SAMPLING WITH CONNECTIONS

C. C. RODRIGUEZ
Department of Mathematics and Statistics
University at Albany, SUNY
Albany NY 12222, USA
carlos@math.albany.edu
http://omega.albany.edu:8008/

Abstract.

The Bayesian recipe is simple but powerful: "Compute the posterior by multiplying the likelihood by the prior". Often Monte Carlo simulation is the only effective technique for dealing with realistic likelihoods in many dimensions. Algorithms based on Markov Chains (MCMC) are able to approximate samples from complicated distributions that are known only up to a normalization constant but most MCMC methods are asymptotic and may take a long time to converge. This paper shows how the time to convergence and the quality of the approximation can be dramatically improved by making the algorithms travel along paths in the space of distributions.

More specifically. Let f and g be two probability densities (may be known only up to normalization constants) on the same probability space. We show a collection of new algorithms for approximating samples from f by sampling from g and uniforms only. These algorithms make use of a newly discovered exact rejection constant for two mixtures, $h_t = tf + (1-t)g$ between g and f. Specifically,

$$h_{t+\epsilon} < (1 + \epsilon/t)h_t.$$

This bound allows exact rejection algorithms to be combined with approximate MCMC algorithms producing remarkable improvements in performance. The methods are general. There are no constraints on the forms of f or g or the number of dimensions.

Key words: Markov Chain Monte Carlo, MCMC, Rejection Method, Metropolis Algorithm, Simulated Annealing, Mixture Connection

1. Introduction

The problem of computer simulation is simply the problem of generating samples from a given probability distribution. Nowadays there exist efficient algorithms for

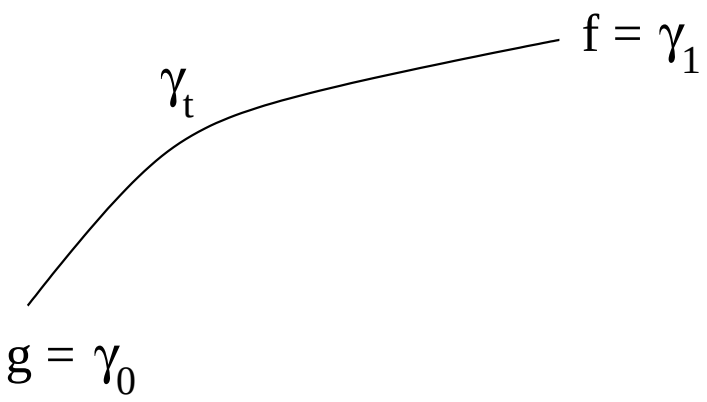

Figure 1. Path Connecting g and f

generating from most of the standard distributions of mathematical statistics such as Cauchy, Poisson, Gaussian, etc (see Devroye's book [1]).

In this paper we focus our attention on the problem of generating approximate samples from a probability distribution that typically does not have a name. Our interest are multimodal, high dimensional densities that are known only up to a normalization constant. An important application is the problem of simulating posterior distributions for realistic (complex) priors and likelihoods. Without extra information about the form of the density, the number of off the shelf methods available for simulation gets reduced to variations of the Metropolis algorithm ([2–4]).

The Metropolis algorithm approximates samples from a given distribution by sampling, for a sufficiently long time, a Markov chain that is known to have the desired density as its long run stationary distribution. The theory of Markov chains assures asymptotic convergence for any initial distribution but in practice the quality of the samples depends on the starting point and on the length of the run. Clearly, the closer (e.g. in total variation norm) the initial distribution is to the target, the faster we should expect to reach the stationary distribution.

An obvious idea, already used in the so called method of *thermodynamic integration*, is to connect an initial density g to the target density f via a path in the space of distributions (see Figure 1). Now, instead of trying to go with Metropolis from g straight towards f, we go to f by climbing up the path with little intermediate steps. This simple trick can generate dramatic improvements in the quality of estimates ([5]).

