Chapter III
Image Modeling

Chapter Contents:

1 Problem Statement

The theory of random processes provides a fundamental framework to study the properties of random signals, once a statistical model has been defined. The statistical model consists of a probability space $(\Omega, \mathscr{F}, P)$ and a function X mapping any outcome $\omega \in \Omega$ to a signal $X(\omega)$. In contrast, statistical signal modeling may be thought of as "reverse engineering": construct a statistical model based on physical considerations, empirical evidence, perhaps even vague beliefs. This raises three questions:

1. Why can images be viewed as random?
2. How can one construct a statistical model for images?

3. How can the model be validated?

These questions are addressed in Sections 1—3. Detailed model constructions are studied in Sections 4—10.

First, why can images be viewed as random? This question finds a natural answer in view of the *uncertainty* an observer has about the image before seeing it. In the absence of uncertainty (e.g., if our observer was already informed s/he would be shown the standard *Lena* picture), no statistical model is needed.

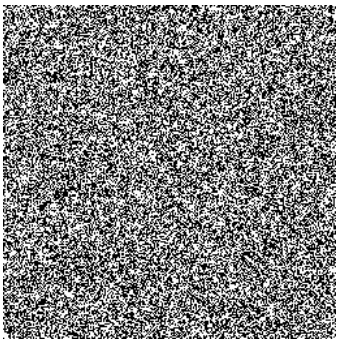
Second, how can one construct a statistical model for images? Let us start with a simple example. Taking a helicopter ride above Manhattan and taking pictures, we may view any such picture as the realization $X(\omega)$ of a random process, where ω is an outcome from a sample space Ω. We would like to construct a reasonable probability space $(\Omega, \mathscr{F}, P)$.

A possible view of this problem is that the set of all pictures of Manhattan viewed from the skies is already available from Google Earth, and therefore the only source of uncertainty resides in the specific location and orientation of our camera in the sky, as well as a few camera-specific parameters (such as the zoom). We may view these parameters as *geometric parameters*. Their value constitutes an outcome ω, and Google Earth associates an image $X(\omega)$ to that outcome. The construction of a probability space $(\Omega, \mathscr{F}, P)$ might be relatively straightforward in this case.

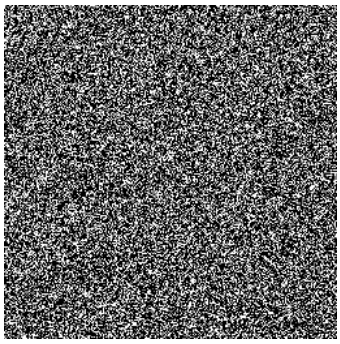
A weakness of this argument is that Google Earth has limited spatial resolution, and our high-resolution digital camera is able to pick up details at a finer scale. The uncertainty in our image would then come from the nature of these fine-scale details (in addition to the geometric parameters discussed above). We might be able to construct reasonable probabilistic models for these detail image components. Our discussion of statistical texture analysis later in this chapter provides some examples of this approach.

In the absence of Google Earth or other reference database, we would still be left with the problem of modeling the low-resolution version of our pictures. These pictures contain considerable geometric structure, in the form of buildings, rooftops, streets, cars, etc. The problem of constructing a model for Ω that adequately captures image structure is a fundamental problem in image analysis and computer vision. Assigning a probability measure on Ω is challenging as well.

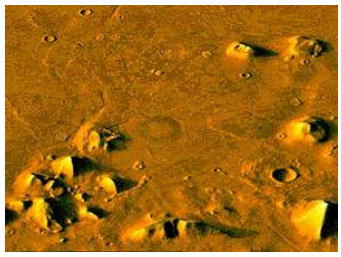
Exercise: Study the nine images shown in Fig. 1. Define sample spaces from which these images are drawn. Discuss whether realistic probability distributions over these sample spaces should have the properties of stationarity, isotropy, and ergodicity.



(a) iid binary image, 2 equiprobable values (0 and 255)

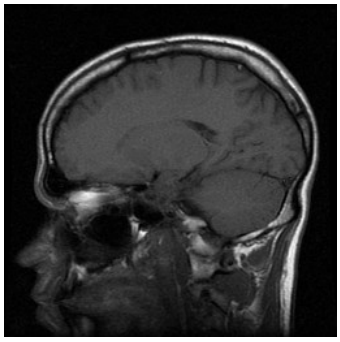


(b) iid binary image, 2 equiprobable values (0 and A), A = binary r.v. with 2 equiprobable values (127 and 255)



(c) section of Mars surface

(d) fingerprint



(e) MRI brain image

(f) soccer crowd

(g) head butt

(h) lego head butt

(i) building

Figure 1: Are these images realizations of stationary, isotropic, and/or ergodic processes?

2 Model Estimation

To construct a probability measure on Ω, we may collect images and view the selection of P as a problem of *statistical inference*. In our example, denote by N the number of camera pixels; the image process X is just a random N-vector with pdf $p(X)$. Taking a large number of pictures with the camera, we obtain a set of K pictures $x^1, \cdots, x^K$, called *training set*. From this set, we would like to estimate the N-dimensional pdf $p(X)$.

At first sight this method is doomed to failure. Indeed, we have only KN data. To reliably estimate the pdf of a *scalar random variable* (as is the case if $N = 1$), we need, say at least 10 independent training data. To reliably estimate the joint pdf of *two* random variables, we need about 100 training data. To reliably estimate a N-dimensional pdf, we would in general need of the order of 10^N independent training data, which is an astronomical number as soon as N is moderately large. This problem is generally known as the *curse of dimensionality*.

This argument ignores the fact that images have structure, and therefore $p(X)$ has structure as well – it cannot be just *any* N-dimensional pdf. As discussed above, statistical image models express uncertainty about geometric structure and texture, a statistical description of which is relatively low-dimensional. For instance, we may formulate a parametric model $p_\theta(X)$ for the images, where θ is a parameter in a low-dimensional set Θ. Once the family of pdf's $\{p_\theta, \theta \in \Theta\}$ is fixed, constructing a probability measure on Ω is equivalent to selecting $\theta \in \Theta$. Several examples of this parametric approach will be given in this chapter. A crude one for textured areas is that of i.i.d. Gaussian pixels with mean μ and variance σ^2. Then we have $\theta = \{\mu, \sigma^2\}$.

Even when a good model for Ω is available, the limited size of training sets makes it difficult to construct a reliable estimate of the probability measure on Ω. To add to the challenge, there is generally no guarantee that the training set is representative of *all* possible images of interest.

In summary, the difficulties involved in image modeling are:

1. Definition of an appropriate sample space Ω;
2. Limited size of training sets;
3. The model should be computationally tractable;
4. The model should be robust (generalize to new images).

Despite the enormous complexity of the modeling problem, the quality of the best known models has dramatically improved over the last twenty years, as will be evident from results obtained in a variety of applications.

3 Model Validation

We now address the issue of validating a candidate statistical model. The problem can be set up in different ways, but the basic one is: given two competing probability measures P_1 and P_2, which one is a better fit to reality?

A simple attempt to answer this question is to generate realizations from P_1 and P_2 and decide which ones "look" more similar to x. This method is quite subjective but is sometimes used in texture analysis. In general however, even the best models lack visual realism, and this approach is not satisfactory.

From a scientific perspective, approaches based on the likelihood principle (Sections 3.1–3.3) and on empirical cost minimization (Section 3.4) are more attractive.

3.1 Likelihood Principle

The basic problem is to decide whether data x were generated from a distribution P_1 or from another distribution P_2. The likelihood principle states that all relevant experimental information is contained in the "odds" $\frac{P_1(x)}{P_2(x)}$, also called likelihood ratio.

If the models P_1 and P_2 are a priori equally likely, then according to the likelihood principle, we should select the model that maximizes the likelihood ratio.

Here is an example of this approach. We are given a sequence of coin tosses and asked whether the coin is fair (model P_1) or is biased in favor of Heads with probability 0.9 (model P_2). Say we observe the following sequence of six independent coin tosses:

$$x = \{H, T, H, H, H, T\}$$

(four occurrences of Heads and only two occurrences of Tails). We compute

$$\begin{aligned} P_1(x) &= \left(\frac{1}{2}\right)^6 = \frac{1}{64}, \\ P_2(x) &= \left(\frac{1}{10}\right)^2 \left(\frac{9}{10}\right)^4 = \frac{6561}{1,000,000}. \end{aligned}$$

The odds in favor of the fair-coin hypothesis are

$$\frac{P_1(x)}{P_2(x)} = \frac{1,000,000}{64 \times 6561} \approx 2.38.$$

Hence we accept this hypothesis and reject the other one.

The quantities $P_1(x)$ and $P_2(x)$ quantify the respective likelihoods of the competing models P_1 and P_2, based on the observed data x.

This model selection paradigm satisfies fundamental optimality properties, the most important one being that for a sequence of independent observations, this selection principle is asymptotically optimal as the number of observations tends to infinity. For instance, suppose that in our example we had postulated that the probability of Heads was 6/10 under P_2. Repeating the calculation above, we would have obtained

$$P_2(x) = \left(\frac{4}{10}\right)^2 \left(\frac{6}{10}\right)^4 = \frac{20,736}{1,000,000}$$

and we would have selected P_2 because the odds in favor of P_1 are now lower than 1:

$$\frac{P_1(x)}{P_2(x)} = \frac{1,000,000}{64 \times 20,736} \approx 0.75.$$

We made the wrong choice because we were unlucky to observe a sequence x that happened to nearly match the model P_2. However the probability of such events vanishes asymptotically.

A simple illustration of the problem of limited observations is the following. Let us postulate the following trivial model for P_1: probability 1 is assigned to the *Lena* image, and probability 0 to any other image. The model P_2 can be anything else. If we apply the likelihood principle to the *Lena* image, we conclude that P_1 was correct. The problem with the trivial model P_1 is that it lacks generalization, or universality.

Returning to asymptotic considerations, we can make correct decisions with high probability if our test set of images is large enough. Let x^i, $1 \leq i \leq K$, be K images drawn independently from an image class $\mathcal{X}$. The likelihoods of these images under models P_1 and P_2 are respectively

$$\prod_{i=1}^{N} P_1(x^i) \quad \text{and} \quad \prod_{i=1}^{N} P_2(x^i).$$

Hence we choose model P_1 if

$$\sum_{i=1}^{N} \ln P_1(x^i) > \sum_{i=1}^{N} \ln P_2(x^i)$$

and P_2 otherwise.

Model Mismatch. Given a large set of test images, the likelihood ratio test will provide the correct answer (with high probability) if these images were generated from either P_1 and P_2. In reality, P_1 and P_2 are simplified models, and the underlying distribution P that generated the images may be neither P_1 nor P_2. In this case, the likelihood ratio test picks the model that is closest to P in the Kullback-Leibler sense. For large sets of test images, it may be shown that the likelihood ratio test picks P_1 with high probability if

$$D(P||P_1) < D(P||P_2)$$

where

$$D(P||Q) = \sum_x P(x) \ln \frac{P(x)}{Q(x)}$$

is the Kullback-Leibler divergence between distributions P and Q. The Kullback-Leibler divergence is not symmetric in its arguments and is therefore not a distance in the mathematical sense. It is however nonnegative and is equal to zero only if $P = Q$.

3.2 Maximum-Likelihood Parameter Estimation

The likelihood principle may be applied to the problem of parameter estimation. Consider for instance the problem of estimating the probability of Heads in our coin flip example; this probability is viewed as an unknown parameter $\theta \in [0, 1]$.

Formally, we consider a family of distributions $\mathcal{P} = \{P_\theta(x),\, 0 \leq \theta \leq 1\}$ parameterized by θ (not just two distributions as considered before) and use the likelihood principle to estimate θ. Given x, the quantity $P_\theta(x)$ is viewed as a function of θ, called the likelihood function, and to be maximized over θ:

$$L(\theta) \triangleq P_\theta(x) = \theta^{N_H}(1-\theta)^{N_T}.$$

It is slightly more convenient to work with the log likelihood function

$$\ln L(\theta) = \ln P_\theta(x) = N_H \ln\theta + N_T \ln(1-\theta).$$

Maximizing $L(\theta)$ is equivalent to maximizing $\ln L(\theta)$, because the log function is monotonic increasing.

The log likelihood function above is concave. If $N_H = 0$, the maximizing θ is equal to 0; if $N_T = 0$, the maximizing θ is equal to 1. If both N_H and N_T are positive, the likelihood function tends to $-\infty$ as $\theta \to 0$ and $\theta \to 1$. Hence the maximizer is an interior point of the interval $[0, 1]$ and is obtained by setting the derivative of the likelihood function to 0:

$$0 = \left.\frac{d \ln L(\theta)}{d\theta}\right|_{\theta=\hat{\theta}_{ML}}.$$

This is the so-called likelihood equation. Its solution is

$$\begin{aligned} 0 &= \frac{N_H}{\theta} - \frac{N_T}{1-\theta} \\ \hat{\theta}_{ML} &= \frac{N_H}{N} \end{aligned}$$

and admits an intuitive meaning: the estimator is simply the proportion of observed Heads in the sequence.

3.3 Composite Test

The likelihood principle can also be used to solve more complex model validation problems. Imagine we have two competing models, each of which is a parametric family of distributions:

$$\mathcal{P}_1 = \{P_\theta,\, \theta \in \Theta_1\}, \quad \mathcal{P}_2 = \{P_\theta,\, \theta \in \Theta_2\}.$$

For the coin flip example, we could have $\Theta_1 = [0.4, 0.6]$ and $\Theta_2 = [0.1, 0.4) \cap (0.6, 0.9]$.

Let

$$\begin{aligned}\hat{\theta}_1 &= \underset{\theta\in\Theta_1}{\operatorname{argmax}}\, P_\theta(x)\\ \hat{\theta}_2 &= \underset{\theta\in\Theta_2}{\operatorname{argmax}}\, P_\theta(x)\end{aligned}$$

be the ML estimates of θ assuming models 1 and 2, respectively. Then we decide in favor of model 1 if

$$P_{\hat{\theta}_1}(x) > P_{\hat{\theta}_2}(x).$$

This is known as the Generalized Likelihood Ratio Test (GLRT) approach to model validation.

In the context of images, this approach can be used to evaluate some fairly rich models.

