10-708: Probabilistic Graphical Models 10-708, Spring 2017

16 : Approximate Inference: Markov Chain Monte Carlo

Lecturer: Eric P. Xing *Scribes: Yuan Yang, Chao-Ming Yen*

1 Introduction

As the target distribution $p(x)$ getting complicated, it becomes practical to approximate the distribution instead of looking for a closed form expression. The Monte Carlo method can generate samples from $p(x)$ to approximate $p(x)$ itself, and allows us to graphical model inference when we can't compute $p(x)$. Nonetheless, there is still limitations for Monte Carlo methods. For example, in direct sampling, we are usually dealing with high-dimensional cases (million-dimension multivariate Gaussian for example), and it's hard to get rare events in such high dimension spaces. On the other hand, in rejection sampling and importance sampling, the Monte Carlo method doesn't work well if the proposal $Q(x)$ is very different from $P(x)$, not to mention it's sometimes hard to construct a suitable $Q(x)$.

Therefore, instead of giving a fixed proposal $Q(x)$, what if we could an **adaptive** proposal?

2 Markov Chain Monte Carlo

2.1 Concept of Markov Chain

A Markov chain is a sequence of random variables that possess the *Markov property*

$$P(x^{(n)} = x|x^{(1)}, ...x^{(n-1)}) = P(x^{(n)} = x|x^{(n-1)}),$$

where $P(x^{(n)}|x^{(n-1)})$ is also known as *transition kernel*. an intuitive explanation of Markov property is that, the next random variable only depends on previous one but other historical random variables. In other word, only transition kernel matter when we take sampling into account. Such property is often called "memoryless" of the sequence of random variables.

In this lecture, we study homogeneous Markov chain where transition kernel $P(x^{(t)}|x^{(t-1)})$ is fixed with time. To simplify, we denote the kernel as $T(x'|x)$, where x represents the previous state and x' represents the next state.

A brief summary for notations in Markov chain:

- Probabilistic distributions over states x at time t: $\pi^{(t)}(x)$.
- Transitions from state $x^{(t)}$ to state $x^{(t+1)}$: $T(x'|x)$. The transitions from all source is $\pi^{(t+1)}(x') = \sum_x \pi^{(t)}(x)T(x'|x)$.
- Stationary distributions: $\pi(x)$ is called stationary if it doesn't change under the transition kernel for all x':

$$\pi(x') = \sum_x \pi(x) T(x'|x) \tag{1}$$

To establish a workable Markov chain, we need to define some notions:

- **Irreducible**: a Markov chain is called irreducible if one can get from any state x to any other state x with probability greater than zero in a finite number of steps.
- **Aperiodic**: a Markov chain is called aperiodic if one can return to any state x at any time.
- **Ergodic**:a Markov chain is called ergodic if it is both irreducible and aperiodic.
- **Reversible (detailed balance)**: a Markov chain is called reversible if there exists a distribution $\pi(x)$ such that $\pi(x')T(x|x') = \pi(x)T(x'|x)$. It's worth notice that an reversible MC will always have a stationary distribution, since:

$$\begin{aligned}
\pi(x')T(x|x') &= \pi(x)T(x'|x) \\
\sum_x \pi(x')T(x|x') &= \sum_x \pi(x)T(x'|x) \\
\pi(x') \sum_x T(x|x') &= \sum_x \pi(x)T(x'|x) \\
\pi(x') &= \sum_x \pi(x)T(x'|x)
\end{aligned}$$

For an ergodic Markov chain, it is guaranteed that the stationary distribution can be reached from any given initial distribution. In other words, the ergodicity makes sure that a Markov chain can eventually converge.

2.2 Metropolis-Hastings Algorithm

The application of Markov chain in Monte Carlo method can be demonstrated with the Metropolis-Hastings algorithm. Here, $Q(x'|x)$ is used instead of $Q(x')$, which features adaptive proposals compared to the original non-Markov chain Monte Carlo.