We introduce in this paper a general approach for improving the quality of Metropolis samples. Our method is based on the fact that every density $h_t = tf + (1-t)g$ along the mixture connection between g and f envelopes all the other densities $h_{t+\epsilon}$ located further up the mixture path with rejection constant $1 + \epsilon/t$. This simple but extremely powerful result is proved as Lemma (1) below. A number of algorithms for combining Metropolis with the exact Rejection method (see e.g.

[1]) are immediately suggested by this Lemma and we look at some of them in the paper. The idea behind these algorithms is that it is not difficult to arrive at h_t (for $t \approx 0$) if we start Metropolis iterations from g since $\|h_t - g\| \leq t$ in the total variation norm. Once we obtain a sample from h_t we may try to climb up the mixture path towards f by using the exact rejection constants given by Lemma (1). We may, for example, try to climb straight towards f but we should expect $1/t$ rejections before success. A better strategy is to take intermediate steps along the mixture path. We use simulated annealing to approximate optimal strategies obtaining dramatic improvements of efficiency. For example, if we subdivide the mixture path from g to f into 1000 equal steps $[1, 2, \ldots, 1000]$ then the annealing algorithm produced $[1, 2, 5, 14, 40, 117, 344, 1000]$ as the best sequence of steps. The expected number of rejections for this sequence is only 16.1 which is two orders of magnitude better than the naive sequence $[1, 1000]$ that has 1000 expected rejections. These best found sequences are essentially the same for all g, all f and all dimensions.

2. The Main Lemma

Lemma 1 *Let,*

$$\gamma_t(x) = tv(x) + (1-t)u(x)$$

be the mixture connection between two positive functions. Then, for all values of x

$$\gamma_{t+\epsilon}(x) < \left(1 + \frac{\epsilon}{t}\right) \gamma_t(x)$$

for all t *and* ϵ *so that the mixture parameters are always in* $[0, 1]$.

Proof: just write,

$$\begin{aligned} \lambda = \frac{\gamma_{t+\epsilon}(x)}{\gamma_t(x)} &= \frac{\gamma_t(x) + \epsilon(v(x) - u(x))}{\gamma_t(x)} \\ &= 1 + \epsilon \frac{v(x) - u(x)}{\gamma_t(x)} \end{aligned}$$

and consider two cases.

Case I: $v(x) \leq u(x)$ In this case it follows from the last equation that, $\lambda \leq 1$.

Case II: $v(x) > u(x)$ For this case write the denominator in the last equation above as, $\gamma_t(x) = u(x) + t(v(x) - u(x))$ and divide the numerator and the denominator by $(v(x) - u(x))$ to obtain,

$$\lambda = 1 + \frac{\epsilon}{t + u(x)/(v(x) - u(x))} < 1 + \frac{\epsilon}{t}$$

Q.E.D.

3. Climbing up the Mixture Path

What I call The MontyCarlos method is a general class of algorithms that makes use of Lemma (1) to combine Metropolis with Rejection. Let us assume that we want to go from an initial distribution with density proportional to g to a target distribution with density proportional to f along the mixture path $\gamma_t = tf + (1-t)g$. Choose a positive integer n and divide the interval $[0,1]$ into n equal pieces with end points $1/n, 2/n, \ldots, n/n$. The MontyCarlos algorithm depends on a given sequence of integers $L = [1, n_1, n_2, \ldots, n_k, n]$ with $1 < n_1 < n_2 < \ldots < n$. The sequence L encodes the points along the mixture path

$$\gamma_{1/n}, \gamma_{n_1/n}, \gamma_{n_2/n}, \ldots, \gamma_{n_k/n}, \gamma_{n/n}$$

that the algorithm will visit. The algorithm alternates Metropolis steps, starting from the last visited mixture, with Rejection tests attempting to climb up to the next point in the L sequence. Figure 2 shows the complete algorithm in pseudo code. The two subroutines used in the main algorithm, *Reject* and *Met* are shown in Figure 3 and Figure 4.

