Coin Sampling: Gradient-Based Bayesian Inference without Learning Rates

Louis Sharrock, Christopher Nemeth

Lancaster University

ICML International Conference On Machine Learning

1. Introduction

Motivation

- **Sampling from an unnormalised probability distribution** $\pi(\mathrm{d}x)$ on $\mathbb{R}^d$, with density

$$\boxed{\pi(x) \propto e^{-U(x)}},$$

is a central problem in computational statistics and machine learning.
- Many existing methods such as **Langevin Monte Carlo** (LMC) and **Stein variational gradient descent** (SVGD) [1] depend on a **learning rate** γ, which must be **carefully tuned** to ensure convergence to the target distribution π at a suitable rate.

Contributions

- We introduce coin sampling, a general framework for **gradient-based Bayesian inference** which is entirely learning-rate free.
- We propose **coin sampling analogues** of several existing particle-based sampling algorithms, including **Stein variational gradient descent** (SVGD), **kernel Stein discrepancy descent** (KSDD) [2], and **Laplacian adjusted Wasserstein gradient descent** (LAWGD) [3].
- We illustrate the performance of our approach on a range of numerical examples. Our method achieves **comparable performance** to existing particle-based sampling algorithms with **no need to tune a learning rate**.

2. Background: Parameter-Free Optimisation

Suppose we were interested in solving the optimisation problem

$$x^* = \underset{x\in\mathbb{R}^d}{\arg\min} f(x).$$

In [4], Orabona and Pal introduced a **parameter-free** method for solving this optimisation problem based on **coin betting**.

- Consider a gambler who bets on a series of coin flips.
- The gambler starts with initial wealth $w_0 > 0$, and **bets on the outcomes of coin flips** $c_t \in \{-1, 1\}$, where $+1$ denotes heads and -1 denotes tails.
- The gambler bets $x_t \in \mathbb{R}$, where $\mathrm{sign}(x_t) \in \{-1, 1\}$ denotes **whether the bet is on heads or tails**, and $|x_t| \in \mathbb{R}$ denotes the **size of the bet**.
- The **wealth** w_t of the gambler thus accumulates as

$$w_t = w_0 + \sum_{s=1}^{t} c_s x_s.$$

- We will assume the gambler's bets satisfy $x_t = \beta_t w_{t-1}$, where $\beta_t \in [-1, 1]$ is a **betting fraction**, given by $\beta_t = t^{-1}\sum_{s=1}^{t-1} c_s$.
- The **sequence of bets** made by the gambler is thus given by

$$\boxed{x_t = \frac{\sum_{s=1}^{t-1} c_s}{t}\left(w_0 + \sum_{s=1}^{t-1} c_s x_s\right)}.$$

- Remarkably, if we consider a betting game in which $c_t = -\nabla f(x_t)$, then the **average of the bets** $\frac{1}{T}\sum_{t=1}^{T} x_t$ **converges** to $x^* = \arg\min_{x\in\mathbb{R}^d} f(x)$ at a **rate determined by the betting strategy** [4].
- Moreover, this approach is **completely learning-rate free**!

3. Sampling as Optimisation

To extend the coin betting framework to our setting, we will leverage the view of **sampling** as an **optimisation problem on the space of probability measures**:

$$\boxed{\pi = \underset{\mu\in\mathscr{P}_2(\mathbb{R}^d)}{\arg\min}\, \mathscr{F}(\mu)},$$

where $\mathscr{F} : \mathscr{P}(\mathbb{R}^d) \to \mathbb{R}$ is a **dissimilarity functional** uniquely minimised at π. A natural solution to this problem is to simulate a discretisation of the **Wasserstein gradient flow** of $\mathscr{F}$ over $(\mathscr{P}_2(\mathbb{R}^d), W_2)$, namely,

$$\partial_t \mu_t + \nabla \cdot (v_t \mu_t) = 0, \quad v_t = -\nabla_{W_2}\mathscr{F}(\mu_t),$$

where $\nabla_{W_2}\mathscr{F}(\mu)$ denotes the **Wasserstein gradient** of $\mathscr{F}$ at μ.