The processes Metropolis-Hastings Algorithm are shown as following:

1. (Randomly) initialize starting state $x^{(0)}$, and set $t = 0$.
2. Draw a sample x' from $Q(x'|x)$.
3. Compute $A(x'|x) = min(1, \frac{P(x')Q(x|x')}{P(x)Q(x'|x)})$. Notice that it's literally comparing the ratios of true probability over proposal probability between state x and x', or $\frac{P(x')/Q(x'|x)}{P(x)/Q(x|x')}$, where $P(x')/Q(x'|x)$ is the importance weight for x' and vice versa.
4. sample u from Uniform(0,1) to decide acceptance threshold.
5. if $u < A(x'|x)$ then do transition ($x^{(t)} = x'$), otherwise stay in current state ($x^{(t)} = x$).
6. repeat step 2 - 5 (burn-in phase) until sample converges.
7. Take sample from $P(x)$. Rest $t = 0$, and update $x^{(t+1)}$.

2.3 Why does Metropolis-Hastings Algorithm work?

When we draw a sample x' given $Q(x'|x)$, the transition kernel is $T(x'|x) = Q(x'|x)A(x'|x)$. In Metropolis-Hastings Algorithm, we compute the ratio of importance weight where $A(x'|x) = min(1, \frac{P(x')Q(x|x')}{P(x)Q(x'|x)})$. Suppose $A(x'|x) < 1$ and $A(x|x') = 1$, we have:

$$\begin{aligned} A(x'|x) = & \frac{P(x')Q(x|x')}{P(x)Q(x'|x)} \\ P(x)Q(x'|x)A(x'|x) = & P(x')Q(x|x') \\ P(x)Q(x'|x)A(x'|x) = & P(x')Q(x|x')A(x|x') \\ P(x)T(x'|x) = & P(x')T(x|x'), \end{aligned}$$

which is exactly the detailed balance condition (See section 2.1). It means that, after some iterations the Metropolis-Hastings Algorithm will lead to a stationary distribution $P(x)$, which is defined to be the true distribution of x. In other word, the Metropolis-Hastings Algorithm will eventually converge to the true distribution, given enough runs.

While the Metropolis-Hastings Algorithm is proved to converge to the true distribution, there is no guarantee that when this will occur. The art of Markov chain Monte Carlo lies knowing when to stop the burn-in period. This will be discussed in section 6.2.

3 Gibbs Sampling

Gibbs sampling is a special case of the Metropolis-Hastings Algorithm, and it samples each random variable of a graphical model, one at a time. In addition, the acceptance rate A is always equal to 1 in Gibbs sampling (detailed discussion is presented in section 5). It performs a random walk where at each iteration the value along a randomly selected dimension is updated according to the conditional distribution. Also, the detailed balance property holds since transition in Gibbs sampling are ergodic if all dimensions are updated with positive probability.

For graphical model with variables $x_1, ..., x_n$, the procedures of Gibbs Sampling:

1. Initializing starting values for $x_1, ..., x_n$
2. Do the following until convergence:
   - Pick an ordering of the n variables (can be random)
   - For each variable x_i, sample x from $P(x_j|x_1, ...x_{j-1}, x_{j+1}, ..., x_n)$ and then update x_i with x. Notice that the update follows immediately for x_i once its new value is obtained for sampling other variables x_j.

   The graphical model is especially suitable for Gibbs sampling since in graphical model,

$$P(x_j|x_1, ...x_{j-1}, x_{j+1}, ..., x_n) = P(x_i|MB(x_i)), \tag{2}$$

   where $MB(x_i)$ denotes Markov Blanket of x_i.

   It's also worth notice that Gibbs sampling is useful when sampling from $P(x'|x)$ is relatively easy. In these cases, each random walk iteration is quick and all proposed values are accepted, which in turn speed up the whole sampling procedure.

4 Topic Models: Collapsed Gibbs

In this section, a more complicated example of applying MCMC is given: inference on topic model (In particular, Latent Dirichlet Allocation (LDA) (Blei et al., 2003)) using Collapsed Gibbs sampling (Griffiths and Steyvers, 2004). We first give a brief introduction to LDA model.

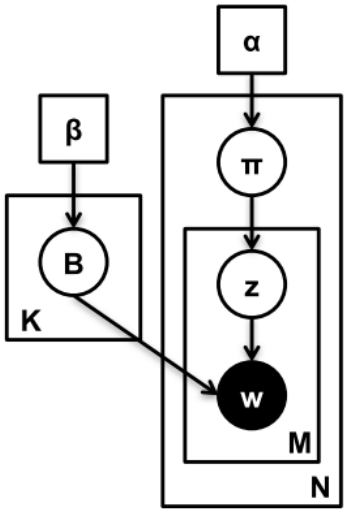


Figure 1: A graphical representation of LDA model. A different notation is used for Γ (as B) and Φ (as π).