Figure 5 illustrates the MontyCarlos method for $n = 12$ and the sequence $L = [1, 2, 5, 10, 12]$. The algorithm starts by generating a sample from $g = \gamma_0$ then starting from this sample it moves (horizontally in the picture) with a few Metropolis iterations to get a new sample from $\gamma_{1/n} = \gamma_{1/12}$ and from here again Metropolis towards $\gamma_{2/n}$. Then the algorithm tries to qualify the observed sample from $\gamma_{2/n}$ as a sample from $\gamma_{4/n}$ (since $4+1$ is the next entry of L) by using the exact rejection constants given by Lemma (1). If the sample is accepted then it is saved as a backup by starting Metropolis from there towards $\gamma_{5/n}$. If the sample from $\gamma_{2/n}$ is rejected as a sample from $\gamma_{4/n}$ then the algorithm falls back to the previous backup sample and starts again from there. The algorithm continues in this way up and down until it eventually reaches the top producing a sample from f. Notice that once a sample gets accepted at a given height, that position gets anchored (indicated by the open circles in the picture) and it becomes the safety net to fall on to in the event of a rejection.

When n is small and g is close to f the method can be used for generating independent samples from f. For really complicated high dimensional f a high value for n is needed and there is no guarantee that g is close to f. In such cases, the method can be used for producing a single sample from f and after that continue with a version of metropolis but now knowing that metropolis has converged.

4. Searching for the Optimal Stepping Steps

The performance of the MontyCarlos method depends on the sequence of intermediate steps L. In this section we indicate how to obtain good sequences L for a given value of n. To do this we compute the expected cost of a sequence L under the assumption that the cost of a Metropolis step is M much bigger than the cost m of a rejection test. The value of M should reflect the computational cost of producing a sample from $\gamma_{t+1/n}$ by performing Metropolis iterations starting from

```
MontyCarlos (u, v, n, L, Stop_crit)

    {
    PARS ← (u, v, n, Stop_crit)
    li ← 0
    x ← sample u()
    xsaved ← x
    for each lf in L

            {
            if (lf = li + 1)

                    {
                    xsaved ← x
                    x ← Met(xsaved, lf/n, PARS)
                    }
                else if (lf < n)

                    {
                    xsaved ← Reject(x, xsaved, li, lf − 1, PARS)
                    x ← Met(xsaved, lf/n, PARS)
                    }
                else

                    {
                    x ← Reject(x, xsaved, li, n, PARS)
                    }
                li ← lf
            }
        Return x }
```

Figure 2. The MontyCarlos Algorithm

a sample from γ_t. Simulations experiments indicate that the best sequences L do not depend on the values of M and m as long as $M > m$.

For a given sequence L we denote the path followed by the algorithm as a string of symbols of the form

$$s \Rightarrow t \nearrow t + \epsilon$$

where this indicates going with Metropolis from γ_s to γ_t (with cost M) and attempting to qualify the sample from γ_t as a sample from $\gamma_{t+\epsilon}$ (at a price of m). If

Reject $(x, xsaved, li, lf, PARS)$

```
{
while (unif() lf/li [u(x) + li/n (v(x) − u(x))] > u(x) + lf/n (v(x) − u(x)))

        {
        x ← Met(xsaved, li/n, PARS)
        }
   Return x }
```

Figure 3. The Rejection Function

Met $(x, t, PARS)$

```
{
hx ← u(x) + t(v(x) − u(x))
until (Stop_crit)

        {
        y ← sample p(·|x)
        hy ← u(y) + t(v(y) − u(y))
        if ((hy > hx) or (unif() < hy/hx))

              {
              x ← y
              hx ← hy }
          }
   Return x }
```

Figure 4. The Metropolis Function

the sample gets rejected then we fall back to s and start again. By the fundamental Lemma (1) the expected number of rejections is $(1 + \epsilon/t)$ so the expected cost is,

$$< C >= (M + m)\left(1 + \frac{\epsilon}{t}\right) \tag{1}$$

To compute the expected cost of a general path we break up the path into independent segments and add the corresponding expected costs using equation (1). For example $L = [1, 2, 5, 10, 12]$ follows the path (see Figure 5),

$$0 \Rightarrow 1 \Rightarrow 2 \nearrow 4 \Rightarrow 5 \nearrow 9 \Rightarrow 10 \nearrow 12$$