4. Coin Sampling

We take a different approach, based on **coin betting**:

- Consider a gambler with initial wealth $w_0 > 0$. We suppose the **gambler bets** $x_t - x_0$ on **outcomes** $c_t \in [-L, L]$, where $x_0 \sim \mu_0$ for some $\mu_0 \in \mathscr{P}_2(\mathbb{R}^d)$. We assume the bets satisfy $x_t - x_0 = \beta_t w_{t-1}$, and that $\beta_t = \frac{1}{Lt}\sum_{s=1}^{t-1} c_s$.
- Let $\varphi_t : \mathbb{R}^d \to \mathbb{R}^d$ denote the functions which map $\varphi_t : x_0 \mapsto x_t$. We can then define a **sequence of measures** via $\mu_t = (\varphi_t)_{\#}\mu_0$, so that $x_t \sim \mu_t$.
- Inspired by [4], and the view of sampling as optimisation, we will consider a **betting game** with $c_t = -\frac{1}{L}\nabla_{W_2}\mathscr{F}(\mu_t)(x_t)$. The **gambler's bets** are thus

$$\boxed{x_t - x_0 = -\frac{\sum_{s=1}^{t-1}\nabla_{W_2}\mathscr{F}(\mu_s)(x_s)}{Lt}\left(w_0 - \frac{1}{L}\sum_{s=1}^{t-1}\langle\nabla_{W_2}\mathscr{F}(\mu_s)(x_s), x_s - x_0\rangle\right)}.$$

- In this case, under certain conditions, it is possible to show that $\mathscr{F}(\frac{1}{T}\sum_{t=1}^{T}\mu_t) \to \mathscr{F}(\pi)$, where $\mu_t = \mathrm{Law}(x_t)$.

The updates above depend on the **unknown measures** $(\mu_t)_{t\in\mathbb{N}}$, so in practice we will use a **particle-based approximation.** For different choices of $\mathscr{F}$ and different approximations of $\nabla_{W_2}\mathscr{F}(\mu)$, this results in **learning-rate free analogues** of several **existing particle-based algorithms** (e.g., SVGD, KSDD, LAWGD).

Coin SVGD. Inspired by SVGD [1], suppose we let $\mathscr{F}(\mu) = \mathrm{KL}(\mu||\pi)$, and that we replace $\nabla_{W_2}\mathscr{F}(\mu)$ by $P_\mu\nabla_{W_2}\mathscr{F}(\mu)$, where P_μ is the integral operator $P_\mu f(x) = \int k(x,y)f(y)\mathrm{d}y$. Integrating by parts, we then have

$$P_\mu\nabla_{W_2}\mathscr{F}(\mu_s)(x) := P_\mu\nabla\log\left(\frac{\mu_s}{\pi}\right)(x) = \int\left[k(x,y)\nabla U(y) - \nabla_2 k(x,y)\right]\mu_s(\mathrm{d}y),$$

which we can **easily approximate using samples** $x_s^i \sim \mu_s$. This suggests the following **particle-based approximation**. Let $(x_0^i)_{i=1}^N \sim \mu_0$, and $(w_0^i)_{i=1}^N \in \mathbb{R}_+$. Then, writing $\hat{\mu}_s^N = \frac{1}{N}\sum_{j=1}^N \delta_{x_s^j}$, update the particles according to

$$\boxed{x_t^i = x_0^i - \frac{\sum_{s=1}^{t-1} P_{\hat{\mu}_s^N}\nabla\log\left(\frac{\hat{\mu}_s^N}{\pi}\right)(x_s^i)}{t} \times \left(w_0^i - \frac{1}{L}\sum_{s=1}^{t-1}\langle P_{\hat{\mu}_s^N}\nabla\log\left(\frac{\hat{\mu}_s^N}{\pi}\right)(x_s^i), x_s^i - x_0^i\rangle\right)}.$$

