

Chapter 8 Sampled solutions to Inverse Problems

8.1 Introduction

The end result of modeling in the Bayesian formalism is a posterior distribution $\Pr(\mathbf{f}|\mathbf{d})$ where \mathbf{d} is a vector of data and \mathbf{f} is a vector of parameter values of interest which is transformed by the physical process into the data. Observation of the data \mathbf{d} involve a loss of information, as the transformation of \mathbf{f} into \mathbf{d} (the forward problem) may have random components (“noise”) as well as a possible reduction of dimension. We assume that the physical process leading from \mathbf{f} to \mathbf{d} is understood, at least in a probabilistic sense, so that the forward probability function $\Pr(\mathbf{d}|\mathbf{f})$ may be calculated. In the Bayesian formalism, the forward probability function is regarded as a function of \mathbf{f} , and enters Bayes’ theorem as the likelihood function. We then have

$$\Pr(\mathbf{f}|\mathbf{d}) \propto \Pr(\mathbf{d}|\mathbf{f})\Pr(\mathbf{f}) \quad (8.1)$$

up to a normalizing constant independent of \mathbf{f} .

In many cases of interest, the full posterior probability distribution is hopelessly analytically intractable, since the number of components in \mathbf{f} may be very large and the prior probability function $\Pr(\mathbf{f})$ may involve information which is difficult to express in analytic terms (e.g. in an image reconstruction problem, the true object \mathbf{f} may be known to consist of a single “blob” of material so that its boundary is simply connected). How then do we get at \mathbf{f} ? What are the most likely values for \mathbf{f} given the data, and what are typical or “average” values? Our problem may be treated by simulation: it is enough that we can test the plausibility of any particular guess at \mathbf{f} by simulating the physical process leading from \mathbf{f} to \mathbf{d} . We draw samples from the set Ω of all possible \mathbf{f} ’s, each sample drawn with probability $\Pr(\mathbf{f}|\mathbf{d})$. In this way we get a set $\Theta = \{\mathbf{f}_1, \mathbf{f}_2, \dots, \mathbf{f}_N\}$ of samples distributed like the posterior distribution. Inference on $\Pr(\mathbf{f}|\mathbf{d})$ becomes inference on $\{\mathbf{f}_i\}_{i=1}^N$. For example, the mode and mean of the samples in Θ give us an estimate of the most likely and average values of \mathbf{f} . Given some set of parameters A , we can estimate $\Pr(\mathbf{f} \in A|\mathbf{d})$ by calculating the fraction of samples which lie in the set A . Sampling-based methods are called **Monte-Carlo** methods.

Again, however, standard sampling techniques are useless when the posterior involves many variables and is otherwise intractable. In particular, one generally needs to have a closed form for the constant which normalizes the probability distribution we want to sample. (The exception to this, rejection sampling, is restricted in other respects). Markov chain sampling methods enable us to sample the kinds of complex distributions that “natural” models of physical processes tend to generate. In the previous chapter, we have considered Markov chains on discrete state spaces. In this chapter we shall also extend the MCMC method to continuous and mixed state spaces, allowing the solution of parameter estimation problems.

8.2 Recovering a binary matrix from noisy data

We first consider an example involving a finite (but large) state space based on the binary Markov random field described in the previous chapter. Consider the problem of reconstructing a binary image (i.e., one in which each pixel is either black or white) from observations of the image which are corrupted by independent zero-mean Gaussian noise. Let the image \mathbf{f} be of size $M \times N$ pixels, and denote the value of the (m, n) ’th pixel by $f_{mn} \in \{-1, 1\}$. The state space of all images Ω thus has 2^{MN} elements. The observed data $\mathbf{d} = \{d_{mn}\}$ are related to \mathbf{f} via

$$d_{mn} = f_{mn} + \varepsilon_{mn}$$

where each ε_{mn} is an independent sample of zero-mean Gaussian noise of variance σ^2 . Note that although each f_{mn} is black (-1) or white ($+1$), the values of d_{mn} can take on any real value. Figure 8.1 shows an example of the true image \mathbf{f} and a data set \mathbf{d} obtained for $\sigma = 2$

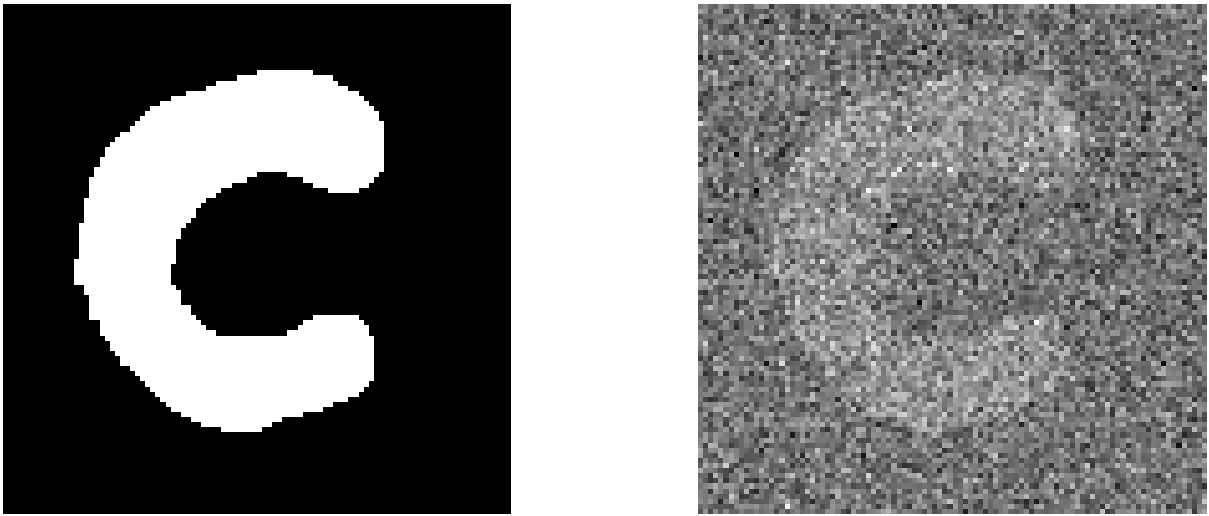


Figure 8.1 A binary image \mathbf{f} and a noise-corrupted data set \mathbf{d} obtained by adding samples of independent zero mean Gaussian noise of standard deviation $\sigma = 2$ to the image.

The likelihood function for this problem is given by

$$\Pr(\mathbf{d}|\mathbf{f}) \propto \exp \left[-\frac{1}{2\sigma^2} \sum_{m,n} (d_{mn} - f_{mn})^2 \right].$$

In order to find the posterior probability function, a prior probability distribution $\Pr(\mathbf{f})$ is required. This encodes our state of knowledge about which images are (à priori, without any data) more likely to occur. For example, we might

1. have no prior prejudice whatsoever, i.e., $\Pr(\mathbf{f}) = 2^{-MN}$, which is uniform on Ω .
2. favour smooth images: since the material is likely to be in lumps, we regard reconstructions in which the 1's and -1's separate out in blobs as *a priori* more probable. In this case, the binary Markov random field of the last section might make a reasonable choice:

$$\Pr(\mathbf{f}) = \frac{1}{Z} \exp(-2J\#\mathbf{f}), \quad (8.2)$$

where J is our lumping parameter. When $J = 0$, there is no smoothing, whereas if J is large, we favour a uniform image of a single colour. A model which favours “simple” reconstructions is called parsimonious.