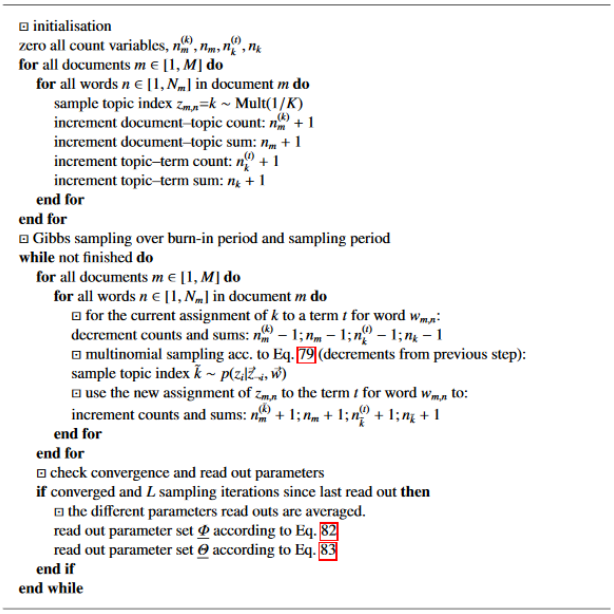
As illustrated in Fig.1, LDA is a probabilistic graphical model for topic discovery in text documents. It describes how the words in documents are explained by a set of K topics in a generative procedure. Each of the topic is represented as a V-dimensional multinomial distribution φ_k, where V is the vocabulary size, and is referred to a topic-word distribution. The topics are often assumed to follow a conjugate Dirichlet prior, that is, $\varphi_k \sim Dir(\beta)$, with hyperparameter β. For each document $\mathbf{w}_d$ that contains N_d tokens, where each token in it is denoted as w_{dn}, a K-dimensional topic mixing distribution γ_d is sampled from a Dirichlet prior $Dir(\alpha)$. Then, for each token, a topic assignment z_{dn} is sampled from a multinomial distribution $z_{dn} \sim \text{Multi}(\gamma_d)$, followed by sampling the token itself again from a multinomial distribution $w_{dn} \sim \text{Multi}(\varphi_{z_{dn}})$. The matrix that contains all φ_k is denoted as Φ, and the one that contains all γ_d is Γ.

Inference over LDA is to determine the posterior distribution of all topic assignment $P(\mathbf{z}|\mathbf{w}, \alpha, \beta)$, where the Φ and Γ are integrated out. A classical method for performing MCMC sampling on this marginalized posterior is collapsed Gibbs sampling Griffiths and Steyvers (2004). For each z_{dn} we sample a new topic using

$$P(z_{dn} = k|rest) \propto \frac{(\alpha + n_{dk}^{-z_{dn}})(\beta + n_{kw}^{-z_{dn}})}{\sum_k \beta + n_{kw}^{-z_{dn}}} \tag{3}$$

where n_{dk} denotes the number of times that topic k is assigned to document d, n_{kw} denotes the number of times that topic k is assigned to word w, superscript $-z_{dn}$ denotes the counts matrix without z_{dn}, and we omit the condition $\mathbf{w}$ for simplicity. The term $\sum_k \alpha + n_{dk}^{-z_{dn}}$ in denominator is also neglected since it's a constant.

Operationally, we first initiate $z_d n$ with random topic assignments and then constructs its corresponding counts matrices n_{dk} and n_{kw}. Then we scan through all the documents, for each token w we sample new $z_d n$ by first "popping out" the old $z_d n$ from the two counts matrices and then use Eq.(3) to draw new sample and increment the corresponding field in the matrices. A sweep through all tokens is referred to as an epoch and usually a few dozens of epochs are needed in order to converge. The last sample of $\bar{\mathbf{z}}$ is used to estimate



