Preferential data subsampling in stochastic gradient MCMC

Chris Nemeth, Lancaster University

Joint work with: Srshti Putcha and Paul Fearnhead

Engineering and Physical Sciences Research Council

Mathematics & Statistics | Lancaster University

Background and Notation

Background and Notation

The **posterior density** for $\theta \in \mathbb{R}^d$ given data $\mathbf{y} = \{y_1, y_2, \ldots, y_N\}$, up to a constant of proportionality, is

$$\pi(\theta) := p(\theta \,|\, \mathbf{y}) \propto p(\theta) \prod_{i=1}^{N} p(y_i \,|\, \theta),$$

where $p(\theta)$ is a **prior** for the parameters θ.

Background and Notation

The **posterior density** for $\theta \in \mathbb{R}^d$ given data $\mathbf{y} = \{y_1, y_2, \ldots, y_N\}$, up to a constant of proportionality, is

$$\pi(\theta) := p(\theta \,|\, \mathbf{y}) \propto p(\theta) \prod_{i=1}^{N} p(y_i \,|\, \theta),$$

where $p(\theta)$ is a **prior** for the parameters θ.

For notational convenience we define $f_i(\theta) = -\log p(y_i \,|\, \theta)$ for $i = 1, \ldots, N$, where $f_0(\theta) = -\log p(\theta)$ and $f(\theta) = \sum_{i=0}^{N} f_i(\theta)$ is the **potential function**.

In this setting, the posterior density can be rewritten as,

$$\pi(\theta) \propto \exp(-f(\theta))$$

Langevin diffusion

Langevin diffusion

The Langevin diffusion, $\theta(t)$, is defined by the **stochastic differential equation** (SDE),

$$d\theta(t) = -\frac{1}{2}\nabla f(\theta(t))dt + dB_t,$$

where $\nabla f(\theta(t))dt$ is a **drift term** and B_t denotes a d-dimensional Wiener process.

Langevin diffusion

The Langevin diffusion, $\theta(t)$, is defined by the **stochastic differential equation** (SDE),

$$d\theta(t) = -\frac{1}{2}\nabla f(\theta(t))dt + dB_t,$$

where $\nabla f(\theta(t))dt$ is a **drift term** and B_t denotes a d-dimensional Wiener process.

Under certain regularity conditions, the stationary distribution of this diffusion is the posterior π. In practice, we need to discretise the SDE in order to simulate from it and this introduces error.

Langevin diffusion

The Langevin diffusion, $\theta(t)$, is defined by the **stochastic differential equation** (SDE),

$$d\theta(t) = -\frac{1}{2}\nabla f(\theta(t))dt + dB_t,$$

where $\nabla f(\theta(t))dt$ is a **drift term** and B_t denotes a d-dimensional Wiener process.

Under certain regularity conditions, the stationary distribution of this diffusion is the posterior π. In practice, we need to discretise the SDE in order to simulate from it and this introduces error.

For a small step-size $\epsilon > 0$, the Langevin diffusion can be approximated by the **unadjusted Langevin algorithm** (ULA),

$$\theta^{(t+1)} = \theta^{(t)} - \frac{\epsilon}{2}\nabla f\left(\theta^{(t)}\right) + \sqrt{\epsilon}\,\eta^{(t)},$$

where the noise $\eta^{(t)} \sim \mathcal{N}_d(\mathbf{0}, I_{d\times d})$ is drawn independently at each update.

Stochastic gradient Langevin Dynamics

Stochastic gradient Langevin Dynamics

The stochastic gradient Langevin dynamics (SGLD) algorithm improves the per-iteration computational burden of MCMC by replacing the full-data gradient $\nabla f(\theta)$ with an unbiased estimate $\nabla \hat{f}(\theta)$.

If the full-data gradient is

$$\nabla f(\theta) = \nabla f_0(\theta) + \sum_{i=1}^{N} \nabla f_i(\theta)$$

then an unbiased estimate is given by

$$\nabla \hat{f}(\theta) = \nabla f_0(\theta) + \frac{N}{n} \sum_{i \in \mathcal{S}} \nabla f_i(\theta),$$

where $\mathcal{S}$ is a subset of $\{1, \ldots, N\}$ with $|\mathcal{S}| = n$, $(n \ll N)$.

Stochastic gradient Langevin Dynamics

The stochastic gradient Langevin dynamics (SGLD) algorithm improves the per-iteration computational burden of MCMC by replacing the full-data gradient $\nabla f(\theta)$ with an unbiased estimate $\nabla \hat{f}(\theta)$.

If the full-data gradient is

$$\nabla f(\theta) = \nabla f_0(\theta) + \sum_{i=1}^{N} \nabla f_i(\theta)$$

then an unbiased estimate is given by

$$\nabla \hat{f}(\theta) = \nabla f_0(\theta) + \frac{N}{n} \sum_{i \in \mathcal{S}} \nabla f_i(\theta),$$

where $\mathcal{S}$ is a subset of $\{1, \ldots, N\}$ with $|\mathcal{S}| = n$, $(n \ll N)$.

A single update of the vanilla SGLD algorithm is thus given by,

$$\theta^{(t+1)} = \theta^{(t)} - \frac{\epsilon^{(t)}}{2} \cdot \nabla \hat{f}(\theta^{(t)}) + \xi^{(t)}, \qquad \xi^{(t)} \sim \mathcal{N}_d(0, \epsilon^{(t)} I_{d \times d}),$$

where $\{\epsilon^{(t)}\}$ corresponds to a schedule of step-sizes which may be fixed or decreasing.

Controlling the variance of the gradients

Controlling the variance of the gradients

Dalalyan & Karagulyan (2019) Thm.3

Let $\tilde{\pi}_t$ be the posterior distribution after t iterations of the SGLD algorithm and be its distribution. If π satisfies the strongly log-concave assumption then

$$W_2(\tilde{\pi}_t, \pi) \leq (1 - C_0\epsilon)^t W_2(\tilde{\pi}_0, \pi) + C_1(\epsilon d)^{1/2} + C_2(\epsilon d)^{1/2}\sigma,$$

where C_0, C_1 and C_2 are constants and σ^2 is the variance of $\nabla \hat{f}(\theta)$ and $W_2(\cdot, \cdot)$ is the 2-Wasserstein distance.

Controlling the variance of the gradients

Dalalyan & Karagulyan (2019) Thm.3

Let $\tilde{\pi}_t$ be the posterior distribution after t iterations of the SGLD algorithm and be its distribution. If π satisfies the strongly log-concave assumption then

$$W_2(\tilde{\pi}_t, \pi) \leq (1 - C_0\epsilon)^t W_2(\tilde{\pi}_0, \pi) + C_1(\epsilon d)^{1/2} + C_2(\epsilon d)^{1/2}\sigma,$$

where C_0, C_1 and C_2 are constants and σ^2 is the variance of $\nabla \hat{f}(\theta)$ and $W_2(\cdot, \cdot)$ is the 2-Wasserstein distance.

Under SGLD, the variance σ^2 is of order N^2/n. Therefore the bound on $W_2(\tilde{\pi}_k, \pi)$ is dominated by a term of order $N(\epsilon d/n)^{1/2}$.

