Gaussian approximations of SDEs in Metropolis-adjusted Langevin algorithms.

Simo Särkkä, Christos Merkatas, and Toni Karvonen

This article has been published in *Proceedings of 31st IEEE International Workshop on Machine Learning for Signal Processing*, held in Gold Coast, Australia, on October 25–28, 2021. When citing this article, use the following reference:

> S. Särkkä, C. Merkatas, and T. Karvonen, "Gaussian approximations of SDEs in Metropolis-adjusted Langevin algorithms," in *31st IEEE International Workshop on Machine Learning for Signal Processing*, 2021.

GAUSSIAN APPROXIMATIONS OF SDES IN METROPOLIS-ADJUSTED LANGEVIN ALGORITHMS

Simo Särkkä, Christos Merkatas[*]

Department of Electrical Engineering and Automation
Aalto University, Finland
{simo.sarkka, christos.merkatas}@aalto.fi

Toni Karvonen[†]

Alan Turing Institute
London, UK
toni.s.karvonen@gmail.com

ABSTRACT

Markov chain Monte Carlo (MCMC) methods are a cornerstone of Bayesian inference and stochastic simulation. The Metropolis-adjusted Langevin algorithm (MALA) is an MCMC method that relies on the simulation of a stochastic differential equation (SDE) whose stationary distribution is the desired target density using the Euler–Maruyama algorithm and accounts for simulation errors using a Metropolis step. In this paper we propose a modification of the MALA which uses Gaussian assumed density approximations for the integration of the SDE. The effectiveness of the algorithm is illustrated on simulated and real data sets.

Index Terms— Metropolis–Hastings, Langevin dynamics, stochastic differential equation, Gaussian approximation

1. INTRODUCTION

Markov chain Monte Carlo (MCMC) methods [1] are important computational methods which are widely used in Bayesian data analysis [2] as well as in statistical physics and various other fields. In Bayesian data analysis the central problem is to sample from a posterior distribution

$$p(\theta \mid y) = \frac{p(y \mid \theta)\, p(\theta)}{\int p(y \mid \theta)\, p(\theta)\, \mathrm{d}\theta}, \tag{1}$$

where $\theta \in \mathbb{R}^D$ is a vector of unknown parameters with prior distribution $p(\theta)$ and $y \in \mathbb{R}^n$ is a set of measurements with likelihood $p(y \mid \theta)$. In most real-world applications, the integral in the denominator of (1) is intractable however, the unnormalized posterior distribution

$$\pi(\theta) = p(y \mid \theta)\, p(\theta) \tag{2}$$

can be easily evaluated.

An example of a model in this form is a Bayesian neural network (BNN) model [3], which we also consider as an example in this paper. In hierarchical fashion, a simple BNN model can be written as

$$\begin{aligned} &y_t \mid x_t, w, \lambda \sim \mathrm{N}(y_t \mid g(x_t; w), \lambda^{-1}), \quad 1 \le t \le n \\ &w \sim p(w \mid \zeta),\ \zeta \sim p(\zeta),\ \lambda \sim p(\lambda), \end{aligned} \tag{3}$$

where y_t are the measurements, λ is the precision of the normal distribution, and x_t are the covariates. The function $g(x; w)$ is a (deep) neural network parametrized by w, the collection of weights and biases of the network with hierarchical prior defined by the hyperparameters ζ. In this case the collection of parameters is $\theta = (w, \zeta, \lambda)$ and the set of measurements is $y = (y_1, \ldots, y_n)$.

Posterior inference relies on the computation of the full posterior distribution

$$\begin{aligned} &p(w, \lambda, \zeta \mid \{(x_t, y_t)\}_{t=1}^n) \\ &\quad = \frac{1}{Z}\, p(\lambda)\, p(\zeta)\, p(w \mid \zeta) \prod_{t=1}^n \mathrm{N}(y_t \mid g(x_t; w), \lambda^{-1}), \end{aligned} \tag{4}$$

which requires the computation of

$$\begin{aligned} Z = \int \prod_{t=1}^n \mathrm{N}(y_t \mid g(x_t; w), \lambda^{-1}) \\ \times\, p(\lambda)\, p(\zeta)\, p(w \mid \zeta)\, \mathrm{d}w\, \mathrm{d}\zeta\, \mathrm{d}\lambda \end{aligned} \tag{5}$$

that is clearly intractable.

