BAYESIAN MODEL FITTING AND MCMC

A6523

Robert Wharton
Apr 18, 2017
Outline

• Bayesian Inference

• MCMC Sampling
  • Basic Idea
  • Examples

• A Pulsar Example
Information Theory, Inference, and Learning Algorithms

David J.C. MacKay
mackay@mrao.cam.ac.uk

©Cambridge University Press 2003

Especially recommended

Bayesian Inference
Bayesian Inference

When should I use Bayesian methods?

• Parameter estimation of non-linear models

• Want / need full posterior

• All the time

When can I just use least squares?

• Parameter estimation of linear models

• Only need covariances
Bayesian Inference

\[ p(\theta|D,I) = \frac{p(D|\theta,I) \ p(\theta,I)}{p(D|I)} \]

Posterior = \frac{\text{Likelihood} \times \text{Prior}}{\text{Evidence}}
Bayesian Inference

\[ p(D|\theta, \mathcal{I}) = p(d_1, \ldots, d_N|\theta, \mathcal{I}) \]

\[ p(D|\theta, \mathcal{I}) = \prod_{i=1}^{N} p(d_i|\theta, \mathcal{I}) \]
Bayesian Inference

\[
p(d_i | \theta, \mathcal{I}) = \frac{1}{\sqrt{2\pi \sigma_i^2}} \exp \left[ -\frac{1}{2} \left( \frac{d_i - f(\theta)}{\sigma_i} \right)^2 \right]
\]

\[
p(D | \theta, \mathcal{I}) = (2\pi \sigma^2)^{-N/2} \exp \left[ -\frac{1}{2} \sum_{i=1}^{N} \left( \frac{d_i - f(\theta)}{\sigma_i} \right)^2 \right]
\]
Bayesian Inference

\[ p(\theta|D, \mathcal{I}) = \frac{p(D|\theta, \mathcal{I}) \ p(\theta, \mathcal{I})}{p(D|\mathcal{I})} \]

\[ p(\theta|D, \mathcal{I}) \propto p(D|\theta, \mathcal{I}) \ p(\theta, \mathcal{I}) \]
How do we evaluate the posterior?

• Just do the integral

• Taylor Expand and Approximate

• MCMC
How do we evaluate the posterior?

• Just do the integral
• Taylor Expand and Approximate
• MCMC
How do we evaluate the posterior?

• Just do the integral

• Taylor Expand and Approximate

• MCMC
The idea behind the Laplace approximation is simple. We assume that an unnormalized probability density $P^*(x)$, whose normalizing constant

$$Z_P \equiv \int P^*(x) \, dx$$  \hspace{1cm} (27.1)$$

is of interest, has a peak at a point $x_0$. We Taylor-expand the logarithm of $P^*(x)$ around this peak:

$$\ln P^*(x) \simeq \ln P^*(x_0) - \frac{c}{2}(x - x_0)^2 + \cdots,$$  \hspace{1cm} (27.2)$$

where

$$c = - \left. \frac{\partial^2}{\partial x^2} \ln P^*(x) \right|_{x=x_0}.$$  \hspace{1cm} (27.3)$$

We then approximate $P^*(x)$ by an unnormalized Gaussian,

$$Q^*(x) \equiv P^*(x_0) \exp \left[ -\frac{c}{2}(x - x_0)^2 \right],$$  \hspace{1cm} (27.4)$$

and we approximate the normalizing constant $Z_P$ by the normalizing constant of this Gaussian,

$$Z_Q = P^*(x_0) \sqrt{\frac{2\pi}{c}}.$$  \hspace{1cm} (27.5)$$
Laplace’s Method (M27)

The idea behind the Laplace approximation is simple. Consider the unnormalized probability density \( P^*(x) \), whose normalized counterpart is of interest, has a peak at a point \( x_0 \). We Taylor-expand \( P^*(x) \) around this peak:

\[
\ln P^*(x) \simeq \ln P^*(x_0) - \frac{c}{2}(x - x_0)^2
\]

where

\[
c = -\frac{\partial^2}{\partial x^2} \ln P^*(x) \bigg|_{x=x_0}
\]

We then approximate \( P^*(x) \) by an unnormalized Gaussian:

\[
Q^*(x) \equiv P^*(x_0) \exp \left[-\frac{c}{2}(x - x_0)^2\right]
\]

and we approximate the normalizing constant \( Z_P \) by the integral of this Gaussian,

\[
Z_Q = P^*(x_0) \sqrt{\frac{2\pi}{c}}.
\]
How do we evaluate the posterior?

