (b) AMSMMALA

Figure 3: Time complexity of ALSMMALA with an exponential schedule given by (10) and of AMSMMALA with a mod (a) schedule as a function of the number n_θ of parameters for log-targets with complexity of order $\mathcal{O}(1)$. Different curves correspond to varying levels of decaying rate a for ALSMMALA and of modulus a for AMSMMALA. The cubic complexity of SMMALA and quadratic complexity of MALA with a preconditioning matrix other than the identity matrix are displayed as baseline cases.

example approximate the nearest symmetric positive semi-definite matrix in the Frobenius norm to an arbitrary real matrix. If future research succeeds in adapting the approach of Higham (1988) in the context of geometric MCMC methods, then a computationally cheaper alternative to the SoftAbs metric could become available.

Non-trivial models can render the analytic derivation of log-target derivatives impossible or impractical. Automatic differentiation (AD), a computationally driven research activity that has evolved since the mid 1950's, helps compute derivatives in a numerically exact way. Indeed, Griewank (2014) has shown that AD is backward stable in the sense of Wilkinson (1971). Thus, small perturbations of the original function due to machine precision still yield accurate derivatives calculated via AD.

There are different methods of automatic differentiation that mainly differ in the way they traverse the chain rule; reverse mode AD is better suited for functions $f : \mathbb{R}^n \to \mathbb{R}$, in contrast to forward mode AD that is more suitable for functions $f : \mathbb{R} \to \mathbb{R}^m$ (Griewank and Walther (2008)). Consequently, reverse mode AD is utilized for computing derivatives of probability distributions, and finds use in statistical inference. Reverse mode AD is not worse than that of the respective analytical derivatives of a target density in terms of complexity, but it poses high memory requirements. Hybrid AD procedures combining elements of forward and backward propagation of derivatives can be constructed for achieving a compromise between execution time and memory usage when differentiating functions of the form $f : \mathbb{R}^n \to \mathbb{R}^m$.

4. Example Applications

ALSMMALA and AMSMMALA are put to test via three examples to empirically validate these two MCMC algorithms and to assess their computational efficacy. Ten chains are simulated from each sampler of each example. $110,000$ iterations are run for the realization of each chain, of which the first $10,000$ burn-in are discarded, so $n_m = 100,000$ MCMC samples are retained.

The effective sample size (ESS) of each dimension $\boldsymbol{\theta}_{(i)},\ i \in \{1, 2, \ldots, n_\theta\}$, of the parameter vector $\boldsymbol{\theta}$ is computed as

$$\text{ESS} = \frac{n_m \hat{\sigma}^2_{IID}}{\hat{\sigma}^2_{MCMC}}, \tag{28}$$

where $\hat{\sigma}^2_{IID}$ and $\hat{\sigma}^2_{MCMC}$ denote the ordinary and Monte Carlo variance of the associated chain. The Monte Carlo variance $\hat{\sigma}^2_{MCMC}$ is calculated by means of the initial monotone sequence estimator of Geyer (1992). To get more reliable estimates, the ESS of each dimension and the CPU runtime are averaged by taking their respective means across every set of ten chains simulated from a sampler. The computational efficiency of each sampler is defined as the ratio $\min\{\text{ESS}_i : i = 1, 2, \ldots, n_\theta\}$ /time of the smallest ESS $\min\{\text{ESS}_i : i = 1, 2, \ldots, n_\theta\}$ among all n_θ dimensions over the simulation time. Finally, the relative speed-up of one MCMC method over another is set to be the ratio of their computational efficiencies.

A package, called PGUManifoldMC, has been developed to implement ALSMMALA and AMSMMALA using the Julia programming language. PGUManifoldMC is based on Lora, a package for MCMC inference written in Julia by one of the two authors. PGUManifoldMC is publicly available at `https://github.com/scidom/PGUManifoldMC.jl` under an MIT license. The PGUManifoldMC package ships with the three examples of this paper.

4.1 Bayesian Logistic Regression

As a first example, consider a Bayesian logistic regression model consisting of n_d samples and n_θ covariates. The log-likelihood of the model expresses as

$$L(\mathbf{y}, X|\boldsymbol{\theta}) = (X\boldsymbol{\theta})^T \mathbf{y} - \sum_{i=1}^{n_d} \log\left(1 + \exp(\boldsymbol{\theta}^T \mathbf{x}_{\mathbf{i},})\right), \tag{29}$$

where $y \in \{0, 1\}^{n_d}$ is the binary response variable, X the $n_d \cdot n_\theta$ design matrix, $\mathbf{x}_{\mathbf{i},}$ the i-th row of X and $\boldsymbol{\theta} \in \mathbb{R}^{n_\theta}$ the regression coefficients. A normal prior is assumed for the model parameters $\boldsymbol{\theta} \sim \pi(\boldsymbol{\theta}) = \mathcal{N}(\mathbf{0}, vI)$ for some positive hyperparameter v. The unnormalized log-target density for Bayesian logistic regression thus takes the form

$$\log\left(p(\boldsymbol{\theta}|\mathbf{y}, X)\right) = L(\mathbf{y}, X|\boldsymbol{\theta}) + \log\left(\pi(\boldsymbol{\theta})\right), \tag{30}$$

where the log-likelihood is given by (29).

Differentiating (29) gives the gradient of the log-likelihood

$$\nabla_{\boldsymbol{\theta}}(L(\mathbf{y}, X|\boldsymbol{\theta})) = X^T \left(\mathbf{y} - \frac{1}{1 + \exp\left(-X\boldsymbol{\theta}\right)}\right). \tag{31}$$

The gradient of the log-prior evaluates as

$$\nabla_{\boldsymbol{\theta}}(\log\left(\pi(\boldsymbol{\theta})\right)) = -\frac{1}{v}\boldsymbol{\theta}. \tag{32}$$

So, the gradient $\nabla_{\boldsymbol{\theta}}(\log(p(\boldsymbol{\theta}|\mathbf{y}, X)))$ of the log-target equals the sum of (31) and (32).

The metric of a log-target $\log(p)$ is conventionally taken to be the expected Fisher information of $\log(p)$, which is equal to the negative Hessian of log-likelihood L minus the Hessian of log-prior $\log(\pi)$ in a Bayesian setting. By differentiating (31) and (32), it is deduced that the metric $G(\boldsymbol{\theta}|X)$ for the Bayesian logistic regression model with a normal prior $\mathcal{N}(\mathbf{0}, vI)$ is

$$G(\boldsymbol{\theta}|X)) = X^T \Lambda(\boldsymbol{\theta}) X + \frac{1}{v} I, \tag{33}$$

where the $n_d \cdot n_d$ diagonal matrix $\Lambda(\boldsymbol{\theta}) = \text{diag}\left[p_i(1-p_i)\right]$ has diagonal elements

$$p_i = P(y_i = 1) = \exp(\boldsymbol{\theta}^T \mathbf{x_{i,}})/(1 + \exp(\boldsymbol{\theta}^T \mathbf{x_{i,}})),\ i \in \{1, 2, \ldots, n_d\}. \tag{34}$$

The gradient $\nabla_{\boldsymbol{\theta}}(\log(p(\boldsymbol{\theta}|\mathbf{y}, X)))$ of the Bayesian logistic log-target is needed for MALA proposals made by ALSMMALA. Both the log-target gradient $\nabla_{\boldsymbol{\theta}}(\log(p(\boldsymbol{\theta}|\mathbf{y}, X)))$ and metric $G(\boldsymbol{\theta}|X))$ are needed for SMMALA proposals made by either of ALSMMALA and AMSMMALA.