8.2.1 Uniform Prior

If we use the uniform prior, the posterior probability is equal to the likelihood. An implementation of the Metropolis-Hastings Markov Chain Monte Carlo (MH MCMC) algorithm which draws samples from the posterior probability involves the following steps:

1. Let $X_n = \mathbf{f}$ denote the current state of the Markov chain. A pixel with coordinates kl is selected at random and the colour of the pixel is flipped, producing a candidate state \mathbf{f}' where

$$f'_{ij} = \begin{cases} -f_{kl} & \text{if } i = k \text{ and } j = l \\ f_{ij} & \text{otherwise} \end{cases} \quad (8.3)$$

The generation probability $g(\mathbf{f}'|\mathbf{f})$ is zero if \mathbf{f}' and \mathbf{f} differ by more than one pixel, and is equal to $1/(MN)$ if they differ by exactly one pixel.

2. Calculate the acceptance probability $\alpha(\mathbf{f}'|\mathbf{f})$ using

$$\alpha(\mathbf{f}'|\mathbf{f}) = \min \left\{ 1, \frac{\Pr(\mathbf{f}'|\mathbf{d}) g(\mathbf{f}|\mathbf{f}')}{\Pr(\mathbf{f}|\mathbf{d}) g(\mathbf{f}'|\mathbf{f})} \right\} \quad (8.4)$$

If \mathbf{f}' is generated as described above, the ratio $g(\mathbf{f}'|\mathbf{f})/g(\mathbf{f}|\mathbf{f}') = 1$. The ratio of posterior probabilities is

$$\frac{\Pr(\mathbf{f}'|\mathbf{d})}{\Pr(\mathbf{f}|\mathbf{d})} = \frac{\Pr(\mathbf{d}|\mathbf{f}') \Pr(\mathbf{f}')}{\Pr(\mathbf{d}|\mathbf{f}) \Pr(\mathbf{f})} \quad (8.5)$$

which for a uniform prior reduces to the likelihood ratio $\Pr(\mathbf{d}|\mathbf{f}')/\Pr(\mathbf{d}|\mathbf{f})$.

When using the MH MCMC algorithm, calculating the ratio of posterior probabilities occurs on every step, and so should be done as efficiently as possible. For additive Gaussian noise,

$$\frac{\Pr(\mathbf{d}|\mathbf{f}')}{\Pr(\mathbf{d}|\mathbf{f})} = \exp \left[-\frac{1}{2\sigma^2} \sum_{i,j} \left\{ (d_{ij} - f'_{ij})^2 - (d_{ij} - f_{ij})^2 \right\} \right]. \quad (8.6)$$

We can compute this much more efficiently by using the fact that the only term in the sum which changes when \mathbf{f} is replaced by \mathbf{f}' is that which involves the single pixel with indices k, l . Hence

$$\frac{\Pr(\mathbf{d}|\mathbf{f}')}{\Pr(\mathbf{d}|\mathbf{f})} = \exp \left[-\frac{1}{2\sigma^2} \left\{ (d_{kl} - f'_{kl})^2 - (d_{kl} - f_{kl})^2 \right\} \right] = \exp \left[-\frac{1}{2\sigma^2} \left\{ -2d_{kl}(f'_{kl} - f_{kl}) + (f'^2_{kl} - f^2_{kl}) \right\} \right] \quad (8.7)$$

Finally, using the fact that $f'_{kl} = -f_{kl}$, we find

$$\frac{\Pr(\mathbf{d}|\mathbf{f}')}{\Pr(\mathbf{d}|\mathbf{f})} = \exp \left[\frac{d_{kl}(f'_{kl} - f_{kl})}{\sigma^2} \right], \quad (8.8)$$

which can be computed very quickly.

3. Accept the candidate state \mathbf{f}' with probability $\alpha(\mathbf{f}'|\mathbf{f})$. If the state is accepted, set $X_{n+1} = \mathbf{f}'$, otherwise set $X_{n+1} = \mathbf{f}$.

A Matlab implementation of the algorithm is given below, and the result of the reconstruction is shown in Figure 8.2.

```
F = imread('blob1.bmp');
[M,N] = size(F);
sigma = 2;
d = double(F); d(find(d==0))=-1;
d = d + sigma*randn(size(d));
figure(1);
subplot(1,2,1); imagesc(F); set(gca, 'Visible', 'off'); colormap gray; axis square;
subplot(1,2,2); imagesc(d); set(gca, 'Visible', 'off'); colormap gray; axis square;
drawnow
J = 0.5;
f = ones(M,N);
figure(3);
subplot(1,2,1); hf=imagesc(f); set(gca, 'Visible', 'off');
colormap gray; axis square; drawnow;
mf = zeros(M,N);
subplot(1,2,2); hm=imagesc(mf); set(gca, 'Visible', 'off');
colormap gray; axis square; drawnow;
SS = 10000;
misfit = [];
adj = [-1 1 0 0; 0 0 -1 1];
iter = 0;
```

```

while 1
  ix = ceil(N*rand(1)); iy = ceil(M*rand(1));
  pos = iy + M*(ix-1); fp = -f(pos);
  LkdRat = exp(d(pos)*(fp - f(pos))/sigma.^2);
  alpha = LkdRat;
  if rand<alpha % Prob of acceptance = min(1,alpha)
    f(pos) = fp;
  end
  iter = iter + 1;
  if rem(iter,SS) == 0,
    mf = mf+f; NS = iter/SS; iter
    set(hf,'CData',f);
    set(hm,'CData',mf); drawnow
    if iter/SS > length(misfit)
      misfit = [misfit,zeros(100,1)];
      misfit(iter/SS) = sum(sum((d-f).^2))/sigma;
    end
  end
end
end
end

```



Figure 8.2 Left hand figure shows a single sample from the MH MCMC algorithm, using a uniform prior and right hand figure shows the mean of 600 sweeps, each consisting of 10000 iterations of the algorithm.

From the reconstruction, it is clear that the uniform prior does not significantly reduce the noise, since the noise amplitude is so large that there is a significant probability that a given measured value arose from either a +1 or -1. For a uniform prior, the state of each pixel is independent of every other, so no inter-pixel correlations are used to improve the reconstruction.

8.2.2 Markov Radom Field with Ising Prior

As discussed in the previous chapter, an Ising prior on the space of binary images is one of the form

$$\Pr(\mathbf{f}) = \frac{1}{\mathcal{Z}_J} \exp(-2J\#\mathbf{f}) \quad (8.9)$$

where $\#\mathbf{f}$ is the number of edges linking disagreeing pixels in \mathbf{f} , $J > 0$ is a constant and \mathcal{Z}_J is the normalization constant. In the MH MCMC algorithm, when comparing the current state \mathbf{f} with a candidate state \mathbf{f}' which differs from \mathbf{f} by a single pixel, the ratio of posterior probabilities is given by

$$\frac{\Pr(\mathbf{f}'|\mathbf{d})}{\Pr(\mathbf{f}|\mathbf{d})} = \frac{\Pr(\mathbf{d}|\mathbf{f}') \Pr(\mathbf{f}')}{\Pr(\mathbf{d}|\mathbf{f}) \Pr(\mathbf{f})} = \exp\left[\frac{d_{kl}(f'_{kl} - f_{kl})}{\sigma^2}\right] \exp[2J(\#\mathbf{f} - \#\mathbf{f}')]. \quad (8.10)$$

Notice how the normalization constant \mathcal{Z}_J (which is difficult to evaluate) has cancelled in this ratio. The quantity $\#f - \#f'$ is easy to evaluate for a single-pixel change as it only involves examining at most four edges which link the pixel to its neighbours.