⊡ initialisation
zero all count variables, $n_m^{(k)}, n_m, n_k^{(t)}, n_k$
for all documents $m \in [1, M]$ **do**
  for all words $n \in [1, N_m]$ in document m **do**
    sample topic index $z_{m,n}=k \sim \text{Mult}(1/K)$
    increment document–topic count: $n_m^{(k)} + 1$
    increment document–topic sum: $n_m + 1$
    increment topic–term count: $n_k^{(t)} + 1$
    increment topic–term sum: $n_k + 1$
  end for
end for
⊡ Gibbs sampling over burn-in period and sampling period
while not finished **do**
  for all documents $m \in [1, M]$ **do**
    for all words $n \in [1, N_m]$ in document m **do**
      ⊡ for the current assignment of k to a term t for word $w_{m,n}$:
      decrement counts and sums: $n_m^{(k)} - 1; n_m - 1; n_k^{(t)} - 1; n_k - 1$
      ⊡ multinomial sampling acc. to Eq. 79 (decrements from previous step):
      sample topic index $\tilde{k} \sim p(z_i | \vec{z}_{-i}, \vec{w})$
      ⊡ use the new assignment of $z_{m,n}$ to the term t for word $w_{m,n}$ to:
      increment counts and sums: $n_m^{(\tilde{k})} + 1; n_m + 1; n_{\tilde{k}}^{(t)} + 1; n_{\tilde{k}} + 1$
    end for
  end for
  ⊡ check convergence and read out parameters
  if converged and L sampling iterations since last read out **then**
    ⊡ the different parameters read outs are averaged.
    read out parameter set $\underline{\Phi}$ according to Eq. 82
    read out parameter set $\underline{\Theta}$ according to Eq. 83
  end if
end while

Figure 2: Pseudo code implementation for LDA Gibbs sampling (Heinrich, 2008). Here Eq.29 corresponds to Eq.(3), and Eq.82, Eq.83 refers to Eq.(4).

Φ and Γ with following equations

$$\begin{aligned} \hat{\varphi}_{kw} &= \frac{n_{kw}}{\sum_w n_{kw}} \\ \hat{\gamma}_{dk} &= \frac{n_{dk}}{\sum_k n_{dk}}. \end{aligned} \tag{4}$$

In (Heinrich, 2008) a nice pseudo code implementation is presented, we paste it here in Fig.2 for readers' reference.

5 Gibbs sampling as a special case of MH

In this section, we show that the Gibbs sampling is in fact a special case of MH with acceptance rate equal to 1. By definition the proposal distribution $Q(\mathbf{x}^*|\mathbf{x})$ of Gibbs sampling is

$$Q(x^*, rest|x, rest) = P(x^*|rest).$$

Then substitute this into the definition of acceptance rate $A(x^*, rest|x, rest)$, where we have

$$\begin{aligned} A(x^*, rest|x, rest) &= min\left(1, \frac{P(x^*, rest)Q(x, rest|x^*, rest)}{P(x, rest)Q(x^*, rest|x, rest)}\right) \\ &= min\left(1, \frac{P(x^*, rest)P(x|rest)}{P(x, rest)P(x^*|rest)}\right) \\ &= min\left(1, \frac{P(x^*|rest)P(rest)P(x|rest)}{P(x|rest)P(rest)P(x^*|rest)}\right) \\ &= 1, \end{aligned}$$

which proves our claim.

6 Practical Aspects of MCMC

This section turns to some practical discussions about general MCMC method such as MH. We discuss two issues: how to know if proposal is good or not; and how to know when to stop burn-in, and provide two solutions for each of them respectively.

6.1 Determine if proposal is good

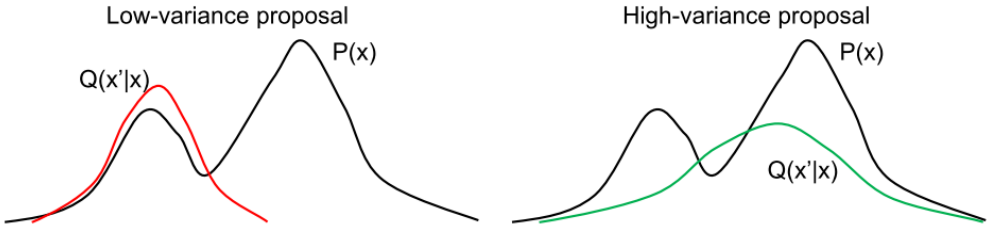


Figure 3: An illustration of different strategies of choosing proposal and its effect with respect to the target distribution.