MALA, SMMALA, ALSMMALA and AMSMMALA are run on the bank dataset taken from Flury and Riedwyl (1988) with the Bayesian logistic regression log-target outlined in the current section and with the prior hyperparameter set to $v = 100$, which implies a flat non-informative prior. The bank dataset contains the measurements of $n_\theta = 4$ covariates on $n_d = 200$ Swiss banknotes, 100 of which are genuine and 100 counterfeit. The four covariates include the length of the bill, the width of the left and right edge, and the bottom margin width. The binary response variable is the type of banknote, 0 being genuine and 1 counterfeit.

Table 2 summarizes the sampling efficacy metrics for the Bayesian logistic example. MALA is taken as the baseline sampler in the speedup calculations. As expected, the effective sample size of SMMALA is higher across most of covariates in comparison to MALA. The relative advantage of SMMALA over MALA in terms of ESS is attributed to the fact that the former sampler exploits local geometric information via successive evaluations of metric $G(\boldsymbol{\theta})$. On the other hand, SMMALA is more than twice slower in comparison to MALA as the respective runtimes show. Overall, SMMALA exhibits a slowdown factor of 0.73 with respect to the MALA baseline. This is a demonstration of how the increased sampling effectiveness of manifold MCMC algorithms is outweighed by their computational cost.

AMSMMALA has higher ESS, with the exception of one covariate, and is thus slightly more effective than MALA. However, AMSMMALA appears to be somewhat slower than MALA, which is manifested in the 0.85 slowdown factor. Moreover, AMSMMALA has lower ESS than SMMALA, but is much faster than SMMALA. On the basis of their relative speedups, AMSMMALA does better than SMMALA but worse than MALA.

ALSMMALA has higher ESS than MALA and SMMALA. The simulations show that ALSMMALA has more than three times higher ESS than SMMALA, so it appears that the switching between MALA and SMMALA proposal mechanisms results in a hybrid sampler with increased sampling effectiveness. Moreover, ALSMMALA is slower than MALA and faster than SMMALA in terms of absolute CPU runtimes. The relative speedups show that ALSMMALA is superior than its MALA and SMMALA counterparts roughly by a factor of 2 and 3, respectively.

Method	AR	ESS				Time	Efficiency	Speedup
		θ_1	θ_2	θ_3	θ_4			
MALA	.60	23077	8039	8892	8562	3.05	2637.47	1.00
SMMALA	.69	15138	15246	15098	12989	6.78	1916.05	0.73
AMSMMALA	.25	9114	9159	9147	8635	3.84	2246.86	0.85
ALSMMALA	.63	31562	31046	30656	26535	4.82	5503.74	2.09

Table 2: Comparison of sampling efficacy between MALA, SMMALA, AMSMMALA and ALSMMALA by applying Bayesian logistic regression on the Swiss banknotes data. AR: acceptance rate; ESS: effective sample size; Time: CPU runtime in seconds; Efficiency: smaller ESS across four covariates divided by runtime; Speedup: ratio of computational efficiencies with a MALA baseline. All the tabulated numbers have been rounded to the second decimal place, apart from the effective sample sizes, which have been rounded to the nearest integer.

The AMSMMALA and ALSMMALA acceptance rates reported in Table 2 have been acquired by tuning the drift step empirically so as to maximize ESS. The mean acceptance rate of AMSMMALA across the ten simulated chains is 0.25, so it is close to the asymptotically optimal acceptance rate of 0.234 for random walk Metropolis algorithms (Roberts et al. (1997)). This is intuitively plausible, as the bulk of AMSMMALA proposals are drawn from the AM proposal kernel. In analogous fashion, the mean acceptance rate of ALSMMALA is 0.63, not far from the optimal rate of 0.574 for MALA, from which the bulk of proposals are made.

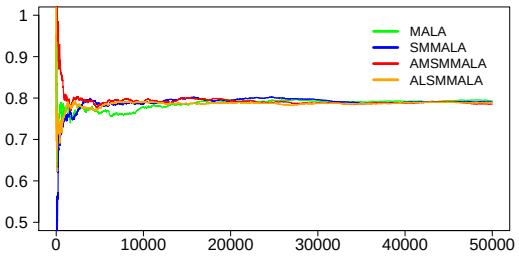
Figures 4c-4f visualize the traceplots of four chains corresponding to the regression coefficient of covariate $\boldsymbol{\theta}_{(2)}$, with a single chain simulated from each of MALA, SMMALA, AMSMMALA and ALSMMALA using the Bayesian logistic regression log-target on the Swiss banknotes data. All four overlaid running means in Figure 4a, plotted as a function of the number of iterations, demonstrate that the respective samplers converge towards the same value. The overlaid linear autocorrelations in Figure 4b for the four chains exhibit magnitudes that agree with the effective sample sizes of Table 2. More specifically, ALSMMALA has the smallest linear autocorrelation and highest ESS by far, followed by SMMALA, while AMSMMALA and MALA have roughly the same linear autocorrelation and ESS. All four traceplots of MALA, SMMALA and of the two algorithms of this paper exhibit strong mixing properties.

4.2 Bayesian Poisson Regression

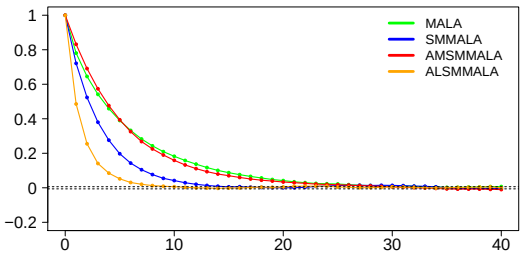
The log-likelihood of a Bayesian Poisson regression model with n_d samples and n_θ covariates has the form

$$L(\mathbf{y}, X|\boldsymbol{\theta}) = (X\boldsymbol{\theta})^T\mathbf{y} - \sum_{i=1}^{n_d} \left(\exp(\boldsymbol{\theta}^T\mathbf{x_{i,}}) + \log(y_i!)\right), \tag{35}$$

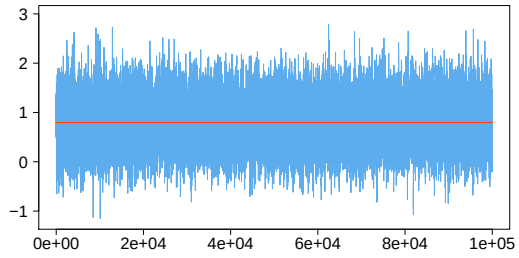
where $\mathbf{y} \in \mathbb{R}^d$ is the response variable, X the $n_d \cdot n_\theta$ design matrix and $\boldsymbol{\theta} \in \mathbb{R}^{n_\theta}$ the regression coefficients. By differentiating (35), the gradient $\nabla_{\boldsymbol{\theta}}(L(\mathbf{y}, X|\boldsymbol{\theta}))$ and the Hessian