3.4 Empirical Cost Minimization

In contrast with the previous approach, which aims at constructing a probability model for images and validate it independently of any application, we now consider an application-driven evaluation method. We assume that a mathematical abstraction for the application is available as described in Chapter I. The ingredients of this framework are

- a probability model $P(x)$ (considered unknown here) for images;
- a decision function ψ mapping data y to a decision $\hat{x}$;
- a cost function $C(\psi|x)$ quantifying the cost of using ψ when the underlying image is x.

The average cost under probability model P is

$$C(\psi, P) = \sum_x P(x)C(\psi|x).$$

For instance, in an image restoration problem, $\hat{x}$ is the restored image, and $C(\psi|x)$ is the mean-squared reconstruction error for a particular image x (the average is over the possible realizations of the noise in the image degradation model).

The function ψ is not allowed to depend on x, so if P is known, the best one can do is to choose ψ that minimizes the expected cost $C(\cdot, P)$. The resulting ψ is known as the Bayes decision rule.

If P is not known but a test set $x^i, \cdots, x^K$ is available, the average cost for this test set is

$$\frac{1}{K}\sum_{i=1}^{K} C(\psi|x^i) = C(\psi, \hat{P})$$

where

$$\hat{P}(x) = \sum_{i=1}^{K} \delta(x - x^i)$$

is the empirical probability measure for the images in the test set.

To compare two competing models P_1 and P_2, let

$$\psi_1 = \operatorname*{argmin}_{\psi} C(\psi, P_1) \quad \text{and} \quad \psi_2 = \operatorname*{argmin}_{\psi} C(\psi, P_2)$$

and choose P_1 if ψ_1 performs better than ψ_2 over the test set:

$$C(\psi_1, \hat{P}) < C(\psi_2, \hat{P})$$

This method also possesses an asymptotic optimality property. Since $C(\psi, \hat{P}) \to C(\psi, P)$ in probability as $K \to \infty$, the model validation criterion is asymptotically equivalent to the Bayes model validation criterion (based on the true P)

$$C(\psi_1, P) < C(\psi_2, P)$$

as $K \to \infty$.

4 Autoregressive Models

Autoregressive (AR) models provide a flexible and tractable way to model the spectral characteristics of WSS signals – both in 1D and in higher dimensions. Note that AR models are intended to model neither the mean of a signal, nor its higher-order statistics. The high correlation of neighboring pixel values in images may be captured using fairly simple AR models.

AR models are based on the notion of prediction filters, which are commonplace in 1-D signal processing. We review some background material and terminology on predictive filters, then extend it to 2D. For video, predictive filtering may also be used in the temporal direction.

4.1 1-D AR Models

Consider a length-p, linear time-invariant filter with coefficients $\mathbf{a} = \{a_1, a_2, \cdots, a_p\}$ and delay at least equal to 1. Its z-transform is

$$A(z) = \sum_{k=1}^{p} a_k z^{-k}.$$

If the filter is excited by white noise $e(n)$, the filter output

$$x(n) = \underbrace{\sum_{k=1}^{p} a_k x(n-k)}_{\hat{x}(n)} + e(n) \tag{1}$$

is said to follow an AR(p) model, where p is the model order.

The right side of (1) is the sum of two terms. The first one,

$$\hat{x}(n) \triangleq \sum_{k=1}^{p} a_k x(n-k)$$

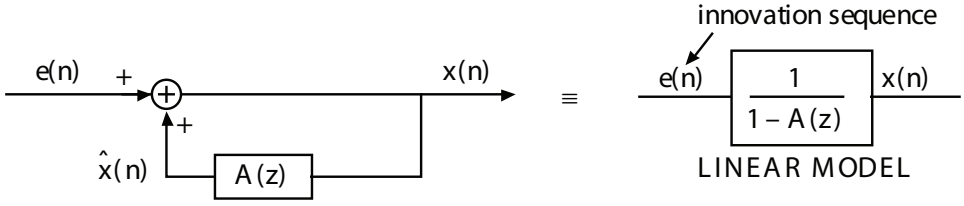
is a prediction based on past values of $x(n)$. The second term, $e(n)$, is viewed as an innovation, or prediction error. Taking the z-transform of both sides of (1), we obtain the transfer function from $e(n)$ to $x(n)$:

$$X(z) = \frac{1}{1 - A(z)} E(z). \tag{2}$$

To ensure stability of this system, we always assume that the poles of the transfer function (i.e., the zeroes of $1 - A(z)$) are inside the unit circle. Since

$$E(z) = [1 - A(z)]X(z),$$

is white noise, the filter $1 - A(z)$ is simply called *whitening filter*.



Let us study the second-order statistics of this system. The second-order statistics of $e(n)$ are

$$\begin{aligned} \mathbb{E}[E(n)] &= 0 \\ \mathbb{E}[E(n)E(m)] &= \sigma^2 \delta_{nm} \end{aligned}$$

where δ_{nm} denotes the Kronecker delta, taking value 1 if $n = m$ and zero otherwise. Since $A(z)$ is causal, $x(n)$ depends only and current and past values of $e(n)$, and we have

$$\mathbb{E}[X(n)E(m)] = 0, \quad \forall m > n.$$

Multiply both sides of (1) by $x(0)$ and take expectations.

$$R_X(n) = \mathbb{E}[X(n)X(0)] = \sum_{k=1}^{p} a_k R_X(n-k) + \sigma^2 \delta_{n0}, \quad \forall n. \tag{3}$$

For $1 \leq n \leq p$, these equations take the following matrix form:

$$\underbrace{\begin{bmatrix} R_X(0) & R_X(1) & R_X(2) & \cdots & R_X(p-1) \\ R_X(1) & R_X(0) & R_X(1) & \cdots & R_X(p-2) \\ R_X(2) & R_X(1) & R_X(0) & \cdots & R_X(p-3) \\ \vdots & \vdots & \vdots & & \vdots \\ R_X(p-1) & R_X(p-2) & R_x(p-3) & \cdots & R_X(0) \end{bmatrix}}_{\mathsf{R}} \underbrace{\begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_p \end{bmatrix}}_{\mathbf{a}} = \underbrace{\begin{bmatrix} R_X(1) \\ R_X(2) \\ R_X(3) \\ \vdots \\ R_X(p) \end{bmatrix}}_{\mathbf{r}} \tag{4}$$

For $n = 0$, we obtain

$$R_X(0) = \mathbf{a} \cdot \mathbf{r} + \sigma^2 \tag{5}$$

Equations (4) and (5) are the so-called *Yule-Walker equations*. They form a linear system with $p + 1$ equations and $p + 1$ unknowns $\mathbf{a}$ and σ^2. Given the correlation values $R_X(0), \cdots, R_X(p)$, this system may be solved to determine the prediction filter and variance of innovations. The matrix R in (4) is Toeplitz symmetric. An $O(p^2)$ numerical solution to that problem is obtained using the Levinson-Durbin algorithm.

For $n > p$, the covariance lags may all be computed by application of the recursion (3).

Since $e(n)$ is white noise with variance σ^2, from (2) we obtain the spectral density of X as

$$S_X(f) = \frac{\sigma^2}{|1 - A(e^{j2\pi f})|^2}. \tag{6}$$

Different designs of the prediction filter lead to different spectral characteristics of X. In particular, this design is very good at capturing spectral peaks; in this case, the whitening filter has zeroes in the vicinity of the corresponding frequencies on the unit circle.

Example 1: AR(1) process. Let $p = 1$ and $a_1 = \rho$; thus $A(z) = \rho z^{-k}$. The whitening filter $1 - A(z)$ has a zero at location $z = \rho$. To ensure stability, this zero must be inside the unit circle, i.e., $|\rho| < 1$. Applying (3) with $n = 0, 1$, we obtain

$$\begin{aligned} R_X(0) &= \rho R_X(1) + \sigma^2 \\ R_X(1) &= \rho R_X(0) \end{aligned}$$

hence $R_X(0) = \frac{\sigma^2}{1-\rho^2}$. Applying (3) again, we obtain

$$R_X(n) = R_X(0)\rho^{|n|} = \frac{\sigma^2}{1 - \rho^2}\rho^{|n|} \quad \forall n.$$

From (6), we have

$$S_X(f) = \frac{\sigma^2}{|1 - \rho e^{-j2\pi f}|^2}.$$

Example 2: AR(2) process. Let $p = 2$. The zeroes of the whitening filter $1 - A(z) = 1 - a_1 z^{-1} - a_2 z^{-2}$ are inside the unit circle if $|a_2| < 1$ and $|a_1| < 1 + a_2$.

4.2 Parameter Estimation

Given a realization $x = \{x(0), x(1), \cdots, x(N-1)\}$ of an AR(p) process with unknown parameters $\mathbf{a}$ and σ^2, we want to estimate these parameters. The Yule-Walker procedure for doing so is as follows.

First, compute the so-called *biased sample covariance*

$$\hat{R}_X(k) \triangleq \frac{1}{N} \sum_{n=0}^{N-1-k} x(n)x(n+k), \quad 0 \leq k \leq p. \tag{7}$$

Since there are only $N-k$ terms in the sum, and $\mathbb{E}[X(n)X(n+k)] = R_X(k)$, the expected value of the estimate (7) is given by

$$\mathbb{E}[\hat{R}_X(k)] = \left(1 - \frac{k}{N}\right) R_X(k) \quad \neq R_X(k).$$

While the estimator is biased, it is *asymptotically unbiased* as $N \to \infty$. Moreover, $\hat{R}_X(k)$ converges in probability to $R_X(k)$ for $0 \leq k \leq p$.

Second, use the estimate from (7) in the Yule-Walker equations (4) and (5), obtaining estimates $\hat{\mathbf{a}}$ and $\hat{\sigma}^2$ for the AR(p) model parameters. The resulting whitening filter $1 - \hat{A}(z)$ is guaranteed to have its zeroes inside the unit circle. (There is no such guaranteed if one uses the *unbiased sample covariance*

$$\tilde{R}_X(k) \triangleq \frac{1}{N} \sum_{n=0}^{N-1-k} x(n)x(n+k)$$

as an estimate for $R_X(k)$.)

4.3 2-D AR Models

Predictive models can be constructed in 2D as well, giving rise to 2-D AR models. The model still takes the form of (1), where $n = (n_1, n_2)$ and $k = (k_1, k_2)$ are now defined over the 2-D lattice $\mathbb{Z}^2$. Equivalently, we may write

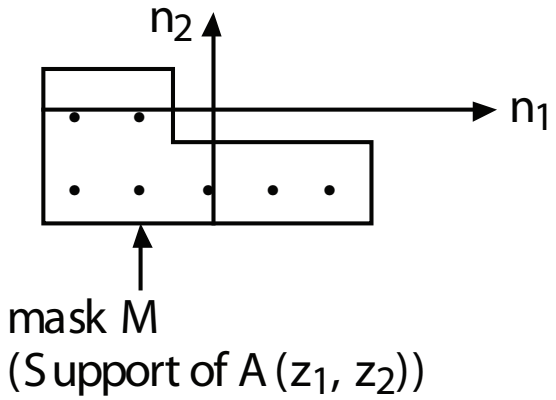
$$x(n) = \sum_{k \in \mathcal{M}} a_k \, x(n-k) + e(n)$$

or

$$x(n_1, n_2) = \sum_{(k_1,k_2) \in \mathcal{M}} a_{k_1 k_2} \, x(n_1 - k_1, n_2 - k_2) + e(n_1, n_2)$$

where $\mathcal{M}$ is the support of the prediction filter.

The only restriction on the filter is that it be causal and not include the origin. Causality in 2D is defined relative to the scanning scheme, which effectively reorders the 2-D sequence



into a 1-D sequence. For the usual raster scanning scheme, $\mathcal{M}$ may include any (k_1, k_2) such that

$$k_2 < 0 \quad \text{or} \quad k_1 < 0,\ k_2 = 0;$$

other values of (k_1, k_2) are not allowed.

As in 1D, the whitening filter should be stable. Stability may be studied using the 2-D z-transform,

$$A(z_1, z_2) \triangleq \sum_{k_1,k_2} a_{k_1 k_2} z_1^{-k_1} z_2^{-k_2}$$

where z_1 and z_2 are complex numbers. To guarantee stability, the zeroes of the whitening filter $1 - A(z_1, z_2)$ must be inside the *unit bicircle*

$$\{(z_1, z_2) \ : \ |z_1| = |z_2| = 1\}.$$

The innovation sequence $e(n)$ is spatially white noise with zero mean and variance σ^2. The covariance of $X(n)$ is derived as in the 1-D problem:

$$R_X(n) = \mathbb{E}[X(n)X(0)] = \sum_{k \in \mathcal{M}} a_k R_X(n-k) + \sigma^2 \delta_{n0}, \quad \forall n \tag{8}$$

Evaluating (8) at all n in $\mathcal{M} \cup \{0\}$, we obtain a linear system of $|\mathcal{M}| + 1$ equations $|\mathcal{M}| + 1$ unknowns $\mathbf{a}$ and σ^2, analogous to the Yule-Walker system (4) (5) in 1D.

The spectral density for X is given by

$$S_X(f_1, f_2) = \frac{\sigma^2}{|1 - A(e^{j2\pi f_1}, e^{j2\pi f_2})|^2}.$$

Nonzero Mean $x(n) = x_c(n) + \mu_X(n)$

where the *centered signal* $x_c(n)$ has zero mean and follows the previously described model.

Example 3: separable AR(1) process. The prediction filter has three taps: $\mathcal{M} = \{(1,0),(1,1),(0,1)\}$ where

$$a_{k_1,k_2} = \rho[\delta(k_1,0) + \delta(0,k_2)] - \rho^2\delta(k_1,k_2).$$

Here δ denotes the 2-D unit impulse, and $\rho \in [-1,1]$ is the correlation coefficient between horizontal or vertical neighbors; for photographic images, ρ is usually in the range 0.90 — 0.98. The whitening filter is separable:

$$\begin{aligned} 1 - A(z_1,z_2) &= 1 - \rho z_1^{-1} - \rho z_2^{-1} + \rho^2 z_1^{-1} z_2^{-1} \\ &= (1 - \rho z_1^{-1})(1 - \rho z_2^{-1}) \end{aligned}$$

The spectral density of X is therefore also separable:

$$S_X(f_1,f_2) = \frac{\sigma^2}{|1 - \rho e^{j2\pi f_1}|^2 |1 - \rho e^{j2\pi f_2}|^2}$$

and so is the correlation sequence

$$R_X(n_1,n_2) = \frac{\sigma^2}{(1-\rho^2)^2}\,\rho^{|n_1|+|n_2|}.$$

Example 4: nonseparable, isotropic AR(1) process. Here

$$R_X(n_1,n_2) = c\,\rho^{(n_1^2+n_2^2)^{1/2}}.$$

Both models are simple and classical in statistical image processing. However they fail to capture inhomogeneities in images. The next section introduces a more general family of models which contains AR processes as a special case and explicitly model non-Gaussian statistics.