MCMC methods aim to sample from the posterior distribution – which from now on will be called the target distribution. Posterior distributions arising from neural network models have complicated geometric structure due to the nonlinearities induced by $g(x; w)$, making the use of simple random walk Metropolis–Hastings (MH) algorithms [1, 4, 5] inefficient, due to their slow exploration of the parameter space. To overcome this difficulty, MCMC methods that use gradient information have become popular after the use of the hybrid or Hamiltonian Monte Carlo [6–8] (HMC) for training Bayesian neural networks [3].

[*]Thanks to Academy of Finland for funding.

[†]The author was supported by the Lloyd's Register Foundation programme on data-centric engineering.

978-1-7281-6338-3/21/$31.00 ©2021 IEEE

Another class of gradient-based MCMC algorithms which can be seen as a special case of the HMC, is the Metropolis-adjusted Langevin algorithm (MALA) [8–10]. Roughly speaking, in MALA, candidate samples are proposed by solving a stochastic differential equation (SDE) of the Langevin type using the Euler–Maruyama integrator [11, 12]. Due to SDE approximation errors a MH step is included to the algorithm which determines if the proposed sample is accepted or not. However, the problem with the Euler–Maruyama approximation as proposal distribution is that it only works for very small time steps Δt.

The contribution of this paper is to propose an MCMC algorithm based on Gaussian assumed density approximation [12] of the Langevin diffusion which is a commonly used approximation used in state-estimation context [13, 14]. This provides greater flexibility on the choice of integration interval. Brief theoretical analysis of the algorithm is also provided and its practical performance is empirically illustrated and compared to competing approaches.

The paper is organized as follows. Section 2 introduces the necessary background on MH and MALA samplers. The Gaussian assumed density approximation for the solution of an SDE and the proposed Gaussian Metropolis-adjusted algorithm (GMALA) are presented in Section 3. Theoretical analysis of the algorithm is presented in Section 4. In Section 5 the effectiveness of the algorithm is illustrated on simulated and real data sets. Finally, Section 6 concludes the paper.

2. BACKGROUND

In this section we provide the necessary background on the MH and MALA methods for sampling from predetermined target densities. A more detailed exposition of MCMC methods can be found in [1].

2.1. Metropolis–Hastings algorithm

A general MCMC framework for sampling from a given target density is the Metropolis–Hastings algorithm [4, 5] where candidate samples are drawn from a proposal distribution with density $q(\theta' \mid \theta)$ and the following steps are performed until the required number of samples S is acquired. The chain starts from $\theta \sim \pi_0(\theta)$ and a new sample θ' is proposed according to the proposal distribution $q(\theta' \mid \theta)$. The new sample is accepted if a uniform random variable $u \sim \mathrm{U}(0, 1)$ is less than $\alpha(\theta' \mid \theta)$, where

$$\alpha(\theta' \mid \theta) = \min\left[1, \frac{\pi(\theta')\, q(\theta \mid \theta')}{\pi(\theta)\, q(\theta' \mid \theta)}\right] \tag{6}$$

is the acceptance probability. If the sample is rejected, the chain remains at θ.

Samples $\theta^{(i)}$, $1 \leq i \leq S$, obtained from the MH define a Markov chain whose stationary distribution is defined by the target distribution $p(\theta \mid y) \propto \pi(\theta)$. Asymptotically, the $\theta^{(i)}$ samples are draws from the target distribution [1].

2.2. Metropolis-adjusted Langevin algorithm

The key to the MH algorithm performance is the choice of the proposal distribution q. Metropolis-adjusted Langevin algorithm (MALA) [8–10] is a more sophisticated approach than the classical random walk MH as it uses gradient information in the proposal density. Consider an SDE

$$\mathrm{d}\theta(t) = f(\theta(t))\,\mathrm{d}t + \mathrm{d}W(t), \tag{7}$$

where $f(\cdot)$ is a given (locally Lipschitz) drift function and W is a Brownian motion with diffusion matrix Q_c. Provided that the drift function is suitably stable, then as $t \to \infty$ the distribution of $\theta(t)$ converges to a stationary distribution with a density $p_\infty(\theta)$.