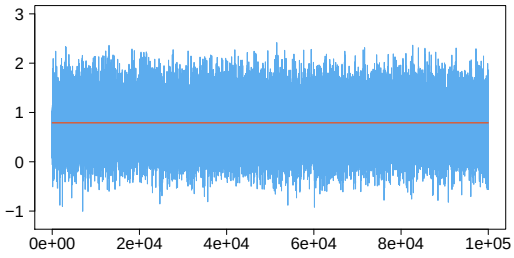
(a) Running mean



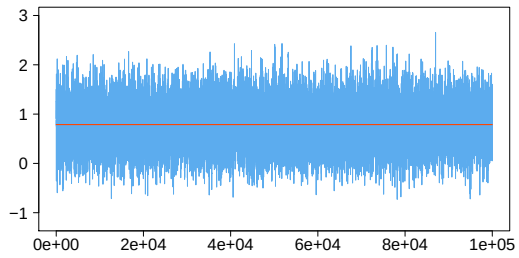
(b) Linear autocorrelation



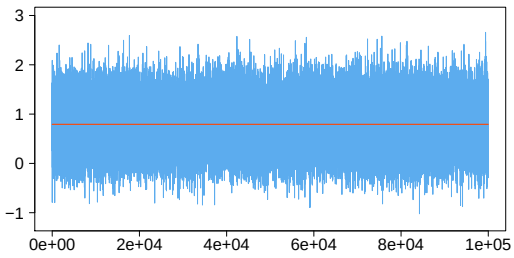
(c) MALA traceplot



(d) SMMALA traceplot



(e) AMSMMALA traceplot



(f) ALSMMALA traceplot

Figure 4: Plots of single chains corresponding to the regression coefficient of covariate $\boldsymbol{\theta}_{(2)}$ simulated from each of MALA, SMMALA, AMSMMALA and ALSMMALA samplers by applying Bayesian logistic regression on the Swiss banknotes data. (a) Overlaid running means, (b) overlaid linear autocorrelations for the four chains, (c)-(f) individual traceplots per chain (the red horizontal lines represent the Monte Carlo mean).

$H(L(\mathbf{y}, X|\boldsymbol{\theta}))$ of the Bayesian Poisson log-likelihood are found to be

$$\nabla_{\boldsymbol{\theta}}(L(\mathbf{y}, X|\boldsymbol{\theta})) = X^T\left(\mathbf{y} - \exp\left(X\boldsymbol{\theta}\right)\right), \tag{36}$$

$$H(L(\mathbf{y}, X|\boldsymbol{\theta})) = -X^T \text{diag}\left[\exp(\boldsymbol{\theta}^T\mathbf{x}_{\mathbf{i},})\right] X. \tag{37}$$

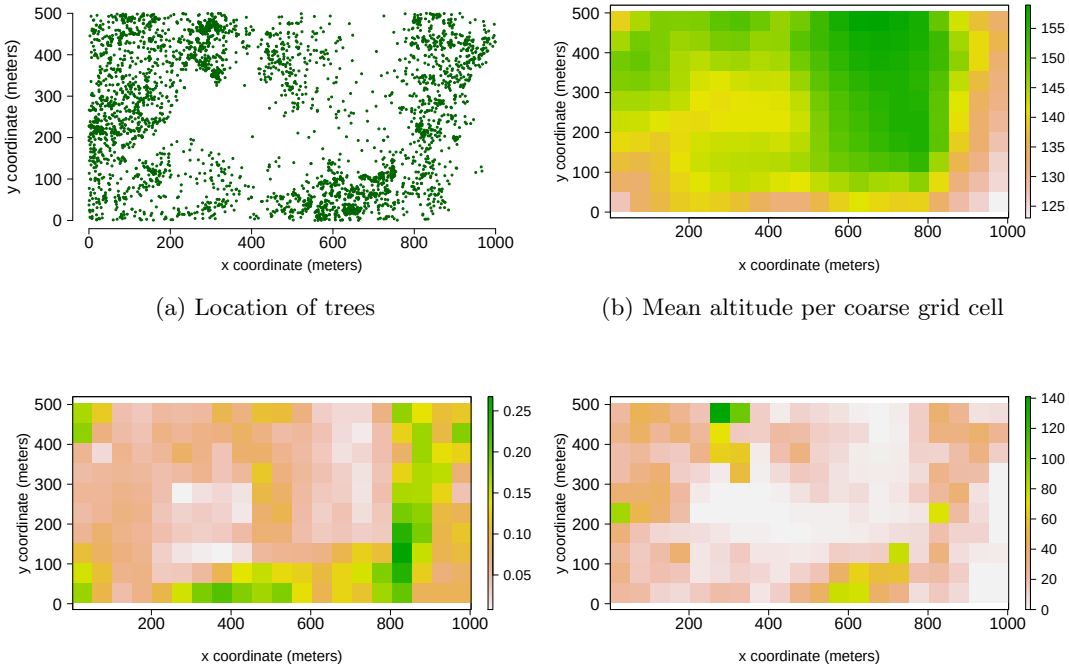
Assuming a normal prior $\boldsymbol{\theta} \sim \pi(\boldsymbol{\theta}) = \mathcal{N}(\mathbf{0}, vI)$ with hyperparameter $v > 0$, the unnormalized log-target of Bayesian Poisson regression is given by (30) with the log-likelihood defined by (35). Furthermore, the gradient of the log-target equals the sum of (36) and (32), while the associated metric expresses as the negated sum $G(\boldsymbol{\theta}|X) = -H(L(\mathbf{y}, X|\boldsymbol{\theta})) + I/v$ of the log-likelihood Hessian (37) and of the log-prior Hessian $-I/v$.

MALA, SMMALA, ALSMMALA and AMSMMALA are run on a tropical tree data set using the Bayesian Poisson log-target of this section and the flat normal prior $\mathcal{N}(\mathbf{0}, 100I)$. The employed data are extracted from a big data set collected in the 500 by 1000 meter Barro Colorado Island plot in Panama (Condit et al. (1996)). More specifically, the data used in this example describe the spatial positions in 1995 of 3605 trees of the species Beilschmiedia pendula Lauraceae. Interest is in predicting the number of trees per unit area from the altitude and elevational gradient (slope) of the tree locations.

Before conducting MCMC inference, the altitude and gradient predictors are aggregated in a coarse 50 by 50 meter grid by taking their means over the initial finer 5 by 5 meter grid. Figure 5 visualizes the tree locations and the coarsed data. Consequently, the processed data consist of the response variable $\mathbf{y}$ holding the number of trees in each of the $n_d = 200$ coarse grid cells as well as the altitude and slop per coarse grid cell. The Poisson regression model comprises $n_\theta = 4$ covariates, namely an intercept, the altitude and its square, and the elevational gradient.

As it can be seen from the ESS of Table 3, which is averaged across all ten simulated chains per sampler, and from the linear autocorrelation Figure 6b, which corresponds to a single simulated chain per sampler for the regression coefficient of altitude, two main groups of sampling effectiveness are formed. ALSMMALA and SMMALA acquire samples more effectively than AMSMMALA and MALA, plus it is observed that ALSMMALA having a marginal advantage over SMMALA. Moreover, ALSMMALA has smaller runtime than SMMALA. Overall, ALSMMALA stands out among all four samplers with a speedup of 2.09 overtaking SMMALA. SMMALA exhibits a speedup of 1.03, so it is doing better than MALA in this example. Although both ALSMMALA and SMMALA exploit local geometric information effectively, the former runs faster, hence it is preferred over latter.

