pSGLD FOR SAMPLING: AN INVESTIGATION

Yunjie Tim Wang
Department of Mathematics
Imperial College London
tw1320@ic.ac.uk

ABSTRACT

Recent developments in Diffusion Probabilistic Models (DPMs) and in particular Score-based Generative Models(SGMs) have shown that improvement in speed and sample quality can be gained by adopting techniques such as preconditioned Langevin Monte Carlo from computational statistics. In this article, we investigate the performance of pSGLD and its variants in sampling tasks including from SGMs. We show that pSGLD can adapt to various noise levels without annealing. A by-product of the experiments demonstrate a tuned version of PC sampling can produce impressive FID with low number of network function evaluations (NFEs). In addition, we introduce 'forgetful' versions of pSGLD with RMSProp.

Keywords MCMC · SGLD · RMSProp · Score-based generative models

1 Introduction

Conventional Markov Chain Monte Carlo (MCMC) methods based on the Metropolis-Hastings algorithm (Metropolis et al., 1953) often fail to scale to high-dimensional distributions and large datasets due to the computational cost at each step, which makes them unsuitable for Score-based Generative Models (SGMs). In recent years, computationally efficient alternatives based on the Unadjusted Langevin Algorithm (ULA) have been proposed (Welling and Teh, 2011), which led to the subsequent development of a class of methods called Stochastic Gradient Langevin Dynamics (SGLD) and their extensions using preconditioners, which we will refer to as pSGLD. (Girolami and Calderhead, 2011; Ahn et al., 2012; Patterson and Teh, 2013; Li et al., 2015; Simsekli et al., 2016; Kim et al., 2020; Yu et al., 2023).

However, since the Langevin diffusion is isotropic, the performance of SGLD suffers when the target distribution has vastly different scales or high correlation across dimensions. To remedy this, Girolami and Calderhead (2011) leverages Langevin diffusion on Riemannian manifolds and demonstrates its efficacy with the Fisher-Rao metric tensor; they have also provided a general form of SGLD on manifolds with any positive-definite metric $G(\boldsymbol{\theta})$. Using similar ideas, Ahn et al. (2012) constructs an adaptive preconditioner that uses a moving average of the sample covariance of noisy gradients. On the other hand, Patterson and Teh (2013) explores the special case of Langevin dynamics on the probability simplex and adopts analytically tractable expected Fisher information matrices as preconditioners, which is not generally applicable in other situations. To avoid this intractability and computational expense, Li et al. (2015) and Kim et al. (2020) considers SGLD incoporated with adaptive gradient optimizers like RMSProp (Tieleman and Hinton, 2012) and Adam (Kingma and Ba, 2017). Although these approaches alleviate the scale issue, they are unable to account for the high correlation across dimensions due to the preconditioners' diagonal nature. To overcome this, Simsekli et al. (2016) proposes SGLD with empirical estimates of the Hessian as a full-matrix preconditioner, while Yu et al. (2023) explores SGLD with Monge metric (Hartmann et al., 2022) and Shampoo metric (Gupta et al., 2018).

Application of ULA/SGLD-type methods to SGMs/DPMs have not received much attention until very recently, even though one of the foundational papers on SGMs by Song and Ermon (2019) have based their sampling process on ULA. However, as Song et al. (2021) have noted, this sampler cannot achieve promising performance on complex data distributions compared to the sampling process based on discretizing Stochastic Differential Equations (SDEs), which suggests that direct applications of SGLD-type methods are not feasible. Nonetheless, convincing empirical results have been demonstrated in various works, based on both the continuous-time framework (*e.g.* SGMs (Song et al., 2021)) or the discrete-time framework (*e.g.* DDIM (Song et al., 2022) and DDPM (Ho et al., 2020)). For continuous-time

frameworks, Ma et al. (2022) computes a fixed preconditioner from data by utilizing frequency and pixel information through Discrete Fourier Transform; they have applied this preconditioner in both SDE discretization and Langevin dynamics. For discrete-time frameworks, Wang et al. (2023) has demonstrated improvements in sample quality using Adam-like updates in DDIM and DDPMs. There are also works that use the idea of a momentum sampler (Wu et al., 2023) or critically-damped Langevin diffusion (Dockhorn et al., 2022). While they also demonstrate promising performance, both methods require (re-)training of the diffusion model on which they are based.

In Section 2, we provide discussion on the background materials of this article. In Section 3 and Section 4, we introduce samplers used in experiments. We present our experiment results in Section 5 and Section 6. In particular, in Section 4.4, we describe a simple modification to the PC sampling in (Song et al., 2021) that can produce samples with FID close to that of samples produced by the preconditioned SGM (Ma et al., 2022) using the same NFEs (see Section 6).

2 Background

We first review Langevin Dynamics and variants of algorithms built on it. Then we revisit diffusion probabilistic models and its various formulations.

2.1 Langevin Dynamics

The Langevin Dynamics is defined by a stochastic differential equation (SDE) of the form:

$$\mathrm{d}\mathbf{x} = -\nabla U(\mathbf{x})\mathrm{d}t + \sqrt{2}\mathrm{d}\mathbf{W} \tag{1}$$

where $U(\mathbf{x})$ is a continuously differentiable potential usually associated with a probability density function $p_\star(\mathbf{x}) \propto e^{-U(\mathbf{x})}$. It can be shown that under mild conditions, the solution of Equation (1) converges to a stationary distribution proportional to $p_\star(\mathbf{x})$.

2.1.1 Unadjusted Langevin Dynamics

Most of the methods discussed in this article are based on the Unadjusted Langevin Dynamics algorithm (ULA). This is an Euler-Maruyama discretization of Equation (1) given by:

$$\mathbf{x}_t = \mathbf{x}_{t-1} - \gamma_t \nabla_{\mathbf{x}} U(\mathbf{x}_{t-1}) + \sqrt{2\gamma_t}\mathbf{z}_t \tag{2}$$

where $(\gamma_t)_{t\geq 0} \subseteq \mathbb{R}_{++}$ is a sequence of step-sizes and $(\mathbf{z}_t)_{t\geq 0}$ is a sequence of *i.i.d* $\mathcal{N}(0, I)$ random variables.

The convergence of ULA is well-studied in for instance (Roberts and Tweedie, 1996) .

2.1.2 Preconditioned Unadjusted Langevin Dynamics

The preconditioned Unadjusted Langevin Dynamics (PULA) modifies ULA by introducing a symmetric, positive definite preconditioner $\mathbf{P}$ to Equation (1), which admits the form:

$$\mathrm{d}\mathbf{x} = -\mathbf{P}\nabla U(\mathbf{x})\mathrm{d}t + \sqrt{2}\mathbf{P}^{1/2}\mathrm{d}\mathbf{W} \tag{3}$$

The addition of a preconditioner does not change the stationary distribution of Equation (1) (Akyildiz et al., 2021). However, this may not be the case when $\mathbf{P}$ depends on $\mathbf{x}$. In Appendix B, we show the acceleration of PULA in the Gaussian case explicitly.

2.1.3 Stochastic Gradient Langevin Dynamics (SGLD)

First introduced by Welling and Teh (2011) in the context of Bayesian inference, SGLD samples from the posterior distribution $p(\boldsymbol{\theta} \mid X) \propto p(\theta) \prod_{i=1}^{N} p(\mathbf{x}_i \mid \boldsymbol{\theta})$ by using a noisy, unbiased estimate of the gradient in ULA taken from a mini-batch of data $\{\mathbf{x}_i\}_{1\leq i\leq n}$. One update step is given by:

$$\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} - \gamma_t \left(\nabla \log p(\theta_{t-1}) + \frac{N}{n} \nabla \log p(\mathbf{x}_i \mid \boldsymbol{\theta_{t-1}}) \right) + \sqrt{2\gamma_t}\mathbf{z}_t \tag{4}$$

When we are not in a Bayesian setting, Equation (4) reduces to ULA.

2.2 Diffusion Probabilistic Models

Diffusion Probabilistic Models (DPMs), first introduced by Sohl-Dickstein et al. (2015) have emerged as one of state-of-the-art approaches for image generation tasks (Dhariwal and Nichol, 2021). While there are numerous variants of DPMs, most of its variants consist of a forward process that slowly adds random noise directly to the data distribution and a reverse process that learns to recover the data distribution from noise for generation. We now introduce two popular versions of DPMs.

2.2.1 Score-based Generative Models (SGMs)

In SGMs, the data distribution is perturbed with a diffusion process and this process is reversed for generative modelling. The forward process is described by the following SDE:

$$\mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x}, t)dt + g(t)\mathrm{d}\mathbf{W} \tag{5}$$

Anderson (1982) shows that the reverse SDE is also a diffusion process, which is described by:

$$\mathrm{d}\mathbf{x} = [\mathbf{f}(\mathbf{x}, t) - g(t)^2 \nabla_{\mathbf{x}} \log p_t(\mathbf{x})dt + g(t)\mathrm{d}\bar{\mathbf{W}} \tag{6}$$

where $\bar{\mathbf{W}}$ is a standard Brownian motion with time flowing from T to 0.