5. Numerical Experiments

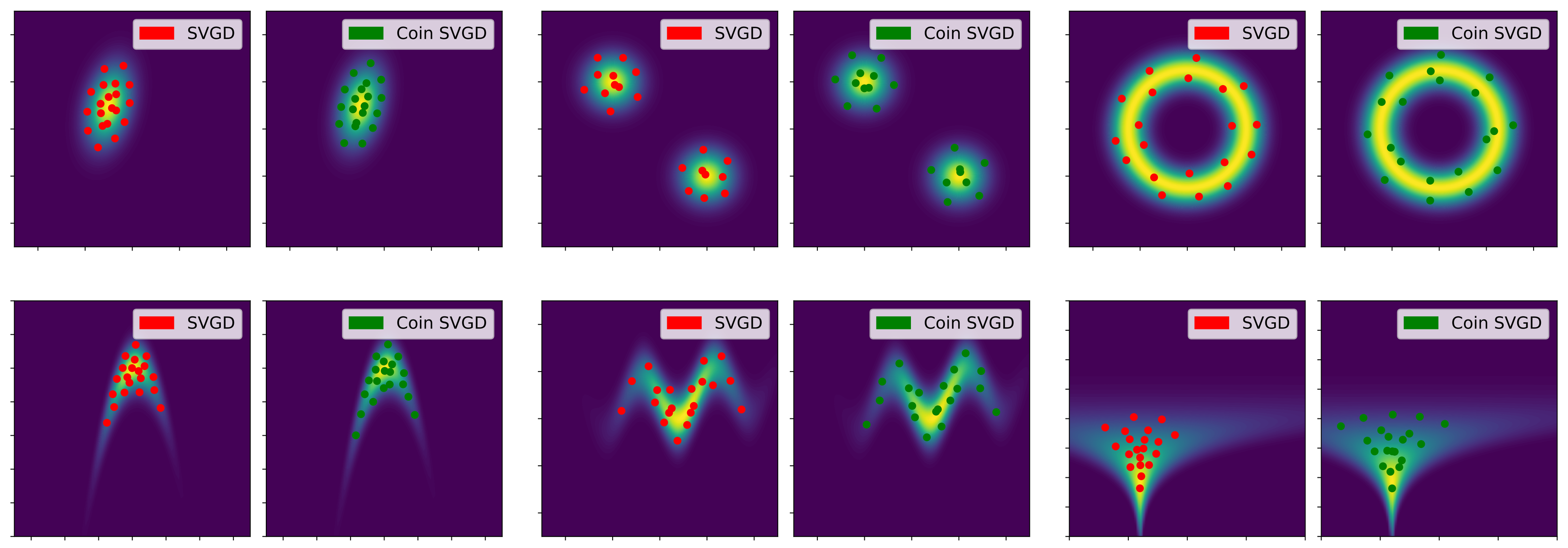