5 Markov Random Fields

The notion of Markov random field (MRF) is a significant extension of the notion of Markov process or Markov chain in 1D. As in 1D, the statistical dependency of a sample on the rest of the sequence appears only via a localized *neighborhood* of that sample. For an order-p (1D) Markov process, this neighborhood is comprised of the past p samples. For images, the notion of causality is unnatural. Thus neighborhoods for Markov random fields are generally noncausal, which results in a rich class of models but also in significant complications because information does not propagate in a predetermined direction. As a result, inference algorithms for MRFs are iterative in nature (require a stopping criterion), unlike inference algorithms for Markov chains which are recursive but yield exact solutions.

We first review basic definitions in 1D before defining MRFs and examining their properties and suitability as image models. Throughout this chapter, we use the same notation $p(\cdot)$ to denote a pmf or a pdf.

5.1 1-D Markov Processes

Definition: $X = \{X_n, n \in \mathbb{Z}\}$ is a 1-D Markov process if

$$p(x_n|x_{n-1}, x_{n-2}, \cdots, x_0) = p(x_n|x_{n-1}) \quad \forall n,$$

i.e., $x(n)$ depends on the past only via $x(n-1)$.

Let x_0^n denote the $(n+1)$-tuple $\{x_0, \cdots, x_n\}$. The joint pmf for a 1-D Markov process satisfies the property

$$\begin{aligned} p(x_0^n) &= p(x_n|x_0^{n-1})p(x_0^{n-1}) \\ &= p(x_n|x_{n-1})p(x_0^{n-1}) \end{aligned}$$

where the second equality uses the Markov property. Similarly we obtain

$$p(x_0^{n-1}) = p(x_{n-1}|x_{n-2})p(x_0^{n-2}).$$

Recursive application of this decomposition yields

$$p(x_0^n) = \prod_{i=1}^{n} p(x_i|x_{i-1})p(x_0).$$

Terminology: the samples X_i are often termed *states*, and $p(x_i|x_{i-1})$ are the *transition probabilities*. If X_i takes values in a finite set with L elements, then X is said to be a *Markov chain*, and $p(x_i|x_{i-1})$ is the $L \times L$ *transition matrix* for the Markov chain.

Note that any AR(1) process is Markov but not all Markov processes are AR(1). A simple example is the nonlinear process

$$x_i = x_{i-1}^2 + e_i$$

where e_i is white noise.

The notion of a Markov process may be generalized to *order-p Markov processes*. These processes satisfy the condition

$$p(x_n|x_{n-1}, x_{n-2}, \cdots, x_0) = p(x_n|x_{n-1}, \cdots, x_{n-p}) \quad \forall n,$$

i.e., x_n depends on the past only via $x_{n-1}, \cdots, x_{n-p}$.

5.2 Markov Random Fields

In Sec. 4.3, we extended the notion of causal prediction filters from 1D to 2D in a straightforward manner, by defining a scanning order and a causal filter, and applying this filter to the scanned sequence. However causality is unnatural in 2D, and constraining the image model to satisfy causal relationships is artificial. For instance, some image processes are isotropic, but

causal AR processes cannot be isotropic (except for the trivial white noise process, for which no prediction is needed).

Fortunately it is possible to design image models free of causality constraints while retaining the concept of local statistical dependencies. This is done by modeling images as Markov random fields. MRF theory was initially developed by Dobrushin [3], also see Besag [4] and the book by Kinderman and Snell [5]. The use of MRFs in image processing became popular following Geman and Geman's landmark paper [6]. The MRF framework can be used to develop image models that have long-range correlations. The image processing algorithms (restoration, classification, etc.) associated with MRF's are iterative.

We first introduce the mathematical element of this framework. Let $\mathcal{I}$ be a finite subset of the lattice $\mathbb{Z}^2$. The elements of $\mathcal{I}$ are called *sites* (or pixel locations).

Definition. A neighborhood system $\mathscr{N} = \{\mathcal{N}_i,\ i \in \mathcal{I}\}$ is a collection of subsets of $\mathcal{I}$, called neighborhoods, that satisfy two conditions:

1. $\forall i \in \mathcal{I}: \quad i \notin \mathcal{N}_i$;
2. $\forall i, j \in \mathcal{I}: \quad i \in \mathcal{N}_j \quad \Rightarrow \quad j \in \mathcal{N}_i$.

Definition. Sites i and j are neighbors if $i \in \mathcal{N}_j$ or equivalently, if $j \in \mathcal{N}_i$.

Example 1: denote by i^E, i^N, i^W, i^S the eastern, northern, western and southern neighbors of i on the lattice. The *first-order neighborhood* of i is defined as

$$\mathcal{N}_i = \{i^E, i^N, i^W, i^S\} \tag{9}$$

and is represented below. Sites i and i^E are neighbors, but i^E and i^N are not. The resulting $\mathscr{N}$ is called the nearest-neighbor system.

$$\begin{array}{ccc} & i^N & \\ i^W & i & i^E \\ & i^S & \end{array}$$

Example 2: The *second-order neighborhood* of i is made of 8 sites:

$$\mathcal{N}_i = \{i^E, i^{NE}, i^N, i^{NW}, i^W, i^{SW}, i^S, i^{SE}\} \tag{10}$$

as depicted below.

$$\begin{array}{ccc} i^{NW} & i^N & i^{NE} \\ i^W & i & i^E \\ i^{SW} & i^S & i^{SE} \end{array}$$

One may similarly define 3rd-order neighborhoods, etc.

Assume a finite state space $\Lambda = \{0, \cdots, L-1\}$, which could represent the possible gray levels of an image. Any realization $x \in \Lambda^{\mathcal{I}}$ of the state process is called a *configuration*. Given a subset $\mathcal{C}$ of $\mathcal{I}$, we denote by

$$x_{\mathcal{C}} = \{x_i,\ i \in \mathcal{C}\}$$

the *restriction* of x to $\mathcal{C}$.

Definition: $X = \{X_i,\ i \in \mathcal{I}\}$ is a MRF with respect to $\mathscr{N}$ if the following two conditions are satisfied. First, X_i depends on the other samples only via the neighborhood $\mathcal{N}_i$:

$$p(x_i|x_{\mathcal{I}\setminus\{i\}}) = p(x_i|x_{\mathcal{N}_i}), \quad i \in \mathcal{I}. \tag{11}$$

Second, we have the positivity condition

$$p(x) > 0, \quad \forall x \in \Lambda^{\mathcal{I}}.$$

Definition. The $|\mathcal{I}|$ conditional distributions $p(x_i|x_{\mathcal{N}_i})$ in the right side of (11) are called the *local characteristics* of the MRF.

Remarks.

1. Neighborhoods may be small (e.g., the nearest-neighbor system) yet have long-range interactions by virtue of the selection of local characteristics (similarly to Markov processes in 1D).

2. Any $p(x)$ is a MRF. To see this, it suffices to define $\mathcal{N}_i \triangleq \mathcal{I} \setminus \{i\}$ for all $i \in \mathcal{I}$. The prime advantage of MRFs resides in the possibility of generating interesting models from relatively small neighborhoods.

3. A MRF is stationary if its neighborhoods and local characteristics are invariant to spatial shifts. (This requires either an infinite domain $\mathcal{I}$, e.g., the entire lattice $\mathbb{Z}^2$, or a periodic domain as discussed in the next section.)

4. If the neighborhoods are causal (relative to a particular scanning order of the domain $\mathcal{I}$), the MRF is said to be a *Markov mesh* [7], or a *unilateral MRF* [8].

5. In general (unilateral MRFs being an exception), $p(x)$ cannot be expressed simply in terms of the local characteristics $p(x_i|x_{\mathcal{N}_i})$ because the chain rule that was used in 1D is not applicable. Therefore, computing $p(x)$ is a significant challenge.

6. Even worse, given a set of local characteristics $p(x_i|x_{\mathcal{N}_i})$, there may not exist a $p(x)$ that is consistent with these local characteristics.

5.3 Boundary conditions

Denote by $\partial\mathcal{I}$ the boundary of the finite domain $\mathcal{I}$. Since the neighborhoods $\mathcal{N}_i$ may not include sites outside $\mathcal{I}$, the treatement of boundaries requires special attention. Consider as an example the nearest-neighbor system with rectangular domain $\mathcal{I}$. If site i is on the left boundary of $\mathcal{I}$, then i^E is outside $\mathcal{I}$. Three methods have been used to address this problem.

- Fixed boundaries: x_i determined for all $i \in \partial\mathcal{I}$, and the neighborhoods are centered at interior points of $\mathcal{I}$.

- Free boundaries: the shape of $\mathcal{N}_i$ is modified in the vicinity of boundaries. (For instance, i^E would be omitted from $\mathcal{N}_i$ if i is a site on the left boundary of $\mathcal{I}$.)
- Periodic boundaries: the domain $\mathcal{I}$ wraps around (is toroidal).

5.4 Gibbs Distributions

Markov random fields have an important connection to a class of distributions introduced by Gibbs a hundred years ago. These distributions play a fundamental role in statistical physics.

Definition: $\mathcal{C}$ is a clique with respect to a neighborhood system $\mathcal{N}$ if $\mathcal{C}$ is a singleton, or if every pair of distinct sites in $\mathcal{C}$ are neighbors.

Example: The cliques defined over the first-order neighborhood system of (9) have cardinality equal to 1 or 2:

$$\mathcal{C} = \{i\} \quad \text{(singleton)} \tag{12}$$
$$\mathcal{C} = \{i, i^E\} \text{ or } \{i, i^N\} \quad \text{(horizontal or vertical pairs).} \tag{13}$$

Definition: A Gibbs distribution relative to $\mathcal{N}$ is a probability measure P with the following representation:

$$p(x) = \frac{1}{Z} \exp\left\{-\frac{1}{T} U(x)\right\} \tag{14}$$

where T is the *temperature* which controls the peakiness of the pmf $p(x)$,

$$Z = \sum_{x \in \Lambda^{\mathcal{I}}} \exp\left\{-\frac{1}{T} U(x)\right\} \tag{15}$$

is a normalization constant, also called *partition function*, and

$$U(x) = \sum_{\mathcal{C} \in \mathscr{C}} V_{\mathcal{C}}(x_{\mathcal{C}}) \tag{16}$$

is the *energy function*. In (16), $\mathscr{C}$ is a set of cliques, and $V_{\mathcal{C}}(x_{\mathcal{C}})$ is the *potential function*, which depends on x only via $x_{\mathcal{C}}$, the restriction of x to $\mathcal{C}$.

From this definition, we see that the likely configurations are those with low energy, and the lower the temperature, the more likely such configurations. Let us examine some examples.

Example 1. Markov chain: let $\mathcal{I} = \{0, 1, \cdots, n\}$,

$$\mathcal{N}_i = \{i-1, i+1\} \quad \text{for} \quad 1 \le i < n, \qquad \mathcal{N}_0 = \{1\}, \qquad \mathcal{N}_n = \{n-1\}.$$

For this neighborhood system consider the following system $\mathscr{C}$ of n cliques:

$$\mathcal{C}_i = \{i-1, i\} \quad \text{for} \quad 1 \le i \le n, \qquad \mathcal{C}_0 = \{0\}.$$

Define the potential functions

$$V_{\mathcal{C}}(x_{\mathcal{C}}) = \begin{cases} -\ln p(x_i|x_{i-1}) & : \mathcal{C} = \{i-1, i\}, \quad 1 \le i \le n, \\ -\ln p(x_0) & : \mathcal{C} = \{0\}. \end{cases}$$

Then

$$\begin{aligned} p(x) &= p(x_0) \prod_{i=1}^{n} p(x_i|x_{i-1}) \\ &= \exp\left\{ \ln p(x_0) + \sum_{i=1}^{n} \ln p(x_i|x_{i-1}) \right\} \\ &= \exp\left\{ -\sum_{\mathcal{C} \in \mathscr{C}} V_{\mathcal{C}}(x_{\mathcal{C}}) \right\}. \end{aligned}$$

Example 2: Ising model, with $\Lambda = \{\pm 1\}$.

This model was introduced by Ernest Ising (1925) to study ferromagnetism; here x_i represents the state of the magnetic spin at location i. The Ising model is a binary isotropic random field with $\Lambda = \{\pm 1\}$ and the first-order cliques of (12) (13):

$$U(x) = \sum_i \alpha x_i + \beta(x_i x_{i^N} + x_i x_{i^E}). \tag{17}$$

The parameter α represents an external force, and the parameter β is called *bonding strength*. The cases $\beta < 0$ and $\beta > 0$ are known as the *attractive case* and the *repulsive case*, respectively. For $\alpha = 0$ and $\beta < 0$, the Ising model penalizes the length of boundaries between regions labeled -1 and 1. A similar interpretation is found in the example below.

Example 3: Potts model, with $\Lambda = \{0, 1\}$.

Consider the first-order cliques of (12) (13) and the energy function

$$\begin{aligned} U(x) &= \sum_i \beta \left[1(x_i \neq x_{i^N}) + 1(x_i \neq x_{i^E}) \right] &(18) \\ &= \beta l &(19) \end{aligned}$$

where $\beta > 0$, and l is the length of the boundary between regions labeled 0 and 1. In (18), the function $1(\mathcal{E})$ takes value 1 if the event $\mathcal{E}$ is true, and 0 otherwise. See the following configuration, obtained when $\mathcal{I} = [-n, n]^2$ and $n = 2$:

$$x = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & |\overline{\mathbf{1}}| & 0 & 0 & 0 \\ 0 & |\underline{\mathbf{1}} & \overline{\mathbf{1}}| & 0 & 0 \\ 0 & 0 & |\underline{\mathbf{1}}| & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Here the length of the boundary is $l = 10$.

The inverse of the bonding parameter β plays the role of the temperature: $T = \beta^{-1}$. There exists a *critical temperature* $T_c = 0.88^{-1}$ below which infinite-dimensional clusters appear. This phenomenon is unique to 2D and does not exist in 1D. For temperatures higher than T_c, we have

$$Pr[X_0 = 0|X_{\partial\mathcal{I}}] \sim Pr[X_0 = 1|X_{\partial\mathcal{I}}] \sim \frac{1}{2} \quad \text{as } n \to \infty.$$

that is, the value of the field at the center of the domain is asymptotically independent of the values on the boundary. However this property does not hold at temperatures lower than T_c.

