Stepsize selection in Langevin Monte Carlo via coupling

Matteo Sordello

Student Seminar Day

Joint work with James Johndrow and Weijie Su

Langevin Dynamics

Goal:

Sample from a target distribution

$$\pi(x) = \frac{e^{-U(x)}}{\int e^{-U(y)} dy}$$

Langevin Dynamics

Goal:

Sample from a target distribution

$$\pi(x) = \frac{e^{-U(x)}}{\int e^{-U(y)} dy}$$

The Langevin stochastic differential equation (SDE)

$$dY_t = -\nabla U(Y_t) dt + \sqrt{2} dB_t$$

has π as unique invariant distribution.

Langevin Dynamics

Goal:

Sample from a target distribution

$$\pi(x) = \frac{e^{-U(x)}}{\int e^{-U(y)}dy}$$

The Langevin stochastic differential equation (SDE)

$$dY_t = -\nabla U(Y_t)dt + \sqrt{2}dB_t$$

has π as unique invariant distribution.The Euler discretization of this SDE is

$$X_{t+1} = X_t - \eta_{t+1}\nabla U(X_t) + \sqrt{2\eta_{t+1}} \cdot Z_{t+1}$$

where (Z_t) is a sequence of standard Gaussian distributions. It is usually referred to as Unadjusted Langevin Algorithm (ULA).

Bias in Langevin Dynamics

When we discretize the SDE and introduce the stepsize η_k, we inevitably get biased estimates for π.

Bias in Langevin Dynamics

When we discretize the SDE and introduce the stepsize η_k, we inevitably get biased estimates for π.

- If η_k is small, we introduce small bias but it takes a large number of iterations to explore the support of the target distribution.

Bias in Langevin Dynamics

When we discretize the SDE and introduce the stepsize η_k, we inevitably get biased estimates for π.

- If η_k is small, we introduce small bias but it takes a large number of iterations to explore the support of the target distribution.
- If η_k is large, we quickly explore the support of the target distribution but introduce more bias.

Upper bounds for Langevin in Wasserstein distance

Definition:

Let Q_η^t be the kernel associated with the first t jumps $\{X_1, ..., X_t\}$.

Upper bounds for Langevin in Wasserstein distance

Definition:

Let Q_η^t be the kernel associated with the first t jumps $\{X_1, ..., X_t\}$.

Durmus and Moulines[1] proved a bound in Wasserstein distance between the distribution π and one iterate of ULA:

$$W_2^2(\delta_x Q_\eta^t, \pi) \leq \underbrace{u_t^{(1)}(\eta)\left(\|x - x^*\|^2 + \frac{d}{m}\right)}_{\text{transient}} + \underbrace{u_t^{(2)}(\eta)}_{\text{stationary}}$$

[1] Durmus, Moulines (2018) High-dimensional Bayesian inference via the Unadjusted Langevin Algorithm

Upper bounds for Langevin in Wasserstein distance

Definition:

Let Q_η^t be the kernel associated with the first t jumps $\{X_1, ..., X_t\}$.

Durmus and Moulines[1] proved a bound in Wasserstein distance between the distribution π and one iterate of ULA:

$$W_2^2(\delta_x Q_\eta^t, \pi) \leq \underbrace{u_t^{(1)}(\eta)\left(\|x - x^*\|^2 + \frac{d}{m}\right)}_{\text{transient}} + \underbrace{u_t^{(2)}(\eta)}_{\text{stationary}}$$

where

$$u_t^{(1)}(\eta) = 2\prod_{k=1}^{t}(1 - \kappa\eta_k/2)$$

$$u_t^{(2)}(\eta) = L^2 d \sum_{i=1}^{t}\left(\eta_i^2\left(\frac{1}{\kappa} + \eta_i\right)\left(2 + \frac{L^2\eta_i}{m} + \frac{L^2\eta_i^2}{6}\right)\prod_{k=i+1}^{t}(1 - \kappa\eta_k/2)\right)$$

[1] Durmus, Moulines (2018) High-dimensional Bayesian inference via the Unadjusted Langevin Algorithm

Bound for Normals in dimension 2

Comment:

- The Wasserstein distance satisfies $W_2\left(\frac{1}{t}\sum_{k=1}^{t}\delta_x Q_\eta^k, \pi\right) \leq \frac{1}{t}\sum_{k=1}^{t} W_2\left(\delta_x Q_\eta^k, \pi\right)$

Bound for Normals in dimension 2

Comment:

- The Wasserstein distance satisfies $W_2\left(\frac{1}{t}\sum_{k=1}^{t}\delta_x Q_\eta^k, \pi\right) \leq \frac{1}{t}\sum_{k=1}^{t} W_2\left(\delta_x Q_\eta^k, \pi\right)$

Question:

- Is it better to have a constant or decaying stepsize?

