Notes for Daniel Foreman-Mackey et al. "emcee: The MCMC Hammer"

Malte Brinch
University of Copenhagen
Septemper 2019

1. Introduction

We can use Bayesian data analysis is that it is possible to marginalize over nuisance parameters. A nuisance parameter is one that is required in order to model the process that generates the data, but is otherwise of little interest. Marginalization is the process of integrating over all possible values of the parameter and hence propagating the effects of uncertainty about its value into the final result. For a model we wish to obtain the marginalized probability function $p(\Theta|D)$ of the set (list or vector) of model parameters Θ given the set of observations D

$$p(\Theta|D) = \int p(\Theta, \alpha|D) d\alpha$$

where α is the set nuisance parameters. For large α this is very difficult to do analytically, but with a MCMC sampling can generate values (Θ_t, α_t) which come from the joint distribution $p(\Theta, \alpha|D)$ will automatically provide a sampling of model values Θ_t which come from the marginalized probability density function (PDF) $p(\Theta|D)$. An other important function of MCMC is how it can be be used sample the posterior PDF (Bayesian analysis).

traditional Metropolis-Hastings methods have N(N+1)/2 tuning parameters where N is the dimension of the parameter space (since each term in the covariance matrix of this proposal distribution is an unspecified parameter). Furthermore these samplers are sensitive to the values of these tuning parameters and it is difficult/computationally expensive (having small burn-in calls) to obtain these optimal tuning parameters.

anisotropic PDF are difficult for traditional MCMC but with a affine transformation the problem can be transform to sampling from an isotropic PDF, this is because an algorithm that is affine invariant performs equally well under all linear transformations; it will therefore be insensitive to covariances among parameters.

2. The Algorithm

The general goal of MCMC algorithms is to draw M samples Θ_i from the posterior probability density

$$p(\Theta, \alpha|D) = \frac{1}{Z} p(\Theta, \alpha) p(D|\Theta, \alpha)$$

where the $p(\Theta, \alpha)$ is the prior and $p(D|\Theta, \alpha)$ is the likelihood function. The normalization Z=p(D) is parameter independent for a given model so we can sample the posterior distribution without knowing Z which is good since it is computationally expensive. With the MCMC samples the marginalized constraints on Θ can be approximated by the histogram of the samples projected into the parameter subspace spanned by Θ. This implies hat the expectation value of a function of the model parameters f(Θ) is

$$< f(\Theta) >= \int p(\Theta|D) f(\Theta) d\Theta \approx \frac{1}{M} \sum_{i=1}^{M} f(\Theta_i)$$

MCMC is a procedure for generating a random walk in the parameter space that, over time, draws a representative set of samples from the distribution. Each point in a Markov chain depends only on the position of the previous step in the chain.

2.1. The Metropolis-Hastings Algorithm

the idea of MH is to

- 1. for a value X(t) we sample a proposed value from the transition distribution Q(Y;X(t))

- 2. accept the proposal with a probability $min(1, \frac{P(Y|D)}{p(X(t)|D)} \frac{Q(X(t);Y)}{Q(Y;X(t))})$ if accepted we have X(t+1)=Y otherwise X(t+1)=X(t)

many algorithm converge faster than MH and faster convergence is good because then less likelihood calculations have to be made for the same accuracy of the result.

2.2. The stretch move

This is a affine-invariant ensemble sampling algorithm which outperforms MH. The samplers evolves an ensemble of walkers and finds the proposal distribution for one walker k based on the current positions of the K − 1 walkers in the complementary ensemble which does not include k. Position in this case means a vector in the N-dimensional, real-valued parameter space. a new position for a walker X_k is found by randomly taking another of the walkers X_j and calculating

$$X_k(t) \longrightarrow Y = X_j + Z[X_k(t) - X_j]$$

where Z is a random variable drawn from a distribution g(Z = z). For a distribution $g(z^{-1}) = zg(z)$ the proposal will be symmetric and the chain will satisfy detailed balance if the proposal is accepted with probability

$$q = min(1, z^{N-1} \frac{p(Y)}{p(X_k(t))})$$

where again N is the dimension of the parameter space. This can be done for each walker in series. The form of g(z) could be

$$f(x) = \begin{cases} \frac{1}{\sqrt{z}}, & \text{if z}\in [\frac{1}{a}, a] \quad (1) \\ 0, & \text{otherwise} \quad (2) \end{cases}$$

here a is scale parameter which seems to be equal to 2.

This algorithm can also be run in parallel though to not break detailed balance this must be done by running two populations of walkers and then using one population to update the other and vice versa

3. Tests

The acceptance fraction a_f is the fraction of proposed steps that are accepted. There is no strict rule for what this fraction should be though a rule of thumb is 0.2-0.5. For the stretch move this rate can be controlled by the parameter a for g(z) which is like controlling the step size.

The autocorrelation time is a direct measure of the number of evaluations of the posterior PDF required to produce independent samples of the target density. In other words how much time does it take before the sampler has "forgotten" where is came from. Shorter times are better as they take less PDF computations. The autocorrelation time is also affine invariant.

emcee can use acor to estimate the autocorrelation time.

4. Discussion and Tips

Write the code yourself you dummy.

Ensemble samplers like emcee require some thought for initialization. One general approach is to start the walkers at a sampling of the prior or spread out over a reasonable range in parameter space. Another general approach is to start the walkers in a very tight N-dimensional ball in parameter space around one point that is expected to be close to the maximum probability point. The first one is more objective, the second one is more effective.

use many walkers (100 or more) and run for a few autocorrelations times 10 or more to get errorbars on a couple of parameters. For multi-modal target densities walkers may become stuck and other samplers like Diffusive Nested Samling is needed.