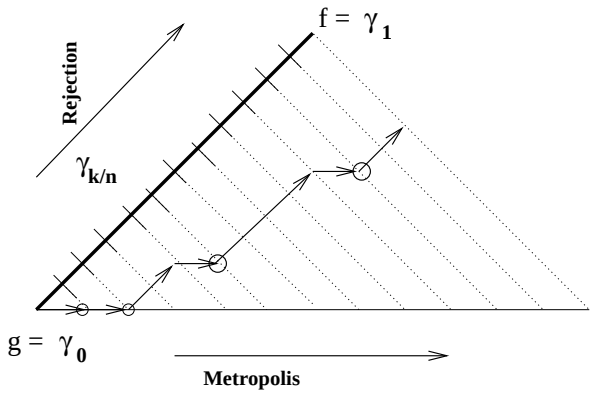

Figure 5. The MontyCarlos Method with Steps $[1, 2, 5, 10, 12]$

that decomposes into the independent segments,

$$\begin{aligned} 0 \Rightarrow 1 \quad & (< C >= M) \\ 1 \Rightarrow 2 \nearrow 4 \quad & (< C >= \frac{4}{2}(M+m)) \\ 4 \Rightarrow 5 \nearrow 9 \quad & (< C >= \frac{9}{5}(M+m)) \\ 9 \Rightarrow 10 \nearrow 12 \quad & (< C >= \frac{12}{10}(M+m)) \end{aligned}$$

adding the expected costs for each segment we obtain a total expected cost of $M + 5(M + m)$. In this way we can assign an expected cost to every sequence of steps L. The number of sequences L that end with n increases exponentially with n. For values of $n > 50$ or so, it becomes impractical to search the whole space of L's but the simulated annealing algorithm produces excellent values very quickly. Table 6 shows the best found sequences for different values of n. The expected costs were computed with $M = 100$ and $m = 1$. The last column shows the expected number of rejections associated to the sequence. Figure 7 shows the least squares fit of the observed best expected number of rejections $< R >$ against $\log n$. All the data fall almost exactly on the fitted line indicating that the best number of expected rejections increases logarithmically with n. The linear fit is so remarkably accurate that it is reasonable to believe that it holds for all n but we have not been able to prove it.

5. A Universal Class of Starting Points

The performance of the MontyCarlos method depends on the choice of starting density g. Feasible g's must possess the following characteristics:

n	best found L	$< Cost >$	$< Rejections >$
10	[1,2,4,10]	504.0	4.0
20	[1,2,4,9,20]	677.9	5.7
50	[1,2,6,17,50]	918.9	8.1
100	[1,2,5,13,36,100]	1096.9	9.9
200	[1,2,5,12,30,76,200]	1286.6	11.7
500	[1,2,6,18,54,164,500]	1548.9	14.3
1000	[1,2,5,14,40,117,344,1000]	1726.6	16.1
2000	[1,2,5,13,35,96,262,721,2000]	1915.0	18.0
5000	[1,2,5,15,48,153,487,1556,5000]	2189.0	20.7

