Statistical and Learning Techniques in Computer Vision
Lecture 6: Markov Chain Monte Carlo and Gibbs Sampling

Jens Rittscher and Chuck Stewart

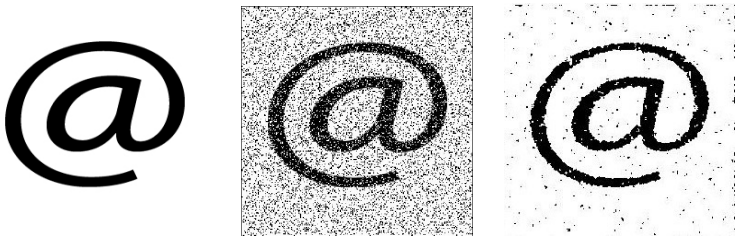

Figure 1: **Noisy binary image.** *The left image displays the a binary image containing no noise. The given noisy image, I_0 is shown in the middle. Here 20% of all pixels are changed. Median filtering is one possibility of removing noise. The result of applying a median filter (3×3 mask) is shown on the right.*

1 Outline

- Review of MRFs in the context of de-noising
- Goal: MAP estimation of MRFs
- Ideas and discussion of the challenges
- Sampling techniques
- Markov Chain Monte Carlo
- Gibbs sampling

2 Review of MRFs for Denoising

Our goal is to produce the best reconstructions of an image given a noisy input image I_0. We write any possible reconstruction of the image as a random vector $\mathbf{I}$ of pixel values. The best reconstruction is the one that maximizes the posterior probability

$$p(\mathbf{I}|I_0) = p(I_0|\mathbf{I})\ p(\mathbf{I}) \tag{1}$$

This posterior probability is constructed as a Markov Random Field (MRF).

More specifically, the random variable $\mathbf{I} = \{\mathbf{i}_p\}$ represents a sample image (vector of pixel values). Remember, $\mathbf{I}$ is in our case a large set of random

variables. The random variable $\mathbf{i}_p$ represents the intensity at an individual pixel in an image.

Prior. The smoothness assumption was modelled by the prior distribution:

$$p(\mathbf{I}) = \frac{1}{Z} \exp\left(-\alpha \sum_{<s,p>} |\mathbf{i}_s - \mathbf{i}_p|^2 \right) , \tag{2}$$

where Z is the normalization constant and $<s,p>$ are the individual cliques of the neighbourhood system.

Evidence. The fact that the sample image actually resembles a clean version of the given noisy image I_0 is enforced by the "evidence" of the from

$$p(\mathbf{I}) = \frac{1}{Z} \left(-\beta \sum_{s} |\mathbf{i}_p - \mathbf{i}_p^0|^2 \right) . \tag{3}$$

Posterior. The posterior can be written as

$$p(\mathbf{I}|I_0) \propto exp\left(-\alpha \sum_{<s,p>} |\mathbf{i}_s - \mathbf{i}_p|^2 - \beta \sum_{s} |\mathbf{i}_p - \mathbf{i}_p^0|^2 \right) \tag{4}$$

In effect, the "best" reconstruction of the image is a trade-off between staying close to the original image and building a smooth result image.

3 Goal: MAP estimation of MRFs

We now want to use this model to produce the de-noised image by computing the MAP estimate I^*, such that

$$I^* = \max_I p(I|I_0) . \tag{5}$$

In other words we want to find the values of the random variables (the pixels) that produce the most-probable final image. Our goal for this lecture and for the next several lectures is to find a way to do this. We will start with *sampling* techniques, since these were the earliest methods for MAP estimation of MRFs. Also, although they have been supplanted by other estimation techniques for MRFs in image analysis, they are still used in practice in many other contexts.

4 Intuitions and Ideas

4.1 Gradient Ascent

Before introducing sampling, it is important to consider why we do not try a more direct method, like gradient ascent. The problem is that the posterior

has many, many local maxima! Gradient ascent starts from one location — e.g. the input image for the denoising problem — and repeatedly (a) computes the gradient, (b) takes a small step in the gradient direction (e.g. makes a small change in the values of the image pixels), and (c) tests to see if a maximum has been reached. This is only likely to reach the local maximum on the "hill" that the initial estimate started on.