Figures 6c-6f display the traceplots of four chains corresponding to the regression coefficient of covariate $\boldsymbol{\theta}_{(2)}$ (altitude), with a single chain simulated from each of MALA, SMMALA, ALSMMALA and AMSMMALA. All four traceplots show that the involved samplers attain sufficient mixing. The overlaid running means of Figure 6a demonstrate that all four samplers converge to the same value for the coefficient corresponding to the altitude covariate.



(a) Location of trees

(b) Mean altitude per coarse grid cell

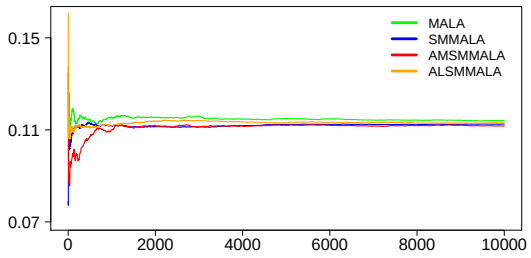
(c) Mean slope per coarse grid cell

(d) Number of trees per coarse grid cell

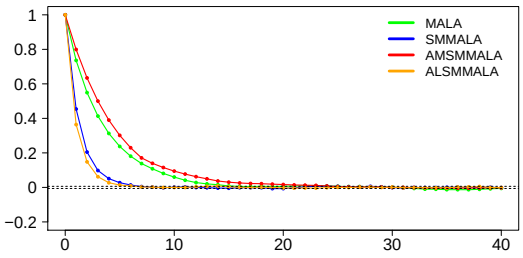
Figure 5: Visualization of tree data of the species Beilschmiedia pendula Lauraceae collected in 1995 from the 500 by 1000 meter Barro Colorado Island plot, Panama. (a): location of 3605 observed trees (original data); (b)-(d): mean altitude, mean elevational gradient and mean number of observed trees per coarse grid cell.

Method	AR	ESS				Time	Efficiency	Speedup
		θ_1	θ_2	θ_3	θ_4			
MALA	.60	12649	14519	12942	32161	2.36	5354.88	1.00
SMMALA	.71	40132	39657	39522	39552	5.48	7212.76	1.35
AMSMMALA	.26	11812	11753	11675	11998	3.12	3740.16	0.70
ALSMMALA	.60	43157	43089	42892	43249	3.83	11194.52	2.09

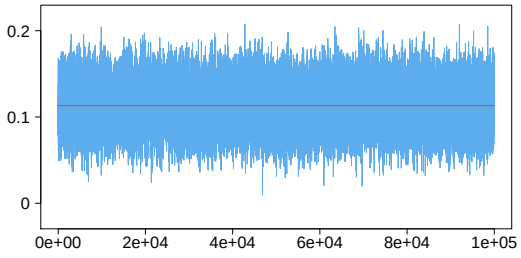
Table 3: Comparison of sampling efficacy between MALA, SMMALA, AMSMMALA and ALSMMALA by applying Bayesian Poisson regression on the tree data of the species Beilschmiedia pendula Lauraceae. AR: acceptance rate; ESS: effective sample size; Time: CPU runtime in seconds; Efficiency: smaller ESS across four covariates divided by runtime; Speedup: ratio of computational efficiencies with a MALA baseline. All the tabulated numbers have been rounded to the second decimal place, apart from the effective sample sizes, which have been rounded to the nearest integer.



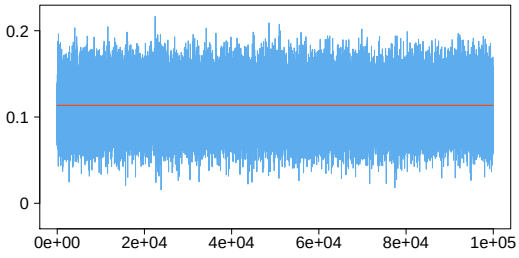
(a) Running mean



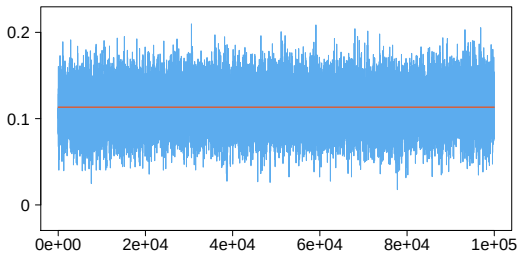
(b) Linear autocorrelation



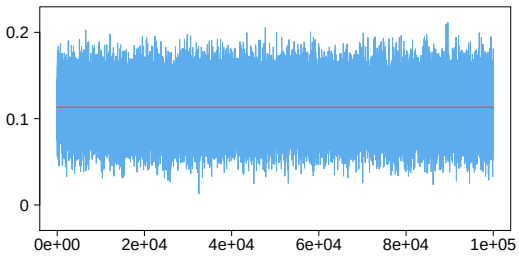
(c) MALA traceplot



(d) SMMALA traceplot



(e) AMSMMALA traceplot



(f) ALSMMALA traceplot

Figure 6: Plots of single chains corresponding to the regression coefficient of the altitude covariate $\boldsymbol{\theta}_{(2)}$ simulated from each of MALA, SMMALA, AMSMMALA and ALSMMALA samplers by applying Bayesian Poisson regression on the tree data of the species Beilschmiedia pendula Lauraceae. (a) Overlaid running means, (b) overlaid linear autocorrelations for the four chains, (c)-(f) individual traceplots per chain (the red horizontal lines represent the Monte Carlo mean).

4.3 Multivariate t-Distribution

Monte Carlo samples are drawn from an n_θ-dimensional Student-t target $t_\nu(\mathbf{0}, \frac{\nu-2}{\nu}\Sigma(c))$ with ν degrees of freedom and covariance matrix

$$\Sigma(c) = \begin{pmatrix} 1 & c^1 & c^2 & \dots & c^{n_\theta-3} & c^{n_\theta-2} & c^{n_\theta-1} \\ c^1 & 1 & c^1 & \dots & c^{n_\theta-4} & c^{n_\theta-3} & c^{n_\theta-2} \\ c^2 & c^1 & 1 & \dots & c^{n_\theta-5} & c^{n_\theta-4} & c^{n_\theta-3} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ c^{n_\theta-3} & c^{n_\theta-4} & c^{n_\theta-5} & \dots & 1 & c^1 & c^2 \\ c^{n_\theta-2} & c^{n_\theta-3} & c^{n_\theta-4} & \dots & c^1 & 1 & c^1 \\ c^{n_\theta-1} & c^{n_\theta-2} & c^{n_\theta-3} & \dots & c^2 & c^1 & 1 \end{pmatrix}, \tag{38}$$

for some constant $0 < c < 1$ that determines the level of correlation between parameter dimensions. The elements of the i-th diagonal of the $n_\theta \times n_\theta$ covariance matrix $\Sigma(c)$ equal $c^{i-1},\ i = 1, 2, \dots, n_\theta$. The scale matrix $\frac{\nu-2}{\nu}\Sigma(c)$ of the t-distribution scales $\Sigma(c)$ by a factor of $(\nu-2)/\nu$ so that the covariance matrix of the t-distribution is $\Sigma(c)$.