By training a time-dependent denoising score matching network $\mathbf{s}_\theta(\mathbf{x}, t)$ to estimate the score $\nabla_{\mathbf{x}} \log p_{0T}(\mathbf{x}(t)|\mathbf{x}(0))$ at each time step, it is possible to solve the reverse SDE by discretization:

$$\mathbf{x}_{t-\Delta t} = \mathbf{x}_t - [\mathbf{f}(\mathbf{x}_t, t) - g(t)^2 \mathbf{s}_\theta(\mathbf{x}_t, t)]\Delta t + g(t)\mathbf{z}_t\sqrt{\Delta t} \tag{7}$$

The connection between ULA and SGMs is established by Song and Ermon (2019), in which a modified version of ULA called annealed Langevin dynamics (ALD) is introduced. ALD is originally proposed for a noise-dependent network $\mathbf{s}_\theta(\mathbf{x}, l)$ that uses noise levels $(\sigma_l)_{l=1}^L$ and a trained network is used to predict the score for data perturbed by Gaussian noise $\mathcal{N}(0, \sigma_l^2 I)$ at each level l . ALD thus admits the update step at noise level l as:

$$\mathbf{x}_{t-1} = \mathbf{x}_t + \gamma\frac{\sigma_l}{\sigma_L} + \mathbf{s}_\theta(\mathbf{x}_t, l) + \sqrt{\frac{\sigma_l}{\sigma_L}}\mathbf{z}_t \tag{8}$$

In the continuous-time framework (Song et al., 2021), given a learned, time-dependent score estimating network $\mathbf{s}_\theta(\mathbf{x}_t, t)$, ALD is used as a complement to the reverse diffusion Equation (7):

$$\mathbf{x}_{t-1} = \mathbf{x}_t + \gamma_t \mathbf{s}_\theta(\mathbf{x}_t, t) + \sqrt{2\gamma_t}\mathbf{z}_t \tag{9}$$

where $(\gamma_t)_{1\leq t\leq T}$ is a sequence of decreasing stepsizes, which is formulated in detail in Section 4.1.

2.2.2 Alternative views of Diffusion Models

While Song et al. (2021) have established connections between DDPM and SGMs through variance-preserving (VP) SDEs, DDIM introduced by Song et al. (2022) uses a non-Markovian forward process, which does not fit into the continuous-time framework. The forward process is defined implicitly through the Bayes' Rule by specifying:

$$q_\sigma(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}\left(\sqrt{\alpha_{t-1}}\mathbf{x}_0 + \sqrt{1 - \alpha_{t-1} - \sigma_t^2} \cdot \frac{\mathbf{x}_t - \sqrt{\alpha_t}\mathbf{x}_0}{\sqrt{1-\alpha_t}},\ \sigma_t^2 I\right) \tag{10}$$

where $\sigma = (\sigma_t)_{1\leq t\leq T}$ is a real non-negative valued vector. The reverse sampling process of DDIM is given by the one-step update:

$$\mathbf{x}_{t-1} = \sqrt{\alpha_{t-1}} f_\theta(\mathbf{x}_t, t) + \sqrt{1 - \alpha_{t-1} - \sigma_t^2} \cdot \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) + \sigma_t \mathbf{z}_t \tag{11}$$

Here the $f_\theta(\mathbf{x}, t) = \frac{\mathbf{x} - \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)}{\sqrt{\alpha_t}}$ predicts the $\mathbf{x}_0$ from the noised sample $\mathbf{x}$ and $\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)$, in contrast with $\mathbf{s}_\theta(\mathbf{x}_t, t)$, predicts the noise added at time t to the sample $\mathbf{x}_t$.

3 Adaptive SGLD

The similarity between the update step in Langevin dynamics and gradient descent has led to the development of various SGLD-type algorithms borrowing ideas from the optimization literature. In particular, the empirical success of adaptive gradient optimizers in deep learning has inspired pSGLD with RMSProp (Li et al., 2015) and variants of it. In this section, we formulate the algorithms that will be used, briefly discuss their advantage and shortcomings, and introduce two new algorithms. We denote the target distribution as $p_\star(\mathbf{x})$ and the score function $\nabla_{\mathbf{x}} \log p_\star(\mathbf{x}_t)$ as $\nabla_{\mathbf{x}} U(\mathbf{x})$.

3.1 pSGLD with RMSProp

Originally proposed by Li et al. (2015) for Bayesian inference and optimization, pSGLD with RMSProp uses a preconditioner as an exponential moving average of past gradients. Under their general framework, a single update step is given by:

$$\Delta \mathbf{x}_t \sim \gamma_t \left(G(\mathbf{x}_t) \nabla_{\mathbf{x}} U(\mathbf{x}_t) + \Gamma(\mathbf{x_t}) \right) + \sqrt{2\gamma_t} G(\theta)^{1/2} \mathbf{z}_t \tag{12}$$

where $G(\mathbf{x}_t)$ is the preconditioner, $(\gamma_t)_{t \geq 0}$ is a sequence of stepsizes ,$\mathbf{z}_t \sim \mathcal{N}(0, I)$ is an independent standard normal random variable, and $\Gamma(\mathbf{x}) = \sum_j \frac{\partial G_{i,j}(\mathbf{x})}{\partial \mathbf{x}^{(j)}}$ is the term accounting for the bias. In practice, this $\Gamma(\mathbf{x})$ is often ignored to save compute, but this will introduce permanent bias from the target distribution, which is a common shortcoming for all algorithms discussed in this section.

In the specific case of RMSProp, the preconditioner is constructed by

$$\begin{aligned} G_t &= \left(V_t^{-1/2} + \varepsilon I \right) \\ V_t &= \beta V_{t-1} + (1-\beta) \text{diag} \left(\nabla_{\mathbf{x}} U(\mathbf{x}_{t-1}) \nabla_{\mathbf{x}} U(\mathbf{x}_{t-1})^T \right) \end{aligned}$$

where we have abbreviated $G(\mathbf{x}_t)$ as G_t, ε is a small constant to control the extreme values of V_t, and β is a hyperparameter for smoothing the past gradients.

Algorithm 1 pSGLD with RMSProp

Initialize $\mathbf{x}_0$ and moving average $V_0 \leftarrow 0$
for $t = 1$ to N **do**
  Compute score $U(\mathbf{x}_t)$
  Update $V_t \leftarrow \beta V_{t-1} + (1-\beta) \text{diag} \left(\nabla_{\mathbf{x}} U(\mathbf{x}_{t-1}) \nabla_{\mathbf{x}} U(\mathbf{x}_{t-1})^T \right)$
  Construct $G_t \leftarrow \left(V_t^{-1/2} + \varepsilon I \right)$
  Update $\mathbf{x}_t = \mathbf{x}_{t-1} + \gamma_t \left(G_t \nabla_{\mathbf{x}} U(\mathbf{x}_{t-1}) \right) + \sqrt{2\gamma_t} G_t^{1/2} \mathbf{Z}_t$, where $\mathbf{Z}_t \sim \mathcal{N}(0, I)$
end for
return $\{\mathbf{x}_0, \ldots, \mathbf{x}_N\}$

3.2 pSGLD with Adam

Kim et al. (2020) have proposed a variant of pSGLD called ASGLD which admits the update step:

$$\Delta \mathbf{x}_t \sim \gamma_t \left(\nabla_{\mathbf{x}_t} U(\mathbf{x}_t) + a G_t M_t \right) + \sqrt{2\gamma_t \tau} \mathbf{Z}_t \tag{13}$$

where a is a bias factor and τ is a temperature parameter. The preconditioner G_t and momentum M_t 's update rules are as follows:

$$\begin{aligned} G_t &= (V_t + \varepsilon I)^{-1/2} \\ M_t &= \beta_1 M_{t-1} + (1-\beta_1) \nabla_{\mathbf{x}} U(\mathbf{x}_{t-1}) \\ V_t &= \beta_2 V_{t-1} + (1-\beta_2) \text{diag} \left(\nabla_{\mathbf{x}} U(\mathbf{x}_{t-1}) \nabla_{\mathbf{x}} U(\mathbf{x}_{t-1})^T \right) \end{aligned}$$

To further explore the analogies between adaptive gradient optimizers and SGLD algorithms, we modify the update step in Equation (13) and add momentum correction to get:

$$\Delta \mathbf{x}_t \sim \gamma_t (1-\beta_2^t)^{-1} G_t (1-\beta_1^t)^{-1} M_t + ((1-\beta_2^t)^{-1} G_t)^{1/2} \sqrt{2\gamma_t} \mathbf{Z}_t \tag{14}$$

which matches the original Adam (Kingma and Ba, 2017) algorithm more closely and is more consistent with pSGLD with RMSProp (Algorithm 1). We summarize the algorithm below.

Algorithm 2 pSGLD with Adam

Initialize $\mathbf{x}_0$, moving average $V_0 \leftarrow 0$, momentum
for $t = 1$ to N **do**
  Compute score $U(\mathbf{x}_t)$
  Update $M_t \leftarrow \beta_1 M_{t-1} + (1-\beta_1)\nabla U(\mathbf{x}_{t-1})$
  Update $V_t \leftarrow \beta V_{t-1} + (1-\beta)\mathrm{diag}\left(\nabla_{\mathbf{x}} U(\mathbf{x}_{t-1})\nabla_{\mathbf{x}} U(\mathbf{x}_{t-1})^T\right)$
  Add correction, $\hat{M}_t \leftarrow M_t/(1-\beta_1^t)$ and $\hat{V}_t \leftarrow V_t/(1-\beta_2^t)$
  Construct $G_t \leftarrow \left(\hat{V}_t + \varepsilon I\right)^{-1/2}$
  Update $\mathbf{x}_t = \mathbf{x}_{t-1} + \gamma_t \hat{G}_t \hat{M}_t + \hat{G}_t^{1/2}\sqrt{2\gamma_t}\mathbf{Z}_t$, where $\mathbf{Z}_t \sim \mathcal{N}(0, I)$
end for
return $\{\mathbf{x}_0, \ldots, \mathbf{x}_N\}$

3.3 SGRLD with Monge metric

Inspired by Langrangian Monte Carlo with the Monge metric (Hartmann et al., 2022), Yu et al. (2023) propose to use

$$I + \alpha^2 \nabla_{\mathbf{x}} U(\mathbf{x}_t)\nabla_{\mathbf{x}} U(\mathbf{x}_t)^T$$

as the metric, which induces the preconditioner

$$G_t = I - \frac{\alpha^2}{1 + \alpha^2 \|\nabla_{\mathbf{x}} U(\mathbf{x}_t)\|_2^2}\nabla_{\mathbf{x}} U(\mathbf{x}_t)\nabla_{\mathbf{x}} U(\mathbf{x}_t)^T$$

Compared to Algorithm 1, SGRLD with Monge metric uses a full-matrix for preconditioning, which makes it more suitable for sampling from distributions with highly correlated dimensions.