4.2 Sampling: The Intuition

The idea of sampling is to generate a set of one or more possible values (samples) of the random vector and evaluate the posterior on these, choosing the sample that maximizes the posterior. This could be repeated several or even many times, until the best possible sample is selected and retained, and the MAP estimation problem is solved. Each set of samples (especially the best ones) could be used as the basis for choosing the next set of samples.

The gradient ascent algorithm could be viewed as a special case of sampling, with the gradient direction being used to generate the sample from the current sample. We would like to do better than this in several ways:

- We want to choose samples according to the posterior distribution, meaning that the higher the probability the more likely a sample is to be chosen. In a way, gradient ascent does this, but we would like to do better.
- We may want to generate and evaluate multiple samples at once.
- We do not necessarily want to choose samples that are close to the current sample, but if we jump too far from the current sample, most of our generated samples may be bad.

There are several associated problems, which will be explained as we proceed:

- It is hard to generate samples directly from our posterior distribution. This is true of almost all distributions.
- The search space is enormous for most problems of interest.

Research on sampling-based estimation algorithms focuses on addressing these two problems.

4.3 Changing the Shape of the Posterior

Here is one additional idea, not directly related to sampling (yet), that will help solve our MAP estimation problem. This has been used for a number of different estimation problems.

Consider the following family of distributions

$$p_T(\mathbf{I}|I_0) \propto [p(\mathbf{I}|I_0)\}^{\frac{1}{T}}] \ , \tag{6}$$

where $T > 0$. As illustrated in figure 2 we have

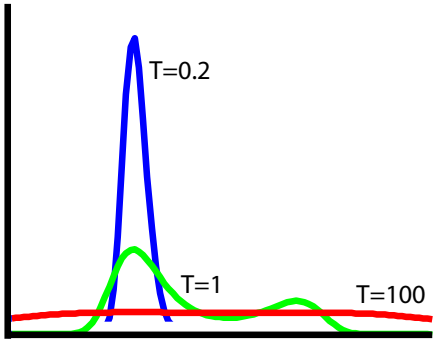

Figure 2: **Simulated Annealing.** *Shown is a distribution p_T for different values of T. In case T is large the distribution tends to be uniform. If, on the other hand T is very small then p_T is sharply peaked and the peak around the maximum of p_T with $T = 1$.*

- in the limit
$$\lim_{T \to \infty} p_T \tag{7}$$
is a uniform distribution
- in case $T = 1$ we have
$$p_T(\mathbf{I}|I_0) = p(\mathbf{I}|I_0) \tag{8}$$
- if T becomes very small then $p_T(\cdot)$ is sharply peaked around I^* the maximum of $p(\mathbf{I}|I_0)$.

Based on these properties, here is an outline of a non-sampling based algorithm to solve our problem:

1. Set T very high and find the $\mathbf{I}$ that maximizes $p_T(\mathbf{I}|I_0)$, starting from the initial estimate, $\mathbf{I} = I_0$.
2. Repeat
   (a) Reduce T
   (b) Use gradient ascent to maximize $p_T(\mathbf{I}|I_0)$ starting from the estimate of $\mathbf{I}$ from the previous T.
3. Until $T \downarrow 0$.

As described, this idea is the basis for both a non-stochastic method called "graduate non-convexity" and a stochastic (probabilistic) method called *simulated annealing*.

4.4 Putting These Together: Sequential Sampling

We cannot generate samples directly from $p_T(\mathbf{I}|I_0)$. But we will see that it is possible to generate a sequence of samples

$$I^{(1)}, I^{(2)}, I^{(3)}, I^{(4)}, \ldots \tag{9}$$

where the first N_1 samples are generated from a distribution

$$I^{(k)} \sim Q_{T_1}(\mathbf{I}, I^{(k-1)}) := p_{T_1}(\mathbf{I}|I^{(k-1)}, I_0) \tag{10}$$

the next N_2 samples will be generated from Q_{T_2}, with $T_2 < T_1$ and so on. The sequence $T_1, T_2, T_3, \ldots$, is usually called an *annealing schedule*. Because of the local characteristics of the MRF the expression Q_{T_1} is simple enough to formulate an effective sampling scheme.