A particularly useful case is obtained with $Q_c = I$ and $f(\theta) = \frac{1}{2} \nabla \log \pi(\theta)$. In this case the Langevin diffusion converges at an exponential rate [10] to the stationary distribution which is given by $p_\infty(\theta) \propto \pi(\theta)$, namely, the density of the target distribution. The MALA method is based on the observation that samples from the target distribution can be obtained by simulating (long enough) trajectories from the SDE

$$\mathrm{d}\theta(t) = \frac{1}{2} \nabla \log \pi(\theta(t))\,\mathrm{d}t + \mathrm{d}W(t). \tag{8}$$

The choice of SDE with the given stationary distribution is not unique and it is often useful to introduce a symmetric positive definite preconditioner matrix M such that the SDE becomes

$$\mathrm{d}\theta(t) = \frac{1}{2}\, M \nabla \log \pi(\theta(t))\ \mathrm{d}t + \sqrt{M}\,\mathrm{d}W(t), \tag{9}$$

where $\sqrt{M}$ is a matrix such that $M = \sqrt{M}\,\sqrt{M}^{\mathsf{T}}$.

Unfortunately exact simulation from an SDE is usually intractable [11, 12] and numerical simulation methods need to be employed. The idea in MALA [8–10] is to use the Euler–Maruyama method [11, 12] to simulate from the SDE which amounts to replacing the SDE with the stochastic difference equation

$$\theta(t + \Delta t) = \theta(t) + \frac{\Delta t}{2}\, M \nabla \log \pi(\theta(t)) + \sqrt{M}\,\epsilon, \tag{10}$$

where $\epsilon \sim \mathrm{N}(0, \Delta t)$. This approximation is then used as the proposal distribution in the Metropolis–Hastings algorithm:

$$q(\theta' \mid \theta) = \mathrm{N}\left(\theta' \mid \theta + \frac{\Delta t}{2}\, M \nabla \log \pi(\theta), M\,\Delta t\right). \tag{11}$$

To implement the algorithm we also need to be able to evaluate the backward transition probability density which in this case is given as

$$q(\theta \mid \theta') = \mathrm{N}\left(\theta \mid \theta' + \frac{\Delta t}{2}\, M \nabla \log \pi(\theta'), M\,\Delta t\right). \tag{12}$$

By plugging these into the Metropolis–Hastings algorithm in the previous section we now get the algorithm. The shortcoming of this algorithm is that the Euler–Maruyama method provides a reasonable approximation for the SDE solution only for small values of Δt which allows the proposal distribution to only take small steps. This reduces the behavior of MALA to be close to random-walk MH algorithm.

3. GAUSSIAN APPROXIMATIONS OF SDES IN MALA

In this section, we now show that the solution of the Langevin diffusion in MALA can be approximated using a Gaussian assumed density approximation.

3.1. Gaussian assumed density approximations of SDEs

One kind of approximations for SDEs are Gaussian assumed density approximations that are commonly used in stochastic filtering context [12–14]. In these approximations the probability density function of the SDE

$$\mathrm{d}\theta(t) = f(\theta(t))\,\mathrm{d}t + \mathrm{d}W(t) \tag{13}$$

is approximated with a Gaussian distribution. The mean and covariance of the Gaussian distribution are from the ordinary differential equations (ODEs)

$$\begin{aligned}\frac{\mathrm{d}m(t)}{\mathrm{d}t} &= \mathrm{E}_t[f(\theta)],\\ \frac{\mathrm{d}P(t)}{\mathrm{d}t} &= \mathrm{E}_t[f(\theta)\,(\theta - m(t))^{\mathsf{T}}]\\ &\quad + \mathrm{E}_t[(\theta - m(t))\,f^{\mathsf{T}}(\theta)] + Q_c,\end{aligned} \tag{14}$$

where Q_c is the diffusion matrix of the Brownian motion W, and $\mathrm{E}_t[\cdot]$ denotes an expectation over a Gaussian distribution with mean $m(t)$ and covariance $P(t)$:

$$\mathrm{E}_t[g(\theta)] = \int g(\theta)\ \mathrm{N}(\theta \mid m(t), P(t))\,\mathrm{d}\theta. \tag{15}$$