The Matlab code for this problem is essentially the same as for the uniform prior, except that the calculation of the acceptance probability is modified to the following:

```
LkdRat = exp(d(pos)*(fp - f(pos))/sigma.^2);
nbrs = pos + [-1,1,-M,M]; nbrs(find([iy==1,iy==M,ix==1,ix==N])) = [];
disagreef = sum(f(nbrs)~=f(pos)); disagreefp = sum(f(nbrs)~=fp);
DelLogPr = 2*J*(disagreef - disagreefp);
alpha = exp(DelLogPr)*LkdRat;
if rand<alpha
    f(pos) = fp;
end
```

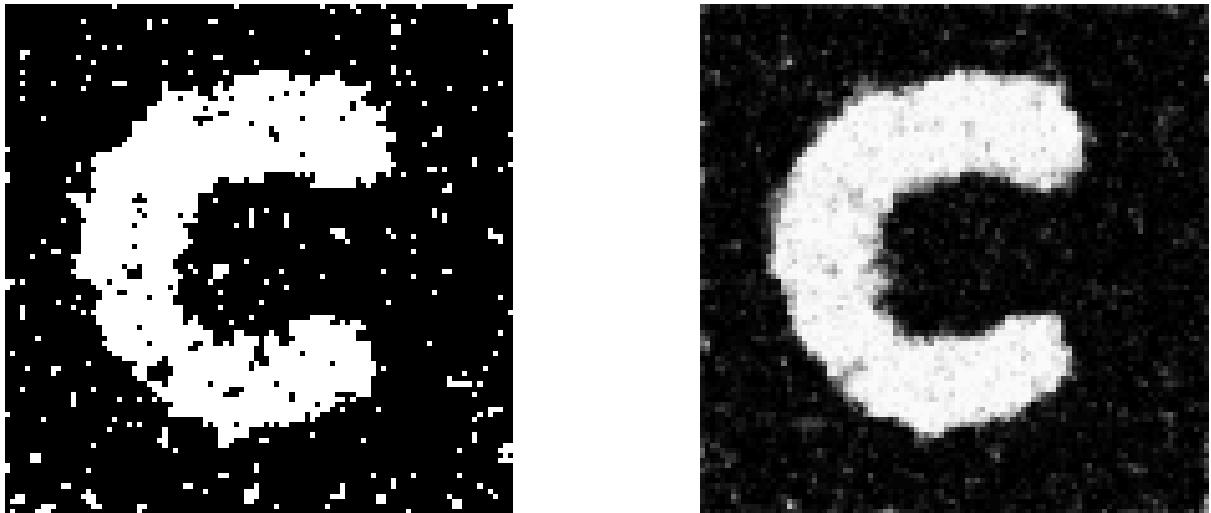


Figure 8.3 Left hand figure shows a single sample from the MH MCMC algorithm, using a Ising prior with $J = 0.5$ and right hand figure shows the mean of 600 sweeps, each consisting of 10000 iterations of the algorithm.

From the reconstruction shown in Figure 8.3, the clumping encouraged by the Ising prior is apparent. This example illustrates the power of using prior information to help constrain the reconstruction in image processing.

8.3 Recovering a binary matrix from its row and column sums

In this section we consider an artificial problem: the estimation of a matrix (or “image”) of zeros and ones from its row and column sums, as observed in the presence of noise.

An archaeologist is able to X-ray a square pillar from two sides only. We are given a single slice of this data - two 64-pixel long sequences of noisy intensity readings representing projections onto two orthogonal axes parallel to the sides of the pillar, at a fixed height up the pillar. Suppose that the intensity of the rays falls off from its initial value in proportion to the total amount of mass in the path of the x-rays incident a given pixel. The pillar is expected to be composed of two types of material of known density. The materials are expected to be separated out in lumps of unknown position.

The problem might be recast: give a Bayesian formulation for the problem of reconstructing an $N \times N$ matrix $\mathbf{f} = [f_{mn}]$ given that $f_{mn} \in \{-1, 1\}$ and given the row and column sums of \mathbf{f} . Each row and each column sum is an independent measurement, with an uncertainty which is assumed to be Gaussian, mean zero, and standard deviation σ .

Let the data be $\mathbf{d} = (\mathbf{r}, \mathbf{c})$ where \mathbf{r} and \mathbf{c} are N -component vectors. In terms of \mathbf{f} ,

$$r_m = \sum_{n=1}^N f_{mn} + \epsilon_m^r, \quad (8.11)$$

$$c_n = \sum_{m=1}^N f_{mn} + \epsilon_n^c. \quad (8.12)$$

where ϵ_m^r and ϵ_n^c are normally distributed r.v. mean zero and standard deviation σ , i.e.,

$$\epsilon_m^r \sim \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(\epsilon_m^r)^2/2\sigma^2} \quad \epsilon_n^c \sim \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(\epsilon_n^c)^2/2\sigma^2}, \text{ for } m, n = 1 \dots N$$

Given an image \mathbf{f} , let $[\mathbf{f}]_m$ denote the sum over the m 'th row of \mathbf{f} and let $[\mathbf{f}]^n$ denote the sum over the n 'th column of \mathbf{f} . The posterior probability is given by

$$\Pr(\mathbf{f}|\mathbf{d}) \propto \Pr(\mathbf{d}|\mathbf{f}) \Pr(\mathbf{f}) \quad (8.13)$$

The likelihood function is given by

$$\Pr(\mathbf{d}|\mathbf{f}) = \prod_{m=1}^N \Pr(\text{observe } r_m | [\mathbf{f}]_m) \prod_{n=1}^N \Pr(\text{observe } c_n | [\mathbf{f}]^n) \quad (8.14)$$

$$\propto \exp\left(-\frac{1}{2\sigma^2} \sum_{m=1}^N (r_m - [\mathbf{f}]_m)^2 - \frac{1}{2\sigma^2} \sum_{n=1}^N (c_n - [\mathbf{f}]^n)^2\right). \quad (8.15)$$

If we use an Ising prior for \mathbf{f} , the posterior probability is

$$\Pr(\mathbf{f}|\mathbf{d}) = \frac{1}{\mathcal{Z}_{J,\sigma}} \exp\left(-2J\#\mathbf{f} - \frac{1}{2\sigma^2} \sum_{m=1}^N (r_m - [\mathbf{f}]_m)^2 - \frac{1}{2\sigma^2} \sum_{n=1}^N (c_n - [\mathbf{f}]^n)^2\right) \quad (8.16)$$

where $\mathcal{Z}_{J,\sigma}^{-1}$ is again an unknown, intractable normalizing constant. We can sample from this posterior probability function using Markov chain Monte Carlo and the Metropolis-Hastings update rule as before. We aim to produce a Markov chain with equilibrium distribution $\pi_{\mathbf{f}} = \Pr(\mathbf{f}|\mathbf{d})$, $\mathbf{f} \in \Omega$. We can use much the same algorithm as above:

Let $X_n = \mathbf{f}$. X_{n+1} is determined in the following way:

1. Given $\mathbf{f} = (f_{1,1}, f_{1,2}, \dots, f_{N,N})$, pick one of the N^2 pixels (m, n) at random.
Set $\mathbf{f}' = (f_{1,1}, f_{1,2}, \dots, -f_{m,n}, \dots, f_{N,N})$. Our generation scheme is suitable for MH MCMC.
2. With probability

$$\begin{aligned} (\mathbf{f}'|\mathbf{f}) &= \min\left\{1, \frac{\Pr(\mathbf{f}'|\mathbf{d})g(\mathbf{f}|\mathbf{f}')}{\Pr(\mathbf{f}|\mathbf{d})g(\mathbf{f}'|\mathbf{f})}\right\} \\ &= \min\left\{1, \exp\left(\begin{aligned} &-2J(\#\mathbf{f}' - \#\mathbf{f}) - \frac{1}{2\sigma^2} \sum_{m=1}^N [(r_m - [\mathbf{f}']_m)^2 - (r_m - [\mathbf{f}]_m)^2] \\ &- \frac{1}{2\sigma^2} \sum_{n=1}^N [(c_n - [\mathbf{f}']^n)^2 - (c_n - [\mathbf{f}]^n)^2] \end{aligned}\right)\right\} \\ &= \min\left\{1, \exp\left(-2J(\#\mathbf{f}' - \#\mathbf{f}) - \frac{2f_{m,n}}{\sigma^2} (r_m + c_n - [\mathbf{f}]_m - [\mathbf{f}]^n + 2f_{m,n})\right)\right\} \end{aligned}$$

set $X_{n+1} = \mathbf{f}'$. Otherwise we set $X_{n+1} = \mathbf{f}$. The above algorithm was implemented in *C*. The row and column sums of a matrix of side 64 were observed under i.i.d. Gaussian noise, mean zero, standard deviation 2.5. A sequence of states from MCMC sampling of the posterior for prior parameter $J = 0.5$ are shown in Figure 8.4.

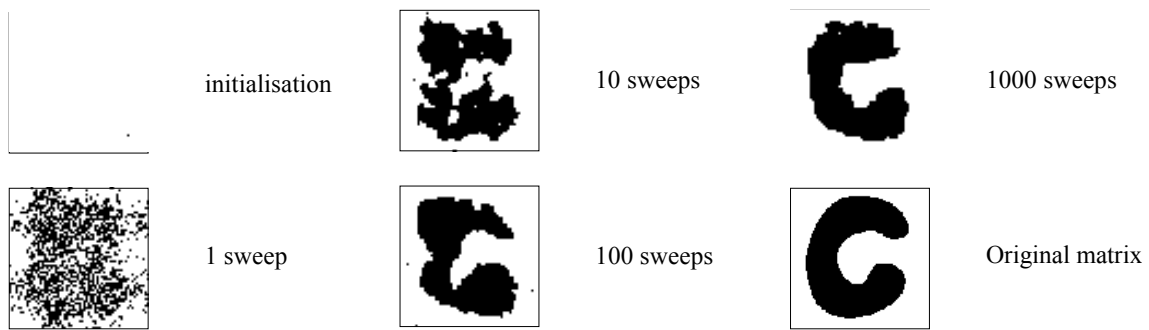


Figure 8.4 Sample from posterior for matrix reconstruction problem. 1 sweep = 4096 MC steps.

8.3.1 References

1. D. Geman and S. Geman (1984) “Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images.” IEEE Transactions on Pattern Analysis and Machine Intelligence, **6**, 721–741.
2. D.M. Grieg, B.T. Porteous and A.H. Seheult (1989) “Exact maximum à posteriori estimation for binary images”, J. Royal Stat Soc B, **51**, 271–279.

8.4 Markov Chain Monte Carlo on Continuous State Spaces

Up to here, Ω has been discrete. Irreducibility is hard to define for continuous state spaces. We shall regard the continuous case as approximated by a distribution on a discrete space.

Problem: How to construct a Markov chain $\{X_n\}_{n=0}^\infty$ given an equilibrium probability density $q(x)$?

Definition: The kernel of a Markov chain on a continuous space Ω , corresponding to the transition matrix P_{ij} of the discrete chain is the density $p(x'|x)$ where

$$p(x'|x) dx' = \Pr(x' \leq X_{n+1} < x' + dx' | X_n = x), \tag{8.17}$$

i.e., given that we are in state $X_n = x$ at update n , $p(x'|x) dx'$ gives the probability to land in the interval $[x', x' + dx')$ at the next update.

For correct normalization, we have

$$\int_{\Omega} p(x'|x) dx' = 1 \tag{8.18}$$

since X_{n+1} must take **some** value. Also if $\pi^{(n)}(x)$ is the probability density, so that $\pi^{(n)}(x) dx$ is the probability to be in the set $[x, x + dx)$ after n steps, then

$$\pi^{(n+1)}(x') = \int_{\Omega} \pi^{(n)}(x) p(x'|x) dx \tag{8.19}$$

is the probability density at the next step. This should be compared with

$$\pi_j^{(n+1)} = \sum_i \pi_i^{(n)} P_{ij} \tag{8.20}$$

for discrete Ω .

Definition: A probability density $\pi(x)$ is stationary for $p(x'|x)$ if

$$\pi(x') = \int_{\Omega} p(x'|x) \pi(x) dx, \tag{8.21}$$

i.e., if the update preserves the distribution.

Assertion: The kernel $p(x \rightarrow dx')$ is reversible with respect to the distribution $\pi(x)$ iff

$$p(x'|x) \pi(x) = p(x|x') \pi(x') \tag{8.22}$$

Notice that if p is reversible with respect to π then π is stationary for p (integrate both sides dx).

Irreducibility is harder to state.

Definition: $p(x'|x)$ is π -irreducible if for any set $A \subset \Omega$ with $\int_A \pi(x) dx > 0$, $\Pr(X_n \in A \text{ for some finite } n | X_0 = x) > 0$, so that the chain can hit any set that has finite probability in π .

Note that the last definition is not quite precise—see e.g., Tierney in “Markov Chain Monte Carlo in practice” eds. Gilks, Richardson and Spiegelhalter.

Theorem (ergodicity from reversibility)

Let $q(x)$ be a given probability density on Ω . If $p(x'|x)$ is q irreducible and if p is reversible and aperiodic with respect to q , then $\int_A \pi^{(n)}(x) dx \rightarrow \int_A \pi(x) dx$ as $n \rightarrow \infty$ for any set $A \subset \Omega$ and starting distribution $\pi^{(0)}$.

We can get reversibility using the Metropolis-Hastings prescription as before. Suppose we wish to generate a MC with unique probability density $q(x)$. Let $X_n = x$. X_{n+1} is determined in the following way:

1. Select x' with probability density $g(x'|x)$.
2. Accept x' (i.e., set $X_{n+1} = x'$) with probability

$$\alpha(x'|x) = \min \left\{ 1, \frac{q(x') g(x|x')}{q(x) g(x'|x)} \right\}$$

If x' is not accepted, then set $X_{n+1} = x$.

8.5 Estimating the parameters of a Gaussian Distribution

Suppose that we collect K samples $\mathbf{y} = \{y_1, \dots, y_K\}$ from a normal distribution with mean μ and standard deviation σ . We wish to estimate μ and σ from the sample. This is a somewhat contrived example in that there are only two parameters involved, and it is easy to visualize the posterior probability $p(\mu, \sigma | \mathbf{y})$ without using Monte-Carlo methods. Nevertheless, it is instructive to see how the problem may be solved via sampling from the posterior probability.