In this example, the setting is not Bayesian, so there is no prior distribution involved (Papamarkou et al. (2015)). Instead, MCMC is used for drawing samples from a Student-t distribution $t_\nu(\mathbf{0}, \frac{\nu-2}{\nu}\Sigma(c))$ acting as a random number generator. The simulated chains are randomly initialized away from the zero mode of the t-distribution, and they are expected to converge to zero. In other words, the zero mode of $t_\nu(\mathbf{0}, \frac{\nu-2}{\nu}\Sigma(c))$ is seen as the parameter vector to be estimated.

To emulate some of the conditions of real-life applications, the dimension of the t-target in the MCMC simulations is increased to $n_\theta = 20$ from the $n_\theta = 4$ dimensions of Bayesian logistic and Poisson regression models, relatively high correlation is induced by selecting $c = 0.9$ in (38), and some amount of probability mass is maintained in the distribution tails by choosing $\nu = 30$ degrees of freedom. Furthermore, automatic differentiation is employed instead of deriving the gradient and Hessian of the t-target analytically and the SoftAbs metric is used for correcting the indefinite inverse metric evaluations arising in the course of simulations. Of course the present example does not reach the realm of a fully-fledged application, especially in terms of log-target complexity, but it gives a first indication of other common computational costs appearing in more realistic applications.

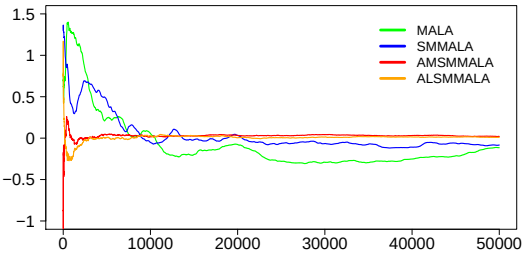
Table 4 reports the minimum, median and maximum ESS of the $n_\theta = 20$ parameter dimensions after averaging across the ten simulated chains per sampler. As seen from Table 4, MALA and SMMALA struggle with the t-distribution yielding very low effective sample size. This is explained by the fact that Langevin Monte Carlo does not produce geometrically ergodic chains for targets with heavier tails. ALSMMALA also does poorly in terms of ESS, albeit somewhat better than its Langevin proposal counterparts. ALSMMALA is impractical for this example, because it reaches its optimal ESS for the simulated t-target after empirical tuning that results in a high acceptance rate of 0.96. Besides, ALSMMALA is somewhat unstable in the sense that if the acceptance rate is not very close to 0.95 then ESS drops dramatically. After all, ALSMMALA is in essence a Langevin Monte Carlo sampler, so it is not anticipated to overcome the intrinsic limitations of its MALA and SMMALA proposal kernels.

On the other hand, AMSMMALA attains much higher ESS than the three Langevin-centric samplers. The AM algorithm of Haario et al. (2001) is not directly applicable to a t-target for two reasons. The original AM sampler is designed to work with targets of bounded support; this shortcoming can be circumvented by a truncated proposal. However, the AM sampler assumes that the probability mass of the target vanishes outside the region of support, so the t-target can not be accommodated directly by the AM algorithm as initially conceived by Haario et al. (2001). Empirical validation via simulations shows that AMSMMALA converges to the mode of the simulated t-target despite the heavier target tails and correlation between parameter dimensions. In other words, AMSMMALA offers a new algorithm with greater range of applicability than its AM proposal kernel. This is explained by the recurring local geometric corrections of the empirical covariance matrix via SMMALA steps. It is worth noticing that Metropolis-Hastings is less effective than Langevin Monte Carlo samplers, such as MALA and SMMALA, in the case of the simulated t-target, so AMSMMALA offers a viable way to simulate from the Student t-distribution of this example, where its contestants fail. In the future, it would be of interest to further compare AMSMMALA with other adaptive Metropolis samplers such as the random adaptive Metropolis (RAM) sampler by Vihola (2012) and the adaptive Metropolis-within-Gibbs (AMWG) algorithm by Roberts and Rosenthal (2009).

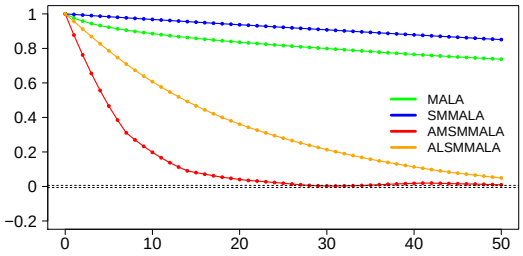
SMMALA has been run using two modes of automatic differentiation, namely reverse AD with source transformation and forward mode. The average SMMALA runtime across ten simulations with forward mode is 575.29 seconds, roughly fourteen times slower than the respective runtime of 41.39 seconds for SMMALA with reverse mode. In spite of using the ForwardDiff Julia package (Revels et al. (2016)), which benefits from tuples that are stack-allocated instead of being heap-allocated, the computational cost for forward mode AD remains prohibitive for this example and for probabilistic problems more generally.

AMSMMALA has an average runtime of 15.13 seconds, and therefore attains a threefold increase in execution time in comparison to SMMALA when both samplers rely on reverse mode. Overall, AMSMMALA achieves a speedup factor of 7.75 thanks to its increased ESS and speed of execution.

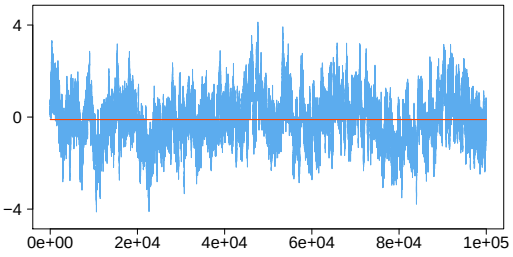
Figure 7 visualizes the output of four chains that correspond to the seventh dimension $\boldsymbol{\theta}_{(7)}$ of the 20-dimensional t-target $t_{30}(\mathbf{0}, \frac{28}{30}\Sigma(0.9))$, with a single chain simulated from each of MALA, SMMALA, ALSMMALA and AMSMMALA. Figure 7b shows the substantial reduction in linear autocorrelation achieved by AMSMMALA, leading to optimal ESS among the four compared samplers. ALSMMALA also reduces the linear autocorrelation, although it is not the sampler of choice for the t-target. As seen from Figure 7a, the running means of AMSMMALA and ALSMMALA converge to the same value, as opposed to the means of MALA and SMMALA. The traceplots of Figures 7c-7f demonstrate the superior mixing of AMSMMALA and the problematic mixing of MALA and SMMALA for the t-target of this example.