Our desired MAP estimate I^* is then computed as a careful combination of sequential sampling and annealing. Once the temperature T is low enough we will have generated a sample from a very sharply peaked distribution. Provided sufficient care was taken we should have

$$\lim_{T\downarrow 0} \lim_{k\uparrow\infty} I^{(k)} = I^* \ . \tag{11}$$

This is more or less the grand idea of treating MRF's using sampling or what are known as Markov chain Monte Carlo methods. We won't be able to review any of the convergence properties of this method. But in the following we will highlight certain mathematical aspects of this technique since they evolved into important tools that can be also be applied in different application contexts.

We now turn to more of the details.

5 Sampling Techniques

In general, sampling techniques are important tools to run simulations in areas such operations research (simulating queues, etc.) and financial modeling. For numerical computations they are essential in case models are too complex to be analyzed analytically. We will, for example, use sampling techniques to

(a) generate samples $\mathcal{X} = \{x_1, x_2, \ldots, x_N\}$ from a given probability distribution $p(\mathbf{x})$.

(b) estimate expectations of functions under this distribution for example

$$\mathbf{E}[\psi(\mathbf{x})] = \int \psi(\mathbf{x}) p(\mathbf{x})\, dx \ , \tag{12}$$

This is an instance of what is referred to as Monte Carlo integration.

Note that here we generate one set of samples, evaluate the result, and quit. We have not generated a sequence of samples, as outlined above, where each sample in the sequence depends on the previoius sample. We will try to keep the difference between sets of samples and sequences of samples clear, but many of the issues in generating samples pertain to both.

5.1 Why is sampling hard?

Let us consider the following two points:

- **Normalizing constant.** The normalizing constant Z is usually hard to find, i.e.
$$Z = \int p(\mathbf{x}) dx \tag{13}$$
- **High dimensional spaces.** Sampling a random variable $\mathbf{X}$
$$\mathbf{x} = (\mathbf{x_1}, \ldots, \mathbf{x_M}) \tag{14}$$
according to some know density $p(\mathbf{x})$ is hard if M is very large. While it is possible for Gaussian distributions this no longer holds for general distributions. We will illustrate this fact using and example.

Example 5.1 Estimate the normalization constant for the following distribution
$$p(\mathbf{x}) = \exp(0.4 * (\mathbf{x} - 0.4)^2 - 0.008\mathbf{x}^4) \tag{15}$$
In case $\mathbf{x} \in \mathcal{R}^1$ about 50 sample points are necessary to estimate $\hat{Z}$. In case $\mathbf{x} \in \mathcal{R}^{1000}$ we would need to evaluate about 50^{1000} points. A graph of this density is shown in figure 3.

5.2 Uniform Sampling

An important idea is to generate samples from uniform distribution and transform these samples. Let $f(\cdot)$ be the cumulative distribution of the given probability density $p(\mathbf{x})$. The method is used as follows

- draw K samples $y_1, \ldots, y_K$ from a uniform distribution with range $[0, 1]$.
- compute samples $x_1, \ldots, x_K$ according to
$$x_k = f^{-1}(y_k) \tag{16}$$

The set of samples $x_1, \ldots, x_K$ will be distributed according to $p(\mathbf{x})$. The basic principle is illustrated in figure 3. It is problematic to apply this method for high-dimensional random variables. The mass of the probability density is often concentrated in very small regions. For a more detailed discussion refer to [Mac98].

5.3 Importance Sampling

Similar to the idea of uniform sampling we generate the samples (a set of samples) using a simpler distribution. In a certain way this technique can be viewed

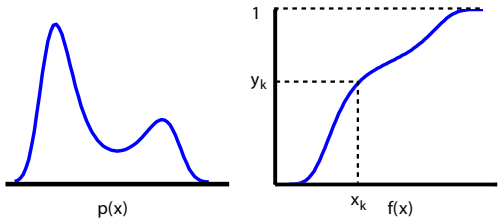

Figure 3: **Uniform sampling.** *The given probability density $p(\mathbf{x})$ is shown on the left. The figure on the right illustrates the sampling technique. The sample y_k is generated from a uniform distribution. The inverse of the cumulative density $f^{-1}(\cdot)$ is then used to generate x_k.*

as a generalization of uniform sampling. The samples are generated from a *sampling density* or *proposal distribution* $g(\mathbf{x})$. In order to account for the fact that they are not sampled from the original distribution an *importance weight*

$$w(x_p) = \frac{p(x_p)}{g(x_p)} \tag{17}$$

is associated with every sample every sample x_p. The reason behind this is that where $g(x) > p(x)$, our sampling method will tend to generate too many samples. To compensate for this, we downweight such samples. This only make sense when we are generating a set of samples rather than a sequence.