Please note that θ is time-dependent also in (14) although we have dropped the dependence for notation convenience. Due to (15), the equation for the covariance can be equivalently written as

$$\frac{\mathrm{d}P(t)}{\mathrm{d}t} = \mathrm{E}_t[F_\theta(\theta)]\,P(t) + P(t)\ \mathrm{E}_t[F_\theta(\theta)^{\mathsf{T}}] + Q_c, \tag{16}$$

where $F_\theta(\cdot)$ is the Jacobian matrix of $\theta \mapsto f(\theta)$. This follows from Stein's lemma [12, Theorem 9.2].

3.2. Gaussian Metropolis-adjusted Langevin algorithm

In the case $f(\theta) = \frac{1}{2}\,M\,\nabla \log \pi(\theta)$ and $Q_c = M$ the mean and covariance equations become

$$\frac{\mathrm{d}m(t)}{\mathrm{d}t} = \frac{1}{2} M\ \mathrm{E}_t[\nabla \log \pi(\theta)], \tag{17}$$

$$\begin{aligned}\frac{\mathrm{d}P(t)}{\mathrm{d}t} &= \frac{1}{2} M\ \mathrm{E}_t[\nabla \log \pi(\theta)\,(\theta - m(t))^{\mathsf{T}}]\\ &\quad + \frac{1}{2}\,\mathrm{E}_t[(\theta - m(t))\,\nabla^{\mathsf{T}} \log \pi(\theta)]\,M + M\\ &= \frac{1}{2} M\ \mathrm{E}_t[\nabla\nabla^{\mathsf{T}} \log \pi(\theta)]\,P(t)\\ &\quad + \frac{1}{2} P(t)\ \mathrm{E}_t[\nabla\nabla^{\mathsf{T}} \log \pi(\theta)]\,M + M.\end{aligned} \tag{18}$$

We can now form the proposal density $q(\theta' \mid \theta)$ by integrating these ODEs for time span Δt starting from $m(0) = \theta$, $P(0) = 0$, and by using the corresponding Gaussian transition density as the proposal. This corresponds to

$$q(\theta' \mid \theta) = \mathrm{N}(\theta' \mid m(\Delta t \mid \theta), P(\Delta t \mid \theta)), \tag{19}$$

where we have explicitly denoted the starting point in the mean and covariance. To implement a MH for this, we need to evaluate $q(\theta \mid \theta')$ as well. This corresponds to integration of the equations starting from θ' over the period Δt providing

$$q(\theta \mid \theta') = \mathrm{N}(\theta \mid m(\Delta t \mid \theta'), P(\Delta t \mid \theta')). \tag{20}$$

Thus, a single iteration of the Gaussian Metropolis-adjusted Langevin algorithm (GMALA) includes the steps of integrating the ODEs given in (17)–(18) as shown in Algorithm. 1. We discuss computational methods for the mean and covariance ODEs in the next section.

3.3. Numerical methods for mean and covariance ODEs

Unfortunately, the mean and covariance differential equations (17) and (18) cannot be solved exactly due to the intractable Gaussian integrals and due to the fact that they are non-linear vector and matrix valued ODEs. However, we can approximate them easily (see, e.g., [12]).

Simple Taylor-series type of approximation can be obtained by putting $\mathrm{E}_t[f(\theta)] \approx f(m(t))$ and $\mathrm{E}_t[F_\theta(\theta)] \approx F_\theta(m(t))$, which by plugging in the definition of f becomes

$$\frac{\mathrm{d}m(t)}{\mathrm{d}t} = \frac{1}{2} M\,\nabla \log \pi(m(t)), \tag{21}$$

$$\begin{aligned}\frac{\mathrm{d}P(t)}{\mathrm{d}t} &= \frac{1}{2} M\,\nabla\nabla^{\mathsf{T}} \log \pi(m(t))\,P(t)\\ &\quad + \frac{1}{2} P(t)\,\nabla\nabla^{\mathsf{T}} \log \pi(m(t))\,M + M.\end{aligned} \tag{22}$$