Comment:

- The Wasserstein distance satisfies $W_2\left(\frac{1}{t}\sum_{k=1}^{t}\delta_x Q_\eta^k, \pi\right) \leq \frac{1}{t}\sum_{k=1}^{t} W_2\left(\delta_x Q_\eta^k, \pi\right)$

Question:

- Is it better to have a constant or decaying stepsize?

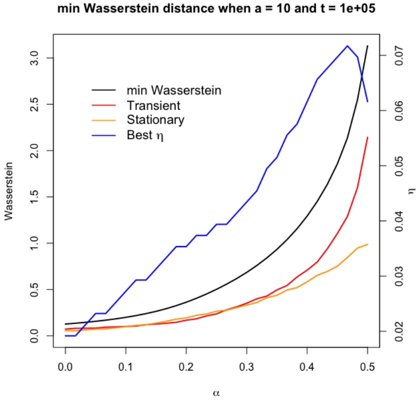
Consider

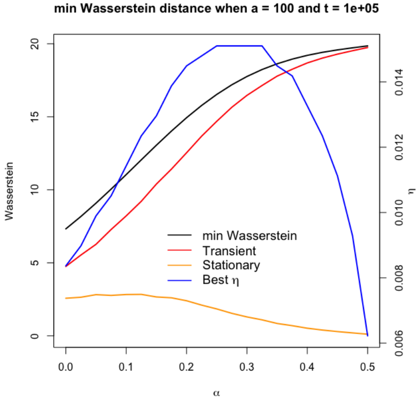
$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \sim N\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} a & 0 \\ 0 & 1 \end{pmatrix}\right) \quad \Rightarrow \quad U(x) = \frac{x_1^2}{2a} + \frac{x_2^2}{2}$$

with smoothness $L = 1$ and strong convexity $m = 1/a$.

STEPSIZE: $$\eta_t = \frac{\eta}{t^\alpha} \qquad \text{with} \quad \alpha \in [0,1]$$

Choice of α and η





- For each value of α we search the value of η that minimizes the bound for the Wasserstein distance
- It turns out that, with the correct choice of η, a constant stepsize ($\alpha = 0$) is optimal

How to pick optimal η

Problem:

A priori we don't know how to pick η.

How to pick optimal η

Problem:

A priori we don't know how to pick η.

Idea:

Use **coupling** to decide when it is time to decrease the stepsize. If we start from points that are far enough (in the sense that the distributions induced by one step of ULA are sufficiently far in Total Variation distance), the coupling time approximates the time when we reached stationarity.

How to pick optimal η

Problem:

A priori we don't know how to pick η.

Idea:

Use **coupling** to decide when it is time to decrease the stepsize. If we start from points that are far enough (in the sense that the distributions induced by one step of ULA are sufficiently far in Total Variation distance), the coupling time approximates the time when we reached stationarity.

Tools for coupling:

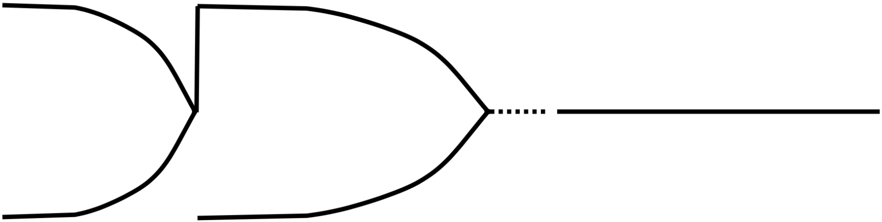
- One step coupled ULA threads. Use the same noise for both

$$\begin{cases} X_{t+1} = X_t - \eta_{t+1} \nabla U(X_t) + \sqrt{2\eta_{t+1}} \cdot Z_{t+1} \\ X'_{t+1} = X'_t - \eta_{t+1} \nabla U(X'_t) + \sqrt{2\eta_{t+1}} \cdot Z_{t+1} \end{cases}$$