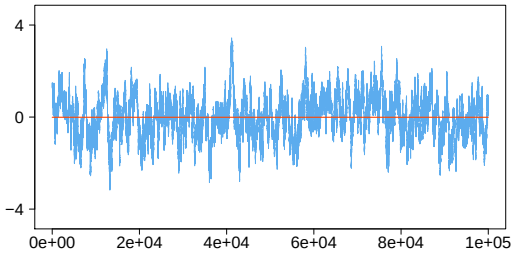
(a) Running mean



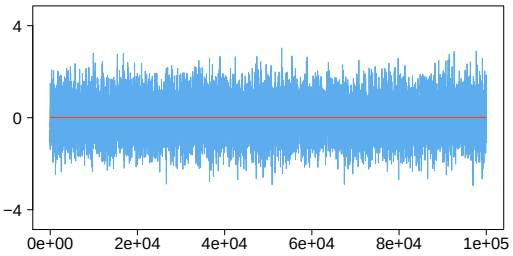
(b) Linear ACF



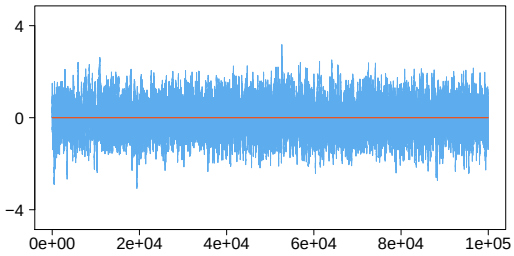
(c) MALA traceplot



(d) SMMALA traceplot



(e) AMSMMALA traceplot



(f) ALSMMALA traceplot

Figure 7: Plots of single chains corresponding to the seventh dimension of the 20-dimensional t-target $t_{30}(\mathbf{0}, \frac{28}{30}\Sigma(0.9))$ simulated from each of MALA, SMMALA, AMSMMALA and ALSMMALA samplers. (a) Overlaid running means, (b) overlaid linear autocorrelations for the four chains, (c)-(f) individual traceplots per chain (the red horizontal line represent the Monte Carlo mean).

Method	AR	ESS			Time	Efficiency	Speedup
		mean	median	max			
MALA	.60	127	145	239	1.95	65.09	1.00
SMMALA (r)	.67	103	114	123	41.39	2.48	0.04
SMMALA (f)	.68	97	104	121	575.29	0.17	0.00
AMSMMALA	.16	7629	7863	7922	15.13	504.11	7.75
ALSMMALA	.96	2465	2538	2574	16.16	152.51	2.34

Table 4: Comparison of sampling efficacy between MALA, SMMALA, AMSMMALA and ALSMMALA by sampling from the 20-dimensional t-target $t_{30}(\mathbf{0}, \frac{28}{30}\Sigma(0.9))$. AR: acceptance rate; ESS: effective sample size; Time: CPU runtime in seconds; Efficiency: smaller ESS across four covariates divided by runtime; Speedup: ratio of computational efficiencies with a MALA baseline. All the tabulated numbers have been rounded to the second decimal place, apart from the effective sample sizes, which have been rounded to the nearest integer.

5. Discussion

MCMC methods have proven critical to the adoption of Bayesian parameter estimation in many scientific and engineering disciplines. In the era of big data, researchers are eager to perform inference on models of increasing realism, often requiring many model parameters or computationally demanding models. Despite increasing computational resources, there will always be a tension between model complexity and computational resources. For many realistic problems, traditional MCMC methods can struggle, even in moderately-high dimensional parameter spaces, due to target densities with complex shapes. Manifold MCMC algorithms hold considerable promise for improving the efficiency of posterior sampling for such problems. However, existing manifold MCMC algorithms can rapidly become prohibitively expensive with increasing dimensionality for problems with costly model evaluations of order $\mathcal{O}(f^3(n_\theta))$ and $\mathcal{O}(f^2(n_\theta))$ for MMALA and SMMALA. Therefore, developing geometric MCMC algorithms that are efficient for such problems has the potential to enable more realistic models, so as improve the accuracy and robustness of scientific research across multiple disciplines.

This paper initiates a conceptually straightforward, yet potentially powerful, approach to the problem of making manifold MCMC algorithms more accessible with increasing number of parameters. The main idea is to combine proposal mechanisms to find a balance between computational complexity and sampling efficacy. TALSMMALA and AMSMMALA have been empirically validated on standard statistical models. Even for these computationally trivial examples, the ALSMMALA and AMSMMALA algorithms demonstrate considerable promise in terms of performance relative to previous algorithms. The increased effective sample size per model evaluation could have an even bigger impact for more complex models which entail more model parameters and more expensive model evaluations.

For example, MCMC methods are essential for the accurate characterization of the masses and orbits of extrasolar planets (Ford (2005)). For many datasets, a well-chosen

random walk Metropolis-Hastings MCMC sampler can be effective Ford (2006). However, designing an efficient sampler requires considerable human effort for each system. Recently, ensemble samplers, such as the differential evolution MCMC (DE-MC) introduced by Braak (2006) and adapted by Nelson et al. (2014a) and the affine-invariant sampler Hou et al. (2012); Foreman-Mackey et al. (2013), have become go-to algorithms for interpreting exoplanet observations, thanks to their balance of computational efficiency and the minimal level human effort needed. When analyzing a planetary system containing one star and $n_{\rm pl}$ planets, specifying the model requires at least $n_\theta \geq 5 \times n_{\rm pl} - 1$ or $n_\theta \geq 7 \times n_{\rm pl} - 1$ model parameters depending on whether the orbital planes are assumed to be coplanar or potentially inclined, plus additional model parameters for modelling the observing process. Each full model evaluation consists of integrating a set of $6 \times n_{\rm pl}$ first-order ordinary differential equations. For some planetary systems, the mutual gravitational interactions can be ignored, so using a simpler approximate model can dramatically reduce the computational cost. However, many of the most scientifically interesting planetary systems and data sets require using the full and more expensive model, including Doppler observations of near-resonant planetary systems and transit timing measurements for near-resonant or tightly packed systems.

Even if the DE-MC algorithm is parallelized over a cluster of CPUs or a GPU, computing usable posterior samples for such systems typically requires weeks of computation (Doyle et al. (2011); Carter1 et al. (2012); Orosz et al. (2012); Nelson et al. (2014b, 2016); Jontof-Hutter et al. (2015, 2016); Mills et al. (2016). Despite these heroic efforts and scientific successes, the resulting effective sample size is often smaller than one would like. Further, dozens of scientifically interesting data sets remain unanalyzed (Holczer et al. (2016)), largely due to the computational cost. For example, the Kepler-57 planetary system has a complex posterior shape (Jontof-Hutter et al. (2016)) that is not well suited to the above ensemble samplers.

Algorithms that exploit geometric information about the posterior shape are likely to be more efficient in terms of the number of model evaluations. manifold MCMC methods could make make it practical to generate posterior samples with increased effective sample sizes. Unfortunately, computing partial derivatives for every proposal, as required for MMALA or SMMALA, would be extremely expensive. The ALSMMALA and AMSMMALA algorithms have the potential to significantly reduce the number of gradient and Hessian evaluations required.

This paper represents one step in the effort to develop more efficient samplers and suggests several future elaborations. Other combinations of geometric proposals, such as manifold Hamiltonian Monte Carlo, with computationally cheaper proposals can be examined in search of efficient hybrid samplers. Seeking an adaptive or geometrically informed schedule for switching between proposal mechanisms could further improve computational efficiency. Another avenue of research would be to apply the samplers of this paper to perform thermodynamic integration upon challenging multi-modal posterior densities.