3.4 Forgetful pSGLD

In the situation where pSGLD starts at a badly initialised location, initial values of the gradients may have a large impact that is not entirely eliminated by the weight β^{-t}. While this may be solved by decreasing β, such an approach could reduce the effect of more recent gradients, which is undesirable. Thus, we propose the forgetful pSGLD with RMSProp with similar motivations as in (Agarwal et al., 2020).

We introduce a sequence times $\{T_1, \ldots, T_n\}$ at which the accumulated squared gradients V_t is set to zero. In choosing this sequence, we can make the sampler adapt better when the geometry of the underlying manifold is rapidly changing. We summarize the algorithm below, which is essentially the same as Algorithm 1 except for the forgetting step.

Algorithm 3 Forgetful pSGLD with RMSProp 1

Initialize $\mathbf{x}_0$ and moving average $V_0 \leftarrow 0$
for $t = 1$ to N **do**
  Compute score $U(\mathbf{x}_t)$
  Update $V_t \leftarrow \beta V_{t-1} + (1-\beta)\mathrm{diag}\left(\nabla U(\mathbf{x}_{t-1})\nabla U(\mathbf{x}_{t-1})^T\right)$
  Construct $G_t \leftarrow (V_t + \varepsilon I)^{-1/2}$
  Update $\mathbf{x}_t = \mathbf{x}_{t-1} + \gamma_t\left(G_t \nabla U(\mathbf{x}_{t-1})\right) + \sqrt{2\gamma_t} G_t^{1/2}\mathbf{Z}_t$, where $\mathbf{Z}_t \sim \mathcal{N}(0, I)$
  if $t = T_k$ **then**
    Set $V_t = 0$ ▷ Forget step
  end if
end for
return $\{\mathbf{x}_0, \ldots, \mathbf{x}_N\}$

In our experiments, we generally set T_k to be ck for some positive integer c and $cn < N$, so we set $V_t = 0$ whenever $t \equiv 0 \pmod{c}$ and fall back to the usual pSGLD with RMSProp after iteration cn.

An alternative form that follows directly from Agarwal et al. (2020) requires storage of a subset of size $r > 0$ consisting of past gradients, which may add a substantial overhead to the compute. We summarize the algorithm below.

Algorithm 4 Forgetful pSGLD with RMSProp 2

Initialize $\mathbf{x}_0$, window size $r > 0$
for $t = 1$ to N **do**
  Compute score $\nabla U(\mathbf{x}_{t-1})$
  Update storage of gradients $\{\nabla U(\mathbf{x}_i)\}_{\min(0,t-r)\leq i\leq t-1}$
  Compute $V_t = (1-\beta)\sum_{i=\min(0,t-r)}^{t-1}\beta^{t-i}\text{diag}(\nabla U(\mathbf{x}_i)\nabla U(\mathbf{x}_i)^T)$
  Construct $G_t \leftarrow (V_t + \varepsilon I)^{-1/2}$
  Update $\mathbf{x}_t = \mathbf{x}_{t-1} + \gamma_t\left(G_t\nabla U(\mathbf{x}_{t-1})\right) + \sqrt{2\gamma_t}G_t^{1/2}\mathbf{Z}_t$, where $\mathbf{Z}_t \sim \mathcal{N}(0, I)$
end for
return $\{\mathbf{x}_0, \ldots, \mathbf{x}_N\}$

4 Adaptive SGLD for SGMs

Sampling from diffusion models can often be quite expensive due to the evaluation of the score-estimating neural network at each step. Given the empirical success of Ma et al. (2022)'s application of preconditioner to the sampling process of DPMs, we explore the possibilities of applying adaptive preconditioners to the reverse process. The advantages of adaptively preconditioned samplers are evident: in contrast to (Ma et al., 2022), they do not need to be learnt from the data distribution. In addition, the adaptive preconditioners we will discuss do not require the model to be re-trained. One concurrent work (Wang et al., 2023) has proposed a similar idea of using Adam inspired adaptive sampler for the reverse process in DDIM, while our method is based on SGMs.

We introduce two adaptive samplers which are naive modifications of the predictor (the discretization of the reverse SDE in Equation (6)) and corrector (a version of annealed Langevin dynamicss (ALD)) in (Song et al., 2021). Additionally, we give variants of those two samplers that use a moving average of the mean squared scores, which is shown to be effective. Moreover, we propose a simple modification to the Predictor-Corrector (PC) sampling for low number of NFEs.

In this section, we use the notation $\mathbf{x}_T$ for the input at time T and the final denoised sample as $\mathbf{x}_0$; the sampler will run from time T to 0.

4.1 RMSProp Corrector

The first method follows from pSGLD with RMSProp, which modifies the corrector sampler update step.

$$\mathbf{x}_{t-1} = \mathbf{x}_t + \gamma_t G_t \mathbf{s}_\theta(\mathbf{x}_t, t) + \sqrt{2\gamma_t}G_t^{1/2}\mathbf{z}_t \tag{15}$$

where G_t is constructed as in Algorithm 1 . For the original corrector update in Song et al. (2021), the stepsizes are constructed as

$$\gamma_t = 2r\left(\frac{\|\mathbf{z}_t\|}{\|\mathbf{s}_\theta(\mathbf{x}_t, t)\|}\right)^2$$

where r is a constant called the signal-to-nosie ratio. In our experiments, however, we will use constant γ_t during sampling and demonstrate the similar annealing effect of RMSProp corrector, which is not surprising, as V_t is a moving average of the $\mathbf{s}_\theta(\mathbf{x}_t, t)$ squared component-wise

We also apply Section 4.1 to NCSN sampling (Song and Ermon, 2019), which uses ALD by default. The proposed ALD require n steps at each noise level l and the stepsize is explicitly based on the noise levels. We instead use the adaptive preconditioner G_t across all noise levels, as outlined in Algorithm 5

4.2 RMSProp Predictor

Following the approach similar to (Ma et al., 2022), we can apply the preconditioner naively to the reverse process in SGMs. The presence of an estimated gradient term of the transition kernel $p_{0T}(\mathbf{x}(t) \mid \mathbf{x}(0))$ motivates the update rule:

$$\mathbf{x}_{t-1} = \mathbf{x}_t - [\mathbf{f}(\mathbf{x}_t, t) - g(t)^2 G_t \mathbf{s}_\theta(\mathbf{x}_t, t)]\Delta t + g(t)\mathbf{z}_t G_t^{1/2}\sqrt{\Delta t} \tag{16}$$

where G_t is a preconditioner constructed similarly in Algorithm 1 and the rest is a modification of the update Equation (7). The pseudocode of this algorithm is given in the Appendix E.

We comment that this algorithm apparently does not achieve a promising level of acceleration (Section 6), as the theoretical foundation is not sound.

Algorithm 5 RMSProp Corrector for NCSN

Initialize $\mathbf{x}_0 \sim \mathcal{N}(0, \sigma_1)$, $V_{0,0} \leftarrow 0$, require noise levels $(\sigma_l)_{l=1}^L$ and stepsize γ
for $l = 1$ to L **do**
  for $i = 1$ to n **do**
    Compute score $\mathbf{s}_i \leftarrow \mathbf{s}_\theta(\mathbf{x}_i, l)$ at current noise level
    Update $V_{l,i} \leftarrow \beta V_{l,i-1} + (1-\beta)\text{diag}(\mathbf{s}_{i-1}\mathbf{s}_{i-1}^T)$
    Construct $G_{l,i} \leftarrow (V_{l,i})^{-1/2} + \varepsilon I$
    Update $\mathbf{x}_i = \mathbf{x}_{i-1} + \gamma\left(G_{l,i}\mathbf{s}_i\right)) + \sqrt{2\gamma}G_{l,i}^{1/2}\mathbf{z}_t$, where $\mathbf{z}_t \sim \mathcal{N}(0, I)$
  end for
  Set $\mathbf{x}_0 \leftarrow \mathbf{x}_n$
end for
return $\mathbf{x}_n$

4.3 Scalar Variants of RMSProp samplers

In application to SGMs, we find that similar/better results can be achieved when the moving average V_t is constructed as a scalar multiple of the identity matrix $\lambda\mathbf{I}$:

$$V_{t-1} = \beta V_t + (1-\beta)\frac{1}{d}\|\mathbf{s}_\theta(\mathbf{x}_t, t)\|_2^2\mathbf{I} \tag{17}$$

where d is the dimension of $\mathbf{x}_t$. When used in practice, the preconditioner can be constructed using much less memory if the dimension is high. This construction also follows more closely from the annealed stepsizes in Section 4.1.