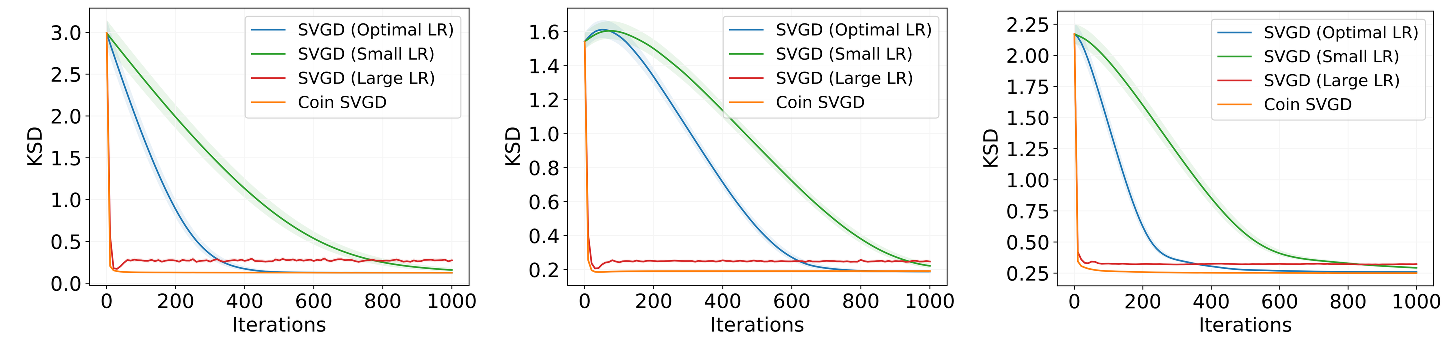
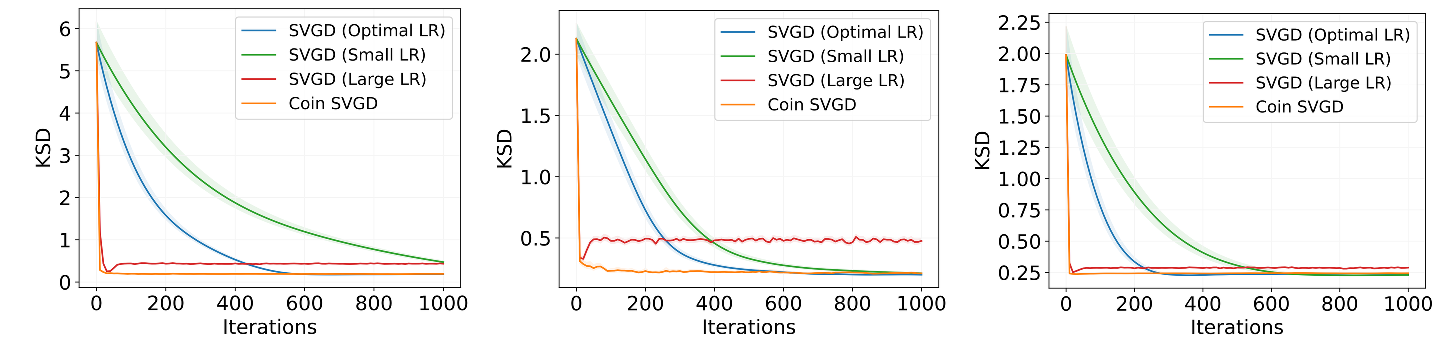