Another possibility is to use Gaussian cubature or sigma-point approximations [14] such that $\theta_j(t) = m(t) + \sqrt{P(t)}\xi_j$ and

$$\mathrm{E}_t[f(\theta)] \approx \sum_j \omega_j\, f(m(t) + \sqrt{P(t)}\,\xi_j) \tag{23}$$

Algorithm 1 Gaussian Metropolis-adjusted Langevin algorithm

procedure GMALA(θ, Δt)
  Input: previous state, integration interval
  Output: new state θ
  Integrate ODEs in (17)-(18) with initial conditions $m(0) = \theta, P(0) = 0$ for time Δt.
  Sample $\theta' \sim \mathrm{N}(\theta' \mid m(\Delta t \mid \theta), P(\Delta t \mid \theta))$.
  Integrate ODEs in (17)-(18) with initial conditions $m(0) = \theta', P(0) = 0$ for time Δt.
  ▷ Backward solve
  Compute acceptance probability

$$\alpha(\theta' \mid \theta) = \min\left[1, \frac{\pi(\theta')\, q\,(\theta \mid \theta')}{\pi(\theta)\, q\,(\theta' \mid \theta)}\right].$$

  ▷ Evaluate q using (19)-(20)
  Sample $u \sim \mathrm{U}(0,1)$ ▷ Uniform random variable
  if $u \leq \alpha(\theta' \mid \theta)$ **then**
    $\theta \leftarrow \theta'$ ▷ accept new state
  else
    $\theta \leftarrow \theta$ ▷ reject new state
  return θ

Then

$$\begin{aligned}
&\mathrm{E}_t[f(\theta)\,(\theta - m(t))^{\mathsf{T}}] \\
&\quad \approx \sum_j \omega_j\, f(m(t) + \sqrt{P(t)}\,\xi_j)\,\xi_j^{\mathsf{T}}\,\sqrt{P(t)}^{\mathsf{T}}, \\
&\mathrm{E}_t[F_\theta(\theta)] = \mathrm{E}_t[f(\theta)\,(\theta - m(t))^{\mathsf{T}}]P^{-1} \\
&\quad \approx \sum_j \omega_j\, f(m(t) + \sqrt{P(t)}\,\xi_j)\,\xi_j^{\mathsf{T}}\,\sqrt{P(t)}^{\mathsf{T}}\,P^{-1} \qquad (24) \\
&\quad = \sum_j \omega_j\, f(m(t) + \sqrt{P(t)}\,\xi_j)\,\xi_j^{\mathsf{T}}\,\sqrt{P(t)}^{-1}, \qquad (25)
\end{aligned}$$

which can be plugged into (14) or (16) to obtain approximations to the mean and covariance functions. In practice these equations are badly behaving when covariance is close to singular and hence it is advisable to replace the zero initial condition with $P(0) = \lambda\, I$, where $\lambda > 0$. Note that this does not change the validity of the MCMC algorithm, provided that we use the same initial condition in both directions.

We can now approximate the solution of the mean equation by taking, for example, an Euler integration step of length Δt. For the covariance equation we need to assure that the covariance remains positive definite. This can be assured by using a discretization, where we fix $F = \mathrm{E}_t[F_\theta(\theta)]$ which reduces (16) to the following:

$$\frac{\mathrm{d}P(t)}{\mathrm{d}t} = F\,P(t) + P(t)\,F^{\mathsf{T}} + Q_c, \qquad (26)$$

where in this case we actually have $F = F^{\mathsf{T}}$. A positive-definiteness preserving discretization is now obtained by

$$P(t + \Delta t) = A(\Delta t)\,P(t)\,A^{\mathsf{T}}(\Delta t) + Q(\Delta t), \qquad (27)$$

where

$$\begin{aligned}
A(\Delta t) &= \exp(\Delta t\, F), \\
Q(\Delta t) &= \int_0^{\Delta t} \exp((\Delta t - \tau)\,F)\,Q_c\,\exp((\Delta t - \tau)\,F)^{\mathsf{T}}\,\mathrm{d}\tau.
\end{aligned}$$

In practice, we can now form a solution for a large time step by taking K steps of length Δt, which leads to a integration time interval of length $K\,\Delta t$. This differs from the Euler–Maruyama approximation used in MALA where we are only allowed to take a single step of length Δt.