Controlling the variance of the gradients

Dalalyan & Karagulyan (2019) Thm.3

Let $\tilde{\pi}_t$ be the posterior distribution after t iterations of the SGLD algorithm and be its distribution. If π satisfies the strongly log-concave assumption then

$$W_2(\tilde{\pi}_t, \pi) \leq (1 - C_0\epsilon)^t W_2(\tilde{\pi}_0, \pi) + C_1(\epsilon d)^{1/2} + C_2(\epsilon d)^{1/2}\sigma,$$

where C_0, C_1 and C_2 are constants and σ^2 is the variance of $\nabla \hat{f}(\theta)$ and $W_2(\cdot, \cdot)$ is the 2-Wasserstein distance.

Under SGLD, the variance σ^2 is of order N^2/n. Therefore the bound on $W_2(\tilde{\pi}_k, \pi)$ is dominated by a term of order $N(\epsilon d/n)^{1/2}$.

The quality of the posterior approximation is directly tied to the variance in the gradient estimate. A similar story follows for **minibatching the Metropolis-Hastings ratio** (Quiroz et al. 2018) and **piecewise deterministic Markov process** (PDMP) algorithms, such as the zig-zag sampler (Bierkens et al. 2019).

SGLD with control variates

SGLD with control variates

Let $\hat{\theta}$ be a fixed value of the parameter, typically chosen to be close to the *maximum a posteriori* value of the target posterior density. The **control variate gradient estimator** takes the form (Baker et al. 2019),

$$\nabla \hat{f}_{cv}(\theta^{(t)}) = \left[\nabla f(\hat{\theta}) + \nabla f_0\left(\theta^{(t)}\right) - \nabla f_0(\hat{\theta})\right] + \frac{N}{n} \sum_{i \in \mathcal{S}} \left[\nabla f_i\left(\theta^{(t)}\right) - \nabla f_i(\hat{\theta})\right].$$

SGLD with control variates

Let $\hat{\theta}$ be a fixed value of the parameter, typically chosen to be close to the *maximum a posteriori* value of the target posterior density. The **control variate gradient estimator** takes the form (Baker et al. 2019),

$$\nabla \hat{f}_{cv}(\theta^{(t)}) = \left[\nabla f(\hat{\theta}) + \nabla f_0\left(\theta^{(t)}\right) - \nabla f_0(\hat{\theta})\right] + \frac{N}{n} \sum_{i \in \mathcal{S}} \left[\nabla f_i\left(\theta^{(t)}\right) - \nabla f_i(\hat{\theta})\right].$$

When $\theta^{(t)}$ is close to $\hat{\theta}$, the variance of the gradient estimator will be small.

SGLD with control variates

Let $\hat{\theta}$ be a fixed value of the parameter, typically chosen to be close to the *maximum a posteriori* value of the target posterior density. The **control variate gradient estimator** takes the form (Baker et al. 2019),

$$\nabla \hat{f}_{cv}(\theta^{(t)}) = \left[\nabla f(\hat{\theta}) + \nabla f_0\left(\theta^{(t)}\right) - \nabla f_0(\hat{\theta})\right] + \frac{N}{n} \sum_{i \in \mathcal{S}} \left[\nabla f_i\left(\theta^{(t)}\right) - \nabla f_i(\hat{\theta})\right].$$

When $\theta^{(t)}$ is close to $\hat{\theta}$, the variance of the gradient estimator will be small.

The SGLD-CV algorithm is the same as SGLD except we replace $\nabla \hat{f}(\theta)$ with $\nabla \hat{f}_{cv}(\theta)$.

SGLD with control variates

Let $\hat{\theta}$ be a fixed value of the parameter, typically chosen to be close to the *maximum a posteriori* value of the target posterior density. The **control variate gradient estimator** takes the form (Baker et al. 2019),

$$\nabla \hat{f}_{cv}(\theta^{(t)}) = \left[\nabla f(\hat{\theta}) + \nabla f_0\left(\theta^{(t)}\right) - \nabla f_0(\hat{\theta})\right] + \frac{N}{n} \sum_{i \in \mathcal{S}} \left[\nabla f_i\left(\theta^{(t)}\right) - \nabla f_i(\hat{\theta})\right].$$

When $\theta^{(t)}$ is close to $\hat{\theta}$, the variance of the gradient estimator will be small.

The SGLD-CV algorithm is the same as SGLD except we replace $\nabla \hat{f}(\theta)$ with $\nabla \hat{f}_{cv}(\theta)$.

Implementing the SGLD-CV estimator involves a **one-off pre-processing step** to find $\hat{\theta}$, which is typically done using stochastic gradient descent (SGD). The gradient terms $\nabla f_i\left(\hat{\theta}\right)$ are calculated and stored. While these steps are both $O(N)$ in computational cost, the optimisation step to find the mode can replace the typical burn-in phase of the SGLD chain.

Simple Gaussian model

Simulated $N = 10{,}000$ data point from,

$$y_i \sim \mathcal{N}(\theta, 1)$$

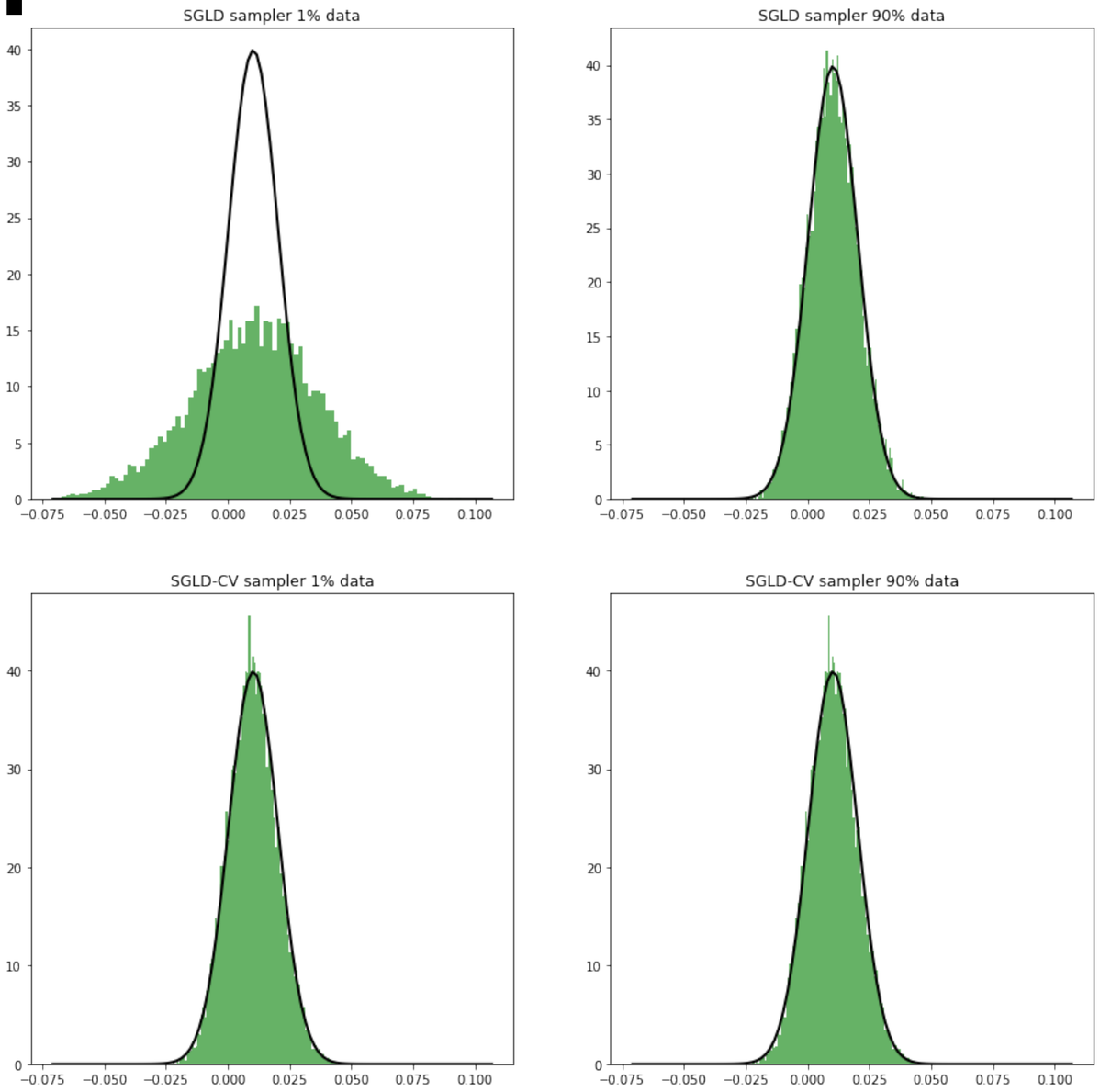
with $\theta = 0$.