- Just do the integral
- Taylor Expand and Approximate
- MCMC
Monte Carlo Sampling
Monte Carlo
29.1 The problems to be solved

Monte Carlo methods are computational techniques that make use of random numbers. The aims of Monte Carlo methods are to solve one or both of the following problems.

**Problem 1:** to generate samples \( \{ \mathbf{x}^{(r)} \}_{r=1}^{R} \) from a given probability distribution \( P(\mathbf{x}) \).

**Problem 2:** to estimate expectations of functions under this distribution, for example

\[
\Phi = \langle \phi(\mathbf{x}) \rangle \equiv \int d^{N}\mathbf{x} \ P(\mathbf{x})\phi(\mathbf{x}). \tag{29.3}
\]
Why is sampling from $P(x)$ hard?

We will assume that the density from which we wish to draw samples, $P(x)$, can be evaluated, at least to within a multiplicative constant; that is, we can evaluate a function $P^*(x)$ such that

$$P(x) = \frac{P^*(x)}{Z}. \quad (29.8)$$
Why is Sampling Hard?

Why is sampling from $P(x)$ hard?

We will assume that the density from which we wish to draw samples, $P(x)$, can be evaluated, at least to within a multiplicative constant; that is, we can evaluate a function $P^*(x)$ such that

$$P(x) = \frac{P^*(x)}{Z}. \tag{29.8}$$

Two main reasons:

1. We don’t know normalization

2. Need to enumerate most (or all) possible states in high dimensional data
Types of Sampling

Consider a few types of sampling methods:

1. Rejection
2. Metropolis
3. Gibbs
4. Slice
5. Hamiltonian
Consider a few types of sampling methods:

1. Rejection
2. Metropolis
3. Gibbs
4. Slice
5. Hamiltonian
29.3 Rejection sampling

We assume again a one-dimensional density \( P(x) = P^*(x)/Z \) that is too complicated a function for us to be able to sample from it directly. We assume that we have a simpler *proposal density* \( Q(x) \) which we can evaluate (within a multiplicative factor \( Z_Q \), as before), and from which we can generate samples. We further assume that we know the value of a constant \( c \) such that

\[
cQ^*(x) > P^*(x), \quad \text{for all } x. \tag{29.29}
\]
Rejection sampling in many dimensions

In a high-dimensional problem it is very likely that the requirement that $c Q^*$ be an upper bound for $P^*$ will force $c$ to be so huge that acceptances will be very rare indeed. Finding such a value of $c$ may be difficult too, since in many problems we know neither where the modes of $P^*$ are located nor how high they are.

$$c = \frac{(2\pi \sigma_Q^2)^{N/2}}{(2\pi \sigma_P^2)^{N/2}} = \exp \left( N \ln \frac{\sigma_Q}{\sigma_P} \right).$$

With $N = 1000$ and $\frac{\sigma_Q}{\sigma_P} = 1.01$, we find $c = \exp(10) \approx 20,000$. 
Consider a few types of sampling methods:

1. Rejection
2. Metropolis
3. Gibbs
4. Slice
5. Hamiltonian
Hard to find a single good proposal distribution $Q(x)$ for large problems

The Metropolis-Hastings method uses a proposal density $Q$ that depends on the current state $x(t)$.
As before, we assume that we can evaluate $P^*(x)$ for any $x$. A tentative new state $x'$ is generated from the proposal density $Q(x'; x^{(t)})$. To decide whether to accept the new state, we compute the quantity

$$a = \frac{P^*(x')}{{P^*(x^{(t)})}} \cdot \frac{Q(x^{(t)}; x')}{{Q(x'; x^{(t)})}}.$$  \hspace{1cm} (29.31)

If $a \geq 1$ then the new state is accepted. Otherwise, the new state is accepted with probability $a$.