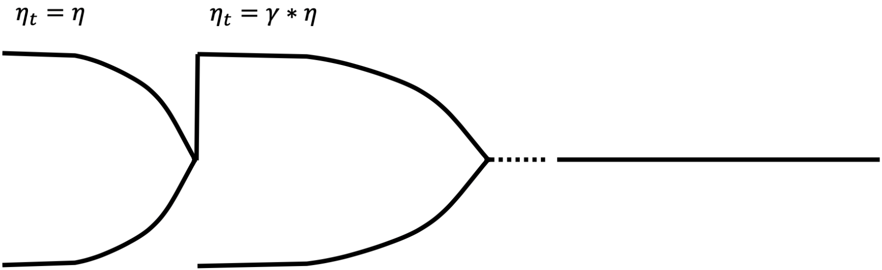
- One step maximal coupling, to maximize the probability that the two chains meet.

Algorithm

$\eta_t = \eta$ $\eta_t = \gamma * \eta$

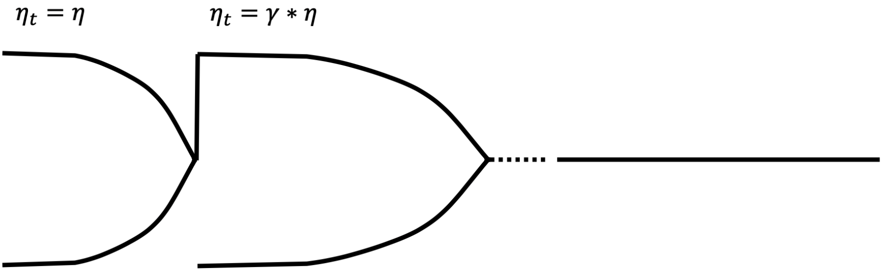


Algorithm



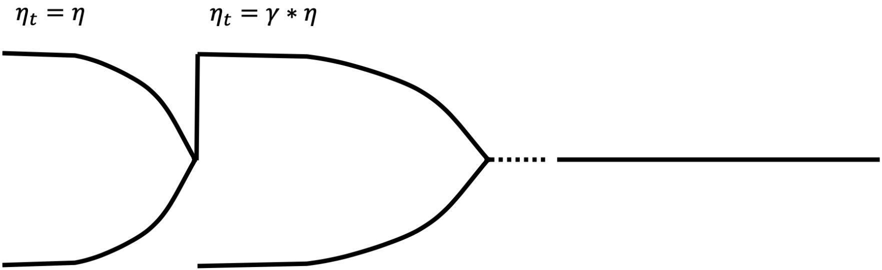
- Start from the pair of points $X^{(1)}$ and $X^{(2)}$

Algorithm



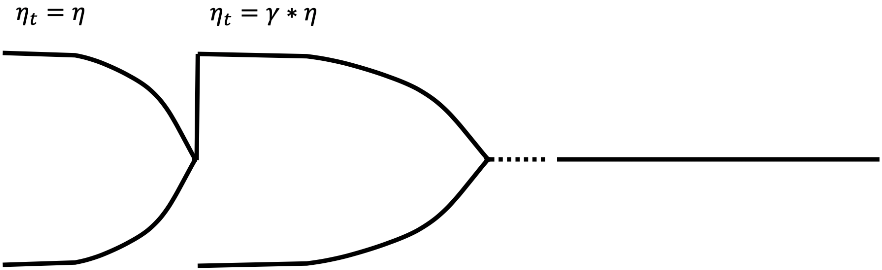
- Start from the pair of points $X^{(1)}$ and $X^{(2)}$
- Run coupled Langevin steps with stepsize η until the two threads are sufficiently close, then attempt a maximal coupling step. If it fails, continue with the coupled Langevin steps