4.4 Tuning the step size in PC-Sampling

We introduce an extra stepsize parameter $r > 0$ for the predictor update in (Song et al., 2021), where $r = 1.0$ corresponds to the original algorithm. We modify the predictor update step to become:

$$\mathbf{x}_{t-\Delta t} = \mathbf{x}_t - [f(\mathbf{x}_t, t) - g(t)^2\mathbf{s}_\theta(\mathbf{x}_t, t)]r\Delta t + g(t)\mathbf{z}_t\sqrt{r\Delta t} \tag{18}$$

We do not alter the corrector step. The additional hyperparameter r is adjusted according to the number of time steps T; it can be seen as a further simplification of the scalar RMSProp (Equation (17)) with a constant 'preconditioner'.

5 Experiments on Tractable Distributions

We present quantitative and qualitative results for analytically tractable distributions in Section 5.1 and Section 5.2; the performance of our sampler is measured using Wasserstein-2 distance those cases. The Wasserstein-2 distance between two probability distributions μ_p and μ_q is defined as:

$$W_2(\mu_p, \mu_q) = \left(\inf_{\zeta \in \Gamma(\mu_p, \mu_q)} \int_{\mathbb{R}^d \times \mathbb{R}^d} \|x - y\|^2 d\zeta(x, y)\right)^{1/2}$$

where $\Gamma(\mu_p, \mu_q)$ denotes the set of joint probability measures called transport plans such that the marginals are μ_p and μ_q. The Wasserstein-2 distance is often seen as a suitable metric for comparing probability distributions and a better metric than total variation (Dalalyan, 2017).

5.1 Simple Gaussians

We test the performance of pSGLD with RMSProp Algorithm 1, pSGLD with Adam, two forgetful variants of pSGLD, and SRGLD with Monge metric against ULA on the following target distributions:

- ill-conditioned Gaussian on a small scale: $\mathcal{N}(0, \Sigma_1)$, where $\Sigma_1 = \begin{pmatrix} 10^{-3} & 0 \\ 0 & 1 \end{pmatrix}$
- ill-conditioned Gaussian on a large scale: $\mathcal{N}(0, \Sigma_2)$, where $\Sigma_2 = \begin{pmatrix} 1 & 0 \\ 0 & 10^3 \end{pmatrix}$

- Gaussian with high covariance: $\mathcal{N}(0, \Sigma_3)$, where $\Sigma_3 = \begin{pmatrix} 1 & 1-10^{-3} \\ 1-10^{-3} & 1 \end{pmatrix}$

In all our experiments in this section, the constant step size and hyperparameters are determined using grid-search. (see Appendix C)

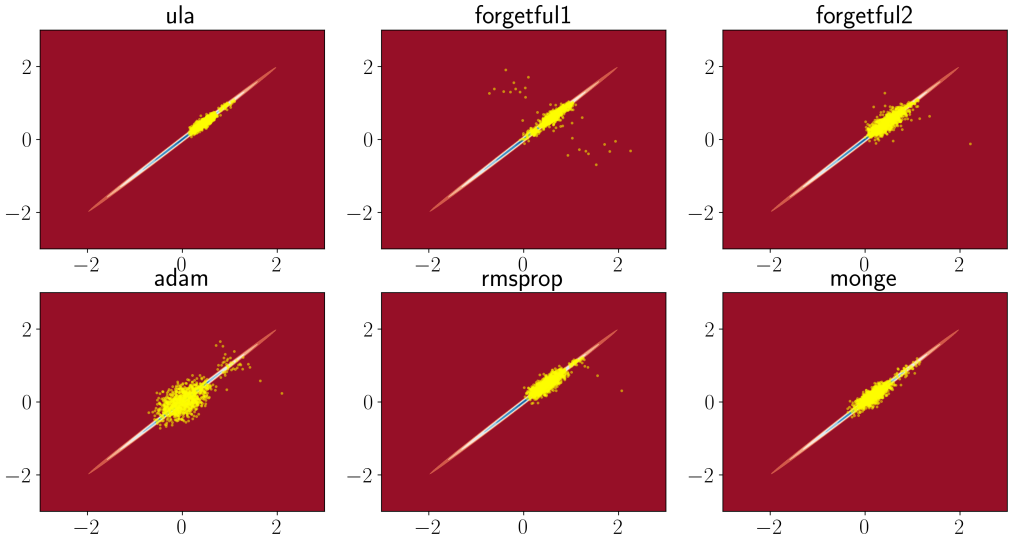
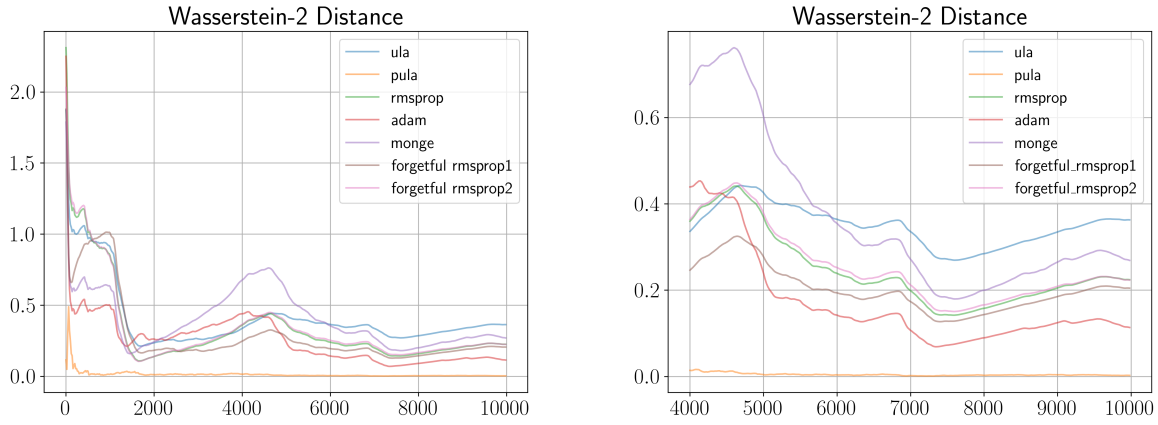


Figure 1: Comparison of methods on $\mathcal{N}(0, \Sigma_1)$ when number of samples $N = 1000$. The generated samples are marked in yellow. All samplers are initialised at $\mathbf{x}_0 = (1, 1)$

In the case of Gaussian distributions μ_p and μ_q, we have closed-form expressions for the Wasserstein-2 distance:

$$W_2(\mu_p, \mu_q)^2 = \frac{1}{2}\left(\|\mu_p - \mu_q\|_2^2 + \text{tr}\left(\Sigma_p + \Sigma_q - 2\sqrt{\Sigma_p \Sigma_q}\right)\right)$$



(a) Wasserstein-2 distance

(b) Zoomed in Wasserstein-2 distance

Figure 2: Wasserstein-2 distance between generated samples and ground truth $\mathcal{N}(0, \Sigma_1)$ computed every 20 samples

As we can observe from both Figure 1 and Figure 2b, the adaptive pSGLD methods are not effective as the PULA, which is expected in the high covariance setting.

For uncorrelated Gaussians with large difference in scales, our experiment shows that adaptive preconditioners can effectively acclerate the sampling process.

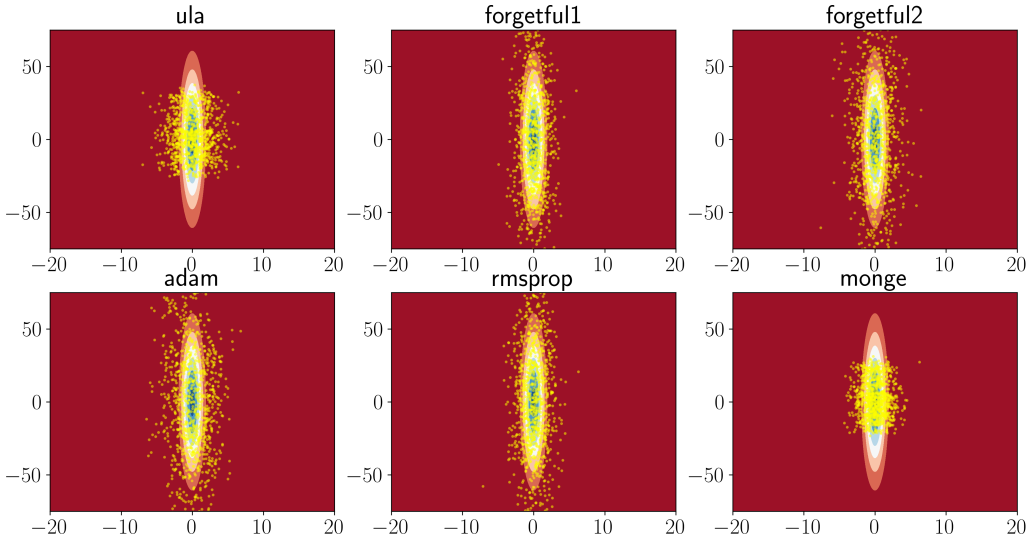
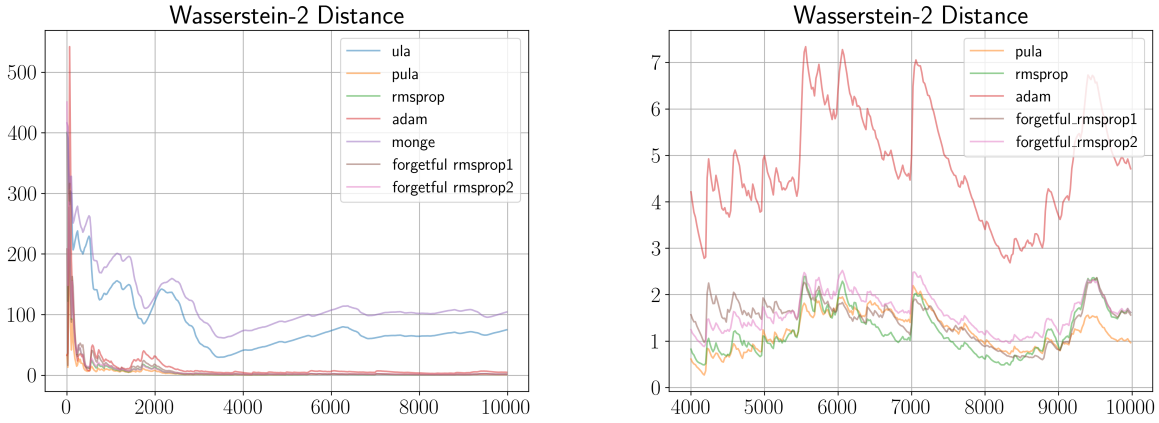


Figure 3: Comparison of methods on $\mathcal{N}(0, \Sigma_2)$ when number of samples $N = 1000$.