If the step is accepted, we set $x^{(t+1)} = x'$.
If the step is rejected, then we set $x^{(t+1)} = x^{(t)}$. 
Convergence of the Metropolis method to the target density

It can be shown that for any positive $Q$ (that is, any $Q$ such that $Q(x';x) > 0$ for all $x, x'$), as $t \to \infty$, the probability distribution of $x^{(t)}$ tends to $P(x) = P^*(x)/Z$. [This statement should not be seen as implying that $Q$ has to assign positive probability to every point $x'$ – we will discuss examples later where $Q(x';x) = 0$ for some $x, x'$; notice also that we have said nothing about how rapidly the convergence to $P(x)$ takes place.]
The Metropolis method is an example of a Markov chain Monte Carlo method (abbreviated MCMC). In contrast to rejection sampling, where the accepted points \( \{ x^{(r)} \} \) are independent samples from the desired distribution, Markov chain Monte Carlo methods involve a Markov process in which a sequence of states \( \{ x^{(t)} \} \) is generated, each sample \( x^{(t)} \) having a probability distribution that depends on the previous value, \( x^{(t-1)} \). Since successive samples are dependent, the Markov chain may have to be run for a considerable time in order to generate samples that are effectively independent samples from \( P \).
Rule of thumb: lower bound on number of iterations of a Metropolis method. If the largest length scale of the space of probable states is $L$, a Metropolis method whose proposal distribution generates a random walk with step size $\epsilon$ must be run for at least

$$T \simeq (L/\epsilon)^2$$

(29.32)

iterations to obtain an independent sample.
Consider a few types of sampling methods:

1. Rejection
2. Metropolis
3. Gibbs
4. Slice
5. Hamiltonian
Like Metropolis, but proposal distribution $Q$ is defined in terms of the conditional joint distribution $P$

\[ x_1^{(t+1)} \sim P(x_1 | x_2^{(t)}, x_3^{(t)}, \ldots, x_K^{(t)}) \]
\[ x_2^{(t+1)} \sim P(x_2 | x_1^{(t+1)}, x_3^{(t)}, \ldots, x_K^{(t)}) \]
\[ x_3^{(t+1)} \sim P(x_3 | x_1^{(t+1)}, x_2^{(t+1)}, \ldots, x_K^{(t)}) \], etc.
Gibbs Sampling

(a) $P(x)$

(b) $P(x_1 | x_2^{(t)})$

(c) $P(x_2 | x_1)$

(d) $P(x_1 | x_2^{(t+1)})$
Convergence of Gibbs sampling to the target density

Exercise 29.4. [2] Show that a single variable-update of Gibbs sampling can be viewed as a Metropolis method with target density $P(x)$, and that this Metropolis method has the property that every proposal is always accepted.
Convergence of Gibbs sampling to the target density

Exercise 29.4. [2] Show that a single variable-update of Gibbs sampling can be viewed as a Metropolis method with target density $P(x)$, and that this Metropolis method has the property that every proposal is always accepted.

$$p(x_j | x_1, \ldots, x_{j-1}, x_{j+1}, \ldots, x_k) = \frac{p(x_1, \ldots, x_k)}{p(x_1, \ldots, x_{j-1}, x_{j+1}, \ldots, x_k)}$$

$\propto p(x_1, \ldots, x_k)$

Because Gibbs sampling is a Metropolis method, the probability distribution of $x^{(t)}$ tends to $P(x)$ as $t \to \infty$, as long as $P(x)$ does not have pathological properties.
Consider a few types of sampling methods:

1. Rejection
2. Metropolis
3. Gibbs
4. Slice
5. Hamiltonian
Slice Sampling
Consider a few types of sampling methods:

1. Rejection
2. Metropolis
3. Gibbs
4. Slice
5. Hamiltonian
30.1 Hamiltonian Monte Carlo

The Hamiltonian Monte Carlo method is a Metropolis method, applicable to continuous state spaces, that makes use of gradient information to reduce random walk behaviour. [The Hamiltonian Monte Carlo method was originally called hybrid Monte Carlo, for historical reasons.]