Example 4: Multilevel Potts model, with $\Lambda = \{0, 1, \cdots, L-1\}$. This model is a special M-level MRF, with the same first-order neighborhood system as the bilevel Ising model, and the same energy function (18). The model penalizes changes in intensity levels, and all changes are penalized equally. Given a first-order neighborhood, the most likely intensity value for the center site is the value assumed by the majority of the neighbors (this value is generally not unique). This model is fairly good for modeling edges.

$$x = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & |\overline{\mathbf{1}}| & \overline{\mathbf{2}} & \overline{\mathbf{2}}| & 0 \\ 0 & |\underline{\mathbf{1}} & \overline{\mathbf{1}}| & \underline{\mathbf{2}}| & 0 \\ 0 & 0 & |\underline{\mathbf{1}}| & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Example 5 We may extend the Potts model to the second-order neighborhood system of (10). The energy function for an isotropic binary field takes the form

$$U(x) = \sum_i \beta(1)\,[1(x_i \neq x_{i^N}) + 1(x_i \neq x_{i^E})] + \sum_i \beta(2)\,[1(x_i \neq x_{i^{NE}}) + 1(x_i \neq x_{i^{NW}})].$$

One can similarly define the third-order neighborhood of i by enlarging the second-order neighborhood with the 4 sites that are at a distance 2 from i. This concept is extended straightforwardly to neighborhoods of order 4 and beyond. The figure below displays the weight assignments for an isotropic fourth-order model.

$$\begin{bmatrix} & \beta(4) & \beta(3) & \beta(4) & \\ \beta(4) & \beta(2) & \beta(1) & \beta(2) & \beta(4) \\ \beta(3) & \beta(1) & \bullet & \beta(1) & \beta(3) \\ \beta(4) & \beta(2) & \beta(1) & \beta(2) & \beta(4) \\ & \beta(4) & \beta(3) & \beta(4) & \end{bmatrix}$$

Example 6 For first-order neighborhoods, anisotropic models are obtained by introducing asymmetries between horizontal and vertical directions. Consider for instance

$$U(x) = \sum_i [\beta_h\, 1(x_i \neq x_{i^N}) + \beta_v\, 1(x_i \neq x_{i^E})]$$

which tends to produce more vertical edges than horizontal edges if $\beta_v < \beta_h$. Likewise, for second-order neighborhoods, anisotropic models may also be obtained by introducing asymmetries between diagonal directions.

Remarks.

1. The most general form of the energy function for the nearest-neighbor system defined on $\Lambda = \{\pm 1\}$ is given by
$$U(x) = \sum_i V_i(x_i) + V_{i,i^N}(x_i x_{i^N}) + V_{i,i^E}(x_i x_{i^E}).$$
2. A nice feature of Gibbs random fields is that the unnormalized pmf $Zp(x) = \exp\{-U(x)\}$ is easy to compute.
3. Unfortunately, computation of partition function Z in (15) is prohibitively expensive because it involves a summation over all possible configurations.

5.5 Hammersley-Clifford Theorem

The Hammersley-Clifford theorem establishes a fundamental connection between MRF and Gibbs distributions. The original proof of the theorem is fairly involved, but a simpler proof was derived by Besag [4].

Theorem: $X = \{X_i,\ i \in \mathcal{I}\}$ is a MRF with respect to $\mathscr{N}$ if and only if $p(x)$ is a Gibbs distribution with respect to $\mathscr{N}$.

An important consequence of this theorem is that one may specify any MRF via associated potential functions $\{V_{\mathcal{C}}(x_{\mathcal{C}}), \mathcal{C} \in \mathscr{C}\}$.

There exist explicit formulas for computing the energy $U(x)$ from the local characteristics of the MRF. Conversely, one can compute local characteristics from $U(x)$, as follows. For any configuration $x \in \Lambda^{\mathcal{I}}$, we have

$$p(x_i|x_{\mathcal{I}\setminus\{i\}}) = \frac{\frac{1}{Z}\exp\{-\sum_{\mathcal{C}\in\mathscr{C}} V_{\mathcal{C}}(x_{\mathcal{C}})\}}{\sum_{x_i\in\Lambda}\frac{1}{Z}\exp\{-\sum_{\mathcal{C}\in\mathscr{C}} V_{\mathcal{C}}(x_{\mathcal{C}})\}}.$$

All terms cancel out except those involving the cliques to which site i belong. Denoting by $\mathscr{C}_i$ this set of cliques, we obtain

$$\begin{aligned} p(x_i|x_{\mathcal{I}\setminus\{i\}}) &= \frac{\exp\{-\sum_{\mathcal{C}\in\mathscr{C}_i} V_{\mathcal{C}}(x_{\mathcal{C}})\}}{\sum_{x_i\in\Lambda}\exp\{-\sum_{\mathcal{C}\in\mathscr{C}_i} V_{\mathcal{C}}(x_{\mathcal{C}})\}} \\ &= p(x_i|x_{\mathcal{N}_i}). \end{aligned} \tag{20}$$

For the Ising model, the elements of $\mathscr{C}_i$ are the cliques $\{i\}$, $\{i, i^E\}$, $\{i, i^N\}$, $\{i, i^W\}$, and $\{i, i^S\}$, and thus

$$\sum_{\mathcal{C}\in\mathscr{C}_i} V_{\mathcal{C}}(x_{\mathcal{C}}) = \alpha x_i + \beta x_i \underbrace{(x_{i^E} + x_{i^N} + x_{i^W} + x_{i^S})}_{v_i}. \tag{21}$$

When $\Lambda = \{0, 1\}$, (20) and (21) yield the so-called autologistic model for the local characteristics of the MRF,

$$p(x_i|x_{\mathcal{N}_i}) = \frac{\exp\{-x_i(\alpha + \beta v_i)\}}{1 + \exp\{-\alpha - \beta v_i\}}$$

where $v_i \triangleq x_{iE} + x_{iN} + x_{iW} + x_{iS}$.

5.6 Image Texture Models

MRFs have been used to approximate the statistics of some image textures. The goal of such an approximation may be

- to generate natural-looking textures for image understanding or computer graphics, or
- to use the statistical model to design a segmentation, classification, restoration, or compression algorithm.

Regarding the first point, Tamura *et al.* [9] define six attributes of texture: coarseness, contrast, directionality, line-likeness, regularity, and roughness.

Cross and Jain [10] have applied the various models of Sec. 5.4 to texture modeling. Low-order neighborhood systems can be used to generate textures with various properties:

- isotropic or anisotropic (with patterns along preferred horizontal, vertical or diagonal directions).
- clustering properties: for the first-order neighborhood system, typical cluster size depends on the bonding parameter β. If $\beta < 0$, the formation of clusters is discouraged. Similar effects can be created with higher-order neighborhood systems; the formation of certain types of structure (e.g., diagonal streaks) can be inhibited.

In general, these neighborhood systems are not suitable for modeling regular textures such as brickwalls and other patterned objects: the corresponding MRF models are based on very local pixel interactions and are nongeometric.

Textures can be generated from the selected model using the *Gibbs sampler* of Sec. 5.8. Conversely, for a given texture, model parameters may be estimated using the methods of Sec. 5.7.

5.7 Parameter Estimation

As an example, assume that an observed configuration x was generated by an Ising model with parameters α and β, and consider the problem of estimating those parameters. Referring to (17), we use the short hand

$$U(x) = \alpha \sum_i x_i - \beta \sum_{i \sim j} x_i x_j$$

where $\sum_{i\sim j}$ denotes the sum over all nearest neighbors (i,j). We first examine the obvious choice for parameter estimation: the maximum-likelihood method. The method turns out to be unpractical but can be replaced with a good approximation, using the local characteristics of the MRF.

Maximum-Likelihood Method. Given x, view $p(x)$ as a function of α and β. In principle we may then maximize the log likelihood function

$$\begin{aligned} l(\alpha,\beta) &= \ln p(x) \\ &= -\ln Z - \alpha \sum_i x_i - \beta \sum_{i\sim j} x_i x_j \end{aligned}$$

over α and β. Unfortunately, computation of the partition function

$$Z = \sum_x \exp\left\{-\alpha \sum_i x_i - \beta \sum_{i\sim j} x_i x_j\right\}$$

involves a summation over all possible configurations $x \in \Lambda^{\mathcal{I}}$. Thus, if $\mathcal{I}$ is large, Z is an uncomputable function of α and β, and the ML estimator cannot be implemented. The following simpler method is often used to estimate MRF parameters.

Pseudo-Likelihood Method (also known as coding method) [4]. A coding is a set of sites which are conditionally independent given their own neighborhood. Consider for instance the nearest-neighbor system. Subsampling $\mathcal{I}$ according to a quincunx scheme, we obtain two codings $C_\bullet$ and $C_\circ$ whose elements are respectively bullets and circles in the figure below.

$$\begin{bmatrix} \bullet & \circ & \bullet & \circ \\ \circ & \bullet & \circ & \bullet \\ \bullet & \circ & \bullet & \circ \\ \circ & \bullet & \circ & \bullet \end{bmatrix}$$

Define the configurations $x_\bullet \triangleq \{x_i,\, i \in C_\bullet\}$ and $x_\circ \triangleq \{x_i,\, i \in C_\circ\}$ defined over these quincunx lattices. Conditioned on $x_\circ$, the joint probability of $x_\bullet$ takes a product form:

$$p(x_\bullet) = \prod_{i\in C_\bullet} p(x_i|x_{\mathcal{N}_i})$$

where the local characteristics are computed explicitly according to (21). Defining the local log likelihood function

$$l(\alpha,\beta|x_i,x_{\mathcal{N}_i}) = \ln p(x_i|x_{\mathcal{N}_i})$$

we obtain a computationally feasible optimization problem:

$$(\hat{\alpha}_\bullet,\hat{\beta}_\bullet) = \operatorname*{argmax}_{\alpha,\beta} \sum_{i\in C_\bullet} l(\alpha,\beta|x_i,x_{\mathcal{N}_i}).$$

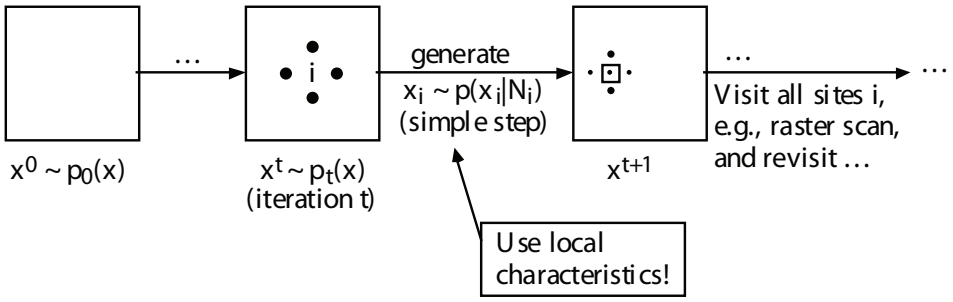
Furthermore, switching the roles of $x_\bullet$ and $x_\circ$, one may similarly obtain a second set of estimates $(\hat{\alpha}_\circ, \hat{\beta}_\circ)$. (Note that the estimates $(\hat{\alpha}_\bullet, \hat{\beta}_\bullet)$ and $(\hat{\alpha}_\circ, \hat{\beta}_\circ)$ are highly dependent because the configurations $x_\bullet$ and $x_\circ$ are highly dependent.) Averaging the two parameter estimates yield the pseudo-ML estimates of the unknown parameters. This procedure is popular because it is simple to implement and has nice theoretical properties, including convergence in probability to the true parameter values as $|\mathcal{I}| \to \infty$.

For a MRF with second-order neighborhhood, a similar procedure would be used, with *four* codings $C_\bullet$, $C_\circ$, $C_\star$ and $C_\triangle$ defined below. The samples of $x_\bullet$ defined over $C_\bullet$ are conditionally independent given the other three codings, etc.

$$\begin{bmatrix} \bullet & \triangle & \bullet & \triangle \\ \star & \circ & \star & \circ \\ \bullet & \triangle & \bullet & \triangle \\ \star & \circ & \star & \circ \end{bmatrix}$$

5.8 Gibbs Sampler

The problem of generating random samples with a prescribed distribution is of course fundamental in many applications of engineering and statistics. If X is an homogeneous, irreducible, aperiodic Markov chain, the distribution of X_n converges to the stationary distribution as $n \to \infty$, for any initial distribution of X_0. For MRFs, the problem is more challenging. Geman and Geman [6] studied a possible method, which they called *Gibbs sampler*. Their method falls in the general category of Monte Carlo Markov Chain methods, which have become very popular in the last ten years [8]; also see the Sept. 1998 issue of the IEEE Signal Processing Magazine and the Feb. 2002 issue of the IEE Transactions on Signal Processing.



The Gibbs sampler repeatedly draws samples from the local characteristics of the MRF and construct a sequence of configurations x^t whose distribution converges to the desired distribution as $t \to \infty$. Define a scanning sequence $i(t)$ for $t = 1, 2, \cdots$ and use the following iterative algorithm:

1. Generate an initial x^0 from some arbitrary distribution $p_0(x)$ on the configuration space (e.g., an i.i.d. process).

2. For $t = 1, 2, \cdots$, perform the following operations:

   (a) visit site $i = i(t)$ and generate x_i according to the local characteristic $p(x_i|x_{\mathcal{N}_i})$;

   (b) define the updated field x^t as the one that has value x_i at site i and agrees with x^{t-1} at all other sites.

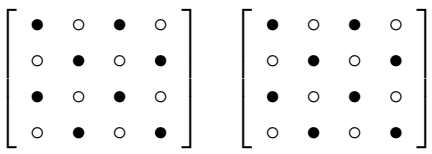
This procedure induces a pmf $p_t(x)$ at time t on the configuration space. The scanning sequence could be periodic with period $|\mathcal{I}|$ (e.g., repeated applications of the raster scan) but does not have to. Assuming each site i is visited an infinite number of times, it may be shown that p_t converges in distribution to the desired p, independently of the initial distribution p_0. The reason is that $\{x^t, 0 \leq t < \infty\}$ is a (field-valued) Markov chain and that the sufficient conditions for convergence of p_t are satisfied.