4. THEORETICAL ANALYSIS

In this section, the aim is to investigate some theoretical properties of the method.

4.1. Validity of the algorithm

In order to have a valid MCMC algorithm, the proposal densities given in (19) and (20) must be well defined. This means that the covariance matrix of the Gaussian must be positive definite with bounded covariance. As it has already been mentioned, positive definiteness is preserved due to the discretization given in (27). The boundedness of the covariance is verified in the next section.

4.2. Boundedness of the covariance

Let $\lambda_{\max}(A)$ denote the largest eigenvalue of a matrix A and define the constant

$$L_\pi = \sup_\theta \lambda_{\max}\big[\nabla\nabla^{\mathsf{T}} \log \pi(\theta)\big].$$

Proposition 1. *If $L_\pi < \infty$ and $M = aI$ for $a > 0$, then the covariance in* (18) *satisfies, for every $t \geq 0$,*

$$\operatorname{tr} P(t) \leq \exp(aL_\pi t)\operatorname{tr} P(0) + \frac{(1 - \exp(L_\pi t))aD}{aL_\pi}. \qquad (28)$$

In particular, there is a time-uniform bound if $L_\pi < 0$.

Proof. By the trace inequality

$$\operatorname{tr}\left(\mathrm{E}_t[\nabla\nabla^{\mathsf{T}} \log \pi(\theta)]\,P(t)\right) \leq L_\pi \operatorname{tr} P(t),$$

which follows from [15, Prop. 8.4.13], we have

$$\begin{aligned}
\frac{\mathrm{d}\operatorname{tr} P(t)}{\mathrm{d}t} &= \frac{1}{2}a \operatorname{tr}\left(\mathrm{E}_t[\nabla\nabla^{\mathsf{T}} \log \pi(\theta)]\,P(t)\right) \\
&\quad + \frac{1}{2}a \operatorname{tr}\left(P(t)\,\mathrm{E}_t[\nabla\nabla^{\mathsf{T}} \log \pi(\theta)]\right) + a \operatorname{tr} I \\
&\leq a\,L_\pi \operatorname{tr} P(t) + aD.
\end{aligned}$$

The bound in (28) then follows from Grönwall's inequality which states that $u'(t) \leq \alpha u(t) + \beta$ implies that

$$u(t) \leq u(0)\exp(\alpha t) - \frac{(1-\exp(\alpha t))\beta}{\alpha}.$$

□

The above proposition can be generalized to arbitrary M as well as to some of the numerical methods reviewed in Section 3.3 by following [16, Sect. II].

5. NUMERICAL EXPERIMENTS

In this section we demonstrate the performance of the proposed methodology on simulated and real data sets. For comparison, we implemented the standard version of MALA [9, 10] and Hamiltonian Monte Carlo (HMC) [6, 7] algorithms.

5.1. Sampling from Rosenbrock banana function

The Rosenbrock banana density function is defined on $\mathbb{R}^D$ as

$$\pi(\theta) \propto \exp\left(-\frac{\theta_1^2}{200} - \frac{1}{2}\sum_{d=3}^{D}\theta_d^2 - \frac{1}{2}(\theta_2 + B\theta_1^2 - 100B)^2\right).$$

For the simulation, we set $B = 0.1$ and $D = 10$ and draw samples from the target density using GMALA, with step size $\Delta t = 0.2$ and number of steps $K = 50$. We compare the method with standard MALA with step size 0.2 and a standard HMC algorithm using leapfrog integration with step size 0.2 and 50 integration steps. We sample 10 independent chains consisting of $5{,}000$ samples obtained after a burn-in period of 500 samples. In Fig. 1 the sampled values for the