For Figure 4b, we exclude the comparison of the monge and ULA samplers to better visualise the evolution of Wasserstein-2 distance for other samplers.



(a) Wasserstein-2 distance

(b) Zoomed in Wasserstein-2 distance

Figure 4: Wasserstein-2 distance between generated samples and ground truth $\mathcal{N}(0, \Sigma_2)$ computed every 20 samples

Analogous to the uncorrelated Gaussian on a large scale, our experiment demonstrates competitive performance of the two forgetful pSGLD with RMSProp algorithms.

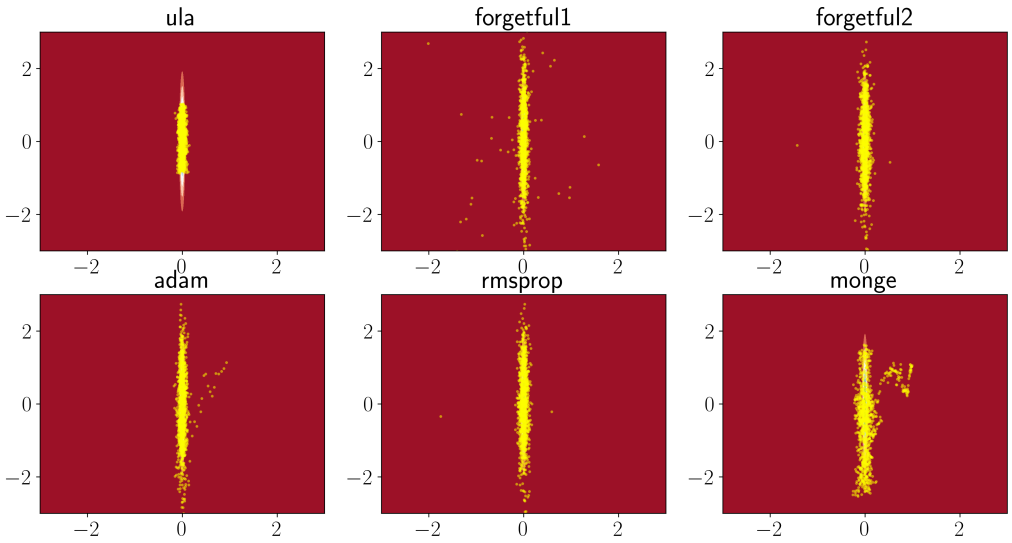
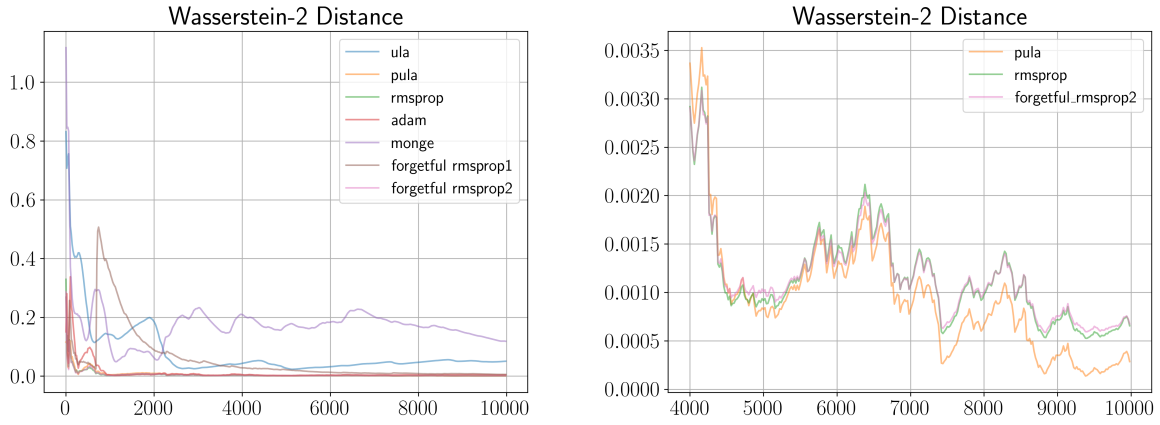


Figure 5: Comparison of methods on $\mathcal{N}(0, \Sigma_3)$ when number of samples $N = 1000$.

We comment that the improvement in performance of the forgetful samplers compared to the rmsprop sampler is a result of the small scale of the Gaussian: the initial values now have a larger impact on the subsequent values of the V_t, which in turn affects the trajectory of the sampler.



(a) Wasserstein-2 distance

(b) Zoomed in Wasserstein-2 distance

Figure 6: Wasserstein-2 distance between generated samples and ground truth $\mathcal{N}(0, \Sigma_3)$ computed every 20 samples

5.2 Twisted Gaussians

We also consider the samplers' performance on highly nonlinear combination of Gaussian distributions called "Twisted Gaussians" introduced by Haario et al. (1999), this is a distribution with density f_{b,σ_1} given by:

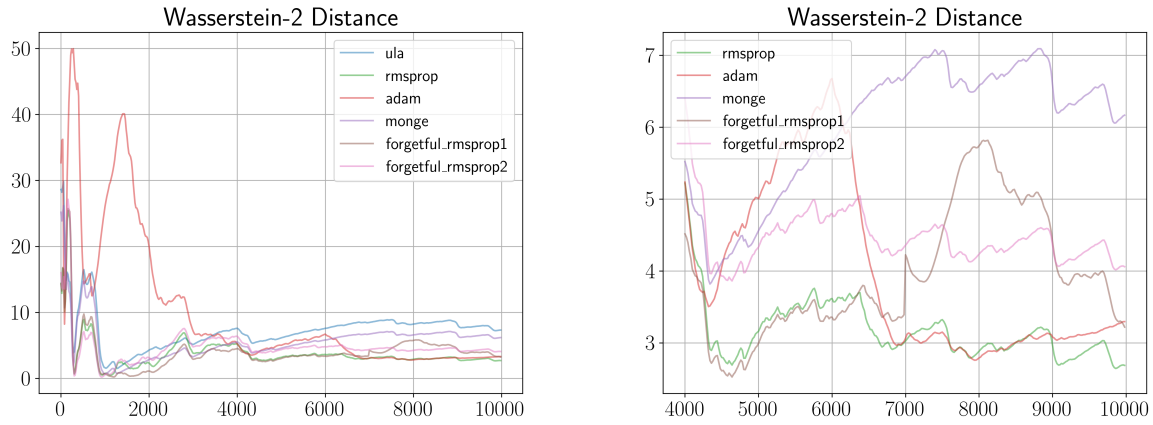
$$f_{b,\sigma_1^2} = f_{\sigma_1^2} \circ g_b$$

where $f_{\sigma_1^2}$ is the density of a $d-$dimensional Gaussian $\mathcal{N}(0, \Sigma)$ with covariance $\Sigma = \text{diag}(\sigma_1^2, 1, \ldots, 1)$ and $g_b : \mathbb{R}^d \to \mathbb{R}^d$ is defined as:

$$g_b(x) = (x^{(1)}, x^{(2)} + (bx^{(1)})^2 - \sigma_1^2 b, x^{(3)}, \ldots, x^{(d)})$$

By applying an inverse transformation, we can compute the Wasserstein-2 distance and Kullback-Leibler divergence as in the Gaussian case.

In our experiments, we use $b = 0.05$ and $\sigma_1^2 = 100$.



(a) Wasserstein-2 distance

(b) Zoomed in Wasserstein-2 distance

Figure 7: Wasserstein-2 distance between generated samples and twisted Gaussian $f_{0.05,100}$ computed every 20 samples

The empirical results illustrate some advantage of the adaptive pSGLD with RMSProp samplers over ULA, especially their capability of sampling from low probability density regions Figure 8, while the Adam sampler displays a random walk behaviour at $N = 1000$ samples.