Set the prior to be $\mathcal{N}(0, 10)$ then the posterior is given in closed-form

$$\theta \,|\, y \sim \mathcal{N}(\mu, \sigma^2).$$

Run SGLD and SGLD-CV with data subsample sizes of 1% and 90%.

https://tinyurl.com/2tdh97z4



Gaussian posterior samples

Simple Gaussian model

Simulated $N = 10{,}000$ data point from,

$$y_i \sim \mathcal{N}(\theta, 1)$$

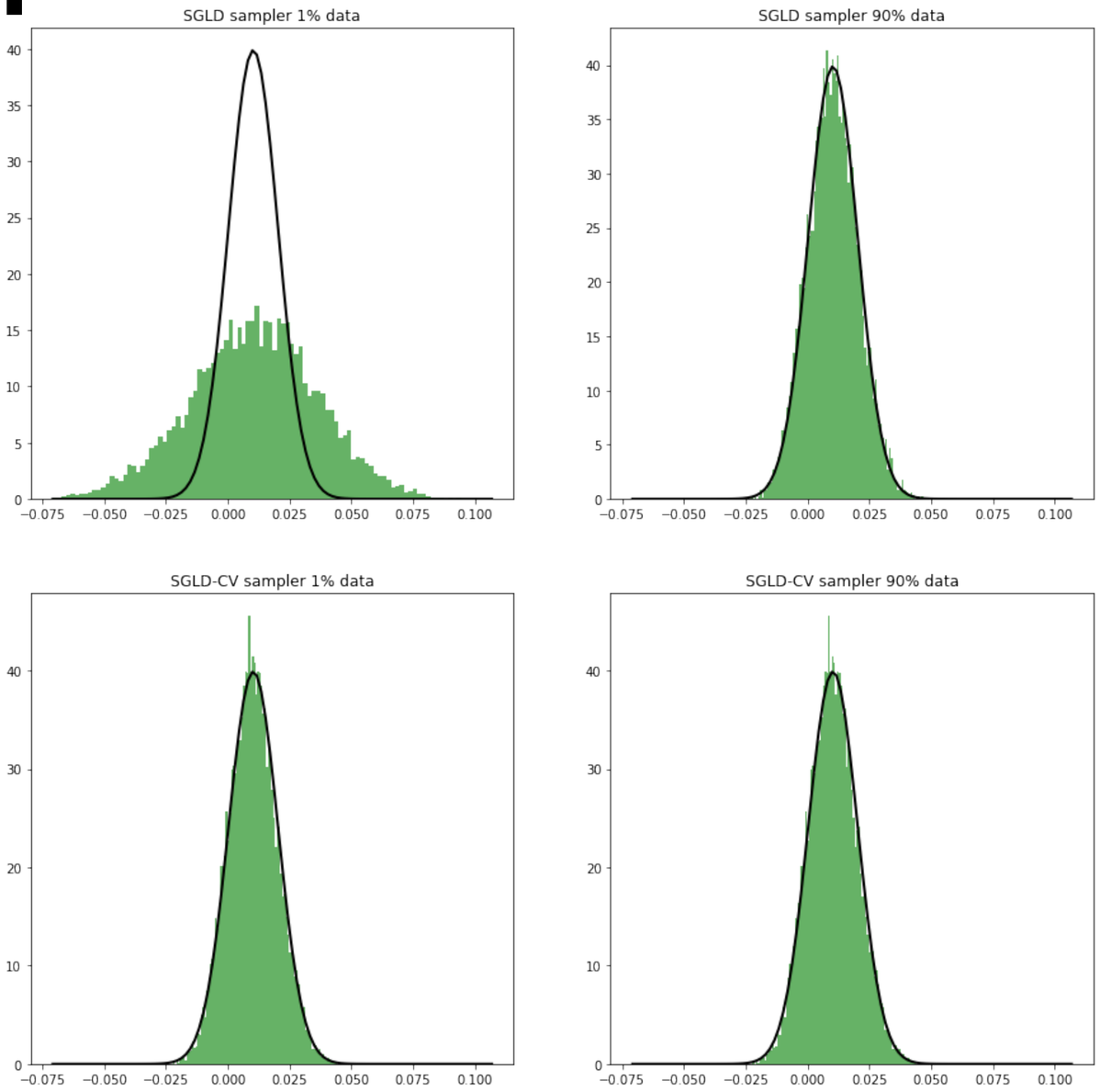
with $\theta = 0$.

Set the prior to be $\mathcal{N}(0, 10)$ then the posterior is given in closed-form

$$\theta \,|\, y \sim \mathcal{N}(\mu, \sigma^2).$$

Run SGLD and SGLD-CV with data subsample sizes of 1% and 90%.

https://tinyurl.com/2tdh97z4



Gaussian posterior samples

Preferential data subsampling

Preferential data subsampling

An alternative gradient estimator for SGLD can be given by reweighting the simple estimator

$$\nabla \hat{f}_{ps}(\theta^{(t)}) = \nabla f_0\left(\theta^{(t)}\right) + \frac{1}{n} \sum_{i \in \mathcal{S}} \frac{1}{p_i^t} \nabla f_i\left(\theta^{(t)}\right),$$

where $\mathcal{S} \subset \{1, \ldots, N\}$ is selected according to $\mathbf{p}^{(t)} = (p_1^t, \ldots, p_N^t)^T$ and $|\mathcal{S}| = n \quad (n \ll N)$.