A related question is ergodicity: given a function $F(X)$, can statistical averages of the form $\mathbb{E}[F(X)]$ be approximated by temporal averages? Specifically, do we have the property

$$\frac{1}{T}\sum_{t=1}^{T} F(X^t) \to \mathbb{E}[F(X)] \quad (a.s.) \quad \text{as } T \to \infty.$$

The answer is positive provided that the sites are visited "frequently enough." Technically, there should exist some τ such that all sites are visited during any time interval $[t, t+\tau]$, for any value of t.

Practical implementation of the Gibbs sampler. The theory above provides some mild conditions on the scanning sequence which guarantee that p_t eventually converges to the desired distribution. In practice, we are interested in designing a scanning sequence that makes this convergence fast. Motivated by the "coding schemes" of Sec. 5.7, we note that work can be performed simultaneously on conditionally independent sites such as the bullets in the lattice of Sec. 5.7 (conditioned on $x_{\mathcal{N}_i}$ where the sites in $\mathcal{N}_i$ are denoted by circles). For this example, one would visit the sites in two passes, one for each coding. The procedure would then be repeated iteratively. The ergodicity condition above is satisfied, with $\tau = |\mathcal{I}| - 1$.



First pass: Update • Second pass: Update ∘

5.9 Real-Valued MRFs

We turn our attention to the case $\Lambda = \mathbb{R}$, i.e., intensities are real-valued. (This is often used as an approximation to the case where Λ is a finite but large set.) The framework studied above may be used to define real–valued Gibbs and Markov random fields.

Consider for instance the following Gaussian MRF, defined over the nearest-neighbor system:

$$p(x) = \frac{1}{Z} \exp\left\{ -\alpha \sum_i x_i^2 - \frac{1}{2\sigma^2} \sum_{i \sim j} (x_i - x_j)^2 \right\}$$

where again, $\sum_{i \sim j}$ denotes a sum over nearest neighbors. Observe that the inverse covariance matrix $\mathsf{B} = \mathsf{R}^{-1}$ (also called *information matrix*) for this process is sparse because $\mathsf{B}(i,j) = 0$ unless $i = j$ or $i \sim j$. Also note that the partition function Z is now given by an integral over $\mathbb{R}^{\mathcal{I}}$ instead of a sum in the case of discrete MRFs.

A Gaussian MRF model strongly penalizes images with large changes in adjacent pixel intensity values and therefore favors homogeneity. The main drawback of this model is that it penalizes edges too much.

The general form of a Gaussian MRF with first-order neighborhood system is

$$p(x) = \frac{1}{Z} \exp\left\{ -\sum_i a_i x_i^2 - \sum_{i \sim j} b_{ij} (x_i - x_j)^2 \right\}.$$

A useful altenative to Gaussian MRFs, which penalizes edges less heavily, is given by the following family of pdf's:

$$p(x) = \frac{1}{Z} \exp\left\{ -\alpha \sum_i x_i^2 - \sum_{i \sim j} \rho\left(\frac{x_i - x_j}{\sigma} \right) \right\} \tag{22}$$

where ρ is the potential function, and σ is a scale parameter that controls gray scale variation. The Gaussian MRF model is a special case of (22) with potential function $\rho(u) = \frac{1}{2} u^2$.

A very popular choice for the potential function, proposed by Bouman and Sauer [11], is

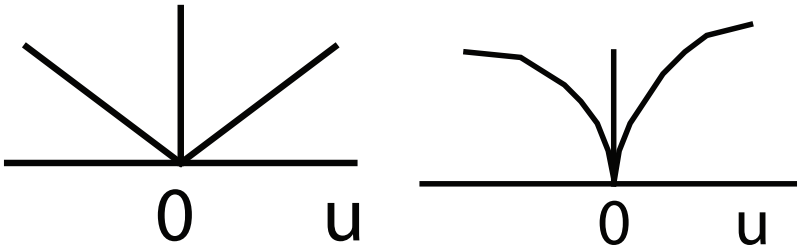
$$\rho(u) = \frac{1}{p} |u|^p, \quad p > 0. \tag{23}$$

The smaller the exponent p is, the less edges will be penalized. The Gaussian MRF model is a special case of (23) with $p = 2$. The MRF models associated with (23) are known as *generalized Gaussian MRFs*.

For $p > 1$, the function $\ln p(x)$ is strictly concave in x (the pdf is said to be strictly log-concave), which is a very attractive property for optimization, as we shall see later. The case $p = 1$ is known as Laplacian MRF. The limiting case $p \to 0$ admits an interesting interpretation. If Λ was discrete with L levels, in the limit as $p \to 0$ we would obtain $\rho(u) \sim \frac{1}{p} 1(u \neq 0)$, i.e., the multilevel Potts model which does not penalize the magnitude of jumps.

The derivative of the potential function,

$$\psi(u) = \frac{d\rho(u)}{du}, \tag{24}$$



(a) $\rho(u) = |u|$ (b) $\rho(u) = \frac{1}{p}|u|^p$, $p < 1$

is called *influence function*. It quantifies how much a pixel value is attracted to the values of its neighbors. The influence function for GGMRFs is antisymmetric and takes the form

$$\psi(u) = |u|^{p-1} \operatorname{sgn}(u)$$

where $\operatorname{sgn}(u)$ is the signum function. For $p > 1$, $\psi(u)$ increases monotonically from $-\infty$ to ∞. For $p < 1$, we have $\lim_{|u|\to\infty} \psi(u) = 0$.

Estimation of scale parameter. As mentioned earlier, computation of the partition function Z is untractable, so at first sight it is difficult to derive the ML estimate of the scale parameter σ in (22). However this is not the case. Viewing Z as a function of σ, note that we have the scaling property $Z(\sigma) = \sigma^{|\mathcal{I}|} Z(1)$. Therefore the likelihood equation

$$\begin{aligned} 0 &= \frac{d \ln p(x)}{d\sigma} \\ &= \frac{d}{d\sigma}\left[-\ln Z(1) - |\mathcal{I}| \ln \sigma - \sum_{i\sim j} \rho\left(\frac{x_i - x_j}{\sigma}\right)\right] \\ &= -\frac{|\mathcal{I}|}{\sigma} + \frac{1}{\sigma^2} \sum_{i\sim j} (x_i - x_j)\, \psi\left(\frac{x_i - x_j}{\sigma}\right) \end{aligned}$$

yields the ML estimate $\hat{\sigma}$ as the solution to the nonlinear equation

$$\sum_{i\sim j} \frac{x_i - x_j}{\sigma}\ \psi\left(\frac{x_i - x_j}{\sigma}\right) = |\mathcal{I}|,$$

which may be solved numerically. For the GGMRF model, noting that $u\psi(u) = |u|^p$, we obtain the closed-form solution

$$\hat{\sigma} = \left(\frac{1}{|\mathcal{I}|} \sum_{i\sim j} |x_i - x_j|^p\right)^{1/p}.$$

5.10 Line Processes

Among other important contributions, Geman and Geman [6] introduced a *line process* to model image edges. The line process l_{ij} is a binary field taking value 1 if an edge is present between sites i and j, and 0 otherwise. Sites i and j are either horizontal or vertical neighbors. Consider the following example:

$$x = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & |\overline{1}| & \overline{7} & \overline{8}| & 0 \\ 0 & |\underline{2} & \overline{3}| & \underline{9}| & 0 \\ 0 & 0 & |\underline{2}| & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

where the total boundary length is 15. The horizontal and vertical segments in this diagram indicate locations where the line process takes value 1.

The joint distribution of the configuration x and the label field l may be modeled as a Gibbs random field:

$$p(x, l) = \frac{1}{Z} \exp\{-U(x, l)\}$$

where $U(x, l)$ is the joint energy function. A popular choice, using the nearest-neighbor system, is

$$U(x, l) = \alpha \sum_i x_i^2 + \sum_{i \sim j} \left[\frac{1}{2\sigma^2} (1 - l_{ij}) (x_i - x_j)^2 + \gamma \, l_{ij} \right]. \tag{25}$$

Observe that $p(x, l)$ is a Gaussian MRF if $\alpha = 0$. If $\alpha > 0$ and $l_{ij} = 1$, the model (25) suspends the penalty $\frac{1}{2\sigma^2}(x_i - x_j)^2$ and replaces it with the edge penalty γ.

The line process model admits an interesting interpretation in light of (22). The minimum of the energy functional $U(x, l)$ in (25)over l is given by

$$\min_l U(x, l) = \alpha \sum_i x_i^2 + \sum_{i \sim j} \rho \left(\frac{x_i - x_j}{\sigma} \right)$$

where

$$\rho(u) = \min \left(\frac{1}{2} u^2, \gamma \right),$$

i.e., a clipped version of the quadratic $\rho(u)$ associated with Gaussian MRFs.

6 Multiresolution Image Representations

The concept of scale, or resolution of an image, is very intuitive. A person observing a scene perceives the objects in that scene at a certain level of resolution that depends on the distance to these objects. For instance, walking towards a distant building, she would first perceive a rough outline of the building. The main entrance becomes visible only in relative proximity to the buiding. Finally, the door bell is visible only in the entrance area. As this example illustrates, the notions of resolution and scale loosely correspond to the size of the details that can be perceived by the observer. It is of course possible to formalize these intuitive concepts, and indeed signal processing theory gives them a more precise meaning.

These concepts are particularly useful in image and video processing and in computer vision. A variety of digital image processing algorithms decompose the image being analyzed into several components, each of which captures information present at a given scale. While there exists several types of multiscale image decompositions, we consider three main methods [12]—[17]:

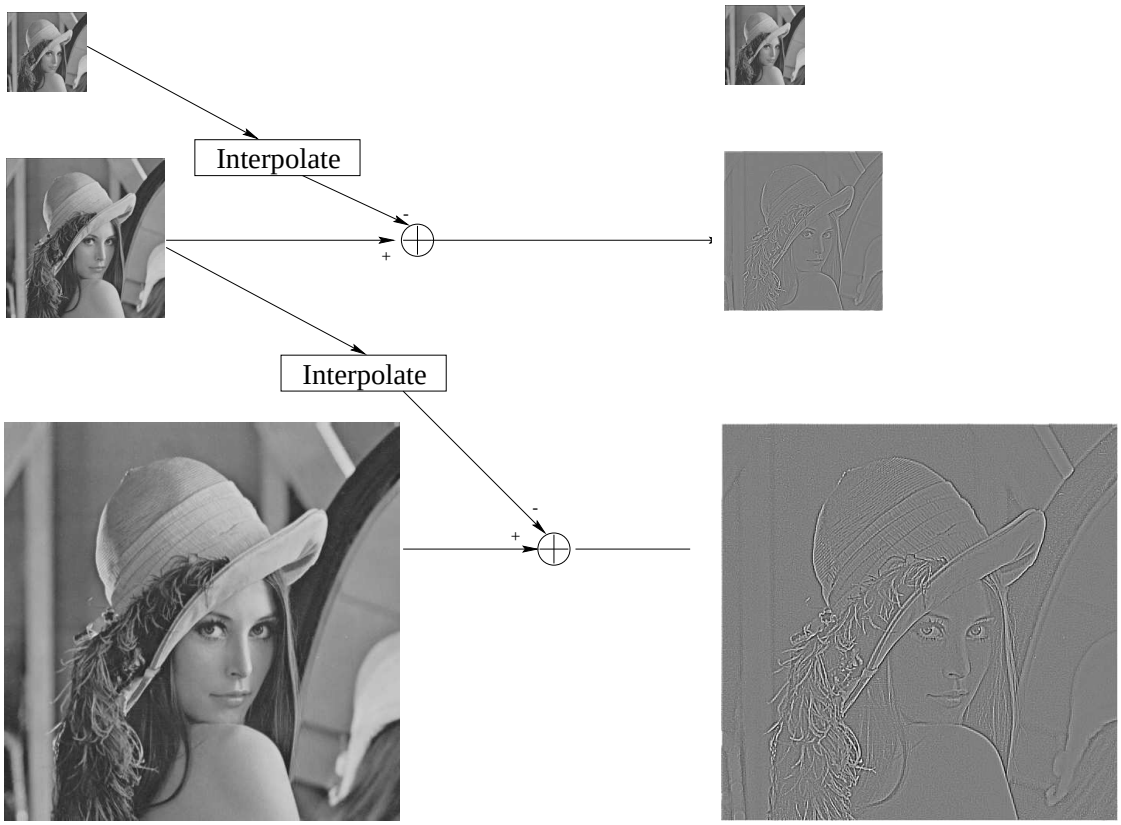
1. In a **Gaussian pyramid** representation of an $n_1 \times n_2$ image (Fig. 2a), the original image appears at the bottom of a pyramidal stack of images. This image is then lowpass filtered and subsampled by a factor of two in each coordinate. The resulting $n_1/2 \times n_2/2$ image appears at the second level of the pyramid. This procedure can be iterated several times. Here resolution can be measured by the size of the image at any given level of the pyramid. The pyramid in Fig. 2a has three resolution levels, or scales. In the original application of this method to computer vision, the lowpass filter used was often a Gaussian filter [1], hence the terminology *Gaussian pyramid*. This terminology is commonly adopted even when the lowpass filter is not a Gaussian filter. Another terminology is simply *lowpass pyramid*. Note that the total number of pixels in a pyramid representation is $n_1 n_2 + n_1 n_2/4 + n_1 n_2/16 + \cdots \approx \frac{4}{3} n_1 n_2$. This is said to be an *overcomplete* representation of the original image, due to the increase in the number of pixels.

2. The **Laplacian pyramid** representation of the image is closely related to the Gaussian pyramid, but here the difference between approximations at two successive scales is computed and displayed for different scales, see Fig. 2b. The displayed images represent details of the image that are significant at each scale. An equivalent way to obtain the image at a given scale is to apply the difference between two Gaussian filters to the original image. This is analogous to filtering the image using a Laplacian filter, a technique commonly employed for edge detection. Laplacian filters are bandpass, hence the name Laplacian pyramid, also termed *bandpass pyramid*.

3. In a **wavelet decomposition**, the image is decomposed into a set of subimages (or subbands) which also represent details at different scales (Fig. 3). Unlike pyramid representations, the subimages also represent details with different spatial orientations (such

[1] This design was motivated by analogies to the Human Visual System.

as edges with horizontal, vertical, and diagonal orientations). The number of pixels in a wavelet decomposition is only $n_1 n_2$.



(a) Gaussian pyramid

(b) Laplacian pyramid

Figure 2: Two pyramid representations of *Lena*: (a) Gaussian pyramid; (b) Laplacian pyramid.