Figure 6. The Best Found L Sequences

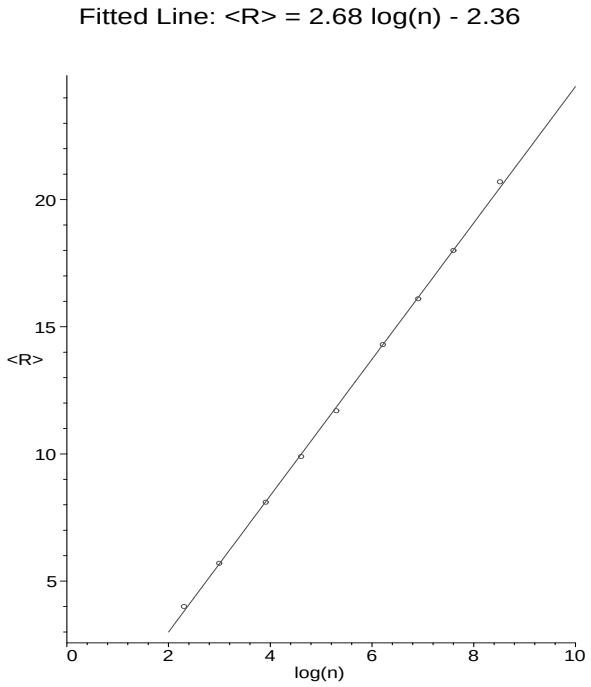

Figure 7. Best Found Expected Number of Rejections against $\log n$

1. They are easy to evaluate up to a constant. i.e., we must know how to evaluate $u(x)$ with $g(x) \propto u(x)$.
2. There is available an efficient algorithm to produce samples with density g.

3. g should be close to the target f.

A general class of starting points with the above characteristics is given by the *first Metropolis transition* towards f starting from a point x_0 where $f(x_0)$ is large and a sufficiently spread proposal.

Clearly there is an algorithm available. Starting from x_0 we generate a sample x from a proposal distribution with density $p(x|x_0)$. If $f(x) > f(x_0)$ we go there for sure. Otherwise, we flip a coin with probability of heads $f(x)/f(x_0)$ and we go to visit the proposed sample x if the coin comes up heads. If the coin comes up tails we stay at x_0 and try another proposal proceeding in the same way as before until we eventually move away from x_0. Let $g(x)$ be the density of the first visited state which is different from x_0. Let $g(x) = u(x)/Z_u$ with $Z_u = \int u(x)dx$ and $f(x) = v(x)/Z_v$ with $Z_v = \int v(x)dx$. Then denoting by,

$$(a)_+ = \begin{cases} a & \text{if } a > 0 \\ 0 & \text{otherwise} \end{cases}$$

we have,

Theorem 1

$$u(x) = \{v(x) - (v(x) - v(x_0))_+\}p(x|x_0) \tag{2}$$

Moreover,

$$\text{if } p(x|x_0) < 1 \text{ for all } x, \text{ then } Z_v > Z_u. \tag{3}$$

Proof: Let $X_0, X_1, \ldots$ be the Metropolis Markov Chain and to simplify the notation assume, without loss of generality, a discrete state space. Then for $y \neq x_0$

$$\begin{aligned} P[X_1 = y|X_0 = x_0] &= P[\text{propose } y \text{ and accept } y|X_0 = x_0] \\ &= p(y|x_0)P[\text{accept } y|\text{propose } y, X_0 = x_0] \\ &= \begin{cases} p(y|x_0) & \text{if } v(y) > v(x_0) \\ \frac{v(y)}{v(x_0)}p(y|x_0) & \text{if } v(y) \leq v(x_0) \end{cases} \end{aligned}$$

and this can be written on a single line as

$$P[X_1 = y|X_0 = x_0] = \left\{\frac{v(y) - (v(y) - v(x_0))_+}{v(x_0)}\right\}p(y|x_0) \tag{4}$$

Thus, the probability of staying on x_0 will be q with,

$$\begin{aligned} q &= P[X_1 = x_0|X_0 = x_0] \\ &= 1 - \sum_{y \neq x_0}\left\{\frac{v(y) - (v(y) - v(x_0))_+}{v(x_0)}\right\}p(y|x_0) \end{aligned} \tag{5}$$

