Ensemble Samplers with Affine Invariance

by Jonathan Goodman AND Jonathan Weare

(Write up)

E. Martiny and A. K. Ravn

March 7, 2018

1 What is the challenge/introduction

When you are trying to sample using Markov chain Monte Carlo, the big challenge is usually to tune the proposal function, that proposes a new step. A simple way of doing this is to take Gaussian steps in each dimension, each with its own value of σ. This can end up in a lot of different meta-parameters to tune, to do your MCMC. Another challenge is that the same step-size/sigma is not necessarily appropriate throughout one dimension. Maybe your distribution is very wide in one area and very peaked in another. This paper tries to deal with both problems, both optimizing the speed of the MCMC and minimizing the number of meta-parameters.
The main idea in the paper is to use an ensemble of walkers to estimate the shape of the distribution you are sampling. The assumption is that after a burn-in period the ensemble represents the distribution, and can therefore be used to determine what the proposal function should look like. The authors presents different proposal functions that is based on an ensemble and shows that these keep the useful properties of standard MCMC, specifically: Ergodicity and the symmetry of the standard proposal function of Metropolis Hastings. They do this partly through the use of affine transformations

1.1 Affine transformation of parameters and affine invariance.

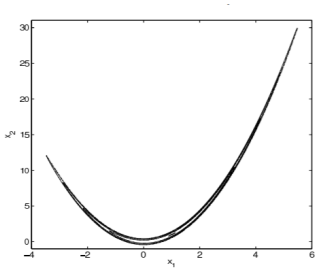
An affine transformation is any transformation on the form $y = Ax + b$ that preserves points, parallel lines and parallel planes. The trick is if you have e.g. a skewed Gaussian on the form $\pi(x) \propto \exp\left(-\frac{(x_1-x_2)^2}{2\sqrt{\epsilon}} - \frac{x_1+x_2}{2}\right)$ a good MCMC sampler would use perturbations of order $\sqrt{\epsilon}$ $(1,-1)$ direction and perturbations of order one in the $(1,1)$ direction. However, if you transform the problem to a parameter space $y_1 = \frac{x_1-x_2}{\sqrt{\epsilon}}$ and $y_2 = x_1 + x_2$ then the new distribution $\pi_A(y) \propto \exp\left(-\frac{y_1+y_2}{2}\right)$, is well scaled, and the complexity of sampling is drastically reduced. A general MCMC sampler on the from $X(t+1) = R(X(t), \xi(t), \pi)$, where $X(t)$ is the sample after t iterations, $\xi(t)$ is a sequence of independent identically distributed random variables, and π is a probability density, is called *affine invariant* if, for any affine transformation $Ax + b$, $R(Ax(t) + b, \xi(t), \pi) = AR(x(t), \xi(t), \pi) + b$. While this property is well desired the authors of the paper are not aware of a practical sampler that has this affine invariance property for any general class of densities. Instead the authors present a family of affine invariant *ensemble* algorithms motivated, in part, by the Nelder–Mead simplex optimization scheme.

2 Proposed alternatives to Metropolis sampling

2.1 Stretch move algorithm

Given an ensemble of walkers, $\vec{X} = \{X_1, \ldots, X_N\}$, the next step for a walker, $X_k(t)$, at time t, is proposed by the schema

$$X_k(t) \to Y = X_j + Z(X_k(t) - X_j), \tag{1}$$



(a) The Rosenbrock density used to test the samplers.

method↓ ensemble size →	$f(x_1,x_2)=x_1$ 1	10	100	∞	$f(x_1,x_2)=x_2$ 1	10	100	∞
Metropolis	163	–	–	–	322	–	–	–
stretch moves	–	19.4	8.06	8.71	–	67.0	18.4	23.5
walk moves, $\|S\|=3$	–	46.4	19.8	18.6	–	68.0	44.2	47.1

(b) Efficiency test for Metropolis, walk-move and stretch-move. Given in autocorrelation time(times 10^{-3}) for the two parameters of the Rosenbrock density

Figure 1: Excepts from the paper

where X_j is one *other* ($j \neq k$) walker in the ensemble of walkers, $\vec{X}$, and Z is a scaling variable that satisfies the density distribution symmetry

$$g\left(\frac{1}{z}\right) = zg(z). \quad \text{where in the paper the authors chooses} \quad g(z) = \begin{cases} \frac{1}{\sqrt{z}} & \text{if in } \left[\frac{1}{a}, a\right] \\ 0 & \text{otherwise} \end{cases}, \tag{2}$$

because it is simple. The parameter $a > 1$ can be adjusted to improve performance. In this way the set of possible proposals lies within the straight line containing both $X_k(t)$ and X_j.

2.2 Walk move algorithm

The walk move begins by choosing a subset of walkers S form the compliment ensemble to X_k, $\vec{X}_{[k]}$. It is necessary that $|S| \geq 2$ and that the choice of S is independent of X_k. The walk move offers a proposal $X_k \to X_k + W$ where W is normal with mean zero and the same covariance as the walkers $X_j \in S$. The empirical distribution of S is

$$\pi_S(x) = \frac{1}{|S|} \sum_{X_j \in S} \delta(x - X_j), \tag{3}$$

and the mean of a random variable drawn form π_S is $\bar{X}_S = \frac{1}{|S|} \sum_{X_j \in S} X_j$. The covariance is $C_S = \frac{1}{|S|} \sum_{X_j \in S} (X_j - \bar{X}_S)(X_j - \bar{X}_S)^t$ and thus if Z_j are univariate normally distributed random variables, then $W = \sum_{X_j \in S} Z_j (X_j - \bar{X}_S)$ is with mean zero and covariance C_S.

3 The new algorithms vs Metropolis sampling.

They test the efficiency of their algorithms vs standard Metropolis-Hastings sampling on the Rosenbrock distribution ,see fig(1a)), which main property is that it is hard to sample. The results are shown in fig(1b), and are the auto-correlations times, which roughly speaking are an estimate of the number of steps required to generate an independent sample from the distribution. In this test the stretch move beats Metropolis Hastings by a factor of about 20, and the walk-move by 2. Also generate samples from the infinite-dimensional measure of an stochastic partial differential equation, where the walk move performs the best.

4 Ending remarks

The stretch move is the original basis of python package emcee optimized for parallelization. For the stretch move algorithm to sample a distribution properly it is necessary to use a large ensemble, if not, the ensemble is not a good representation of the density. The stretch move should give an easy way of making MCMC without spending a lot of time tuning a proposal function. The walk move algorithm is scheduled to be included in emcee in the next major release.