Hierarchical image representations such as those in Fig. 2 and 3 are useful in many applications. In particular, they lend themselves to effective designs of reduced–complexity algorithms for texture analysis and segmentation, edge detection, image analysis, motion analysis, and image understanding in computer vision. Moreover, the Laplacian pyramid and wavelet image representations are *sparse* in the sense that most detail images contain few significant pixels (little significant detail). This sparsity property is very useful in image compression, as bits are allocated only to the few significant pixels; in image recognition, because the search for significant image features is facilitated; and in the restoration of images corrupted by noise, as images and noise possess rather distinct properties in the wavelet domain.

The Gaussian and Laplacian pyramids and wavelet decompositions are presented in more detail below. First we review some background on decimation and interpolation.

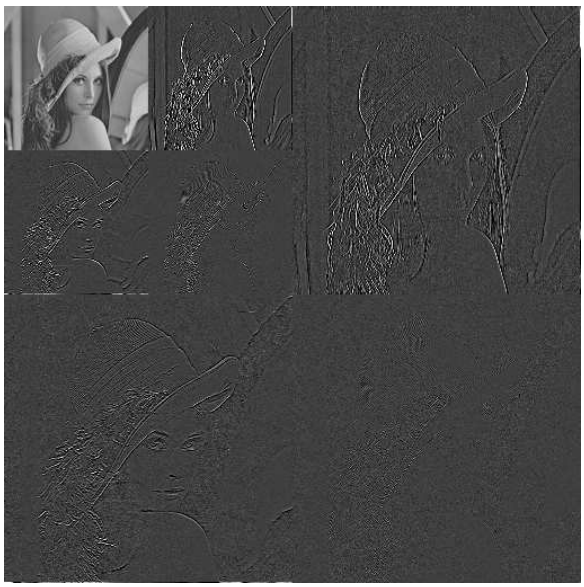


Figure 3: Wavelet decomposition of *Lena*.

6.1 Decimation and Interpolation

Consider the problem of decimating a 1–D signal by a factor of two, namely, reducing the sample rate by a factor of two. The basic operations involved in decimation are lowpass filtering (using a digital anti–aliasing filter) and subsampling, as shown in Fig. 4. The impulse response of the lowpass filter is denoted by $h(n)$, and its Discrete–Time Fourier Transform by $H(e^{j\omega})$. The relationship between input $x(n)$ and output $y(n)$ of the filter is the convolution equation

$$y(n) = x(n) * h(n) = \sum_k h(k)x(n-k).$$

The downsampler discards every other sample of its input $y(n)$. Its output is given by

$$z(n) = y(2n).$$

Combining these two operations, we obtain

$$z(n) = \sum_k h(k)x(2n-k). \tag{26}$$

Downsampling usually implies a loss of information, as the original signal $x(n)$ cannot be exactly reconstructed from its decimated version $z(n)$. The traditional solution for reducing this information loss consists in using an "ideal" digital anti–aliasing filter $h(n)$ with cutoff frequency $\omega_c = \pi/2$. However such "ideal" filters have infinite length. In image processing, short Finite Impulse Response (FIR) filters are preferred for obvious computational reasons.

Furthermore, approximations to the "ideal" filters above have an oscillating impulse response, which unfortunately results in visually annoying ringing artifacts in the vicinity of edges. The FIR filters typically used in image processing are symmetric, with length between three and twenty taps. Two common examples are the 3–tap FIR filter $h(n) = (\frac{1}{4}, \frac{1}{2}, \frac{1}{4})$, and the length–$(2L+1)$ truncated Gaussian, $h(n) = Ce^{-n^2/(2\sigma^2)}, |n| \leq L$, where $C = 1/\sum_{|n|\leq L} e^{-n^2/(2\sigma^2)}$. The coefficients of both filters add up to one: $\sum_n h(n) = 1$, which implies that the DC response of these filters is unity.

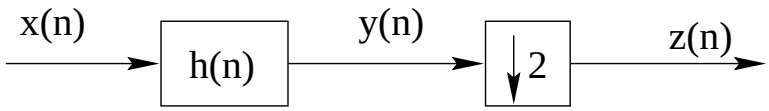


Figure 4: Decimation of a signal by a factor of two, obtained by cascade of a lowpass filter $h(n)$ and a subsampler $\downarrow 2$.

The second operation of interest here is *interpolation*, which increases the sample rate of a signal. Signal processing theory tells us that interpolation may be performed by cascading two basic signal processing operations: upsampling and lowpass filtering, see Fig. 5. The upsampler inserts a zero between every other sample of the signal $x(n)$:

$$y(n) = \begin{cases} x(n/2) & : n \text{ even} \\ 0 & : n \text{ odd} \end{cases}$$

The upsampled signal is then filtered using a lowpass filter $h(n)$. The interpolated signal is given by $z(n) = h(n) * y(n)$ or, in terms of the original signal $x(n)$,

$$z(n) = \sum_k h(k)x(n-2k). \tag{27}$$

The so-called ideal interpolation filters have infinite length. Again, in practice, short FIR filters are used.

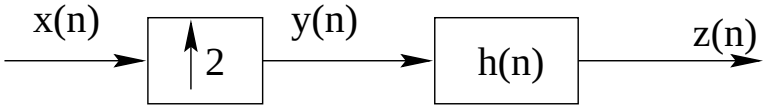


Figure 5: Interpolation of a signal by a factor of two, obtained by cascade of an upsampler $\uparrow 2$ and a lowpass filter $h(n)$.

6.2 Gaussian Pyramid

The construction of a Gaussian pyramid involves 2–D lowpass filtering and subsampling operations. The 2–D filters used in image processing practice are *separable*, which means that they

can be implemented as the cascade of 1–D filters operating along image rows and columns. This is a convenient choice in many respects, and the 2–D decimation scheme is then separable as well. Specifically, 2–D decimation is implemented by applying 1–D decimation to each row of the image followed by 1–D decimation to each column of the resulting image. The same result would be obtained by first processing columns and then rows. Likewise, 2–D interpolation is obtained by first applying 1-D interpolation to each row of the image, and then again to each column of the resulting image, or vice-versa.

This technique was used at each stage of the Gaussian pyramid decomposition in Fig. 2(a). The lowpass filter used for both horizontal and vertical filtering was the 3–tap filter $h(n) = (\frac{1}{4}, \frac{1}{2}, \frac{1}{4})$.

6.3 Laplacian Pyramid

We define a *detail image* as the difference between an image and its approximation at the next coarser scale. The Gaussian pyramid generates images at multiple scales, but these images have different sizes. In order to compute the difference between a $n_1 \times n_2$ image and its approximation at resolution $n_1/2 \times n_2/2$, one should interpolate the smaller image to the $n_1 \times n_2$ resolution level before performing the subtraction. This operation was used to generate the Laplacian pyramid in Fig. 2(b). The interpolation filter used was the 3–tap filter $h(n) = (\frac{1}{2}, 1, \frac{1}{2})$.

As illustrated in Fig. 2(b), the Laplacian representation is *sparse* in the sense that most pixel values are zero or near zero. The significant pixels in the detail images correspond to edges and textured areas such as *Lena*'s hair. Just like the Gaussian pyramid representation, the Laplacian representation is also *overcomplete*, as the number of pixels is greater (by a factor of approximately 33%) than in the original image representation.

Laplacian pyramid representations have found numerous applications in image processing, and in particular in texture analysis and segmentation [12]. Indeed, different textures often present very different spectral characteristics which can be analyzed at appropriate levels of the Laplacian pyramid. For instance, a nearly uniform region such as the surface of a lake contributes mostly to the coarse–level image, while a textured region like grass often contributes significantly to other resolution levels. Some of the earlier applications of Laplacian representations include image compression, but the emergence of wavelet compression techniques has made this approach somewhat less attractive. However, a Laplacian–type compression technique was adopted in the hierarchical mode of the lossy JPEG image compression standard.

6.4 Wavelet Representations

While the sparsity of the Laplacian representation is useful in many applications, overcompleteness is a serious disadvantage in applications such as compression. The wavelet transform offers both the advantages of a sparse image representation and a complete representation. The development of this transform and its theory has had a profound impact on a variety of

applications. In this section, we first describe the basic tools needed to construct the wavelet representation of an image. We begin with filter banks, which are elementary building blocks in the construction of wavelets.

6.4.1 Filter Banks

Fig. 6a depicts an *analysis filter bank*, with one input $x(n)$ and two outputs $x_0(n)$ and $x_1(n)$. The input signal $x(n)$ is processed through two paths. In the upper path, $x(n)$ is passed through a lowpass filter $H_0(e^{j\omega})$ and decimated by a factor of two. In the lower path, $x(n)$ is passed through a highpass filter $H_1(e^{j\omega})$ and also decimated by a factor of two. For convenience, we make the following assumptions. First, the number N of available samples of $x(n)$ is even. Second, the filters perform a circular convolution, which is equivalent to assuming that $x(n)$ is a periodic signal. Under these assumptions, the output of each path is periodic with period equal to $N/2$ samples. Hence the analysis filter bank can be thought of as a *transform* that maps the original set $\{x(n)\}$ of N samples into a new set $\{x_0(n), x_1(n)\}$ of N samples.

Fig. 6b shows a *synthesis filter bank*. Here there are two inputs $y_0(n)$ and $y_1(n)$, and one single output $y(n)$. The input signal $y_0(n)$ (resp. $y_1(n)$) is upsampled by a factor of two and filtered using a lowpass filter $G_0(e^{j\omega})$ (resp. highpass filter $G_1(e^{j\omega})$.) The output $y(n)$ is obtained by summing the two filtered signals. We assume that the input signals $y_0(n)$ and $y_1(n)$ are periodic with period $N/2$. This implies that the output $y(n)$ is periodic with period equal to N. So the synthesis filter bank can also be thought of as a transform that maps the original set of N samples $\{y_0(n), y_1(n)\}$ into a new set of N samples $\{y(n)\}$.

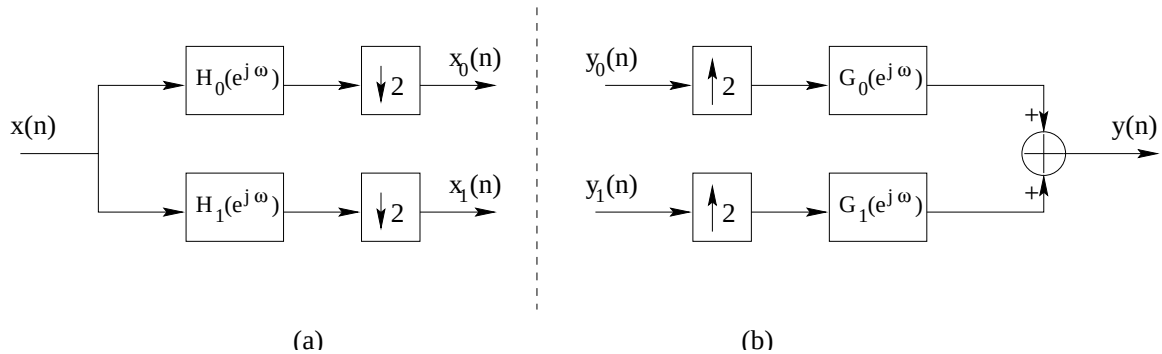


Figure 6: (a) Analysis filter bank, with lowpass filter $H_0(e^{j\omega})$ and highpass filter $H_1(e^{j\omega})$. (b) Synthesis filter bank, with lowpass filter $G_0(e^{j\omega})$ and highpass filter $G_1(e^{j\omega})$.

What happens when the output $x_0(n), x_1(n)$ of an analysis filter bank is applied to the input of a synthesis filter bank? As it turns out, under some specific conditions on the four filters $H_0(e^{j\omega}), H_1(e^{j\omega})$, $G_0(e^{j\omega})$ and $G_1(e^{j\omega})$, the output $y(n)$ of the resulting *analysis/synthesis* system is *identical* (possibly up to a constant delay) to its input $x(n)$. This condition is known as *Perfect Reconstruction*. It holds, for instance, for the following trivial set of 1–tap filters: $h_0(n)$ and $g_1(n)$ are unit impulses, and $h_1(n)$ and $g_0(n)$ are unit delays. In this case, the reader

can verify that $y(n) = x(n-1)$. In this simple example, all four filters are allpass. It is however not obvious to design more useful sets of FIR filters that also satisfy the Perfect Reconstruction condition. A general methodology for doing so was discovered in the mid eighties. We refer the reader to [16, 15] for more details.

Under some additional conditions on the filters, the transforms associated with both the analysis and the synthesis filter banks are orthonormal. Orthonormality implies that the energy of the samples is preserved under the transformation. If these conditions are met, the filters possess the following remarkable properties: the synthesis filters are a time–reversed version of the analysis filters, and the highpass filters are modulated versions of the lowpass filters, namely, $g_0(n) = (-1)^n h_1(n)$, $g_1(n) = (-1)^{n+1} h_0(n)$, and $h_1(n) = (-1)^{-n} h_0(K - n)$, where K is an integer delay. Such filters are often known as Quadrature Mirror Filters (QMF), or Conjugate Quadrature Filters (CQF), or power–complementary filters [16], because both lowpass (resp. highpass) filters have the same frequency response, and the frequency responses of the lowpass and highpass filters are related by the power-complementary property $|H_0(e^{j\omega})|^2 + |H_1(e^{j\omega})|^2 = 2$, valid at all frequencies. The filter $h_0(n)$ is viewed as a prototype filter, because it automatically determines the other three filters.

Finally, if the prototype lowpass filter $H_0(e^{j\omega})$ has a zero at frequency $\omega = \pi$, the filters are said to be *regular filters*, or *wavelet filters*. Fig. 7 shows the frequency responses of the four filters generated from a famous 4–tap filter designed by Daubechies [15, p. 195]:

$$h_0(n) = \frac{1}{4\sqrt{2}}(1 + \sqrt{3},\, 3 + \sqrt{3},\, 3 - \sqrt{3},\, 1 - \sqrt{3}).$$

This filter is the first member of a family of FIR wavelet filters that have been constructed by Daubechies and possess nice properties.

Another choice is *biorthogonal wavelet filters*, a design that sets aside degrees of freedom for choosing the synthesis lowpass filter $h_1(n)$ given the analysis lowpass filter $h_0(n)$. Such filters are subject to regularity conditions [15]. The transforms are no longer orthonormal, but the filters can have linear phase (unlike nontrivial QMF filters).