By Bayes' theorem,

$$p(\mu, \sigma | y_1, \dots, y_K) \propto p(y_1, \dots, y_K | \mu, \sigma) p(\mu, \sigma) \tag{8.23}$$

The likelihood function is determined by the forward problem. For a single sample from a normal distribution,

$$p(y_i | \mu, \sigma) = \frac{1}{\sigma \sqrt{2\pi}} \exp \left[-\frac{1}{2} \left(\frac{y_i - \mu}{\sigma} \right)^2 \right] \tag{8.24}$$

and so for K independent samples,

$$p(y_1, \dots, y_K | \mu, \sigma) = \frac{1}{\sigma^K (2\pi)^K} \exp \left[-\frac{1}{2} \sum_{i=1}^K \left(\frac{y_i - \mu}{\sigma} \right)^2 \right]. \tag{8.25}$$

Let us consider a prior probability which is the product of a non-informative prior for μ and a non-informative prior for σ . If we initially consider any value of μ as being equally likely as any other, we need to set $p(\mu)$ to be a constant. On the other hand, since σ measures the width of the distribution, it must be non-negative. A possible choice is to make the prior for σ “scale-invariant”. This means that we initially think that

$\Pr(a \leq \sigma < ka)$ to be independent of a . This is equivalent to making the probability density uniform as a function of $\log \sigma$, or $p(\sigma) \propto 1/\sigma$. Note that both of these priors are non-normalizable. This need not be a problem if the likelihood function is sharp enough, as the posterior probability density will be essentially independent of any limits we apply to the ranges of μ or σ in order to make the priors normalizable. We thus choose $p(\mu, \sigma) \propto 1/\sigma$.

Substituting into Bayes' theorem yields

$$p(\mu, \sigma | y_1, \dots, y_K) \propto \frac{1}{\sigma^{K+1}} \exp \left[-\frac{1}{2} \sum_{i=1}^K \left(\frac{y_i - \mu}{\sigma} \right)^2 \right] \quad (8.26)$$

Although this is not normalized, we already have enough information to use the MCMC method to sample from the posterior density.

Let $X_n = (\mu, \sigma)$. X_{n+1} is found as follows

1. Let r_1 and r_2 be drawn from a uniform distribution on $[0, 1]$. Let w_1 and w_2 be positive constants. Set

$$\mu' = \mu + w_1(2r_1 - 1) \quad (8.27)$$

$$\sigma' = \sigma + w_2(2r_2 - 1) \quad (8.28)$$

This means that the proposal density function is

$$g(\mu', \sigma' | \mu, \sigma) = \begin{cases} \frac{1}{4w_1w_2} & \text{if } |\mu' - \mu| < w_1 \text{ and } |\sigma' - \sigma| < w_2 \\ 0 & \text{otherwise} \end{cases} \quad (8.29)$$

Clearly $g(\mu', \sigma' | \mu, \sigma) = g(\mu, \sigma | \mu', \sigma')$.

2. With probability

$$\begin{aligned} \alpha(\mu', \sigma' | \mu, \sigma) &= \min \left\{ 1, \frac{p(\mu', \sigma' | y_1, \dots, y_K) g(\mu', \sigma' | \mu, \sigma)}{p(\mu, \sigma | y_1, \dots, y_K) g(\mu, \sigma | \mu', \sigma')} \right\} \\ &= \min \left\{ 1, \left(\frac{\sigma}{\sigma'} \right)^{K+1} \exp \left[-\frac{1}{2} \sum_{i=1}^K \left\{ \left(\frac{y_i - \mu'}{\sigma'} \right)^2 - \left(\frac{y_i - \mu}{\sigma} \right)^2 \right\} \right] \right\} \end{aligned} \quad (8.30)$$

set $X_{n+1} = (\mu', \sigma')$, otherwise set $X_{n+1} = (\mu, \sigma)$. Note that if $\sigma' < 0$ the posterior probability $p(\mu', \sigma' | y_1, \dots, y_K)$ is zero (since $p(\mu', \sigma') = 0$) and we simply reject the move.

From the samples $\{X_n\}$ drawn from the posterior probability, we can plot histograms or calculate statistics of interest.

The Matlab code to carry out this simulation is:

```
K = 30;
mu = 1; sigma = 2;
y = mu + sigma*randn(K,1);

N = 10000;

m = 0; s = 1;
X = zeros(2,N);
w = 1;
for n = 1:N
    mp = m + w*(2*rand-1);
    sp = s + w*(2*rand-1);
    ratio = (s/sp)^(K+1)*exp(-0.5*sum(((y-mp)/sp).^2-((y-m)/s).^2));
    if sp>0 & rand<ratio
        m = mp; s = sp;
```

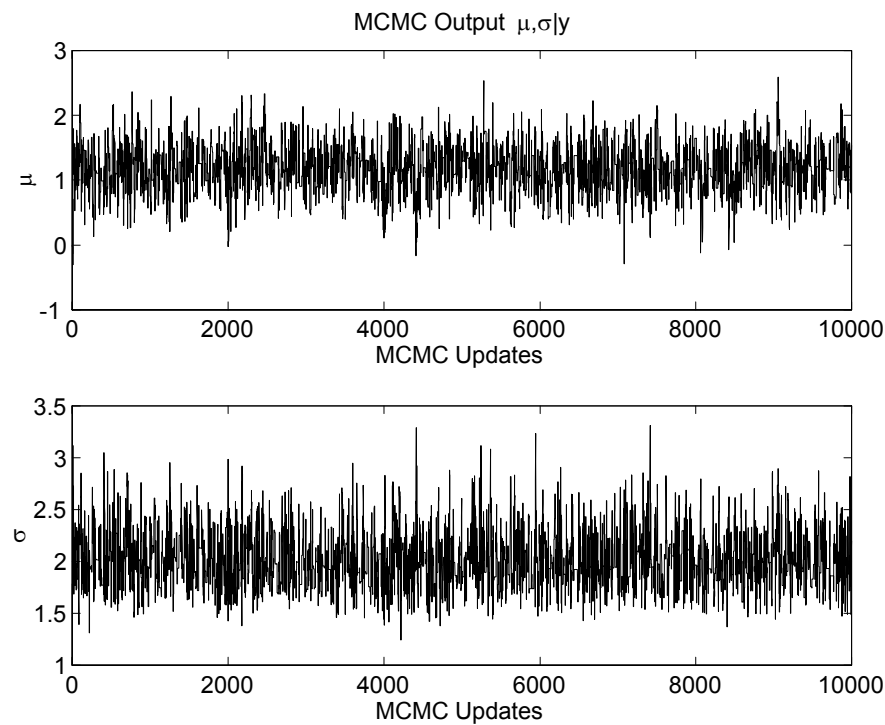


Figure 8.5 Samples from posterior probability distribution for the parameters of a Gaussian obtained from $K = 30$ data points

```

end
X(:,n) = [m;s];
end
figure(1);
subplot(2,1,1); plot(X(1,:))
set(gca,'FontSize',14); xlabel('MCMC Updates'); ylabel('\mu');
title('MCMC Output \mu,\sigma');
subplot(2,1,2); plot(X(2,:))
set(gca,'FontSize',14); xlabel('MCMC Updates'); ylabel('\sigma');
figure(2);
subplot(1,2,1); hist(X(1,:),30)
set(gca,'FontSize',14); xlabel('\mu'); ylabel('Frequency');
subplot(1,2,2); hist(X(2,:),30)
set(gca,'FontSize',14); xlabel('\sigma'); ylabel('Frequency');