Preferential data subsampling

An alternative gradient estimator for SGLD can be given by reweighting the simple estimator

$$\nabla \hat{f}_{ps}(\theta^{(t)}) = \nabla f_0\left(\theta^{(t)}\right) + \frac{1}{n} \sum_{i \in \mathcal{S}} \frac{1}{p_i^t} \nabla f_i\left(\theta^{(t)}\right),$$

where $\mathcal{S} \subset \{1, \ldots, N\}$ is selected according to $\mathbf{p}^{(t)} = (p_1^t, \ldots, p_N^t)^T$ and $|\mathcal{S}| = n \quad (n \ll N)$.

How do we set the weights $(p_1^t, \ldots, p_N^t)^T$?

Preferential data subsampling

An alternative gradient estimator for SGLD can be given by reweighting the simple estimator

$$\nabla \hat{f}_{ps}(\theta^{(t)}) = \nabla f_0\big(\theta^{(t)}\big) + \frac{1}{n} \sum_{i \in \mathcal{S}} \frac{1}{p_i^t} \nabla f_i\big(\theta^{(t)}\big),$$

where $\mathcal{S} \subset \{1, \ldots, N\}$ is selected according to $\mathbf{p}^{(t)} = (p_1^t, \ldots, p_N^t)^T$ and $|\mathcal{S}| = n \quad (n \ll N)$.

How do we set the weights $(p_1^t, \ldots, p_N^t)^T$?

Our goal is to find a **preferential subsampling distribution $\mathbf{p}$**, which minimises the gradient variance:

$$\min_{\mathbf{p}^{(t)}, p_i^t \in [0,1], \sum_i p_i^t = 1} \mathrm{Var}\Big[\nabla \hat{f}_{ps}(\theta^{(t)})\Big] .$$

Preferential data subsampling

An alternative gradient estimator for SGLD can be given by reweighting the simple estimator

$$\nabla \hat{f}_{ps}(\theta^{(t)}) = \nabla f_0\big(\theta^{(t)}\big) + \frac{1}{n} \sum_{i \in \mathcal{S}} \frac{1}{p_i^t} \nabla f_i\big(\theta^{(t)}\big),$$

where $\mathcal{S} \subset \{1, \ldots, N\}$ is selected according to $\mathbf{p}^{(t)} = (p_1^t, \ldots, p_N^t)^T$ and $|\mathcal{S}| = n \quad (n \ll N)$.

How do we set the weights $(p_1^t, \ldots, p_N^t)^T$?

Our goal is to find a **preferential subsampling distribution $\mathbf{p}$**, which minimises the gradient variance:

$$\min_{\mathbf{p}^{(t)}, p_i^t \in [0,1], \sum_i p_i^t = 1} \mathrm{Var}\Big[\nabla \hat{f}_{ps}(\theta^{(t)})\Big] .$$

Note that $p_i = 1/N$ gives us the original gradient estimator.

Optimal preferential data subsampling

Proposition

For the SGLD with preferential subsampling gradient estimator $\nabla \hat{f}_{ps}(\theta)$, the variance is given by

$$\min_{\mathbf{p}^{(t)},\, p_i^t \in [0,1],\, \sum_i p_i^t = 1} \frac{1}{n} \sum_{i=1}^{N} \frac{1}{p_i^t} \left\| \nabla f_i\left(\theta^{(t)}\right) \right\|^2,$$

and the optimal weights which minimise the gradient variance are,

$$p_i^t = \frac{\| \nabla f_i\left(\theta^{(t)}\right) \|}{\sum_{k=1}^{N} \| \nabla f_k\left(\theta^{(t)}\right) \|} \quad \text{for } i = 1, \ldots, N.$$

Optimal preferential data subsampling

Proposition

For the SGLD-CV with preferential subsampling gradient estimator $\nabla \hat{f}_{ps}(\theta)$, the variance is given by

$$\min_{\mathbf{p}^{(t)}, p_i^t \in [0,1], \sum_i p_i^t = 1} \frac{1}{n} \sum_{i=1}^{N} \frac{1}{p_i^t} \left\| \nabla f_i\left(\theta^{(t)}\right) - \nabla f_i(\hat{\theta}) \right\|^2,$$

and the optimal weights which minimise the gradient variance are,

$$p_i^t = \frac{\| \nabla f_i\left(\theta^{(t)}\right) - \nabla f_i(\hat{\theta})) \|}{\sum_{k=1}^{N} \| \nabla f_k\left(\theta^{(t)}\right) - \nabla f_k(\hat{\theta}) \|} \quad \text{for } i = 1, \ldots, N.$$

Approximate preferential data subsampling

Approximate preferential data subsampling

The optimal weights, $p_i^t = \|\nabla f_i\left(\theta^{(t)}\right)\| / \sum_{k=1}^{N} \|\nabla f_k\left(\theta^{(t)}\right)\|$, depend on the current state of the Markov chain $\theta^{(t)}$ and require N gradient calculations at each iteration t.

Approximate preferential data subsampling

The optimal weights, $p_i^t = \|\nabla f_i\left(\theta^{(t)}\right)\| / \sum_{k=1}^{N} \|\nabla f_k\left(\theta^{(t)}\right)\|$, depend on the current state of the Markov chain $\theta^{(t)}$ and require N gradient calculations at each iteration t.

A simple approximation of the optimal weights is given by substituting the current state $\theta^{(t)}$ with some fixed point $\hat{\theta}$, (i.e. the posterior mode). This is a sensible choice as it represents the most probable estimate of the parameters.

Approximate preferential data subsampling

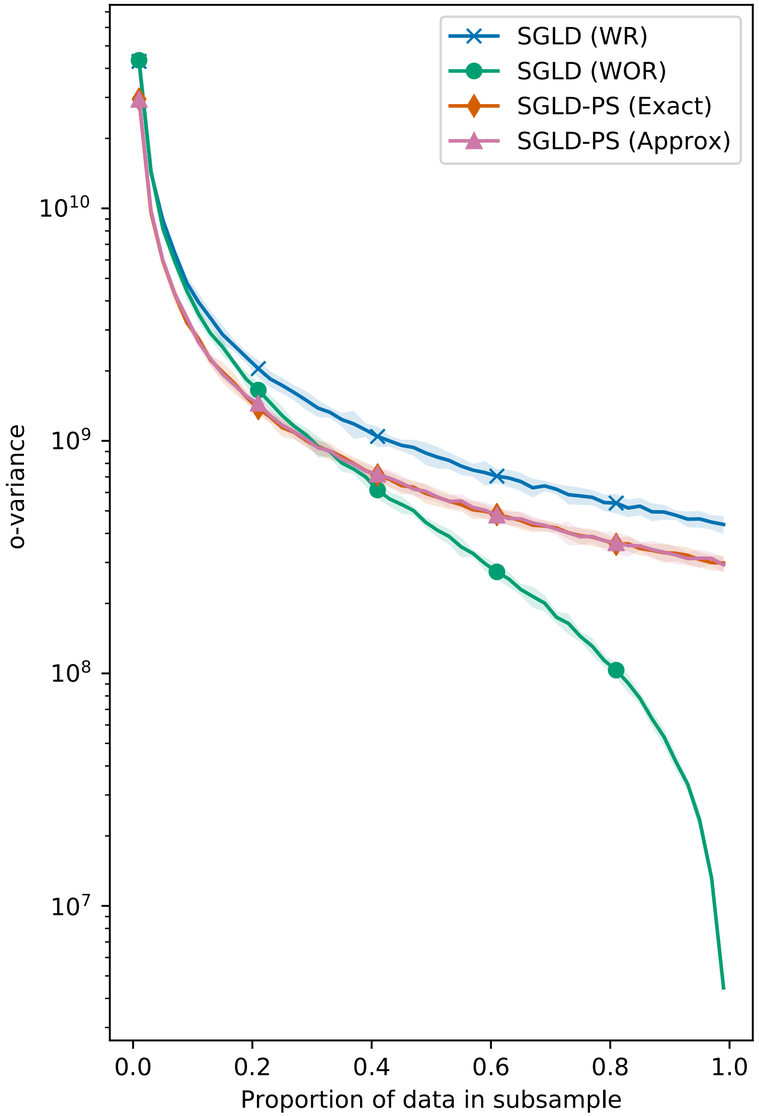
The optimal weights, $p_i^t = \|\nabla f_i\left(\theta^{(t)}\right)\| / \sum_{k=1}^{N} \|\nabla f_k\left(\theta^{(t)}\right)\|$, depend on the current state of the Markov chain $\theta^{(t)}$ and require N gradient calculations at each iteration t.