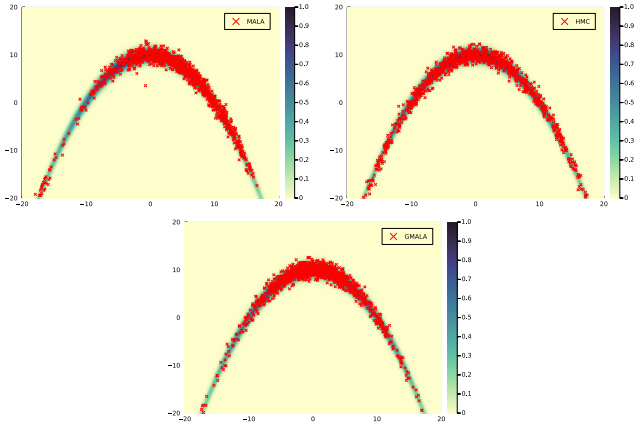

Fig. 1. Samples drawn via MALA (top left), HMC (top right), and GMALA (bottom) on the first two dimensions of a 10-dimensional Rosenbrock banana distribution.

first two dimensions are superimposed over the true density.

Table 1 shows the effective sample size (ESS) [2] of the Markov chains obtained from the three methods for the first

Table 1. Effective sample size (ESS) of the Markov chains for the first two dimensions of the banana density. Results are summarized over 10 chains.

θ	MALA	GMALA	HMC
θ_1	112.1	289.4	2558.6
θ_2	111.0	264.0	2152.6

two dimensions (θ_1, θ_2). In Fig. 2 we provide the potential scaling reduction factor (PSRF) [17] summarized over 10 chains. A value close to one suggests that the parameter chain is at equilibrium. Table 1 and Fig. 2 suggest that GMALA mixes better than MALA in the particular example.

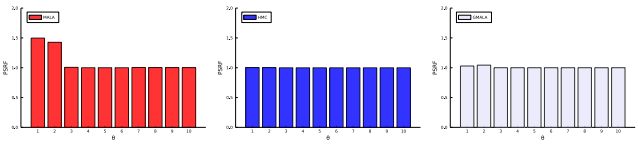

Fig. 2. PSRFs for MALA, HMC and GMALA (left to right).

5.2. Sampling from Bayesian neural network

In this section we consider Bayesian neural network (BNN) based modeling of Canadian Lynx data which consists of the annual trappings of lynx in the Mackenzie River District of North–West Canada for the period from 1821 to 1934 resulting to a total of 114 observations. We aim to identify the process responsible for the generation of the data with an autoregressive BNN of lag 2, consisting of 5 hidden neurons with $\tanh$ activation functions. The first 100 observations were used for training and each trained model predictive capability is tested on the last 14 observations. We note here that similarly to [3] only w, the collection of weights and biases ($D = 21$) is sampled via MALA, GMALA, and HMC with common step size 0.01 and $K = 1, 10, 10$ respectively. For each algorithm, 5 chains of $20{,}000$ were sampled. The rest of the hyperparameters are sampled via Gibbs.

In Fig. 3 we represent the model estimated with GMALA and the out-of-sample predictions for the next 14 values summarized over 5 chains. Fig. 4 shows PSRFs for the summarized chain. Due to neural network weight space symmetry, it is impossible to choose important coordinates of the θ vector. For this reason, we report in Table 2 the minimum ESS summarized over the 5 chains.

Table 2. Minimum effective sample size (minESS) of the Markov chains sampled for the Bayesian neural network. Results are summarized over 5 chains.

	GMALA	HMC	MALA
minESS	212.2	300.0	196.3

We note that GMALA and HMC algorithms produce sim-

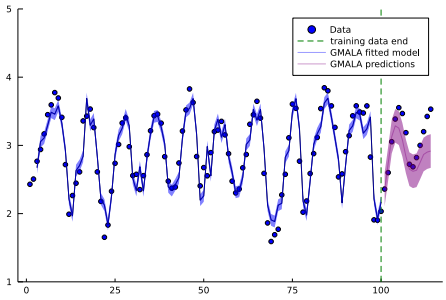

Fig. 3. Fitted time series and 14-step ahead predictions for the Canadian Lynx data (blue dots). Solid dark blue (purple) lines represent the mean posterior function (prediction) and the shaded regions represent the associated uncertainties.