```

8.6 Estimating diffusivity D from solution of a partial differential equation

Suppose that $u \equiv u(x, t)$ is the number density of some animal diffusing in the interval $[0, L]$. The animal is killed if it leaves $[0, L]$ so that $u(0, t) = u(L, t) = 0$. Local conservation of number of animals gives the diffusion equation

$$\frac{\partial u}{\partial t} = D \frac{\partial^2 u}{\partial x^2}. \quad (8.31)$$

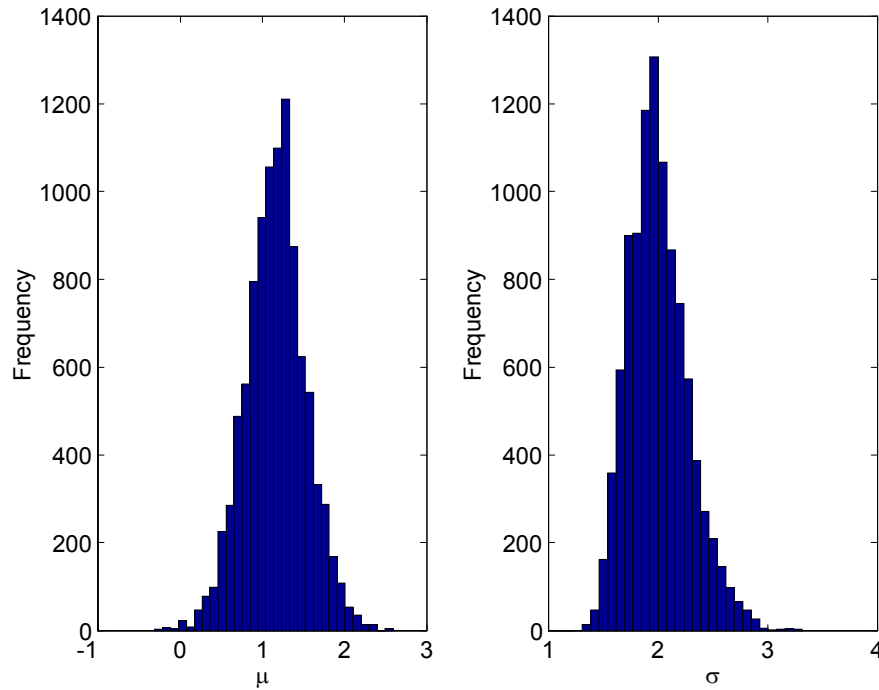


Figure 8.6 Marginal posterior histograms for mean and standard deviation of a Gaussian

At time $t = 0$, the population density is

$$u(x, 0) = \begin{cases} 1 & \text{for } 0.75L \leq x \leq 0.8L \\ 0 & \text{otherwise} \end{cases} \quad (8.32)$$

At time $t = T$, the population density is measured at points x_1, x_2, \dots, x_K . Since the measurements are inexact, the data vector is $\mathbf{y} = (y_i)$ where

$$y_i = u(x_i, T) + \varepsilon_i \quad (8.33)$$

and ε_i are normally distributed with standard deviation s . From these measurements, we wish to sample from the posterior distribution of D , assuming that the prior distribution of D is uniform on $D \geq 0$.

By Bayes' theorem,

$$p(D|\mathbf{y}) \propto p(\mathbf{y}|D)p(D) \quad (8.34)$$

The likelihood function is determined by the noise process. Given the diffusivity is D and given the initial conditions, we may solve the partial differential equation and obtain the expected number density at the locations x_i after a time T , namely $u(x_i, T; D)$. The probability that we measure the data vector \mathbf{y} is

$$p(\mathbf{y}|D) = \prod_{i=1}^K p(\varepsilon_i = y_i - u(x_i, T; D)) \quad (8.35)$$

$$\propto \exp \left[-\frac{1}{2} \sum_{i=1}^K \left(\frac{y_i - u(x_i, T; D)}{s} \right)^2 \right] \quad (8.36)$$

where we have absorbed into the proportionality quantities which do not depend on D .

For the prior, we may choose

$$p(D) \propto \begin{cases} 1 & \text{if } D \geq 0 \\ 0 & \text{otherwise} \end{cases} \quad (8.37)$$

As before, this is improper (i.e., not normalizable), but could be made proper without affecting the analysis, for all practical purposes by imposing a conservative upper bound D_{\max} on D .

The posterior probability density is thus given by

$$p(D|\mathbf{y}) \propto \begin{cases} \exp\left[-\frac{1}{2} \sum_{i=1}^K \left(\frac{y_i - u(x_i, T; D)}{s}\right)^2\right] & \text{if } D \geq 0 \\ 0 & \text{otherwise} \end{cases} \quad (8.38)$$

We can estimate D given \mathbf{y} using sample-based inference.

Let $X_n = D$, X_{n+1} is given in the following way:

1. Let w be a positive constant. Draw r from a uniform distribution on $[0, 1]$ and set $D' = D + w(2r - 1)$
2. With probability

$$\alpha(D'|D) = \min \left\{ 1, \frac{\exp\left[-\frac{1}{2} \sum_{i=1}^K \left(\frac{y_i - u(x_i, T; D')}{s}\right)^2\right]}{\exp\left[-\frac{1}{2} \sum_{i=1}^K \left(\frac{y_i - u(x_i, T; D)}{s}\right)^2\right]} \right\}$$

set $X_{n+1} = D'$, otherwise set $X_{n+1} = D$.

Notice that at each step of the MCMC algorithm, we must compute $u(x_i, T; D)$, (i.e., solve the boundary value problem for a trial value of D) to work out what the solution would have looked like at T if the true diffusivity were given.

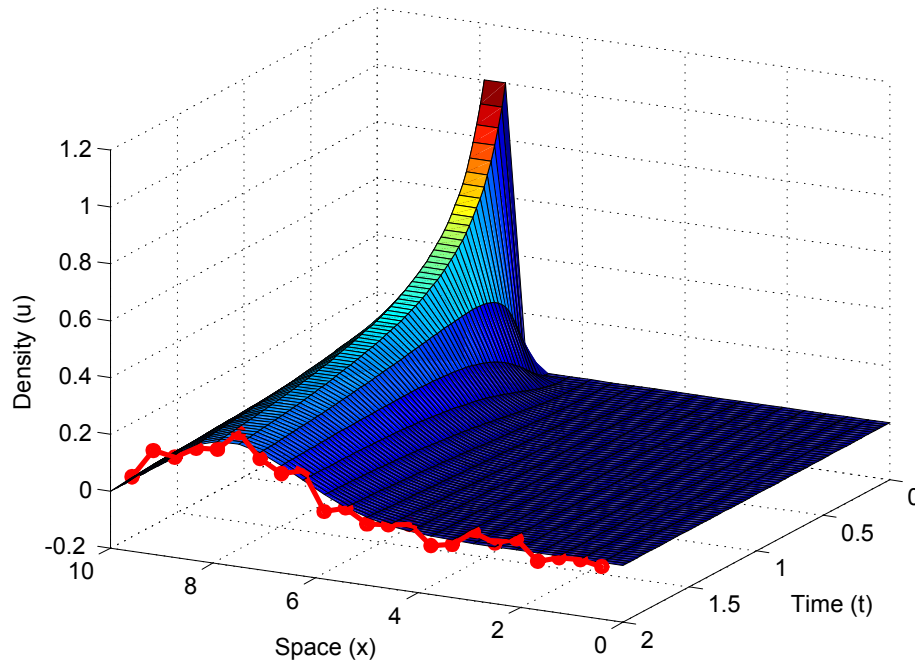


Figure 8.7 Surface is $u(x, t)$ with the true value of D and the points show the data measured at T .