Also refer to the illustration in figure 4.

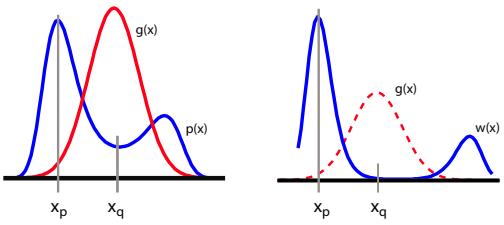

Figure 4: **Importance sampling.** *As opposed to sampling from the original distribution $p(\mathbf{x})$ we sample from a sampling density or proposal distribution $g(\mathbf{x})$. A probability density $p(\mathbf{x})$ and proposal distribution $g(\mathbf{x})$ are shown on the left. The introduced importance weights $w(\mathbf{x})$ are shown on the right as a function of $\mathbf{x}$. Notice that although sample x_q has a very low importance weight it will be selected far more frequently than sample x_p.*

In certain practical applications importance sampling can be incredibly useful. Suppose one needs to estimate the probability $\theta = E[\phi(\mathbf{x})]$ for an observation $\mathbf{x}$ on a random system. Then some outcomes of $\mathbf{x}$ may be more important

than others in determining θ and it would be advantageous to select such values more frequent. For example θ can be the probability of a very rare event. The only way to estimate θ accurately may be to produce the rare events more frequently. Therefore one can simulate a model which gives the probability distribution g to $\mathbf{x}$ rather than the correct pdf p. This

$$\hat{\theta}_g = \frac{1}{W} \sum_i \psi(x_i) \frac{p(x_i)}{g(x_i)} \quad \text{with} \quad W = \sum_i \frac{p(x_i)}{g(x_i)} \tag{18}$$

is an unbiased estimator for θ. It can be shown that this technique can be used to reduce the variance of an estimator.

In class we will discuss this with a very practical example. Before we move on a word of warning: One needs to be careful when applying importance sampling techniques. They only work well of the proposal density $g(\mathbf{x})$ is similar to the probability density $p(\mathbf{x})$

6 Markov Chain Monte Carlo

Now we are going to return to the issue of generating a sequence of samples, with each sample depending on the previous sample, rather than generating a set of samples $x_1, \ldots, x_N$ as with uniform or importance sample. In particular, we try to generate a sequence of random samples

$$x^{(1)}, x^{(2)}, x^{(3)}, \ldots \tag{19}$$

by recording the current state $x^{(k)}$ and then generating the next sample $x^{(k+1)}$ such that

$$x^{(k+1)} \sim p(\mathbf{x}|x^{(k)}) \ . \tag{20}$$

7 Markov Chains

Definition 7.1 A *Markov chain* is a sequence of random variables $\mathbf{x}^1, \mathbf{x}^2.\mathbf{x}^3 \ldots$. such that the present state $\mathbf{x}^k$ only depends on the previous state, i.e.

$$p(\mathbf{x}^{k+1}|\mathbf{x}^k, \mathbf{x}^{k-1}, \ldots, \mathbf{x}^0) = p(\mathbf{x}^{k+1}|\mathbf{x}^k) \ . \tag{21}$$

The probabilities $p(\mathbf{x}^{k+1}|\mathbf{x}^k)$ are commonly referred to as transition probabilities $Q(\mathbf{x}^{k+1}|\mathbf{x}^k)$. A Markov chain is called *homogeneous* if the transition probabilities are the same for all k.

From now on we assume that the Markov chain is homogeneous. We have the following marginal probability

$$p(\mathbf{x}^{t+1}) = \sum_{\mathbf{x}^t} Q(\mathbf{x}^{t+1}|\mathbf{x}^t) p(\mathbf{x}^t) \ . \tag{22}$$

We construct the Markov chain such that

(a) The desired distribution $p(\mathbf{x})$ is the *invariant distribution* of the chain. A distribution is an invariant distribution of

$$\pi(\mathbf{x}') = \int Q(\mathbf{x}', \mathbf{x})\pi(\mathbf{x})\, d\mathbf{x} \ . \tag{23}$$

(b) The chain is also *ergodic*, that is,

$$\lim_{k\uparrow\infty} p^k(\mathbf{x}) = \pi(\mathbf{x}') \tag{24}$$

for any initial distribution $p^0(\mathbf{x})$.