Let the random variable Y denote the value of the first visited state different from x_0. Hence, using the Markov property and equations (5) and (4) we obtain,

$$P[Y = y|X_0 = x_0] = \sum_{k=1}^{\infty} P[X_1 = x_0, X_2 = x_0, \ldots, X_{k-1} = x_0, X_k = y, |X_0 = x_0]$$

$$
\begin{aligned}
&= \sum_{k=1}^{\infty} q^{k-1} \left\{ \frac{v(y) - (v(y) - v(x_0))_+}{v(x_0)} \right\} p(y|x_0) \\
&= \left(\frac{1}{v(x_0)(1-q)} \right) u(y) \\
&\propto u(y)
\end{aligned}
$$

which shows (2). To show (3) assume the premise and partition the state space $\mathcal{X}$ into $W = \{x : v(x) > v(x_0)\}$ and the rest $W^c = W \setminus \mathcal{X}$ to get,

$$
\begin{aligned}
Z_v &= \int_W v(x)dx + \int_{W^c} v(x)dx \\
&> \int_W v(x_0)dx + \int_{W^c} v(x)dx \\
&> \int_W v(x_0)p(x|x_0)dx + \int_{W^c} v(x)p(x|x_0)dx = Z_u
\end{aligned}
$$

Q.E.D.

As a Corollary of the main Lemma (1) and Theorem (1) we have,

Corollary 1 *For any sequence of steps L and any pair of positive functions u and v with $Z_u < Z_v$ the MontyCarlos algorithm will finish after a finite number of iterations with probability* 1.

Proof: We show that for a given sequence of steps L, the expected number of rejections for the MontyCarlos method along the normalized mixture path from g to f is larger than for the unnormalized path from u to v when $Z_u < Z_v$ in particular when u is taken as (2) with proposal distribution satisfying (3).

Let

$$\gamma_t(x) = tv(x) + (1-t)u(x)$$

and let,

$$Z_t = \int \gamma_t(x)dx = Z_u + t(Z_v - Z_u)$$

By the basic Lemma (1) we have,

$$\gamma_{t+\epsilon}(x) < \left(1 + \frac{\epsilon}{t}\right) \gamma_t(x) \tag{6}$$

this inequality is used in the rejection method to sample uniformly under the graph of $\gamma_{t+\epsilon}$ producing as a consequence a sample with density proportional to $\gamma_{t+\epsilon}$. However, the expected number of rejections is given by the rejection constant only when the densities are normalized (see [1]). Thus, denoting by

$$\bar{\gamma}_t(x) = \frac{1}{Z_t}\gamma_t(x)$$

we obtain from (6) that,

$$\bar{\gamma}_{t+\epsilon}(x) < \left(1 + \frac{\epsilon}{t}\right) \frac{Z_t}{Z_{t+\epsilon}} \bar{\gamma}_t(x) \tag{7}$$

By Theorem (1), $Z_v > Z_u$. Thus,

$$\frac{Z_t}{Z_{t+\epsilon}} = \frac{Z_t}{Z_t + \epsilon(Z_v - Z_u)} < 1$$

Using this last inequality together with (7) we obtain, that the expected number of rejections when using the unnormalized u and v functions is smaller than $(1 + \epsilon/t)$ anywhere along the path. **Q.E.D.**

5.1. UNNORMALIZED MIXTURES

The above corollary shows that the expected number of iterations for the MontyCarlos algorithm is finite. Hence, the algorithm halts with probability one. The same conclusion is reached for any positive functions u and v as long as $Z_v > Z_u$. It may seem that by making Z_u smaller we could improve the performance of the algorithm but this is obviously incorrect. It is not difficult to show that as Z_u decreases the algorithm speeds up only at the expense of the quality of the approximation. To see this let,

$$\lambda = \frac{Z_v}{Z_u} \tag{8}$$

When g and f are normalized densities the total variation distance between the mixtures $\gamma_t = tf + (1-t)g$ and $\gamma_{t+\epsilon}$ is at most ϵ. However, for unnormalized positive functions u and v, we have,

$$\bar{\gamma}_t(x) = \left(\frac{tZ_v}{Z_t}\right) f(x) + \left(\frac{(1-t)Z_u}{Z_t}\right) g(x) \tag{9}$$

which shows that a t-mixture of v and u is really a t^*-mixture between the corresponding normalized densities f and g where,