For many systems whose probability $P(x)$ can be written in the form

$$P(x) = \frac{e^{-E(x)}}{Z},$$

not only $E(x)$ but also its gradient with respect to $x$ can be readily evaluated. It seems wasteful to use a simple random-walk Metropolis method when this gradient is available – the gradient indicates which direction one should go in to find states that have higher probability!
Hamiltonian Monte Carlo

Overview of Hamiltonian Monte Carlo

In the Hamiltonian Monte Carlo method, the state space \( x \) is augmented by \textit{momentum variables} \( p \), and there is an alternation of two types of proposal. The first proposal randomizes the momentum variable, leaving the state \( x \) unchanged. The second proposal changes both \( x \) and \( p \) using simulated Hamiltonian dynamics as defined by the Hamiltonian

\[
H(x, p) = E(x) + K(p),
\]

where \( K(p) \) is a ‘kinetic energy’ such as \( K(p) = p^\top p / 2 \). These two proposals are used to create (asymptotically) samples from the joint density

\[
P_H(x, p) = \frac{1}{Z_H} \exp[-H(x, p)] = \frac{1}{Z_H} \exp[-E(x)] \exp[-K(p)].
\]

This density is separable, so the marginal distribution of \( x \) is the desired distribution \( \exp[-E(x)] / Z \). So, simply discarding the momentum variables, we obtain a sequence of samples \( \{x^{(\ell)}\} \) that asymptotically come from \( P(x) \).
Details of Hamiltonian Monte Carlo

The first proposal, which can be viewed as a Gibbs sampling update, draws a new momentum from the Gaussian density \( \exp[-K(p)]/Z_K \). This proposal is always accepted. During the second, dynamical proposal, the momentum variable determines where the state \( x \) goes, and the gradient of \( E(x) \) determines how the momentum \( p \) changes, in accordance with the equations

\[
\dot{x} = p \\
\dot{p} = -\frac{\partial E(x)}{\partial x}.
\]

(30.4)  (30.5)

Because of the persistent motion of \( x \) in the direction of the momentum \( p \) during each dynamical proposal, the state of the system tends to move a distance that goes linearly with the computer time, rather than as the square root.

The second proposal is accepted in accordance with the Metropolis rule. If the simulation of the Hamiltonian dynamics is numerically perfect then the proposals are accepted every time, because the total energy \( H(x, p) \) is a constant of the motion and so \( a \) in equation (29.31) is equal to one. If the simulation is imperfect, because of finite step sizes for example, then some of the dynamical proposals will be rejected. The rejection rule makes use of the change in \( H(x, p) \), which is zero if the simulation is perfect. The occasional rejections ensure that, asymptotically, we obtain samples \( (x^{(t)}, p^{(t)}) \) from the required joint density \( P_H(x, p) \).
How many samples are needed?

At the start of this chapter, we observed that the variance of an estimator $\hat{\Phi}$ depends only on the number of independent samples $R$ and the value of

$$\sigma^2 = \int d^N x \, P(x)(\phi(x) - \Phi)^2.$$  \hspace{1cm} (29.48)

We have now discussed a variety of methods for generating samples from $P(x)$. How many independent samples $R$ should we aim for?
How many samples are needed?

At the start of this chapter, we observed that the variance of an estimator $\hat{\Phi}$ depends only on the number of independent samples $R$ and the value of

\[ \sigma^2 = \int d^N x \ P(x) (\phi(x) - \Phi)^2. \]  

(29.48)

We have now discussed a variety of methods for generating samples from $P(x)$. How many independent samples $R$ should we aim for?

There is little point in knowing $\Phi$ to a precision finer than about $\sigma/3$. After all, the true cost is likely to differ by $\pm \sigma$ from $\Phi$. If we obtain $R = 12$ independent samples from $P(x)$, we can estimate $\Phi$ to a precision of $\sigma/\sqrt{12}$—which is smaller than $\sigma/3$. So twelve samples suffice.
Located very near (in projection, at least) from Sgr A*

Pulsars can be used to measure the distribution of electrons along the line of sight (LOS) to the GC

Some things we can test:

- What is the dispersion measure?
- Does dispersion follow $1/f^2$ law?
- What is the amount of scattering?
Dispersion

Frequency-dependent delay caused by propagation through the cold electron plasma of the ISM

Delays given by:

$$\tau_{\text{DM}}(\nu) = k_{\text{DM}} \cdot \text{DM} \cdot \nu_{\text{GHz}}^{-2}$$

where

$$\text{DM} = \int_0^L n_e \, d\ell$$
Scattering

Frequency-dependent change to pulse shape caused by multi-path propagation in the ISM

Broadening time given by:

\[ \tau_{sc} \propto \nu^{-\frac{4}{GHz}} \]
What’s is our data?