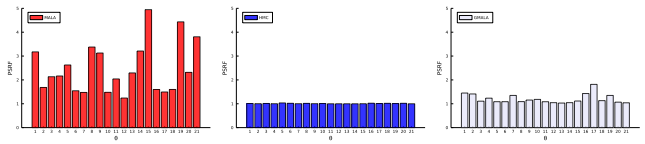

Fig. 4. PSRFs of MALA, HMC, and GMALA (from left to right) for the Bayesian neural network example. Results are summarized over 5 chains.

ilar results in the identification and prediction of future values. On the other hand the MALA-trained Bayesian neural network has acceptance rate below 5% for this step size, leading to biased estimates and predictions with high variance.

6. CONCLUSION

In this article, a variation of the Metropolis-adjusted Langevin algorithm namely, the Gaussian Metropolis-adjusted Langevin algorithm (GMALA), has been proposed. In GMALA, the traditional Euler–Maruyama scheme used for simulation of the SDE in MALA has been replaced with a Gaussian assumed density approximation making the algorithm more amenable to larger integration steps. Experiments in simulated and real data sets suggest that GMALA performs at least similarly or better than MALA.

7. REFERENCES

[1] S. Brooks, A. Gelman, G. L. Jones, and X.-L. Meng, *Handbook of Markov Chain Monte Carlo*, Chapman & Hall/CRC, 2011.

[2] A. Gelman, J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin, *Bayesian Data Analysis*, Chapman and Hall/CRC, 3rd edition, 2013.

[3] R. M. Neal, *Bayesian Learning for Neural Networks*, Springer, 1996.

[4] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of state calculations by fast computing machines," *The Journal of Chemical Physics*, vol. 21, no. 6, pp. 1087–1092, 1953.

[5] W. K. Hastings, "Monte Carlo sampling methods using Markov chains and their applications," *Biometrika*, vol. 57, no. 1, pp. 97–109, 1970.

[6] S. Duane, A. D. Kennedy, B. J. Pendleton, and D. Roweth, "Hybrid Monte Carlo," *Physics Letters B*, vol. 195, no. 2, pp. 216–222, 1987.

[7] R. M. Neal et al., "MCMC using Hamiltonian dynamics," in *Handbook of Markov Chain Monte Carlo*, S. Brooks, A. Gelman, G. L. Jones, and X.-L. Meng, Eds., chapter 5. Chapman & Hall/CRC, 2011.

[8] M. Girolami and B. Calderhead, "Riemann manifold Langevin and Hamiltonian Monte Carlo methods," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 73, no. 2, pp. 123–214, 2011.

[9] J. Besag, "Comments on "Representations of knowledge in complex systems"," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 56, no. 56, pp. 591–592, 1994.

[10] G. O. Roberts and R. L. Tweedie, "Exponential convergence of Langevin distributions and their discrete approximations," *Bernoulli*, vol. 2, no. 4, pp. 341–363, 1996.

[11] P. E. Kloeden and E. Platen, *Numerical Solution to Stochastic Differential Equations*, vol. 23 of *Applications of Mathematics*, Springer, New York, NY, 1999.

[12] S. Särkkä and A. Solin, *Applied Stochastic Differential Equations*, Cambridge University Press, 2019.

[13] H. Kushner, "Approximations to optimal nonlinear filters," *IEEE Transactions on Automatic Control*, vol. 12, no. 5, pp. 546–556, 1967.

[14] S. Särkkä and J. Sarmavuori, "Gaussian filtering and smoothing for continuous-discrete dynamic systems," *Signal Processing*, vol. 93, no. 2, pp. 500–510, 2013.

[15] D. S. Bernstein, *Matrix Mathematics: Theory, Facts, and Formulas*, Princeton University Press, 2009.

[16] T. Karvonen, S. Bonnabel, E. Moulines, and S. Särkkä, "Bounds on the error covariance of a class of Kalman–Bucy filters for systems with non-linear dynamics," in *57th IEEE Conference on Decision and Control*, 2018, pp. 7176–7181.

[17] A. Gelman and D. B. Rubin, "Inference from iterative simulation using multiple sequences," *Statistical science*, vol. 7, no. 4, pp. 457–472, 1992.