Introduction to Stochastic Gradient Markov Chain Monte Carlo Methods

Changyou Chen

Department of Electrical and Computer Engineering, Duke University
cc448@duke.edu

Duke-Tsinghua Machine Learning Summer School
August 10, 2016

Preface

Stochastic gradient Markov chain Monte Carlo (SG-MCMC):

- A new technique for approximate Bayesian sampling.
- It is about scalable Bayesian learning for big data.
- It draws samples $\{\theta\}$'s from $p(\theta; \mathbf{D})$ where $p(\theta; \mathbf{D})$ is too expensive to be evaluated in each iteration.

This lecture:

- Will cover: basic ideas behind SG-MCMC.
- Will not cover: different kinds of SG-MCMC algorithms, applications, and the corresponding convergence theory.

Outline

Outline

1. Markov Chain Monte Carlo Methods
   - Monte Carlo methods
   - Markov chain Monte Carlo
2. Stochastic Gradient Markov Chain Monte Carlo Methods
   - Introduction
   - Stochastic gradient Langevin dynamics
   - Stochastic gradient Hamiltonian Monte Carlo
   - Application in Latent Dirichlet allocation

Monte Carlo methods

- Monte Carlo method is about drawing a set of samples from $p(\theta)$:

$$\theta_l \sim p(\theta), \quad l = 1, 2, \cdots, L$$

- Approximate the target distribution $p(\theta)$ as count frequency:

$$p(\theta) \approx \frac{1}{L} \sum_{l=1}^{L} \delta(\theta, \theta_l)$$

- An intractable integration is approximated as:

$$\int f(\theta) p(\theta) \approx \frac{1}{L} \sum_{l=1}^{L} f(\theta_l)$$

- In Bayesian modeling, $p(\theta)$ is usually a posterior distribution, the integral is a predicted quantity.

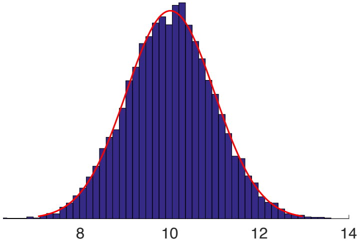
Monte Carlo methods

- Monte Carlo method is about drawing a set of samples from $p(\boldsymbol{\theta})$:

$$\boldsymbol{\theta}_l \sim p(\boldsymbol{\theta}), \quad l = 1, 2, \cdots, L$$

- Approximate the target distribution $p(\boldsymbol{\theta})$ as count frequency:

$$p(\boldsymbol{\theta}) \approx \frac{1}{L} \sum_{l=1}^{L} \delta(\boldsymbol{\theta}, \boldsymbol{\theta}_l)$$



- An intractable integration is approximated as:

$$\int f(\theta) p(\theta) \approx \frac{1}{L} \sum_{l=1}^{L} f(\theta_l)$$

- In Bayesian modeling, $p(\theta)$ is usually a posterior distribution, the integral is a predicted quantity.

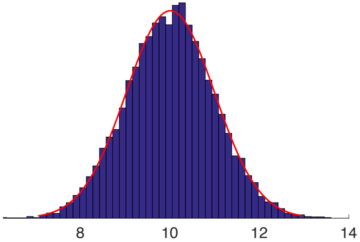
Monte Carlo methods

- Monte Carlo method is about drawing a set of samples from $p(\boldsymbol{\theta})$:

$$\boldsymbol{\theta}_l \sim p(\boldsymbol{\theta}), \quad l = 1, 2, \cdots, L$$

- Approximate the target distribution $p(\boldsymbol{\theta})$ as count frequency:

$$p(\boldsymbol{\theta}) \approx \frac{1}{L} \sum_{l=1}^{L} \delta(\boldsymbol{\theta}, \boldsymbol{\theta}_l)$$



- An intractable integration is approximated as:

$$\int f(\boldsymbol{\theta}) p(\boldsymbol{\theta}) \approx \frac{1}{L} \sum_{l=1}^{L} f(\boldsymbol{\theta}_l)$$

- In Bayesian modeling, $p(\theta)$ is usually a posterior distribution, the integral is a predicted quantity.

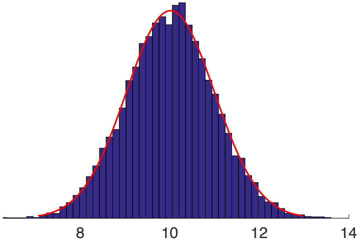
Monte Carlo methods

- Monte Carlo method is about drawing a set of samples from $p(\boldsymbol{\theta})$:

$$\boldsymbol{\theta}_l \sim p(\boldsymbol{\theta}), \quad l = 1, 2, \cdots, L$$

- Approximate the target distribution $p(\boldsymbol{\theta})$ as count frequency:

$$p(\boldsymbol{\theta}) \approx \frac{1}{L} \sum_{l=1}^{L} \delta(\boldsymbol{\theta}, \boldsymbol{\theta}_l)$$



- An intractable integration is approximated as:

$$\int f(\boldsymbol{\theta}) p(\boldsymbol{\theta}) \approx \frac{1}{L} \sum_{l=1}^{L} f(\boldsymbol{\theta}_l)$$

- In Bayesian modeling, $p(\boldsymbol{\theta})$ is usually a posterior distribution, the integral is a predicted quantity.

How does the approximation work?

1. An intractable integration is approximated as:

$$\int f(\theta)p(\theta) \approx \frac{1}{L}\sum_{l=1}^{L} f(\theta_l) \triangleq \tilde{f}$$

2. If $\{\theta_l\}$'s are independent:

$$\mathbb{E}\tilde{f} = \mathbb{E}\left[\frac{1}{L}\sum_{l=1}^{L} f(\theta_l)\right] = \mathbb{E}f, \quad \mathrm{Var}(\tilde{f}) = \mathrm{Var}\left(\frac{1}{L}\sum_{l=1}^{L} f(\theta_l)\right) = \frac{1}{L}\mathrm{Var}(f)$$

   - the variance decreases linearly w.r.t. the number of samples, and independent of the dimension of θ

3. However, obtaining independent samples is hard:
   - usually resort to drawing dependent samples with Markov chain Monte Carlo (MCMC)

How does the approximation work?

1. An intractable integration is approximated as:

$$\int f(\theta)p(\theta) \approx \frac{1}{L}\sum_{l=1}^{L} f(\theta_l) \triangleq \tilde{f}$$

2. If $\{\theta_l\}$'s are independent:

$$\mathbb{E}\tilde{f} = \mathbb{E}\left[\frac{1}{L}\sum_{l=1}^{L} f(\theta_l)\right] = \mathbb{E}f, \quad \text{Var}(\tilde{f}) = \text{Var}\left(\frac{1}{L}\sum_{l=1}^{L} f(\theta_l)\right) = \frac{1}{L}\text{Var}(f)$$

   - the variance decreases linearly w.r.t. the number of samples, and independent of the dimension of θ

3. However, obtaining independent samples is hard:
   - usually resort to drawing dependent samples with Markov chain Monte Carlo (MCMC)

How does the approximation work?

1. An intractable integration is approximated as:

$$\int f(\theta)p(\theta) \approx \frac{1}{L}\sum_{l=1}^{L} f(\theta_l) \triangleq \tilde{f}$$

2. If $\{\theta_l\}$'s are independent:

$$\mathbb{E}\tilde{f} = \mathbb{E}\left[\frac{1}{L}\sum_{l=1}^{L} f(\theta_l)\right] = \mathbb{E}f, \quad \text{Var}(\tilde{f}) = \text{Var}\left(\frac{1}{L}\sum_{l=1}^{L} f(\theta_l)\right) = \frac{1}{L}\text{Var}(f)$$

   - the variance decreases linearly w.r.t. the number of samples, and independent of the dimension of θ

3. However, obtaining independent samples is hard:
   - usually resort to drawing dependent samples with Markov chain Monte Carlo (MCMC)

Outline

MCMC example: a Gaussian model

1. Assume the following generative process (with $\alpha = 5, \beta = 1$):

$$\begin{aligned} x_i|\mu, \tau &\sim N(\mu, 1/\tau), \quad i = 1, \cdots, n = 1000 \\ \mu|\tau, \{x_i\} &\sim N(\mu_0, 1/\tau), \\ \tau &\sim \text{Gamma}(\alpha, \beta) \end{aligned}$$

2. Posterior distribution:
$p(\mu, \tau|\{x_i\}) \propto \left[\prod_{i=1}^n N(x_i; \mu, 1/\tau)\right] N(\mu; \mu_0, 1/\tau)\text{Gamma}(\tau; \alpha, \beta)$
3. Marginal posterior distributions for μ and τ are available:

$$p(\mu|\{x_i\}) \propto \left(2\beta + (\mu - \mu_0)^2 + \sum_i (x_i - \mu)^2\right)^{-\alpha-(n+1)/2}$$

$$p(\tau|\{x_i\}) = \text{Gamma}\left(\alpha + \frac{n}{2}, \beta + \frac{1}{2}\sum_i (x_i - \bar{x})^2 + \frac{n}{2(n+1)}(\bar{x} - \mu_0)^2\right)$$

- $p(\mu|\{x_i\})$ is a non-standardized Student's t-distribution with mean $(\sum_i x_i + \mu_0)/(n+1)$

MCMC example: a Gaussian model

1. Assume the following generative process (with $\alpha = 5, \beta = 1$):

$$\begin{aligned} x_i|\mu, \tau &\sim N(\mu, 1/\tau), \quad i = 1, \cdots, n = 1000 \\ \mu|\tau, \{x_i\} &\sim N(\mu_0, 1/\tau), \\ \tau &\sim \text{Gamma}(\alpha, \beta) \end{aligned}$$

2. Posterior distribution:
$p(\mu, \tau|\{x_i\}) \propto \left[\prod_{i=1}^n N(x_i; \mu, 1/\tau)\right] N(\mu; \mu_0, 1/\tau)\text{Gamma}(\tau; \alpha, \beta)$

3. Marginal posterior distributions for μ and τ are available:

$$p(\mu|\{x_i\}) \propto \left(2\beta + (\mu - \mu_0)^2 + \sum_i (x_i - \mu)^2\right)^{-\alpha-(n+1)/2}$$

$$p(\tau|\{x_i\}) = \text{Gamma}\left(\alpha + \frac{n}{2}, \beta + \frac{1}{2}\sum_i (x_i - \bar{x})^2 + \frac{n}{2(n+1)}(\bar{x} - \mu_0)^2\right)$$

   - $p(\mu|\{x_i\})$ is a non-standardized Student's t-distribution with mean $(\sum_i x_i + \mu_0)/(n+1)$

MCMC example: a Gaussian model

1. Assume the following generative process (with $\alpha = 5, \beta = 1$):

$$\begin{aligned} x_i|\mu, \tau &\sim N(\mu, 1/\tau), \quad i = 1, \cdots, n = 1000 \\ \mu|\tau, \{x_i\} &\sim N(\mu_0, 1/\tau), \\ \tau &\sim \text{Gamma}(\alpha, \beta) \end{aligned}$$

2. Posterior distribution:
$p(\mu, \tau|\{x_i\}) \propto \left[\prod_{i=1}^n N(x_i; \mu, 1/\tau)\right] N(\mu; \mu_0, 1/\tau)\text{Gamma}(\tau; \alpha, \beta)$
3. Marginal posterior distributions for μ and τ are available:

$$p(\mu|\{x_i\}) \propto \left(2\beta + (\mu - \mu_0)^2 + \sum_i (x_i - \mu)^2\right)^{-\alpha-(n+1)/2}$$

$$p(\tau|\{x_i\}) = \text{Gamma}\left(\alpha + \frac{n}{2}, \beta + \frac{1}{2}\sum_i (x_i - \bar{x})^2 + \frac{n}{2(n+1)}(\bar{x} - \mu_0)^2\right)$$

- $p(\mu|\{x_i\})$ is a non-standardized Student's t-distribution with mean $(\sum_i x_i + \mu_0)/(n+1)$

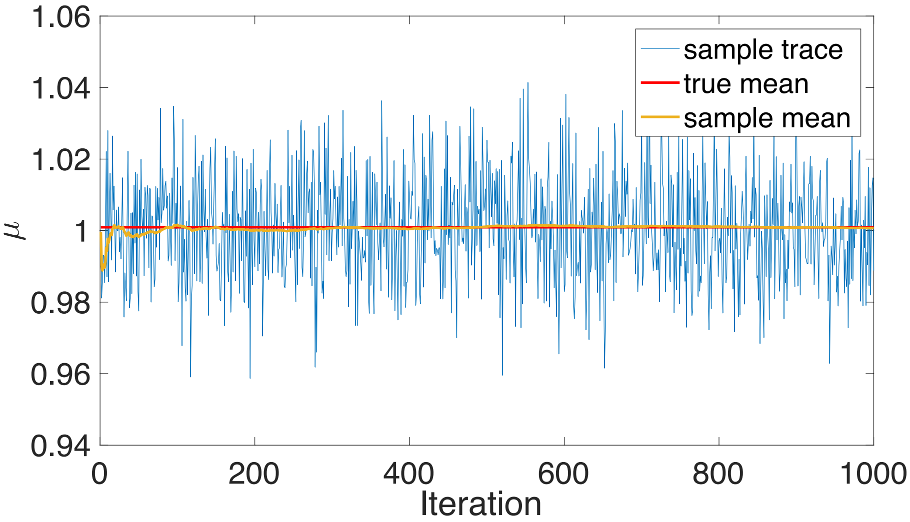
Gibbs sampling μ and τ

1. Conditional distributions:

$$\mu|\tau, \{x_i\} \sim N\left(\frac{n}{n+1}\bar{x} + \frac{1}{n+1}\mu_0, \frac{1}{(n+1)\tau}\right)$$

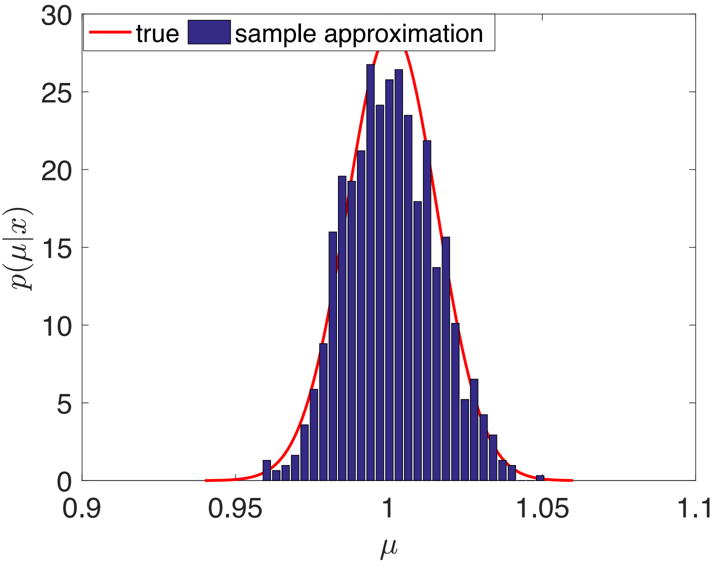
$$\tau|\mu, \{x_i\} \sim \text{Gamma}\left(\alpha + \frac{n+1}{2}, \beta + \frac{\sum_i (x_i - \mu)^2 + (\mu - \mu_0)^2}{2}\right)$$

Trace plot for μ

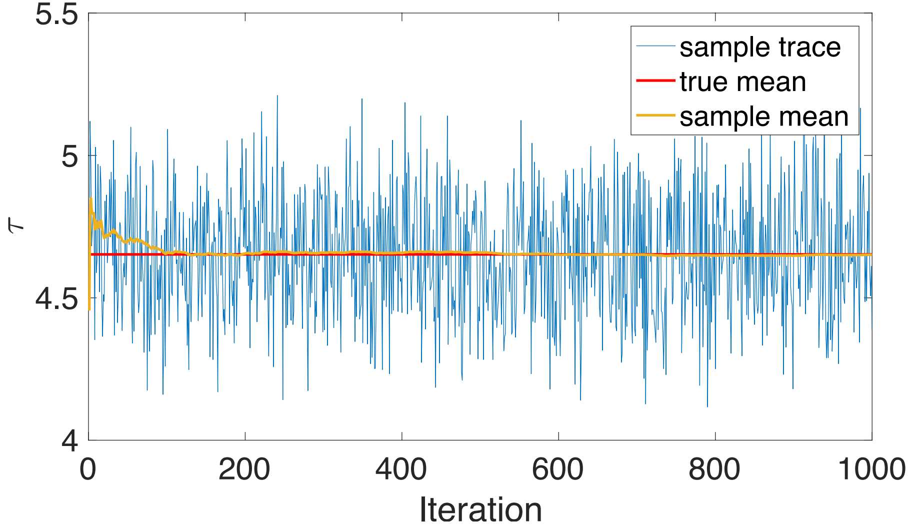


Sample approximation for μ

- True posterior is a non-standardized Student's t-distribution.

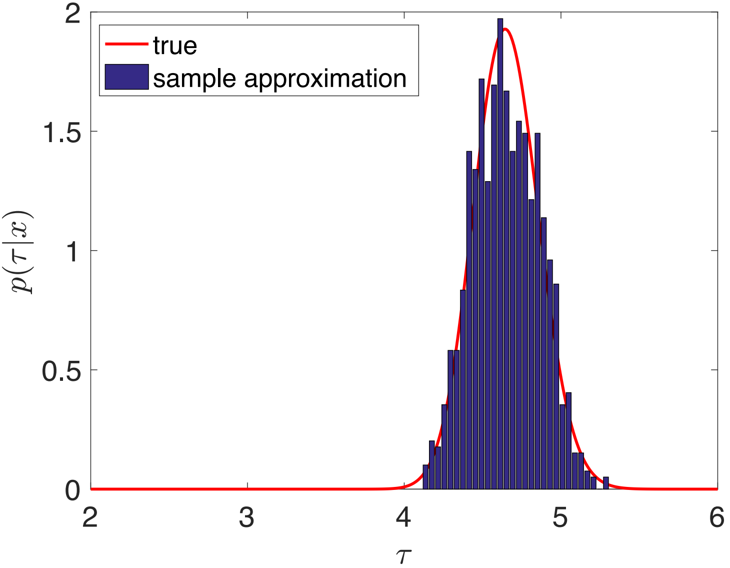


Trace plot for τ



Sample approximation for τ

- True posterior is a Gamma distribution.



Markov chain Monte Carlo methods

1. We are interested in drawing samples from some desired distribution $p^*(\theta) = \frac{1}{Z}\tilde{p}^*(\theta)$.
2. Define a Markov chain:

$$\theta_0 \to \theta_1 \to \theta_2 \to \theta_3 \to \theta_4 \to \theta_5 \to \cdots$$

   where $\theta_0 \sim p_0(\theta)$, $\theta_1 \sim p_1(\theta)$, $\cdots$, satisfying

$$p_t(\theta') = \int p_{t-1}(\theta) T(\theta \to \theta') \mathrm{d}\,\theta \ ,$$

   where $T(\theta \to \theta')$ is the Markov chain transition probability from θ to θ'.
3. We say $p^*(\theta)$ is an invariant (stationary) distribution of the Markov chain iff:

$$p^*(\theta') = \int p^*(\theta) T(\theta \to \theta') \mathrm{d}\,\theta$$

Markov chain Monte Carlo methods

1. We are interested in drawing samples from some desired distribution $p^*(\theta) = \frac{1}{Z}\tilde{p}^*(\theta)$.
2. Define a Markov chain:

$$\theta_0 \to \theta_1 \to \theta_2 \to \theta_3 \to \theta_4 \to \theta_5 \to \cdots$$

   where $\theta_0 \sim p_0(\theta)$, $\theta_1 \sim p_1(\theta)$, $\cdots$, satisfying

$$p_t(\theta') = \int p_{t-1}(\theta) T(\theta \to \theta') \mathrm{d}\,\theta\ ,$$

   where $T(\theta \to \theta')$ is the Markov chain transition probability from θ to θ'.
3. We say $p^*(\theta)$ is an invariant (stationary) distribution of the Markov chain iff:

$$p^*(\theta') = \int p^*(\theta) T(\theta \to \theta') \mathrm{d}\,\theta$$

Markov chain Monte Carlo methods

1. We are interested in drawing samples from some desired distribution $p^*(\theta) = \frac{1}{Z}\tilde{p}^*(\theta)$.
2. Define a Markov chain:

$$\theta_0 \to \theta_1 \to \theta_2 \to \theta_3 \to \theta_4 \to \theta_5 \to \cdots$$

where $\theta_0 \sim p_0(\theta)$, $\theta_1 \sim p_1(\theta)$, $\cdots$, satisfying

$$p_t(\theta') = \int p_{t-1}(\theta) T(\theta \to \theta') \mathrm{d}\,\theta \ ,$$

where $T(\theta \to \theta')$ is the Markov chain transition probability from θ to θ'.

3. We say $p^*(\theta)$ is an invariant (stationary) distribution of the Markov chain iff:

$$p^*(\theta') = \int p^*(\theta) T(\theta \to \theta') \mathrm{d}\,\theta$$

Metropolis-Hasting algorithm

1. Design $T(\theta \to \theta')$ as the composition of a proposal distribution $q_t(\theta' \mid \theta)$ and an accept-reject mechanism.
2. At step t, draw a sample[1] $\theta^* \sim q_t(\theta \mid \theta_{t-1})$, and accept it with probability:

$$A_t(\theta^*, \theta_{t-1}) = \min\left(1, \frac{\tilde{p}(\theta^*)q_t(\theta_{t-1} \mid \theta^*)}{\tilde{p}(\theta_{t-1})q_t(\theta^* \mid \theta_{t-1})}\right)$$

3. The acceptance can be done by:
   - draw a random variable $u \sim \text{Uniform}(0, 1)$
   - accept the sample if $A_t(\theta^*, \theta_{t-1}) > u$
4. The corresponding transition kernel satisfies the detailed balance condition, thus has an invariant probability $p^*(\theta)$.

[1] A standard setting of $q_t(\theta \mid \theta_{t-1})$ is a normal distribution with mean θ_{t-1} and tunable variance.

Metropolis-Hasting algorithm

1. Design $T(\theta \to \theta')$ as the composition of a proposal distribution $q_t(\theta' \,|\, \theta)$ and an accept-reject mechanism.
2. At step t, draw a sample[1] $\theta^* \sim q_t(\theta \,|\, \theta_{t-1})$, and accept it with probability:

$$A_t(\theta^*, \theta_{t-1}) = \min\left(1, \frac{\tilde{p}(\theta^*) q_t(\theta_{t-1} \,|\, \theta^*)}{\tilde{p}(\theta_{t-1}) q_t(\theta^* \,|\, \theta_{t-1})}\right)$$

3. The acceptance can be done by:
   - draw a random variable $u \sim \mathrm{Uniform}(0, 1)$
   - accept the sample if $A_t(\theta^*, \theta_{t-1}) > u$
4. The corresponding transition kernel satisfies the detailed balance condition, thus has an invariant probability $p^*(\theta)$.

[1] A standard setting of $q_t(\theta \mid \theta_{t-1})$ is a normal distribution with mean θ_{t-1} and tunable variance.

Metropolis-Hasting algorithm

1. Design $T(\boldsymbol{\theta} \to \boldsymbol{\theta}')$ as the composition of a proposal distribution $q_t(\boldsymbol{\theta}' \,|\, \boldsymbol{\theta})$ and an accept-reject mechanism.
2. At step t, draw a sample[1] $\boldsymbol{\theta}^* \sim q_t(\boldsymbol{\theta} \,|\, \boldsymbol{\theta}_{t-1})$, and accept it with probability:

$$A_t(\boldsymbol{\theta}^*, \boldsymbol{\theta}_{t-1}) = \min\left(1, \frac{\tilde{p}(\boldsymbol{\theta}^*) q_t(\boldsymbol{\theta}_{t-1} \,|\, \boldsymbol{\theta}^*)}{\tilde{p}(\boldsymbol{\theta}_{t-1}) q_t(\boldsymbol{\theta}^* \,|\, \boldsymbol{\theta}_{t-1})}\right)$$

3. The acceptance can be done by:
   - draw a random variable $u \sim \text{Uniform}(0, 1)$
   - accept the sample if $A_t(\boldsymbol{\theta}^*, \boldsymbol{\theta}_{t-1}) > u$
4. The corresponding transition kernel satisfies the detailed balance condition, thus has an invariant probability $p^*(\theta)$.

[1] A standard setting of $q_t(\boldsymbol{\theta} \mid \boldsymbol{\theta}_{t-1})$ is a normal distribution with mean $\boldsymbol{\theta}_{t-1}$ and tunable variance.

Metroplis-Hasting algorithm

1. Design $T(\boldsymbol{\theta} \to \boldsymbol{\theta}')$ as the composition of a proposal distribution $q_t(\boldsymbol{\theta}' \mid \boldsymbol{\theta})$ and an accept-reject mechanism.
2. At step t, draw a sample[1] $\boldsymbol{\theta}^* \sim q_t(\boldsymbol{\theta} \mid \boldsymbol{\theta}_{t-1})$, and accept it with probability:

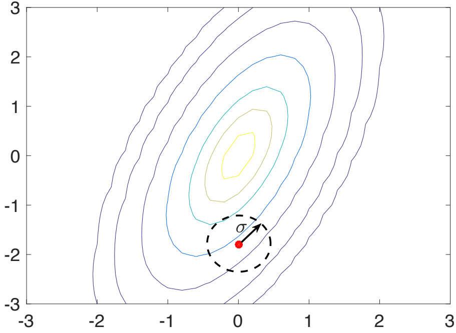
$$A_t(\boldsymbol{\theta}^*, \boldsymbol{\theta}_{t-1}) = \min\left(1, \frac{\tilde{p}(\boldsymbol{\theta}^*) q_t(\boldsymbol{\theta}_{t-1} \mid \boldsymbol{\theta}^*)}{\tilde{p}(\boldsymbol{\theta}_{t-1}) q_t(\boldsymbol{\theta}^* \mid \boldsymbol{\theta}_{t-1})}\right)$$

3. The acceptance can be done by:
   - draw a random variable $u \sim \text{Uniform}(0, 1)$
   - accept the sample if $A_t(\boldsymbol{\theta}^*, \boldsymbol{\theta}_{t-1}) > u$
4. The corresponding transition kernel satisfies the detailed balance condition, thus has an invariant probability $p^*(\boldsymbol{\theta})$.

[1] A standard setting of $q_t(\boldsymbol{\theta} \mid \boldsymbol{\theta}_{t-1})$ is a normal distribution with mean $\boldsymbol{\theta}_{t-1}$ and tunable variance.

Discussion on the proposal distribution

1. Standard proposal distribution is an isotropic Gaussian center at the current state with variance σ:
   - small σ leads to high acceptance rate, but moves too slowly
   - large σ moves fast, but leads to high rejection rate
2. How to choose better proposals?



Gibbs sampler

1. Assume θ is multi-dimensional[2], $\theta = (\theta_1, \cdots, \theta_k, \cdots, \theta_K)$, denote $\theta_{-k} \triangleq \{\theta_j : j \neq k\}$.
2. Sample θ_k sequentially, with proposal distribution being the true conditional distribution:

$$q_k(\theta^* \mid \theta) = p(\theta_k^* \mid \theta_{-k})$$

3. Note $\theta_{-k}^* = \theta_{-k}$, $p(\theta) = p(\theta_k \mid \theta_{-k}) p(\theta_{-k})$.
4. The MH acceptance probability is:

$$A(\theta^*, \theta) = \frac{p(\theta^*) q_k(\theta \mid \theta^*)}{p(\theta) q_k(\theta^* \mid \theta)} = \frac{p(\theta_k^* \mid \theta_{-k}^*) p(\theta_{-k}^*) p(\theta_k \mid \theta_{-k}^*)}{p(\theta_k^* \mid \theta_{-k}) p(\theta_{-k}) p(\theta_k \mid \theta_{-k})}$$
$$= 1$$

[2] One dimensional random variable is relatively easy to sample.

Gibbs sampler

1. Assume $\boldsymbol{\theta}$ is multi-dimensional[2], $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \cdots, \boldsymbol{\theta}_k, \cdots, \boldsymbol{\theta}_K)$, denote $\boldsymbol{\theta}_{-k} \triangleq \{\boldsymbol{\theta}_j : j \neq k\}$.
2. Sample $\boldsymbol{\theta}_k$ sequentially, with proposal distribution being the true conditional distribution:

$$q_k(\boldsymbol{\theta}^* \mid \boldsymbol{\theta}) = p(\boldsymbol{\theta}_k^* \mid \boldsymbol{\theta}_{-k})$$

3. Note $\theta^*_{-k} = \theta_{-k}$, $p(\theta) = p(\theta_k \mid \theta_{-k})p(\theta_{-k})$.
4. The MH acceptance probability is:

$$A(\theta^*, \theta) = \frac{p(\theta^*)q_k(\theta \mid \theta^*)}{p(\theta)q_k(\theta^* \mid \theta)} = \frac{p(\theta_k^* \mid \theta^*_{-k})p(\theta^*_{-k})p(\theta_k \mid \theta^*_{-k})}{p(\theta_k^* \mid \theta_{-k})p(\theta_{-k})p(\theta_k \mid \theta_{-k})}$$
$$= 1$$

[2] One dimensional random variable is relatively easy to sample.

Gibbs sampler

1. Assume $\boldsymbol{\theta}$ is multi-dimensional[2], $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \cdots, \boldsymbol{\theta}_k, \cdots, \boldsymbol{\theta}_K)$, denote $\boldsymbol{\theta}_{-k} \triangleq \{\boldsymbol{\theta}_j : j \neq k\}$.
2. Sample $\boldsymbol{\theta}_k$ sequentially, with proposal distribution being the true conditional distribution:

$$q_k(\boldsymbol{\theta}^* \mid \boldsymbol{\theta}) = p(\boldsymbol{\theta}_k^* \mid \boldsymbol{\theta}_{-k})$$

3. Note $\boldsymbol{\theta}^*_{-k} = \boldsymbol{\theta}_{-k}$, $p(\boldsymbol{\theta}) = p(\boldsymbol{\theta}_k \mid \boldsymbol{\theta}_{-k})p(\boldsymbol{\theta}_{-k})$.
4. The MH acceptance probability is:

$$\begin{aligned} A(\boldsymbol{\theta}^*, \boldsymbol{\theta}) &= \frac{p(\boldsymbol{\theta}^*)q_k(\boldsymbol{\theta} \mid \boldsymbol{\theta}^*)}{p(\boldsymbol{\theta})q_k(\boldsymbol{\theta}^* \mid \boldsymbol{\theta})} = \frac{p(\boldsymbol{\theta}_k^* \mid \boldsymbol{\theta}^*_{-k})p(\boldsymbol{\theta}^*_{-k})p(\boldsymbol{\theta}_k \mid \boldsymbol{\theta}^*_{-k})}{p(\boldsymbol{\theta}_k^* \mid \boldsymbol{\theta}_{-k})p(\boldsymbol{\theta}_{-k})p(\boldsymbol{\theta}_k \mid \boldsymbol{\theta}_{-k})} \\ &= 1 \end{aligned}$$

[2] One dimensional random variable is relatively easy to sample.

Discussion of Gibbs sampler

1. No accept-reject step, very efficient.
2. Conditional distributions are not always easy to sample.
3. May not mix well when in high-dimensional space with highly correlated variables.

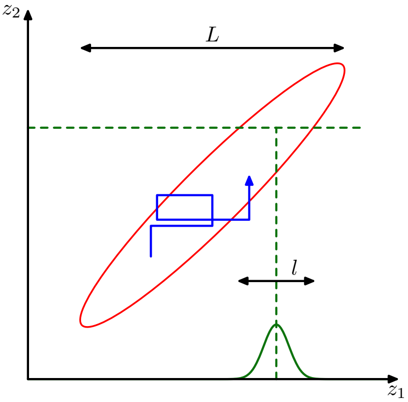
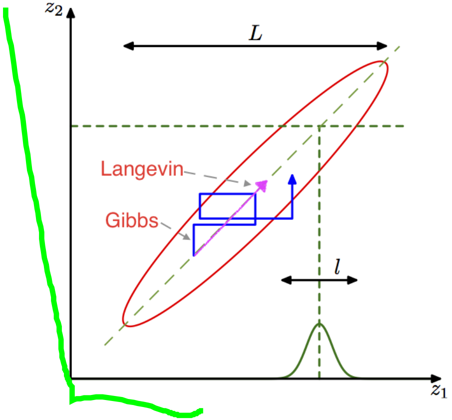


Figure: Sample path does not follow gradients. Figure from PRML, Bishop (2006)

The Metropolis-adjusted Langevin: a better proposal

1. Gibbs sampling travels the parameter space following a zipzag curve, which might be slow in high-dimensional space.
2. The Metropolis-adjusted Langevin uses a proposal that points directly to the center of the probabilistic contour.



The Metropolis-adjusted Langevin: a better proposal

1. Let $E(\theta) \triangleq -\log \tilde{p}(\theta)$, the direction of the contour is just the gradient: $-\nabla_{\boldsymbol{\theta}} E(\boldsymbol{\theta})$.
2. In iteration l, define the proposal as a Gaussian centering at $\boldsymbol{\theta}^* = \boldsymbol{\theta}_{l-1} - \nabla_{\boldsymbol{\theta}} E(\boldsymbol{\theta}_{l-1}) h_l$, where h_l is a small stepsize:

$$q(\boldsymbol{\theta}_l \mid \boldsymbol{\theta}_{l-1}) = N\left(\boldsymbol{\theta}_l; \boldsymbol{\theta}^*, \sigma^2\right) .$$

3. Need to do an accept-reject step:
   - calculate the acceptance probability:

$$A(\theta^*, \theta_{l-1}) = \frac{\tilde{p}(\theta^*) q(\theta_{l-1} \mid \theta^*)}{\tilde{p}(\theta) q(\theta^* \mid \theta_{l-1})}$$

   - accept θ^* with probability $A(\theta^*, \theta_{l-1})$, otherwise set $\theta_l = \theta_{l-1}$

The Metropolis-adjusted Langevin: a better proposal

1. Let $E(\boldsymbol{\theta}) \triangleq -\log \tilde{p}(\boldsymbol{\theta})$, the direction of the contour is just the gradient: $-\nabla_{\boldsymbol{\theta}} E(\boldsymbol{\theta})$.
2. In iteration l, define the proposal as a Gaussian centering at $\boldsymbol{\theta}^* = \boldsymbol{\theta}_{l-1} - \nabla_{\boldsymbol{\theta}} E(\boldsymbol{\theta}_{l-1}) h_l$, where h_l is a small stepsize:
$$q(\boldsymbol{\theta}_l \,|\, \boldsymbol{\theta}_{l-1}) = N\left(\boldsymbol{\theta}_l; \boldsymbol{\theta}^*, \sigma^2\right) .$$
3. Need to do an accept-reject step:
    - calculate the acceptance probability:
$$A(\boldsymbol{\theta}^*, \boldsymbol{\theta}_{l-1}) = \frac{\tilde{p}(\boldsymbol{\theta}^*) q(\boldsymbol{\theta}_{l-1} \,|\, \boldsymbol{\theta}^*)}{\tilde{p}(\boldsymbol{\theta}) q(\boldsymbol{\theta}^* \,|\, \boldsymbol{\theta}_{l-1})}$$
    - accept $\boldsymbol{\theta}^*$ with probability $A(\boldsymbol{\theta}^*, \boldsymbol{\theta}_{l-1})$, otherwise set $\boldsymbol{\theta}_l = \boldsymbol{\theta}_{l-1}$

Basically: SGD step with a

Hamiltonian Monte Carlo

Frictionless ball rolling:

1. A dynamic system with total energy or Hamiltonian: $H = E(\boldsymbol{\theta}) + K(\mathbf{v})$, where $E(\boldsymbol{\theta}) \triangleq -\log \tilde{p}(\boldsymbol{\theta})$, $K(\mathbf{v}) \triangleq \mathbf{v}^T \mathbf{v} / 2$.
2. Hamiltonian's equation describes the equations of motion of the ball:

$$\frac{\mathrm{d}\,\boldsymbol{\theta}}{\mathrm{d}t} = \frac{\partial H}{\partial \mathbf{v}} = \mathbf{v}$$
$$\frac{\mathrm{d}\,\mathbf{v}}{\mathrm{d}t} = -\frac{\partial H}{\partial \boldsymbol{\theta}} = \frac{\partial \log \tilde{p}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}$$

3. Joint distribution: $p(\boldsymbol{\theta}, \mathbf{v}) \propto e^{-H(\boldsymbol{\theta}, \mathbf{v})}$.

Figure: Rolling ball. Movie from Matthias Liepe

Hamiltonian Monte Carlo

Frictionless ball rolling:

1. A dynamic system with total energy or Hamiltonian: $H = E(\boldsymbol{\theta}) + K(\mathbf{v})$, where $E(\boldsymbol{\theta}) \triangleq -\log \tilde{p}(\boldsymbol{\theta})$, $K(\mathbf{v}) \triangleq \mathbf{v}^T \mathbf{v} / 2$.
2. Hamiltonian's equation describes the equations of motion of the ball:

$$\frac{\mathrm{d}\,\boldsymbol{\theta}}{\mathrm{d}t} = \frac{\partial H}{\partial\,\mathbf{v}} = \mathbf{v}$$

$$\frac{\mathrm{d}\,\mathbf{v}}{\mathrm{d}t} = -\frac{\partial H}{\partial\,\boldsymbol{\theta}} = \frac{\partial \log \tilde{p}(\boldsymbol{\theta})}{\partial\,\boldsymbol{\theta}}$$

3. Joint distribution: $p(\boldsymbol{\theta}, \mathbf{v}) \propto e^{-H(\boldsymbol{\theta}, \mathbf{v})}$.

Figure: Rolling ball. Movie from Matthias Liepe

Hamiltonian Monte Carlo

Frictionless ball rolling:

1. A dynamic system with total energy or Hamiltonian: $H = E(\boldsymbol{\theta}) + K(\mathbf{v})$, where $E(\boldsymbol{\theta}) \triangleq -\log \tilde{p}(\boldsymbol{\theta})$, $K(\mathbf{v}) \triangleq \mathbf{v}^T \mathbf{v} / 2$.
2. Hamiltonian's equation describes the equations of motion of the ball:

$$\frac{\mathrm{d}\,\boldsymbol{\theta}}{\mathrm{d}t} = \frac{\partial H}{\partial \mathbf{v}} = \mathbf{v}$$
$$\frac{\mathrm{d}\,\mathbf{v}}{\mathrm{d}t} = -\frac{\partial H}{\partial \boldsymbol{\theta}} = \frac{\partial \log \tilde{p}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}$$

3. Joint distribution: $p(\boldsymbol{\theta}, \mathbf{v}) \propto e^{-H(\boldsymbol{\theta}, \mathbf{v})}$.

Figure: Rolling ball. Movie from Matthias Liepe

Solving Hamiltonian dynamics

1. Solving the continuous-time differential equation with discretized-time approximation:

$$\begin{cases} \mathrm{d}\,\boldsymbol{\theta} &= \mathbf{v}\,\mathrm{d}t \\ \mathrm{d}\,\mathbf{v} &= \nabla_{\boldsymbol{\theta}} \log \tilde{p}(\boldsymbol{\theta})\mathrm{d}t \end{cases} \Longrightarrow \begin{cases} \boldsymbol{\theta}_l &= \boldsymbol{\theta}_{l-1} + \mathbf{v}_{l-1}\, h_l \\ \mathbf{v}_l &= \mathbf{v}_{l-1} + \nabla_{\boldsymbol{\theta}} \log \tilde{p}(\boldsymbol{\theta}_l) h_l \end{cases}$$

   - proposals follow historical gradients of the distribution contour

2. Need an accept-reject test to design whether accept the proposal, because of the discretization error:
   - proposal is deterministic
   - acceptance probability: $\min\left(1, \exp\left\{H(\boldsymbol{\theta}_l, \mathbf{v}_l) - H(\boldsymbol{\theta}_{l+1}, \mathbf{v}_{l+1})\right\}\right)$

3. Almost identical to SGD with momentum:
   - $\begin{cases} \boldsymbol{\theta}_l &= \boldsymbol{\theta}_{l-1} + \mathbf{p}_{l-1} \\ \mathbf{p}_l &= (1-m)\,\mathbf{p}_{l-1} + \nabla_{\boldsymbol{\theta}} \log \tilde{p}(\boldsymbol{\theta}_l)\epsilon_l \end{cases}$
   - they will be make equivalent in the context of stochastic gradient MCMC

Solving Hamiltonian dynamics

1. Solving the continuous-time differential equation with discretized-time approximation:

$$\left\{\begin{array}{ll} \mathrm{d}\,\boldsymbol{\theta} & = \mathbf{v}\,\mathrm{d}t \\ \mathrm{d}\,\mathbf{v} & = \nabla_{\boldsymbol{\theta}} \log \tilde{p}(\boldsymbol{\theta})\mathrm{d}t \end{array}\right. \Longrightarrow \left\{\begin{array}{ll} \boldsymbol{\theta}_l & = \boldsymbol{\theta}_{l-1} + \mathbf{v}_{l-1}\, h_l \\ \mathbf{v}_l & = \mathbf{v}_{l-1} + \nabla_{\boldsymbol{\theta}} \log \tilde{p}(\boldsymbol{\theta}_l) h_l \end{array}\right.$$

   - proposals follow historical gradients of the distribution contour

2. Need an accept-reject test to design whether accept the proposal, because of the discretization error:
   - proposal is deterministic
   - acceptance probability: $\min\left(1, \exp\left\{H(\boldsymbol{\theta}_l, \mathbf{v}_l) - H(\boldsymbol{\theta}_{l+1}, \mathbf{v}_{l+1})\right\}\right)$

3. Almost identical to SGD with momentum:
   - $\left\{\begin{array}{ll} \theta_l & = \theta_{l-1} + \mathbf{p}_{l-1} \\ \mathbf{p}_l & = (1-m)\,\mathbf{p}_{l-1} + \nabla_{\theta} \log \tilde{p}(\theta_l)\epsilon_l \end{array}\right.$
   - they will be make equivalent in the context of stochastic gradient MCMC

Solving Hamiltonian dynamics

Hamiltonian's equation.

1. Solving the continuous-time differential equation with discretized-time approximation:

$$\begin{cases} \mathrm{d}\,\boldsymbol{\theta} &= \mathbf{v}\,\mathrm{d}t \\ \mathrm{d}\,\mathbf{v} &= \nabla_{\boldsymbol{\theta}} \log \tilde{p}(\boldsymbol{\theta})\mathrm{d}t \end{cases} \Longrightarrow \begin{cases} \boldsymbol{\theta}_l &= \boldsymbol{\theta}_{l-1} + \mathbf{v}_{l-1}\, h_l \\ \mathbf{v}_l &= \mathbf{v}_{l-1} + \nabla_{\boldsymbol{\theta}} \log \tilde{p}(\boldsymbol{\theta}_l) h_l \end{cases}$$

   - proposals follow historical gradients of the distribution contour

2. Need an accept-reject test to design whether accept the proposal, because of the discretization error:
   - proposal is deterministic
   - acceptance probability: $\min\left(1, \exp\left\{H(\boldsymbol{\theta}_l, \mathbf{v}_l) - H(\boldsymbol{\theta}_{l+1}, \mathbf{v}_{l+1})\right\}\right)$

3. Almost identical to SGD with momentum:
   - $\begin{cases} \boldsymbol{\theta}_l &= \boldsymbol{\theta}_{l-1} + \mathbf{p}_{l-1} \\ \mathbf{p}_l &= (1-m)\,\mathbf{p}_{l-1} + \nabla_{\boldsymbol{\theta}} \log \tilde{p}(\boldsymbol{\theta}_l)\epsilon_l \end{cases}$
   - they will be make equivalent in the context of stochastic gradient MCMC

Demo: MH vs. HMC

1. Nine mixtures of Gaussians[3].
2. Sequential of samples connected by yellow lines.

[3] Demo by T. Broderick and D. Duvenaud.

Recap

1. Bayesian sampling with traditional MCMC methods, in each iteration:
   - generate a candidate sample from a proposal distribution
   - calculate the acceptance probability
   - accept or reject the proposed sample

Discussion

1. All the above traditional MCMC methods are not scalable in a big-data setting[4], in each iteration:
   - the whole data need to be used to generate a proposal
   - the whole data need to be used to calculate the acceptance probability
   - scales $O(N)$, where N is the number of data samples
2. Scalable MCMC uses sub-data in each iteration,
   - to calculate the acceptance probability[5]
   - to generate proposals, and ignore the acceptance step – stochastic gradient MCMC methods (SG-MCMC)

[4] when the number of data samples are large.

[5] A. Korattikara, Y. Chen, and M. Welling. "Austerity in MCMC Land: Cutting the Metropolis-Hastings Budget". In: *ICML*. 2014; R. Bardenet, A. Doucet, and C. Holmes. "Towards scaling up Markov chain Monte Carlo: an adaptive subsampling approach". In: *ICML*. 2014.

Discussion

1. All the above traditional MCMC methods are not scalable in a big-data setting[4], in each iteration:
   - the whole data need to be used to generate a proposal
   - the whole data need to be used to calculate the acceptance probability
   - scales $O(N)$, where N is the number of data samples
2. Scalable MCMC uses sub-data in each iteration,
   - to calculate the acceptance probability[5]
   - to generate proposals, and ignore the acceptance step – stochastic gradient MCMC methods (SG-MCMC)

mini-batches

[4] when the number of data samples are large.

[5] A. Korattikara, Y. Chen, and M. Welling. "Austerity in MCMC Land: Cutting the Metropolis-Hastings Budget". In: *ICML*. 2014; R. Bardenet, A. Doucet, and C. Holmes. "Towards scaling up Markov chain Monte Carlo: an adaptive subsampling approach". In: *ICML*. 2014.

Outline

Two key steps in SG-MCMC

1. Proposals typically follow stochastic gradients of log-posteriors:
   - make samples concentrate on the modes
2. Adding random Gaussian noise to proposals.
   - encourage algorithms to jump out of local modes, and to explore the parameter space
   - the noise in stochastic gradients not sufficient to make the algorithm move around parameter space

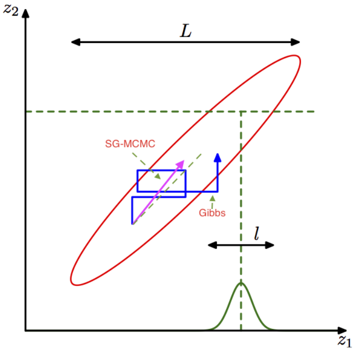


Figure: Proposals of Gibbs and SG-MCMC.

Two key steps in SG-MCMC

1. Proposals typically follow stochastic gradients of log-posteriors:
   - make samples concentrate on the modes
2. Adding random Gaussian noise to proposals.
   - encourage algorithms to jump out of local modes, and to explore the parameter space
   - the noise in stochastic gradients not sufficient to make the algorithm move around parameter space

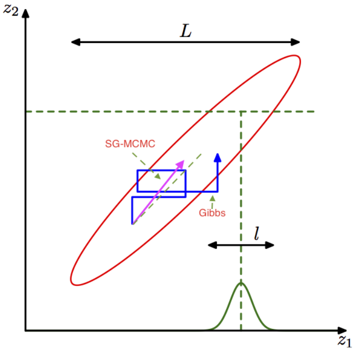


Figure: Proposals of Gibbs and SG-MCMC.

Basic setup

1. Given data $\mathbf{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_N\}$, a generative model (likelihood) $p(\mathbf{X} | \theta) = \prod_{i=1}^N p(\mathbf{x}_i \mid \boldsymbol{\theta})$ and prior $p(\boldsymbol{\theta})$, we want to sample from the posterior:

$$p(\boldsymbol{\theta} \mid \mathbf{X}) \propto p(\boldsymbol{\theta}) p(\mathbf{X} \mid \boldsymbol{\theta}) = p(\boldsymbol{\theta}) \prod_{i=1}^N p(\mathbf{x}_i \mid \boldsymbol{\theta})$$

2. We are interested in the case when N is extremely large, so that computing $p(\mathbf{X} \mid \theta)$ is prohibitively expensive.
3. Define the following two quantities (unnormalized log-posterior and stochastic unnormalized log-posterior):

$$U(\theta) \triangleq -\sum_{i=1}^N \log p(\mathbf{x}_i \mid \theta) - \log p(\theta)$$

$$\tilde{U}(\boldsymbol{\theta}) \triangleq -\frac{N}{n} \sum_{i=1}^n \log p(\mathbf{x}_{\pi_i} \mid \theta) - \log p(\theta)$$

where $(\pi_1, \cdots, \pi_N)$ is a random permutation of $(1, \cdots, N)$.

Basic setup

1. Given data $\mathbf{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_N\}$, a generative model (likelihood) $p(\mathbf{X}\,|\theta) = \prod_{i=1}^{N} p(\mathbf{x}_i\,|\,\boldsymbol{\theta})$ and prior $p(\boldsymbol{\theta})$, we want to sample from the posterior:

$$p(\boldsymbol{\theta}\,|\,\mathbf{X}) \propto p(\boldsymbol{\theta})p(\mathbf{X}\,|\,\boldsymbol{\theta}) = p(\boldsymbol{\theta}) \prod_{i=1}^{N} p(\mathbf{x}_i\,|\,\boldsymbol{\theta})$$

2. We are interested in the case when N is extremely large, so that computing $p(\mathbf{X}\,|\,\boldsymbol{\theta})$ is prohibitively expensive.
3. Define the following two quantities (unnormalized log-posterior and stochastic unnormalized log-posterior):

$$U(\theta) \triangleq -\sum_{i=1}^{N} \log p(\mathbf{x}_i\,|\,\boldsymbol{\theta}) - \log p(\boldsymbol{\theta})$$

$$\tilde{U}(\boldsymbol{\theta}) \triangleq -\frac{N}{n} \sum_{i=1}^{n} \log p(\mathbf{x}_{\pi_i}\,|\,\boldsymbol{\theta}) - \log p(\boldsymbol{\theta})$$

where $(\pi_1, \cdots, \pi_N)$ is a random permutation of $(1, \cdots, N)$.

Basic setup

1. SG-MCMC relies on the following quantity (stochastic gradient):

$$\nabla_{\boldsymbol{\theta}} \tilde{U}(\boldsymbol{\theta}) \triangleq -\frac{N}{n} \sum_{i=1}^{n} \nabla_{\boldsymbol{\theta}} \log p(\mathbf{x}_{\pi_i} \,|\, \boldsymbol{\theta}) - \nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{\theta}) ~,$$

2. $\nabla_{\theta} \tilde{U}(\theta)$ is an unbiased estimate of $\nabla_{\theta} U(\theta)$:
   - SG-MCMC samples parameters based on $\nabla_{\theta} \tilde{U}(\theta)$
   - very cheap to compute
   - bringing the name "stochastic gradient MCMC"

Basic setup

1. SG-MCMC relies on the following quantity (stochastic gradient):

$$\nabla_{\boldsymbol{\theta}} \tilde{U}(\boldsymbol{\theta}) \triangleq -\frac{N}{n} \sum_{i=1}^{n} \nabla_{\boldsymbol{\theta}} \log p(\mathbf{x}_{\pi_i} \mid \boldsymbol{\theta}) - \nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{\theta}) \ ,$$

2. $\nabla_{\boldsymbol{\theta}} \tilde{U}(\boldsymbol{\theta})$ is an unbiased estimate of $\nabla_{\boldsymbol{\theta}} U(\boldsymbol{\theta})$:
   - SG-MCMC samples parameters based on $\nabla_{\boldsymbol{\theta}} \tilde{U}(\boldsymbol{\theta})$
   - very cheap to compute
   - bringing the name "stochastic gradient MCMC"

Comparing with traditional MCMC

1. Ignore the acceptance step:
   - the detailed balance condition typically not hold, and the algorithm is not reversible[6]
   - typically leads to biased, but controllable estimations
2. Use sub-data in each iteration:
   - yielding stochastic gradients
   - does not affect the convergence properties (*e.g.*, convergence rates), compared to using the whole data in each iteration

[6] These are sufficient conditions for a valid MCMC method, but not necessary conditions.

Comparing with traditional MCMC

1. Ignore the acceptance step:
   - the detailed balance condition typically not hold, and the algorithm is not reversible[6]
   - typically leads to biased, but controllable estimations
2. Use sub-data in each iteration:
   - yielding stochastic gradients
   - does not affect the convergence properties (*e.g.*, convergence rates), compared to using the whole data in each iteration

[6] These are sufficient conditions for a valid MCMC method, but not necessary conditions.

Demo: the two key steps

1. Proposals follow stochastic gradients of log-posteriors:
   - stuck in a local mode

Demo: the two key steps

1. After adding random Gaussian noise:
    - it works !!

Outline

First attempt

1. A 1st-order method: stochastic gradients directly applied on the model parameter $\boldsymbol{\theta}$.
2. Use a proposal that follows the stochastic gradient of the log-posterior:

$$\boldsymbol{\theta}_{l+1} = \boldsymbol{\theta}_l - h_{l+1} \nabla_{\boldsymbol{\theta}} \tilde{U}(\boldsymbol{\theta}_l)$$

   - h_l's are the stepsizes, could be fixed ($\forall l, h_l = h$) or deceasing ($\forall l, h_l > h_{l+1}$)
3. Ignore the acceptance step.
4. Resulting in Stochastic Gradient Descend (SGD).

Random noise to the rescue

1. Need to make the algorithm explore the parameter space:
   - adding random Gaussian noise to the update[7]

$$\begin{aligned} \boldsymbol{\theta}_{l+1} &= \boldsymbol{\theta}_l - h_{l+1} \nabla_{\boldsymbol{\theta}} \tilde{U}(\theta_l) + \sqrt{2 h_{l+1}} \zeta_{l+1} \\ \zeta_{l+1} &\sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \end{aligned}$$

2. The magnitude of the Gaussian needs to be $\sqrt{2h_{l+1}}$ in order to guarantee a correct sampler:
   - guaranteed by the Fokker-Planck Equation
3. This is called stochastic gradient Langevin dynamics (SGLD).

[7] In the following, we will directly use $\mathcal{N}(\mathbf{0}, \mathbf{I})$ to represent a normal random variable with zero-mean and covariance matrix $\mathbf{I}$.

Random noise to the rescue

Gaussian noise with a variance given by th

1. Need to make the algorithm explore the parameter space:
   - adding random Gaussian noise to the update[7]

$$\boldsymbol{\theta}_{l+1} = \boldsymbol{\theta}_l - h_{l+1}\nabla_{\boldsymbol{\theta}}\tilde{U}(\theta_l) + \sqrt{2h_{l+1}}\zeta_{l+1}$$
$$\zeta_{l+1} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

2. The magnitude of the Gaussian needs to be $\sqrt{2h_{l+1}}$ in order to guarantee a correct sampler:
   - guaranteed by the Fokker-Planck Equation
3. This is called stochastic gradient Langevin dynamics (SGLD).

Basically: SGD + properly scaled Gaussian noise.

[7] In the following, we will directly use $\mathcal{N}(\mathbf{0}, \mathbf{I})$ to represent a normal random variable with zero-mean and covariance matrix $\mathbf{I}$.

SGLD in algorithm

Input: Parameters $\{h_l\}$
Output: Approximate samples $\{\boldsymbol{\theta}_l\}$

Initialize $\boldsymbol{\theta}_0 \in \mathbb{R}^n$
for $l = 1, 2, \ldots$ **do**

  Evaluate $\nabla_{\boldsymbol{\theta}} \tilde{U}(\boldsymbol{\theta}_{l-1})$ from the l-th minibatch

  $\boldsymbol{\theta}_l = \boldsymbol{\theta}_{l-1} - \nabla \tilde{U}(\boldsymbol{\theta}_{l-1}) h_l + \sqrt{2h_l}\,\mathcal{N}(\mathbf{0}, \mathbf{I})$

end
Return $\{\boldsymbol{\theta}_l\}$

Algorithm 1: Stochastic Gradient Langevin Dynamics

Example[8]

1. A simple Gaussian mixture:

$$\theta_1 \sim \mathcal{N}(0, 10), \quad \boldsymbol{\theta}_2 \sim \mathcal{N}(0, 1)$$

$$x_i \sim \frac{1}{2}\mathcal{N}(\boldsymbol{\theta}_1, 2) + \frac{1}{2}\mathcal{N}(\boldsymbol{\theta}_1 + \boldsymbol{\theta}_2, 2), \quad i = 1, \cdots, 100$$

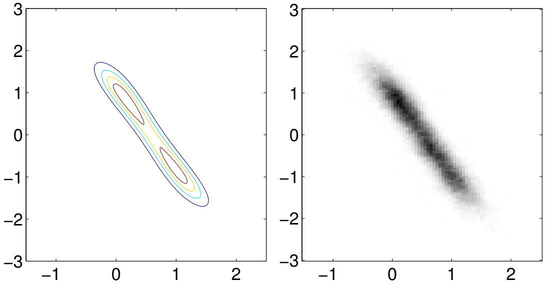


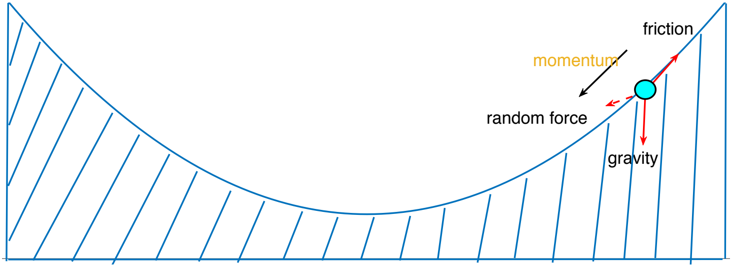
Figure: Left: true posterior; Right: sample-based estimation.

[8] M. Welling and Y. W. Teh. "Bayesian learning via stochastic gradient Langevin dynamics". In: *ICML*. 2011.

Outline

SGHMC

1. A 2nd-order method: stochastic gradients applied on some auxiliary parameters (momentum).
2. SGLD is slow when parameter space exhibits uneven curvatures.
3. Use the momentum idea to improve SGLD:
   - a generalization of the HMC, in that the ball is rolling on a friction surface
   - the ball follows the momentum instead of gradients, which is a summarization of historical gradients, thus could jump out local modes easier and move faster
   - needs a balance between these extra forces



Adding a friction term

1. Without a friction term, the random Gaussian noise would drive the ball too far away from their stationary distribution.
2. After adding a friction term:

$$\theta_l = \theta_{l-1} + \mathbf{v}_{l-1}\, h_l$$
$$\mathbf{v}_l = \mathbf{v}_{l-1} - \nabla_\theta \tilde{U}(\theta_l) h_l - A\,\mathbf{v}_{l-1}\, h_l + \sqrt{2Ah_l}\,\mathcal{N}(\mathbf{0}, \mathbf{I})\ ,$$

where $A > 0$ is a constant[9], controlling the magnitude of the friction.

3. The fraction term penalize the momentum:
   - the more momentum, the more fraction it has, thus slowing down the ball

[9] In the original SGHMC paper, A is decomposed into a known variance of injected noise and an unknown variance of stochastic gradients.

Adding a friction term

1. Without a friction term, the random Gaussian noise would drive the ball too far away from their stationary distribution.
2. After adding a friction term:

$$\begin{aligned}\boldsymbol{\theta}_l &= \boldsymbol{\theta}_{l-1} + \mathbf{v}_{l-1}\, h_l \\ \mathbf{v}_l &= \mathbf{v}_{l-1} - \nabla_{\boldsymbol{\theta}} \tilde{U}(\boldsymbol{\theta}_l) h_l - A\, \mathbf{v}_{l-1}\, h_l + \sqrt{2Ah_l}\, \mathcal{N}(\mathbf{0}, \mathbf{I})\ ,\end{aligned}$$

   where $A > 0$ is a constant[9], controlling the magnitude of the friction.
3. The fraction term penalize the momentum:
   - the more momentum, the more fraction it has, thus slowing down the ball

[9] In the original SGHMC paper, A is decomposed into a known variance of injected noise and an unknown variance of stochastic gradients.

Adding a friction term

1. Without a friction term, the random Gaussian noise would drive the ball too far away from their stationary distribution.
2. After adding a friction term:

$$\boxed{\begin{aligned}\theta_l &= \theta_{l-1} + \mathbf{v}_{l-1}\, h_l \\ \mathbf{v}_l &= \mathbf{v}_{l-1} - \nabla_\theta \tilde{U}(\theta_l) h_l - A\,\mathbf{v}_{l-1}\, h_l + \sqrt{2Ah_l}\,\mathcal{N}(\mathbf{0}, \mathbf{I})\ ,\end{aligned}}$$

where $A > 0$ is a constant[9], controlling the magnitude of the friction.
3. The fraction term penalize the momentum:
   - the more momentum, the more fraction it has, thus slowing down the ball

friction*

[9] In the original SGHMC paper, A is decomposed into a known variance of injected noise and an unknown variance of stochastic gradients.

SGHMC in algorithm

Input: Parameters $A, \{h_l\}$
Output: Approximate samples $\{\boldsymbol{\theta}_l\}$

Initialize $\boldsymbol{\theta}_0 \in \mathbb{R}^n$
for $l = 1, 2, \ldots$ **do**

  Evaluate $\nabla_{\boldsymbol{\theta}} \tilde{U}(\boldsymbol{\theta}_{l-1})$ from the l-th minibatch

  $\boldsymbol{\theta}_l = \boldsymbol{\theta}_{l-1} + \mathbf{v}_{l-1}\, h_l$

  $\mathbf{v}_l = \mathbf{v}_{l-1} - \nabla \tilde{U}(\boldsymbol{\theta}_l) h_l - A\, \mathbf{v}_{l-1}\, h_l + \sqrt{2Ah_l}\, \mathcal{N}(\mathbf{0}, \mathbf{I})$

end
Return $\{\boldsymbol{\theta}_l\}$

Algorithm 2: Stochastic Gradient Hamiltonian Monte Carlo

Reparametrize SGHMC

for $l = 1, 2, \ldots$ **do**

Evaluate $\nabla_{\boldsymbol{\theta}} \tilde{U}(\boldsymbol{\theta}_{l-1})$ from the l-th minibatch

$\boldsymbol{\theta}_l = \boldsymbol{\theta}_{l-1} + \mathbf{v}_{l-1}\, h_l$

$\mathbf{v}_l = \mathbf{v}_{l-1} - \nabla \tilde{U}(\boldsymbol{\theta}_l) h_l - A\,\mathbf{v}_{l-1}\, h_l + \sqrt{2Ah_l}\,\mathcal{N}(\mathbf{0}, \mathbf{I})$

end

- Reparametrization: $\epsilon = h^2$, $m = Ah$, $\mathbf{p} = \mathbf{v}\, h$

Reparametrize SGHMC

for $l = 1, 2, \ldots$ **do**

Evaluate $\nabla_{\boldsymbol{\theta}} \tilde{U}(\boldsymbol{\theta}_{l-1})$ from the l-th minibatch

$\boldsymbol{\theta}_l = \boldsymbol{\theta}_{l-1} + \mathbf{v}_{l-1}\, h_l$

$\mathbf{v}_l = \mathbf{v}_{l-1} - \nabla \tilde{U}(\boldsymbol{\theta}_l) h_l - A\, \mathbf{v}_{l-1}\, h_l + \sqrt{2Ah_l}\, \mathcal{N}(\mathbf{0}, \mathbf{I})$

end

for $l = 1, 2, \ldots$ **do**

Evaluate $\nabla_{\boldsymbol{\theta}} \tilde{U}(\boldsymbol{\theta}_{l-1})$ from the l-th minibatch

$\boldsymbol{\theta}_l = \boldsymbol{\theta}_{l-1} + \mathbf{p}_{l-1}$

$\mathbf{p}_l = (1 - m)\, \mathbf{p}_{l-1} - \nabla \tilde{U}(\boldsymbol{\theta}_l) \epsilon_l + \sqrt{2m\epsilon_l}\, \mathcal{N}(\mathbf{0}, \mathbf{I})$

end

- Reparametrization: $\epsilon = h^2$, $m = Ah$, $\mathbf{p} = \mathbf{v}\, h$

Reparametrize SGHMC

for $l = 1, 2, \ldots$ **do**

Evaluate $\nabla_{\boldsymbol{\theta}} \tilde{U}(\boldsymbol{\theta}_{l-1})$ from the l-th minibatch

$\boldsymbol{\theta}_l = \boldsymbol{\theta}_{l-1} + \mathbf{v}_{l-1}\, h_l$

$\mathbf{v}_l = \mathbf{v}_{l-1} - \nabla \tilde{U}(\boldsymbol{\theta}_l) h_l - A\, \mathbf{v}_{l-1}\, h_l + \sqrt{2Ah_l}\, \mathcal{N}(\mathbf{0}, \mathbf{I})$

end

for $l = 1, 2, \ldots$ **do**

Evaluate $\nabla_{\boldsymbol{\theta}} \tilde{U}(\boldsymbol{\theta}_{l-1})$ from the l-th minibatch

$\boldsymbol{\theta}_l = \boldsymbol{\theta}_{l-1} + \mathbf{p}_{l-1}$

$\mathbf{p}_l = (1 - m)\, \mathbf{p}_{l-1} - \nabla \tilde{U}(\boldsymbol{\theta}_l) \epsilon_l + \sqrt{2m\epsilon_l}\, \mathcal{N}(\mathbf{0}, \mathbf{I})$

end

- Reparametrization: $\epsilon = h^2$, $m = Ah$, $\mathbf{v} = \mathbf{p}\, h$
- ϵ_l: learning rate; m: momentum weight

SGD vs. SGLD

Just added properly scal

$$\nabla_{\boldsymbol{\theta}} \tilde{U}(\boldsymbol{\theta}_{l-1}) \triangleq -\frac{N}{n} \sum_{i=1}^{n} \nabla_{\boldsymbol{\theta}} \log p(\mathbf{x}_{\pi_i} \mid \boldsymbol{\theta}_{l-1}) - \nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{\theta}_{l-1}) ,$$

SGD:

for $l = 1, 2, \ldots$ **do**

  Evaluate $\nabla_{\boldsymbol{\theta}} \tilde{U}(\boldsymbol{\theta}_{l-1})$ from the l-th minibatch

  $\boldsymbol{\theta}_l = \boldsymbol{\theta}_{l-1} - \nabla \tilde{U}(\boldsymbol{\theta}_l)\epsilon_l$

end

SGLD:

for $l = 1, 2, \ldots$ **do**

  Evaluate $\nabla_{\boldsymbol{\theta}} \tilde{U}(\boldsymbol{\theta}_{l-1})$ from the l-th minibatch

  $\boldsymbol{\theta}_l = \boldsymbol{\theta}_{l-1} - \nabla \tilde{U}(\boldsymbol{\theta}_l)\epsilon_l + \delta_l$

  $\delta_l \sim \mathcal{N}(\mathbf{0}, 2\epsilon_l \mathbf{I})$

end

SGD with Momentum (SGD-M) vs. SGHMC

Just added properly scale

$$\nabla_{\boldsymbol{\theta}} \tilde{U}(\boldsymbol{\theta}_{l-1}) \triangleq -\frac{N}{n} \sum_{i=1}^{n} \nabla_{\boldsymbol{\theta}} \log p(\mathbf{x}_{\pi_i} \mid \boldsymbol{\theta}_{l-1}) - \nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{\theta}_{l-1}) ,$$

SGD-M:

for $l = 1, 2, \ldots$ **do**

Evaluate $\nabla_{\boldsymbol{\theta}} \tilde{U}(\boldsymbol{\theta}_{l-1})$ from the l-th minibatch

$\boldsymbol{\theta}_l = \boldsymbol{\theta}_{l-1} + \mathbf{p}_{l-1}$

$\mathbf{p}_l = (1 - m)\, \mathbf{p}_{l-1} - \nabla \tilde{U}(\boldsymbol{\theta}_l) \epsilon_l$

end

SGHMC:

for $l = 1, 2, \ldots$ **do**

Evaluate $\nabla_{\boldsymbol{\theta}} \tilde{U}(\boldsymbol{\theta}_{l-1})$ from the l-th minibatch

$\boldsymbol{\theta}_l = \boldsymbol{\theta}_{l-1} + \mathbf{p}_{l-1}$

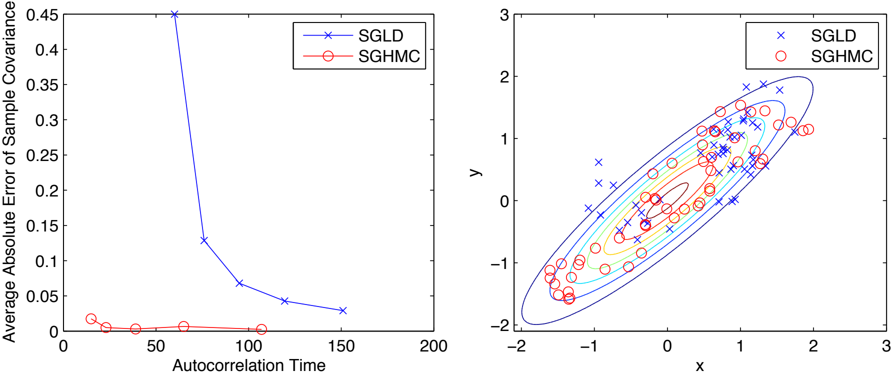
$\mathbf{p}_l = (1 - m)\, \mathbf{p}_{l-1} - \nabla \tilde{U}(\boldsymbol{\theta}_l) \epsilon_l + \delta_l$

$\delta_l \sim \mathcal{N}(\mathbf{0}, 2m\epsilon_l \mathbf{I})$

end

Example[10]

1. Sample from a 2D Gaussian distribution:
   - $U(\boldsymbol{\theta}) = \frac{1}{2}\boldsymbol{\theta}^T \Sigma^{-1} \boldsymbol{\theta}$



[10] T. Chen, E. B. Fox, and C. Guestrin. "Stochastic Gradient Hamiltonian Monte Carlo". In: *ICML*. 2014.

Recap

1. For SG-MCMC methods, in each iteration:
   - calculate the stochastic gradient based on the current parameter sample
   - generate the next sample by moving the current sample (probably in an extended space) along the direction of the stochastic gradient, plus a suitable random Gaussian noise
   - no need for accept-reject
   - guaranteed to converge close to the true posterior in some sense

Outline

Latent Dirichlet allocation

1. For each topic k, draw the topic-word distribution:

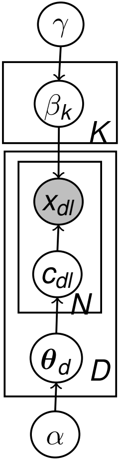
$$\beta_k \sim \mathrm{Dir}(\gamma)$$

2. For each document d, draw its topic distribution: $\boldsymbol{\theta}_d \sim \mathrm{Dir}(\alpha)$
   - For each word l, draw its topic indicator:

$$c_{dl} \sim \mathrm{Discrete}(\boldsymbol{\theta}_d)$$

   - Draw the observed word:

$$x_{dl} \sim \mathrm{Discrete}(\beta_{c_{dl}})$$



Latent Dirichlet allocation

1. Let $\beta \triangleq (\beta_k)_{k=1}^K$, $\boldsymbol{\theta} \triangleq (\boldsymbol{\theta}_d)_{d=1}^D$, $\mathbf{C} \triangleq (c_{dl})_{d,l=1}^{D,n_d}$, $\mathbf{X} \triangleq (x_{dl})_{d,l=1}^{D,n_d}$, the posterior distribution

$$p(\beta, \boldsymbol{\theta}, \mathbf{C} \,|\, \mathbf{X}) \propto \left[\prod_{k=1}^K p(\beta_k|\gamma)\right] \left[\prod_{d=1}^D p(\boldsymbol{\theta}_d \,|\alpha) \prod_{l=1}^{n_d} p(c_{dl}|\theta_d) p(x_{dl}|\beta, c_{dl})\right]$$

2. From previous lectures:

$$p(c_{dl}|\,\boldsymbol{\theta}_d) = \prod_{k=1}^K (\boldsymbol{\theta}_{dk})^{1(c_{dl}=k)}$$

$$p(x_{dl}|\,\boldsymbol{\theta}, c_{dl}) = \prod_{k=1}^K \prod_{v=1}^V \beta_{kv}^{1(x_{dl}=v)1(c_{dl}=k)}$$

3. Together with the fact:

$$\int_{\boldsymbol{\theta}\in\triangle_{K-1}} \prod_{k=1}^K \theta_k^{\alpha_k-1} \,\mathrm{d}\,\boldsymbol{\theta}_k = \frac{\prod_{k=1}^K \Gamma(\alpha_k)}{\Gamma(\sum_{k=1}^K \alpha_k)}$$

Latent Dirichlet allocation

1. Let $\beta \triangleq (\beta_k)_{k=1}^K$, $\boldsymbol{\theta} \triangleq (\boldsymbol{\theta}_d)_{d=1}^D$, $\mathbf{C} \triangleq (c_{dl})_{d,l=1}^{D,n_d}$, $\mathbf{X} \triangleq (x_{dl})_{d,l=1}^{D,n_d}$, the posterior distribution

$$p(\beta, \boldsymbol{\theta}, \mathbf{C} \,|\, \mathbf{X}) \propto \left[\prod_{k=1}^{K} p(\beta_k|\gamma)\right] \left[\prod_{d=1}^{D} p(\boldsymbol{\theta}_d \,|\alpha) \prod_{l=1}^{n_d} p(c_{dl}|\theta_d) p(x_{dl}|\beta, c_{dl})\right]$$

2. From previous lectures:

$$p(c_{dl}|\, \boldsymbol{\theta}_d) = \prod_{k=1}^{K} (\boldsymbol{\theta}_{dk})^{1(c_{dl}=k)}$$

$$p(x_{dl}|\, \boldsymbol{\theta}, c_{dl}) = \prod_{k=1}^{K} \prod_{v=1}^{V} \beta_{kv}^{1(x_{dl}=v)1(c_{dl}=k)}$$

3. Together with the fact:

$$\int_{\boldsymbol{\theta} \in \triangle_{K-1}} \prod_{k=1}^{K} \boldsymbol{\theta}_k^{\alpha_k - 1} \,\mathrm{d}\,\boldsymbol{\theta}_k = \frac{\prod_{k=1}^{K} \Gamma(\alpha_k)}{\Gamma(\sum_{k=1}^{K} \alpha_k)}$$

Latent Dirichlet allocation

1. Integrate out the local parameters: topic distributions $\boldsymbol{\theta}$ for each document, it results in the following semi-collapsed distribution: $p(\mathbf{X}, \mathbf{C}, \beta | \alpha, \gamma) =$

$$\prod_{d=1}^{D} \frac{\Gamma(K\alpha)}{\Gamma(K\alpha + n_{d\cdot\cdot})} \prod_{k=1}^{K} \frac{\Gamma(\alpha + n_{dk\cdot})}{\Gamma(\alpha)} \prod_{k=1}^{K} \frac{\Gamma(V\gamma)}{\Gamma(\gamma)^V} \prod_{v=1}^{V} \beta_{kv}^{\gamma + n_{\cdot kv} - 1} ,$$

where $n_{dkw} \triangleq \sum_{l=1}^{n_d} 1(c_{dl} = k) 1(x_{dl} = w)$ is #word w in doc d with topic k; $\cdot$ means marginal sum, *e.g.* $n_{\cdot kw} \triangleq \sum_{d=1}^{D} n_{dkw}$.

2. SG-MCMC requires parameter spaces unconstrained:
   - reparameterization: $\beta_{kv} = \lambda_{kv} / \sum_{v'} \lambda_{kv'}$, with the following prior:

$$\lambda_{kv} \sim \text{Ga}(\lambda_{kv}; \gamma, 1)$$

$$\prod_{k=1}^{K} \frac{\Gamma(V\gamma)}{\Gamma(\gamma)^V} \prod_{v=1}^{V} \beta_{kv}^{\gamma + n_{\cdot kv} - 1} \Longrightarrow \prod_{k=1}^{K} \prod_{v=1}^{V} \text{Ga}(\lambda_{kv}; \gamma, 1) \prod_{v=1}^{V} (\lambda_{kv} / \sum_{v'} \lambda_{kv'})^{n_{\cdot kw}}$$

Latent Dirichlet allocation

1. Integrate out the local parameters: topic distributions $\boldsymbol{\theta}$ for each document, it results in the following semi-collapsed distribution: $p(\mathbf{X}, \mathbf{C}, \beta|\alpha, \gamma) =$

$$\prod_{d=1}^{D} \frac{\Gamma(K\alpha)}{\Gamma(K\alpha + n_{d\cdot\cdot})} \prod_{k=1}^{K} \frac{\Gamma(\alpha + n_{dk\cdot})}{\Gamma(\alpha)} \prod_{k=1}^{K} \frac{\Gamma(V\gamma)}{\Gamma(\gamma)^V} \prod_{v=1}^{V} \beta_{kv}^{\gamma + n_{\cdot kv} - 1} ,$$

where $n_{dkw} \triangleq \sum_{l=1}^{n_d} 1(c_{dl} = k) 1(x_{dl} = w)$ is #word w in doc d with topic k; $\cdot$ means marginal sum, *e.g.* $n_{\cdot kw} \triangleq \sum_{d=1}^{D} n_{dkw}$.

2. SG-MCMC requires parameter spaces unconstrained:
   - reparameterization: $\beta_{kv} = \lambda_{kv} / \sum_{v'} \lambda_{kv'}$, with the following prior:

$$\lambda_{kv} \sim \text{Ga}(\lambda_{kv}; \gamma, 1)$$

$$\prod_{k=1}^{K} \frac{\Gamma(V\gamma)}{\Gamma(\gamma)^V} \prod_{v=1}^{V} \beta_{kv}^{\gamma + n_{\cdot kv} - 1} \Longrightarrow \prod_{k=1}^{K} \prod_{v=1}^{V} \text{Ga}(\lambda_{kv}; \gamma, 1) \prod_{v=1}^{V} (\lambda_{kv} / \sum_{v'} \lambda_{kv'})^{n_{\cdot kw}}$$

Latent Dirichlet allocation

1. Still need to integrate out the local parameter **C**:

$$p(\mathbf{X}, \lambda|\alpha, \gamma) = \mathbb{E}_C\left[p(\mathbf{X}, \mathbf{C}, \beta|\alpha, \gamma)\right] = \mathbb{E}_C\left[\prod_{d=1}^{D} \frac{\Gamma(K\alpha)}{\Gamma(K\alpha + n_{d\cdot\cdot})} \prod_{k=1}^{K} \frac{\Gamma(\alpha + n_{dk\cdot})}{\Gamma(\alpha)} \prod_{v=1}^{V} \mathrm{Ga}(\lambda_{kv}; \gamma, 1) \left(\frac{\lambda_{kv}}{\sum_{v'} \lambda_{kv'}}\right)^{n_{\cdot kw}}\right]$$

2. The stochastic gradient with a minibatch documents $\bar{D}$ of size $|\bar{D}| \ll D$ is:

$$\frac{\partial \log \tilde{p}(\lambda|\alpha, \gamma, \mathbf{X})}{\partial \lambda_{kw}} = \frac{\gamma - 1}{\lambda_{kw}} - 1 + \frac{D}{|\bar{D}|} \sum_{d \in \bar{D}} \mathbb{E}_{c_d|\mathbf{x}_d, \lambda, \alpha}\left[\frac{n_{dkw}}{\lambda_{kw}} - \frac{n_{dk\cdot}}{\lambda_{k\cdot}}\right]$$

3. SGLD update:

$$\lambda_{kw}^{t+1} = \lambda_{kw}^{t} + \frac{\partial \log \tilde{p}(\lambda|\alpha, \gamma, \mathbf{X})}{\partial \lambda_{kw}} h_{t+1} + \sqrt{2h_{t+1}} N(0, \mathbf{I})$$

Latent Dirichlet allocation

1. Still need to integrate out the local parameter **C**:

$$p(\mathbf{X}, \lambda|\alpha, \gamma) = \mathbb{E}_C \left[p(\mathbf{X}, \mathbf{C}, \beta|\alpha, \gamma)\right] = \mathbb{E}_C \left[\prod_{d=1}^{D} \frac{\Gamma(K\alpha)}{\Gamma(K\alpha + n_{d\cdot\cdot})} \prod_{k=1}^{K} \frac{\Gamma(\alpha + n_{dk\cdot})}{\Gamma(\alpha)} \prod_{v=1}^{V} \mathrm{Ga}(\lambda_{kv}; \gamma, 1) \left(\frac{\lambda_{kv}}{\sum_{v'} \lambda_{kv'}}\right)^{n_{\cdot kw}}\right]$$

2. The stochastic gradient with a minibatch documents $\bar{D}$ of size $|\bar{D}| \ll D$ is:

$$\frac{\partial \log \tilde{p}(\lambda|\alpha, \gamma, \mathbf{X})}{\partial \lambda_{kw}} = \frac{\gamma - 1}{\lambda_{kw}} - 1 + \frac{D}{|\bar{D}|} \sum_{d \in \bar{D}} \mathbb{E}_{c_d|\,\mathbf{x}_d, \lambda, \alpha} \left[\frac{n_{dkw}}{\lambda_{kw}} - \frac{n_{dk\cdot}}{\lambda_{k\cdot}}\right]$$

3. SGLD update:

$$\lambda_{kw}^{t+1} = \lambda_{kw}^{t} + \frac{\partial \log \tilde{p}(\lambda|\alpha, \gamma, \mathbf{X})}{\partial \lambda_{kw}} h_{t+1} + \sqrt{2h_{t+1}}\, N(0, \mathbf{I})$$

Latent Dirichlet allocation

1. Still need to integrate out the local parameter $\mathbf{C}$:

$$p(\mathbf{X}, \lambda|\alpha, \gamma) = \mathbb{E}_C\left[p(\mathbf{X}, \mathbf{C}, \beta|\alpha, \gamma)\right] = \mathbb{E}_C\left[\prod_{d=1}^{D} \frac{\Gamma(K\alpha)}{\Gamma(K\alpha + n_{d\cdot\cdot})}\right.$$

$$\left.\prod_{k=1}^{K} \frac{\Gamma(\alpha + n_{dk\cdot})}{\Gamma(\alpha)} \prod_{v=1}^{V} \mathrm{Ga}(\lambda_{kv}; \gamma, 1) \left(\frac{\lambda_{kv}}{\sum_{v'} \lambda_{kv'}}\right)^{n_{\cdot kw}}\right]$$

2. The stochastic gradient with a minibatch documents $\bar{D}$ of size $|\bar{D}| \ll D$ is:

$$\frac{\partial \log \tilde{p}(\lambda|\alpha, \gamma, \mathbf{X})}{\partial \lambda_{kw}} = \frac{\gamma - 1}{\lambda_{kw}} - 1 + \frac{D}{|\bar{D}|} \sum_{d \in \bar{D}} \mathbb{E}_{c_d|\mathbf{x}_d, \lambda, \alpha}\left[\frac{n_{dkw}}{\lambda_{kw}} - \frac{n_{dk\cdot}}{\lambda_{k\cdot}}\right]$$

3. SGLD update:

$$\lambda_{kw}^{t+1} = \lambda_{kw}^{t} + \frac{\partial \log \tilde{p}(\lambda|\alpha, \gamma, \mathbf{X})}{\partial \lambda_{kw}} h_{t+1} + \sqrt{2h_{t+1}} N(0, \mathbf{I})$$

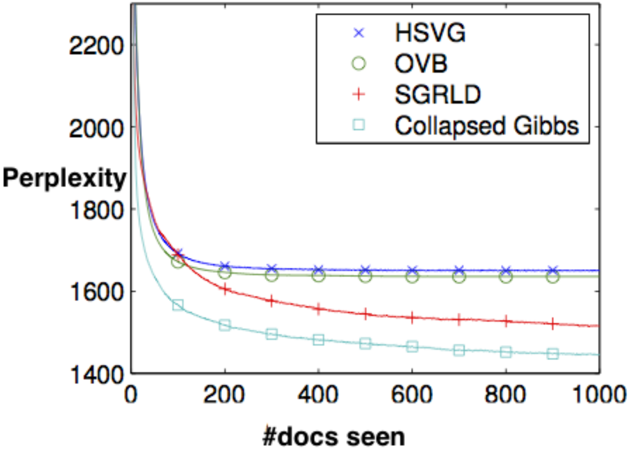
Latent Dirichlet allocation

1. LDA with the above SGLD update would not work well in practice because of the high dimensionality of model parameters.
2. To make it work, Riemannian geometry information (2nd-order information) need to bring in SGLD:
   - leading to Stochastic Gradient Riemannian Langevin Dynamics (SGRLD) for LDA[11]
   - it considers parameter geometry so that step sizes for each dimension of the parameter are adaptive

[11] S. Patterson and Y. W. Teh. "Stochastic Gradient Riemannian Langevin Dynamics on the Probability Simplex". In: *NIPS*. 2013.

Experiments: SGRLD for LDA[12]

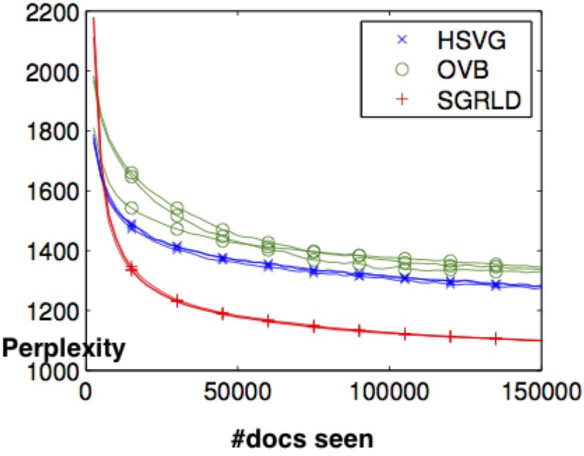
1. NIPS dataset:
   - the collection of NIPS papers from 1988-2003, with 2483 documents, 50 topics



[12] S. Patterson and Y. W. Teh. "Stochastic Gradient Riemannian Langevin Dynamics on the Probability Simplex". In: *NIPS*.

Experiments: SGRLD for LDA[13]

1. Wikipedia dataset:
   - a set of articles downloaded at random from Wikipedia, with 150,000 documents



[13] S. Patterson and Y. W. Teh. "Stochastic Gradient Riemannian Langevin Dynamics on the Probability Simplex". In: *NIPS*. 2013

Conclusion

1. I have introduced:
    - basic concepts in MCMC
    - basic ideas in SG-MCMC, two SG-MCMC algorithms, and application in LDA
2. Topics not covered:
    - a general review of SG-MCMC algorithms
    - theory related to stochastic differential equations and Itó diffusions
    - convergence theory
    - various applications in deep learning, including SG-MCMC for learning weight uncertainty and SG-MCMC for deep generative models
    - interested readers should refer to related references

Thank You