A simple approximation of the optimal weights is given by substituting the current state $\theta^{(t)}$ with some fixed point $\hat{\theta}$, (i.e. the posterior mode). This is a sensible choice as it represents the most probable estimate of the parameters.

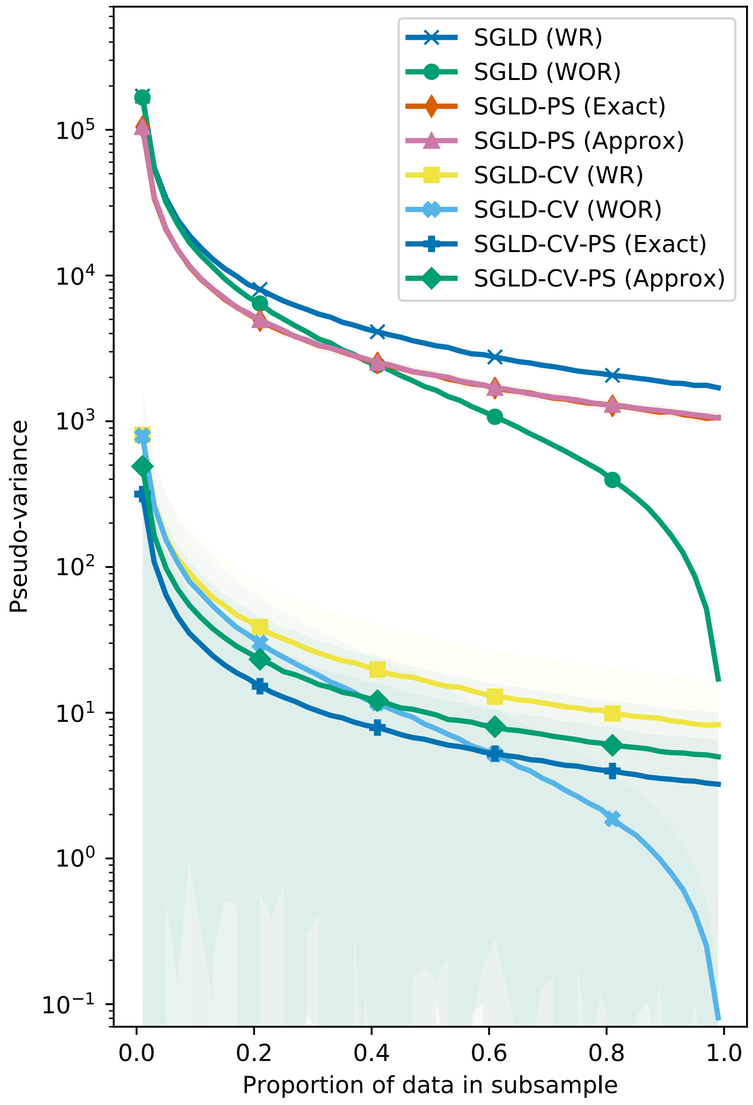
In this case, the approximate subsampling scheme would be given by,

$$\hat{p}_i^t = \frac{\|\nabla f_i\left(\hat{\theta}\right)\|}{\sum_{k=1}^{N} \|\nabla f_k\left(\hat{\theta}\right)\|} \quad \text{for } i = 1, \ldots, N.$$

Variance of the gradients



Gaussian model



Logistic regression model

Bivariate Gaussian example

We want to simulate independent data from:

$$Y_i \,|\, \theta \sim \mathcal{N}_2(\theta, \Sigma_x) \;\; \text{for } i = 1, \ldots, N\,.$$

The conjugate prior for θ is set to be

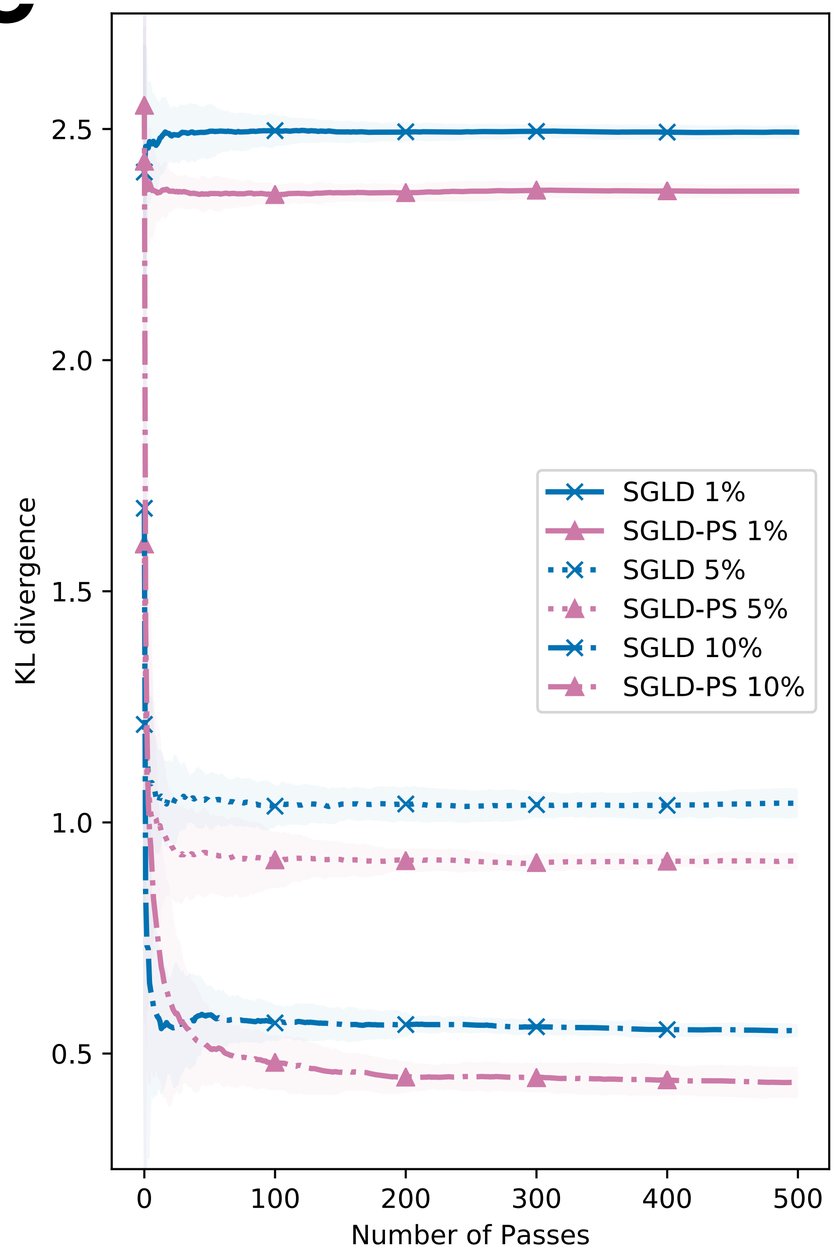
$$\theta \sim \mathcal{N}_2(\mu_0, \Lambda_0)\,.$$