6.4.2 Wavelet Decomposition

An analysis filter bank decomposes 1–D signals into lowpass and highpass components. One can perform a similar decomposition on images by first applying 1–D filtering along rows of the image and then along columns, or vice-versa. This operation is illustrated in Fig. 8. The same filters $H_0(e^{j\omega})$ and $H_1(e^{j\omega})$ are used for horizontal and vertical filtering. The output of the analysis system is a set of four $n_1/2 \times n_2/2$ subimages: the so–called LL (low low), LH (low high), HL (high high) and HH (high high) *subbands*, which correspond to different spatial frequency bands in the image. The decomposition of *Lena* into four such subbands is shown in Fig. 9. Observe that the LL subband is a coarse (low resolution) version of the original image, and that the HL, LH and HH subbands respectively contain details with vertical, horizontal and diagonal orientations. The total number of pixels in the four subbands is equal to the original number of pixels, $n_1 n_2$.

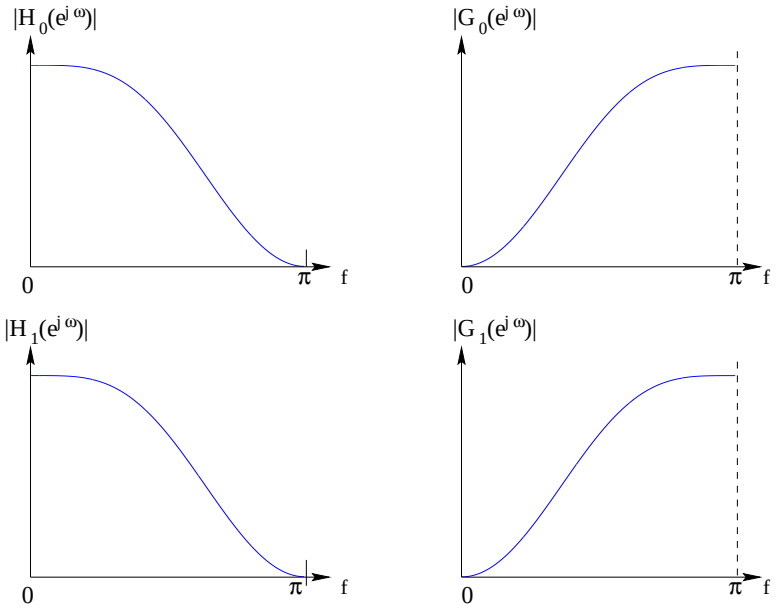


Figure 7: Magnitude frequency response of the four subband filters for a QMF filter bank generated from the prototype Daubechies' 4–tap lowpass filter.

In order to perform the wavelet decomposition of an image, one recursively applies the scheme of Fig. 8 to the LL subband. Each stage of this recursion produces a coarser version of the image as well as three new detail images at that particular scale. Fig. 10 shows the cascaded filter banks that implement this wavelet decomposition, and Fig. 3 shows a 3–stage wavelet decomposition of *Lena*. There are seven subbands, each corresponding to a different set of scales and orientations (different spatial frequency bands).

Both the Laplacian decomposition in Fig. 2(b) and the wavelet decomposition in Fig. 3 provide a coarse version of the image as well as details at different scales, but the wavelet representation is complete and provides information about image components at different spatial orientations.

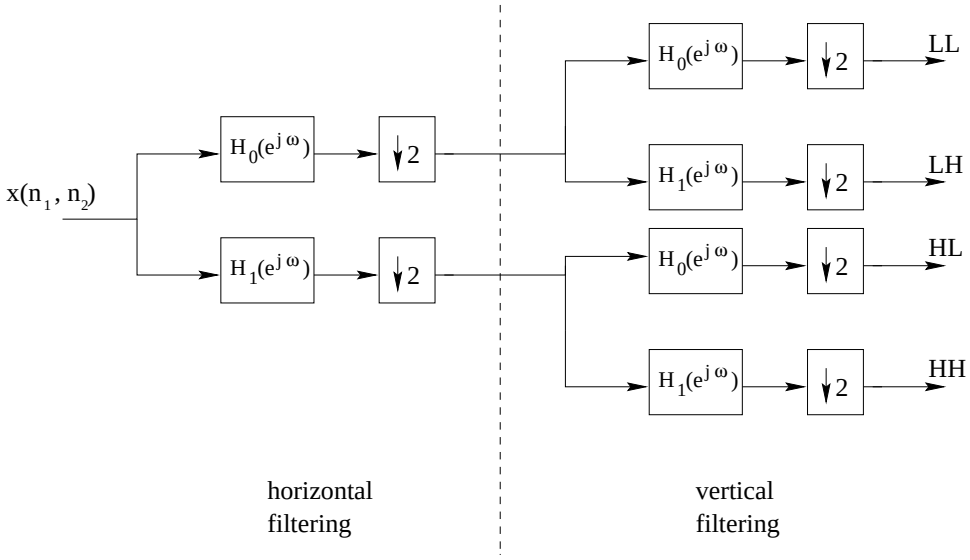


Figure 8: Decomposition of $n_1 \times n_2$ image into four $n_1/2 \times n_2/2$ subbands.

Figure 9: Four-band decomposition of *Lena*, using Daubechies' 4–tap wavelet filters.

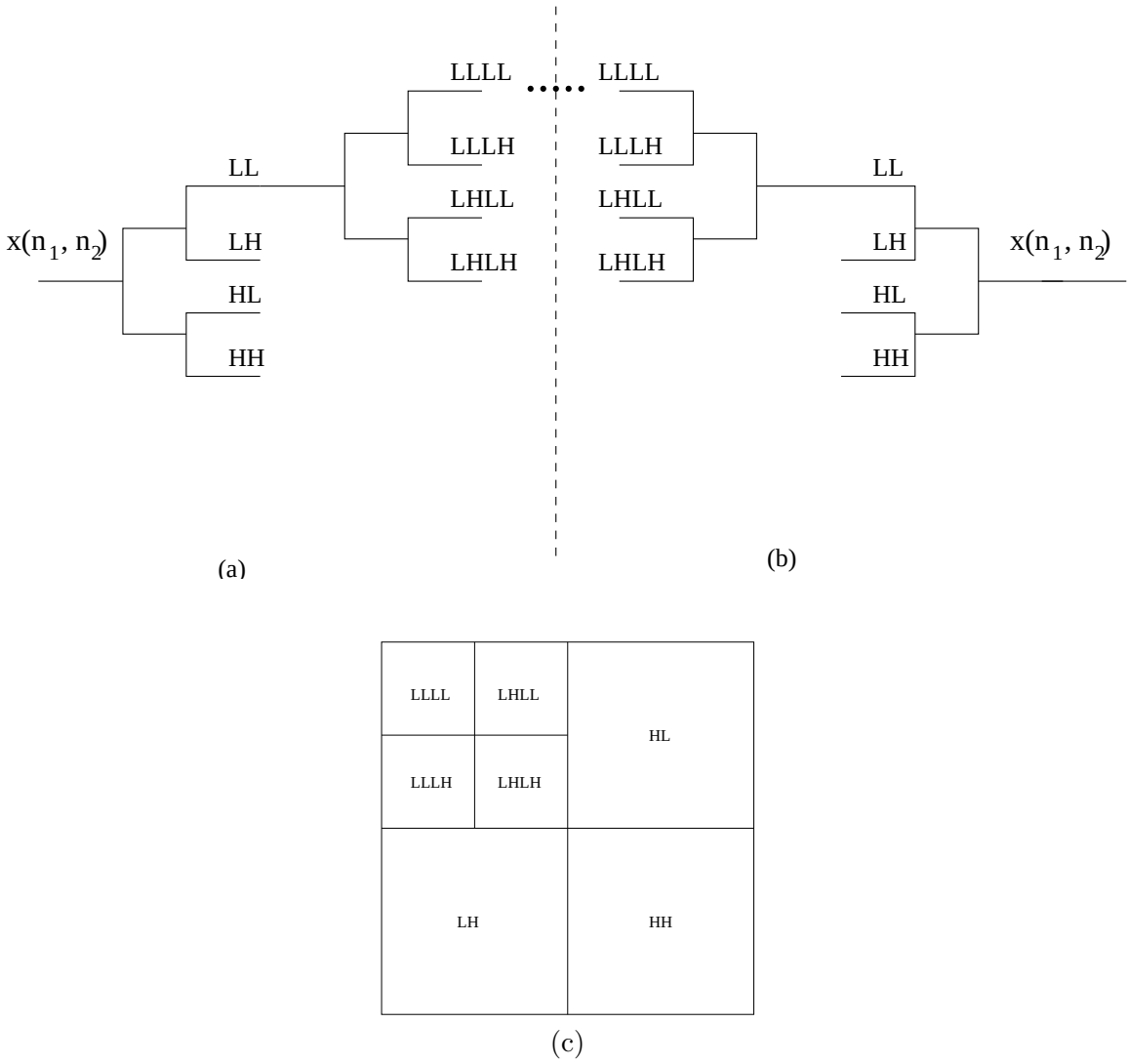


Figure 10: Implementation of wavelet image decomposition using cascaded filter banks: (a) wavelet decomposition of input image $x(n_1, n_2)$; (b) reconstruction of $x(n_1, n_2)$ from its wavelet coefficients; (c) nomenclature of subbands for a 3–level decomposition.

7 Graphical Models

The basic theory of MRFs is elegant and in principle powerful: simple models can capture nonlinear dependencies between variables and lend themselves to computationally efficient algorithms. Yet the practical difficulty remains to model complex dependencies using small neighborhoods: for instance, examples generated from the Potts and other simple models lack visual realism. It is not obvious how to construct a MRF model that would generate edges, curved boundaries, let alone buildings, cars and people.

Graphical models provide an elegant framework for constructing interesting image models that contain the above features. The application of graphical models to image processing and computer vision began in earnest during the second part of the 1990s. Graphical models have also found applications to areas as diverse as bioinformatics, error correcting codes, pattern classification, and speech recognition [18, 19]. The reader is referred to the book by Lauritzen [20] for a rigorous theory of graphical models, to the book by Frey [18] for an applications-oriented introduction to the field, and to articles by Jordan and his coworkers [19] for a study of graph-based inference algorithms.

We begin this section with some definitions and notation. Then specific graphical models are introduced, including hidden Markov models, trees and other multiresolution models. A connection to Grenander's pattern theory [21] is also presented. We introduce message-passing algorithms (belief propagation) which form the basis for design of computationally efficient inference algorithms. However a detailed study of such algorithms will be deferred to subsequent chapters in these Notes.

7.1 Definitions

The following terminology is adopted from Lauritzen [20]. A graph is a pair $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is a set of *vertices* (or *nodes*) and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is a set of ordered pairs of vertices, called *edges*. An edge $(i, j) \in \mathcal{E}$ is *directed* if $(j, i) \notin \mathcal{E}$; otherwise the edge is *undirected*. We denote directed and undirected edges by the symbols $i \to j$ and $i \sim j$, respectively. Graphs in which *all* edges are directed (resp. undirected) are termed directed (resp. undirected).

Any subset of vertices $\mathcal{U} \subset \mathcal{V}$ induces a subgraph $(\mathcal{U}, \mathcal{E}_{\mathcal{U}})$ whose edge set is obtained from $\mathcal{E}$ by keeping edges with both endpoints in $\mathcal{U}$. A set $\mathcal{U}$ of vertices is said to be *maximal* if any pair of vertices in $\mathcal{U}$ is linked by an edge.

Vertex i is a *parent* of vertex j if $i \to j$, in which case j is also called a *child* of i. We denote by $\pi(j)$ the set of parents of j. In the case of undirected edges $i \sim j$, we say that vertices i and j are *adjacent*, or *neighbors*. A *clique* of the graph is a subset of completely connected vertices.

A *path* of length n from i to j is a sequence $i = k_0, k_1, \cdots, k_n = j$ of distinct vertices such that $(k_{m-1}, k_m) \in \mathcal{E}$ for all $m = 1, \cdots, n$. We designate such a path by $i \mapsto j$. If $i \mapsto j$ and $j \not\mapsto i$, then i is an *ancestor* of j, and j is a *descendent* of i. If both $i \mapsto j$ and $j \mapsto i$, then i and j are said to *connect*. A subset $\mathcal{U}$ of $\mathcal{V}$ is a *connectivity component* of the graph if all vertices in $\mathcal{U}$ are connected.

An *n-cycle*, or *loop*, is a path of length n with the modification that $i = j$. A *tree* is a connected, undirected graph without cycles; it has a unique path between any two vertices. A *rooted tree* is the directed acyclic graph obtained from a tree by choosing as vertex as root and directing all edges away from tis root. A *leaf* is a vertex without children. A *forest* is an undirected graph where all connectivity components are trees.

7.2 Hidden Markov Models

As a first example of a graphical model, we turn to Hidden Markov Models (HMMs), which were developed by Baum and his coworkers during the 1960s [22, 23] and popularized by Rabiner and Juang in the signal processing community [24, 25]. HMMs have been particularly popular in the area of speech recognition. In 1D, a HMM is comprised of a hidden state sequence $\{S_n, n \geq 0\}$ and an observation process $\{X_n, n \geq 0\}$ which depends on the state sequence.

Specifically, $\{S_n, n \geq 0\}$ is a Markov chain with alphabet Λ and transition probability matrix

$$Q(\lambda, \lambda') = p(S_n = \lambda | S_{n-1} = \lambda), \quad \lambda, \lambda' \in \Lambda.$$

The initial state S_0 follows a pmf $p(s_0)$. The dependency of $X_n, n \geq 0$ on $S_n, n \geq 0$ appears via the conditional pmf

$$p(x_0^n | s_0^n) = \prod_{i=0}^{n} p(x_i | s_i).$$

Therefore X_i depends on the state sequence only via the current sample S_i. In summary, the HMM is fully described by the following distributions:

$$Q(\lambda, \lambda'), \quad p(s_0), \quad p(x|s).$$

While $\{X_n, n \geq 0\}$ is generally not a Markov process, it is linked to a hidden Markov process.

To illustrate computational issues in HMMs, consider the apparently simple problem of evaluating the probability of an observed sequence $\mathbf{x} = x_0^n$. If $\mathbf{x}$ were a Markov chain, the answer would be given by the chain rule:

$$p(\mathbf{x}) = p(x_0) \prod_{i=1}^{n} p(x_i | x_{i-1})$$

which is easy to evaluate. For HMMs, the state sequence $\mathbf{s} = s_0^n$ is a Markov chain but $\mathbf{x}$ is generally not. We may write

$$\begin{aligned} p(\mathbf{x}) &= \sum_{\mathbf{s} \in \mathcal{S}^n} p(\mathbf{x}|\mathbf{s}) p(\mathbf{s}) \\ &= \sum_{\mathbf{s} \in \mathcal{S}^n} \left[\prod_{i=0}^{n} p(x_i|s_i) \right] p(s_0) \prod_{i=0}^{n} p(s_i|s_{i-1}) \\ &= \sum_{\mathbf{s} \in \mathcal{S}^n} p(s_0) \prod_{i=0}^{n} p(x_i|s_i) p(s_i|s_{i-1}). \end{aligned}$$

The sum is over an exponential (in n) number of terms, so the computation appears to be infeasible for large n.