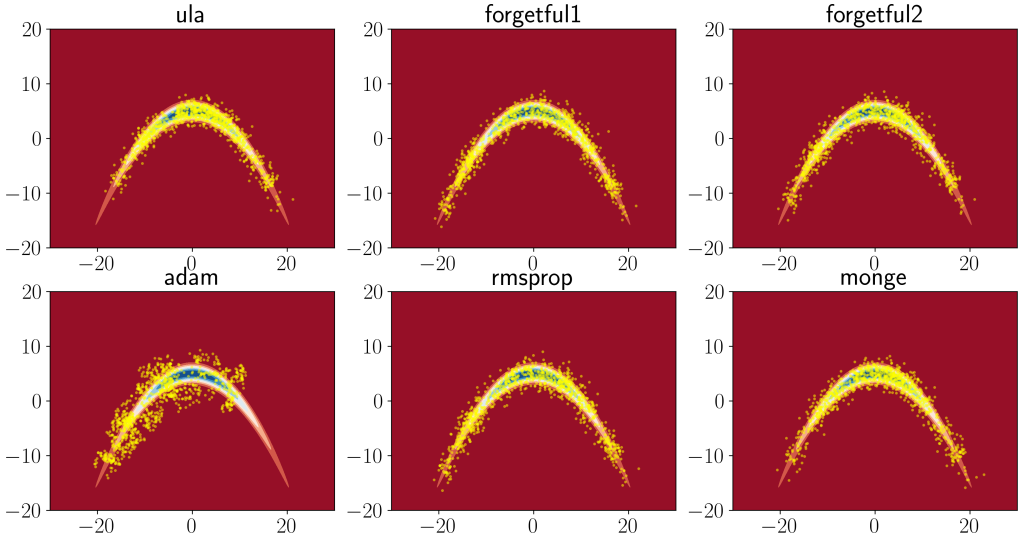


Figure 8: Comparison of methods on the twisted Gaussian when number of samples $N = 1000$.

6 Experiments on SGMs

In this section, we investigate the performance RMSProp corrector (Equation (15)), RMSProp predictor (Equation (16)), and their scalar variants Equation (17) on the MNIST digits (LeCun et al., 2010) as well as the Fashion MNIST dataset (Xiao et al., 2017). For experiments on the CIFAR-10 dateset (Krizhevsky, 2009), we

We use a Variance-Exploding SDE for the forward process used in the tutorial notebook provided by Song et al. (2021):

$$\mathrm{d}\mathbf{x} = \sigma^t \mathrm{d}\mathbf{W} \tag{19}$$

The network architecture and derivation of the perturbation kernel is included in Appendix D.2.

6.1 Corrector Only Sampling

In this subsection, we use constant stepsizes for RMSProp corrector (Equation (15)) and its scalar variant. We show empirically that sampling with those RMSProp methods can achieve similar results when annealed stepsizes are used (Figure 9). For SGM, we set the number of sampling steps to $N = 500$, $\beta = 0.99$, and set a constant stepsize $\gamma_t = 0.05$ for RMSProp correctors. Annealed Langevin dynamics corrector uses the recommended hyperparameter setting by Song et al. (2021) with signal-to-noise ratio 0.16.

(a) MNIST samples

(b) Fashion MNIST samples

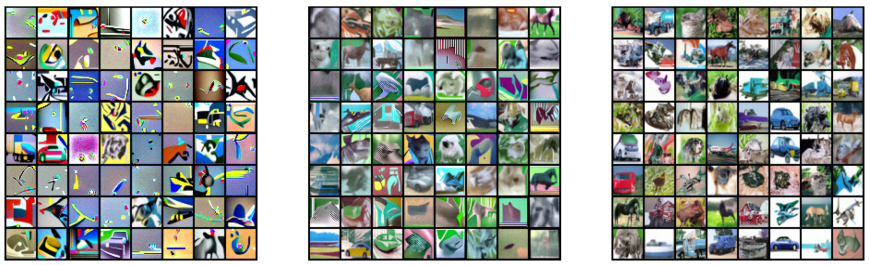
Figure 9: Samples generated using $N = 500$ steps. From left to right: RMSProp corrector, Scalar RMSProp corrector, and annealed Langevin dynamics corrector

For experiments on NCSN, we set $n = 10, \gamma = 0.01$ in Algorithm 5 for the MNIST dataset and $n = 100, \gamma = 0.015$ for the CIFAR-10 dataset. The comparison is made against ALD with the same n and recommended $\gamma = 2 \times 10^{-5}$ in Song and Ermon (2019).

	Baseline (Song et al., 2021)	PDS (Ma et al., 2022)	Baseline with increased step-size
$T = 50$	306.91	4.90	5.64
$T = 100$	29.39	2.90	3.04

Table 1: FIDs computed from $50,000$ samples.

(a) MNIST samples



(b) CIFAR-10 samples

Figure 10: Samples from NCSN. From left to right: RMSProp corrector, Scalar RMSProp corrector, and annealed Langevin dynamics corrector

This empirical result demonstrates the ability of RMSProp correctors to adapt to multiple noise levels automatically. Given the connection between SGMs and other discrete-time diffusion models trained across different noise scales, we comment that adaptive samplers like the RMSProp correctors can be possibly applied in those models as well, which we leave for future work.

6.2 Predictor Only Sampling

For sampling with RMSProp predictors (Equation (16)), our experiments show an apparent acceleration when the number of time steps is small. This is due to the fact that RMSProp counteracts the effect of explosion of drift coefficient (Kim et al., 2022). We set the sampling steps to $N = 50$ and $\beta = 0.99$.

We first visualise the samples at the final time-step, which shows the apparent acceleration effect of RMSProp predictors (Figure 11)

However, when we plot the samples from the previous iteration, the samples from the reverse diffusion predictor are of better quality and higher diversity among samples. (Figure 12)

For quantitative results, we provide the loss and L^2 norm of the estimated score in Appendix D.3.

6.3 Baseline with increased step-size

We now provide quantitative and qualitative results produced by the scaled stepsize sampling described in Section 4.4. We compare the FIDs to Ma et al. (2022) and demonstrate the efficacy of this simple modification on CIFAR-10 with NCSN++ model (Song et al., 2021) (Table 1). For $T = 50$, we use $r = 1.27$ and for $T = 100$ we use $r = 1.08$. In all three cases, the PC sampler is used, which consists of one predictor step and one corrector step at each t, thus the NFE is double the number of time steps T.

(a) MNIST samples

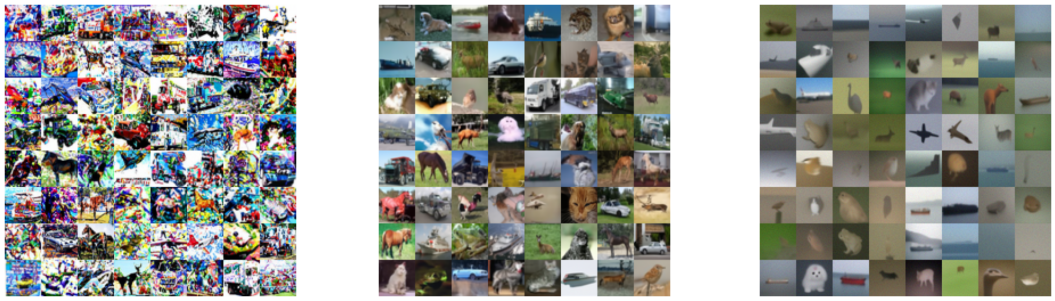
(b) Fashion MNIST samples

Figure 11: Samples generated with predictors using $N = 50$ steps. From left to right: RMSProp predictor, Scalar RMSProp predictor, and reverse diffusion predictor

(a) MNIST samples

(b) Fashion MNIST samples

Figure 12: Samples at iteration $i = 49$ generated with predictors using $N = 50$ steps.



(a) $r = 1.0$ (b) $r = 1.2$ (c) $r = 2.5$

Figure 13: Samples at different step sizes using reverse diffusion predictor with $N = 100$

However, scaled stepsize sampling may produce smooth and blurry images when r is large (Figure 13c). A similar phenomenon is discussed in Dhariwal and Nichol (2021) as navie attempts to accelerate sampling. However, their attempts either only scale the Gaussian noise $\mathbf{z}_t$ or the estimated score, where we apply scaling to both components (Section 4.4).

7 Conclusion

In this article, variants of pSGLD have been proposed and their performance have been evaluated on tractable distributions as well as SGMs .

We have introduced two new variants of pSGLD that attempt to alleviate the influence of an undesirable initial value on the sampler. In addition, we have proposed adaptive correctors and predictors for SGMs.

Our experiments on analytically tractable distributions partially verify the speculation that pSGLD and its variants are able to resolve problems arising with vastly different scales across dimensions. While our results for corrector-only sampling with RMSProp have shown to be a viable alternative for annealed Langevin dynamics, RMSProp predictors have not shown competitive performance due to their lack of theoretical foundation.

Future work could explore PC-sampling with other adaptive preconditioners like the Monge metric (Yu et al., 2023; Hartmann et al., 2022). Another potential direction is to analyze such adaptive preconditioners in the sampling context, which was discussed in a concurrent work by Bhattacharya and Jiang (2023), but the analysis in Wasserstein-2 distance is limited to a fixed preconditioner.

References

Agarwal, N., Bullins, B., Chen, X., Hazan, E., Singh, K., Zhang, C., and Zhang, Y. (2020). Efficient full-matrix adaptive regularization.

Ahn, S. W., Balan, A. K., and Welling, M. (2012). Bayesian posterior sampling via stochastic gradient fisher scoring. In *International Conference on Machine Learning*.

Akyildiz, Ö. D., Duffin, C., Sabanis, S., and Girolami, M. (2021). Statistical finite elements via langevin dynamics.

Anderson, B. D. O. (1982). Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12:313–326.

Bhattacharya, R. and Jiang, T. (2023). Fast sampling and inference via preconditioned langevin dynamics.