If the state space or the range of the random variables $\mathbf{x}^t$ is finite then the transition probabilities $Q(\mathbf{x}^{t+1}|\mathbf{x}^t)$ are represented by a matrix, called the transition matrix Q. The element (i, j) of the matrix P is equal to

$$p_{ij} = \text{prob}(\mathbf{x}_{t+1} = j|\mathbf{x}_t = i) \ . \tag{25}$$

We will now show that

$$P^{n+m}_{i,j} = \sum_k P^n_{i,k} P^m_{k,j} \ , \tag{26}$$

where $P^c_{i,j} := \text{prob}(\mathbf{x}_{t+c} = j|\mathbf{x}_t = i)$.

8 Gibbs Sampling

As before we aim to generate a sequence of random variables $x^{(1)}, x^{(2)}, x^{(3)}, x^{(4)}, \ldots$. In case $\mathbf{x}$ is a vector of random variables, i.e.

$$\mathbf{x} = (\mathbf{x_1}, \ldots, \mathbf{x_M}) \tag{27}$$

it is not necessarily possible to generate the new sample $x^{(k+1)}$ directly from the transition probability

$$Q(\mathbf{x}, x^{(k)}) \ . \tag{28}$$

We will generate the new sample $x^{(k+1)}$ by 'updating' the different components in separate step.

The *Gibbs sampler* is given as:

(a) Initialize $x^{(0)}_1, \ldots, x^{(0)}_M$.

(b) For $k = 1, \ldots, K$:

- Sample $x^{(k)}_1 \sim p(\mathbf{x_1}|x^{(k)}_2, \ldots, x^{(k)}_M)$
- Sample $x^{(k)}_2 \sim p(\mathbf{x_2}|x^{(k+1)}_1, x^{(k)}_2 \ldots, x^{(k)}_M)$
- ...
- Sample $x^{(k)}_i \sim p(\mathbf{x_i}|x^{(k+1)}_1, \ldots, x^{(k+1)}_{i-1}, x^{(k)}_{i-1}, \ldots, x^{(k)}_M)$

- ...
- Sample $x_M^{(k)} \sim p(\mathbf{x_M}|x_1^{(k+1)}, \ldots, x_{M-1}^{(k+1)})$

In section 2 we designed a posterior probability for the problem of image restoration given a noisy image I_0. The joint probability distribution $p(\mathbf{I}|I_0)$ as modelled using Markov random field. Notice that the Gibbs sampler now allows us to generate a sequence of images

$$I^{(1)}, I^{(2)}, I^{(3)}, I^{(4)}, \ldots \tag{29}$$

where the probabilities

$$p(\mathbf{i_j}|i_1^{(k+1)}, \ldots, i_{j-1}^{(k+1)}, i_{j-1}^{(k)}, \ldots, i_M^{(k)}) \tag{30}$$

are tractable because the make use of the local Markov probability. Time does not permit to give details the combination of Gibbs sampling in combination with the idea of simulated annealing using simulated annealing permits to compute the MAP estimate I^*, the de-noised image. Details can be found in [GG84], although we must warn you that this is a very difficult paper to read.

9 Further Reading

- The two original papers which are of high importance are [GG84] and [Bes86]. Geman and Geman introduce the notion of the Gibbs sampler and prove some of its convergence properties.
- Bishop gives a broad review of different sampling methods [Bis06, Chapter 11]. The chapter introduces a number of other sampling methods that are of use in practice.
- MacKay introduces the concept of Metropolis and Gibbs sampling together with other more basic sampling methods in [Mac98]. In addition to introducing the concepts he also points out drawbacks each of the different methods have.

References

[Bes86] J. Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society. Series B*, 48(3):259–302, 1986.

[Bis06] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[GG84] S. Geman and D. Geman. Stochastic relaxation, gibbs distribtutions, and the bayesian restauration of images. *IEEE. Trans. Pattern Anal*, 6(6):721–741, November 1984.

[Mac98] D.J.C. MacKay. Introduction to monte carlo methods. In M. I. Jordan, editor, *Learning in Graphical Models*, pages 175–204. Kluwer Academic Publishers, 1998.