Algorithm



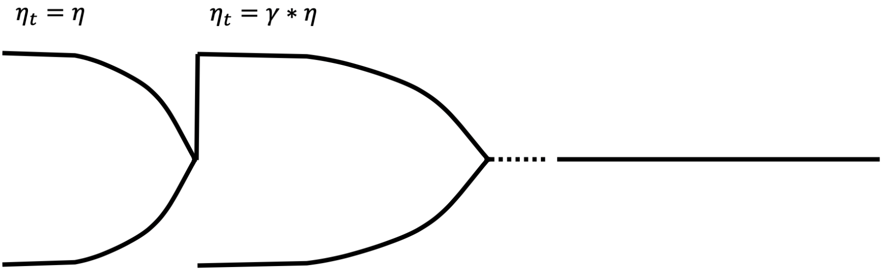
- Start from the pair of points $X^{(1)}$ and $X^{(2)}$
- Run coupled Langevin steps with stepsize η until the two threads are sufficiently close, then attempt a maximal coupling step. If it fails, continue with the coupled Langevin steps
- Wait for them to couple

Algorithm



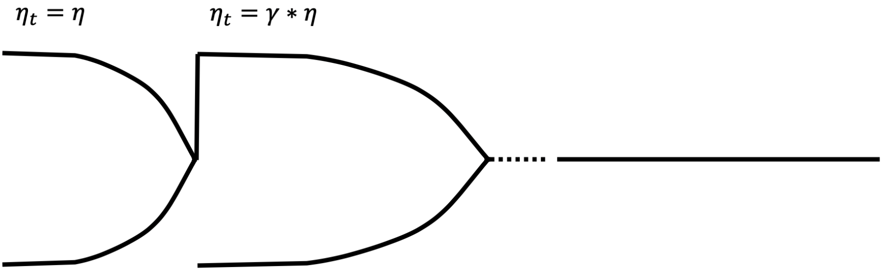
- Sample a new starting point for the second thread

Algorithm

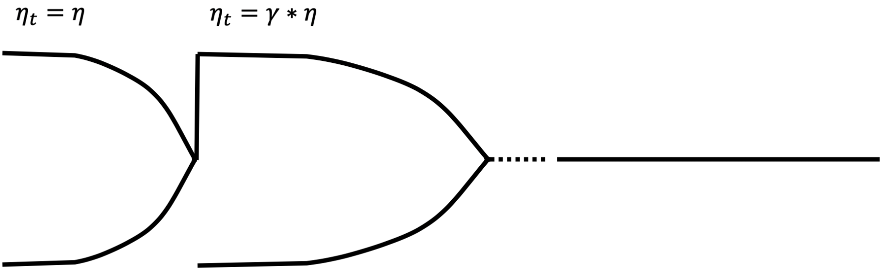


- Sample a new starting point for the second thread
- Reduce the stepsize by a factor γ and wait again for the two threads to couple

Algorithm

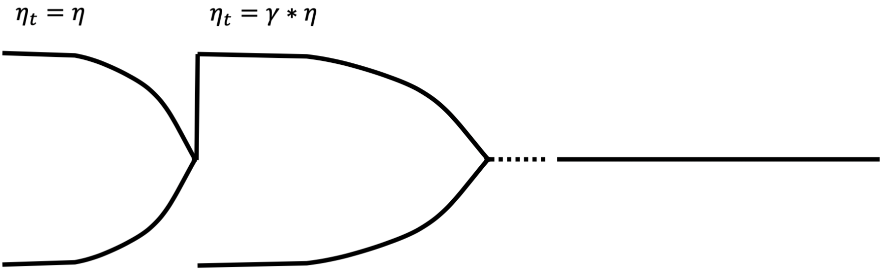


- Sample a new starting point for the second thread
- Reduce the stepsize by a factor γ and wait again for the two threads to couple
- Continue until we used more than a certain portion of the pre-allocated computational budget



- Sample a new starting point for the second thread
- Reduce the stepsize by a factor γ and wait again for the two threads to couple
- Continue until we used more than a certain portion of the pre-allocated computational budget
- Use the remaining budget to run ULA with the last stepsize

Algorithm



- Sample a new starting point for the second thread
- Reduce the stepsize by a factor γ and wait again for the two threads to couple
- Continue until we used more than a certain portion of the pre-allocated computational budget
- Use the remaining budget to run ULA with the last stepsize