Fig 1. Toy Examples. Samples generated by Coin SVGD and SVGD.



(a) Gaussian (b) Gaussian Mixture (c) Donut



(d) Banana (e) Squiggle (f) Funnel

Fig 2. Toy Examples. KSD vs Iterations for Coin SVGD and SVGD.

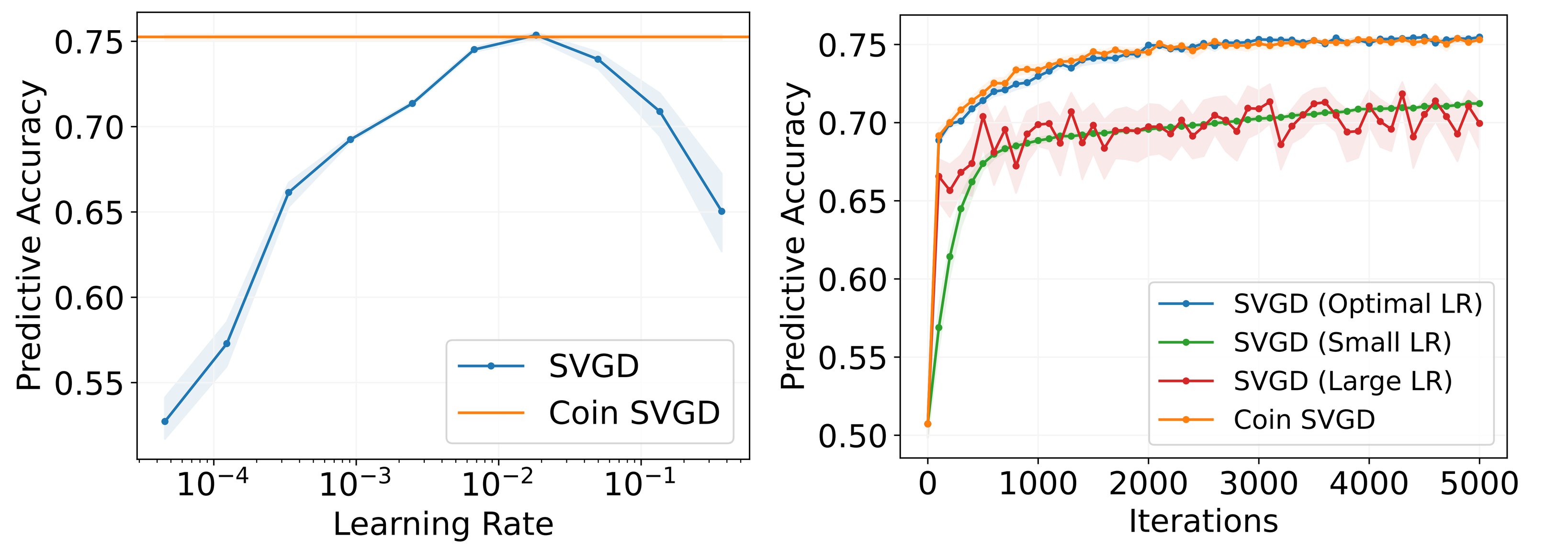
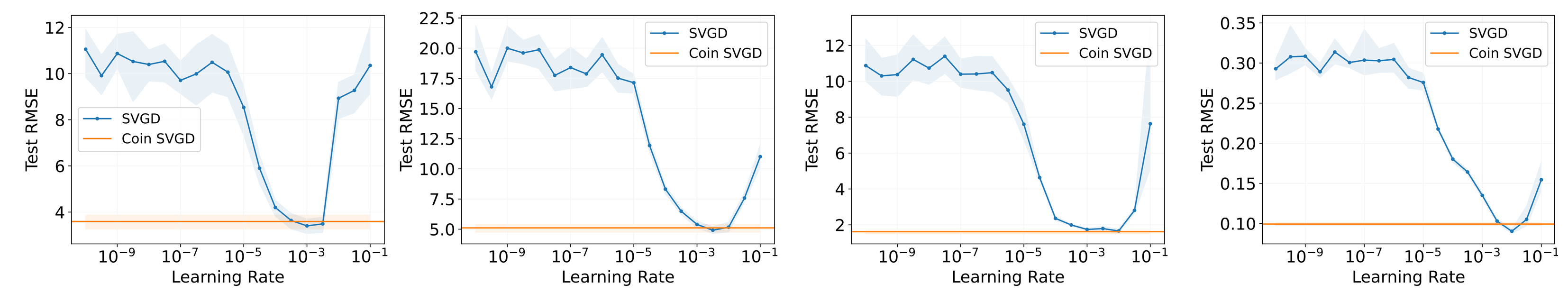


Fig 3. Bayesian Logistic Regression. Test accuracy for Coin SVGD and SVGD.



(a) Boston (b) Concrete (c) Energy (d) Kin8nm

Fig 4. Bayesian Neural Network. Test RMSE for Coin SVGD and SVGD.

6. References

[1] Q. Liu and D. Wang. Stein Variational Gradient Descent: A General Purpose Bayesian Inference Algorithm. *NeurIPS 2016.*

[2] A Korba et al. Kernel Stein Discrepancy Descent. *ICML 2021.*

[3] S. Chewi et al. SVGD as a kernelized Wasserstein gradient flow of the chi-squared divergence. *NeurIPS 2020.*

[4] F. Orabona and D. Pal. Coin Betting and Parameter-Free Online Learning. *NeurIPS 2016.*

7. Code

Code and more results available on **GitHub**: