Markov Chain Monte Carlo: design and optimisation

Krys Latuszynski
(University of Warwick, UK)

September 2018

Prior-posterior: the usual MCMC setting

- let π be a target probability distribution on Θ, typically arising as a posterior distribution in Bayesian inference,
- in Bayesian Statistics we typically consider a model $\mathcal{M}(\theta)$, parametrised by $\theta \in \Theta$
- there is a prior distribution $p(\theta)$ on Θ. It can be chosen to:
- 1. convey prior information or belief about values of θ
  2. be uninformative
  3. facilitate computations and consequently enable inference
- there is data y that has likellihood $l(\theta) = l(\theta|y)$ under model $\mathcal{M}(\theta)$
- the posterior distribution $\pi(\theta)$ arises from the Bayes formula as

$$\pi(\theta) = \pi(\theta|y) := \frac{p(\theta)l(\theta|y)}{\int_\Theta p(\theta)l(\theta|y)d\theta} \propto p(\theta)l(\theta)$$

Prior-posterior: the usual MCMC setting

- let π be a target probability distribution on Θ, typically arising as a posterior distribution in Bayesian inference,
- in Bayesian Statistics we typically consider a model $\mathcal{M}(\theta)$, parametrised by $\theta \in \Theta$
- there is a prior distribution $p(\theta)$ on Θ. It can be chosen to:
- 1. convey prior information or belief about values of θ
  2. be uninformative
  3. facilitate computations and consequently enable inference
- there is data y that has likellihood $l(\theta) = l(\theta|y)$ under model $\mathcal{M}(\theta)$
- the posterior distribution $\pi(\theta)$ arises from the Bayes formula as

$$\pi(\theta) = \pi(\theta|y) := \frac{p(\theta)l(\theta|y)}{\int_{\Theta} p(\theta)l(\theta|y)d\theta} \propto p(\theta)l(\theta)$$

Prior-posterior: the usual MCMC setting

- let π be a target probability distribution on Θ, typically arising as a posterior distribution in Bayesian inference,
- in Bayesian Statistics we typically consider a model $\mathcal{M}(\theta)$, parametrised by $\theta \in \Theta$
- there is a prior distribution $p(\theta)$ on Θ. It can be chosen to:
- 1. convey prior information or belief about values of θ
  2. be uninformative
  3. facilitate computations and consequently enable inference
- there is data y that has likellihood $l(\theta) = l(\theta|y)$ under model $\mathcal{M}(\theta)$
- the posterior distribution $\pi(\theta)$ arises from the Bayes formula as

$$\pi(\theta) = \pi(\theta|y) := \frac{p(\theta)l(\theta|y)}{\int_\Theta p(\theta)l(\theta|y)d\theta} \propto p(\theta)l(\theta)$$

Prior-posterior: the usual MCMC setting

- let π be a target probability distribution on Θ, typically arising as a posterior distribution in Bayesian inference,
- in Bayesian Statistics we typically consider a model $\mathcal{M}(\theta)$, parametrised by $\theta \in \Theta$
- there is a prior distribution $p(\theta)$ on Θ. It can be chosen to:
- 1. convey prior information or belief about values of θ
  2. be uninformative
  3. facilitate computations and consequently enable inference
- there is data y that has likellihood $l(\theta) = l(\theta|y)$ under model $\mathcal{M}(\theta)$
- the posterior distribution $\pi(\theta)$ arises from the Bayes formula as

$$\pi(\theta) = \pi(\theta|y) := \frac{p(\theta)l(\theta|y)}{\int_\Theta p(\theta)l(\theta|y)d\theta} \propto p(\theta)l(\theta)$$

Prior-posterior: the usual MCMC setting

- let π be a target probability distribution on Θ, typically arising as a posterior distribution in Bayesian inference,
- in Bayesian Statistics we typically consider a model $\mathcal{M}(\theta)$, parametrised by $\theta \in \Theta$
- there is a prior distribution $p(\theta)$ on Θ. It can be chosen to:
- 1. convey prior information or belief about values of θ
  2. be uninformative
  3. facilitate computations and consequently enable inference
- there is data y that has likellihood $l(\theta) = l(\theta|y)$ under model $\mathcal{M}(\theta)$
- the posterior distribution $\pi(\theta)$ arises from the Bayes formula as

$$\pi(\theta) = \pi(\theta|y) := \frac{p(\theta)l(\theta|y)}{\int_\Theta p(\theta)l(\theta|y)d\theta} \propto p(\theta)l(\theta)$$

Prior-posterior: the usual MCMC setting

- let π be a target probability distribution on Θ, typically arising as a posterior distribution in Bayesian inference,
- in Bayesian Statistics we typically consider a model $\mathcal{M}(\theta)$, parametrised by $\theta \in \Theta$
- there is a prior distribution $p(\theta)$ on Θ. It can be chosen to:
- 1. convey prior information or belief about values of θ
  2. be uninformative
  3. facilitate computations and consequently enable inference
- there is data y that has likellihood $l(\theta) = l(\theta|y)$ under model $\mathcal{M}(\theta)$
- the posterior distribution $\pi(\theta)$ arises from the Bayes formula as

$$\pi(\theta) \;=\; \pi(\theta|y) \;:=\; \frac{p(\theta)l(\theta|y)}{\int_\Theta p(\theta)l(\theta|y)d\theta} \propto \; p(\theta)l(\theta)$$

Example: a diffusion model

- Consider a diffusion model $\mathcal{M}(\theta)$ where $\theta = (\mu, \sigma)$:

$$dX_t = \mu dt + \sigma dB_t$$

  observed at discrete time points $(t_0, t_1, \dots, t_N)$ as $(x_{t_0}, x_{t_1}, \dots, x_{t_N})$
- The likelihood function is

$$l(\theta|x_{t_0}, x_{t_1}, \dots, x_{t_N}) = \prod_{i=1}^{N} l(\theta|x_{t_i}, x_{t_{i-1}}) = \prod_{i=1}^{N} \phi_{N(\mu(t_i - t_{i-1}), \sigma^2(t_i - t_{i-1}))}(x_{t_i} - x_{t_{i-1}}).$$

- This posterior $\pi(\theta)$ summarises uncertainty about the parameter $\theta \in \Theta$ and is used for all inferential questions like credible sets, decision making, prediction, model choice, etc.
- In the diffusion example predicting the value of the diffusion at time $t > t_N$ would amount to repeating the following steps:
  1. sample from the posterior $\theta = (\mu, \sigma) \sim \pi(\theta)$
  2. sample $X_t \sim N(x_{t_N} + \mu(t - t_N), \sigma^2(t - t_N))$

Example: a diffusion model

- Consider a diffusion model $\mathcal{M}(\theta)$ where $\theta = (\mu, \sigma)$:

$$dX_t = \mu dt + \sigma dB_t$$

  observed at discrete time points $(t_0, t_1, \dots, t_N)$ as $(x_{t_0}, x_{t_1}, \dots, x_{t_N})$
- The likelihood function is

$$l(\theta|x_{t_0}, x_{t_1}, \dots, x_{t_N}) = \prod_{i=1}^{N} l(\theta|x_{t_i}, x_{t_{i-1}}) = \prod_{i=1}^{N} \phi_{N(\mu(t_i - t_{i-1}), \sigma^2(t_i - t_{i-1}))}(x_{t_i} - x_{t_{i-1}}).$$

- This posterior $\pi(\theta)$ summarises uncertainty about the parameter $\theta \in \Theta$ and is used for all inferential questions like credible sets, decision making, prediction, model choice, etc.
- In the diffusion example predicting the value of the diffusion at time $t > t_N$ would amount to repeating the following steps:
  1. sample from the posterior $\theta = (\mu, \sigma) \sim \pi(\theta)$
  2. sample $X_t \sim N(x_{t_N} + \mu(t - t_N), \sigma^2(t - t_N))$

Example: a diffusion model

- Consider a diffusion model $\mathcal{M}(\theta)$ where $\theta = (\mu, \sigma)$:

$$dX_t = \mu dt + \sigma dB_t$$

  observed at discrete time points $(t_0, t_1, \ldots, t_N)$ as $(x_{t_0}, x_{t_1}, \ldots, x_{t_N})$
- The likelihood function is

$$l(\theta|x_{t_0}, x_{t_1}, \ldots, x_{t_N}) = \prod_{i=1}^{N} l(\theta|x_{t_i}, x_{t_{i-1}}) = \prod_{i=1}^{N} \phi_{N(\mu(t_i - t_{i-1}), \sigma^2(t_i - t_{i-1}))}(x_{t_i} - x_{t_{i-1}}).$$

- This posterior $\pi(\theta)$ summarises uncertainty about the parameter $\theta \in \Theta$ and is used for all inferential questions like credible sets, decision making, prediction, model choice, etc.
- In the diffusion example predicting the value of the diffusion at time $t > t_N$ would amount to repeating the following steps:
  1. sample from the posterior $\theta = (\mu, \sigma) \sim \pi(\theta)$
  2. sample $X_t \sim N(x_{t_N} + \mu(t - t_N), \sigma^2(t - t_N))$

Example: a diffusion model

- Consider a diffusion model $\mathcal{M}(\theta)$ where $\theta = (\mu, \sigma)$:
$$dX_t = \mu dt + \sigma dB_t$$
observed at discrete time points $(t_0, t_1, \ldots, t_N)$ as $(x_{t_0}, x_{t_1}, \ldots, x_{t_N})$
- The likelihood function is
$$l(\theta|x_{t_0}, x_{t_1}, \ldots, x_{t_N}) = \prod_{i=1}^{N} l(\theta|x_{t_i}, x_{t_{i-1}}) = \prod_{i=1}^{N} \phi_{N(\mu(t_i - t_{i-1}), \sigma^2(t_i - t_{i-1}))}(x_{t_i} - x_{t_{i-1}}).$$
- This posterior $\pi(\theta)$ summarises uncertainty about the parameter $\theta \in \Theta$ and is used for all inferential questions like credible sets, decision making, prediction, model choice, etc.
- In the diffusion example predicting the value of the diffusion at time $t > t_N$ would amount to repeating the following steps:
  1. sample from the posterior $\theta = (\mu, \sigma) \sim \pi(\theta)$
  2. sample $X_t \sim N(x_{t_N} + \mu(t - t_N), \sigma^2(t - t_N))$

the MAP estimator

- One of the classical estimation tasks is to compute the Maximum a Posteriori Estimator (MAP), say θ_{MAP}.
- $$\theta_{MAP} := \text{argmax}_\theta \pi(\theta) = \text{argmax}_\theta \left\{ p(\theta) l(\theta | y_1, \dots, y_n) \right\}$$
- Computing θ_{MAP} may be nontrivial, especially if $\pi(\theta)$ is multimodal.
- There are specialised algorithms for doing this.
- Some non-bayesian statistical inference approaches can be rewritten as bayesian MAP estimators (for example the LASSO).

the MAP estimator

- One of the classical estimation tasks is to compute the Maximum a Posteriori Estimator (MAP), say θ_{MAP}.
- $$\theta_{MAP} := \text{argmax}_\theta \pi(\theta) = \text{argmax}_\theta \Big\{ p(\theta) l(\theta | y_1, \ldots, y_n) \Big\}$$
- Computing θ_{MAP} may be nontrivial, especially if $\pi(\theta)$ is multimodal.
- There are specialised algorithms for doing this.
- Some non-bayesian statistical inference approaches can be rewritten as bayesian MAP estimators (for example the LASSO).

the MAP estimator

- One of the classical estimation tasks is to compute the Maximum a Posteriori Estimator (MAP), say θ_{MAP}.
- $$\theta_{MAP} := \mathrm{argmax}_\theta \pi(\theta) = \mathrm{argmax}_\theta \Big\{ p(\theta) l(\theta | y_1, \ldots, y_n) \Big\}$$
- Computing θ_{MAP} may be nontrivial, especially if $\pi(\theta)$ is multimodal.
- There are specialised algorithms for doing this.
- Some non-bayesian statistical inference approaches can be rewritten as bayesian MAP estimators (for example the LASSO).

the MAP estimator

- One of the classical estimation tasks is to compute the Maximum a Posteriori Estimator (MAP), say θ_{MAP}.
- $$\theta_{MAP} := \mathrm{argmax}_\theta \pi(\theta) = \mathrm{argmax}_\theta \Big\{ p(\theta) l(\theta | y_1, \ldots, y_n) \Big\}$$
- Computing θ_{MAP} may be nontrivial, especially if $\pi(\theta)$ is multimodal.
- There are specialised algorithms for doing this.
- Some non-bayesian statistical inference approaches can be rewritten as bayesian MAP estimators (for example the LASSO).

the MAP estimator

- One of the classical estimation tasks is to compute the Maximum a Posteriori Estimator (MAP), say θ_{MAP}.
- $$\theta_{MAP} := \mathrm{argmax}_\theta \pi(\theta) = \mathrm{argmax}_\theta \Big\{ p(\theta) l(\theta | y_1, \ldots, y_n) \Big\}$$
- Computing θ_{MAP} may be nontrivial, especially if $\pi(\theta)$ is multimodal.
- There are specialised algorithms for doing this.
- Some non-bayesian statistical inference approaches can be rewritten as bayesian MAP estimators (for example the LASSO).

the Bayesian estimator

- Bayesian estimator is an estimator that minimizes the posterior expected value of a loss function.
- The loss function
$$L(\cdot,\cdot) : \Theta \times \Theta \to \mathbb{R}$$
- After seeing data $(y_1, \ldots, y_n)$ we choose an estimator $\hat{\theta}(y_1, \ldots, y_n)$
- Its expected loss is
$$\begin{aligned}\mathbb{E}L(\theta, \hat{\theta}(y_1, \ldots, y_n)) &= \int_{\mathbb{Y}^n \times \Theta} L(\theta, \hat{\theta}(y_1, \ldots, y_n)) m(y_1, \ldots, y_n|\theta) p(\theta) \\ &= \int_{\mathbb{Y}^n \times \Theta} L(\theta, \hat{\theta}(y_1, \ldots, y_n)) \pi(\theta) p(dy)\end{aligned}$$
- $\hat{\theta}(y_1, \ldots, y_n)$ is a Bayesian estimator if it minimizes the above expected loss.

the Bayesian estimator

- Bayesian estimator is an estimator that minimizes the posterior expected value of a loss function.
- The loss function
$$L(\cdot,\cdot) : \Theta \times \Theta \to \mathbb{R}$$
- After seeing data $(y_1, \ldots, y_n)$ we choose an estimator $\hat{\theta}(y_1, \ldots, y_n)$
- Its expected loss is
$$\begin{aligned}\mathbb{E}L(\theta, \hat{\theta}(y_1, \ldots, y_n)) &= \int_{\mathbb{Y}^n \times \Theta} L(\theta, \hat{\theta}(y_1, \ldots, y_n)) m(y_1, \ldots, y_n | \theta) p(\theta) \\ &= \int_{\mathbb{Y}^n \times \Theta} L(\theta, \hat{\theta}(y_1, \ldots, y_n)) \pi(\theta) p(dy)\end{aligned}$$
- $\hat{\theta}(y_1, \ldots, y_n)$ is a Bayesian estimator if it minimizes the above expected loss.

the Bayesian estimator

- Bayesian estimator is an estimator that minimizes the posterior expected value of a loss function.
- The loss function

$$L(\cdot,\cdot) : \Theta \times \Theta \to \mathbb{R}$$

- After seeing data $(y_1, \ldots, y_n)$ we choose an estimator $\hat{\theta}(y_1, \ldots, y_n)$
- Its expected loss is

$$\begin{aligned}\mathbb{E}L(\theta, \hat{\theta}(y_1, \ldots, y_n)) &= \int_{\mathbb{Y}^n \times \Theta} L(\theta, \hat{\theta}(y_1, \ldots, y_n)) m(y_1, \ldots, y_n | \theta) p(\theta) \\ &= \int_{\mathbb{Y}^n \times \Theta} L(\theta, \hat{\theta}(y_1, \ldots, y_n)) \pi(\theta) p(dy)\end{aligned}$$

- $\hat{\theta}(y_1, \ldots, y_n)$ is a Bayesian estimator if it minimizes the above expected loss.

the Bayesian estimator

- Bayesian estimator is an estimator that minimizes the posterior expected value of a loss function.
- The loss function

$$L(\cdot,\cdot) : \Theta \times \Theta \to \mathbb{R}$$

- After seeing data $(y_1, \ldots, y_n)$ we choose an estimator $\hat{\theta}(y_1, \ldots, y_n)$
- Its expected loss is

$$\begin{aligned}
\mathbb{E}L(\theta, \hat{\theta}(y_1, \ldots, y_n)) &= \int_{\mathbb{Y}^n \times \Theta} L(\theta, \hat{\theta}(y_1, \ldots, y_n)) m(y_1, \ldots, y_n | \theta) p(\theta) \\
&= \int_{\mathbb{Y}^n \times \Theta} L(\theta, \hat{\theta}(y_1, \ldots, y_n)) \pi(\theta) p(dy)
\end{aligned}$$

- $\hat{\theta}(y_1, \ldots, y_n)$ is a Bayesian estimator if it minimizes the above expected loss.

the Bayesian estimator

- Bayesian estimator is an estimator that minimizes the posterior expected value of a loss function.
- The loss function

$$L(\cdot,\cdot):\Theta\times\Theta\to\mathbb{R}$$

- After seeing data $(y_1,\ldots,y_n)$ we choose an estimator $\hat{\theta}(y_1,\ldots,y_n)$
- Its expected loss is

$$\begin{aligned}\mathbb{E}L(\theta,\hat{\theta}(y_1,\ldots,y_n)) &= \int_{\mathbb{Y}^n\times\Theta} L(\theta,\hat{\theta}(y_1,\ldots,y_n))m(y_1,\ldots,y_n|\theta)p(\theta)\\ &= \int_{\mathbb{Y}^n\times\Theta} L(\theta,\hat{\theta}(y_1,\ldots,y_n))\pi(\theta)p(dy)\end{aligned}$$

- $\hat{\theta}(y_1,\ldots,y_n)$ is a Bayesian estimator if it minimizes the above expected loss.

the Bayesian estimator and computing integrals

- The most common choice is the quadratic loss function

$$L(\theta_1, \theta_2) = (\theta_1 - \theta_2)^2$$

- in which case

$$\hat{\theta}(y_1, \dots, y_n) = \mathbb{E}_\pi \theta$$

  so it is the posterior mean.
- So computing the Bayesian estimator is computing the integral wrt the posterior

$$\int_\Theta \theta \pi(\theta)$$

- Similarly answering other inferential questions like credible sets, posterior variance etc involve computing integrals of the form

$$\int_\Theta f(\theta)\pi(\theta).$$

the Bayesian estimator and computing integrals

- The most common choice is the quadratic loss function
$$L(\theta_1, \theta_2) = (\theta_1 - \theta_2)^2$$
- in which case
$$\hat{\theta}(y_1, \ldots, y_n) = \mathbb{E}_\pi \theta$$
so it is the posterior mean.
- So computing the Bayesian estimator is computing the integral wrt the posterior
$$\int_\Theta \theta \pi(\theta)$$
- Similarly answering other inferential questions like credible sets, posterior variance etc involve computing integrals of the form
$$\int_\Theta f(\theta)\pi(\theta).$$

the Bayesian estimator and computing integrals

- The most common choice is the quadratic loss function
$$L(\theta_1, \theta_2) = (\theta_1 - \theta_2)^2$$
- in which case
$$\hat{\theta}(y_1, \ldots, y_n) = \mathbb{E}_\pi \theta$$
so it is the posterior mean.
- So computing the Bayesian estimator is computing the integral wrt the posterior
$$\int_\Theta \theta \pi(\theta)$$
- Similarly answering other inferential questions like credible sets, posterior variance etc involve computing integrals of the form
$$\int_\Theta f(\theta)\pi(\theta).$$

the Bayesian estimator and computing integrals

- The most common choice is the quadratic loss function

$$L(\theta_1, \theta_2) = (\theta_1 - \theta_2)^2$$

- in which case

$$\hat{\theta}(y_1, \ldots, y_n) = \mathbb{E}_\pi \theta$$

  so it is the posterior mean.
- So computing the Bayesian estimator is computing the integral wrt the posterior

$$\int_\Theta \theta \pi(\theta)$$

- Similarly answering other inferential questions like credible sets, posterior variance etc involve computing integrals of the form

$$\int_\Theta f(\theta) \pi(\theta).$$

The MCMC estimator

- let π be a target probability distribution on $\mathcal{X}$, typically arising as a posterior distribution in Bayesian inference,
- the goal is to evaluate
$$I := \int_{\mathcal{X}} f(x)\pi(dx).$$
- direct sampling from π is not possible or inefficient
for example π is known up to a normalising constant
- MCMC approach is to simulate $(X_n)_{n\geq 0}$, an ergodic Markov chain with **transition kernel** P and limiting distribution π, and take ergodic averages as an estimate of I.
- the usual estimate
$$\hat{I} := \frac{1}{n}\sum_{k=t}^{t+n} f(X_k)$$

The MCMC estimator

- let π be a target probability distribution on $\mathcal{X}$, typically arising as a posterior distribution in Bayesian inference,
- the goal is to evaluate

$$I := \int_{\mathcal{X}} f(x)\pi(dx).$$

- direct sampling from π is not possible or inefficient for example π is known up to a normalising constant
- MCMC approach is to simulate $(X_n)_{n\geq 0}$, an ergodic Markov chain with **transition kernel** P and limiting distribution π, and take ergodic averages as an estimate of I.
- the usual estimate

$$\hat{I} := \frac{1}{n}\sum_{k=t}^{t+n} f(X_k)$$

The MCMC estimator

- let π be a target probability distribution on $\mathcal{X}$, typically arising as a posterior distribution in Bayesian inference,
- the goal is to evaluate
$$I := \int_{\mathcal{X}} f(x)\pi(dx).$$
- direct sampling from π is not possible or inefficient
for example π is known up to a normalising constant
- MCMC approach is to simulate $(X_n)_{n\geq 0}$, an ergodic Markov chain with **transition kernel** P and limiting distribution π, and take ergodic averages as an estimate of I.
- the usual estimate
$$\hat{I} := \frac{1}{n}\sum_{k=t}^{t+n} f(X_k)$$

The MCMC estimator

- let π be a target probability distribution on $\mathcal{X}$, typically arising as a posterior distribution in Bayesian inference,
- the goal is to evaluate

$$I := \int_{\mathcal{X}} f(x)\pi(dx).$$

- direct sampling from π is not possible or inefficient
for example π is known up to a normalising constant
- MCMC approach is to simulate $(X_n)_{n\geq 0}$, an ergodic Markov chain with **transition kernel** P and limiting distribution π, and take ergodic averages as an estimate of I.
- the usual estimate

$$\hat{I} := \frac{1}{n}\sum_{k=t}^{t+n} f(X_k)$$

Asymptotic validity

- $$\hat{I} := \frac{1}{n}\sum_{k=t}^{t+n} f(X_k) \qquad I := \int_{\mathcal{X}} f(x)\pi(dx).$$
- **SLLN** for Markov chains holds under very mild conditions
- **CLT** for Markov chains holds under some additional assumptions:
  - a mixing condition on P
  - an integrability condition for f

  and is verifiable in many situations of interest
- **CLT**
  $$n^{1/2}(\hat{I} - I) \to N\big(0, \sigma_{\text{as}}(f, P)\big)$$
- Usually $\sigma_{\text{as}}(f, P) >> Var_{\pi}(f)$

Asymptotic validity

- $$\hat{I} := \frac{1}{n}\sum_{k=t}^{t+n} f(X_k) \qquad I := \int_{\mathcal{X}} f(x)\pi(dx).$$
- **SLLN** for Markov chains holds under very mild conditions
- **CLT** for Markov chains holds under some additional assumptions:
  - a mixing condition on P
  - an integrability condition for f

  and is verifiable in many situations of interest
- **CLT**
  $$n^{1/2}(\hat{I} - I) \to N\big(0, \sigma_{\mathrm{as}}(f, P)\big)$$
- Usually $\sigma_{\mathrm{as}}(f, P) >> Var_{\pi}(f)$

Asymptotic validity

- $$\hat{I} := \frac{1}{n}\sum_{k=t}^{t+n} f(X_k) \qquad I := \int_{\mathcal{X}} f(x)\pi(dx).$$
- **SLLN** for Markov chains holds under very mild conditions
- **CLT** for Markov chains holds under some additional assumptions:
  - a mixing condition on P
  - an integrability condition for f

  and is verifiable in many situations of interest
- **CLT** $$n^{1/2}(\hat{I} - I) \to N\big(0, \sigma_{\mathrm{as}}(f,P)\big)$$
- Usually $\sigma_{\mathrm{as}}(f,P) >> Var_{\pi}(f)$

Asymptotic validity

- $$\hat{I} := \frac{1}{n}\sum_{k=t}^{t+n} f(X_k) \qquad I := \int_{\mathcal{X}} f(x)\pi(dx).$$
- **SLLN** for Markov chains holds under very mild conditions
- **CLT** for Markov chains holds under some additional assumptions:
  - a mixing condition on P
  - an integrability condition for f

  and is verifiable in many situations of interest
- **CLT**
$$n^{1/2}(\hat{I} - I) \to N\big(0, \sigma_{\text{as}}(f, P)\big)$$
- Usually $\sigma_{\text{as}}(f, P) >> Var_{\pi}(f)$

Asymptotic validity

- $$\hat{I} := \frac{1}{n}\sum_{k=t}^{t+n} f(X_k) \qquad I := \int_{\mathcal{X}} f(x)\pi(dx).$$
- **SLLN** for Markov chains holds under very mild conditions
- **CLT** for Markov chains holds under some additional assumptions:
  - a mixing condition on P
  - an integrability condition for f

  and is verifiable in many situations of interest
- **CLT**
$$n^{1/2}(\hat{I} - I) \to N\big(0, \sigma_{\mathrm{as}}(f, P)\big)$$
- Usually $\sigma_{\mathrm{as}}(f, P) >> Var_\pi(f)$

Reversibility and stationarity

- How to design P so that X_n converges in distribution to π ?
- **Definition.** P is reversible with respect to π if

$$\pi(x)P(x,y) = \pi(y)P(y,x)$$

  as measures on $\mathcal{X} \times \mathcal{X}$
- **Lemma.** If P is reversible with respect to π then $\pi P = \pi$, so it is also stationary.

Reversibility and stationarity

- How to design P so that X_n converges in distribution to π ?
- **Definition.** P is reversible with respect to π if

$$\pi(x)P(x,y) = \pi(y)P(y,x)$$

  as measures on $\mathcal{X} \times \mathcal{X}$
- **Lemma.** If P is reversible with respect to π then $\pi P = \pi$, so it is also stationary.

Reversibility and stationarity

- How to design P so that X_n converges in distribution to π ?
- **Definition.** P is reversible with respect to π if

$$\pi(x)P(x,y) = \pi(y)P(y,x)$$

  as measures on $\mathcal{X} \times \mathcal{X}$
- **Lemma.** If P is reversible with respect to π then $\pi P = \pi$, so it is also stationary.

The Metropolis algorithm

- **Idea.** Take any transition kernel Q with transition densities $q(x, y)$ and make it reversible with respect to π
- **Algorithm.** Given X_n
  sample $Y_{n+1} \sim Q(X_n, \cdot)$
- with probability $\alpha(X_n, Y_{n+1})$ set $X_{n+1} = Y_{n+1}$, otherwise set $X_{n+1} = X_n$
- where

$$\alpha(x, y) = \min\{1, \frac{\pi(y)q(y, x)}{\pi(x)q(x, y)}\}.$$

- Under mild assumptions on Q the algorithm is ergodic.
- However it's performance depends heavily on Q
- is is **difficult** to design the proposal Q so that P has **good convergence properties**, especially if $\mathcal{X}$ is high dimensional
- For the **Random Walk Metropolis** $Y_{n+1} \sim Q(X_n, \cdot) = N(X_n, \Sigma)$ often $\Sigma = \sigma I$

The Metropolis algorithm

- **Idea.** Take any transition kernel Q with transition densities $q(x, y)$ and make it reversible with respect to π
- **Algorithm.** Given X_n sample $Y_{n+1} \sim Q(X_n, \cdot)$
- with probability $\alpha(X_n, Y_{n+1})$ set $X_{n+1} = Y_{n+1}$, otherwise set $X_{n+1} = X_n$
- where
$$\alpha(x, y) = \min\{1, \frac{\pi(y)q(y, x)}{\pi(x)q(x, y)}\}.$$
- Under mild assumptions on Q the algorithm is ergodic.
- However it's performance depends heavily on Q
- is is **difficult** to design the proposal Q so that P has **good convergence properties**, especially if $\mathcal{X}$ is high dimensional
- For the **Random Walk Metropolis** $Y_{n+1} \sim Q(X_n, \cdot) = N(X_n, \Sigma)$ often $\Sigma = \sigma I$

The Metropolis algorithm

- **Idea.** Take any transition kernel Q with transition densities $q(x,y)$ and make it reversible with respect to π
- **Algorithm.** Given X_n
  sample $Y_{n+1} \sim Q(X_n, \cdot)$
- with probability $\alpha(X_n, Y_{n+1})$ set $X_{n+1} = Y_{n+1}$, otherwise set $X_{n+1} = X_n$
- where
$$\alpha(x,y) = \min\{1, \frac{\pi(y)q(y,x)}{\pi(x)q(x,y)}\}.$$
- Under mild assumptions on Q the algorithm is ergodic.
- However it's performance depends heavily on Q
- is is **difficult** to design the proposal Q so that P has **good convergence properties**, especially if $\mathcal{X}$ is high dimensional
- For the **Random Walk Metropolis** $Y_{n+1} \sim Q(X_n, \cdot) = N(X_n, \Sigma)$ often $\Sigma = \sigma I$

The Metropolis algorithm

- **Idea.** Take any transition kernel Q with transition densities $q(x, y)$ and make it reversible with respect to π
- **Algorithm.** Given X_n
  sample $Y_{n+1} \sim Q(X_n, \cdot)$
- with probability $\alpha(X_n, Y_{n+1})$ set $X_{n+1} = Y_{n+1}$, otherwise set $X_{n+1} = X_n$
- where

$$\alpha(x, y) = \min\{1, \frac{\pi(y)q(y, x)}{\pi(x)q(x, y)}\}.$$

- Under mild assumptions on Q the algorithm is ergodic.
- However it's performance depends heavily on Q
- is is **difficult** to design the proposal Q so that P has **good convergence properties**, especially if $\mathcal{X}$ is high dimensional
- For the **Random Walk Metropolis** $Y_{n+1} \sim Q(X_n, \cdot) = N(X_n, \Sigma)$ often $\Sigma = \sigma I$

The Metropolis algorithm

- **Idea.** Take any transition kernel Q with transition densities $q(x,y)$ and make it reversible with respect to π
- **Algorithm.** Given X_n
  sample $Y_{n+1} \sim Q(X_n, \cdot)$
- with probability $\alpha(X_n, Y_{n+1})$ set $X_{n+1} = Y_{n+1}$, otherwise set $X_{n+1} = X_n$
- where
$$\alpha(x,y) = \min\{1, \frac{\pi(y)q(y,x)}{\pi(x)q(x,y)}\}.$$
- Under mild assumptions on Q the algorithm is ergodic.
- However it's performance depends heavily on Q
- is is **difficult** to design the proposal Q so that P has **good convergence properties**, especially if $\mathcal{X}$ is high dimensional
- For the **Random Walk Metropolis** $Y_{n+1} \sim Q(X_n, \cdot) = N(X_n, \Sigma)$ often $\Sigma = \sigma I$

The Metropolis algorithm

- **Idea.** Take any transition kernel Q with transition densities $q(x, y)$ and make it reversible with respect to π
- **Algorithm.** Given X_n
  sample $Y_{n+1} \sim Q(X_n, \cdot)$
- with probability $\alpha(X_n, Y_{n+1})$ set $X_{n+1} = Y_{n+1}$, otherwise set $X_{n+1} = X_n$
- where

$$\alpha(x, y) = \min\{1, \frac{\pi(y)q(y, x)}{\pi(x)q(x, y)}\}.$$

- Under mild assumptions on Q the algorithm is ergodic.
- However it's performance depends heavily on Q
- is is **difficult** to design the proposal Q so that P has **good convergence properties**, especially if $\mathcal{X}$ is high dimensional
- For the **Random Walk Metropolis** $Y_{n+1} \sim Q(X_n, \cdot) = N(X_n, \Sigma)$ often $\Sigma = \sigma I$

The Metropolis algorithm

- **Idea.** Take any transition kernel Q with transition densities $q(x,y)$ and make it reversible with respect to π
- **Algorithm.** Given X_n
  sample $Y_{n+1} \sim Q(X_n, \cdot)$
- with probability $\alpha(X_n, Y_{n+1})$ set $X_{n+1} = Y_{n+1}$, otherwise set $X_{n+1} = X_n$
- where
$$\alpha(x,y) = \min\{1, \frac{\pi(y)q(y,x)}{\pi(x)q(x,y)}\}.$$
- Under mild assumptions on Q the algorithm is ergodic.
- However it's performance depends heavily on Q
- is is **difficult** to design the proposal Q so that P has **good convergence properties**, especially if $\mathcal{X}$ is high dimensional
- For the **Random Walk Metropolis** $Y_{n+1} \sim Q(X_n, \cdot) = N(X_n, \Sigma)$ often $\Sigma = \sigma I$

The Gibbs Sampler - exploits conditional distributions

- For $\Theta = \Theta_1 \times \Theta_2 \times \cdots \times \Theta_d$
- denote the marginals of π as
$$\pi(\theta_k|\theta_{-k})$$
where
$$\theta_{-k} = (\theta_1, \dots, \theta_{k-1}, \theta_{k+1}, \dots, \theta_d)$$
- The Gibbs sampler algorithms iterates between updates of
$$\theta_i|\theta_{-i} \sim \pi(\theta_i|\theta_{-i})$$
- There are two basic coordinate selection strategies:
- (1) in each step pick a coordinate at random (Random Scan Gibbs Sampler)
- (2) Update systematically one after another (Systematic Scan Gibbs Sampler)
- If sampling directly from $\sim \pi(\theta_i|\theta_{-i})$ is not possible or not practical, one can design P_i that admits $\sim \pi(\theta_i|\theta_{-i})$ as its invariant distribution. This is called Metropolis within Gibbs

The Gibbs Sampler - exploits conditional distributions

- For $\Theta = \Theta_1 \times \Theta_2 \times \cdots \times \Theta_d$
- denote the marginals of π as
$$\pi(\theta_k|\theta_{-k})$$
where
$$\theta_{-k} = (\theta_1, \ldots, \theta_{k-1}, \theta_{k+1}, \ldots, \theta_d)$$
- The Gibbs sampler algorithms iterates between updates of
$$\theta_i|\theta_{-i} \sim \pi(\theta_i|\theta_{-i})$$
- There are two basic coordinate selection strategies:
- (1) in each step pick a coordinate at random (Random Scan Gibbs Sampler)
- (2) Update systematically one after another (Systematic Scan Gibbs Sampler)
- If sampling directly from $\sim \pi(\theta_i|\theta_{-i})$ is not possible or not practical, one can design P_i that admits $\sim \pi(\theta_i|\theta_{-i})$ as its invariant distribution. This is called Metropolis within Gibbs

The Gibbs Sampler - exploits conditional distributions

- For $\Theta = \Theta_1 \times \Theta_2 \times \cdots \times \Theta_d$
- denote the marginals of π as
$$\pi(\theta_k|\theta_{-k})$$
where
$$\theta_{-k} = (\theta_1, \ldots, \theta_{k-1}, \theta_{k+1}, \ldots, \theta_d)$$
- The Gibbs sampler algorithms iterates between updates of
$$\theta_i|\theta_{-i} \sim \pi(\theta_i|\theta_{-i})$$
- There are two basic coordinate selection strategies:
- (1) in each step pick a coordinate at random (Random Scan Gibbs Sampler)
- (2) Update systematically one after another (Systematic Scan Gibbs Sampler)
- If sampling directly from $\sim \pi(\theta_i|\theta_{-i})$ is not possible or not practical, one can design P_i that admits $\sim \pi(\theta_i|\theta_{-i})$ as its invariant distribution. This is called Metropolis within Gibbs

The Gibbs Sampler - exploits conditional distributions

- For $\Theta = \Theta_1 \times \Theta_2 \times \cdots \times \Theta_d$
- denote the marginals of π as
$$\pi(\theta_k|\theta_{-k})$$
where
$$\theta_{-k} = (\theta_1, \ldots, \theta_{k-1}, \theta_{k+1}, \ldots, \theta_d)$$
- The Gibbs sampler algorithms iterates between updates of
$$\theta_i|\theta_{-i} \sim \pi(\theta_i|\theta_{-i})$$
- There are two basic coordinate selection strategies:
- (1) in each step pick a coordinate at random (Random Scan Gibbs Sampler)
- (2) Update systematically one after another (Systematic Scan Gibbs Sampler)
- If sampling directly from $\sim \pi(\theta_i|\theta_{-i})$ is not possible or not practical, one can design P_i that admits $\sim \pi(\theta_i|\theta_{-i})$ as its invariant distribution. This is called Metropolis within Gibbs

The Gibbs Sampler - exploits conditional distributions

- For $\Theta = \Theta_1 \times \Theta_2 \times \cdots \times \Theta_d$
- denote the marginals of π as
$$\pi(\theta_k|\theta_{-k})$$
where
$$\theta_{-k} = (\theta_1, \ldots, \theta_{k-1}, \theta_{k+1}, \ldots, \theta_d)$$
- The Gibbs sampler algorithms iterates between updates of
$$\theta_i|\theta_{-i} \sim \pi(\theta_i|\theta_{-i})$$
- There are two basic coordinate selection strategies:
- (1) in each step pick a coordinate at random (Random Scan Gibbs Sampler)
- (2) Update systematically one after another (Systematic Scan Gibbs Sampler)
- If sampling directly from $\sim \pi(\theta_i|\theta_{-i})$ is not possible or not practical, one can design P_i that admits $\sim \pi(\theta_i|\theta_{-i})$ as its invariant distribution. This is called Metropolis within Gibbs

The Gibbs Sampler - exploits conditional distributions

- For $\Theta = \Theta_1 \times \Theta_2 \times \cdots \times \Theta_d$
- denote the marginals of π as

$$\pi(\theta_k|\theta_{-k})$$

  where

$$\theta_{-k} = (\theta_1, \ldots, \theta_{k-1}, \theta_{k+1}, \ldots, \theta_d)$$

- The Gibbs sampler algorithms iterates between updates of

$$\theta_i|\theta_{-i} \sim \pi(\theta_i|\theta_{-i})$$

- There are two basic coordinate selection strategies:
- (1) in each step pick a coordinate at random (Random Scan Gibbs Sampler)
- (2) Update systematically one after another (Systematic Scan Gibbs Sampler)
- If sampling directly from $\sim \pi(\theta_i|\theta_{-i})$ is not possible or not practical, one can design P_i that admits $\sim \pi(\theta_i|\theta_{-i})$ as its invariant distribution. This is called Metropolis within Gibbs

The Gibbs Sampler - exploits conditional distributions

- For $\Theta = \Theta_1 \times \Theta_2 \times \cdots \times \Theta_d$
- denote the marginals of π as
$$\pi(\theta_k|\theta_{-k})$$
where
$$\theta_{-k} = (\theta_1, \ldots, \theta_{k-1}, \theta_{k+1}, \ldots, \theta_d)$$
- The Gibbs sampler algorithms iterates between updates of
$$\theta_i|\theta_{-i} \sim \pi(\theta_i|\theta_{-i})$$
- There are two basic coordinate selection strategies:
- (1) in each step pick a coordinate at random (Random Scan Gibbs Sampler)
- (2) Update systematically one after another (Systematic Scan Gibbs Sampler)
- If sampling directly from $\sim \pi(\theta_i|\theta_{-i})$ is not possible or not practical, one can design P_i that admits $\sim \pi(\theta_i|\theta_{-i})$ as its invariant distribution. This is called Metropolis within Gibbs

The MALA Algorithm

- Is based on the $\pi-$limiting Langevin diffusion

$$dX_t = \frac{1}{2}\nabla \log \pi(X_t)dt + dB_t$$

- Euler discretisation of this diffusion suggests the Metropolis-Hastings proposal

$$q(\cdot|X_{(n-1)}) := X_{(n-1)} + \frac{h}{2}\nabla \log \pi(X_{(n-1)}) + h^{1/2}N(0, I_{d\times d})$$

  with the usual accept-reject formula
- MALA works well for "nice" examples, but is unstable for light-tailed π.
- Manifold MALA is based on

$$\begin{aligned} dX_t &= \left(\frac{\sigma(X_t)}{2}\nabla \log \pi(X_t) + \frac{\gamma(X_t)}{2}\right)dt + \sqrt{\sigma}(X_t)dB_t \\ \gamma_i(\theta_t) &= \sum_j \frac{\partial \sigma_{ij}(\theta_t)}{\partial \theta_j}, \end{aligned}$$

- Choosing σ is not obvious, often based on the Hessian of π

The MALA Algorithm

- Is based on the $\pi-$limiting Langevin diffusion

$$dX_t = \frac{1}{2}\nabla \log \pi(X_t)dt + dB_t$$

- Euler discretisation of this diffusion suggests the Metropolis-Hastings proposal

$$q(\cdot|X_{(n-1)}) := X_{(n-1)} + \frac{h}{2}\nabla \log \pi(X_{(n-1)}) + h^{1/2}N(0, I_{d\times d})$$

  with the usual accept-reject formula
- MALA works well for "nice" examples, but is unstable for light-tailed π.
- Manifold MALA is based on

$$\begin{aligned} dX_t &= \left(\frac{\sigma(X_t)}{2}\nabla \log \pi(X_t) + \frac{\gamma(X_t)}{2}\right)dt + \sqrt{\sigma}(X_t)dB_t \\ \gamma_i(\theta_t) &= \sum_j \frac{\partial \sigma_{ij}(\theta_t)}{\partial \theta_j}, \end{aligned}$$

- Choosing σ is not obvious, often based on the Hessian of π.

The MALA Algorithm

- Is based on the $\pi-$limiting Langevin diffusion

$$dX_t = \frac{1}{2}\nabla \log \pi(X_t)dt + dB_t$$

- Euler discretisation of this diffusion suggests the Metropolis-Hastings proposal

$$q(\cdot|X_{(n-1)}) := X_{(n-1)} + \frac{h}{2}\nabla \log \pi(X_{(n-1)}) + h^{1/2}N(0, I_{d\times d})$$

  with the usual accept-reject formula
- MALA works well for "nice" examples, but is unstable for light-tailed π.
- Manifold MALA is based on

$$\begin{aligned} dX_t &= \left(\frac{\sigma(X_t)}{2}\nabla \log \pi(X_t) + \frac{\gamma(X_t)}{2}\right)dt + \sqrt{\sigma}(X_t)dB_t \\ \gamma_i(\theta_t) &= \sum_j \frac{\partial \sigma_{ij}(\theta_t)}{\partial \theta_j}, \end{aligned}$$

- Choosing σ is not obvious, often based on the Hessian of π.

The MALA Algorithm

- Is based on the $\pi-$limiting Langevin diffusion

$$dX_t = \frac{1}{2}\nabla \log \pi(X_t)dt + dB_t$$

- Euler discretisation of this diffusion suggests the Metropolis-Hastings proposal

$$q(\cdot|X_{(n-1)}) := X_{(n-1)} + \frac{h}{2}\nabla \log \pi(X_{(n-1)}) + h^{1/2}N(0, I_{d\times d})$$

  with the usual accept-reject formula
- MALA works well for "nice" examples, but is unstable for light-tailed π.
- Manifold MALA is based on

$$\begin{aligned} dX_t &= \left(\frac{\sigma(X_t)}{2}\nabla \log \pi(X_t) + \frac{\gamma(X_t)}{2}\right)dt + \sqrt{\sigma}(X_t)dB_t \\ \gamma_i(\theta_t) &= \sum_j \frac{\partial \sigma_{ij}(\theta_t)}{\partial \theta_j}, \end{aligned}$$

- Choosing σ is not obvious, often based on the Hessian of π

The MALA Algorithm

- Is based on the $\pi-$limiting Langevin diffusion

$$dX_t = \frac{1}{2}\nabla \log \pi(X_t)dt + dB_t$$

- Euler discretisation of this diffusion suggests the Metropolis-Hastings proposal

$$q(\cdot|X_{(n-1)}) := X_{(n-1)} + \frac{h}{2}\nabla \log \pi(X_{(n-1)}) + h^{1/2}N(0, I_{d\times d})$$

  with the usual accept-reject formula
- MALA works well for "nice" examples, but is unstable for light-tailed π.
- Manifold MALA is based on

$$\begin{aligned} dX_t &= \left(\frac{\sigma(X_t)}{2}\nabla \log \pi(X_t) + \frac{\gamma(X_t)}{2}\right)dt + \sqrt{\sigma}(X_t)dB_t \\ \gamma_i(\theta_t) &= \sum_j \frac{\partial \sigma_{ij}(\theta_t)}{\partial \theta_j}, \end{aligned}$$

- Choosing σ is not obvious, often based on the Hessian of π

Adaptive MCMC in 3 minutes (what it is?)

- Most MCMC algorithms need **tuning** to be efficient and reliable in large scale applications
- Tuning requires **computing time** and **human time** (performing and assessing trial runs) and typically **expert knowledge**
- Hand **tuning may not be practical**: too many variables, when to stop tuning, tuning criterion not clear, etc.
- **Adaptive MCMC is about tuning MCMC without human intervention**
- It uses the **trajectory so far** to tune the sampling kernel **on the fly** (so it is not a Markov chain anymore)

Adaptive MCMC in 3 minutes (what it is?)

- Most MCMC algorithms need **tuning** to be efficient and reliable in large scale applications
- Tuning requires **computing time** and **human time** (performing and assessing trial runs) and typically **expert knowledge**
- Hand **tuning may not be practical**: too many variables, when to stop tuning, tuning criterion not clear, etc.
- **Adaptive MCMC is about tuning MCMC without human intervention**
- It uses the **trajectory so far** to tune the sampling kernel **on the fly** (so it is not a Markov chain anymore)

Adaptive MCMC in 3 minutes (what it is?)

- Most MCMC algorithms need **tuning** to be efficient and reliable in large scale applications
- Tuning requires **computing time** and **human time** (performing and assessing trial runs) and typically **expert knowledge**
- Hand **tuning may not be practical**: too many variables, when to stop tuning, tuning criterion not clear, etc.
- **Adaptive MCMC is about tuning MCMC without human intervention**
- It uses the **trajectory so far** to tune the sampling kernel **on the fly** (so it is not a Markov chain anymore)

Adaptive MCMC in 3 minutes (what it is?)

- Most MCMC algorithms need **tuning** to be efficient and reliable in large scale applications
- Tuning requires **computing time** and **human time** (performing and assessing trial runs) and typically **expert knowledge**
- Hand **tuning may not be practical**: too many variables, when to stop tuning, tuning criterion not clear, etc.
- **Adaptive MCMC is about tuning MCMC without human intervention**
- It uses the **trajectory so far** to tune the sampling kernel **on the fly** (so it is not a Markov chain anymore)

Adaptive MCMC in 3 minutes (what it is?)

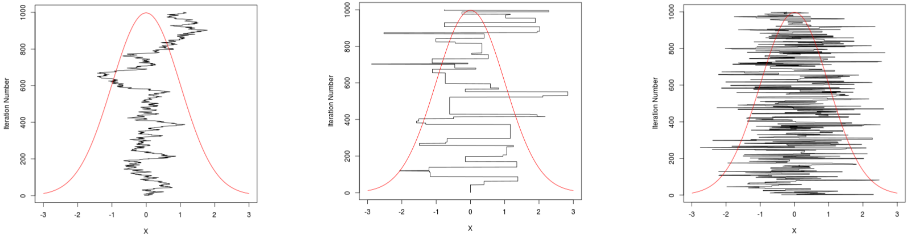
- Most MCMC algorithms need **tuning** to be efficient and reliable in large scale applications
- Tuning requires **computing time** and **human time** (performing and assessing trial runs) and typically **expert knowledge**
- Hand **tuning may not be practical**: too many variables, when to stop tuning, tuning criterion not clear, etc.
- **Adaptive MCMC is about tuning MCMC without human intervention**
- It uses the **trajectory so far** to tune the sampling kernel **on the fly** (so it is not a Markov chain anymore)

Adaptive MCMC in 3 minutes (3 examples)

- Random Walk Metropolis with proposal increments

$$Y_{n+1} \sim q_{\sigma}(X_n, \cdot) = X_n + \sigma N(0, Id).$$

Plots for different σ - Goldilock's principle



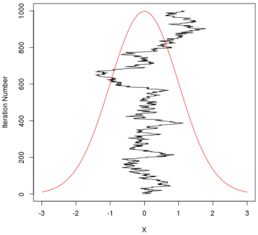
-

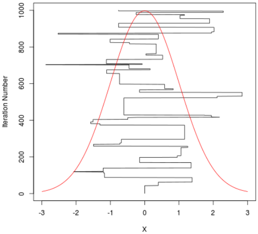
Adaptive MCMC in 3 minutes (3 examples)

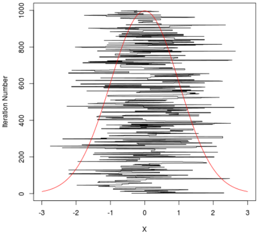
- Random Walk Metropolis with proposal increments

$$Y_{n+1} \sim q_{\sigma}(X_n, \cdot) = X_n + \sigma N(0, Id).$$

Plots for different σ - Goldilock's principle



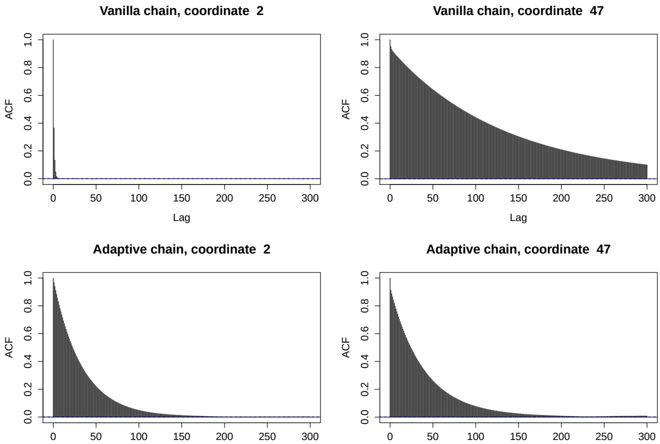




-

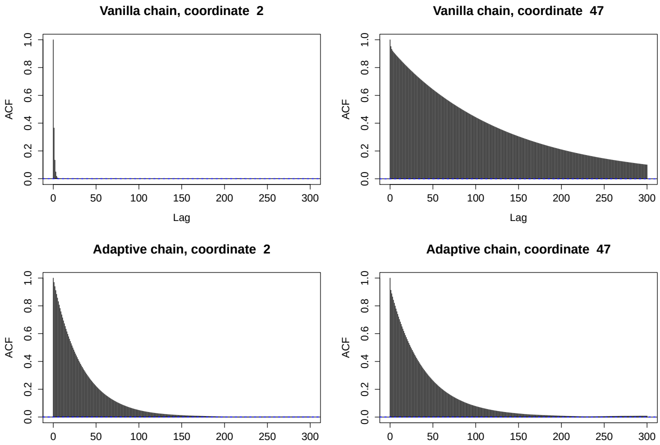
Adaptive MCMC in 3 minutes (3 examples)

- Random Scan Gibbs Sampler for 50d Truncated Multivariate Normals
  Are uniform $1/d$ selection probabilities optimal?



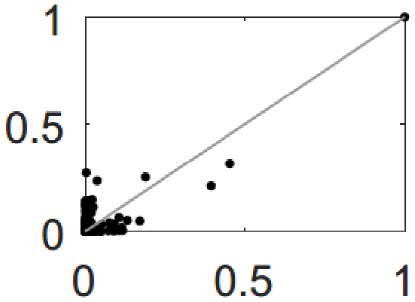
Adaptive MCMC in 3 minutes (3 examples)

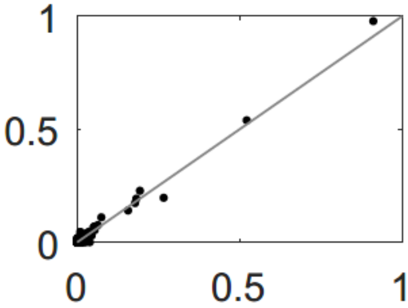
- Random Scan Gibbs Sampler for 50d Truncated Multivariate Normals
  Are uniform $1/d$ selection probabilities optimal?



Adaptive MCMC in 3 minutes (3 examples)

- Variable selection (p = 22576) - Metropolis type algorithms
  Plots of posterior inclusion probabilities Run 1 vs Run 2 (checking agreement)
  Standard Add-Swap-Delete proposal vs. an optimized non-local proposal

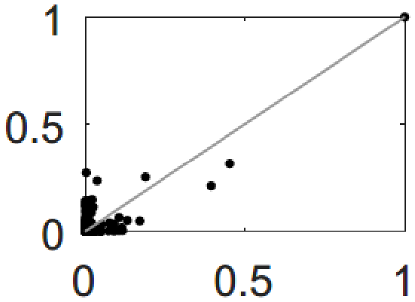


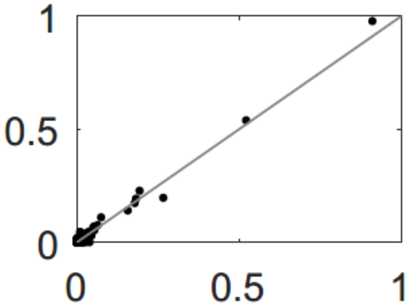


-

Adaptive MCMC in 3 minutes (3 examples)

- Variable selection (p = 22576) - Metropolis type algorithms
  Plots of posterior inclusion probabilities Run 1 vs Run 2 (checking agreement)
  Standard Add-Swap-Delete proposal vs. an optimized non-local proposal





-

Adaptive MCMC in 5 minutes (ingredients that we need)

- For a given MCMC class we need **a parameter to optimize**
- An optimization rule that is **mathematically sound**
- An optimization rule that is **computationally cheap**
- Need underpinning theory to **verify it is ergodic**
  (it is not Markovian - how do we know bizarre things don't happen??)
- It needs to **work in practice**

Adaptive MCMC in 5 minutes (ingredients that we need)

- For a given MCMC class we need **a parameter to optimize**
- An optimization rule that is **mathematically sound**
- An optimization rule that is **computationally cheap**
- Need underpinning theory to **verify it is ergodic**
  (it is not Markovian - how do we know bizarre things don't happen??)
- It needs to **work in practice**

Adaptive MCMC in 5 minutes (ingredients that we need)

- For a given MCMC class we need **a parameter to optimize**
- An optimization rule that is **mathematically sound**
- An optimization rule that is **computationally cheap**
- Need underpinning theory to **verify it is ergodic**
  (it is not Markovian - how do we know bizarre things don't happen??)
- It needs to **work in practice**

Adaptive MCMC in 5 minutes (ingredients that we need)

- For a given MCMC class we need **a parameter to optimize**
- An optimization rule that is **mathematically sound**
- An optimization rule that is **computationally cheap**
- Need underpinning theory to **verify it is ergodic**
  (it is not Markovian - how do we know bizarre things don't happen??)
- It needs to **work in practice**

Adaptive MCMC in 5 minutes (ingredients that we need)

- For a given MCMC class we need **a parameter to optimize**
- An optimization rule that is **mathematically sound**
- An optimization rule that is **computationally cheap**
- Need underpinning theory to **verify it is ergodic**
  (it is not Markovian - how do we know bizarre things don't happen??)
- It needs to **work in practice**

the usual MCMC setting

- let π be a target probability distribution on $\mathcal{X}$, typically arising as a posterior distribution in Bayesian inference,
- the goal is to evaluate

$$I := \int_{\mathcal{X}} f(x)\pi(dx).$$

- direct sampling from π is not possible or inefficient for example π is known up to a normalising constant
- MCMC approach is to simulate $(X_n)_{n\geq 0}$, an ergodic Markov chain with **transition kernel** P and limiting distribution π, and take ergodic averages as an estimate of I.
- the usual estimate

$$\hat{I} := \frac{1}{n}\sum_{k=t}^{t+n} f(X_k)$$

- **SLLN** for Markov chains holds under very mild conditions
- **CLT** for Markov chains holds under some additional assumptions and is verifiable in many situations of interest

the usual MCMC setting

- let π be a target probability distribution on $\mathcal{X}$, typically arising as a posterior distribution in Bayesian inference,
- the goal is to evaluate

$$I := \int_{\mathcal{X}} f(x)\pi(dx).$$

- direct sampling from π is not possible or inefficient for example π is known up to a normalising constant
- MCMC approach is to simulate $(X_n)_{n\geq 0}$, an ergodic Markov chain with **transition kernel** P and limiting distribution π, and take ergodic averages as an estimate of I.
- the usual estimate

$$\hat{I} := \frac{1}{n}\sum_{k=t}^{t+n} f(X_k)$$

- **SLLN** for Markov chains holds under very mild conditions
- **CLT** for Markov chains holds under some additional assumptions and is verifiable in many situations of interest

the usual MCMC setting

- let π be a target probability distribution on $\mathcal{X}$, typically arising as a posterior distribution in Bayesian inference,
- the goal is to evaluate

$$I := \int_{\mathcal{X}} f(x)\pi(dx).$$

- direct sampling from π is not possible or inefficient for example π is known up to a normalising constant
- MCMC approach is to simulate $(X_n)_{n\geq 0}$, an ergodic Markov chain with **transition kernel** P and limiting distribution π, and take ergodic averages as an estimate of I.
- the usual estimate

$$\hat{I} := \frac{1}{n}\sum_{k=t}^{t+n} f(X_k)$$

- **SLLN** for Markov chains holds under very mild conditions
- **CLT** for Markov chains holds under some additional assumptions and is verifiable in many situations of interest

the usual MCMC setting

- let π be a target probability distribution on $\mathcal{X}$, typically arising as a posterior distribution in Bayesian inference,
- the goal is to evaluate

$$I := \int_{\mathcal{X}} f(x)\pi(dx).$$

- direct sampling from π is not possible or inefficient
  for example π is known up to a normalising constant
- MCMC approach is to simulate $(X_n)_{n\geq 0}$, an ergodic Markov chain with **transition kernel** P and limiting distribution π, and take ergodic averages as an estimate of I.
- the usual estimate

$$\hat{I} := \frac{1}{n}\sum_{k=t}^{t+n} f(X_k)$$

- **SLLN** for Markov chains holds under very mild conditions
- **CLT** for Markov chains holds under some additional assumptions and is verifiable in many situations of interest

the usual MCMC setting

- let π be a target probability distribution on $\mathcal{X}$, typically arising as a posterior distribution in Bayesian inference,
- the goal is to evaluate

$$I := \int_{\mathcal{X}} f(x)\pi(dx).$$

- direct sampling from π is not possible or inefficient
  for example π is known up to a normalising constant
- MCMC approach is to simulate $(X_n)_{n\geq 0}$, an ergodic Markov chain with **transition kernel** P and limiting distribution π, and take ergodic averages as an estimate of I.
- the usual estimate

$$\hat{I} := \frac{1}{n}\sum_{k=t}^{t+n} f(X_k)$$

- **SLLN** for Markov chains holds under very mild conditions
- **CLT** for Markov chains holds under some additional assumptions and is verifiable in many situations of interest

the usual MCMC setting

- let π be a target probability distribution on $\mathcal{X}$, typically arising as a posterior distribution in Bayesian inference,
- the goal is to evaluate

$$I := \int_{\mathcal{X}} f(x)\pi(dx).$$

- direct sampling from π is not possible or inefficient
  for example π is known up to a normalising constant
- MCMC approach is to simulate $(X_n)_{n\geq 0}$, an ergodic Markov chain with **transition kernel** P and limiting distribution π, and take ergodic averages as an estimate of I.
- the usual estimate

$$\hat{I} := \frac{1}{n}\sum_{k=t}^{t+n} f(X_k)$$

- **SLLN** for Markov chains holds under very mild conditions
- **CLT** for Markov chains holds under some additional assumptions and is verifiable in many situations of interest

Reversibility and stationarity

- How to design P so that X_n converges in distribution to π ?
- **Definition.** P is reversible with respect to π if

$$\pi(x)P(x,y) = \pi(y)P(y,x)$$

as measures on $\mathcal{X} \times \mathcal{X}$

- **Lemma.** If P is reversible with respect to π then $\pi P = \pi$, so it is also stationary.

Reversibility and stationarity

- How to design P so that X_n converges in distribution to π ?
- **Definition.** P is reversible with respect to π if

$$\pi(x)P(x,y) = \pi(y)P(y,x)$$

  as measures on $\mathcal{X} \times \mathcal{X}$
- **Lemma.** If P is reversible with respect to π then $\pi P = \pi$, so it is also stationary.

Reversibility and stationarity

- How to design P so that X_n converges in distribution to π ?
- **Definition.** P is reversible with respect to π if

$$\pi(x)P(x,y) = \pi(y)P(y,x)$$

  as measures on $\mathcal{X} \times \mathcal{X}$
- **Lemma.** If P is reversible with respect to π then $\pi P = \pi$, so it is also stationary.

The Metropolis algorithm

- **Idea.** Take any transition kernel Q with transition densities $q(x,y)$ and make it reversible with respect to π
- **Algorithm.** Given X_n
  sample $Y_{n+1} \sim Q(X_n, \cdot)$
- with probability $\alpha(X_n, Y_{n+1})$ set $X_{n+1} = Y_{n+1}$, otherwise set $X_{n+1} = X_n$
- where

$$\alpha(x,y) = \min\{1, \frac{\pi(y)q(y,x)}{\pi(x)q(x,y)}\}.$$

- Under mild assumptions on Q the algorithm is ergodic.
- However it's performance depends heavily on Q
- is is **difficult** to design the proposal Q so that P has **good convergence properties**, especially if $\mathcal{X}$ is high dimensional

The Metropolis algorithm

- **Idea.** Take any transition kernel Q with transition densities $q(x, y)$ and make it reversible with respect to π
- **Algorithm.** Given X_n
  sample $Y_{n+1} \sim Q(X_n, \cdot)$
- with probability $\alpha(X_n, Y_{n+1})$ set $X_{n+1} = Y_{n+1}$, otherwise set $X_{n+1} = X_n$
- where

$$\alpha(x, y) = \min\{1, \frac{\pi(y)q(y, x)}{\pi(x)q(x, y)}\}.$$

- Under mild assumptions on Q the algorithm is ergodic.
- However it's performance depends heavily on Q
- is is **difficult** to design the proposal Q so that P has **good convergence properties**, especially if $\mathcal{X}$ is high dimensional

The Metropolis algorithm

- **Idea.** Take any transition kernel Q with transition densities $q(x, y)$ and make it reversible with respect to π
- **Algorithm.** Given X_n
  sample $Y_{n+1} \sim Q(X_n, \cdot)$
- with probability $\alpha(X_n, Y_{n+1})$ set $X_{n+1} = Y_{n+1}$, otherwise set $X_{n+1} = X_n$
- where

$$\alpha(x, y) = \min\{1, \frac{\pi(y)q(y, x)}{\pi(x)q(x, y)}\}.$$

- Under mild assumptions on Q the algorithm is ergodic.
- However it's performance depends heavily on Q
- is is **difficult** to design the proposal Q so that P has **good convergence properties**, especially if $\mathcal{X}$ is high dimensional

The Metropolis algorithm

- **Idea.** Take any transition kernel Q with transition densities $q(x, y)$ and make it reversible with respect to π
- **Algorithm.** Given X_n
  sample $Y_{n+1} \sim Q(X_n, \cdot)$
- with probability $\alpha(X_n, Y_{n+1})$ set $X_{n+1} = Y_{n+1}$, otherwise set $X_{n+1} = X_n$
- where

$$\alpha(x, y) = \min\{1, \frac{\pi(y)q(y, x)}{\pi(x)q(x, y)}\}.$$

- Under mild assumptions on Q the algorithm is ergodic.
- However it's performance depends heavily on Q
- is is **difficult** to design the proposal Q so that P has **good convergence properties**, especially if $\mathcal{X}$ is high dimensional

The Metropolis algorithm

- **Idea.** Take any transition kernel Q with transition densities $q(x, y)$ and make it reversible with respect to π
- **Algorithm.** Given X_n
  sample $Y_{n+1} \sim Q(X_n, \cdot)$
- with probability $\alpha(X_n, Y_{n+1})$ set $X_{n+1} = Y_{n+1}$, otherwise set $X_{n+1} = X_n$
- where

$$\alpha(x, y) = \min\{1, \frac{\pi(y)q(y, x)}{\pi(x)q(x, y)}\}.$$

- Under mild assumptions on Q the algorithm is ergodic.
- However it's performance depends heavily on Q
- is is **difficult** to design the proposal Q so that P has **good convergence properties**, especially if $\mathcal{X}$ is high dimensional

The Metropolis algorithm

- **Idea.** Take any transition kernel Q with transition densities $q(x, y)$ and make it reversible with respect to π
- **Algorithm.** Given X_n sample $Y_{n+1} \sim Q(X_n, \cdot)$
- with probability $\alpha(X_n, Y_{n+1})$ set $X_{n+1} = Y_{n+1}$, otherwise set $X_{n+1} = X_n$
- where

$$\alpha(x, y) = \min\{1, \frac{\pi(y)q(y, x)}{\pi(x)q(x, y)}\}.$$

- Under mild assumptions on Q the algorithm is ergodic.
- However it's performance depends heavily on Q
- is is **difficult** to design the proposal Q so that P has **good convergence properties**, especially if $\mathcal{X}$ is high dimensional

the scaling problem

- take Random Walk Metropolis with proposal increments
- $$Y_{n+1} \sim q_\sigma(X_n, \cdot) = X_n + \sigma N(0, Id).$$
- what happens if σ is small?

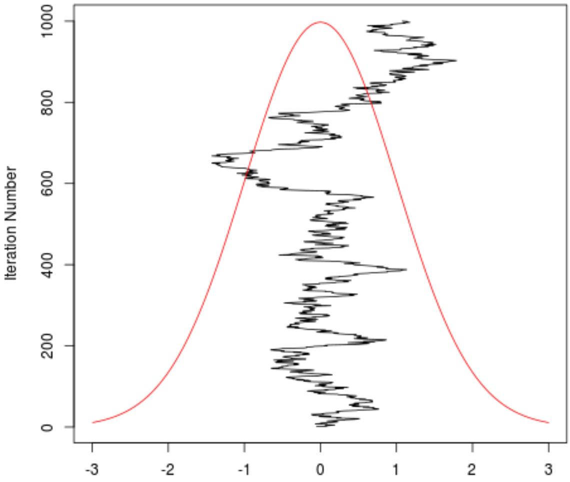
the scaling problem

- take Random Walk Metropolis with proposal increments
- $$Y_{n+1} \sim q_{\sigma}(X_n, \cdot) = X_n + \sigma N(0, Id).$$
- what happens if σ is small?

the scaling problem

- take Random Walk Metropolis with proposal increments
- $$Y_{n+1} \sim q_{\sigma}(X_n, \cdot) = X_n + \sigma N(0, Id).$$
- what happens if σ is small?

small sigma...



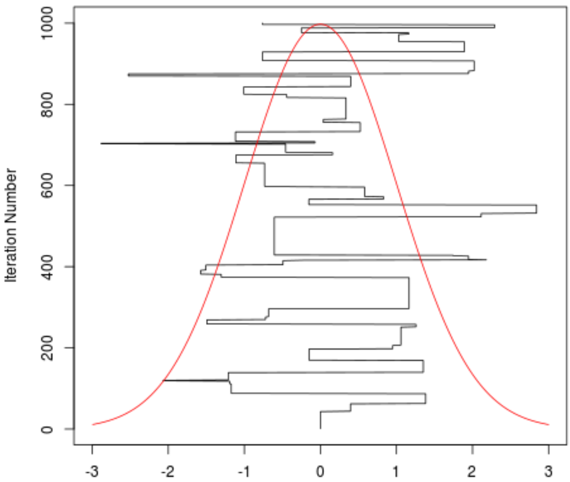
the scaling problem

- take Random Walk Metropolis with proposal increments
- $$Y_{n+1} \sim q_{\sigma}(X_n, \cdot) = X_n + \sigma N(0, Id).$$
- what happens if σ is small?
- what happens if σ is large?

the scaling problem

- take Random Walk Metropolis with proposal increments
- $$Y_{n+1} \sim q_{\sigma}(X_n, \cdot) = X_n + \sigma N(0, Id).$$
- what happens if σ is small?
- what happens if σ is large?

large sigma...



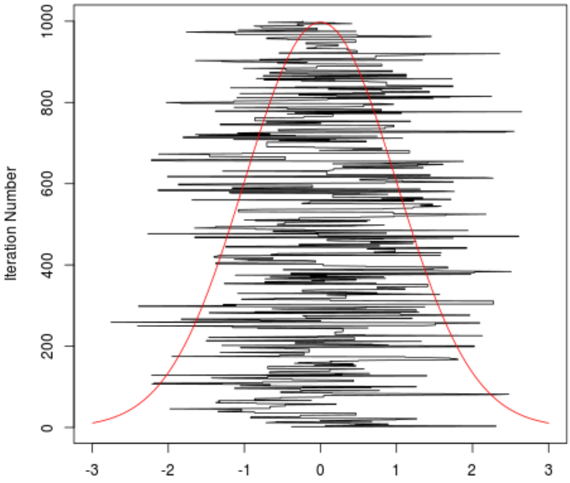
the scaling problem

- take Random Walk Metropolis with proposal increments
- $$Y_{n+1} \sim q_{\sigma}(X_n, \cdot) = X_n + \sigma N(0, Id).$$
- what happens if σ is small?
- what happens if σ is large?
- so σ should be neither too small, nor too large (known as Goldilocks principle)

the scaling problem

- take Random Walk Metropolis with proposal increments
- $$Y_{n+1} \sim q_{\sigma}(X_n, \cdot) = X_n + \sigma N(0, Id).$$
- what happens if σ is small?
- what happens if σ is large?
- so σ should be neither too small, nor too large (known as Goldilocks principle)

not too small and not too large...



diffusion limit [RGG97]

- take Random Walk Metropolis with proposal increments
- $$Y_{n+1} \sim q_\sigma(X_n, \cdot) = X_n + \sigma N(0, Id).$$
- σ should be neither too small, nor too large (known as Goldilocks principle)
- but how to choose it?
- if the dimension of $\mathcal{X}$ goes to ∞, e.g. $\mathcal{X} = \mathbb{R}^d$, and $d \to \infty$,
- if the proposal is set as $Q = N(x, \frac{l^2}{d} I_d)$ for fixed $l > 0$,
- if we consider
$$Z_t = d^{-1/2} X^{(1)}_{\lfloor dt \rfloor}$$
- then Z_t converges to the Langevin diffusion
$$dZ_t = h(l)^{1/2} dB_t + \frac{1}{2} h(l) \nabla \log \pi(Z_t) dt$$

diffusion limit [RGG97]

- take Random Walk Metropolis with proposal increments
- $$Y_{n+1} \sim q_\sigma(X_n, \cdot) = X_n + \sigma N(0, Id).$$
- σ should be neither too small, nor too large (known as Goldilocks principle)
- but how to choose it?
- if the dimension of $\mathcal{X}$ goes to ∞, e.g. $\mathcal{X} = \mathbb{R}^d$, and $d \to \infty$,
- if the proposal is set as $Q = N(x, \frac{l^2}{d} I_d)$ for fixed $l > 0$,
- if we consider
$$Z_t = d^{-1/2} X^{(1)}_{\lfloor dt \rfloor}$$
- then Z_t converges to the Langevin diffusion
$$dZ_t = h(l)^{1/2} dB_t + \frac{1}{2} h(l) \nabla \log \pi(Z_t) dt$$

diffusion limit [RGG97]

- take Random Walk Metropolis with proposal increments
- $$Y_{n+1} \sim q_{\sigma}(X_n, \cdot) = X_n + \sigma N(0, Id).$$
- σ should be neither too small, nor too large (known as Goldilocks principle)
- but how to choose it?
- if the dimension of $\mathcal{X}$ goes to ∞, e.g. $\mathcal{X} = \mathbb{R}^d$, and $d \to \infty$,
- if the proposal is set as $Q = N(x, \frac{l^2}{d} I_d)$ for fixed $l > 0$,
- if we consider
$$Z_t = d^{-1/2} X^{(1)}_{\lfloor dt \rfloor}$$
- then Z_t converges to the Langevin diffusion
$$dZ_t = h(l)^{1/2} dB_t + \frac{1}{2} h(l) \nabla \log \pi(Z_t) dt$$

diffusion limit [RGG97]

- take Random Walk Metropolis with proposal increments
- $$Y_{n+1} \sim q_\sigma(X_n, \cdot) = X_n + \sigma N(0, Id).$$
- σ should be neither too small, nor too large (known as Goldilocks principle)
- but how to choose it?
- if the dimension of $\mathcal{X}$ goes to ∞, e.g. $\mathcal{X} = \mathbb{R}^d$, and $d \to \infty$,
- if the proposal is set as $Q = N(x, \frac{l^2}{d} I_d)$ for fixed $l > 0$,
- if we consider
$$Z_t = d^{-1/2} X^{(1)}_{\lfloor dt \rfloor}$$
- then Z_t converges to the Langevin diffusion
$$dZ_t = h(l)^{1/2} dB_t + \frac{1}{2} h(l) \nabla \log \pi(Z_t) dt$$

diffusion limit [RGG97]

- take Random Walk Metropolis with proposal increments
- $$Y_{n+1} \sim q_{\sigma}(X_n, \cdot) = X_n + \sigma N(0, Id).$$
- σ should be neither too small, nor too large (known as Goldilocks principle)
- but how to choose it?
- if the dimension of $\mathcal{X}$ goes to ∞, e.g. $\mathcal{X} = \mathbb{R}^d$, and $d \to \infty$,
- if the proposal is set as $Q = N(x, \frac{l^2}{d} I_d)$ for fixed $l > 0$,
- if we consider
$$Z_t = d^{-1/2} X^{(1)}_{\lfloor dt \rfloor}$$
- then Z_t converges to the Langevin diffusion
$$dZ_t = h(l)^{1/2} dB_t + \frac{1}{2} h(l) \nabla \log \pi(Z_t) dt$$

diffusion limit [RGG97]

- take Random Walk Metropolis with proposal increments
- $$Y_{n+1} \sim q_{\sigma}(X_n, \cdot) = X_n + \sigma N(0, Id).$$
- σ should be neither too small, nor too large (known as Goldilocks principle)
- but how to choose it?
- if the dimension of $\mathcal{X}$ goes to ∞, e.g. $\mathcal{X} = \mathbb{R}^d$, and $d \to \infty$,
- if the proposal is set as $Q = N(x, \frac{l^2}{d} I_d)$ for fixed $l > 0$,
- if we consider
$$Z_t = d^{-1/2} X^{(1)}_{\lfloor dt \rfloor}$$
- then Z_t converges to the Langevin diffusion
$$dZ_t = h(l)^{1/2} dB_t + \frac{1}{2} h(l) \nabla \log \pi(Z_t) dt$$

optimal acceptance rate [RGG97]

- Z_t converges to the Langevin diffusion

$$dZ_t = h(l)^{1/2}dB_t + \frac{1}{2}h(l)\nabla \log \pi(Z_t)dt$$

- where $h(l) = 2l^2\Phi(-Cl/2)$ is the speed of the diffusion and $A(l) = 2\Phi(Cl/2)$ is the asymptotic acceptance rate.
- maximising the speed $h(l)$ yields the optimal acceptance rate

$$A(l) = 0.234$$

  which is independent of the target distribution π
- it is a remarkable result since it gives a simple criterion (and the same for all target distributions π) to assess how well the Random Walk Metropolis is performing.

optimal acceptance rate [RGG97]

- Z_t converges to the Langevin diffusion

$$dZ_t = h(l)^{1/2} dB_t + \frac{1}{2} h(l) \nabla \log \pi(Z_t) dt$$

- where $h(l) = 2l^2 \Phi(-Cl/2)$ is the speed of the diffusion and $A(l) = 2\Phi(Cl/2)$ is the asymptotic acceptance rate.
- maximising the speed $h(l)$ yields the optimal acceptance rate

$$A(l) = 0.234$$

which is independent of the target distribution π

- it is a remarkable result since it gives a simple criterion (and the same for all target distributions π) to assess how well the Random Walk Metropolis is performing.

optimal acceptance rate [RGG97]

- Z_t converges to the Langevin diffusion

$$dZ_t = h(l)^{1/2} dB_t + \frac{1}{2} h(l) \nabla \log \pi(Z_t) dt$$

- where $h(l) = 2l^2 \Phi(-Cl/2)$ is the speed of the diffusion and $A(l) = 2\Phi(Cl/2)$ is the asymptotic acceptance rate.
- maximising the speed $h(l)$ yields the optimal acceptance rate

$$A(l) = 0.234$$

  which is independent of the target distribution π
- it is a remarkable result since it gives a simple criterion (and the same for all target distributions π) to assess how well the Random Walk Metropolis is performing.

optimal acceptance rate [RGG97]

- Z_t converges to the Langevin diffusion

$$dZ_t = h(l)^{1/2}dB_t + \frac{1}{2}h(l)\nabla \log \pi(Z_t)dt$$

- where $h(l) = 2l^2\Phi(-Cl/2)$ is the speed of the diffusion and $A(l) = 2\Phi(Cl/2)$ is the asymptotic acceptance rate.
- maximising the speed $h(l)$ yields the optimal acceptance rate

$$A(l) = 0.234$$

  which is independent of the target distribution π
- it is a remarkable result since it gives a simple criterion (and the same for all target distributions π) to assess how well the Random Walk Metropolis is performing.

the scaling problem cd

- take Random Walk Metropolis with proposal increments
- $$Y_{n+1} \sim q_\sigma(X_n, \cdot) = X_n + \sigma N(0, Id).$$
- so the theory says the optimal average acceptance rate
$$\bar{\alpha} := \int \int \alpha(x,y) q_\sigma(x, dy)\pi(dx)$$
should be approximately $\alpha^* = 0.234$
- however it is not possible to compute σ^* for which $\bar{\alpha} = \alpha^*$.
- It is very tempting to adjust σ on the fly while simulation progress
- some reasons:
  - when to stop estimating $\bar{\alpha}$? (to increase or decrease σ)
  - we may be in a Metropolis within Gibbs setting of dimension 10000

the scaling problem cd

- take Random Walk Metropolis with proposal increments
- $$Y_{n+1} \sim q_{\sigma}(X_n, \cdot) = X_n + \sigma N(0, Id).$$
- so the theory says the optimal average acceptance rate
$$\bar{\alpha} := \int \int \alpha(x,y) q_{\sigma}(x, dy) \pi(dx)$$
should be approximately $\alpha^* = 0.234$
- however it is not possible to compute σ^* for which $\bar{\alpha} = \alpha^*$.
- It is very tempting to adjust σ on the fly while simulation progress
- some reasons:
  - when to stop estimating $\bar{\alpha}$? (to increase or decrease σ)
  - we may be in a Metropolis within Gibbs setting of dimension 10000

the scaling problem cd

- take Random Walk Metropolis with proposal increments
- $$Y_{n+1} \sim q_\sigma(X_n, \cdot) = X_n + \sigma N(0, Id).$$
- so the theory says the optimal average acceptance rate

$$\bar{\alpha} := \int\int \alpha(x,y) q_\sigma(x, dy) \pi(dx)$$

should be approximately $\alpha^* = 0.234$

- however it is not possible to compute σ^* for which $\bar{\alpha} = \alpha^*$.
- It is very tempting to adjust σ on the fly while simulation progress
- some reasons:
  - when to stop estimating $\bar{\alpha}$? (to increase or decrease σ)
  - we may be in a Metropolis within Gibbs setting of dimension 10000

the scaling problem cd

- take Random Walk Metropolis with proposal increments
- $$Y_{n+1} \sim q_\sigma(X_n, \cdot) = X_n + \sigma N(0, Id).$$
- so the theory says the optimal average acceptance rate
  $$\bar{\alpha} := \int \int \alpha(x,y) q_\sigma(x, dy) \pi(dx)$$
  should be approximately $\alpha^* = 0.234$
- however it is not possible to compute σ^* for which $\bar{\alpha} = \alpha^*$.
- It is very tempting to adjust σ on the fly while simulation progress
- some reasons:
  - when to stop estimating $\bar{\alpha}$? (to increase or decrease σ)
  - we may be in a Metropolis within Gibbs setting of dimension 10000

the scaling problem cd

- take Random Walk Metropolis with proposal increments
- $$Y_{n+1} \sim q_\sigma(X_n, \cdot) = X_n + \sigma N(0, Id).$$
- so the theory says the optimal average acceptance rate
$$\bar{\alpha} := \int \int \alpha(x,y) q_\sigma(x, dy) \pi(dx)$$
should be approximately $\alpha^* = 0.234$
- however it is not possible to compute σ^* for which $\bar{\alpha} = \alpha^*$.
- It is very tempting to adjust σ on the fly while simulation progress
- some reasons:
  - when to stop estimating $\bar{\alpha}$? (to increase or decrease σ)
  - we may be in a Metropolis within Gibbs setting of dimension 10000

the scaling problem cd

- take Random Walk Metropolis with proposal increments
- $$Y_{n+1} \sim q_\sigma(X_n, \cdot) = X_n + \sigma N(0, Id).$$
- so the theory says the optimal average acceptance rate
$$\bar{\alpha} := \int\int \alpha(x,y) q_\sigma(x, dy)\pi(dx)$$
should be approximately $\alpha^* = 0.234$
- however it is not possible to compute σ^* for which $\bar{\alpha} = \alpha^*$.
- It is very tempting to adjust σ on the fly while simulation progress
- some reasons:
  - when to stop estimating $\bar{\alpha}$? (to increase or decrease σ)
  - we may be in a Metropolis within Gibbs setting of dimension 10000

the Adaptive Scaling Algorithm

1. draw proposal
$$Y_{n+1} \sim q_{\sigma_n}(X_n, \cdot) = X_n + \sigma_n N(0, Id),$$
2. Set X_{n+1} according to the usual Metropolis acceptance rate $\alpha(X_n, Y_{n+1})$.
3. Update scale by
$$\log \sigma_{n+1} = \log \sigma_n + \gamma_n(\alpha(X_n, Y_{n+1}) - \alpha^*)$$
where $\gamma_n \to 0$.

- Recall we follow a very precise mathematical advice from diffusion limit analysis [RGG97]
- The algorithm dates back to [GRS98]
(a slightly different version making use of regenerations)
- Exactly this version analyzed in [Vih09]

the Adaptive Scaling Algorithm

1. draw proposal
$$Y_{n+1} \sim q_{\sigma_n}(X_n, \cdot) = X_n + \sigma_n N(0, Id),$$
2. Set X_{n+1} according to the usual Metropolis acceptance rate $\alpha(X_n, Y_{n+1})$.
3. Update scale by
$$\log \sigma_{n+1} = \log \sigma_n + \gamma_n(\alpha(X_n, Y_{n+1}) - \alpha^*)$$
where $\gamma_n \to 0$.

- Recall we follow a very precise mathematical advice from diffusion limit analysis [RGG97]
- The algorithm dates back to [GRS98]
(a slightly different version making use of regenerations)
- Exactly this version analyzed in [Vih09]

the Adaptive Scaling Algorithm

1. draw proposal
$$Y_{n+1} \sim q_{\sigma_n}(X_n, \cdot) = X_n + \sigma_n N(0, Id),$$
2. Set X_{n+1} according to the usual Metropolis acceptance rate $\alpha(X_n, Y_{n+1})$.
3. Update scale by
$$\log \sigma_{n+1} = \log \sigma_n + \gamma_n(\alpha(X_n, Y_{n+1}) - \alpha^*)$$
where $\gamma_n \to 0$.

- Recall we follow a very precise mathematical advice from diffusion limit analysis [RGG97]
- The algorithm dates back to [GRS98]
(a slightly different version making use of regenerations)
- Exactly this version analyzed in [Vih09]

the Adaptive Scaling Algorithm

1. draw proposal
$$Y_{n+1} \sim q_{\sigma_n}(X_n, \cdot) = X_n + \sigma_n N(0, Id),$$
2. Set X_{n+1} according to the usual Metropolis acceptance rate $\alpha(X_n, Y_{n+1})$.
3. Update scale by
$$\log \sigma_{n+1} = \log \sigma_n + \gamma_n(\alpha(X_n, Y_{n+1}) - \alpha^*)$$
where $\gamma_n \to 0$.

- Recall we follow a very precise mathematical advice from diffusion limit analysis [RGG97]
- The algorithm dates back to [GRS98]
(a slightly different version making use of regenerations)
- Exactly this version analyzed in [Vih09]

the Adaptive Scaling Algorithm

1. draw proposal
$$Y_{n+1} \sim q_{\sigma_n}(X_n, \cdot) = X_n + \sigma_n N(0, Id),$$
2. Set X_{n+1} according to the usual Metropolis acceptance rate $\alpha(X_n, Y_{n+1})$.
3. Update scale by
$$\log \sigma_{n+1} = \log \sigma_n + \gamma_n(\alpha(X_n, Y_{n+1}) - \alpha^*)$$
where $\gamma_n \to 0$.

- Recall we follow a very precise mathematical advice from diffusion limit analysis [RGG97]
- The algorithm dates back to [GRS98]
(a slightly different version making use of regenerations)
- Exactly this version analyzed in [Vih09]

the Adaptive Scaling Algorithm

1. draw proposal
$$Y_{n+1} \sim q_{\sigma_n}(X_n, \cdot) = X_n + \sigma_n N(0, Id),$$
2. Set X_{n+1} according to the usual Metropolis acceptance rate $\alpha(X_n, Y_{n+1})$.
3. Update scale by
$$\log \sigma_{n+1} = \log \sigma_n + \gamma_n(\alpha(X_n, Y_{n+1}) - \alpha^*)$$
where $\gamma_n \to 0$.

- Recall we follow a very precise mathematical advice from diffusion limit analysis [RGG97]
- The algorithm dates back to [GRS98]
(a slightly different version making use of regenerations)
- Exactly this version analyzed in [Vih09]

Success story of Adaptive Scaling

- The adaptation rule is mathematically appealing (diffusion limit)
- The adaptation rule is computationally simple (acceptance rate)
- It works in applications (seems to improve convergence significantly)
- Improves convergence even in settings that are neither high dimensional, nor satisfy other assumptions needed for the diffusion limit
-
- Adaptive scaling beyond Metropolis-Hastings?
- YES. Similar optimal scaling results are available for MALA, HMC, etc.
- Every optimal scaling result can be used to design an adaptive version of the algorithm!

Success story of Adaptive Scaling

- The adaptation rule is mathematically appealing (diffusion limit)
- The adaptation rule is computationally simple (acceptance rate)
- It works in applications (seems to improve convergence significantly)
- Improves convergence even in settings that are neither high dimensional, nor satisfy other assumptions needed for the diffusion limit
-
- Adaptive scaling beyond Metropolis-Hastings?
- **YES.** Similar optimal scaling results are available for MALA, HMC, etc.
- Every optimal scaling result can be used to design an adaptive version of the algorithm!

Success story of Adaptive Scaling

- The adaptation rule is mathematically appealing (diffusion limit)
- The adaptation rule is computationally simple (acceptance rate)
- It works in applications (seems to improve convergence significantly)
- Improves convergence even in settings that are neither high dimensional, nor satisfy other assumptions needed for the diffusion limit
-
- Adaptive scaling beyond Metropolis-Hastings?
- YES. Similar optimal scaling results are available for MALA, HMC, etc.
- Every optimal scaling result can be used to design an adaptive version of the algorithm!

Success story of Adaptive Scaling

- The adaptation rule is mathematically appealing (diffusion limit)
- The adaptation rule is computationally simple (acceptance rate)
- It works in applications (seems to improve convergence significantly)
- Improves convergence even in settings that are neither high dimensional, nor satisfy other assumptions needed for the diffusion limit
-
- Adaptive scaling beyond Metropolis-Hastings?
- YES. Similar optimal scaling results are available for MALA, HMC, etc.
- Every optimal scaling result can be used to design an adaptive version of the algorithm!

Success story of Adaptive Scaling

- The adaptation rule is mathematically appealing (diffusion limit)
- The adaptation rule is computationally simple (acceptance rate)
- It works in applications (seems to improve convergence significantly)
- Improves convergence even in settings that are neither high dimensional, nor satisfy other assumptions needed for the diffusion limit
-
- Adaptive scaling beyond Metropolis-Hastings?
- YES. Similar optimal scaling results are available for MALA, HMC, etc.
- Every optimal scaling result can be used to design an adaptive version of the algorithm!

Success story of Adaptive Scaling

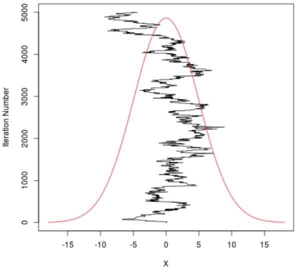
- The adaptation rule is mathematically appealing (diffusion limit)
- The adaptation rule is computationally simple (acceptance rate)
- It works in applications (seems to improve convergence significantly)
- Improves convergence even in settings that are neither high dimensional, nor satisfy other assumptions needed for the diffusion limit
-
- Adaptive scaling beyond Metropolis-Hastings?
- **YES.** Similar optimal scaling results are available for MALA, HMC, etc.
- Every optimal scaling result can be used to design an adaptive version of the algorithm!

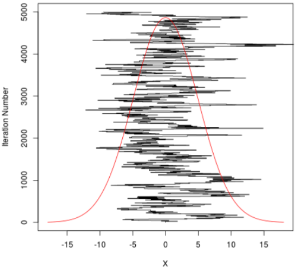
Success story of Adaptive Scaling

- The adaptation rule is mathematically appealing (diffusion limit)
- The adaptation rule is computationally simple (acceptance rate)
- It works in applications (seems to improve convergence significantly)
- Improves convergence even in settings that are neither high dimensional, nor satisfy other assumptions needed for the diffusion limit
-
- Adaptive scaling beyond Metropolis-Hastings?
- **YES.** Similar optimal scaling results are available for MALA, HMC, etc.
- Every optimal scaling result can be used to design an adaptive version of the algorithm!

Adaptive Metropolis algorithm

- Optimal scaling is not the whole story for optimizing the RWM!
- Take target π to be a 20-dimensional $N(0, \Sigma)$ with highly irregular Σ.
- Both of these Metropolis-Hastings are optimally scaled to have acc rate ≈ 0.23



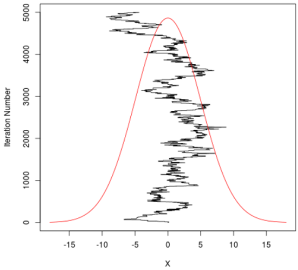


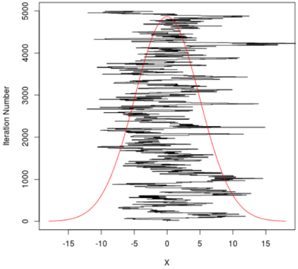
-
- However, the proposal increments are of the form

$$q_\theta = \sigma N(0, Id) \quad \text{and} \quad q_\theta = \sigma N(0, \Sigma)$$

Adaptive Metropolis algorithm

- Optimal scaling is not the whole story for optimizing the RWM!
- Take target π to be a 20-dimensional $N(0, \Sigma)$ with highly irregular Σ.
- Both of these Metropolis-Hastings are optimally scaled to have acc rate ≈ 0.23



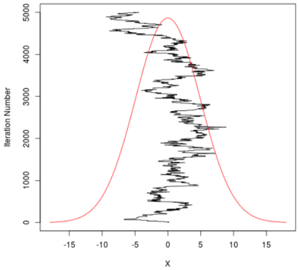


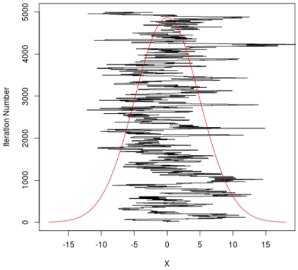
- However, the proposal increments are of the form

$$q_\theta = \sigma N(0, Id) \quad \text{and} \quad q_\theta = \sigma N(0, \Sigma)$$

Adaptive Metropolis algorithm

- Optimal scaling is not the whole story for optimizing the RWM!
- Take target π to be a 20-dimensional $N(0, \Sigma)$ with highly irregular Σ.
- Both of these Metropolis-Hastings are optimally scaled to have acc rate ≈ 0.23



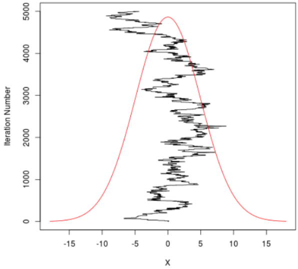


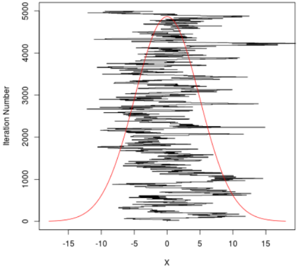
-
- However, the proposal increments are of the form

$$q_\theta = \sigma N(0, Id) \quad \text{and} \quad q_\theta = \sigma N(0, \Sigma)$$

Adaptive Metropolis algorithm

- Optimal scaling is not the whole story for optimizing the RWM!
- Take target π to be a 20-dimensional $N(0, \Sigma)$ with highly irregular Σ.
- Both of these Metropolis-Hastings are optimally scaled to have acc rate ≈ 0.23



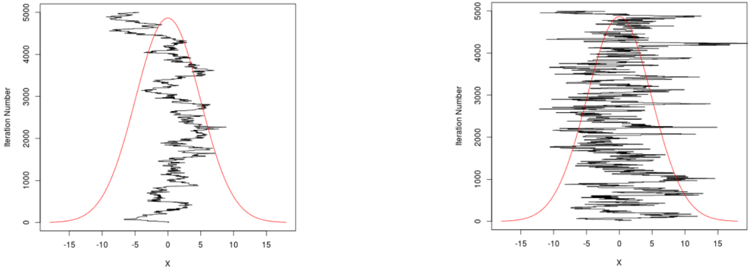


-
- However, the proposal increments are of the form

$$q_\theta = \sigma N(0, Id) \quad \text{and} \quad q_\theta = \sigma N(0, \Sigma)$$

Adaptive Metropolis algorithm

- Optimal scaling is not the whole story for optimizing the RWM!
- Take target π to be a 20-dimensional $N(0, \Sigma)$ with highly irregular Σ.
- Both of these Metropolis-Hastings are optimally scaled to have acc rate ≈ 0.23



-
- However, the proposal increments are of the form

$$q_\theta = \sigma N(0, Id) \quad \text{and} \quad q_\theta = \sigma N(0, \Sigma)$$

Adaptive Metropolis algorithm

- Indeed, it turns out that the optimal covariance matrix choice is

$$q_\theta = \sigma N(0, \Sigma)$$

- And if $\pi = N(0, \Sigma)$, is a $d-$dimensional Gaussian, then [RR01]

$$q_\theta = N(0, \frac{(2.38)^2}{d}\Sigma)$$

- Moreover, if wrong covariance matrix is used, i.e.

$$q_\theta = \sigma N(0, \tilde{\Sigma})$$

then the slowdown of the algorithm is given by the following inhomogeneity factor [RR01]

$$b = d\frac{\sum_{j=1}^d \lambda_j}{(\sum_{j=1}^d \lambda_j^{1/2})^2}$$

where λ_j are eigenvalues of $\Sigma\tilde{\Sigma}^{-1}$.

Adaptive Metropolis algorithm

- Indeed, it turns out that the optimal covariance matrix choice is

$$q_\theta = \sigma N(0, \Sigma)$$

- And if $\pi = N(0, \Sigma)$, is a $d-$dimensional Gaussian, then [RR01]

$$q_\theta = N(0, \frac{(2.38)^2}{d}\Sigma)$$

- Moreover, if wrong covariance matrix is used, i.e.

$$q_\theta = \sigma N(0, \tilde{\Sigma})$$

then the slowdown of the algorithm is given by the following inhomogeneity factor [RR01]

$$b = d\frac{\sum_{j=1}^{d}\lambda_j}{(\sum_{j=1}^{d}\lambda_j^{1/2})^2}$$

where λ_j are eigenvalues of $\Sigma\tilde{\Sigma}^{-1}$.

Adaptive Metropolis algorithm

- Indeed, it turns out that the optimal covariance matrix choice is

$$q_\theta = \sigma N(0, \Sigma)$$

- And if $\pi = N(0, \Sigma)$, is a $d-$dimensional Gaussian, then [RR01]

$$q_\theta = N(0, \frac{(2.38)^2}{d}\Sigma)$$

- Moreover, if wrong covariance matrix is used, i.e.

$$q_\theta = \sigma N(0, \tilde{\Sigma})$$

then the slowdown of the algorithm is given by the following inhomogeneity factor [RR01]

$$b = d\frac{\sum_{j=1}^{d}\lambda_j}{(\sum_{j=1}^{d}\lambda_j^{1/2})^2}$$

where λ_j are eigenvalues of $\Sigma\tilde{\Sigma}^{-1}$.

Adaptive Metropolis algorithm

- This suggests **we should estimate Σ on the fly** and gives rise to the Adaptive Metropolis algorithm [HST01]
- Σ_n - the covariance matrix used at time n is updated by an **iterative formula**.
- The AM version of [HST01] (the original one) uses

$$N(0, \Sigma_n + \varepsilon Id)$$

- Modification due to [RR09] is to use

$$Q_n = (1-\beta)N(0, (2.38)^2\Sigma_n/d) + \beta N(0, \varepsilon Id/d).$$

- the above modification appears more tractable: containment has been verified for both, exponentially and super-exponentially decaying tails (Bai et al 2009).
- the original version has been analyzed in [SV10] and [FMP10] using different techniques.

Adaptive Metropolis algorithm

- This suggests **we should estimate Σ on the fly** and gives rise to the Adaptive Metropolis algorithm [HST01]
- Σ_n - the covariance matrix used at time n is updated by an **iterative formula.**
- The AM version of [HST01] (the original one) uses

$$N(0, \Sigma_n + \varepsilon Id)$$

- Modification due to [RR09] is to use

$$Q_n = (1-\beta)N(0,(2.38)^2\Sigma_n/d) + \beta N(0, \varepsilon Id/d).$$

- the above modification appears more tractable: containment has been verified for both, exponentially and super-exponentially decaying tails (Bai et al 2009).
- the original version has been analyzed in [SV10] and [FMP10] using different techniques.

Adaptive Metropolis algorithm

- This suggests **we should estimate Σ on the fly** and gives rise to the Adaptive Metropolis algorithm [HST01]
- Σ_n - the covariance matrix used at time n is updated by an **iterative formula.**
- The AM version of [HST01] (the original one) uses

$$N(0, \Sigma_n + \varepsilon Id)$$

- Modification due to [RR09] is to use

$$Q_n = (1-\beta)N(0, (2.38)^2\Sigma_n/d) + \beta N(0, \varepsilon Id/d).$$

- the above modification appears more tractable: containment has been verified for both, exponentially and super-exponentially decaying tails (Bai et al 2009).
- the original version has been analyzed in [SV10] and [FMP10] using different techniques.

Adaptive Metropolis algorithm

- This suggests **we should estimate** Σ **on the fly** and gives rise to the Adaptive Metropolis algorithm [HST01]
- Σ_n - the covariance matrix used at time n is updated by an **iterative formula.**
- The AM version of [HST01] (the original one) uses

$$N(0, \Sigma_n + \varepsilon Id)$$

- Modification due to [RR09] is to use

$$Q_n = (1-\beta)N(0, (2.38)^2\Sigma_n/d) + \beta N(0, \varepsilon Id/d).$$

- the above modification appears more tractable: containment has been verified for both, exponentially and super-exponentially decaying tails (Bai et al 2009).
- the original version has been analyzed in [SV10] and [FMP10] using different techniques.

Adaptive Metropolis algorithm

- This suggests **we should estimate Σ on the fly** and gives rise to the Adaptive Metropolis algorithm [HST01]
- Σ_n - the covariance matrix used at time n is updated by an **iterative formula.**
- The AM version of [HST01] (the original one) uses

$$N(0, \Sigma_n + \varepsilon Id)$$

- Modification due to [RR09] is to use

$$Q_n = (1-\beta)N(0, (2.38)^2\Sigma_n/d) + \beta N(0, \varepsilon Id/d).$$

- the above modification appears more tractable: containment has been verified for both, exponentially and super-exponentially decaying tails (Bai et al 2009).
- the original version has been analyzed in [SV10] and [FMP10] using different techniques.

Adaptive Metropolis algorithm

- This suggests **we should estimate** Σ **on the fly** and gives rise to the Adaptive Metropolis algorithm [HST01]
- Σ_n - the covariance matrix used at time n is updated by an **iterative formula.**
- The AM version of [HST01] (the original one) uses

$$N(0, \Sigma_n + \varepsilon Id)$$

- Modification due to [RR09] is to use

$$Q_n = (1-\beta)N(0, (2.38)^2\Sigma_n/d) + \beta N(0, \varepsilon Id/d).$$

- the above modification appears more tractable: containment has been verified for both, exponentially and super-exponentially decaying tails (Bai et al 2009).
- the original version has been analyzed in [SV10] and [FMP10] using different techniques.

parametric family of transition kernels P_θ

- typically we can design a family of ergodic transition kernels P_θ, $\theta \in \Theta$.
- Ex 1a. $\Theta = R_+$
  P_θ - Random Walk Metropolis with proposal increments
  $$q_\theta = \theta N(0, Id)$$
- Ex 1b. $\Theta = R_+ \times \{d \text{ dimensional covariance matrices}\}$
  P_θ - Random Walk Metropolis with proposal increments
  $$q_\theta = \sigma N(0, \Sigma)$$
- Ex 2. $\Theta = \Delta_{d-1} := \{(\alpha_1, \ldots, \alpha_d) \in \mathbb{R}^d : \alpha_i \geq 0, \sum_{i=1}^d \alpha_i = 1\}$ the $(d-1)-$dimensional probability simplex,
  P_θ - Random Scan Gibbs Sampler with coordinate selection probabilities
  $$\theta = (\alpha_1, \ldots, \alpha_n)$$
- In each case values of θ will affect efficiency of P_θ

parametric family of transition kernels P_θ

- typically we can design a family of ergodic transition kernels P_θ, $\theta \in \Theta$.
- Ex 1a. $\Theta = R_+$
  P_θ - Random Walk Metropolis with proposal increments
  $$q_\theta = \theta N(0, Id)$$
- Ex 1b. $\Theta = R_+ \times \{d \text{ dimensional covariance matrices}\}$
  P_θ - Random Walk Metropolis with proposal increments
  $$q_\theta = \sigma N(0, \Sigma)$$
- Ex 2. $\Theta = \Delta_{d-1} := \{(\alpha_1, \ldots, \alpha_d) \in \mathbb{R}^d : \alpha_i \geq 0, \ \sum_{i=1}^d \alpha_i = 1\}$ the $(d-1)-$dimensional probability simplex,
  P_θ - Random Scan Gibbs Sampler with coordinate selection probabilities
  $$\theta = (\alpha_1, \ldots, \alpha_n)$$
- In each case values of θ will affect efficiency of P_θ

parametric family of transition kernels P_θ

- typically we can design a family of ergodic transition kernels P_θ, $\theta \in \Theta$.
- Ex 1a. $\Theta = R_+$
  P_θ - Random Walk Metropolis with proposal increments
  $$q_\theta = \theta N(0, Id)$$
- Ex 1b. $\Theta = R_+ \times \{d \text{ dimensional covariance matrices}\}$
  P_θ - Random Walk Metropolis with proposal increments
  $$q_\theta = \sigma N(0, \Sigma)$$
- Ex 2. $\Theta = \Delta_{d-1} := \{(\alpha_1, \ldots, \alpha_d) \in \mathbb{R}^d : \alpha_i \geq 0, \ \sum_{i=1}^d \alpha_i = 1\}$ the $(d-1)-$dimensional probability simplex,
  P_θ - Random Scan Gibbs Sampler with coordinate selection probabilities
  $$\theta = (\alpha_1, \ldots, \alpha_n)$$
- In each case values of θ will affect efficiency of P_θ

parametric family of transition kernels P_θ

- typically we can design a family of ergodic transition kernels P_θ, $\theta \in \Theta$.
- Ex 1a. $\Theta = R_+$
  P_θ - Random Walk Metropolis with proposal increments
  $$q_\theta = \theta N(0, Id)$$
- Ex 1b. $\Theta = R_+ \times \{d \text{ dimensional covariance matrices}\}$
  P_θ - Random Walk Metropolis with proposal increments
  $$q_\theta = \sigma N(0, \Sigma)$$
- Ex 2. $\Theta = \Delta_{d-1} := \{(\alpha_1, \dots, \alpha_d) \in \mathbb{R}^d : \alpha_i \geq 0, \sum_{i=1}^d \alpha_i = 1\}$ the $(d-1)-$dimensional probability simplex,
  P_θ - Random Scan Gibbs Sampler with coordinate selection probabilities
  $$\theta = (\alpha_1, \dots, \alpha_n)$$
- In each case values of θ will affect efficiency of P_θ

parametric family of transition kernels P_θ

- typically we can design a family of ergodic transition kernels P_θ, $\theta \in \Theta$.
- Ex 1a. $\Theta = R_+$
  P_θ - Random Walk Metropolis with proposal increments
  $$q_\theta = \theta N(0, Id)$$
- Ex 1b. $\Theta = R_+ \times \{d \text{ dimensional covariance matrices}\}$
  P_θ - Random Walk Metropolis with proposal increments
  $$q_\theta = \sigma N(0, \Sigma)$$
- Ex 2. $\Theta = \Delta_{d-1} := \{(\alpha_1, \ldots, \alpha_d) \in \mathbb{R}^d : \alpha_i \geq 0, \ \sum_{i=1}^d \alpha_i = 1\}$ the $(d-1)-$dimensional probability simplex,
  P_θ - Random Scan Gibbs Sampler with coordinate selection probabilities
  $$\theta = (\alpha_1, \ldots, \alpha_n)$$
- In each case values of θ will affect efficiency of P_θ

The typical Adaptive MCMC setting

- In a typical Adaptive MCMC setting the parameter space Θ is large
- there is an optimal $\theta_* \in \Theta$ s.t. P_{θ_*} converges quickly.
- there are arbitrary bad values in Θ, say if $\theta \in \bar{\Theta} - \Theta$ then P_θ is not ergodic.
- if $\theta \in \Theta_* :=$ a region close to θ_*, then P_θ shall inherit good convergence properties of P_{θ_*}.
- When using adaptive MCMC we hope θ_n will eventually find the region Θ_* and stay there essentially forever. And that the adaptive algorithm $\mathcal{A}$ will inherit the good convergence properties of Θ_* in the limit.
-
- We are looking for a Theorem:
  You can actually run your Adaptive MCMC algorithm $\mathcal{A}$, and it will do what it is supposed to do! (under verifiable conditions)

The typical Adaptive MCMC setting

- In a typical Adaptive MCMC setting the parameter space Θ is large
- there is an optimal $\theta_* \in \Theta$ s.t. P_{θ_*} converges quickly.
- there are arbitrary bad values in Θ, say if $\theta \in \bar{\Theta} - \Theta$ then P_θ is not ergodic.
- if $\theta \in \Theta_* :=$ a region close to θ_*, then P_θ shall inherit good convergence properties of P_{θ_*}.
- When using adaptive MCMC we hope θ_n will eventually find the region Θ_* and stay there essentially forever. And that the adaptive algorithm $\mathcal{A}$ will inherit the good convergence properties of Θ_* in the limit.
-
- We are looking for a Theorem: *You can actually run your Adaptive MCMC algorithm $\mathcal{A}$, and it will do what it is supposed to do! (under verifiable conditions)*

The typical Adaptive MCMC setting

- In a typical Adaptive MCMC setting the parameter space Θ is large
- there is an optimal $\theta_* \in \Theta$ s.t. P_{θ_*} converges quickly.
- there are arbitrary bad values in Θ, say if $\theta \in \bar{\Theta} - \Theta$ then P_θ is not ergodic.
- if $\theta \in \Theta_* :=$ a region close to θ_*, then P_θ shall inherit good convergence properties of P_{θ_*}.
- When using adaptive MCMC we hope θ_n will eventually find the region Θ_* and stay there essentially forever. And that the adaptive algorithm $\mathcal{A}$ will inherit the good convergence properties of Θ_* in the limit.
-
- We are looking for a Theorem:
  You can actually run your Adaptive MCMC algorithm $\mathcal{A}$, and it will do what it is supposed to do! (under verifiable conditions)

The typical Adaptive MCMC setting

- In a typical Adaptive MCMC setting the parameter space Θ is large
- there is an optimal $\theta_* \in \Theta$ s.t. P_{θ_*} converges quickly.
- there are arbitrary bad values in Θ, say if $\theta \in \bar{\Theta} - \Theta$ then P_θ is not ergodic.
- if $\theta \in \Theta_* :=$ a region close to θ_*, then P_θ shall inherit good convergence properties of P_{θ_*}.
- When using adaptive MCMC we hope θ_n will eventually find the region Θ_* and stay there essentially forever. And that the adaptive algorithm $\mathcal{A}$ will inherit the good convergence properties of Θ_* in the limit.
-
- We are looking for a Theorem:
You can actually run your Adaptive MCMC algorithm $\mathcal{A}$, and it will do what it is supposed to do! (under verifiable conditions)

The typical Adaptive MCMC setting

- In a typical Adaptive MCMC setting the parameter space Θ is large
- there is an optimal $\theta_* \in \Theta$ s.t. P_{θ_*} converges quickly.
- there are arbitrary bad values in Θ, say if $\theta \in \bar{\Theta} - \Theta$ then P_θ is not ergodic.
- if $\theta \in \Theta_* :=$ a region close to θ_*, then P_θ shall inherit good convergence properties of P_{θ_*}.
- When using adaptive MCMC we hope θ_n will eventually find the region Θ_* and stay there essentially forever. And that the adaptive algorithm $\mathcal{A}$ will inherit the good convergence properties of Θ_* in the limit.
-
- We are looking for a Theorem:
  You can actually run your Adaptive MCMC algorithm $\mathcal{A}$, and it will do what it is supposed to do! (under verifiable conditions)

The typical Adaptive MCMC setting

- In a typical Adaptive MCMC setting the parameter space Θ is large
- there is an optimal $\theta_* \in \Theta$ s.t. P_{θ_*} converges quickly.
- there are arbitrary bad values in Θ, say if $\theta \in \bar{\Theta} - \Theta$ then P_θ is not ergodic.
- if $\theta \in \Theta_* :=$ a region close to θ_*, then P_θ shall inherit good convergence properties of P_{θ_*}.
- When using adaptive MCMC we hope θ_n will eventually find the region Θ_* and stay there essentially forever. And that the adaptive algorithm $\mathcal{A}$ will inherit the good convergence properties of Θ_* in the limit.
-
- We are looking for a Theorem:
  You can actually run your Adaptive MCMC algorithm $\mathcal{A}$, and it will do what it is supposed to do! (under verifiable conditions)

Adaptive Gibbs Sampler - a generic algorithm

- `AdapRSG`
  1. Set $p_n := R_n(p_{n-1}, X_{n-1}, \ldots, X_0) \in \mathcal{Y} \subset [0,1]^d$
  2. Choose coordinate $i \in \{1, \ldots, d\}$ according to selection probabilities p_n
  3. Draw $Y \sim \pi(\cdot|X_{n-1,-i})$
  4. Set $X_n := (X_{n-1,1}, \ldots, X_{n-1,i-1}, Y, X_{n-1,i+1}, \ldots, X_{n-1,d})$
- Given target distribution π, **what are the optimal selection probabilities** p?
- Similarly clean and operational criteria as in the Metropolis-Hastings case, are not available
- Little guidance in literature
- We need something that
  - has universal appeal,
  - is easy enough to compute and code,
  - works in practice

Adaptive Gibbs Sampler - a generic algorithm

- `AdapRSG`
  1. Set $p_n := R_n(p_{n-1}, X_{n-1}, \ldots, X_0) \in \mathcal{Y} \subset [0,1]^d$
  2. Choose coordinate $i \in \{1, \ldots, d\}$ according to selection probabilities p_n
  3. Draw $Y \sim \pi(\cdot|X_{n-1,-i})$
  4. Set $X_n := (X_{n-1,1}, \ldots, X_{n-1,i-1}, Y, X_{n-1,i+1}, \ldots, X_{n-1,d})$
- Given target distribution π, **what are the optimal selection probabilities** p?
- Similarly clean and operational criteria as in the Metropolis-Hastings case, are not available
- Little guidance in literature
- We need something that
  - has universal appeal,
  - is easy enough to compute and code,
  - works in practice

Adaptive Gibbs Sampler - a generic algorithm

- `AdapRSG`
  1. Set $p_n := R_n(p_{n-1}, X_{n-1}, \ldots, X_0) \in \mathcal{Y} \subset [0,1]^d$
  2. Choose coordinate $i \in \{1, \ldots, d\}$ according to selection probabilities p_n
  3. Draw $Y \sim \pi(\cdot|X_{n-1,-i})$
  4. Set $X_n := (X_{n-1,1}, \ldots, X_{n-1,i-1}, Y, X_{n-1,i+1}, \ldots, X_{n-1,d})$
- Given target distribution π, **what are the optimal selection probabilities** p?
- Similarly clean and operational criteria as in the Metropolis-Hastings case, are not available
- Little guidance in literature
- We need something that
  - has universal appeal,
  - is easy enough to compute and code,
  - works in practice

Adaptive Gibbs Sampler - a generic algorithm

- `AdapRSG`
  1. Set $p_n := R_n(p_{n-1}, X_{n-1}, \ldots, X_0) \in \mathcal{Y} \subset [0,1]^d$
  2. Choose coordinate $i \in \{1, \ldots, d\}$ according to selection probabilities p_n
  3. Draw $Y \sim \pi(\cdot | X_{n-1,-i})$
  4. Set $X_n := (X_{n-1,1}, \ldots, X_{n-1,i-1}, Y, X_{n-1,i+1}, \ldots, X_{n-1,d})$
- Given target distribution π, **what are the optimal selection probabilities p?**
- Similarly clean and operational criteria as in the Metropolis-Hastings case, are not available
- Little guidance in literature
- We need something that
  - has universal appeal,
  - is easy enough to compute and code,
  - works in practice

Adaptive Gibbs Sampler - a generic algorithm

- `AdapRSG`
  1. Set $p_n := R_n(p_{n-1}, X_{n-1}, \ldots, X_0) \in \mathcal{Y} \subset [0,1]^d$
  2. Choose coordinate $i \in \{1, \ldots, d\}$ according to selection probabilities p_n
  3. Draw $Y \sim \pi(\cdot | X_{n-1,-i})$
  4. Set $X_n := (X_{n-1,1}, \ldots, X_{n-1,i-1}, Y, X_{n-1,i+1}, \ldots, X_{n-1,d})$
- Given target distribution π, **what are the optimal selection probabilities p?**
- Similarly clean and operational criteria as in the Metropolis-Hastings case, are not available
- Little guidance in literature
- We need something that
  - has universal appeal,
  - is easy enough to compute and code,
  - works in practice

Adaptive Random Scan Metropolis within Gibbs

`AdapRSMwG`

1. Set $p_n := R_n(p_{n-1}, X_{n-1}, \ldots, X_0) \in \mathcal{Y}$
2. Choose coordinate $i \in \{1, \ldots, d\}$ according to selection probabilities p_n
3. Draw $Y \sim Q_{X_{n-1,-i}}(X_{n-1,i}, \cdot)$
4. With probability

$$\min\left(1, \quad \frac{\pi(Y|X_{n-1,-i})\; q_{X_{n-1,-i}}(Y, X_{n-1,i})}{\pi(X_{n-1}|X_{n-1,-i})\; q_{X_{n-1,-i}}(X_{n-1,i}, Y)}\right), \tag{1}$$

   accept the proposal and set

$$X_n = (X_{n-1,1}, \ldots, X_{n-1,i-1}, Y, X_{n-1,i+1}, \ldots, X_{n-1,d})\,;$$

   otherwise, reject the proposal and set $X_n = X_{n-1}$.

Adaptive RS adaptive Metropolis within Gibbs

AdapRSadapMwG

1. Set $p_n := R_n(p_{n-1}, X_{n-1}, \ldots, X_0, \gamma_{n-1}, \ldots, \gamma_0) \in \mathcal{Y}$
2. Set $\gamma_n := R'_n(\alpha_{n-1}, X_{n-1}, \ldots, X_0, \gamma_{n-1}, \ldots, \gamma_0) \in \Gamma_1 \times \ldots \times \Gamma_n$
3. Choose coordinate $i \in \{1, \ldots, d\}$ according to selection probabilities α, i.e. with $\Pr(i = j) = p_j$
4. Draw $Y \sim Q_{X_{n-1,-i}, \gamma_{n-1}}(X_{n-1,i}, \cdot)$
5. With probability (2),

$$\min\left(1, \ \frac{\pi(Y|X_{n-1,-i})\, q_{X_{n-1,-i},\gamma_{n-1}}(Y, X_{n-1,i})}{\pi(X_{n-1}|X_{n-1,-i})\, q_{X_{n-1,-i},\gamma_{n-1}}(X_{n-1,i}, Y)}\right),$$

accept the proposal and set

$$X_n = (X_{n-1,1}, \ldots, X_{n-1,i-1}, Y, X_{n-1,i+1}, \ldots, X_{n-1,d});$$

otherwise, reject the proposal and set $X_n = X_{n-1}$.

Adapting the Gibbs Sampler: IF... IF...[CLR18a]

- If π was Gaussian...
- If we knew the covariance matrix Σ of π
- Then for $RSGS(p)$ and the target

$$\pi = N(\mu, \Sigma),$$

- we could compute the Spectral Gap (L_2-convergence rate) of RSGS(p) (building on Amit 1991, 1996 and Roberts and Sahu 1997)
- $$G(p) = \frac{1}{\lambda_{max}\Big(M(\Sigma, p)\Big)},$$

  where $M(\cdot,\cdot)$ is a known $d \times d$ matrix-valued function.
- So one could take

$$p^{opt} = \operatorname{argmax}_p G(p) \;=\; \operatorname*{argmin}_{p \in \Delta_{d-1}} \lambda_{max}\Big(M(\Sigma, p)\Big),$$

Adapting the Gibbs Sampler: IF... IF...[CLR18a]

- If π was Gaussian...
- If we knew the covariance matrix Σ of π
- Then for $RSGS(p)$ and the target

$$\pi = N(\mu, \Sigma),$$

- we could compute the Spectral Gap (L_2-convergence rate) of RSGS(p) (building on Amit 1991, 1996 and Roberts and Sahu 1997)
- $$G(p) = \frac{1}{\lambda_{max}\Big(M(\Sigma, p)\Big)},$$

  where $M(\cdot, \cdot)$ is a known $d \times d$ matrix-valued function.
- So one could take

$$p^{opt} = \mathrm{argmax}_p G(p) \;=\; \underset{p \in \Delta_{d-1}}{\mathrm{argmin}}\, \lambda_{max}\Big(M(\Sigma, p)\Big),$$

Adapting the Gibbs Sampler: IF... IF...[CLR18a]

- If π was Gaussian...
- If we knew the covariance matrix Σ of π
- Then for $RSGS(p)$ and the target

$$\pi = N(\mu, \Sigma),$$

- we could compute the Spectral Gap (L_2-convergence rate) of RSGS(p) (building on Amit 1991, 1996 and Roberts and Sahu 1997)
- $$G(p) = \frac{1}{\lambda_{max}\Big(M(\Sigma, p)\Big)},$$

  where $M(\cdot, \cdot)$ is a known $d \times d$ matrix-valued function.
- So one could take

$$p^{opt} = \text{argmax}_p G(p) \;=\; \underset{p \in \Delta_{d-1}}{\text{argmin}}\, \lambda_{max}\Big(M(\Sigma, p)\Big),$$

Adapting the Gibbs Sampler: IF... IF...[CLR18a]

- If π was Gaussian...
- If we knew the covariance matrix Σ of π
- Then for $RSGS(p)$ and the target

$$\pi = N(\mu, \Sigma),$$

- we could compute the Spectral Gap (L_2-convergence rate) of RSGS(p) (building on Amit 1991, 1996 and Roberts and Sahu 1997)
-

$$G(p) = \frac{1}{\lambda_{max}\Big(M(\Sigma, p)\Big)},$$

where $M(\cdot, \cdot)$ is a known $d \times d$ matrix-valued function.
- So one could take

$$p^{opt} = \text{argmax}_p G(p) \;=\; \underset{p \in \Delta_{d-1}}{\text{argmin}}\, \lambda_{max}\Big(M(\Sigma, p)\Big),$$

Adapting the Gibbs Sampler: IF... IF...[CLR18a]

- If π was Gaussian...
- If we knew the covariance matrix Σ of π
- Then for $RSGS(p)$ and the target

$$\pi = N(\mu, \Sigma),$$

- we could compute the Spectral Gap (L_2-convergence rate) of RSGS(p) (building on Amit 1991, 1996 and Roberts and Sahu 1997)
- $$G(p) = \frac{1}{\lambda_{max}\Big(M(\Sigma, p)\Big)},$$

  where $M(\cdot,\cdot)$ is a known $d \times d$ matrix-valued function.
- So one could take

$$p^{opt} = \text{argmax}_p G(p) \; = \; \underset{p \in \Delta_{d-1}}{\text{argmin}} \; \lambda_{max}\Big(M(\Sigma, p)\Big),$$

Adapting the Gibbs Sampler: IF... IF...[CLR18a]

- If π was Gaussian...
- If we knew the covariance matrix Σ of π
- Then for $RSGS(p)$ and the target

$$\pi = N(\mu, \Sigma),$$

- we could compute the Spectral Gap (L_2-convergence rate) of RSGS(p) (building on Amit 1991, 1996 and Roberts and Sahu 1997)
- $$G(p) = \frac{1}{\lambda_{max}\Big(M(\Sigma, p)\Big)},$$

  where $M(\cdot,\cdot)$ is a known $d \times d$ matrix-valued function.
- So one could take

$$p^{opt} = \mathrm{argmax}_p G(p) \; = \; \underset{p \in \Delta_{d-1}}{\mathrm{argmin}}\, \lambda_{max}\Big(M(\Sigma, p)\Big),$$

Adapting the Gibbs Sampler: Complications...

- $$p^{opt} = \text{argmax}_p G(p) \;=\; \underset{p \in \Delta_{d-1}}{\text{argmin}}\, \lambda_{max}\Bigg(M(\Sigma, p)\Bigg),$$
- Issue 1: π is not Gaussian
- Issue 2: Σ and hence $M(\Sigma, p)$ are not known.
- Issue 3: λ_{max} is expensive to compute.

Adapting the Gibbs Sampler: Complications...

- $$p^{opt} = \text{argmax}_p G(p) \;=\; \underset{p \in \Delta_{d-1}}{\text{argmin}}\, \lambda_{max}\Big(M(\Sigma, p)\Big),$$
- Issue 1: π is not Gaussian
- Issue 2: Σ and hence $M(\Sigma, p)$ are not known.
- Issue 3: λ_{max} is expensive to compute.

Adapting the Gibbs Sampler: Complications...

- $$p^{opt} = \mathrm{argmax}_p G(p) \;=\; \operatorname*{argmin}_{p \in \Delta_{d-1}} \lambda_{max}\Big(M(\Sigma, p)\Big),$$
- Issue 1: π is not Gaussian
- Issue 2: Σ and hence $M(\Sigma, p)$ are not known.
- Issue 3: λ_{max} is expensive to compute.

Adapting the Gibbs Sampler: Complications...

- $$p^{opt} = \text{argmax}_p G(p) \;=\; \underset{p \in \Delta_{d-1}}{\text{argmin}}\, \lambda_{max}\Big(M(\Sigma, p)\Big),$$
- Issue 1: π is not Gaussian
- Issue 2: Σ and hence $M(\Sigma, p)$ are not known.
- Issue 3: λ_{max} is expensive to compute.

Some properties of $G(p)$

- $$G(p) = \frac{1}{\lambda_{max}\Big(M(\Sigma, p)\Big)}.$$
- G is concave and a.s. differentiable w.r.t. Lebesgue measure on Δ_{d-1}.
- Gradient of G at p:
$$\nabla G(p) = F(\Sigma, p, x),$$
where F is a known $d-1$ dimensional vector-valued function and x is in the eigenspace of the maximal eigenvalue, i.e.
$$M(\Sigma, p)x = \frac{1}{G(p)}x, \quad \|x\| = 1$$

Some properties of $G(p)$

- $$G(p) = \frac{1}{\lambda_{max}\Big(M(\Sigma, p)\Big)}.$$
- G is concave and a.s. differentiable w.r.t. Lebesgue measure on Δ_{d-1}.
- Gradient of G at p:
$$\nabla G(p) = F(\Sigma, p, x),$$
where F is a known $d-1$ dimensional vector-valued function and x is in the eigenspace of the maximal eigenvalue, i.e.
$$M(\Sigma, p)x = \frac{1}{G(p)}x, \ \ \|x\| = 1$$

Guidance from the Gaussian case

- We can use the guidance from the Gaussian case to optimise general posteriors
- Many posterior distributions in Bayesian inference will be close to Gaussians by the Bernstein-von Mieses Theorem
- We can estimate Σ_n on the fly.
- Solving
$$\mathrm{argmax}_p \Big(G(p) \Big)$$
is expensive and we can not afford a full solution after every update of Σ_n.
- In [CLR18a] a version of sub-gradient stochastic optimisation algorithm for convex functions is developed that progresses gradually stochastic optimisation as Σ_n stabilises.
- The sub-gradient computation relies on a single step of the power method with a noisy matrix estimate.
- The adaptation step is realised after a fixed number of iterations have been obtained that contribute significantly to the covariance matrix estimate.

Guidance from the Gaussian case

- We can use the guidance from the Gaussian case to optimise general posteriors
- Many posterior distributions in Bayesian inference will be close to Gaussians by the Bernstein-von Mieses Theorem
- We can estimate Σ_n on the fly.
- Solving

$$\text{argmax}_p \Big(G(p) \Big)$$

  is expensive and we can not afford a full solution after every update of Σ_n.
- In [CLR18a] a version of sub-gradient stochastic optimisation algorithm for convex functions is developed that progresses gradually stochastic optimisation as Σ_n stabilises.
- The sub-gradient computation relies on a single step of the power method with a noisy matrix estimate.
- The adaptation step is realised after a fixed number of iterations have been obtained that contribute significantly to the covariance matrix estimate.

Guidance from the Gaussian case

- We can use the guidance from the Gaussian case to optimise general posteriors
- Many posterior distributions in Bayesian inference will be close to Gaussians by the Bernstein-von Mieses Theorem
- We can estimate Σ_n on the fly.
- Solving
$$\mathrm{argmax}_p \Big(G(p) \Big)$$
is expensive and we can not afford a full solution after every update of Σ_n.
- In [CLR18a] a version of sub-gradient stochastic optimisation algorithm for convex functions is developed that progresses gradually stochastic optimisation as Σ_n stabilises.
- The sub-gradient computation relies on a single step of the power method with a noisy matrix estimate.
- The adaptation step is realised after a fixed number of iterations have been obtained that contribute significantly to the covariance matrix estimate.

Guidance from the Gaussian case

- We can use the guidance from the Gaussian case to optimise general posteriors
- Many posterior distributions in Bayesian inference will be close to Gaussians by the Bernstein-von Mieses Theorem
- We can estimate Σ_n on the fly.
- Solving
$$\operatorname{argmax}_p \Big(G(p) \Big)$$
is expensive and we can not afford a full solution after every update of Σ_n.
- In [CLR18a] a version of sub-gradient stochastic optimisation algorithm for convex functions is developed that progresses gradually stochastic optimisation as Σ_n stabilises.
- The sub-gradient computation relies on a single step of the power method with a noisy matrix estimate.
- The adaptation step is realised after a fixed number of iterations have been obtained that contribute significantly to the covariance matrix estimate.

Guidance from the Gaussian case

- We can use the guidance from the Gaussian case to optimise general posteriors
- Many posterior distributions in Bayesian inference will be close to Gaussians by the Bernstein-von Mieses Theorem
- We can estimate Σ_n on the fly.
- Solving

  $$\mathrm{argmax}_p \Big(G(p) \Big)$$

  is expensive and we can not afford a full solution after every update of Σ_n.
- In [CLR18a] a version of sub-gradient stochastic optimisation algorithm for convex functions is developed that progresses gradually stochastic optimisation as Σ_n stabilises.
- The sub-gradient computation relies on a single step of the power method with a noisy matrix estimate.
- The adaptation step is realised after a fixed number of iterations have been obtained that contribute significantly to the covariance matrix estimate.

Guidance from the Gaussian case

- We can use the guidance from the Gaussian case to optimise general posteriors
- Many posterior distributions in Bayesian inference will be close to Gaussians by the Bernstein-von Mieses Theorem
- We can estimate Σ_n on the fly.
- Solving
$$\operatorname{argmax}_p \Big(G(p) \Big)$$
is expensive and we can not afford a full solution after every update of Σ_n.
- In [CLR18a] a version of sub-gradient stochastic optimisation algorithm for convex functions is developed that progresses gradually stochastic optimisation as Σ_n stabilises.
- The sub-gradient computation relies on a single step of the power method with a noisy matrix estimate.
- The adaptation step is realised after a fixed number of iterations have been obtained that contribute significantly to the covariance matrix estimate.

Guidance from the Gaussian case

- We can use the guidance from the Gaussian case to optimise general posteriors
- Many posterior distributions in Bayesian inference will be close to Gaussians by the Bernstein-von Mieses Theorem
- We can estimate Σ_n on the fly.
- Solving
$$\mathrm{argmax}_p \Big(G(p) \Big)$$
is expensive and we can not afford a full solution after every update of Σ_n.
- In [CLR18a] a version of sub-gradient stochastic optimisation algorithm for convex functions is developed that progresses gradually stochastic optimisation as Σ_n stabilises.
- The sub-gradient computation relies on a single step of the power method with a noisy matrix estimate.
- The adaptation step is realised after a fixed number of iterations have been obtained that contribute significantly to the covariance matrix estimate.

Toy Example 1 - a difficult pair

- $$\mathrm{Corr} = \begin{pmatrix} 1 & -\rho_1 & 0 & 0 & \cdots & 0 \\ -\rho_1 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & -\rho_2 & \cdots & 0 \\ 0 & 0 & -\rho_2 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & \cdots & 0 & 1 & -\rho_k \\ 0 & \cdots & \cdots & 0 & -\rho_k & 1 \end{pmatrix}$$
- Speedup of up to $k = d/2$ times.

Toy Example 1 - a difficult pair

- $$\mathrm{Corr} = \begin{pmatrix} 1 & -\rho_1 & 0 & 0 & \cdots & 0 \\ -\rho_1 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & -\rho_2 & \cdots & 0 \\ 0 & 0 & -\rho_2 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & \cdots & 0 & 1 & -\rho_k \\ 0 & \cdots & \cdots & 0 & -\rho_k & 1 \end{pmatrix}$$
- Speedup of up to $k = d/2$ times.

Toy Example 2 - a star-like correlation structure

- $$\Sigma = \begin{pmatrix} 1 & c & c & c & \cdots & c \\ c & 1 & 0 & 0 & \cdots & 0 \\ c & 0 & 1 & 0 & \cdots & 0 \\ c & 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ c & \cdots & \cdots & 0 & 1 & 0 \\ c & \cdots & \cdots & 0 & 0 & 1 \end{pmatrix}.$$
- Speedup of up to $d/2$ times.
- Sampling from Graphical Models

Toy Example 2 - a star-like correlation structure

- $$\Sigma = \begin{pmatrix} 1 & c & c & c & \cdots & c \\ c & 1 & 0 & 0 & \cdots & 0 \\ c & 0 & 1 & 0 & \cdots & 0 \\ c & 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ c & \cdots & \cdots & 0 & 1 & 0 \\ c & \cdots & \cdots & 0 & 0 & 1 \end{pmatrix}.$$
- Speedup of up to $d/2$ times.
- Sampling from Graphical Models

Toy Example 2 - a star-like correlation structure

- $$\Sigma = \begin{pmatrix} 1 & c & c & c & \cdots & c \\ c & 1 & 0 & 0 & \cdots & 0 \\ c & 0 & 1 & 0 & \cdots & 0 \\ c & 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ c & \cdots & \cdots & 0 & 1 & 0 \\ c & \cdots & \cdots & 0 & 0 & 1 \end{pmatrix}.$$
- Speedup of up to $d/2$ times.
- Sampling from Graphical Models

Simulations

- Consider coordinate-wise RSGS in $d-$dimensions. Denote

$$h_i = \frac{x_i}{\sqrt{Var_\pi(x_i)}}$$

  to be normalized linear functions depending on one coordinate only.
- We will focus on the worst performing coordinate in the sense of CLT asymptotic variance

$$\max_i \sigma^2_{as}(h_i)$$

Simulations

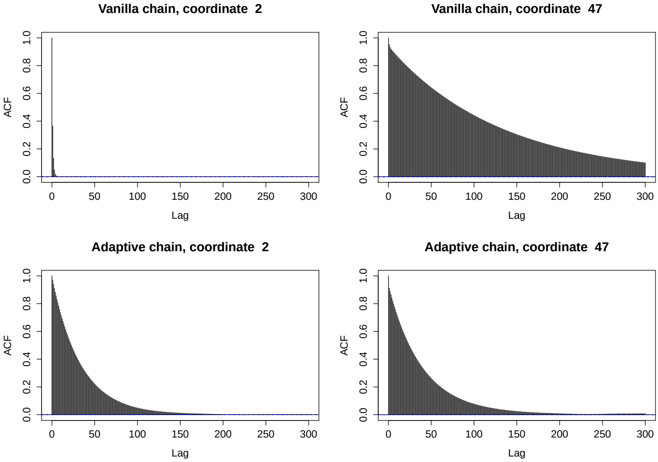
- Consider coordinate-wise RSGS in $d-$dimensions. Denote

$$h_i = \frac{x_i}{\sqrt{Var_\pi(x_i)}}$$

  to be normalized linear functions depending on one coordinate only.
- We will focus on the worst performing coordinate in the sense of CLT asymptotic variance

$$\max_i \sigma^2_{as}(h_i)$$

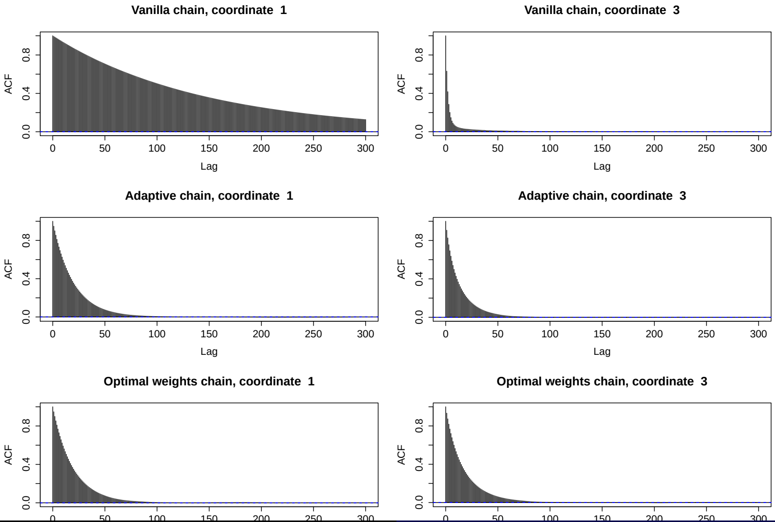
Truncated Multivariate Normals, d=50



Truncated Multivariate Normals, d=50

	$1/G(p)$	$\max_i \sigma^2_{as}(h_i)$
vanilla	6384	248
adaptive	1850	72
$\frac{\text{vanilla}}{\text{adaptive}}$	3.45	3.44

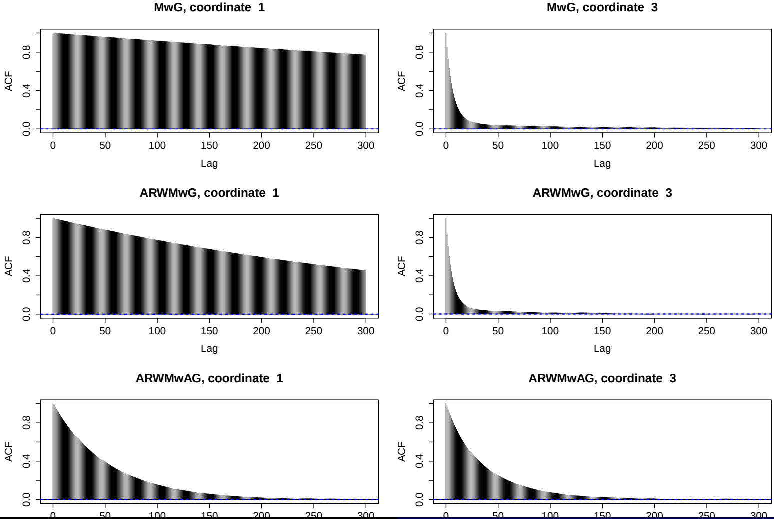
Poisson Hierarchical Model, d=50, Gibbs Sampler



Poisson Hierarchical Model, d=50, Gibbs Sampler

	$1/G(p)$	$\max_i \sigma^2_{as}(h_i)$
vanilla	13435	482
adaptive	1355	52
$\frac{\text{vanilla}}{\text{adaptive}}$	9.9	9.27

Poisson Hierarchical, d=50, Metropolis within Gibbs



Poisson Hierarchical, d=50, Metropolis within Gibbs

	$1/G(p)$	$\max_i \sigma^2_{as}(h_i)$
RWMwG (vanilla)	13244	1993
ARWMwG (partially adaptive)	13244	971
ARWMwAG (adaptive)	1376	138
$\frac{\text{partially adaptive}}{\text{adaptive}}$	9.63	7
$\frac{\text{vanilla}}{\text{adaptive}}$	9.63	14.45

Computational cost for the Poisson Hierarchical Model

	$\max_i \sigma^2_{as}(h_i)$	Cost per 5000 iterations	Cost of adaptation
ARSGS	52	0.37	0.0025
ARWMwAG	138	0.028	0.0025

Summary of Adaptive Gibbs

- The Adaptive Gibbs Sampler uses a principled optimisation strategy based on the Gaussian model and the Spectral Gap to guide adaptation
- ARSGS and ARWMwAG are useful even if the target is not normal or even not continuous
- [CLR18a] provides full implementations of the algorithms that can be readily used in applications
- Adaptation can be done in parallel with the sampling (and it is only a fraction of the sampling cost anyway)
- Adaptive Gibbs Samplers are provably ergodic under weak regularity conditions (some theory in a moment!)

Summary of Adaptive Gibbs

- The Adaptive Gibbs Sampler uses a principled optimisation strategy based on the Gaussian model and the Spectral Gap to guide adaptation
- ARSGS and ARWMwAG are useful even if the target is not normal or even not continuous
- [CLR18a] provides full implementations of the algorithms that can be readily used in applications
- Adaptation can be done in parallel with the sampling (and it is only a fraction of the sampling cost anyway)
- Adaptive Gibbs Samplers are provably ergodic under weak regularity conditions (some theory in a moment!)

Summary of Adaptive Gibbs

- The Adaptive Gibbs Sampler uses a principled optimisation strategy based on the Gaussian model and the Spectral Gap to guide adaptation
- ARSGS and ARWMwAG are useful even if the target is not normal or even not continuous
- [CLR18a] provides full implementations of the algorithms that can be readily used in applications
- Adaptation can be done in parallel with the sampling (and it is only a fraction of the sampling cost anyway)
- Adaptive Gibbs Samplers are provably ergodic under weak regularity conditions (some theory in a moment!)

Summary of Adaptive Gibbs

- The Adaptive Gibbs Sampler uses a principled optimisation strategy based on the Gaussian model and the Spectral Gap to guide adaptation
- ARSGS and ARWMwAG are useful even if the target is not normal or even not continuous
- [CLR18a] provides full implementations of the algorithms that can be readily used in applications
- Adaptation can be done in parallel with the sampling (and it is only a fraction of the sampling cost anyway)
- Adaptive Gibbs Samplers are provably ergodic under weak regularity conditions (some theory in a moment!)

Summary of Adaptive Gibbs

- The Adaptive Gibbs Sampler uses a principled optimisation strategy based on the Gaussian model and the Spectral Gap to guide adaptation
- ARSGS and ARWMwAG are useful even if the target is not normal or even not continuous
- [CLR18a] provides full implementations of the algorithms that can be readily used in applications
- Adaptation can be done in parallel with the sampling (and it is only a fraction of the sampling cost anyway)
- Adaptive Gibbs Samplers are provably ergodic under weak regularity conditions (some theory in a moment!)

Variable selection setting

- $$y = \alpha \mathbf{1} + X_\gamma \beta_\gamma + \epsilon, \qquad \epsilon \sim N(0, \sigma^2 I_n)$$
  where y is an $(n \times 1)$-dimensional vector of responses, $X = (x_1, \ldots, x_p)$ is an $(n \times p)$-dimensional data matrix and $\gamma = (\gamma_1, \ldots, \gamma_p) \in \Gamma = \{0, 1\}^p$ is a vector of indicator variables in which γ_i denotes whether the i-th variable is included in the model (when $\gamma_i = 1$).
- Bayesian variable selection involves placing a prior on the parameters of the regression model above, $(\alpha, \beta_\gamma, \sigma^2)$, as well as on the model γ.
- Sampling from the posterior model space is often difficult (exponential growth)
- Has been addressed via adaptive MCMC in a number of papers [NK05, JS13, CGL11].
- Briefly talk about [GLS17]

Variable selection setting

- $$y = \alpha \mathbf{1} + X_\gamma \beta_\gamma + \epsilon, \qquad \epsilon \sim N(0, \sigma^2 I_n)$$
  where y is an $(n \times 1)$-dimensional vector of responses, $X = (x_1, \ldots, x_p)$ is an $(n \times p)$-dimensional data matrix and $\gamma = (\gamma_1, \ldots, \gamma_p) \in \Gamma = \{0, 1\}^p$ is a vector of indicator variables in which γ_i denotes whether the i-th variable is included in the model (when $\gamma_i = 1$).
- Bayesian variable selection involves placing a prior on the parameters of the regression model above, $(\alpha, \beta_\gamma, \sigma^2)$, as well as on the model γ.
- Sampling from the posterior model space is often difficult (exponential growth)
- Has been addressed via adaptive MCMC in a number of papers [NK05, JS13, CGL11].
- Briefly talk about [GLS17]

Variable selection setting

- $$y = \alpha \mathbf{1} + X_\gamma \beta_\gamma + \epsilon, \qquad \epsilon \sim N(0, \sigma^2 I_n)$$

  where y is an $(n \times 1)$-dimensional vector of responses, $X = (x_1, \dots, x_p)$ is an $(n \times p)$-dimensional data matrix and $\gamma = (\gamma_1, \dots, \gamma_p) \in \Gamma = \{0, 1\}^p$ is a vector of indicator variables in which γ_i denotes whether the i-th variable is included in the model (when $\gamma_i = 1$).
- Bayesian variable selection involves placing a prior on the parameters of the regression model above, $(\alpha, \beta_\gamma, \sigma^2)$, as well as on the model γ.
- Sampling from the posterior model space is often difficult (exponential growth)
- Has been addressed via adaptive MCMC in a number of papers [NK05, JS13, CGL11].
- Briefly talk about [GLS17]

Variable selection setting

- $$y = \alpha \mathbf{1} + X_\gamma \beta_\gamma + \epsilon, \qquad \epsilon \sim N(0, \sigma^2 I_n)$$
  where y is an $(n \times 1)$-dimensional vector of responses, $X = (x_1, \ldots, x_p)$ is an $(n \times p)$-dimensional data matrix and $\gamma = (\gamma_1, \ldots, \gamma_p) \in \Gamma = \{0, 1\}^p$ is a vector of indicator variables in which γ_i denotes whether the i-th variable is included in the model (when $\gamma_i = 1$).
- Bayesian variable selection involves placing a prior on the parameters of the regression model above, $(\alpha, \beta_\gamma, \sigma^2)$, as well as on the model γ.
- Sampling from the posterior model space is often difficult (exponential growth)
- Has been addressed via adaptive MCMC in a number of papers [NK05, JS13, CGL11].
- Briefly talk about [GLS17]

Variable selection setting

- $$y = \alpha\mathbf{1} + X_\gamma\beta_\gamma + \epsilon, \qquad \epsilon \sim N(0, \sigma^2 I_n)$$
  where y is an $(n \times 1)$-dimensional vector of responses, $X = (x_1, \ldots, x_p)$ is an $(n \times p)$-dimensional data matrix and $\gamma = (\gamma_1, \ldots, \gamma_p) \in \Gamma = \{0, 1\}^p$ is a vector of indicator variables in which γ_i denotes whether the i-th variable is included in the model (when $\gamma_i = 1$).
- Bayesian variable selection involves placing a prior on the parameters of the regression model above, $(\alpha, \beta_\gamma, \sigma^2)$, as well as on the model γ.
- Sampling from the posterior model space is often difficult (exponential growth)
- Has been addressed via adaptive MCMC in a number of papers [NK05, JS13, CGL11].
- Briefly talk about [GLS17]

The individual adaptation algorithm [GLS17]

- The probability of proposing to move from model γ to γ' is given in a product form

$$q_\eta(\gamma, \gamma') = \prod_{j=1}^{p} q_{\eta,j}(\gamma_j, \gamma'_j)$$

where $\eta = (A, D) = (A_1, \ldots, A_p, D_1, \ldots, D_p)$,
$q_{\eta,j}(\gamma_j = 0, \gamma'_j = 1) = A_j$ and
$q_{\eta,j}(\gamma_j = 1, \gamma'_j = 0) = D_j$.

- The parameters are optimised to approximate iid sampling of variables for which data is not informative.
- How much improvement can we get by addressing the simple part of the posteriors?

The individual adaptation algorithm [GLS17]

- The probability of proposing to move from model γ to γ' is given in a product form
$$q_\eta(\gamma,\gamma') = \prod_{j=1}^{p} q_{\eta,j}(\gamma_j,\gamma'_j)$$
where $\eta = (A, D) = (A_1, \ldots, A_p, D_1, \ldots, D_p)$,
$q_{\eta,j}(\gamma_j = 0, \gamma'_j = 1) = A_j$ and
$q_{\eta,j}(\gamma_j = 1, \gamma'_j = 0) = D_j$.
- The parameters are optimised to approximate iid sampling of variables for which data is not informative.
- How much improvement can we get by addressing the simple part of the posteriors?

The individual adaptation algorithm [GLS17]

- The probability of proposing to move from model γ to γ' is given in a product form

$$q_\eta(\gamma, \gamma') = \prod_{j=1}^{p} q_{\eta,j}(\gamma_j, \gamma'_j)$$

  where $\eta = (A, D) = (A_1, \ldots, A_p, D_1, \ldots, D_p)$,
  $q_{\eta,j}(\gamma_j = 0, \gamma'_j = 1) = A_j$ and
  $q_{\eta,j}(\gamma_j = 1, \gamma'_j = 0) = D_j$.
- The parameters are optimised to approximate iid sampling of variables for which data is not informative.
- How much improvement can we get by addressing the simple part of the posteriors?

Synthetic data example

- Consider the synthetic data example analysed in [YWJ16]
- The speedup over the vanilla sampler of [YWJ16] is as follows

		5 chains				25 chains			
		SNR				SNR			
(n,p)		0.5	1	2	3	0.5	1	2	3
(500, 500)	IA	4.9	1.8	5.5	5.1	1.2	1.5	2.4	2.3
	ASI	1.7	21.3	31.8	7.5	2.0	36.0	42.7	12.6
(500, 5000)	IA	8.7	2.2	718.0	81.5	7.1	2.9	2267.2	147.2
	ASI	29.9	126.9	2053.1	2271.3	53.5	353.3	12319.5	7612.3
(1000, 500)	IA	5.9	16.3	7.7	4.2	1.6	80.7	4.4	1.8
	ASI	41.9	2.1	16.9	12.0	32.8	34.0	27.9	14.4
(1000, 5000)	IA	2.2	2.2	9167.2	11.3	5.6	2.5	15960.7	199.8
	ASI	15.4	37.0	4423.1	30.8	54.9	53.4	11558.2	736.4

Synthetic data example

- Consider the synthetic data example analysed in [YWJ16]
- The speedup over the vanilla sampler of [YWJ16] is as follows

		5 chains				25 chains			
		SNR				SNR			
(n, p)		0.5	1	2	3	0.5	1	2	3
(500, 500)	IA	4.9	1.8	5.5	5.1	1.2	1.5	2.4	2.3
	ASI	1.7	21.3	31.8	7.5	2.0	36.0	42.7	12.6
(500, 5000)	IA	8.7	2.2	718.0	81.5	7.1	2.9	2267.2	147.2
	ASI	29.9	126.9	2053.1	2271.3	53.5	353.3	12319.5	7612.3
(1000, 500)	IA	5.9	16.3	7.7	4.2	1.6	80.7	4.4	1.8
	ASI	41.9	2.1	16.9	12.0	32.8	34.0	27.9	14.4
(1000, 5000)	IA	2.2	2.2	9167.2	11.3	5.6	2.5	15960.7	199.8
	ASI	15.4	37.0	4423.1	30.8	54.9	53.4	11558.2	736.4

Synthetic data example

- Consider the synthetic data example analysed in [YWJ16]
- The speedup over the vanilla sampler of [YWJ16] is as follows

		5 chains				25 chains			
		SNR				SNR			
(n, p)		0.5	1	2	3	0.5	1	2	3
(500, 500)	IA	4.9	1.8	5.5	5.1	1.2	1.5	2.4	2.3
	ASI	1.7	21.3	31.8	7.5	2.0	36.0	42.7	12.6
(500, 5000)	IA	8.7	2.2	718.0	81.5	7.1	2.9	2267.2	147.2
	ASI	29.9	126.9	2053.1	2271.3	53.5	353.3	12319.5	7612.3
(1000, 500)	IA	5.9	16.3	7.7	4.2	1.6	80.7	4.4	1.8
	ASI	41.9	2.1	16.9	12.0	32.8	34.0	27.9	14.4
(1000, 5000)	IA	2.2	2.2	9167.2	11.3	5.6	2.5	15960.7	199.8
	ASI	15.4	37.0	4423.1	30.8	54.9	53.4	11558.2	736.4

-

a fundamental problem

- adaptive MCMC algorithms learn about π on the fly and use this information during the simulation
- the transition kernel P_n used for obtaining $X_n|X_{n-1}$ is allowed to depend on $\{X_0, \ldots, X_{n-1}\}$
- consequently the algorithms are **not Markovian!**
- standard MCMC theory of validating the simulation does not apply

a fundamental problem

- adaptive MCMC algorithms learn about π on the fly and use this information during the simulation
- the transition kernel P_n used for obtaining $X_n|X_{n-1}$ is allowed to depend on $\{X_0, \dots, X_{n-1}\}$
- consequently the algorithms are **not Markovian!**
- standard MCMC theory of validating the simulation does not apply

a fundamental problem

- adaptive MCMC algorithms learn about π on the fly and use this information during the simulation
- the transition kernel P_n used for obtaining $X_n|X_{n-1}$ is allowed to depend on $\{X_0, \ldots, X_{n-1}\}$
- consequently the algorithms are **not Markovian!**
- standard MCMC theory of validating the simulation does not apply

a fundamental problem

- adaptive MCMC algorithms learn about π on the fly and use this information during the simulation
- the transition kernel P_n used for obtaining $X_n|X_{n-1}$ is allowed to depend on $\{X_0, \ldots, X_{n-1}\}$
- consequently the algorithms are **not Markovian!**
- standard MCMC theory of validating the simulation does not apply

ergodicity: a toy counterexample

- Let $\mathcal{X} = \{0, 1\}$ and π be uniform.
- $$P_1 = \begin{bmatrix} 1/2 & 1/2 \\ 1/2 & 1/2 \end{bmatrix} \quad \text{and} \quad P_2 = (1-\varepsilon)\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \varepsilon P_1 \quad \text{for some} \quad \varepsilon > 0.$$
- π is the stationary distribution for both, P_1 and P_2.
- Consider X_n , evolving for $n \geq 1$ according to the following adaptive kernel:
$$Q_n = \begin{cases} P_1 & \text{if} \quad X_{n-1} = 0 \\ P_2 & \text{if} \quad X_{n-1} = 1 \end{cases}$$
- Note that after two consecutive 1 the adaptive process X_n is trapped in 1 and can escape only with probability ε.
- Let $\bar{q}_1 := \lim_{n\to\infty} P(X_n = 1)$ and $\bar{q}_0 := \lim_{n\to\infty} P(X_n = 0)$.
- Now it is clear, that for small ε we will have $\bar{q}_1 \gg \bar{q}_0$ and the procedure fails to give the expected asymptotic distribution.

Adaptive Gibbs sampler - a generic algorithm

AdapRSG

1. Set $\alpha_n := R_n(\alpha_{n-1}, X_{n-1}, \dots, X_0) \in \mathcal{Y} \subset [0,1]^d$
2. Choose coordinate $i \in \{1, \dots, d\}$ according to selection probabilities α_n
3. Draw $Y \sim \pi(\cdot | X_{n-1,-i})$
4. Set $X_n := (X_{n-1,1}, \dots, X_{n-1,i-1}, Y, X_{n-1,i+1}, \dots, X_{n-1,d})$

- It is easy to get tricked into thinking that if step 1 is not doing anything "crazy" then the algorithm must be ergodic.
- Theorem 2.1 of [LC06] states that ergodicity of adaptive Gibbs samplers follows from the following two conditions:
  - (i) $\alpha_n \to \alpha$ a.s. for some fixed $\alpha \in (0,1)^d$; and
  - (ii) The random scan Gibbs sampler with fixed selection probabilities α induces an ergodic Markov chain with stationary distribution π.
- The above theorem is simple, neat and wrong.

Adaptive Gibbs sampler - a generic algorithm

AdapRSG

1. Set $\alpha_n := R_n(\alpha_{n-1}, X_{n-1}, \dots, X_0) \in \mathcal{Y} \subset [0,1]^d$
2. Choose coordinate $i \in \{1, \dots, d\}$ according to selection probabilities α_n
3. Draw $Y \sim \pi(\cdot | X_{n-1,-i})$
4. Set $X_n := (X_{n-1,1}, \dots, X_{n-1,i-1}, Y, X_{n-1,i+1}, \dots, X_{n-1,d})$

- It is easy to get tricked into thinking that if step 1 is not doing anything ”crazy” then the algorithm must be ergodic.
- Theorem 2.1 of [LC06] states that ergodicity of adaptive Gibbs samplers follows from the following two conditions:
  - (i) $\alpha_n \to \alpha$ a.s. for some fixed $\alpha \in (0,1)^d$; and
  - (ii) The random scan Gibbs sampler with fixed selection probabilities α induces an ergodic Markov chain with stationary distribution π.
- The above theorem is simple, neat and wrong.

Adaptive Gibbs sampler - a generic algorithm

`AdapRSG`

1. Set $\alpha_n := R_n(\alpha_{n-1}, X_{n-1}, \dots, X_0) \in \mathcal{Y} \subset [0,1]^d$
2. Choose coordinate $i \in \{1, \dots, d\}$ according to selection probabilities α_n
3. Draw $Y \sim \pi(\cdot | X_{n-1,-i})$
4. Set $X_n := (X_{n-1,1}, \dots, X_{n-1,i-1}, Y, X_{n-1,i+1}, \dots, X_{n-1,d})$

- It is easy to get tricked into thinking that if step 1 is not doing anything "crazy" then the algorithm must be ergodic.
- Theorem 2.1 of [LC06] states that ergodicity of adaptive Gibbs samplers follows from the following two conditions:
  - (i) $\alpha_n \to \alpha$ a.s. for some fixed $\alpha \in (0,1)^d$; and
  - (ii) The random scan Gibbs sampler with fixed selection probabilities α induces an ergodic Markov chain with stationary distribution π.
- The above theorem is simple, neat and wrong.

Adaptive Gibbs sampler - a generic algorithm

`AdapRSG`

1. Set $\alpha_n := R_n(\alpha_{n-1}, X_{n-1}, \ldots, X_0) \in \mathcal{Y} \subset [0,1]^d$
2. Choose coordinate $i \in \{1, \ldots, d\}$ according to selection probabilities α_n
3. Draw $Y \sim \pi(\cdot | X_{n-1,-i})$
4. Set $X_n := (X_{n-1,1}, \ldots, X_{n-1,i-1}, Y, X_{n-1,i+1}, \ldots, X_{n-1,d})$

- It is easy to get tricked into thinking that if step 1 is not doing anything ”crazy” then the algorithm must be ergodic.
- Theorem 2.1 of [LC06] states that ergodicity of adaptive Gibbs samplers follows from the following two conditions:
  - (i) $\alpha_n \to \alpha$ a.s. for some fixed $\alpha \in (0,1)^d$; and
  - (ii) The random scan Gibbs sampler with fixed selection probabilities α induces an ergodic Markov chain with stationary distribution π.
- The above theorem is simple, neat and wrong.

a cautionary example that disproves [LC06]

- Let $\mathcal{X} = \{(i,j) \in \mathbb{N} \times \mathbb{N} : i = j \text{ or } i = j+1\}$,
- with target distribution given by $\pi(i,j) \propto j^{-2}$
- consider a class of adaptive random scan Gibbs samplers with update rule given by:

$$R_n\Big(\alpha_{n-1}, X_{n-1} = (i,j)\Big) = \begin{cases} \left\{\frac{1}{2} + \frac{4}{a_n}, \frac{1}{2} - \frac{4}{a_n}\right\} & \text{if} \quad i = j, \\ \left\{\frac{1}{2} - \frac{4}{a_n}, \frac{1}{2} + \frac{4}{a_n}\right\} & \text{if} \quad i = j+1, \end{cases}$$

  for some choice of the sequence $(a_n)_{n=0}^{\infty}$ satisfying $8 < a_n \nearrow \infty$
- if $a_n \to \infty$ slowly enough, then X_n is **transient** with positive probability, i.e. $\mathbb{P}(X_{1,n} \to \infty) > 0$.

a cautionary example that disproves [LC06]

- Let $\mathcal{X} = \{(i,j) \in \mathbb{N} \times \mathbb{N} : i = j \text{ or } i = j+1\}$,
- with target distribution given by $\pi(i,j) \propto j^{-2}$
- consider a class of adaptive random scan Gibbs samplers with update rule given by:

$$R_n\Big(\alpha_{n-1}, X_{n-1} = (i,j)\Big) = \begin{cases} \left\{\frac{1}{2} + \frac{4}{a_n}, \frac{1}{2} - \frac{4}{a_n}\right\} & \text{if} \quad i = j, \\ \\ \left\{\frac{1}{2} - \frac{4}{a_n}, \frac{1}{2} + \frac{4}{a_n}\right\} & \text{if} \quad i = j+1, \end{cases}$$

  for some choice of the sequence $(a_n)_{n=0}^{\infty}$ satisfying $8 < a_n \nearrow \infty$
- if $a_n \to \infty$ slowly enough, then X_n is **transient** with positive probability, i.e. $\mathbb{P}(X_{1,n} \to \infty) > 0.$

a cautionary example that disproves [LC06]

- Let $\mathcal{X} = \{(i,j) \in \mathbb{N} \times \mathbb{N} : i = j \text{ or } i = j+1\}$,
- with target distribution given by $\pi(i,j) \propto j^{-2}$
- consider a class of adaptive random scan Gibbs samplers with update rule given by:

$$R_n\Big(\alpha_{n-1}, X_{n-1} = (i,j)\Big) = \begin{cases} \left\{\frac{1}{2} + \frac{4}{a_n}, \frac{1}{2} - \frac{4}{a_n}\right\} & \text{if} \quad i = j, \\ \\ \left\{\frac{1}{2} - \frac{4}{a_n}, \frac{1}{2} + \frac{4}{a_n}\right\} & \text{if} \quad i = j+1, \end{cases}$$

  for some choice of the sequence $(a_n)_{n=0}^{\infty}$ satisfying $8 < a_n \nearrow \infty$
- if $a_n \to \infty$ slowly enough, then X_n is **transient** with positive probability, i.e. $\mathbb{P}(X_{1,n} \to \infty) > 0$.

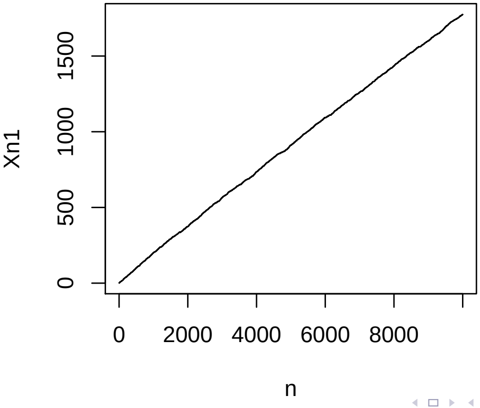
a cautionary example that disproves [LC06]

- Let $\mathcal{X} = \{(i,j) \in \mathbb{N} \times \mathbb{N} : i = j \text{ or } i = j+1\}$,
- with target distribution given by $\pi(i,j) \propto j^{-2}$
- consider a class of adaptive random scan Gibbs samplers with update rule given by:

$$R_n\Big(\alpha_{n-1}, X_{n-1} = (i,j)\Big) = \begin{cases} \left\{\frac{1}{2} + \frac{4}{a_n}, \frac{1}{2} - \frac{4}{a_n}\right\} & \text{if} \quad i = j, \\ \left\{\frac{1}{2} - \frac{4}{a_n}, \frac{1}{2} + \frac{4}{a_n}\right\} & \text{if} \quad i = j+1, \end{cases}$$

  for some choice of the sequence $(a_n)_{n=0}^{\infty}$ satisfying $8 < a_n \nearrow \infty$
- if $a_n \to \infty$ slowly enough, then X_n is **transient** with positive probability, i.e. $\mathbb{P}(X_{1,n} \to \infty) > 0$.

a cautionary example...



Ergodicity of an adaptive algorithm - framework

- $\mathcal{X}$ valued process of interest X_n
- Θ valued random parameter θ_n
  representing the choice of kernel when updating X_n to X_{n+1}
- Define the filtration generated by $\{(X_n, \theta_n)\}$

$$\mathcal{G}_n = \sigma(X_0, \ldots, X_n, \theta_0, \ldots, \theta_n),$$

- Thus

$$P(X_{n+1} \in B \mid X_n = x, \theta_n = \theta, \mathcal{G}_{n-1}) = P_\theta(x, B)$$

- The distribution of θ_{n+1} given $\mathcal{G}_n$ depends on the algorithm.
- Define

$$A^{(n)}(x, \theta, B) = P(X_n \in B \,\|\, X_0 = x, \theta_0 = \theta)$$
$$T(x, \theta, n) = \|A^{(n)}(x, \theta, \cdot) - \pi(\cdot)\|_{TV}$$

- We say the adaptive algorithm is ergodic if

$$\lim_{n\to\infty} T(x, \theta, n) = 0 \qquad \text{for all } x \in \mathcal{X} \quad \text{and } \theta \in \Theta.$$

Ergodicity of an adaptive algorithm - framework

- $\mathcal{X}$ valued process of interest X_n
- Θ valued random parameter θ_n
  representing the choice of kernel when updating X_n to X_{n+1}
- Define the filtration generated by $\{(X_n, \theta_n)\}$

$$\mathcal{G}_n = \sigma(X_0, \ldots, X_n, \theta_0, \ldots, \theta_n),$$

- Thus

$$P(X_{n+1} \in B \mid X_n = x, \theta_n = \theta, \mathcal{G}_{n-1}) = P_\theta(x, B)$$

- The distribution of θ_{n+1} given $\mathcal{G}_n$ depends on the algorithm.
- Define

$$A^{(n)}(x, \theta, B) = P(X_n \in B \,\|\, X_0 = x, \theta_0 = \theta)$$

$$T(x, \theta, n) = \|A^{(n)}(x, \theta, \cdot) - \pi(\cdot)\|_{TV}$$

- We say the adaptive algorithm is ergodic if

$$\lim_{n\to\infty} T(x, \theta, n) = 0 \qquad \text{for all } x \in \mathcal{X} \quad \text{and } \theta \in \Theta.$$

Ergodicity of an adaptive algorithm - framework

- $\mathcal{X}$ valued process of interest X_n
- Θ valued random parameter θ_n
  representing the choice of kernel when updating X_n to X_{n+1}
- Define the filtration generated by $\{(X_n, \theta_n)\}$

$$\mathcal{G}_n = \sigma(X_0, \ldots, X_n, \theta_0, \ldots, \theta_n),$$

- Thus

$$P(X_{n+1} \in B \mid X_n = x, \theta_n = \theta, \mathcal{G}_{n-1}) = P_\theta(x, B)$$

- The distribution of θ_{n+1} given $\mathcal{G}_n$ depends on the algorithm.
- Define

$$A^{(n)}(x, \theta, B) = P(X_n \in B \parallel X_0 = x, \theta_0 = \theta)$$

$$T(x, \theta, n) = \|A^{(n)}(x, \theta, \cdot) - \pi(\cdot)\|_{TV}$$

- We say the adaptive algorithm is ergodic if

$$\lim_{n\to\infty} T(x, \theta, n) = 0 \qquad \text{for all } x \in \mathcal{X} \quad \text{and } \theta \in \Theta.$$

Ergodicity of an adaptive algorithm - framework

- $\mathcal{X}$ valued process of interest X_n
- Θ valued random parameter θ_n
  representing the choice of kernel when updating X_n to X_{n+1}
- Define the filtration generated by $\{(X_n, \theta_n)\}$

$$\mathcal{G}_n = \sigma(X_0, \ldots, X_n, \theta_0, \ldots, \theta_n),$$

- Thus

$$P(X_{n+1} \in B \mid X_n = x, \theta_n = \theta, \mathcal{G}_{n-1}) = P_\theta(x, B)$$

- The distribution of θ_{n+1} given $\mathcal{G}_n$ depends on the algorithm.
- Define

$$A^{(n)}(x, \theta, B) = P(X_n \in B \,\|\, X_0 = x, \theta_0 = \theta)$$
$$T(x, \theta, n) = \|A^{(n)}(x, \theta, \cdot) - \pi(\cdot)\|_{TV}$$

- We say the adaptive algorithm is ergodic if

$$\lim_{n\to\infty} T(x, \theta, n) = 0 \qquad \text{for all } x \in \mathcal{X} \quad \text{and } \theta \in \Theta.$$

Ergodicity of an adaptive algorithm - framework

- $\mathcal{X}$ valued process of interest X_n
- Θ valued random parameter θ_n
  representing the choice of kernel when updating X_n to X_{n+1}
- Define the filtration generated by $\{(X_n, \theta_n)\}$

$$\mathcal{G}_n = \sigma(X_0, \ldots, X_n, \theta_0, \ldots, \theta_n),$$

- Thus

$$P(X_{n+1} \in B \mid X_n = x, \theta_n = \theta, \mathcal{G}_{n-1}) = P_\theta(x, B)$$

- The distribution of θ_{n+1} given $\mathcal{G}_n$ depends on the algorithm.
- Define

$$A^{(n)}(x, \theta, B) = P(X_n \in B \,\|\, X_0 = x, \theta_0 = \theta)$$
$$T(x, \theta, n) = \|A^{(n)}(x, \theta, \cdot) - \pi(\cdot)\|_{TV}$$

- We say the adaptive algorithm is ergodic if

$$\lim_{n\to\infty} T(x, \theta, n) = 0 \qquad \text{for all } x \in \mathcal{X} \quad \text{and } \theta \in \Theta.$$

Ergodicity of an adaptive algorithm - framework

- $\mathcal{X}$ valued process of interest X_n
- Θ valued random parameter θ_n
  representing the choice of kernel when updating X_n to X_{n+1}
- Define the filtration generated by $\{(X_n, \theta_n)\}$

$$\mathcal{G}_n = \sigma(X_0, \ldots, X_n, \theta_0, \ldots, \theta_n),$$

- Thus

$$P(X_{n+1} \in B \mid X_n = x, \theta_n = \theta, \mathcal{G}_{n-1}) = P_\theta(x, B)$$

- The distribution of θ_{n+1} given $\mathcal{G}_n$ depends on the algorithm.
- Define

$$A^{(n)}(x, \theta, B) = P(X_n \in B \parallel X_0 = x, \theta_0 = \theta)$$
$$T(x, \theta, n) = \|A^{(n)}(x, \theta, \cdot) - \pi(\cdot)\|_{TV}$$

- We say the adaptive algorithm is ergodic if

$$\lim_{n \to \infty} T(x, \theta, n) = 0 \qquad \text{for all } x \in \mathcal{X} \quad \text{and } \theta \in \Theta.$$

Ergodicity of an adaptive algorithm - framework

- $\mathcal{X}$ valued process of interest X_n
- Θ valued random parameter θ_n
  representing the choice of kernel when updating X_n to X_{n+1}
- Define the filtration generated by $\{(X_n, \theta_n)\}$

$$\mathcal{G}_n = \sigma(X_0, \ldots, X_n, \theta_0, \ldots, \theta_n),$$

- Thus

$$P(X_{n+1} \in B \mid X_n = x, \theta_n = \theta, \mathcal{G}_{n-1}) = P_\theta(x, B)$$

- The distribution of θ_{n+1} given $\mathcal{G}_n$ depends on the algorithm.
- Define

$$A^{(n)}(x, \theta, B) = P(X_n \in B \,\|\, X_0 = x, \theta_0 = \theta)$$
$$T(x, \theta, n) = \|A^{(n)}(x, \theta, \cdot) - \pi(\cdot)\|_{TV}$$

- We say the adaptive algorithm is ergodic if

$$\lim_{n\to\infty} T(x, \theta, n) = 0 \qquad \text{for all } x \in \mathcal{X} \quad \text{and } \theta \in \Theta.$$

Tools for establishing ergodicity

- **(Diminishing Adaptation)** Let $D_n = \sup_{x\in\mathcal{X}} \|P_{\Gamma_{n+1}}(x,\cdot) - P_{\Gamma_n}(x,\cdot)\|$ and assume $\lim_{n\to\infty} D_n = 0$ in probability
- **(Simultaneous uniform ergodicity)** For all $\varepsilon > 0$, there exists $N = N(\varepsilon)$ s.t. $\|P^N_\gamma(x,\cdot) - \pi(\cdot)\| \leq \varepsilon$ for all $x \in \mathcal{X}$ and $\gamma \in \mathcal{Y}$
- **(Containment condition)** Let $M_\varepsilon(x,\gamma) = \inf\{n \geq 1 : \|P^n_\gamma(x,\cdot) - \pi(\cdot)\| \leq \varepsilon\}$ and assume $\{M_\varepsilon(X_n,\gamma_n)\}_{n=0}^\infty$ is bounded in probability, i.e. given $X_0 = x_*$ and $\Gamma_0 = \gamma_*$, for all $\delta > 0$, there exists N s.t. $\mathbb{P}[M_\varepsilon(X_n,\Gamma_n) \leq N | X_0 = x_*, \Gamma_0 = \gamma_*] \geq 1 - \delta$ for all $n \in \mathbb{N}$.

Theorem (Roberts Rosenthal 2007)

(diminishing adaptation) + (simultaneous uniform ergodicity) ⇒ ergodicity.

Theorem (Roberts Rosenthal 2007)

(diminishing adaptation) + (containment) ⇒ ergodicity.

Tools for establishing ergodicity

- **(Diminishing Adaptation)** Let $D_n = \sup_{x\in\mathcal{X}} \|P_{\Gamma_{n+1}}(x,\cdot) - P_{\Gamma_n}(x,\cdot)\|$ and assume $\lim_{n\to\infty} D_n = 0$ in probability
- **(Simultaneous uniform ergodicity)** For all $\varepsilon > 0$, there exists $N = N(\varepsilon)$ s.t. $\|P^N_\gamma(x,\cdot) - \pi(\cdot)\| \leq \varepsilon$ for all $x \in \mathcal{X}$ and $\gamma \in \mathcal{Y}$
- **(Containment condition)** Let $M_\varepsilon(x,\gamma) = \inf\{n \geq 1 : \|P^n_\gamma(x,\cdot) - \pi(\cdot)\| \leq \varepsilon\}$ and assume $\{M_\varepsilon(X_n,\gamma_n)\}_{n=0}^\infty$ is bounded in probability, i.e. given $X_0 = x_*$ and $\Gamma_0 = \gamma_*$, for all $\delta > 0$, there exists N s.t. $\mathbb{P}[M_\varepsilon(X_n,\Gamma_n) \leq N | X_0 = x_*, \Gamma_0 = \gamma_*] \geq 1 - \delta$ for all $n \in \mathbb{N}$.

Theorem (Roberts Rosenthal 2007)

(diminishing adaptation) + (simultaneous uniform ergodicity) $\Rightarrow$ ergodicity.

Theorem (Roberts Rosenthal 2007)

(diminishing adaptation) + (containment) $\Rightarrow$ ergodicity.

Tools for establishing ergodicity

- **(Diminishing Adaptation)** Let $D_n = \sup_{x\in\mathcal{X}} \|P_{\Gamma_{n+1}}(x,\cdot) - P_{\Gamma_n}(x,\cdot)\|$ and assume $\lim_{n\to\infty} D_n = 0$ in probability
- **(Simultaneous uniform ergodicity)** For all $\varepsilon > 0$, there exists $N = N(\varepsilon)$ s.t. $\|P_\gamma^N(x,\cdot) - \pi(\cdot)\| \leq \varepsilon$ for all $x \in \mathcal{X}$ and $\gamma \in \mathcal{Y}$
- **(Containment condition)** Let $M_\varepsilon(x,\gamma) = \inf\{n \geq 1 : \|P_\gamma^n(x,\cdot) - \pi(\cdot)\| \leq \varepsilon\}$ and assume $\{M_\varepsilon(X_n,\gamma_n)\}_{n=0}^\infty$ is bounded in probability, i.e. given $X_0 = x_*$ and $\Gamma_0 = \gamma_*$, for all $\delta > 0$, there exists N s.t. $\mathbb{P}[M_\varepsilon(X_n,\Gamma_n) \leq N | X_0 = x_*, \Gamma_0 = \gamma_*] \geq 1 - \delta$ for all $n \in \mathbb{N}$.

Theorem (Roberts Rosenthal 2007)

(diminishing adaptation) + (simultaneous uniform ergodicity) ⇒ ergodicity.

Theorem (Roberts Rosenthal 2007)

(diminishing adaptation) + (containment) ⇒ ergodicity.

Tools for establishing ergodicity

- **(Diminishing Adaptation)** Let $D_n = \sup_{x\in\mathcal{X}} \|P_{\Gamma_{n+1}}(x,\cdot) - P_{\Gamma_n}(x,\cdot)\|$ and assume $\lim_{n\to\infty} D_n = 0$ in probability
- **(Simultaneous uniform ergodicity)** For all $\varepsilon > 0$, there exists $N = N(\varepsilon)$ s.t. $\|P^N_\gamma(x,\cdot) - \pi(\cdot)\| \leq \varepsilon$ for all $x \in \mathcal{X}$ and $\gamma \in \mathcal{Y}$
- **(Containment condition)** Let $M_\varepsilon(x,\gamma) = \inf\{n \geq 1 : \|P^n_\gamma(x,\cdot) - \pi(\cdot)\| \leq \varepsilon\}$ and assume $\{M_\varepsilon(X_n,\gamma_n)\}_{n=0}^\infty$ is bounded in probability, i.e. given $X_0 = x_*$ and $\Gamma_0 = \gamma_*$, for all $\delta > 0$, there exists N s.t. $\mathbb{P}[M_\varepsilon(X_n,\Gamma_n) \leq N | X_0 = x_*, \Gamma_0 = \gamma_*] \geq 1-\delta$ for all $n \in \mathbb{N}$.

Theorem (Roberts Rosenthal 2007)

(diminishing adaptation) $+$ *(simultaneous uniform ergodicity)* $\Rightarrow$ *ergodicity.*

Theorem (Roberts Rosenthal 2007)

(diminishing adaptation) $+$ *(containment)* $\Rightarrow$ *ergodicity.*

Tools for establishing ergodicity

- **(Diminishing Adaptation)** Let $D_n = \sup_{x\in\mathcal{X}} \|P_{\Gamma_{n+1}}(x,\cdot) - P_{\Gamma_n}(x,\cdot)\|$ and assume $\lim_{n\to\infty} D_n = 0$ in probability
- **(Simultaneous uniform ergodicity)** For all $\varepsilon > 0$, there exists $N = N(\varepsilon)$ s.t. $\|P_\gamma^N(x,\cdot) - \pi(\cdot)\| \leq \varepsilon$ for all $x \in \mathcal{X}$ and $\gamma \in \mathcal{Y}$
- **(Containment condition)** Let $M_\varepsilon(x,\gamma) = \inf\{n \geq 1 : \|P_\gamma^n(x,\cdot) - \pi(\cdot)\| \leq \varepsilon\}$ and assume $\{M_\varepsilon(X_n,\gamma_n)\}_{n=0}^\infty$ is bounded in probability, i.e. given $X_0 = x_*$ and $\Gamma_0 = \gamma_*$, for all $\delta > 0$, there exists N s.t. $\mathbb{P}[M_\varepsilon(X_n,\Gamma_n) \leq N | X_0 = x_*, \Gamma_0 = \gamma_*] \geq 1-\delta$ for all $n \in \mathbb{N}$.

Theorem (Roberts Rosenthal 2007)

(diminishing adaptation) $+$ *(simultaneous uniform ergodicity)* $\Rightarrow$ *ergodicity.*

Theorem (Roberts Rosenthal 2007)

(diminishing adaptation) $+$ *(containment)* $\Rightarrow$ *ergodicity.*

Containment: a closer look

- **(Containment condition)** $M_\varepsilon(x,\gamma) = \inf\{n \geq 1 : \|P^n_\gamma(x,\cdot) - \pi(\cdot)\| \leq \varepsilon\}$
  given $X_0 = x_*$ and $\Gamma_0 = \gamma_*$, for all $\delta > 0$,
  there exists N s.t. $\mathbb{P}[M_\varepsilon(X_n,\Gamma_n) \leq N | X_0 = x_*, \Gamma_0 = \gamma_*] \geq 1 - \delta$ for all $n \in \mathbb{N}$.
- Containment can be verified using simultaneous geometrical ergodicity or simultaneous polynomial ergodicity. (details in [BRR10])
- The family $\{P_\gamma : \gamma \in \mathcal{Y}\}$ is Simultaneously Geometrically Ergodic if
  - there exist a uniform ν_m-small set C i.e.
    for each γ $\quad P^m_\gamma(x,\cdot) \geq \delta\nu_\gamma(\cdot) \quad$ for all $x \in C$.
  - $P_\gamma V \leq \lambda V + b\mathbb{I}_C \quad$ for all γ.
- S.G.E. implies containment

Containment: a closer look

- **(Containment condition)** $M_\varepsilon(x,\gamma) = \inf\{n \geq 1 : \|P_\gamma^n(x,\cdot) - \pi(\cdot)\| \leq \varepsilon\}$ given $X_0 = x_*$ and $\Gamma_0 = \gamma_*$, for all $\delta > 0$, there exists N s.t. $\mathbb{P}[M_\varepsilon(X_n,\Gamma_n) \leq N | X_0 = x_*, \Gamma_0 = \gamma_*] \geq 1 - \delta$ for all $n \in \mathbb{N}$.
- Containment can be verified using simultaneous geometrical ergodicity or simultaneous polynomial ergodicity. (details in [BRR10])
- The family $\{P_\gamma : \gamma \in \mathcal{Y}\}$ is Simultaneously Geometrically Ergodic if
  - there exist a uniform ν_m-small set C i.e. for each γ $\quad P_\gamma^m(x,\cdot) \geq \delta\nu_\gamma(\cdot)$ $\quad$ for all $x \in C$.
  - $P_\gamma V \leq \lambda V + b\mathbb{I}_C$ $\quad$ for all γ.
- S.G.E. implies containment

Containment: a closer look

- **(Containment condition)** $M_\varepsilon(x,\gamma) = \inf\{n \geq 1 : \|P_\gamma^n(x,\cdot) - \pi(\cdot)\| \leq \varepsilon\}$
given $X_0 = x_*$ and $\Gamma_0 = \gamma_*$, for all $\delta > 0$,
there exists N s.t. $\mathbb{P}[M_\varepsilon(X_n,\Gamma_n) \leq N | X_0 = x_*, \Gamma_0 = \gamma_*] \geq 1 - \delta$ for all $n \in \mathbb{N}$.
- Containment can be verified using simultaneous geometrical ergodicity or simultaneous polynomial ergodicity. (details in [BRR10])
- The family $\{P_\gamma : \gamma \in \mathcal{Y}\}$ is Simultaneously Geometrically Ergodic if
  - there exist a uniform ν_m-small set C i.e.
    for each γ $\quad P_\gamma^m(x,\cdot) \geq \delta\nu_\gamma(\cdot) \quad$ for all $x \in C$.
  - $P_\gamma V \leq \lambda V + b\mathbb{I}_C \quad$ for all γ.
- S.G.E. implies containment

Containment: a closer look

- **(Containment condition)** $M_\varepsilon(x,\gamma) = \inf\{n \geq 1 : \|P_\gamma^n(x,\cdot) - \pi(\cdot)\| \leq \varepsilon\}$ given $X_0 = x_*$ and $\Gamma_0 = \gamma_*$, for all $\delta > 0$, there exists N s.t. $\mathbb{P}[M_\varepsilon(X_n,\Gamma_n) \leq N | X_0 = x_*, \Gamma_0 = \gamma_*] \geq 1-\delta$ for all $n \in \mathbb{N}$.
- Containment can be verified using simultaneous geometrical ergodicity or simultaneous polynomial ergodicity. (details in [BRR10])
- The family $\{P_\gamma : \gamma \in \mathcal{Y}\}$ is Simultaneously Geometrically Ergodic if
  - there exist a uniform ν_m-small set C i.e. for each γ $\quad P_\gamma^m(x,\cdot) \geq \delta\nu_\gamma(\cdot)$ $\quad$ for all $x \in C$.
  - $P_\gamma V \leq \lambda V + b\mathbb{I}_C$ $\quad$ for all γ.
- S.G.E. implies containment

Adaptive random scan Metropolis within Gibbs

`AdapRSMwG`

1. Set $\alpha_n := R_n(\alpha_{n-1}, X_{n-1}, \dots, X_0) \in \mathcal{Y}$
2. Choose coordinate $i \in \{1, \dots, d\}$ according to selection probabilities α_n
3. Draw $Y \sim Q_{X_{n-1,-i}}(X_{n-1,i}, \cdot)$
4. With probability

$$\min\left(1, \frac{\pi(Y|X_{n-1,-i})\, q_{X_{n-1,-i}}(Y, X_{n-1,i})}{\pi(X_{n-1}|X_{n-1,-i})\, q_{X_{n-1,-i}}(X_{n-1,i}, Y)}\right), \qquad (2)$$

accept the proposal and set

$$X_n = (X_{n-1,1}, \dots, X_{n-1,i-1}, Y, X_{n-1,i+1}, \dots, X_{n-1,d});$$

otherwise, reject the proposal and set $X_n = X_{n-1}$.

Adaptive random scan adaptive Metropolis within Gibbs

`AdapRSadapMwG`

1. Set $\alpha_n := R_n(\alpha_{n-1}, X_{n-1}, \dots, X_0, \gamma_{n-1}, \dots, \gamma_0) \in \mathcal{Y}$
2. Set $\gamma_n := R'_n(\alpha_{n-1}, X_{n-1}, \dots, X_0, \gamma_{n-1}, \dots, \gamma_0) \in \Gamma_1 \times \dots \times \Gamma_n$
3. Choose coordinate $i \in \{1, \dots, d\}$ according to selection probabilities α, i.e. with $\Pr(i = j) = \alpha_j$
4. Draw $Y \sim Q_{X_{n-1,-i}, \gamma_{n-1}}(X_{n-1,i}, \cdot)$
5. With probability (2),

$$\min\left(1, \frac{\pi(Y|X_{n-1,-i})\, q_{X_{n-1,-i},\gamma_{n-1}}(Y, X_{n-1,i})}{\pi(X_{n-1}|X_{n-1,-i})\, q_{X_{n-1,-i},\gamma_{n-1}}(X_{n-1,i}, Y)}\right),$$

   accept the proposal and set

$$X_n = (X_{n-1,1}, \dots, X_{n-1,i-1}, Y, X_{n-1,i+1}, \dots, X_{n-1,d});$$

   otherwise, reject the proposal and set $X_n = X_{n-1}$.

Ergodicity Adaptive Random Scan Gibbs [ŁRR13]

- Assuming that `RSG(`β`)` is uniformly ergodic and $|\alpha_n - \alpha_{n-1}| \to 0$, we can prove ergodicity of
  - `AdapRSG`
  - `AdapRSMwG`
  - `AdapRSadapMwG`

  by establishing diminishing adaptation and simultaneous uniform ergodicity
- Assuming that $|\alpha_n - \alpha_{n-1}| \to 0$ and regularity conditions for the target and proposal distributions (in the spirit of Roberts Rosenthal 98, Fort et al 03) ergodicity of
  - `AdapRSMwG`
  - `AdapRSadapMwG`

  can be verified by establishing diminishing adaptation and containment (by simultaneous geometrical ergodicity, using results of Bai et al 2008)

Ergodicity Adaptive Random Scan Gibbs [ŁRR13]

- Assuming that `RSG(`β`)` is uniformly ergodic and $|\alpha_n - \alpha_{n-1}| \to 0$, we can prove ergodicity of
  - `AdapRSG`
  - `AdapRSMwG`
  - `AdapRSadapMwG`

  by establishing diminishing adaptation and simultaneous uniform ergodicity
- Assuming that $|\alpha_n - \alpha_{n-1}| \to 0$ and regularity conditions for the target and proposal distributions (in the spirit of Roberts Rosenthal 98, Fort et al 03) ergodicity of
  - `AdapRSMwG`
  - `AdapRSadapMwG`

  can be verified by establishing diminishing adaptation and containment (by simultaneous geometrical ergodicity, using results of Bai et al 2008)

Adaptive Metropolis - versions and stability

- Recall the Adaptive Metropolis Algorithm with proposals

$$Y_{n+1} \sim q_{\sigma_n}(X_n, \cdot) = X_n + N(0, \Sigma_n),$$

- The theory suggests increment

$$N(0, (2.38)^2\Sigma_n/d)$$

- The AM version of [HST01] (the original one) uses

$$N(0, \Sigma_n + \varepsilon Id)$$

- Modification due to [RR09] is to use

$$Q_n = (1-\beta)N(0, (2.38)^2\Sigma_n/d) + \beta N(0, \varepsilon Id/d).$$

- the above modification appears more tractable: containment has been verified for both, exponentially and super-exponentially decaying tails (Bai et al 2009).
- the original version has been analyzed in [SV10] and [FMP10] using different techniques.

Adaptive Metropolis - versions and stability

- Recall the Adaptive Metropolis Algorithm with proposals

$$Y_{n+1} \sim q_{\sigma_n}(X_n, \cdot) = X_n + N(0, \Sigma_n),$$

- The theory suggests increment

$$N(0, (2.38)^2\Sigma_n/d)$$

- The AM version of [HST01] (the original one) uses

$$N(0, \Sigma_n + \varepsilon Id)$$

- Modification due to [RR09] is to use

$$Q_n = (1-\beta)N(0, (2.38)^2\Sigma_n/d) + \beta N(0, \varepsilon Id/d).$$

- the above modification appears more tractable: containment has been verified for both, exponentially and super-exponentially decaying tails (Bai et al 2009).
- the original version has been analyzed in [SV10] and [FMP10] using different techniques.

Adaptive Metropolis - versions and stability

- Recall the Adaptive Metropolis Algorithm with proposals

$$Y_{n+1} \sim q_{\sigma_n}(X_n, \cdot) = X_n + N(0, \Sigma_n),$$

- The theory suggests increment

$$N(0, (2.38)^2\Sigma_n/d)$$

- The AM version of [HST01] (the original one) uses

$$N(0, \Sigma_n + \varepsilon Id)$$

- Modification due to [RR09] is to use

$$Q_n = (1-\beta)N(0, (2.38)^2\Sigma_n/d) + \beta N(0, \varepsilon Id/d).$$

- the above modification appears more tractable: containment has been verified for both, exponentially and super-exponentially decaying tails (Bai et al 2009).
- the original version has been analyzed in [SV10] and [FMP10] using different techniques.

Adaptive Metropolis - versions and stability

- Recall the Adaptive Metropolis Algorithm with proposals

$$Y_{n+1} \sim q_{\sigma_n}(X_n, \cdot) = X_n + N(0, \Sigma_n),$$

- The theory suggests increment

$$N(0, (2.38)^2 \Sigma_n / d)$$

- The AM version of [HST01] (the original one) uses

$$N(0, \Sigma_n + \varepsilon Id)$$

- Modification due to [RR09] is to use

$$Q_n = (1 - \beta) N(0, (2.38)^2 \Sigma_n / d) + \beta N(0, \varepsilon Id / d).$$

- the above modification appears more tractable: containment has been verified for both, exponentially and super-exponentially decaying tails (Bai et al 2009).
- the original version has been analyzed in [SV10] and [FMP10] using different techniques.

Adaptive Metropolis - versions and stability

- Recall the Adaptive Metropolis Algorithm with proposals

$$Y_{n+1} \sim q_{\sigma_n}(X_n, \cdot) = X_n + N(0, \Sigma_n),$$

- The theory suggests increment

$$N(0, (2.38)^2 \Sigma_n / d)$$

- The AM version of [HST01] (the original one) uses

$$N(0, \Sigma_n + \varepsilon Id)$$

- Modification due to [RR09] is to use

$$Q_n = (1 - \beta) N(0, (2.38)^2 \Sigma_n / d) + \beta N(0, \varepsilon Id / d).$$

- the above modification appears more tractable: containment has been verified for both, exponentially and super-exponentially decaying tails (Bai et al 2009).
- the original version has been analyzed in [SV10] and [FMP10] using different techniques.

a new class: `AdapFail` Algorithms

- an adaptive algorithm $\mathcal{A} \in$ `AdapFail`, if with positive probability, it is asymptotically less efficient then ANY MCMC algorithm with fixed θ.
- more formally, `AdapFail` can be defined e.g. as follows: $\mathcal{A} \in$ `AdapFail`, if

$$\forall_{\epsilon_*>0}, \quad \exists_{0<\epsilon<\epsilon_*}, \quad \text{s.t.} \quad \lim_{K\to\infty} \inf_{\theta\in\Theta} \lim_{n\to\infty} P\Big(M_\epsilon(X_n,\theta_n) > KM_\epsilon(\tilde{X}_n,\theta)\Big) > 0\,,$$

  where $\{\tilde{X}_n\}$ is a Markov chain independent of $\{X_n\}$, which follows the fixed kernel P_θ.
- Lemma [ŁR14]:If containment doesn't hold for $\mathcal{A}$ then $\mathcal{A} \in$ `AdapFail`.

a new class: AdapFail Algorithms

- an adaptive algorithm $\mathcal{A} \in$ AdapFail, if with positive probability, it is asymptotically less efficient then ANY MCMC algorithm with fixed θ.
- more formally, AdapFail can be defined e.g. as follows: $\mathcal{A} \in$ AdapFail, if

$$\forall_{\epsilon_*>0}, \quad \exists_{0<\epsilon<\epsilon_*}, \quad \text{s.t.} \quad \lim_{K\to\infty} \inf_{\theta\in\Theta} \lim_{n\to\infty} P\Big(M_\epsilon(X_n,\theta_n) > KM_\epsilon(\tilde{X}_n,\theta)\Big) > 0\,,$$

where $\{\tilde{X}_n\}$ is a Markov chain independent of $\{X_n\}$, which follows the fixed kernel P_θ.
- Lemma [ŁR14]:If containment doesn't hold for $\mathcal{A}$ then $\mathcal{A} \in$ AdapFail.

a new class: AdapFail Algorithms

- an adaptive algorithm $\mathcal{A} \in$ AdapFail, if with positive probability, it is asymptotically less efficient then ANY MCMC algorithm with fixed θ.
- more formally, AdapFail can be defined e.g. as follows: $\mathcal{A} \in$ AdapFail, if

$$\forall_{\epsilon_*>0}, \quad \exists_{0<\epsilon<\epsilon_*}, \quad \text{s.t.} \quad \lim_{K\to\infty} \inf_{\theta\in\Theta} \lim_{n\to\infty} P\Big(M_\epsilon(X_n, \theta_n) > K M_\epsilon(\tilde{X}_n, \theta)\Big) > 0\,,$$

  where $\{\tilde{X}_n\}$ is a Markov chain independent of $\{X_n\}$, which follows the fixed kernel P_θ.
- Lemma [ŁR14]:If containment doesn't hold for $\mathcal{A}$ then $\mathcal{A} \in$ AdapFail.

The fly in the ointment

- Theoretical properties of adaptive MCMC have been studied using a range of techniques, such as: coupling, martingale approximations, stability of stochastic approximation (Roberts, Rosenthal, Moulines, Andrieu, Vihola, Saksman, Fort, Atchade, ...)
- Still, the theoretical underpinning of Adaptive MCMC is (even) weaker and (even) less operational than that of standard MCMC
- Using it without theoretical support may be dangerous (convergence counterexamples, AdapFail algorithms)
- Is it possible to modify other challenging adaptive algorithms to make them easier to analyze without destroying their empirical properties?

The fly in the ointment

- Theoretical properties of adaptive MCMC have been studied using a range of techniques, such as: coupling, martingale approximations, stability of stochastic approximation (Roberts, Rosenthal, Moulines, Andrieu, Vihola, Saksman, Fort, Atchade, ...)
- Still, the theoretical underpinning of Adaptive MCMC is (even) weaker and (even) less operational than that of standard MCMC
- Using it without theoretical support may be dangerous (convergence counterexamples, AdapFail algorithms)
- Is it possible to modify other challenging adaptive algorithms to make them easier to analyze without destroying their empirical properties?

The fly in the ointment

- Theoretical properties of adaptive MCMC have been studied using a range of techniques, such as: coupling, martingale approximations, stability of stochastic approximation (Roberts, Rosenthal, Moulines, Andrieu, Vihola, Saksman, Fort, Atchade, ...)
- Still, the theoretical underpinning of Adaptive MCMC is (even) weaker and (even) less operational than that of standard MCMC
- Using it without theoretical support may be dangerous (convergence counterexamples, AdapFail algorithms)
- Is it possible to modify other challenging adaptive algorithms to make them easier to analyze without destroying their empirical properties?

The fly in the ointment

- Theoretical properties of adaptive MCMC have been studied using a range of techniques, such as: coupling, martingale approximations, stability of stochastic approximation (Roberts, Rosenthal, Moulines, Andrieu, Vihola, Saksman, Fort, Atchade, ...)
- Still, the theoretical underpinning of Adaptive MCMC is (even) weaker and (even) less operational than that of standard MCMC
- Using it without theoretical support may be dangerous (convergence counterexamples, AdapFail algorithms)
- Is it possible to modify other challenging adaptive algorithms to make them easier to analyze without destroying their empirical properties?

AirMCMC - Adapting increasingly rarely [CLR18b]

- $P_\gamma, \gamma \in \Gamma$ - a parametric family of π-invariant kernels;
  Adaptive MCMC steps:
  (1) Sample X_{n+1} from $P_{\gamma^n}(X_n, \cdot)$.
  (2) Given $\{X_0, .., X_{n+1}, \gamma^0, .., \gamma^n\}$ update γ^{n+1} according to some adaptation rule.
- How tweak the strategy to make theory easier?
- Do we need to adapt in every step?
- How about adapting increasingly rarely?
- **AirMCMC Sampler** [CLR18b]
  Initiate $X_0 \in \mathcal{X}$, $\gamma^0 \in \Gamma$. $\overline{\gamma} := \gamma^0$ $k := 1$, $n := 0$.
  (1) For $i = 1, .., n_k$
    1.1. sample $X_{n+i} \sim P_{\overline{\gamma}}(X_{n+i-1}, \cdot)$;
    1.2. given $\{X_0, .., X_{n+i}, \gamma_0, .., \gamma_{n+i-1}\}$ update γ_{n+i} according to some adaptation rule.
  (2) Set $n := n + n_k$, $k := k + 1$. $\overline{\gamma} := \gamma_n$.
- Will such a strategy be efficient? With say $n_k = ck^\beta$
- Will it be mathematically more tractable?

AirMCMC - Adapting increasingly rarely [CLR18b]

- $P_\gamma, \gamma \in \Gamma$ - a parametric family of π-invariant kernels;
  Adaptive MCMC steps:
  (1) Sample X_{n+1} from $P_{\gamma^n}(X_n, \cdot)$.
  (2) Given $\{X_0, .., X_{n+1}, \gamma^0, .., \gamma^n\}$ update γ^{n+1} according to some adaptation rule.
- How tweak the strategy to make theory easier?
- Do we need to adapt in every step?
- How about adapting increasingly rarely?
- **AirMCMC Sampler** [CLR18b]
  Initiate $X_0 \in \mathcal{X}$, $\gamma^0 \in \Gamma$. $\overline{\gamma} := \gamma^0$ $k := 1$, $n := 0$.
  (1) For $i = 1, .., n_k$
    1.1. sample $X_{n+i} \sim P_{\overline{\gamma}}(X_{n+i-1}, \cdot)$;
    1.2. given $\{X_0, .., X_{n+i}, \gamma_0, .., \gamma_{n+i-1}\}$ update γ_{n+i} according to some adaptation rule.
  (2) Set $n := n + n_k$, $k := k + 1$. $\overline{\gamma} := \gamma_n$.
- Will such a strategy be efficient? With say $n_k = ck^\beta$
- Will it be mathematically more tractable?

AirMCMC - Adapting increasingly rarely [CLR18b]

- $P_\gamma, \gamma \in \Gamma$ - a parametric family of π-invariant kernels;
  Adaptive MCMC steps:
  (1) Sample X_{n+1} from $P_{\gamma^n}(X_n, \cdot)$.
  (2) Given $\{X_0, .., X_{n+1}, \gamma^0, .., \gamma^n\}$ update γ^{n+1} according to some adaptation rule.
- How tweak the strategy to make theory easier?
- Do we need to adapt in every step?
- How about adapting increasingly rarely?
- **AirMCMC Sampler** [CLR18b]
  Initiate $X_0 \in \mathcal{X}$, $\gamma^0 \in \Gamma$. $\overline{\gamma} := \gamma^0$ $k := 1$, $n := 0$.
  (1) For $i = 1, .., n_k$
    1.1. sample $X_{n+i} \sim P_{\overline{\gamma}}(X_{n+i-1}, \cdot)$;
    1.2. given $\{X_0, .., X_{n+i}, \gamma_0, .., \gamma_{n+i-1}\}$ update γ_{n+i} according to some adaptation rule.
  (2) Set $n := n + n_k$, $k := k + 1$. $\overline{\gamma} := \gamma_n$.
- Will such a strategy be efficient? With say $n_k = ck^\beta$
- Will it be mathematically more tractable?

AirMCMC - Adapting increasingly rarely [CLR18b]

- $P_\gamma, \gamma \in \Gamma$ - a parametric family of π-invariant kernels;
  Adaptive MCMC steps:
  (1) Sample X_{n+1} from $P_{\gamma^n}(X_n, \cdot)$.
  (2) Given $\{X_0, .., X_{n+1}, \gamma^0, .., \gamma^n\}$ update γ^{n+1} according to some adaptation rule.
- How tweak the strategy to make theory easier?
- Do we need to adapt in every step?
- How about adapting increasingly rarely?
- **AirMCMC Sampler** [CLR18b]
  Initiate $X_0 \in \mathcal{X}$, $\gamma^0 \in \Gamma$. $\overline{\gamma} := \gamma^0$ $k := 1$, $n := 0$.
  (1) For $i = 1, .., n_k$
    1.1. sample $X_{n+i} \sim P_{\overline{\gamma}}(X_{n+i-1}, \cdot)$;
    1.2. given $\{X_0, .., X_{n+i}, \gamma_0, .., \gamma_{n+i-1}\}$ update γ_{n+i} according to some adaptation rule.
  (2) Set $n := n + n_k$, $k := k + 1$. $\overline{\gamma} := \gamma_n$.
- Will such a strategy be efficient? With say $n_k = ck^\beta$
- Will it be mathematically more tractable?

AirMCMC - Adapting increasingly rarely [CLR18b]

- $P_\gamma, \gamma \in \Gamma$ - a parametric family of π-invariant kernels;
  Adaptive MCMC steps:
  (1) Sample X_{n+1} from $P_{\gamma^n}(X_n, \cdot)$.
  (2) Given $\{X_0, .., X_{n+1}, \gamma^0, .., \gamma^n\}$ update γ^{n+1} according to some adaptation rule.
- How tweak the strategy to make theory easier?
- Do we need to adapt in every step?
- How about adapting increasingly rarely?
- **AirMCMC Sampler** [CLR18b]
  Initiate $X_0 \in \mathcal{X}$, $\gamma^0 \in \Gamma$. $\overline{\gamma} := \gamma^0$ $k := 1$, $n := 0$.
  (1) For $i = 1, .., n_k$
    1.1. sample $X_{n+i} \sim P_{\overline{\gamma}}(X_{n+i-1}, \cdot)$;
    1.2. given $\{X_0, .., X_{n+i}, \gamma_0, .., \gamma_{n+i-1}\}$ update γ_{n+i} according to some adaptation rule.
  (2) Set $n := n + n_k$, $k := k + 1$. $\overline{\gamma} := \gamma_n$.
- Will such a strategy be efficient? With say $n_k = ck^\beta$
- Will it be mathematically more tractable?

AirMCMC - Adapting increasingly rarely [CLR18b]

- $P_\gamma, \gamma \in \Gamma$ - a parametric family of π-invariant kernels;
  Adaptive MCMC steps:
  (1) Sample X_{n+1} from $P_{\gamma^n}(X_n, \cdot)$.
  (2) Given $\{X_0, .., X_{n+1}, \gamma^0, .., \gamma^n\}$ update γ^{n+1} according to some adaptation rule.
- How tweak the strategy to make theory easier?
- Do we need to adapt in every step?
- How about adapting increasingly rarely?
- **AirMCMC Sampler** [CLR18b]
  Initiate $X_0 \in \mathcal{X}$, $\gamma^0 \in \Gamma$. $\overline{\gamma} := \gamma^0$ $k := 1$, $n := 0$.
  (1) For $i = 1, .., n_k$
    1.1. sample $X_{n+i} \sim P_{\overline{\gamma}}(X_{n+i-1}, \cdot)$;
    1.2. given $\{X_0, .., X_{n+i}, \gamma_0, .., \gamma_{n+i-1}\}$ update γ_{n+i} according to some adaptation rule.
  (2) Set $n := n + n_k$, $k := k + 1$. $\overline{\gamma} := \gamma_n$.
- Will such a strategy be efficient? With say $n_k = ck^\beta$
- Will it be mathematically more tractable?

AirMCMC - a simulation study

- $\pi(x) = \frac{l(|x|)}{|x|^{1+r}}, \; x \in \mathbb{R},$
- Air version of RWM adaptive scaling
- The example is polynomially ergodic (not easy for the sampler)
- **AirRWM**
  Initiate $X_0 \in \mathbb{R}$, $\overline{\gamma} \in [q_1, q_2]$. $k := 1$, $n := 0$, a sequence $\{c_k\}_{k \geq 1}$.
  (1) For $i = 1, .., n_k$
    (1.1.) sample $Y \sim N(X_{n+i-1}, \overline{\gamma})$, $a_{\overline{\gamma}} := \frac{\phi(Y)}{\phi(X_{n+i-1})}$;
    (1.2.) $X_{n+i} := \begin{cases} Y & \text{with probability} \quad a_{\overline{\gamma}}, \\ X_{n+i-1} & \text{with probability} \quad 1 - a_{\overline{\gamma}}; \end{cases}$
    (1.3.) $a := a + a_{\overline{\gamma}}$.
    If $i = n_k$ then
    $\overline{\gamma} := \exp\left(\log(\overline{\gamma}) + c_n\left(\frac{a}{n_k} - 0.44\right)\right)$.
  (2) Set $n := n + n_k$, $k := k + 1$, $a := 0$.

AirMCMC - a simulation study

- $\pi(x) = \frac{l(|x|)}{|x|^{1+r}}, \ x \in \mathbb{R},$
- Air version of RWM adaptive scaling
- The example is polynomially ergodic (not easy for the sampler)
- **AirRWM**
  Initiate $X_0 \in \mathbb{R}$, $\overline{\gamma} \in [q_1, q_2]$. $k := 1$, $n := 0$, a sequence $\{c_k\}_{k \geq 1}$.
  (1) For $i = 1, .., n_k$
    (1.1.) sample $Y \sim N(X_{n+i-1}, \overline{\gamma})$, $a_{\overline{\gamma}} := \frac{\phi(Y)}{\phi(X_{n+i-1})}$;
    (1.2.) $X_{n+i} := \begin{cases} Y & \text{with probability} \quad a_{\overline{\gamma}}, \\ X_{n+i-1} & \text{with probability} \quad 1 - a_{\overline{\gamma}}; \end{cases}$
    (1.3.) $a := a + a_{\overline{\gamma}}$.
    If $i = n_k$ then
    $\overline{\gamma} := \exp\left(\log(\overline{\gamma}) + c_n \left(\frac{a}{n_k} - 0.44\right)\right)$.
  (2) Set $n := n + n_k$, $k := k + 1$, $a := 0$.

AirMCMC - a simulation study

- $\pi(x) = \frac{l(|x|)}{|x|^{1+r}}, \; x \in \mathbb{R},$
- Air version of RWM adaptive scaling
- The example is polynomially ergodic (not easy for the sampler)
- **AirRWM**
  Initiate $X_0 \in \mathbb{R}$, $\overline{\gamma} \in [q_1, q_2]$. $k := 1$, $n := 0$, a sequence $\{c_k\}_{k \geq 1}$.
  (1) For $i = 1, .., n_k$
    (1.1.) sample $Y \sim N(X_{n+i-1}, \overline{\gamma})$, $a_{\overline{\gamma}} := \frac{\phi(Y)}{\phi(X_{n+i-1})}$;
    (1.2.) $X_{n+i} := \begin{cases} Y & \text{with probability} \quad a_{\overline{\gamma}}, \\ X_{n+i-1} & \text{with probability} \quad 1 - a_{\overline{\gamma}}; \end{cases}$
    (1.3.) $a := a + a_{\overline{\gamma}}$.
    If $i = n_k$ then
    $\overline{\gamma} := \exp\left(\log(\overline{\gamma}) + c_n\left(\frac{a}{n_k} - 0.44\right)\right).$
  (2) Set $n := n + n_k$, $k := k + 1$, $a := 0$.

AirMCMC - a simulation study

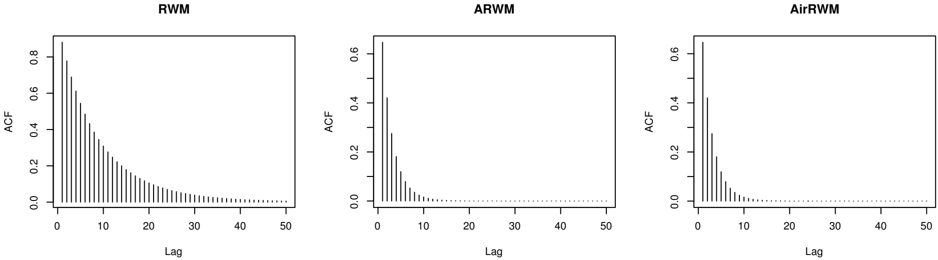
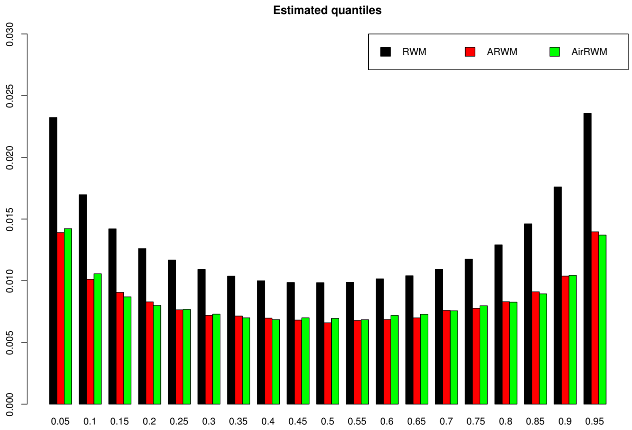
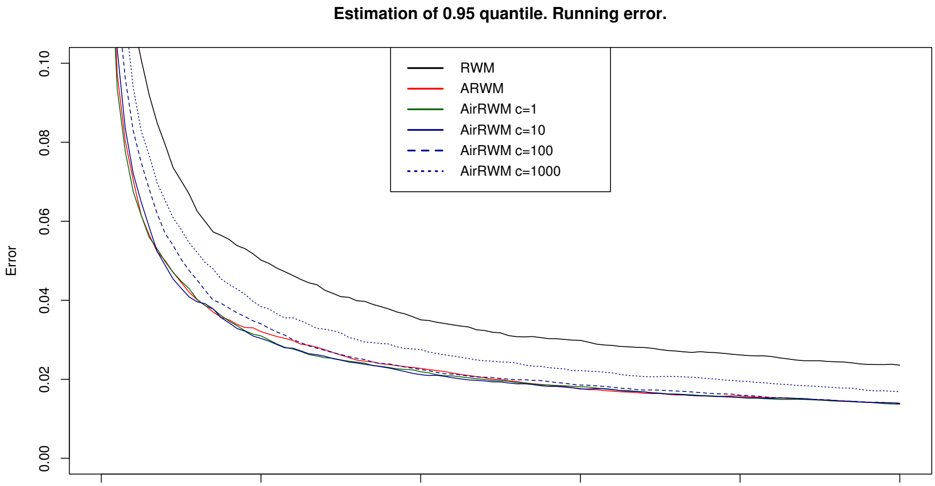


Figure: Autocorrelations (ACF)

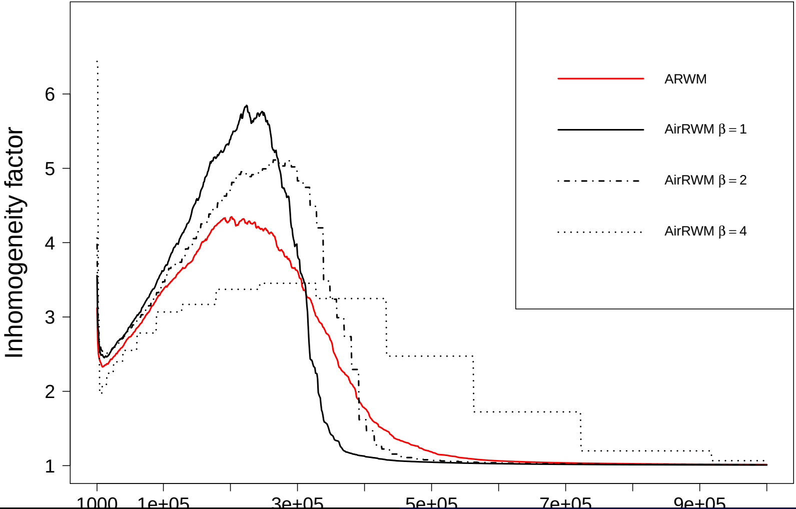
AirMCMC - a simulation study



AirMCMC - a simulation study



AirMCMC - inhomogeneity factor, d=100



AirMCMC - simulation effort, d=100

Table 1: Time to obtain 1 million samples

	ARWM	AirRWM $\beta = 1$	AirRWM $\beta = 2$	AirRWM $\beta = 4$
Time (seconds)	507.6	90.5	86.9	80.2

AirMCMC theory

- **Theorem 1**
  - Kernels Simultaneously Geometrically Ergodic (SGE)
  - $n_k \geq ck^{\beta}, \quad \beta > 0$
  - $\sup \frac{|f(x)|}{V^{1/2}(x)} < \infty$

  Then
  - WLLN
  - if $\beta > 0$, also SLLN
  - if $\beta \geq 1$, also $MSE = \mathcal{O}(1/n)$
  - if $\beta > 1$ and a bit more regularity, also CLT holds!
-
- Counterparts of this theorem also for
  - Kernels locally SGE
  - Kernels Polynomially Simultaneously Ergodic
- Note that diminishing adaptation is not needed!

AirMCMC theory

- **Theorem 1**
  - Kernels Simultaneously Geometrically Ergodic (SGE)
  - $n_k \geq ck^\beta, \quad \beta > 0$
  - $\sup \frac{|f(x)|}{V^{1/2}(x)} < \infty$

  Then
  - WLLN
  - if $\beta > 0$, also SLLN
  - if $\beta \geq 1$, also $MSE = \mathcal{O}(1/n)$
  - if $\beta > 1$ and a bit more regularity, also CLT holds!
-
- Counterparts of this theorem also for
  - Kernels locally SGE
  - Kernels Polynomially Simultaneously Ergodic
- Note that diminishing adaptation is not needed!

AirMCMC theory

- **Theorem 1**
  - Kernels Simultaneously Geometrically Ergodic (SGE)
  - $n_k \geq ck^{\beta}, \quad \beta > 0$
  - $\sup \frac{|f(x)|}{V^{1/2}(x)} < \infty$

  Then
  - WLLN
  - if $\beta > 0$, also SLLN
  - if $\beta \geq 1$, also $MSE = \mathcal{O}(1/n)$
  - if $\beta > 1$ and a bit more regularity, also CLT holds!
-
- Counterparts of this theorem also for
  - Kernels locally SGE
  - Kernels Polynomially Simultaneously Ergodic
- Note that diminishing adaptation is not needed!

Y. Bai, G.O. Roberts, and J.S. Rosenthal.
On the containment condition for adaptive Markov chain Monte Carlo algorithms.
Preprint, 2010.

M. A. Clyde, J. Ghosh, and M. L. Littman.
Bayesian adaptive sampling for variable selection and model averaging.
Journal of Computational and Graphical Statistics, 20:80–101, 2011.

Cyril Chimisov, Krzysztof Latuszynski, and Gareth Roberts.
Adapting the gibbs sampler.
arXiv preprint arXiv:1801.09299, 2018.

Cyril Chimisov, Krzysztof Latuszynski, and Gareth Roberts.
Air markov chain monte carlo.
arXiv preprint arXiv:1801.09309, 2018.

G. Fort, E. Moulines, and P. Priouret.
Convergence of adaptive mcmc algorithms: Ergodicity and law of large numbers.
2010.

Jim Griffin, Krys Latuszynski, and Mark Steel.
In search of lost (mixing) time: Adaptive markov chain monte carlo schemes for bayesian variable selection with very large p.
arXiv preprint arXiv:1708.05678v2, 2017.

W.R. Gilks, G.O. Roberts, and S.K. Sahu.
Adaptive Markov chain Monte Carlo through regeneration.
Journal of the American Statistical Association, 93(443):1045–1054, 1998.

H. Haario, E. Saksman, and J. Tamminen.
An adaptive Metropolis algorithm.
Bernoulli, 7(2):223–242, 2001.

Chunlin Ji and Scott C Schmidler.
Adaptive Markov chain Monte Carlo for Bayesian variable selection.
Journal of Computational and Graphical Statistics, 22(3):708–728, 2013.

R.A. Levine and G. Casella.
Optimizing random scan Gibbs samplers.
Journal of Multivariate Analysis, 97(10):2071–2100, 2006.

Krzysztof Łatuszyński and Jeffrey Seth Rosenthal.
The containment condition and AdapFail algorithms.
Journal of Applied Probability, 51(4):1189–1195, 2014.

K. Łatuszyński, G.O. Roberts, and J.S. Rosenthal.
Adaptive Gibbs samplers and related MCMC methods.
Ann. Appl. Probab., 23(1):66–98, 2013.

D.J. Nott and R. Kohn.
Adaptive sampling for Bayesian variable selection.
Biometrika, 92(4):747–763, 2005.

G.O. Roberts, A. Gelman, and W.R. Gilks.
Weak convergence and optimal scaling of random walk Metropolis algorithms.
The Annals of Applied Probability, 7(1):110–120, 1997.

G.O. Roberts and J.S. Rosenthal.
Optimal scaling for various Metropolis-Hastings algorithms.
Statistical Science, 16(4):351–367, 2001.

G.O. Roberts and J.S. Rosenthal.
Examples of adaptive MCMC.
Journal of Computational and Graphical Statistics, 18(2):349–367, 2009.

E. Saksman and M. Vihola.
On the ergodicity of the adaptive Metropolis algorithm on unbounded domains.
The Annals of Applied Probability, 20(6):2178–2203, 2010.

- M. Vihola.
  On the stability and ergodicity of an adaptive scaling Metropolis algorithm.
  Arxiv preprint arXiv:0903.4061, 2009.

- Y. Yang, M. Wainwright, and M. I. Jordan.
  On the computational complexity of high-dimensional Bayesian variable selection.
  Annals of Statistics, 44:2497–2532, 2016.