$$t^* = \frac{tZ_v}{Z_t} = \frac{t}{t + \frac{1-t}{\lambda}} \tag{10}$$

and we have,

$$\|\bar{\gamma}_t - g\| \leq \frac{t}{t + \frac{1-t}{\lambda}} \tag{11}$$

When λ increases to ∞ the value of t^* approaches 1 and hence, $\bar{\gamma}_t = \bar{\gamma}_1 = f$. On the other hand when λ goes to 1 we have $t^* = t$. Equation (10) shows that there is a tradeoff between the expected number of rejections necessary to pass from t to $t + \epsilon$ along the mixture path of unnormalized functions and the quality of the approximation. For we can only reduce the expected number of rejections by increasing λ which makes the actual position t^* along the mixture of normalized densities g and f to get closer to f. From (6) we can write explicitly the expected

number of rejections to jump from t to $t+\epsilon$ along the mixture path between unnormalized functions u and v as,

$$\begin{aligned} c(t,\epsilon,\lambda) &= (1+\epsilon/t)\,\frac{Z_u + t(Z_v - Z_u)}{Z_t + \epsilon(Z_v - Z_u)} \\ &= \frac{(1+\epsilon/t)}{1+\frac{\epsilon}{t+\frac{Z_u}{Z_v - Z_u}}} \\ &= \frac{1+\epsilon/t}{1+\frac{\epsilon}{t+\frac{1}{\lambda-1}}} \end{aligned} \tag{12}$$

Notice the interesting limits of $c(t,\epsilon,\lambda)$ that can be obtained from (12). First c approaches 1 when either ϵ goes to zero or λ goes to ∞. However, λ going to ∞ makes t^* to go to 1 and this case really gives the same information as when ϵ goes to 0. Also of interest is the fact that c approaches ∞ when t goes to 0 showing that there is a singularity for the rejection algorithm at $t=0$. It takes an infinite amount of time to get out of the initial point u with only rejections.

6. From α to Ω

The existence of exact rejection constants for points along the mixture connection allows to combine exact rejection methods with asymptotic methods based on Markov chains. In principle the same trick can be applied to any path (not just mixtures) provided we have the equivalent of Lemma (1) for that path. I believe that by subdividing a general path into little "mixture segments" it should be possible to exploit Lemma (1) to approximate rejection constants for more general paths. This is desirable, for once we learn how to navigate along different paths we can then search for the best. The notion of "best" will then correspond to a notion of geodesic and therefore to a notion of curvature as well. A lot of unexplored geometry is clearly lurking behind computer sampling. At stake there is the possibility of nothing less than the creation of automatic, objective, bayesian inference engines based on entropic priors with free[1] parameter α. Simulations, source code and new developments will be available from the author's server at *http://omega.albany.edu:8008/*.

Acknowledgments- This paper was conceived while the author was on sabbatical leave at Penn State. I thank the members of the department of Statistics at PSU for providing me with a very enjoyable and productive environment. I am particularly in debt to David Hunter who first realized that my first proposition for climbing up the mixture path without backup would not work.

[1] For a prior to be a prior it must not depend on the observed data, not even on the actual number, n say, of observations. Besides, any prior that claims to depend on entropy must have a free scalar parameter since the notion of entropy itself is only defined up to a proportionality constant

References

1. L. Devroye, *Non-Uniform Random Variate Generation*, no. ISBN number 0-387-96305-7, Springer-Verlag, New York, 1986. (http://www-cgrl.cs.mcgill.ca/~luc/rng.html).
2. N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller, "Equations of state calculations by fast computing machines," *J. Chem. Phys.*, **21**, pp. 1087–1091, 1953.
3. W. K. Hastings, "Monte carlo sampling methods using markov chains and their applications," *Biometrika*, **57**, pp. 97–109, 1970.
4. S. Geman and D. Geman, "Stochastic relaxation, gibbs distribution and the bayesian restoration of images," *IEEE Trans. Pat. Anal. Mach. Intell.*, **6**, pp. 721–741, 1984.
5. A. Gelman and X.-L. Meng, "Simulating normalizing constants: From importance sampling to bridge sampling to path sampling," *Statistical Science*, **13**, pp. 163–195, 1998. http://www.stat.columbia.edu/~gelman/research/published/xlfinal2.ps.