Acknowledgments

This paper is based on work supported by the National Aeronautics and Space Administration (NASA) under grant NNX15AE21G issued through the Exoplanet Research Program. E.B.F. acknowledges the support of the Eberly College of Science, Center for Astrostatistics, Institute for CyberScience and Center for Exoplanets and Habitable Worlds of Pennsylvania State University, USA. The Center for Exoplanets and Habitable Worlds is supported by the Pennsylvania State University, the Eberly College of Science, and the Pennsylvania Space Grant Consortium. The results reported herein benefited from collaborations and/or information exchange within the Nexus for Exoplanet System Science (NExSS) research coordination network sponsored by the Science Mission Directorate (SMD) of NASA.

References

Michael Betancourt. *A General Metric for Riemannian Manifold Hamiltonian Monte Carlo*, pages 327–334. Springer, Berlin Heidelberg, 2013. ISBN 978-3-642-40020-9.

Cajo J. F. Ter Braak. A markov chain monte carlo version of the genetic algorithm differential evolution: Easy bayesian computing for real parameter spaces. *Statistics and Computing*, 16(3):239–249, 2006. ISSN 1573-1375.

Simon Byrne and Mark Girolami. Geodesic monte carlo on embedded manifolds. *Scandinavian Journal of Statistics*, 40(4):825–845, 2013. ISSN 1467-9469.

Ben Calderhead and Mark Girolami. Statistical analysis of nonlinear dynamical systems using differential geometric sampling methods. *Interface Focus*, 1(821–835), 2011. ISSN 2042-8898.

Ben Calderhead, Michael Epstein, Lucia Sivilotti, and Mark Girolami. *Bayesian Approaches for Mechanistic Ion Channel Modeling*, chapter 13, pages 247–272. Humana Press, Totowa, NJ, 2013.

Joshua A. Carter1, Eric Agol, William J. Chaplin, Sarbani Basu, Timothy R. Bedding, Lars A. Buchhave, Jørgen Christensen-Dalsgaard, Katherine M. Deck, Yvonne Elsworth, Daniel C. Fabrycky, Eric B. Ford, Jonathan J. Fortney, Steven J. Hale, Rasmus Handberg, Saskia Hekker, Matthew J. Holman, Daniel Huber, Christopher Karoff, Steven D. Kawaler, Hans Kjeldsen, Jack J. Lissauer, Eric D. Lopez, Mikkel N. Lund, Mia Lundkvist, Travis S. Metcalfe, Andrea Miglio, Leslie A. Rogers, Dennis Stello, William J. Borucki, Steve Bryson, Jessie L. Christiansen, William D. Cochran, John C. Geary, Ronald L. Gilliland, Michael R. Haas, Jennifer Hall, Andrew W. Howard, Jon M. Jenkins, Todd Klaus, David G. Koch, David W. Latham, Phillip J. MacQueen, Dimitar Sasselov, Jason H. Steffen, Joseph D. Twicken, and Joshua N. Winn. Kepler-36: A pair of planets with neighboring orbits and dissimilar densities. *Science*, 337(6094):556–559, 2012.

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

Richard Condit, Stephen P. Hubbell, and Robin B. Foster. Changes in tree species abundance in a neotropical forest: Impact of climate change. *Journal of Tropical Ecology*, 12: 231–256, 3 1996. ISSN 1469-7831.

Umut Şimşekli, Roland Badeau, A. Taylan Cemgil, and Gaël Richard. Stochastic quasi-newton langevin monte carlo. In *International Conference on Machine Learning (ICML)*, New York, NY, United States, June 2016.

Sandy Davie and Andrew J. Stothers. Improved bound for complexity of matrix multiplication. *Proceedings of the Royal Society of Edinburgh, Section: A Mathematics*, 143: 351–369, 4 2013. ISSN 1473-7124.

Laurance R. Doyle, Joshua A. Carter, Daniel C. Fabrycky, Robert W. Slawson, Steve B. Howell, Joshua N. Winn, Jerome A. Orosz, Andrej Prša, William F. Welsh, Samuel N.

Quinn, David Latham, Guillermo Torres, Lars A. Buchhave, Geoffrey W. Marcy, Jonathan J. Fortney, Avi Shporer, Eric B. Ford, Jack J. Lissauer, Darin Ragozzine, Michael Rucker, Natalie Batalha, Jon M. Jenkins, William J. Borucki, David Koch, Christopher K. Middour, Jennifer R. Hall, Sean McCauliff, Michael N. Fanelli, Elisa V. Quintana, Matthew J. Holman, Douglas A. Caldwell, Martin Still, Robert P. Stefanik, Warren R. Brown, Gilbert A. Esquerdo, Sumin Tang, Gabor Furesz, John C. Geary, Perry Berlind, Michael L. Calkins, Donald R. Short, Jason H. Steffen, Dimitar Sasselov, Edward W. Dunham, William D. Cochran, Alan Boss, Michael R. Haas, Derek Buzasi, and Debra Fischer. Kepler-16: A transiting circumbinary planet. *Science*, 333(6049): 1602–1606, 2011.

Simon Duane, A.D. Kennedy, Brian J. Pendleton, and Duncan Roweth. Hybrid monte carlo. *Physics Letters B*, 195(2):216–222, 1987. ISSN 0370-2693.

Bernhard Flury and Hans Riedwyl. *Multivariate Statistics*. Chapman and Hall, 1988.

Eric B. Ford. Quantifying the uncertainty in the orbits of extrasolar planets. *The Astronomical Journal*, 129:1706–1717, March 2005.

Eric B. Ford. Improving the efficiency of markov chain monte carlo for analyzing the orbits of extrasolar planets. *The Astrophysical Journal*, 642:505–522, May 2006.

Daniel Foreman-Mackey, David W. Hogg, Dustin Lang, and Jonathan Goodman. emcee: The mcmc hammer. *Publications of the Astronomical Society of the Pacific*, 125(925): 306–312, 2013.

François Le Gall. Powers of tensors and fast matrix multiplication. In *Proceedings of the 39th International Symposium on Symbolic and Algebraic Computation*, ISSAC '14, pages 296–303, New York, NY, USA, 2014. ACM.

Charles J. Geyer. Practical markov chain monte carlo. *Statistical Science*, 7(4):473–483, 11 1992.

Philip E Gill, Gene H Golub, Walter Murray, and Michael A Saunders. Methods for modifying matrix factorizations. *Mathematics of Computation*, 28(126):505–535, 1974.

Mark Girolami and Ben Calderhead. Riemann manifold langevin and hamiltonian monte carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73(2):123–214, 2011. ISSN 1467-9868.

Andreas Griewank. On automatic differentiation and alalgorithm linearization. *Pesquisa Operacional*, 34:621–645, 12 2014. ISSN 0101-7438.

Andreas Griewank and Andrea Walther. *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*. Number 105 in Other Titles in Applied Mathematics. SIAM, Philadelphia, PA, 2nd edition, 2008. ISBN 978–0–898716–59–7.

Heikki Haario, Eero Saksman, and Johanna Tamminen. An adaptive metropolis algorithm. *Bernoulli*, 7(2):223–242, 04 2001.

Bruce Hajek. *Cooling Schedules for Optimal Annealing*, pages 147–150. Springer New York, New York, NY, 1987.

Nicholas J. Higham. Computing a nearest symmetric positive semidefinite matrix. *Linear Algebra and its Applications*, 103:103–118, 1988. ISSN 0024-3795.