The conjugate posterior that we are ultimately trying to simulate from using SGLD is known to be:

$$\pi(\theta \,|\, \mathbf{y}) \propto \exp\left(-\frac{1}{2}(\theta - \mu_1)^T \Lambda_1^{-1} (\theta - \mu_1) \right) \stackrel{D}{=} \mathcal{N}_2(\mu_1, \Lambda_1),$$

where $\mu_1 = (\Lambda_0^{-1} + N\Sigma_x^{-1})^{-1}(\Lambda_0^{-1}\mu_0 + N\Sigma_x^{-1}\bar{y})$,

and $\Lambda_1^{-1} = \Lambda_0^{-1} + N\Sigma_x^{-1}$.



Bivariate Gaussian example

We want to simulate independent data from:

$$Y_i \,|\, \theta \sim \mathcal{N}_2(\theta, \Sigma_x) \;\; \text{for } i = 1, \ldots, N\,.$$

The conjugate prior for θ is set to be

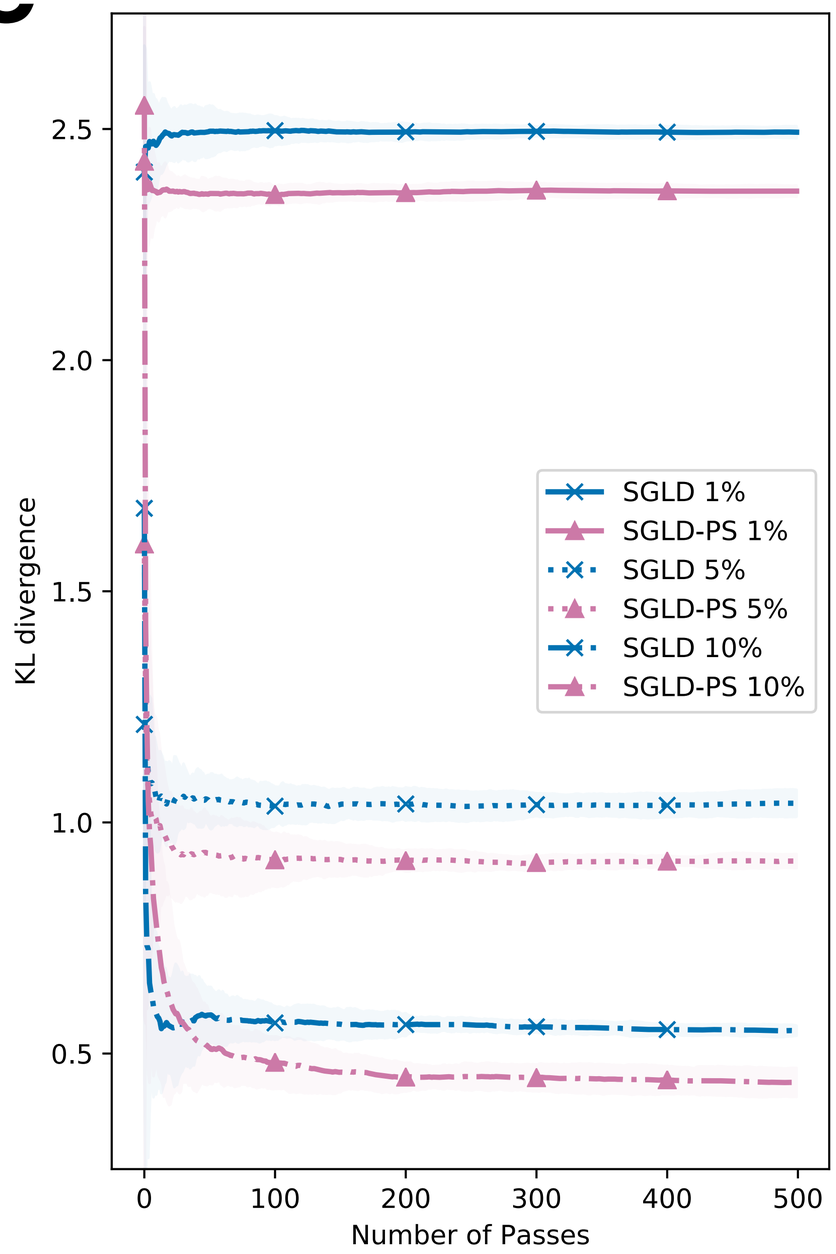
$$\theta \sim \mathcal{N}_2(\mu_0, \Lambda_0)\,.$$

The conjugate posterior that we are ultimately trying to simulate from using SGLD is known to be:

$$\pi(\theta \,|\, \mathbf{y}) \propto \exp\left(-\frac{1}{2}(\theta - \mu_1)^T \Lambda_1^{-1} (\theta - \mu_1) \right) \stackrel{D}{=} \mathcal{N}_2(\mu_1, \Lambda_1),$$

where $\mu_1 = (\Lambda_0^{-1} + N\Sigma_x^{-1})^{-1}(\Lambda_0^{-1}\mu_0 + N\Sigma_x^{-1}\bar{y})$,

and $\Lambda_1^{-1} = \Lambda_0^{-1} + N\Sigma_x^{-1}$.