Dalalyan, A. S. (2017). Further and stronger analogy between sampling and optimization: Langevin monte carlo and gradient descent.

Dhariwal, P. and Nichol, A. (2021). Diffusion models beat gans on image synthesis. *ArXiv*, abs/2105.05233.

Dockhorn, T., Vahdat, A., and Kreis, K. (2022). Score-based generative modeling with critically-damped langevin diffusion.

Girolami, M. A. and Calderhead, B. (2011). Riemann manifold langevin and hamiltonian monte carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73.

Gupta, V., Koren, T., and Singer, Y. (2018). Shampoo: Preconditioned stochastic tensor optimization. In Dy, J. and Krause, A., editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1842–1850. PMLR.

Haario, H., Saksman, E., and Tamminen, J. (1999). Adaptive proposal distribution for random walk metropolis algorithm. *Computational Statistics*, 14(3):375–395.

Hartmann, M., Girolami, M., and Klami, A. (2022). Lagrangian manifold monte carlo on monge patches. In Camps-Valls, G., Ruiz, F. J. R., and Valera, I., editors, *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pages 4764–4781. PMLR.

Ho, J., Jain, A., and Abbeel, P. (2020). Denoising diffusion probabilistic models.

Kim, D., Shin, S., Song, K., Kang, W., and Moon, I.-C. (2022). Soft truncation: A universal training technique of score-based diffusion model for high precision score estimation. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., and Sabato, S., editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 11201–11228. PMLR.

Kim, S., Song, Q., and Liang, F. (2020). Stochastic gradient langevin dynamics algorithms with adaptive drifts. *ArXiv*, abs/2009.09535.

Kingma, D. P. and Ba, J. (2017). Adam: A method for stochastic optimization.

Krizhevsky, A. (2009). Learning multiple layers of features from tiny images. Technical report.

LeCun, Y., Cortes, C., and Burges, C. (2010). Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2.

Li, C., Chen, C., Carlson, D. E., and Carin, L. (2015). Preconditioned stochastic gradient langevin dynamics for deep neural networks. In *AAAI Conference on Artificial Intelligence*.

Ma, H., Zhang, L., Zhu, X., and Feng, J. (2022). Accelerating score-based generative models with preconditioned diffusion sampling. In *European Conference on Computer Vision*.

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

Patterson, S. and Teh, Y. W. (2013). Stochastic gradient riemannian langevin dynamics on the probability simplex. In *NIPS*.

Pidstrigach, J. (2022). Score-based generative models detect manifolds. *Advances in Neural Information Processing Systems*, 35:35852–35865.

Roberts, G. O. and Tweedie, R. L. (1996). Exponential convergence of langevin distributions and their discrete approximations. *Bernoulli*, pages 341–363.

Simsekli, U., Badeau, R., Cemgil, A. T., and Richard, G. (2016). Stochastic quasi-newton langevin monte carlo. In *International Conference on Machine Learning*.

Sohl-Dickstein, J., Weiss, E. A., Maheswaranathan, N., and Ganguli, S. (2015). Deep unsupervised learning using nonequilibrium thermodynamics. *CoRR*, abs/1503.03585.

Song, J., Meng, C., and Ermon, S. (2022). Denoising diffusion implicit models.

Song, Y. and Ermon, S. (2019). Generative modeling by estimating gradients of the data distribution. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. (2021). Score-based generative modeling through stochastic differential equations.

Tieleman, T. and Hinton, G. (2012). Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31.

Wang, X., Dinh, A.-D., Liu, D., and Xu, C. (2023). Boosting diffusion models with an adaptive momentum sampler.

Welling, M. and Teh, Y. W. (2011). Bayesian learning via stochastic gradient langevin dynamics. In *International Conference on Machine Learning*.

Wu, Z., Zhou, P., Kawaguchi, K., and Zhang, H. (2023). Fast diffusion model.

Xiao, H., Rasul, K., and Vollgraf, R. (2017). Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms.

Yu, H., Hartmann, M., Williams, B., and Klami, A. (2023). Scalable stochastic gradient riemannian langevin dynamics in non-diagonal metrics. *ArXiv*, abs/2303.05101.

Zhang, Q. and Chen, Y. (2023). Fast sampling of diffusion models with exponential integrator.

A pSGLD on $\mathcal{N}(0,1)$

In the special case where the target distribution is $\mathcal{N}(0,1)$, we have a simple form [1] of the update steps:

$$V_{t+1} = \beta V_t + (1-\beta)x_t^2$$
$$x_{t+1} = x_t - \gamma V_{t+1}^{-1/2} x_t + \sqrt{2\gamma}(V_{t+1})^{-1/4}\xi_{t+1}$$

where $V_0 = 0$. Those update steps can be easily re-formulated as a Markov chain $\mathbf{z}_t = (x_t, V_t)$ with transition kernel:

$$\mathbf{k}(\mathbf{z}_{t+1} \mid \mathbf{z}_t) = \delta_{\beta V_t + (1-\beta)x_t^2}(V_{t+1})\phi\left(\frac{x_{t+1}-\mu_t}{\sigma_t}\right) \tag{20}$$

where δ_x is a Dirac delta at x, $\mu_t = x_t - \gamma(\beta V_t + (1-\beta)x_t^2)^{-1/2}x_t$, $\sigma_t = \sqrt{2\gamma}(V_{t+1})^{-1/4}$, and ϕ is the density of standard Gaussian.

B pSGLD on diagonal Gaussians

We explore the convergence of the pSGLD with RMSProp in the case of a diagonal Gaussian $\mathcal{N}(\mu_q, \Sigma)$, where$\mu_q \in \mathbb{R}$ and $\Sigma \in \mathbb{R}^{d\times d}$ is a diagonal matrix. For notational convenience, we will denote $H = \Sigma^{-1}$.

Let $P_t = V_t^{-1/2} + \varepsilon I$ denote the pre-conditioner constructed using RMSProp at iteration t with $V_0 = 0, \varepsilon > 0$, and update rule

$$V_{t+1} = \beta V_t + (1-\beta)\text{diag}\left[H(x_t-\mu_q)(x_t-\mu_q)^T H^T\right]$$

where $\text{diag}(A)$ is the matrix constructed using the diagonal entries from A. One update of the algorithm can be written as:

$$x_{t+1} = x_t - \gamma P_{t+1} H(x_t - \mu_q) + \sqrt{2\gamma}P_{t+1}^{1/2}\xi_{t+1} \tag{21}$$

where $\xi_t \sim \mathcal{N}(0, I)$ is a d-dimensional standard Gaussian random variable with ξ_s, ξ_t independent for any $s, t > 0$ and $\gamma > 0$ a constant step size.

By elementary algebra, we may write down a non-recursive formula for the iterates[2]:

$$x_{t+1} - \mu_q = \left[\prod_{i=1}^{t+1}(I - \gamma P_i H)\right](x_0 - \mu_q) + \sqrt{2\gamma}\sum_{i=1}^{t+1}\left[\prod_{j=i+1}^{t+1}(I-\gamma P_j H)\right]P_i^{1/2}\xi_i \tag{22}$$

Taking expectation of the LHS of Equation (22), we have

$$\|\mu_{t+1} - \mu_q\|_2^2 \leq \left\|\mathbb{E}\left[\prod_{i=0}^{t}(I-\gamma P_i H)\right]\right\|_2^2 \|x_0 - \mu_q\|_2^2 + 2\gamma\sum_{i=0}^{t}\left\|\mathbb{E}\left[\prod_{j=i+1}^{t}(I-\gamma P_j H)P_i^{1/2}\xi_i\right]\right\|_2^2$$

using the triangle inequality. We can bound $\|\mu_{t+1} - \mu_q\|_2^2$ further using Jensen's inequality and the fact that spectral norm for matrices A, B satisfies $\|AB\|_2 \leq \|A\|_2\|B\|_2$. Denoting the largest eigenvalue of $(I - \gamma P_i H)$ as L_i and that of $P_i^{1/2}$ as K_i, we have:

$$\|\mu_{t+1} - \mu_q\|_2^2 \leq \mathbb{E}\left[\prod_{i=0}^{t}|L_i|^2\right]\|x_0 - \mu_q\|_2^2 + 2\gamma\sum_{i=0}^{t}\mathbb{E}\left[|K_i|^2\prod_{j=i+1}^{t}|L_j|^2\|\xi_i\|_2^2\right]$$

In the case where Σ and $H = \Sigma^{-1}$ are diagonal matrices, we may compute the entries of V_t explicitly. Let $\sigma_k^2 := \Sigma_{i,i}$, $\mu = (\mu^{(1)}, \ldots, \mu^{(d)})$, and $x_t = (x_t^{(1)} \ldots, x_t^{(d)})$, then the k^{th} diagonal entry of V_t is:

[1] We ignore the constants ε

[2] We adpot the convention that if $m > n$, $\prod_{i=m}^{n} A_i = I$

$$(V_t)_{(k,k)} = (1-\beta)\sum_{i=1}^{t}\beta^{t-i}\sigma_k^4(x_i^{(k)}-\mu^{(k)})^2 \tag{23}$$

Thus, we can estimate the condition number $|L_t|$ by considering:

$$\max_{1\le k\le d}\left|1-\frac{1}{\sqrt{(1-\beta)\beta^t}\sqrt{\sum_{i=1}^{t}\beta^{-i}(x_i^{(k)}-\mu^{(k)})^2}}-\varepsilon\sigma_k^{-2}\right| \tag{24}$$

This expression provides some insights into how pSGLD on diagonal Gaussians behave : first, for a fixed k, the past deviations from the target mean dictates how fast the quantity decays; secondly, for large variance σ_k , the quantity tends to be small, whereas for smaller σ_k , the quantity may still remain large.