Choosing the proposal $Q(x^*|x)$ is a trade-off between acceptance rate and exploration. As shown in Fig.3, a proposal with low variance may tend to have a high acceptance rate, since it sticks around the high probability regions and takes a long time to fully explore. However a wide proposal, i.e. distribution with high variance is more capable in exploring the target distribution, but also with a higher reject rate, which leads to high computational costs.

Monitor acceptance rate: One way to determine if the proposal is good or not is to monitor the acceptance rate. A general guideline of choosing proposal is to reach maintain an acceptance rate around 0.5, an intuitive trade-off point for computational convenience and exploration (Müller, 1991).

However, as suggested in (Roberts et al., 1997), for the case where target and proposal distributions are all Gaussian, the optimal acceptance rate reached with dimensionality equals to 1 is only around 0.45 and this rate decreases to only around 0.23 as the dimensionality of the random variable increases. This means MH method is essentially not very efficient.

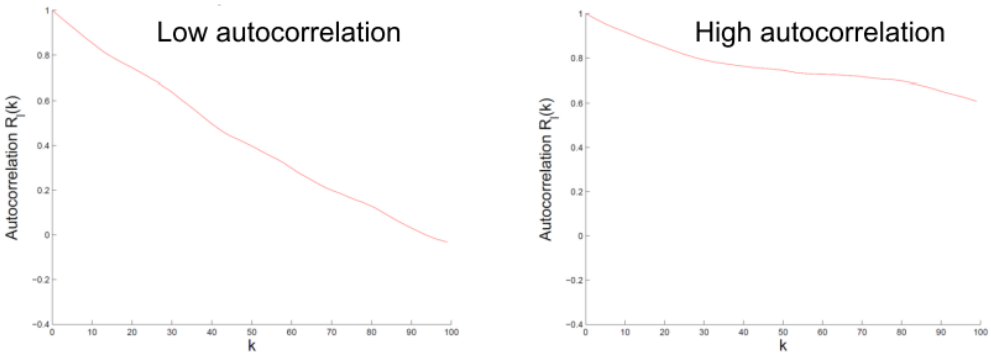


Figure 4: An illustration of how autocorrelation changes as the number of steps increases.

Monitor autocorrelation: Another way to evaluate the proposal function is to check the correlation between samples: high correlations between samples can reduce the effective number of samples, since our

goal of running MCMC is to simulate drawing i.i.d samples directly from the target distribution. Thus samples from a good proposal show exhibits low correlations. This can be evaluated by monitoring the autocorrelation function.

Formally, autocorrelation (AC) is defined as

$$R_x(k) = \frac{\sum_{t=1}^{n-k}(x_t - \bar{x})(x_{t+k} - \bar{x})}{\sum_{t=1}^{n-k}(x_t - \bar{x})},$$

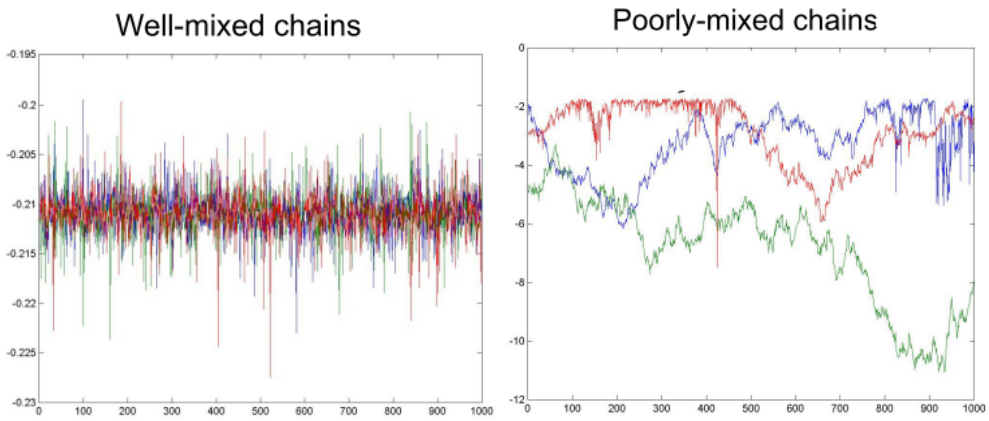
where k indicates the order of the AC which is usually set to be 1. As shown in Fig.4, AC should decrease as k increases, and the general slope of the curve encodes the degree of correlation between samples: the bigger of the slope, the lower of the correlation. The notion of effective sample size we mentioned above can be quantified using Sample Size Inflation Factor (SSIF)

$$s_x = \frac{1 + R_x(1)}{1 - R_x(1)},$$

where $R_x(1)$ is the first-order AC. If one draws n sample with SSIF s_x, then the effective sample size is $\frac{n}{s_x}$. This, again, indicates that a high autocorrelation leads to smaller effective sample size.