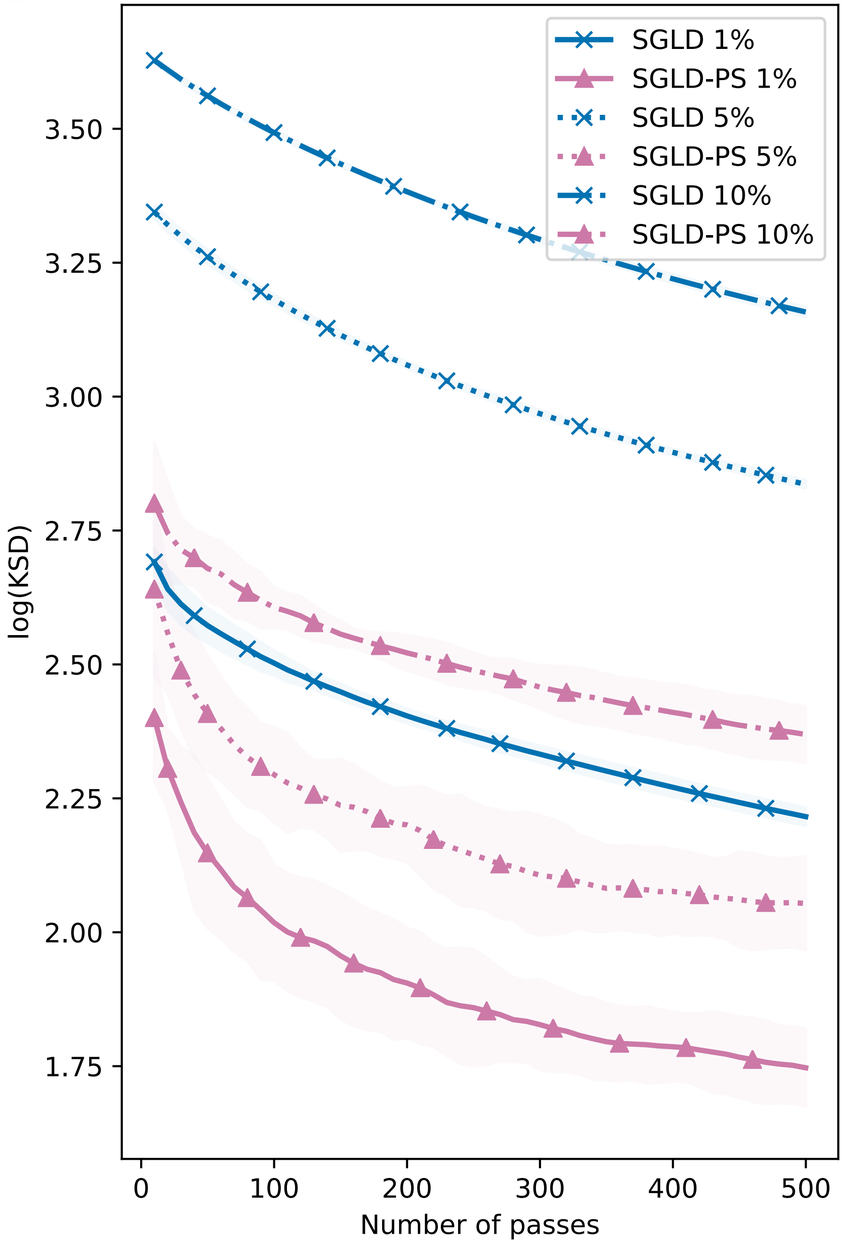
Logistic regression example

A logistic regression model with parameters $\theta = (\beta_0, \beta_1, \ldots, \beta_p)$ representing the coefficients β_j for $j = 1, \ldots, p$ and bias β_0 will have likelihood

$$p(X, y \mid \theta) = \prod_{i=1}^{N} \left[\frac{1}{1 + e^{-\theta^T x_i}} \right]^{y_i} \left[1 - \frac{1}{1 + e^{-\theta^T x_i}} \right]^{1-y_i}$$

The prior for θ is set to be $\theta \sim \text{MVN}(\mu_0, \Lambda_0)$. The hyper-parameters of the prior are $\mu_0 = (0, \ldots, 0)^T$ and $\Lambda_0 = \text{diag}(10, d)$.

We use the *covertype* dataset, where $N = 581{,}012$ and $p = 55$.



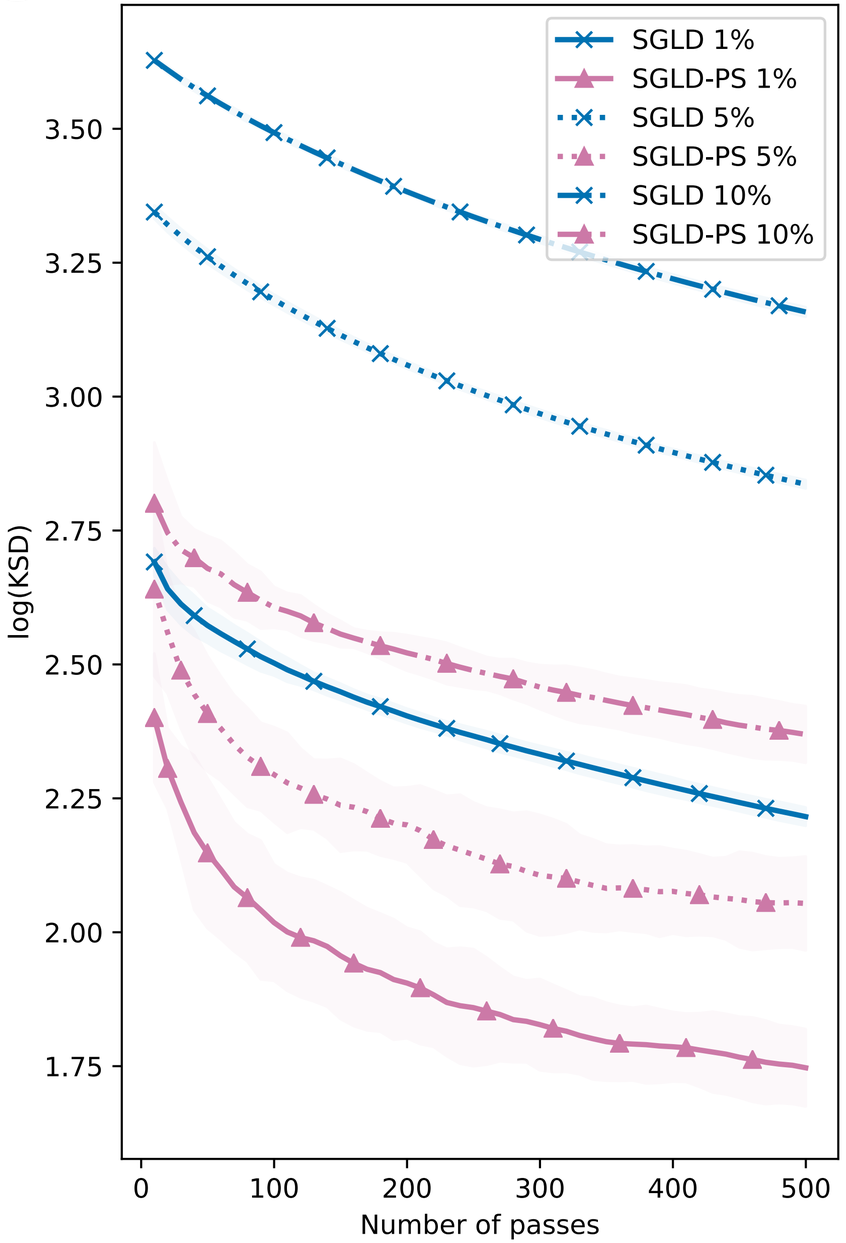
Logistic regression example

A logistic regression model with parameters $\theta = (\beta_0, \beta_1, \ldots, \beta_p)$ representing the coefficients β_j for $j = 1, \ldots, p$ and bias β_0 will have likelihood

$$p(X, y \mid \theta) = \prod_{i=1}^{N} \left[\frac{1}{1 + e^{-\theta^T x_i}} \right]^{y_i} \left[1 - \frac{1}{1 + e^{-\theta^T x_i}} \right]^{1-y_i}$$

The prior for θ is set to be $\theta \sim \text{MVN}(\mu_0, \Lambda_0)$. The hyper-parameters of the prior are $\mu_0 = (0, \ldots, 0)^T$ and $\Lambda_0 = \text{diag}(10, d)$.