C Details of experiments on tractable distributions

For experiments on the tractable distributions Section 5, a grid search is used to find the best hyperparameters over 10 runs starting at different initial values drawn from a $\mathcal{N}(0, I)$ distribution.

We report the hyperparameters below. For forgetful RMSProp samplers, we report them separately, as they have extra sets of forget times as hyperparameters.

	ULA	PULA	RMSProp	Adam	Monge
Gaussian $\mathcal{N}(0,\Sigma_1)$	$\gamma=10^{-3}$	$\gamma=0.1$	$\gamma=0.085,\beta=0.999$	$\gamma=0.05,\beta_1=0.8,\beta_2=0.999$	$\gamma=5.3\times10^{-3},\lambda=0.5,\alpha^2=10^{-3}$
Gaussian $\mathcal{N}(0,\Sigma_2)$	$\gamma=1.5$	$\gamma=0.12$	$\gamma=1.92,\beta=0.999$	$\gamma=1.21,\beta_1=0.8,\beta_2=0.999$	$\gamma=1.0,\lambda=0.6,\alpha^2=10^{-5}$
Gaussian $\mathcal{N}(0,\Sigma_3)$	$\gamma=10^{-3}$	$\gamma=0.1$	$\gamma=0.1,\beta=0.95$	$\gamma=0.5,\beta_1=0.8,\beta_2=0.9$	$\gamma=2.7\times10^{-3},\lambda=0.5,\alpha^2=10^{-4}$
Twisted Gaussian $f_{0.05,100}$	$\gamma=0.41$	NA	$\gamma=1.0,\beta=0.95$	$\gamma=0.5,\beta_1=0.95,\beta_2=0.9$	$\gamma=0.57,\lambda=0.8,\alpha^2=10^{-2}$

Table 2: Hyperparameters of samplers on tractable distributions. PULA is not applicable for the Twisted Gaussian.

The forget times for Forgetful RMSProp 1 (Algorithm 3) is constructed as a sequece of evenly spaced positive integers

$$S(M,k)=\{a: 1\le a\le M, a-1\equiv 0 \pmod k\}$$

where N is the number of iterations. In addition, we denote the window size of Forgetful RMSProp 2 (Algorithm 4) as W.

	Forgetful RMSProp 1	Forgetful RMSProp 2
Gaussian $\mathcal{N}(0,\Sigma_1)$	$\gamma=0.04,\beta=0.999,M=1000,k=50$	$\gamma=0.075,\beta=0.999,W=500$
Gaussian $\mathcal{N}(0,\Sigma_2)$	$\gamma=0.74,\beta=0.999,M=10000,k=500$	$\gamma=2.16,\beta=0.999,W=500$
Gaussian $\mathcal{N}(0,\Sigma_3)$	$\gamma=0.095,\beta=0.99,M=1000,k=50$	$\gamma=0.025,\beta=0.999,W=50$
Twisted Gaussian $f_{0.05,100}$	$\gamma=0.81,\beta=0.95,M=10000,k=500$	$\gamma=0.2,\beta=0.999,W=50$

Table 3: Hyperparameters of forgetful samplers on tractable distributions.

D Details of experiments on SGMs

We trained the SGM for 100 epochs with a batch size of 32 on both the MNIST and Fashion MNIST dataset. For experiments on CIFAR-10, we used the weights provided in Song et al. (2021).

D.1 Network Architecture

The network architecture is exactly the same as the network used in the tutorial provided in Song et al. (2021).

D.2 Derivation of the VE SDE in experiments

The reverse SDE of the VE-SDE in Section 6 is:

$$\mathrm{d}\mathbf{x} = -\sigma^{2t}\nabla_{\mathbf{x}}\log p_t(\mathbf{x})dt + \sigma^t\mathrm{d}\bar{\mathbf{W}} \tag{25}$$

To obtain the perturbation kernel $p_{0T}(\mathbf{x}(t)|\mathbf{x}(0))$, note that a general VE-SDE is of the form:

$$\mathrm{d}\mathbf{x} = \sqrt{\frac{d[g^2(t)]}{dt}}\mathrm{d}\mathbf{W} \tag{26}$$

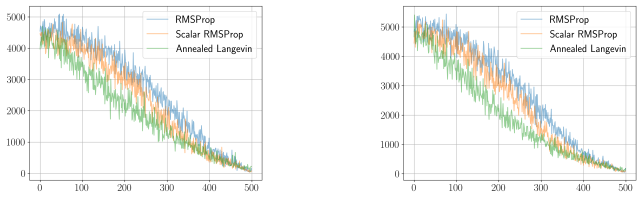
with $p_{0T}(\mathbf{x}(t)|\mathbf{x}(0)) = \mathcal{N}(\mathbf{x}(t); \mathbf{x}(0), [g^2(t) - g^2(0)]\mathbf{I})$.

Solving $\sqrt{\frac{d[g^2(t)]}{dt}} = \sigma^t$ yields $g(t) = \frac{1}{\sqrt{2\log\sigma}}\sigma^t$. Thus the perturbation kernel is:

$$p_{0T}(\mathbf{x}(t)|\mathbf{x}(0)) = \mathcal{N}(\mathbf{x}(t); \mathbf{x}(0), \frac{1}{2\log\sigma}(\sigma^{2t} - 1)\mathbf{I}) \tag{27}$$

D.3 Loss and Score

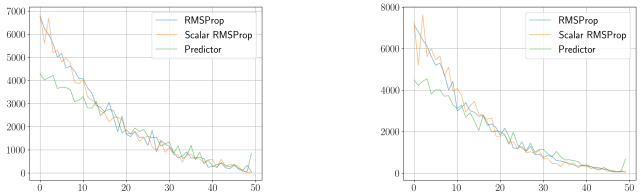
In this section, we provide extra quantitative measures for the convergence of RMSProp corrector and predictors through the visualisation of loss of the score-estimating network and L^2-norm of the estimated score.



(a) MNIST dataset (b) Fahsion MNIST dataset

Figure 14: Loss during corrector sampling evaluated at each step

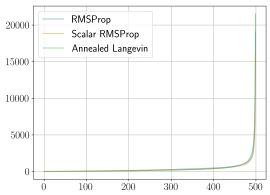
As investigated by Song and Ermon (2019) and Zhang and Chen (2023), the loss of the score-estimating network is large when the samples are drawn from a low density region that is far from the data manifold, thus we argue that visualising how the loss of the network evolves can be helpful to comparing the behaviour of the sampling algorithms.



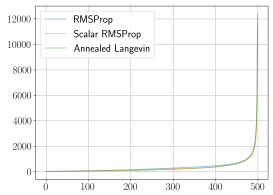
(a) MNIST dataset (b) Fahsion MNIST dataset

Figure 15: Loss during predictor sampling evaluated at each step

Another quantitative measure we use is the L^2-norm of the estimated score. As shown by Pidstrigach (2022), under certain assumptions, the SGM is not able to generalize when the difference between the estimated score $\mathbf{s}_\theta(\mathbf{x}, t)$ and true score is bounded. While this is not straightforward to check without a ground truth score, we plot the values of $\|\mathbf{s}_\theta(\mathbf{x}, t)\|_2$ to roughly illustrate this blow-up phenomenon.



(a) MNIST dataset



(b) Fahsion MNIST dataset

Figure 16: $\|\mathbf{s}_\theta(\mathbf{x}, t)\|_2$ during corrector sampling

E Pseudocode

We include the psuedocode for the RMSProp predictor (Equation (16)). Here we are running the sampler from time T to 0 to recover the denoised sample $\mathbf{x}_0$.

Algorithm 6 RMSProp Predictor

Require:
T: the final time for discretization
N: number of discretization steps for the reverse-time SDE
ϵ: a small constant for numerical stability
$\{t_i\}_{1 \leq i \leq N}$: a sequence of times evenly spaced in $(\epsilon, T]$
β: a decay parameter in $(0, 1)$ controlling the moving average
λ: a small constant to control the extreme values of the moving average

Initialize $\mathbf{x}_T \sim p_{0T}(\mathbf{x}(t)|\mathbf{x}(0))$, set step-size $\Delta t \leftarrow \frac{1}{N}$, and moving average $V_N \leftarrow 0$
for $i = T - 1$ to 0 **do**
  Compute score $\mathbf{s}_{i+1} \leftarrow \mathbf{s}_\theta(\mathbf{x}_{i+1}, t_{i+1})$
  Compute drift coefficient $g_{i+1} \leftarrow g(t_{i+1})$
  Update $V_i \leftarrow \beta V_{i+1} + (1 - \beta)\text{diag}(\mathbf{s}_{i+1}\mathbf{s}_{i+1}^T)$
  Construct $G_i \leftarrow (V_i^{-1/2} + \lambda \mathbf{I})$
  $\mathbf{x}_i' \leftarrow \mathbf{x}_{i+1} - [\mathbf{f}(\mathbf{x}_{i+1}, t) - g_{i+1}^2 G_{i+1}\mathbf{s}_{i+1}]\Delta t$
  $\mathbf{x}_i' = \mathbf{x}_{i+1} + g_{i+1}G_{i+1}^{1/2}\mathbf{z}_{i+1}\sqrt{\Delta t}$, where $\mathbf{z}_{i+1} \sim \mathcal{N}(0, I)$
end for
return $\mathbf{x}_0'$