Nicholas J. Higham. Computing the nearest correlation matrix - a problem from finance. *IMA Journal of Numerical Analysis*, 22(3):329–343, 2002.

Tomer Holczer, Tsevi Mazeh, Gil Nachmani, Daniel Jontof-Hutter, Eric B. Ford, Daniel Fabrycky, Darin Ragozzine, Mackenzie Kane, and Jason H. Steffen. Transit timing observations from kepler. ix. catalog of the full long-cadence data set. *The Astrophysical Journal Supplement Series*, 225(1):9, 2016.

Fengji Hou, Jonathan Goodman, David W. Hogg, Jonathan Weare, and Christian Schwab. An affine-invariant sampler for exoplanet fitting and discovery in radial velocity data. *The Astrophysical Journal*, 745:198, February 2012.

Daniel Jontof-Hutter, Jason F. Rowe, Jack J. Lissauer, Daniel C. Fabrycky, and Eric B. Ford. The mass of the mars-sized exoplanet kepler-138 b from transit timing. *Nature*, 522:321–323, June 2015.

Daniel Jontof-Hutter, Eric B. Ford, Jason F. Rowe, Jack J. Lissauer, Daniel C. Fabrycky, Christa Van Laerhoven, Eric Agol, Katherine M. Deck, Tomer Holczer, and Tsevi Mazeh. Secure mass measurements from transit timing: 10 kepler exoplanets between 3 and 8 $m_{\oplus}$ with diverse densities and incident fluxes. *The Astrophysical Journal*, 820(1):1–23, 2016.

Shiwei Lan, Jeffrey Streets, and Babak Shahbaba. Wormhole hamiltonian monte carlo. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, pages 1953–1959, 2014.

Shiwei Lan, Vasileios Stathopoulos, Babak Shahbaba, and Mark Girolami. Markov chain monte carlo from lagrangian dynamics. *Journal of Computational and Graphical Statistics*, 24(2):357–378, 2015.

Shiwei Lan, Tan Bui-Thanh, Mike Christie, and Mark Girolami. Emulation of higher-order tensors in manifold monte carlo methods for bayesian inverse problems. *Journal of Computational Physics*, 308:81–101, 2016. ISSN 0021-9991.

Samuel Livingstone and Mark Girolami. Information-geometric markov chain monte carlo methods using diffusions. *Entropy*, 16(6):3074, 2014. ISSN 1099-4300.

Sean M. Mills, Daniel C. Fabrycky, Cezary Migaszewski, Eric B. Ford, Erik Petigura, and Howard Isaacson. A resonant chain of four transiting, sub-neptune planets. *Nature*, 533: 509–512, May 2016.

Radford M. Neal. *Bayesian Learning for Neural Networks*, volume 118. Springer, 1996.

Benjamin Nelson, Eric B. Ford, and Matthew J. Payne. Run dmc: An efficient, parallel code for analyzing radial velocity observations using n-body integrations and differential evolution markov chain monte carlo. *The Astrophysical Journal Supplement Series*, 210 (1-13):11, January 2014a.

Benjamin E. Nelson, Eric B. Ford, Jason T. Wright, Debra A. Fischer, Kaspar von Braun, Andrew W. Howard, Matthew J. Payne, and Saleh Dindar. The 55 cancri planetary system: fully self-consistent n-body constraints and a dynamical analysis. *Monthly Notices of the Royal Astronomical Society*, 2014b.

Benjamin E. Nelson, Paul M. Robertson, Matthew J. Payne, Seth M. Pritchard, Katherine M. Deck, Eric B. Ford, Jason T. Wright, and Howard T. Isaacson. An empirically derived three-dimensional laplace resonance in the gliese 876 planetary system. *Monthly Notices of the Royal Astronomical Society*, 455(3):2484–2499, 2016.

Yaghout Nourani and Bjarne Andresen. A comparison of simulated annealing cooling strategies. *Journal of Physics A: Mathematical and General*, 31(41):8373–8385, 1998.

Jerome A. Orosz, William F. Welsh, Joshua A. Carter, Daniel C. Fabrycky, William D. Cochran, Michael Endl, Eric B. Ford, Nader Haghighipour, Phillip J. MacQueen, Tsevi Mazeh, Roberto Sanchis-Ojeda, Donald R. Short, Guillermo Torre, Eric Agol, Lars A. Buchhave, Laurance R. Doyle, Howard Isaacson, Jack J. Lissauer, Geoffrey W. Marcy, Avi Shporer, Gur Windmiller, Thomas Barclay, Alan P. Boss, Bruce D. Clarke, Jonathan Fortney, John C. Geary, Matthew J. Holman, Daniel Huber, Jon M. Jenkins, Karen Kinemuchi, Ethan Kruse, Darin Ragozzine, Dimitar Sasselov, Martin Still, Peter Tenenbaum, Kamal Uddin, Joshua N. Winn, David G. Koch, and William J. Borucki. Kepler-47: A transiting circumbinary multiplanet system. *Science*, 337(6101):1511–1514, 2012.

Theodore Papamarkou, Antonietta Mira, and Mark Girolami. *Monte Carlo Methods and Zero Variance Principle*, chapter 22, pages 457–476. Chapman and Hall/CRC, 2015.

Marcelo Pereyra. Proximal markov chain monte carlo algorithms. *Statistics and Computing*, 26(4):745–760, 2016. ISSN 1573-1375.

Jarrett Revels, Miles Lubin, and Theodore Papamarkou. Forward-mode automatic differentiation in julia. *arXiv*, 2016.

Gareth O. Roberts and Jeffrey S. Rosenthal. Optimal scaling of discrete approximations to langevin diffusions. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 60(1):255–268, 1998. ISSN 1467-9868.

Gareth O. Roberts and Jeffrey S. Rosenthal. Examples of adaptive mcmc. *Journal of Computational and Graphical Statistics*, 18(2):349–367, 2009. URL `http://dx.doi.org/10.1198/jcgs.2009.06134`.

Gareth O. Roberts and Osnat Stramer. Langevin diffusions and metropolis-hastings algorithms. *Methodology And Computing In Applied Probability*, 4(4):337–357, 2002. ISSN 1573-7713.

Gareth O. Roberts and Richard L. Tweedie. Exponential convergence of langevin distributions and their discrete approximations. *Bernoulli*, 2(4):341–363, 12 1996.

Gareth O. Roberts, Andrew Gelman, and Walter R. Gilks. Weak convergence and optimal scaling of random walk metropolis algorithms. *Annals of Applied Probability*, 7(1):110–120, 02 1997.

Raphaela Schwentner, Theodore Papamarkou, Maximilian O. Kauer, Vassilios Stathopoulos, Fan Yang, Sven Bilke, Paul S. Meltzer, Mark Girolami, and Heinrich Kovar. Ews-fli1 employs an e2f switch to drive target gene expression. *Nucleic Acids Research*, 2015.

Matthias Seeger. Low rank updates for the cholesky decomposition. Technical report, 2004.

Matti Vihola. Robust adaptive metropolis algorithm with coerced acceptance rate. *Statistics and Computing*, 22(5):997–1008, 2012. ISSN 1573-1375.

James H. Wilkinson. Modern error analysis. *SIAM Review*, 13(4):548–568, 1971.

Virginia Vassilevska Williams. Breaking the coppersmith-winograd barrier, 2011.