Trick: use more that one coupled thread

If coupling does not happen

Keep track of

$$D_t := \|X_t^{(1)} - X_t^{(2)}\|$$

and count the number of times that

$$D_{t+1} > D_t$$

If coupling does not happen

Keep track of

$$D_t := \|X_t^{(1)} - X_t^{(2)}\|$$

and count the number of times that

$$D_{t+1} > D_t$$

- If η is too SMALL: this distance is decreasing but does not get to 0. Here we can estimate m and L to jump directly to the optimal η

If coupling does not happen

Keep track of

$$D_t := \|X_t^{(1)} - X_t^{(2)}\|$$

and count the number of times that

$$D_{t+1} > D_t$$

- If η is too SMALL: this distance is decreasing but does not get to 0. Here we can estimate m and L to jump directly to the optimal η
- If η is too LARGE: this distance oscillates a lot

Results on Bayesian Logistic Regression

Goal:

Sample from the posterior distribution for β

Results on Bayesian Logistic Regression

Goal:

Sample from the posterior distribution for β

- $\beta = (1, 1)$ and $n = 200$
- X_i are i.i.d. standard normals with correlation ρ
- $Y_i \sim Bernoulli(p_i)$ where $p_i = \sigma(\beta X_i)$
- The prior we have on β is

$$p(\beta) \sim N\left(\begin{pmatrix}1\\1\end{pmatrix}, \begin{pmatrix}1 & 0\\0 & 1\end{pmatrix}\right)$$

Results on Bayesian Logistic Regression

Goal:

Sample from the posterior distribution for β

- $\beta = (1, 1)$ and $n = 200$
- X_i are i.i.d. standard normals with correlation ρ
- $Y_i \sim \mathit{Bernoulli}(p_i)$ where $p_i = \sigma(\beta X_i)$
- The prior we have on β is

$$p(\beta) \sim N\left(\begin{pmatrix}1\\1\end{pmatrix}, \begin{pmatrix}1 & 0\\0 & 1\end{pmatrix}\right)$$

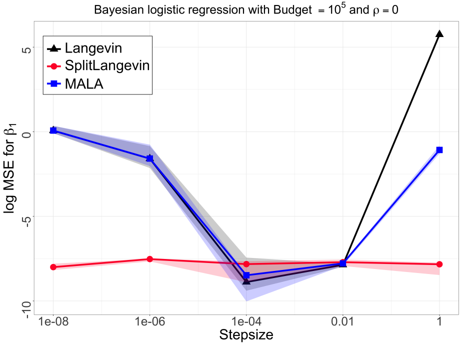
Gradient of the log of the posterior:

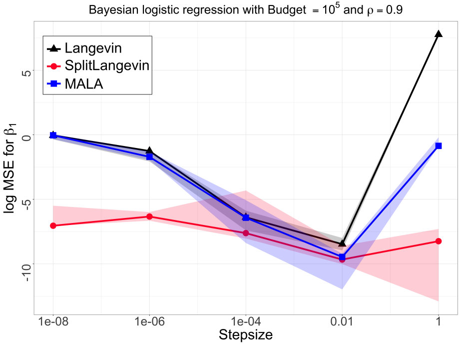
$$\nabla \log(p(\beta|Y, X)) \propto \nabla \log(p(\beta)) + \sum_{i=1}^{n} (Y_i X_i - \sigma(\beta X_i) X_i)$$

Results

- Compare ULA with constant stepsize, our method and Metropolis adjusted Langevin on a range of initial values for the stepsize
- Use 10% of the budget to explore and decide the optimal stepsize

- Compare ULA with constant stepsize, our method and Metropolis adjusted Langevin on a range of initial values for the stepsize
- Use 10% of the budget to explore and decide the optimal stepsize





Directions

- Set the hyperparameters of the algorithm in a principled way

Directions

- Set the hyperparameters of the algorithm in a principled way
- Prove theoretical guarantees on the distance from the target distribution π

Directions

- Set the hyperparameters of the algorithm in a principled way
- Prove theoretical guarantees on the distance from the target distribution π
- Test the performance of the algorithm in other settings

Thank you!