The Matlab code to carry out this simulation is:

```
L = 10;
D = 0.5;
s = 0.03;
Tmax = 2;
xdim = 25; tdim = 75;
x = linspace(0,L,xdim);
t = linspace(0,Tmax,tdim);
```

```

dx = x(2)-x(1); dt = t(2)-t(1);
q = dt/dx^2;
r1 = 0.75*L; r2 = 0.8*L;
u0 = zeros(1,xdim);
u0(find(x>=r1 & x<=r2)) = 1;
xDat = 2:xdim-1;
tDat = tdim;
nxDat = length(xDat);
ntDat = length(tDat);
Z = heat(D,u0,q,tdim);
u = Z(tDat,xDat);
uDat = u + s*randn(ntDat,nxDat);
figure(1); surf(x,t,Z); hold on;
if ntDat>1, mesh(x(xDat),t(tDat),uDat);
else set(plot3(x(xDat),t(tDat)*ones(1,nxDat),uDat,'r-o'),'LineWidth',3);
end; hold off; drawnow
N = 10000; m = 100; XD = 1; X = zeros(1,N); X(1) = XD;
Z = heat(XD,u0,q,tdim);
u = Z(tDat,xDat);
oLLkd = sum(sum(-(u-uDat).^2))/(2*s^2);
LL = zeros(1,N); LL(1) = oLLkd;
w = 0.1;
for n = 2:N
    XDp = XD + w*(2*rand-1);
    if XDp > 0
        Z = heat(XDp,u0,q,tdim);
        u = Z(tDat,xDat);
        nLLkd = sum(sum(-(u-uDat).^2))/(2*s^2);
        alpha = exp(nLLkd - oLLkd);
        if rand < alpha
            XD = XDp;
            oLLkd = nLLkd;
            CZ = Z;
        end
    end
    X(n) = XD; LL(n) = oLLkd;
    if rem(n,m)==0
        figure(2); plot(X(1:n)); drawnow;
        figure(3); surf(x,t,CZ); hold on;
        if ntDat>1, mesh(x(xDat),t(tDat),uDat);
        else set(plot3(x(xDat),t(tDat)*ones(1,nxDat),uDat,'r-o'),'Linewidth',3);
        end; hold off; drawnow
        disp([N/m,n/m]);
    end
end
figure(2); plot(X);

```

The function for solving the diffusion equation is

```

function Z = heat(D,u0,q,tdim)
xdim = length(u0);
Z = zeros(tdim,xdim);
Z(1,:) = u0;
for tin = 2:tdim
    tip = tin - 1;
    Z(tin,2:end-1) = Z(tip,2:end-1) + ...
        D*q*(Z(tip,1:end-2)-2*Z(tip,2:end-1)+Z(tip,3:end));
end

```

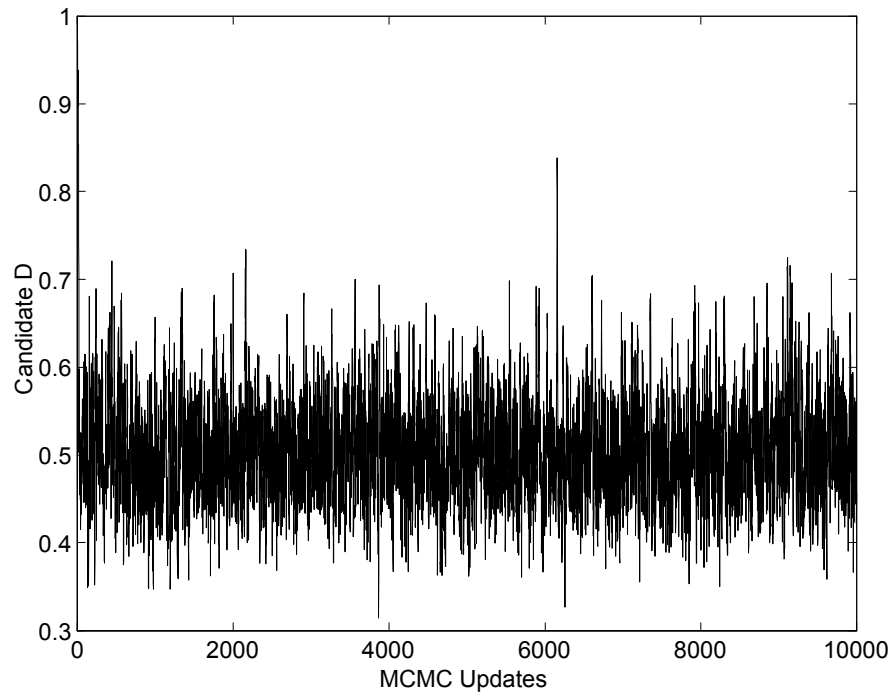


Figure 8.8 Output of Markov chain producing samples from posterior distribution of the diffusivity (True D value was 0.5)

end

8.7 Optimization using Markov Chain Monte Carlo

The Markov chain Monte Carlo method may be used to find the mode of a distribution, which is an optimization problem.

8.7.1 Simulated Annealing

The mode of a distribution is the state of highest probability. In Bayesian analysis, the mode is the state of “maximum à posteriori probability” (the MAP state) and is often presented as the final “answer” in the analysis. This is generally unsatisfactory as it tells us nothing about the degree of uncertainty which our limited data admits to the reconstructed system state. Nevertheless, we are often interested in finding the mode of a function when the state space is large and the distribution is a complicated function of the state.

Let $Q(x)$ be a given probability density on a space Ω . Let Q_{\max} be the maximum value of Q in Ω . Let $\Gamma = \{x : Q(x) = Q_{\max}, x \in \Omega\}$ be the mode set or the set of modal states, (i.e., there may be more than one). We write

$$Q(x) = \exp[-q(x)] \quad (8.39)$$

where $q(x) = -\ln Q(x)$ and introduce a “temperature” parameter T such that

$$Q_T(x) = \frac{\exp[-q(x)/T]}{\mathcal{Z}_T} \quad (8.40)$$

where \mathcal{Z}_T is an unknown normalizing constant.

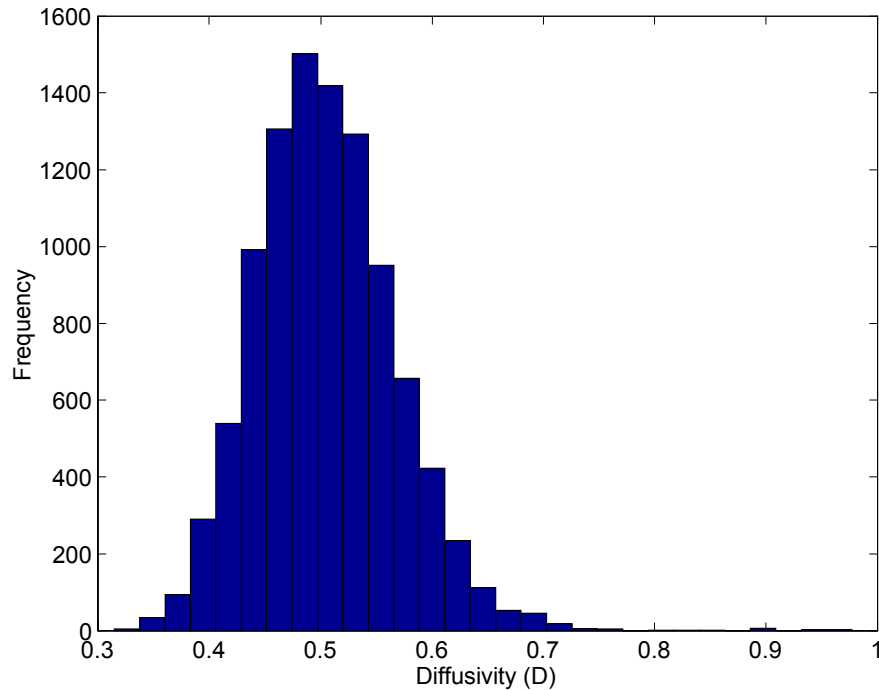


Figure 8.9 Posterior distribution of diffusivity. True D value was 0.5

Consider two states x and $x' \in \Omega$ such that x is a mode (i.e., $x \in \Gamma$) but x' is not modal (i.e., $x' \notin \Gamma$). Since $Q(x') < Q(x)$, $q(x') > q(x)$ and so the ratio

$$\frac{Q_T(x')}{Q_T(x)} = \exp \left[\frac{q(x) - q(x')}{T} \right] \quad (8.41)$$

tends to zero as $T \rightarrow 0$. Hence as $T \rightarrow 0$, all probability mass in distribution Q_T is concentrated on states in Γ : we can find modal states (or state) by sampling $Q_T(x)$ at small T , using MCMC.

Roughly, the idea is to run a MCMC algorithm with equilibrium distribution $\pi_T = Q_T$. If the chain is in equilibrium at temperature T (ie $\pi^{(n)} = Q_T$) at step n and we lower the temperature to $T' < T$, we must then wait for the chain to equilibrate at the new temperature. We lower the temperature slowly, so that the chain remains in or close to equilibrium with Q_T at each new temperature. At small T $\pi^T \sim 1/|\Gamma|$ and our MCMC returns a sample uniform on the mode states.

Suppose an ergodic Metropolis-Hastings algorithm with generation distribution $g(x'|x)$ is given. The algorithm simulates some Markov chain with states $x \in \Omega$, initializing distribution $X_0 \sim \Pr\{X_0 = x\}$ and equilibrium distribution $Q(x) = \exp(-q(x))/\mathcal{Z}$. Specify $T(n)$, a decreasing function of n called the **cooling** or **annealing schedule**. The following **Simulated Annealing Algorithm** simulates an inhomogeneous Markov Chain.

Let $X_n = x$. X_{n+1} is determined in the following way.

1. Generate candidate state x' from $g(x'|x)$
2. With probability

$$\alpha = \min \left\{ 1, \exp \left[- (q(x') - q(x)) / T(n) \right] \frac{g(x|x')}{g(x'|x)} \right\}$$

set $X_{n+1} = x'$, otherwise, set $X_{n+1} = x$.

Definition 1 If $\pi^{(n)}$ becomes uniform on Γ as $n \rightarrow \infty$, ie if $\Pr(x^{(n)} \in \Gamma) \rightarrow 1$ as $n \rightarrow \infty$, we say that the annealing “converges”.

Definition 2 Let $Q(x)$, $x \in \Omega$ be a given distribution with mode-set Γ . A state $x \in \Omega$ communicates with Γ at height h if there is a path from x into Γ with the property that the largest value of $q = -\ln Q$ along the path is $q(x) + h$.

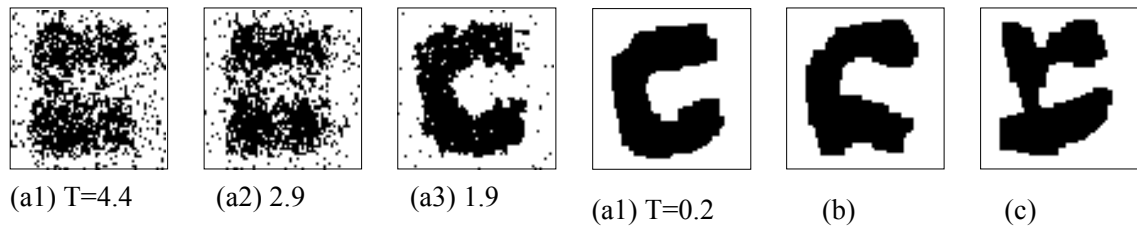


Figure 8.10 Estimating the mode of the posterior for the matrix estimation problem of Section 8.3. Sequence (a1-4) annealing with $r = 0.9999995$ and $T_0 = 10$. (b) Same cooling schedule, different random number seed. (c) annealing with $r = 0.999999$ and $T_0 = 2$.

Theorem 3 *Annealing theorem (Hajek, 1988)*

Let $Q(x)$, $x \in \Omega$ be a given distribution with mode-set Γ . Let d^* be the smallest number such that every $x \in \Omega$ communicates with Γ at height d^* . The simulated annealing algorithm converges if and only if

$$\sum_{n=1}^{\infty} \exp[-d^*/T(n)] = \infty$$

Corollary 4 *If $T(n) = d/\log(n)$ then simulated annealing converges iff $d \geq d^*$.*

Simulated annealing is an optimization algorithm. If $q(x)$ is a cost function on the states $x \in \Omega$, simulated annealing can be used to minimize $q(x)$. Refer to the “travelling salesperson” problem.

Although the simulated annealing algorithm with schedule $T(n) = d/\log(n)$ is guaranteed to converge, it does so too slowly to be of practical value. Also, d^* is hard to estimate. More rapid cooling is usually done. The output is checked by repeating the experiment with a varying random number seed, and making sure that we get the same answer each time.

There is a physical analogy. The process simulates the physical process of “annealing”, i.e., slowly cooling, a material to get special, highly ordered crystalline states.

Example 5 *Using simulated annealing, find the maximum a posteriori state of the matrix reconstruction problem of section 8.3 with the binary Markov random field prior.*

Q is our posterior $Q(x) = \Pr(D = x|g, \sigma)$. We revise the algorithm of section ??, introducing a factor $1/T(n)$ in the power of the exponential. We take as an annealing the schedule the unreliable geometric schedule $T(n) = r^n T_0$ with $0 < r < 1$ a real constant and T_0 an initial, top temperature. In the notation of Section ??, the algorithm is as follows.

Let $X_n = x$. X_{n+1} is determined in the following way.

1. Given $x = (x_{1,1}, x_{1,2}, \dots, x_{N,N})$, pick one of the N^2 pixels (m, n) at random. Set $x' = (x_{1,1}, x_{1,2}, \dots, x_{m,n}, \dots, x_{N,N})$.
2. With probability

$$\alpha(x'|x) = \min \left\{ 1, \exp \left(-2(J\#\Delta_{mn} + \frac{x_{mn}}{\sigma^2} (r_m + c_n - [x]_m - [x]_n + 2x_{mn})) / T(n) \right) \right\}$$

set $X_{n+1} = x'$. Otherwise we set $X_{n+1} = x$.

Sample output is shown in Figure 8.10. If we compute the ratio of the posterior probabilities for states 8.10(a4) and (b), $\Pr(D = (a4)|g, \sigma) / \Pr(D = (b)|g, \sigma)$ we obtain a value around unity. However states 8.10(a4) and (c) have a posterior ratio of around $\exp(-655) / \exp(-590) \sim 10^{-28}$. If the cooling schedule descends too rapidly, the MCMC becomes stuck in the unrepresentative, sub-optimal state Figure 8.10(c).