We use the *covertype* dataset, where $N = 581{,}012$ and $p = 55$.



Adaptive subsample size

Adaptive subsample size

So far we've assumed that the subsample size $|\mathcal{S}| = n \quad (n \ll N)$ is fixed. However, we could think about dynamically controlling n.

Adaptive subsample size

So far we've assumed that the subsample size $|\mathcal{S}| = n \quad (n \ll N)$ is fixed. However, we could think about dynamically controlling n.

Assumption (Lipschitz continuity of gradients)
There exists constants $L_0, \ldots, L_N$ such that

$$\|\nabla f_i(\theta) - \nabla f_i(\theta')\| \leq L_i \|\theta - \theta'\|, \quad \text{for} \quad i = 0, \ldots, N.$$

Adaptive subsample size

So far we've assumed that the subsample size $|\mathcal{S}| = n \quad (n \ll N)$ is fixed. However, we could think about dynamically controlling n.

Assumption (Lipschitz continuity of gradients)
There exists constants $L_0, \ldots, L_N$ such that

$$\|\nabla f_i(\theta) - \nabla f_i(\theta')\| \leq L_i \|\theta - \theta'\|, \quad \text{for} \quad i = 0, \ldots, N.$$

Under the Lipschitz assumption, the variance of the stochastic gradient estimator can be bounded above by

$$\text{Var}[\nabla \hat{f}_{ps}(\theta^{(t)})] \leq \frac{1}{n} \|\theta^{(t)} - \hat{\theta}\|^2 \left(\sum_{i=1}^{N} \frac{L_i^2}{p_i^t} \right)$$

where $(p_1^t, \ldots, p_N^t)^T$ is a set of user-defined discrete weights (chosen such that $\sum_{i=1}^{N} p_i^T = 1$).

Adaptive subsample size

Adaptive subsample size

$$\mathrm{Var}[\,\nabla \hat{f}_{ps}(\theta^{(t)})] \leq \frac{1}{n}\|\theta^{(t)} - \hat{\theta}\|^2 \left(\sum_{i=1}^{N} \frac{L_i^2}{p_i^t} \right).$$

Adaptive subsample size

$$\text{Var}[\nabla \hat{f}_{ps}(\theta^{(t)})] \leq \frac{1}{n}\|\theta^{(t)} - \hat{\theta}\|^2 \left(\sum_{i=1}^{N} \frac{L_i^2}{p_i^t} \right).$$

We can set the upper threshold of the variance to be some fixed value $V_0 > 0$,

$$\frac{1}{n}\|\theta^{(t)} - \hat{\theta}\|^2 \left(\sum_{i=1}^{N} \frac{L_i^2}{p_i^t} \right) < V_0 .$$

Adaptive subsample size

$$\text{Var}[\nabla \hat{f}_{ps}(\theta^{(t)})] \leq \frac{1}{n}\|\theta^{(t)} - \hat{\theta}\|^2 \left(\sum_{i=1}^{N} \frac{L_i^2}{p_i^t} \right).$$

We can set the upper threshold of the variance to be some fixed value $V_0 > 0$,

$$\frac{1}{n}\|\theta^{(t)} - \hat{\theta}\|^2 \left(\sum_{i=1}^{N} \frac{L_i^2}{p_i^t} \right) < V_0 .$$

Rearranging this inequality we obtain the following lower bound on the subsample size,

$$n > \frac{1}{V_0}\|\theta^{(t)} - \hat{\theta}\|^2 \left(\sum_{i=1}^{N} \frac{L_i^2}{p_i^t} \right).$$

Adaptive subsample size

$$\text{Var}[\nabla \hat{f}_{ps}(\theta^{(t)})] \leq \frac{1}{n}\|\theta^{(t)} - \hat{\theta}\|^2 \left(\sum_{i=1}^{N} \frac{L_i^2}{p_i^t} \right).$$

We can set the upper threshold of the variance to be some fixed value $V_0 > 0$,

$$\frac{1}{n}\|\theta^{(t)} - \hat{\theta}\|^2 \left(\sum_{i=1}^{N} \frac{L_i^2}{p_i^t} \right) < V_0 .$$

Rearranging this inequality we obtain the following lower bound on the subsample size,

$$n > \frac{1}{V_0}\|\theta^{(t)} - \hat{\theta}\|^2 \left(\sum_{i=1}^{N} \frac{L_i^2}{p_i^t} \right).$$

This result tells us that for fixed $L_i, \hat{p}_i^t$ and V_0, we should choose a subsample size of $n \propto \|\theta^{(t)} - \hat{\theta}\|^2$.

Wrap-Up

Wrap-Up

- **Key message:** You have to control the variance of the gradient with stochastic gradient MCMC algorithms.

Wrap-Up

- **Key message:** You have to control the variance of the gradient with stochastic gradient MCMC algorithms.
- Control variates and preferential subsampling can reduce the gradient variance, but there are probably lots of other ways of doing this that could be explored.

Wrap-Up

- **Key message:** You have to control the variance of the gradient with stochastic gradient MCMC algorithms.
- Control variates and preferential subsampling can reduce the gradient variance, but there are probably lots of other ways of doing this that could be explored.
- The ideas discussed today carry over to other similar algorithms, such as stochastic gradient Hamiltonian Monte Carlo.

Wrap-Up

- **Key message:** You have to control the variance of the gradient with stochastic gradient MCMC algorithms.
- Control variates and preferential subsampling can reduce the gradient variance, but there are probably lots of other ways of doing this that could be explored.
- The ideas discussed today carry over to other similar algorithms, such as stochastic gradient Hamiltonian Monte Carlo.

- Paper and code will be on *arXiv* soon.

New Stochastic Gradient MCMC Package

SGMCMCJax

latest

Search docs

HOW TO USE SGMCMCJAX

Overview

Build a sampler function

Build a transition kernel

Build a diffusion solver

EXAMPLES

Gaussian posterior

Logistic Regression

Bayesian Neural Network

Flax CNN

Sample from a PyMC model using SGMCMCJax

» Welcome to SGMCMCJax's documentation!

Edit on GitHub

Welcome to SGMCMCJax's documentation!

SGMCMCJax is a lightweight library of stochastic gradient Markov chain Monte Carlo (SGMCMC) algorithms. The aim is to include both standard samplers (SGLD, SGHMC) as well as state of the art samplers (SVRG-langevin, others, ...).

The target audience for this library is researchers and practitioners: simply plug in your JAX model and easily obtain samples.

You can find the source code on Github.

"Hello World" example

Estimate the mean of a Gaussian using SGLD: