Adaptive Markov Chain Monte Carlo Algorithms

Jeffrey S. Rosenthal, University of Toronto, Adap'ski III

jeff@math.toronto.edu   http://probability.ca/jeff/

G.O. Roberts and J.S. Rosenthal, Coupling and Ergodicity of Adaptive MCMC. J. Appl. Prob. **44** (2007), 458–475.

G.O. Roberts and J.S. Rosenthal, Examples of Adaptive MCMC. J. Comp. Graph. Stat. **18** (2009), 349–367.

J.S. Rosenthal, AMCMC: An R/C Package for Running Adaptive MCMC. Comp. Stat. Data Anal. **51** (2007), 5467–5470.

R.V. Craiu, J.S. Rosenthal, and C. Yang, Learn From Thy Neighbor: Parallel-Chain Adaptive MCMC. JASA **488** (2009), 1454–1466.

Y. Bai, G.O. Roberts, and J.S. Rosenthal, On the Containment Condition for Adaptive Markov Chain Monte Carlo Algorithms. Submitted.

Markov Chain Monte Carlo (MCMC)

Have a complicated, high-dimensional target distribution $\pi(\cdot)$.

Define an ergodic Markov chain (random process) $X_0, X_1, X_2, \ldots$, which converges in distribution to $\pi(\cdot)$.

Then for "large enough" n, $\mathcal{L}(X_n) \approx \pi(\cdot)$, so X_n, X_{n+1}, $\ldots$ are approximate samples from $\pi(\cdot)$, and e.g.

$$\mathbf{E}_\pi(h) \approx \frac{1}{m} \sum_{i=n+1}^{n+m} h(X_i), \quad \text{etc.}$$

Extremely popular, especially for Bayesian inference.

Optimising MCMC

A given target distribution $\pi(\cdot)$ can be sampled with many possible Markov chains:

- Metropolis-Hastings
- Gibbs sampler
- Metropolis-within-Gibbs
- Langevin algorithms
- Momentum variables (hybrid chains)
- etc.

Even within one category (e.g. Metropolis-Hastings algorithms), many possible choices of proposal distribution, scaling/tuning, etc.

How to find the good chains among the bad ones?

Adaptive MCMC

Suppose have a family $\{P_\gamma\}_{\gamma \in \mathcal{Y}}$ of possible Markov chains, each with stationary distribution $\pi(\cdot)$.

How to choose among them?

Let the computer decide!

At iteration n, use Markov chain P_{Γ_n}, where $\Gamma_n \in \mathcal{Y}$ chosen according to some adaptive rules (depending on chain's history, etc.).

Can this help us to find better Markov chains? (Yes!)

On the other hand, the Markov property, stationarity, etc. are all destroyed by using an adaptive scheme.

Is the resulting algorithm still ergodic? (Sometimes!)

Example: High-Dimensional Adaptive Metropolis

Dim $d = 100$, with target $\pi(\cdot)$ equal to (say) MVN(μ, Σ).

So target covariance Σ is 100×100 (i.e., 5,050 distinct entries).

Known (Roberts-Gelman-Gilks 1997, Roberts-R. 2001, Bédard 2006) that "optimal" Gaussian RWM proposal is $N(x, \, (2.38)^2 \, d^{-1} \, \Sigma)$.

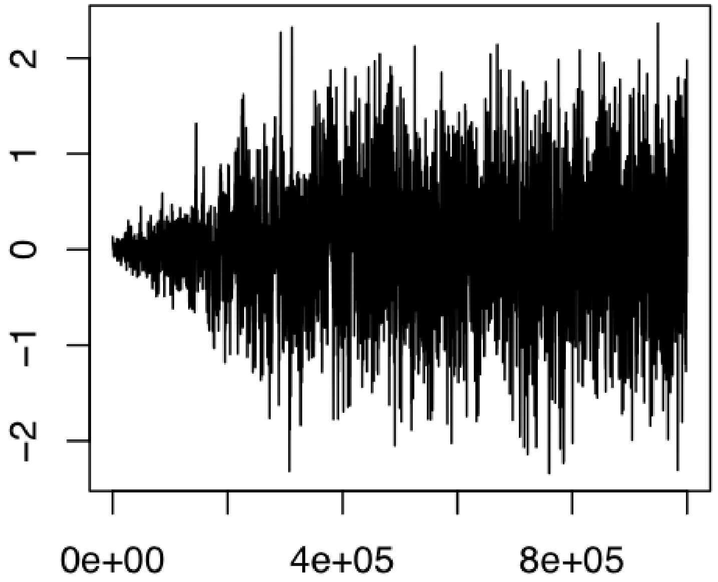
But usually Σ unknown. Instead use empirical estimate, Σ_n. Let

$$Q_n(x, \cdot) \;=\; (1-\beta)\, N(x, \, (2.38)^2 \, d^{-1} \, \Sigma_n) + \beta \, N(x, \, (0.1)^2 \, d^{-1} \, I_d) \, .$$

(Slight variant of the algorithm of Haario et al., Bernoulli 2001.)

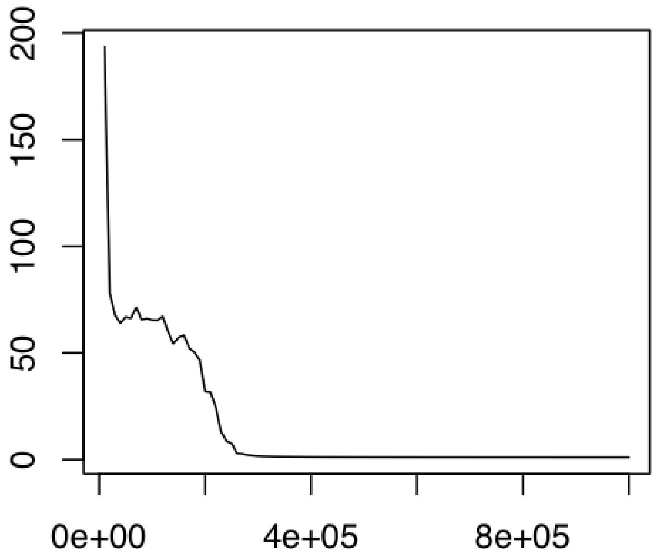
So, how well does it work??

High-Dimensional Adaptive Metropolis (cont'd)



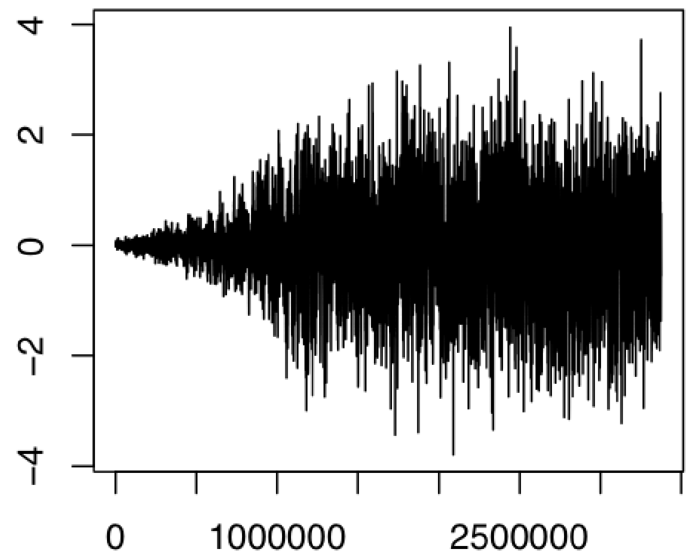
Plot of first coord. Takes about 300,000 iterations, then "finds" good proposal covariance and starts mixing well.

High-Dimensional Adaptive Metropolis (cont'd)



Plot of sub-optimality factor $b_n \equiv d\,\left(\Sigma_{i=1}^{d} \lambda_{in}^{-2} \,/\, (\Sigma_{i=1}^{d} \lambda_{in}^{-1})^2\right)$, where $\{\lambda_{in}\}$ eigenvals of $\Sigma_n^{1/2}\,\Sigma^{-1/2}$. Starts large, converges to 1.

Even Higher-Dimensional Adaptation



In dimension 200, takes about 2,000,000 iterations, then finds good proposal covariance and starts mixing well.

But is it Ergodic?

So, adaptive MCMC seems to work well in practice.

But will it be ergodic, i.e. converge to $\pi(\cdot)$??

Perhaps not, since Markov property, stationarity, etc. are all destroyed by using an adaptive scheme.

[This is in contrast to other MCMC modifications which preserve the stationary Markovian structure, and are thus automatically ergodic, e.g. regenerative adaption (Gilks et al., 1998; Brockwell and Kadane, 2005); delayed rejection sampling (Tierney and Mira, 1999); short-cut Metropolis (Neal, 2005).]

What to do?

Very Simple Example (Java Applet)

$\pi(\cdot)$ simple distribution on $\mathcal{X} = \{1, 2, 3, 4, 5, 6\}$.

[Take $\pi(x) = 0$ for $x \notin \mathcal{X}$.]

Fix $\gamma \in \mathbf{N}$, e.g. $\gamma = 2$. Do "random-walk Metropolis" (RWM):

- Given X_n, first propose a state $Y_{n+1} \in \mathbf{Z}$, with $Y_{n+1} \sim \text{Uniform}\{X_n - \gamma, \ldots, X_n - 1, X_n + 1, \ldots, X_n + \gamma\}$.
- Then, with probability $\min[1, \pi(Y_{n+1})/\pi(X_n)]$, accept proposal and set $X_{n+1} = Y_{n+1}$.
- Otherwise, with probability $1 - \min[1, \pi(Y_{n+1})/\pi(X_n)]$, reject proposal and set $X_{n+1} = X_n$. [APPLET]

Choosing the Tuning Parameter

This works, i.e. $\mathcal{L}(X_n) \to \pi(\cdot)$. But should $\gamma = 1$, or 50, or . . . ?

• If γ too small (say, $\gamma = 1$), then usually accept, but move very slowly – bad.

• If γ too large (say, $\gamma = 50$), then usually $\pi(Y_{n+1}) = 0$, i.e. hardly ever accept – bad.

• Best is a “moderate” value of γ like 3 or 4: step sizes and acceptance probs both non-small. [“Goldilocks principle”]

Obvious in this example. But in general?

Can use “trial and error” (time-consuming, unreliable).

Or can have computer adapt . . .

Adaption in Java Applet Example

Want acceptance rate to be not too small, not too big. So:

Start with γ set to $\Gamma_0 = 2$ (say).

Each time proposal is accepted, set $\Gamma_{n+1} = \Gamma_n + 1$ (so γ increases, and acceptance rate decreases).

Each time proposal is rejected, set $\Gamma_{n+1} = \max(\Gamma_n - 1,\, 1)$ (so γ decreases, and acceptance rate increases).

Logical, natural adaptive scheme, which uses the computer to perform a "search" for a good γ, on the fly.

But does it work? [APPLET]

Adaption in Java Applet Example (cont'd)

NO, IT DOESN'T WORK!!

Nearly works if $\pi(\cdot)$ close to uniform.

But if $\pi\{2\}$ small, the chain gets "stuck" with $X_n = \Gamma_n = 1$ for long stretches of time.

[Asymmetric: entering $\{X_n = \Gamma_n = 1\}$ much easier than leaving.]

Chain doesn't converge to $\pi(\cdot)$ at all.

The adaption has RUINED the algorithm. Disaster!

This shows that adaption is tricky!

[Could convolve with $N(0, 10^{-6})$, to make it continuous . . .]

When Does Adaptation Preserve Stationarity?

Many recent results, by many smart people, e.g.:

Finnish: Haario, Saksman, Tamminen, Vihola, . . .

French: Andrieu, Moulines, Fort, Atchadé, . . .

Australian: Kohn, Giordani, Nott, . . .

Formally, suppose $\{P_\gamma\}_{\gamma \in \mathcal{Y}}$ is a family of Markov chains, with $\pi(\cdot)$ stationary for each P_γ, and adaption algorithm is defined by:

$$\mathbf{P}[X_{n+1} \in A \,|\, X_n = x, \Gamma_n = \gamma, \mathcal{G}_{n-1}] \;=\; P_\gamma(x, A)\,.$$

WANT: Simple conditions guaranteeing $\|\mathcal{L}(X_n) - \pi(\cdot)\| \to 0$,

where $\|\mathcal{L}(X_n) - \pi(\cdot)\| \equiv \sup_{A \subseteq \mathcal{X}} |\mathbf{P}(X_n \in A) - \pi(A)|$.

One Simple Convergence Theorem

THEOREM [Roberts and R., J.A.P. 2007]: An adaptive scheme using $\{P_\gamma\}_{\gamma \in \mathcal{Y}}$ will converge, i.e. $\lim_{n\to\infty} \|\mathcal{L}(X_n) - \pi(\cdot)\| = 0$, if:

(a) [Diminishing Adaptation] Adapt less and less as the algorithm proceeds. Formally, $\sup_{x\in\mathcal{X}} \|P_{\Gamma_{n+1}}(x,\cdot) - P_{\Gamma_n}(x,\cdot)\| \to 0$ in prob. [Can always be made to hold, since adaption is user controlled.]

(b) [Containment] Times to stationary from X_n, if fix $\gamma = \Gamma_n$, remain bounded in probability as $n \to \infty$. [Technical condition, to avoid "escape to infinity". Holds if e.g. $\mathcal{X}$ and $\mathcal{Y}$ finite, or compact, or sub-exponential tails, or ... (Bai, Roberts, and R., sub.). And always seems to hold in practice.]

(Also guarantees WLLN for bounded functionals. Various other results about LLN, CLT under stronger assumptions.)

Implications of Theorem

COR: Validity of the above Adaptive Metropolis algorithms.

Proof: Empirical estimates have Diminishing Adaptation. And, Containment easily guaranteed by that extra "$\beta\, N(x,\ (0.1)^2 d^{-1} I_d)$", or by the "bounded" assumptions of Haario et al. (2001). $\blacksquare$

(So, the theorem provides easier proof of previously-known result.)

COR: Java Applet example is valid <u>if</u>, at time n, adapt only with probability $p(n)$, where $p(n) \to 0$, otherwise leave γ unchanged.

Proof: $p(n) \to 0$ guarantees Diminishing Adaptation. And, since $\mathcal{X}$ is finite, containment is easily verified. $\blacksquare$

e.g. let $p(n) = 1/n$, then $\Sigma_n\, p(n) = \infty$, so <u>infinite</u> adaptation.

Other examples?

Example: Adaptive Metropolis-within-Gibbs

Propose increment $N(0,\ e^{2\,ls_i})$ for i^{th} coordinate, leaving the other coordinates fixed. Choice of ls_i??

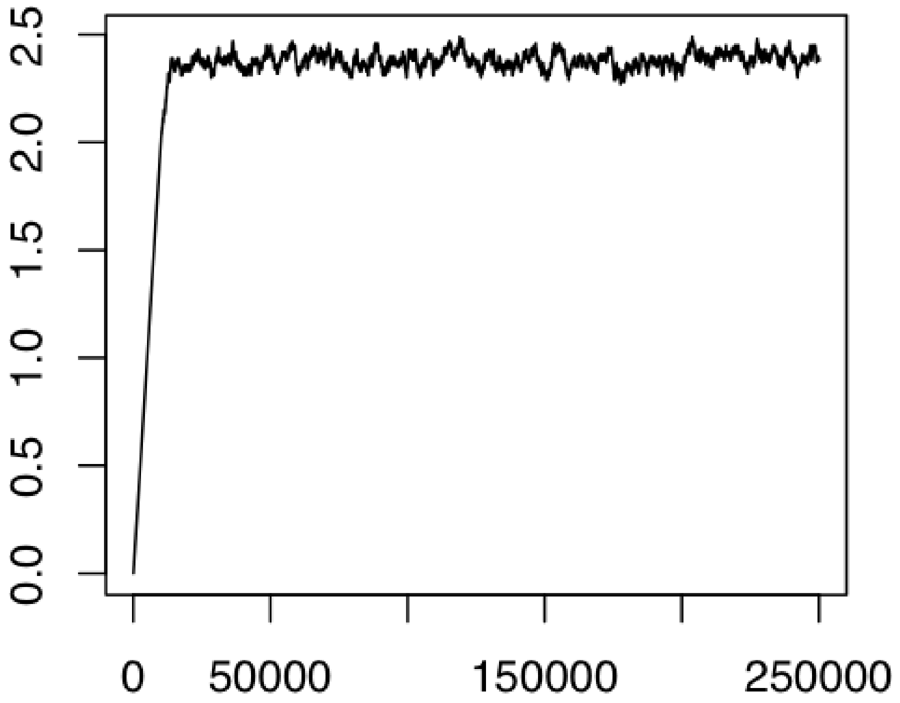
Start with $ls_i \equiv 0$ (say).

Adapt each ls_i, in batches, to again seek 0.44 acceptance rate.

Test on Variance Components Model, with $K = 500$ (dim=503), J_i chosen with $5 \le J_i \le 500$, data $Y_{ij} \sim N(i-1,\ 10^2)$.

How well does it work?

Metropolis-within-Gibbs (cont'd)



Adaption finds "good" values for the ls_i values.

Metropolis-within-Gibbs: Comparisons

Variable	J_i	Algorithm	ls_i	ACT	Avr Sq Dist
θ_1	5	Adaptive	2.4	2.59	14.932
θ_1	5	Fixed	0	31.69	0.863
θ_2	50	Adaptive	1.2	2.72	1.508
θ_2	50	Fixed	0	7.33	0.581
θ_3	500	Adaptive	0.1	2.72	0.150
θ_3	500	Fixed	0	2.67	0.147

Adaptive much better than Fixed, even in dimension 503.

This algorithm applied to statistical genetics models by Turro, Bochkina, Hein, and Richardson (BMC Bioinformatics, 2007).

Adapting Random-Scan Gibbs Sampler Weights

If some coordinates less "important" than others, may want to update them less often.

Hence, adapt the random-scan coordinate weights, $\{\alpha_{n,i}\}$.

What conditions ensure ergodicity?

Claim [J. Mult. Anal. **97** (2006), p. 2075]: suffices that $\lim_{n\to\infty} \alpha_{n,i} = \alpha_i^*$, where the Gibbs sampler with fixed weights $\{\alpha_i^*\}$ is ergodic.

Really?? Counter-example! (K. Latuszyński and R., 2009)

But if careful about continuity, boundedness, etc., then can guarantee ergodicity (Richardson, Bottolo, R., Valencia 9, 2010).

Regional Adaptive Metropolis Algorithm (RAMA)

Partition $\mathcal{X} = \mathcal{X}_1 \dot{\cup} \ldots \dot{\cup} \mathcal{X}_m$.

Run Metropolis-Hastings algorithm with proposal $Q(x, \cdot) = N(x, \exp(2\, a_i))$ whenever $x \in \mathcal{X}_i$.

Adapt each a_i (diminishingly) based on the batch acceptance rate of those proposals from within $\mathcal{X}_i$.

Simulations: efficiently finds good values of the a_i.

How to choose partition? (manually, so far)

Automate based on acceptance patterns so far?

Instead, can perhaps co-adapt from multiple runs (Craiu, R., Yang, JASA 2009).

To Log or Not To Log

If $\pi(\cdot)$ has heavy tails (e.g. Cauchy), then RWM converges slowly, e.g. not geometrically ergodic (Mengersen and Tweedie, 1996).

May be better to take logarithms, i.e. to sample the distribution of $\log(W)$ where $W \sim \pi(\cdot)$, and then exponentiate the results.

[Even better: replace $\log(W)$ by $\operatorname{sgn}(W)\,\log(1+|W|)$.]

Let the computer decide!

Run both versions, each with adaptively updated proposals.

Every 100 (say) iterations, consider whether to switch from one algorithm to the other, based on which chain shows larger effective average squared jumping distance.

To Log or Not To Log (cont'd)

So how does it work on (simple) examples?

Target $\pi(\cdot)$	Log %	ls_{reg}	ls_{log}
Normal(0,1)	3.62%	2.52	2.08
Uniform[-100,100]	4.95%	6.66	2.65
Cauchy	99.0%	3.49	2.66

Computer figures out which distributions require logs!

Multidimensional: consider log separately for each coord.

“AMCMC”: General Software (Preliminary!)

Want others to use adaptive MCMC, too.

“AMCMC” package (in R and C) at: probability.ca/amcmc

Allows user to specify target density, functional of interest. The package then uses adaptive MCMC to estimate the mean.

R interface (more convenient).

Main software (loops, adaption, etc.) runs in C (faster).

Density and functional can be specified in R or C.

Observation: on “typical” example (baseball data), get speedup factor of about 100 if specify density in C instead of R. (Functional, loops less important.)

[See also “Grapham” by M. Vihola.]

Summary

Adaptive MCMC seems promising ("computer learning") – good.

But must be done carefully, or it will destroy stationarity – bad.

To converge to $\pi(\cdot)$, suffices to have stationarity of each P_γ, plus (a) Diminishing Adaptation (important), and (b) Containment (technical condition, usually satisfied).

Works well in examples like Adaptive Metropolis (200×200 covariance matrix) and Metropolis-within-Gibbs (503 dimensions).

General-purpose software "AMCMC" being developed.

Hopefully can be applied to many other examples – try it!

Papers, applets, software: probability.ca/jeff