Calculate the time of arrival (TOA) for the pulse in each frequency channel

Use matched filtering
What’s is our data?

Calculate the time of arrival (TOA) for the pulse in each frequency channel.

Use matched filtering.
What's our model?

Model for the frequency dependent TOAs:

$$\tau(\nu) = k_{DM}DM \left(\frac{\nu}{1 \text{ GHz}}\right)^{-2} + \tau_{sc} \left(\frac{\nu}{1 \text{ GHz}}\right)^{-4} + t_0$$

Three free parameters:

$$DM, \tau_{sc}, t_0$$
Constructing Posterior

\[ p(\theta|D, \mathcal{I}) = \frac{p(D|\theta, \mathcal{I}) \ p(\theta, \mathcal{I})}{p(D|\mathcal{I})} \]

\[ p(\theta|D, \mathcal{I}) \propto p(D|\theta, \mathcal{I}) \ p(\theta, \mathcal{I}) \]
Constructing Posterior

\[ p(\theta | D, I) = \frac{p(D | \theta, I) \ p(\theta, I)}{p(D | I)} \]

\[ p(\theta | D, I) \propto p(D | \theta, I) \ p(\theta, I) \]
Likelihood Function

Assume TOA errors are independent and normally distributed.

\[
\mathcal{L}(\tau|\text{DM}, t_0, \tau_{sc}) = \prod_{i=0}^{N-1} \frac{1}{\sqrt{2\pi\sigma_{\tau,i}^2}} \exp\left[ -\frac{1}{2} \left( \frac{\tau_i - \tau(v_i)}{\sigma_{\tau,i}} \right)^2 \right]
\]
Constructing Posterior

\[ p(\theta | D, I) = \frac{p(D | \theta, I) \ p(\theta, I)}{p(D | I)} \]

\[ p(\theta | D, I) \propto p(D | \theta, I) \ p(\theta, I) \]
Priors shouldn’t matter that much.

\[ \text{DM} \sim \mathcal{N}(\mu = 2000 \text{ pc cm}^{-3}, \sigma = 1000 \text{ pc cm}^{-3}) \]

\[ t_0 \sim \mathcal{N}(\mu = 0 \text{ ms}, \sigma = 500 \text{ ms}) \]

\[ \tau_{sc} \sim \exp(\lambda = 1000 \text{ ms}) \]
emcee: The MCMC Hammer

Daniel Foreman-Mackey\textsuperscript{1,2}, David W. Hogg\textsuperscript{2,3}, Dustin Lang\textsuperscript{4,5}, Jonathan Goodman\textsuperscript{6}

ABSTRACT

We introduce a stable, well tested Python implementation of the affine-invariant ensemble sampler for Markov chain Monte Carlo (MCMC) proposed by Goodman & Weare (2010). The code is open source and has already been used in several published projects in the astrophysics literature. The algorithm behind emcee has several advantages over traditional MCMC sampling methods and it has excellent performance as measured by the autocorrelation time (or function calls per independent sample). One major advantage of the algorithm is that it requires hand-tuning of only 1 or 2 parameters compared to $\sim N^2$ for a traditional algorithm in an $N$-dimensional parameter space. In this document, we describe the algorithm and the details of our implementation. Exploiting the parallelism of the ensemble method, emcee permits any user to take advantage of multiple CPU cores without extra effort. The code is available online at http://dan.iel.fm/emcee under the MIT License.
Multiple “walkers” generating samples at same time

Affine-invariant shift/scale step quickly samples the space

Metropolis-Hastings accept/reject

Let’s try N=50 walkers
500 steps
1,000 steps
10,000 steps
25,000 steps
20,000 steps burn-in
Results
Results

![Graph showing the relationship between delay (ms) and frequency (GHz). The graph depicts a downward trend as frequency increases.]
Results

De-dispersed Time Series

DM = 1763.3 pc cm\(^{-3}\)

\(t_0 = -91.20\) ms

\(\tau_{sc} = 417.5\) ms