6.2 Determine when to stop burn-in

Another issue is to determine when to halt burn-in. This is an art which is, to some extents, similar to when to stop in a gradient based optimization problem.



(a) (b)

Figure 5: An illustration of visualizing the sample value as the function of time.

Monitor sample value against time: One way is to monitor the sample value as the function of time, as shown in Fig.5. Operationally, one runs multiple MCMC chains and pick one scalar value from the random variable to monitor, if the spectrum that the value varies are similar for all chains, we call those chains to be "well-mixed", otherwise "poorly-mixed". Well-mixed chains are the sign of convergence, where one shall consider halting the burn-in.

Monitor log likelihood against time: Many graphical models are high dimensional and is difficult to visualize each value to check convergence. Thus a more convenient way is to check the complete log likelihood as the function of time, as shown in Fig.6. Similar to the error rate or loss function used in gradient based

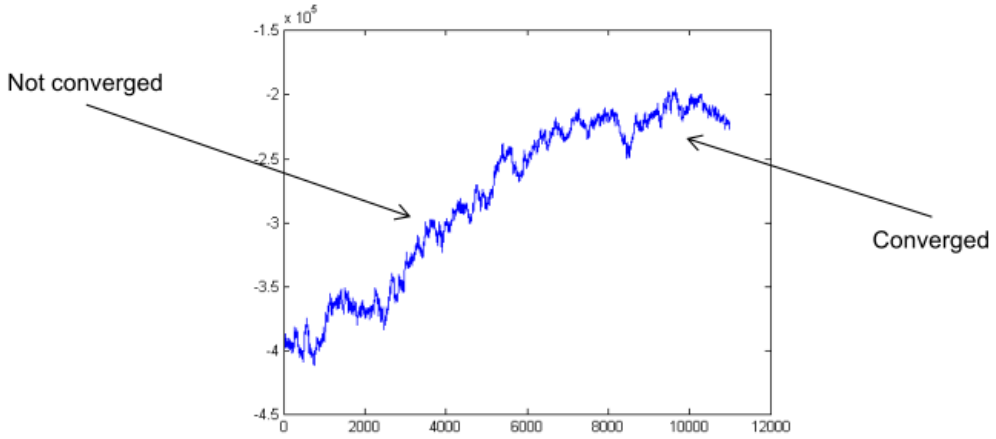


Figure 6: An illustration of how log likelihood increases as the function of time (or number of MCMC steps).

optimization procedure, the complete log likelihood will climb during the process and finally converge, where one can considering halting burn-in.

7 Summary

In this lecture, we conclude:

- MCMC is a numerical approximation method that makes use of an adaptive proposal to simulate drawing samples from the unknown true distribution.
- MH method, one of the main workhorses in MCMC families, allows one to specify arbitrary proposal, but requires a careful design of the proposal to work effectively.
- Gibbs sampling is the special case of MH with acceptance rate equal to 1. It uses the full conditional distribution of the true distribution as its proposal. It is computationally efficient, but is therefore slow in exploration, and is not always the best choice for every model.
- Determining when to halt burn-in is an art. Empirical methods include monitoring the mixing rate or the log likelihood as the function of time.

References

D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

T. L. Griffiths and M. Steyvers. Finding scientific topics. *Proceedings of the National academy of Sciences*, 101(suppl 1):5228–5235, 2004.

G. Heinrich. Parameter estimation for text analysis. *University of Leipzig, Tech. Rep*, 2008.

P. Müller. *A generic approach to posterior integration and Gibbs sampling*. Purdue University, Department of Statistics, 1991.

G. O. Roberts, A. Gelman, W. R. Gilks, et al. Weak convergence and optimal scaling of random walk metropolis algorithms. *The annals of applied probability*, 7(1):110–120, 1997.