However $p(\mathbf{x})$ can be evaluated using a *recursive formula*, by propagating probabilities. Define the short hand

$$\pi_t(\lambda) \triangleq p(x_0, \cdots, x_t,\, S_t = \lambda).$$

The desired $p(\mathbf{x})$ is given by

$$p(\mathbf{x}) = \sum_{\lambda \in \Lambda} \pi_n(\lambda). \tag{28}$$

Also note that

$$\pi_0(\lambda) = p(x_0 | S_0 = \lambda) p_0(\lambda). \tag{29}$$

For $1 \leq t \leq n$, we have the recursion

$$\begin{aligned} \pi_t(\lambda) &= p(x_0, \cdots, x_t,\, S_t = \lambda) \\ &= p(x_0, \cdots, x_{t-1},\, S_t = \lambda)\, p(x_t | x_1, \cdots, x_{t-1},\, S_t = \lambda) \\ &= p(x_t | S_t = \lambda) \sum_{\lambda' \in \Lambda} \pi_{t-1}(\lambda') Q(\lambda', \lambda). \end{aligned} \tag{30}$$

To summarize, $p(\mathbf{x})$ is obtained using the following recursion:

1. Compute $\pi_0(\lambda), \lambda \in \Lambda$, according to (29).
2. For $t = 1, \cdots, n$, compute $\pi_t(\lambda), \lambda \in \Lambda$, according to (30).
3. Obtain $p(\mathbf{x})$ from (28).

Hidden Markov Models for Images. A natural extension of HMMs to images would be to define a hidden MRF $\{S_i, i \in \mathcal{I}\}$ and observations $\{X_i, i \in \mathcal{I}\}$ statistically related to the state process $\{S_i, i \in \mathcal{I}\}$ via the conditional pmf

$$p(\mathbf{x}|\mathbf{s}) = \prod_{i \in \mathcal{I}} p(x_i | s_i).$$

For instance, consider an image made of three textured areas: grass, water, and sky. A *classification map* would assign a value $s_i \in \Lambda = \{1, 2, 3\}$ to each image pixel i to indicate which class this pixel belongs to. The spatial homogeneity of the classification map could be captured by a Potts model with three classes. The classification map is hidden though, and may be thought of as a state process. The observations might be modeled as conditionally independent Gaussian pixels X_i, conditioned on the state S_i. In other words, X_i is a mixture of three Gaussians, where the mixing parameters are spatially correlated.

7.3 Multiresolution Random Processes

Dyadic trees (two children per node) are often used for multiresolution analysis of 1-D signals. Likewise, quadtrees (four children per node) and other image pyramids have been popular for image analysis and processing. Let us see how we can associate statistical image models to trees.

Lowpass Pyramids. Consider the lowpass pyramid of Fig. 2. Denote by $\mathbf{x}^r$ the image at resolution level r, where $r = 0$ and $r = R$ are the finest and coarsest resolution levels, respectively. The domain of $\mathbf{x}^r$ is

$$\mathcal{I}_r = \{1, \cdots, 2^{-r} n_1\} \times \{1, \cdots, 2^{-r} n_2\}.$$

To each pixel x_i^R at the coarsest resolution level, we may associate a depth-R quadtree whose root is x_i^R and whose leaves represent a $2^R \times 2^R$ block of pixels at the finest resolution level. Now assume that

- the pixels x_i^R at the coarse level are statistically independent, and
- for all $r < R$ and $i \in \mathcal{I}_r$, pixel x_i^r is conditionally independent of its nondescendants, conditioned on its parent $x_{\pi(i)}^{(r+1)}$.

Then the image pyramid $\mathbf{x}^r, 0 \leq r \leq R$, may be viewed as a forest where each nonleaf node has four children.

An example of this approach may be found in the tutorial paper by Willsky [26], where the conditional pdf $p(x_i^r \mid x_{\pi(i)}^{(r+1)})$ is modeled as Gaussian with mean $\rho\, x_{\pi(i)}^{(r+1)}$ and some fixed variance. This defines a (linear) Gauss-Markov process. More generally, dependencies between parent and children may be nonlinear, as developed next.

Wavelets. The hierarchical subband representation associated with the wavelet decomposition also suggests quadtree models for the wavelet coefficients, as illustrated in Fig. 11. Simoncelli and his coworkers [28, 29, 30] have studied nonlinear models where each conditional pdf $p(x_i^r \mid x_{\pi(i)}^{(r+1)})$ is Gaussian with mean zero and variance $(x_{\pi(i)}^{r+1})^2 + \sigma^2$; this model captures the *clustering* of intensities across resolution levels and has been validated by experiments on various types of imagery.

Another possible type of multiresolution model captures the clustering of intensities within subbands. For instance, one may model the subbands as statistically independent MRFs. Consider any given subband and define $\mathcal{N}_i$ as the second-order neighborhood (8 pixels) of pixel i. A possible model for the local characteristics $p(x_i \mid x_{\mathcal{N}_i})$ of this MRF is Gaussian with mean zero and variance $\sum_{j \sim i} w_{j-i}\, x_j^2$, where $\{w_i\}$ are positive weights summing to 1. For instance, w could be the bilinear window

$$w_{(0,0)} = \frac{1}{4}, \quad w_{(0,\pm 1)} = w_{(\pm 1,0)} = \frac{1}{8}, \quad w_{(\pm 1,\pm 1)} = \frac{1}{16}.$$

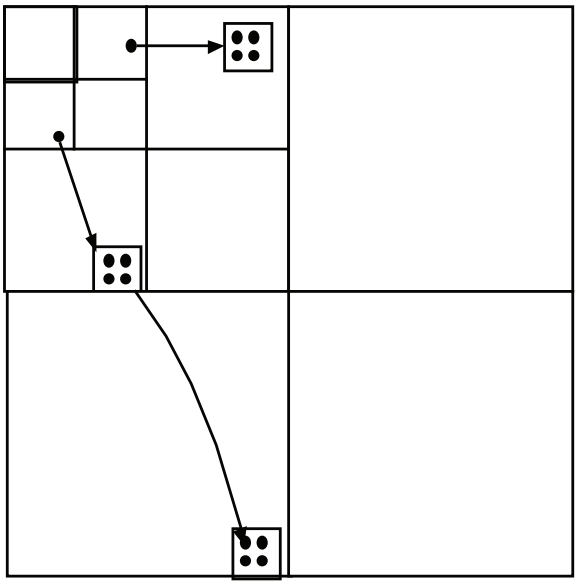


Figure 11: Interscale dependencies between wavelet coefficients.

This model is a variation of a model introduced by LoPresto *et al.* [31], to be further discussed below. A closely related model was developed by Simoncelli [28] modeling the conditional variance of x_i by $(\sum_{j\sim i} w_{j-i}\,|x_j|)^2 + \sigma_e^2$, where w_i and σ_e^2 are parameters to be adjusted.

Another approach that combines the aforementioned ideas is to enlarge the neighborhood $\mathcal{N}_i$ to include the parent of x_i. The paper [33] contains an example of that approach. There the conditional pdf $p(x_i \mid x_{\mathcal{N}_i})$ was not modeled as Gaussian but was estimated from image data. Due to the high dimensionality (ten) of the set of variables involved, the models used were of the form $p(x_i \mid t_i)$ where t_i is a predetermined function of $x_{\mathcal{N}_i}$ which may be thought of as a *summary statistic* for the neighborhood $\mathcal{N}_i$.

Generating Tree Models. Interestingly, some statistical models that do not have an obvious tree structure may be converted to trees. Consider for instance the anchored discrete-time Brownian motion $x(t), t = 1, 2, \cdots, N$, where the endpoint values $x(1)$ and $x(N)$ are given. Observe that the midpoint value $x(N/2)$ is independent of its conditional expectation, $\frac{1}{2}(x(1)+x(N))$. Next, observe that the values of $x(t)$ at times $t = 1, 2, \cdots, N/2$ are independent of the values at times $t = N/2+1, N/2+2, \cdots, N$ *if we condition on* $x(N/2)$. Recursively applying this reasoning, we obtain the tree of Fig. 12 in which the conditional dependencies are made explicit. Note that the nodes of the tree are labeled by three variables, e.g., the root node is labeled by $\{x(1), x(N/2), x(N)\}$. Moreover, there is redundancy in the tree, in that the same variables appear at different levels. At the bottom of the tree, all samples $x(1), \cdots, x(N)$ are present (some multiple times due to the redundancy).

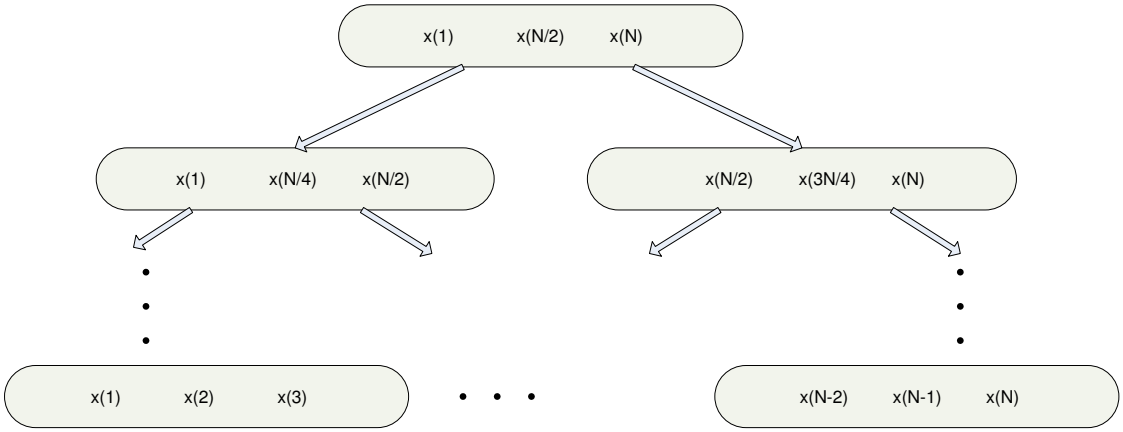


Figure 12: Tree representation of a Brownian process with anchored endpoints.

7.4 Hidden Markov Trees

A useful alternative to the MRF models of Sec. 7.3 can be obtained by defining a set of variances $\{s_i\}$ for the wavelet coefficients at all locations in all subbands and assuming the following hidden Markov model:

- The variances s_i form a hidden state sequence, and
- the wavelet coefficients are conditionally independent and Gaussian given their respective states, i.e., $p(\mathbf{x}|\mathbf{s}) = \prod_i p(x_i|s_i)$ where $p(x_i|s_i)$ is Gaussian with mean zero and variance s_i.

The spatial homogeneity of s_i may be captured using a simple MRF model, e.g., the Potts model of Sec. 7.2. This idea is similar to the model introduced by LoPresto *et al.* [31], where S_i were modeled as a unknown, deterministic, spatially-homogeneous field, to be estimated from the observed data.

Another way to capture some spatial coherence while capturing the clustering of s_i across resolution levels is via the Hidden Markov Tree (HMT) concept introduced by Crouse *et al.* for 1-D signals [34]; see [35] for an extension to images. The statistical dependencies between the state variables S_i are modeled by a tree with transition probabilities $p(s_i \mid s_{\pi(i)})$. This is a $L \times L$ matrix if the states have L possible values; the aforementioned papers used $L = 2$.

A closely related concept has been developed by Wainwright *et al.* [36] in the context of Gaussian scale mixtures (GSM). A GSM is a random vector $\mathbf{X} = S\mathbf{U}$, where $\mathbf{U}$ is a Gaussian vector and S is a scalar random variable. A GSM model may be thought of as an HMM model because S is hidden. By introducing dependencies between the values of S at different locations and/or resolution levels, one may construct HMTs and other fairly general models.

7.5 Directed Acyclic Graphs

We return to graph-theoretic notions and introduce directed acyclic graphs, which are often used to represent hierarchical models and Bayesian networks. A tree is a special kind of directed acyclic graph.

Recall our definition of $\pi(v)$ as the *set of parents* of node $v \in \mathcal{V}$. (A node may have 0, 1, 2, or more parents.) Consider a collection $\mathbf{X} = \{X_v,\, v \in \mathcal{V}\}$ of random variables indexed by the nodes of a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. The pmf for such $\mathbf{X}$ takes the form

$$p(\mathbf{x}) = \prod_{v \in \mathcal{V}} p(x_v | x_{\pi(v)}) \tag{31}$$

where we use the notational convention $p(x_v|x_{\pi(v)}) = p(x_v)$ if node v has no parent, i.e., $\pi(v) = \emptyset$. If $n(v)$ denotes the set of nondescendants of node v, we have the conditional independence property

$$p(x_v | x_{\pi(v)}, x_{\mathcal{U}}) = p(x_v | x_{\pi(v)}), \quad \forall \mathcal{U} \subseteq n(v).$$

A Markov chain is a chain-type directed acyclic graph where $\mathcal{V} = \{1, 2, \cdots, n\}$, and $\pi(v) = v - 1$. The pmf for the sequence $\mathbf{x}$ is obtained from the chain rule

$$p(\mathbf{x}) = p(x_1)p(x_2|x_1)\cdots p(x_n|x_{n-1}).$$

For a more general example, consider the graph $(\mathcal{V}, \mathcal{E})$ where $\mathcal{V} = \{1, 2, 3, 4\}$ and $\mathcal{E}$ consists of the following (directed) edges: $1 \to 2 \to 3 \to 4$ and $1 \to 3$. This graph is acyclic (but if we were to remove the arrows in the graph, nodes 1,2,3 would form a cycle.) In this directed graph, X_1 has two children but no parent, X_2 has one child and one parent, X_3 has one child and two parents, and X_4 has no children and one parent. The pmf $p(\mathbf{x})$ may be decomposed as

$$p(\mathbf{x}) = p(x_1)p(x_2|x_1)p(x_3|x_1, x_2)p(x_4|x_3).$$

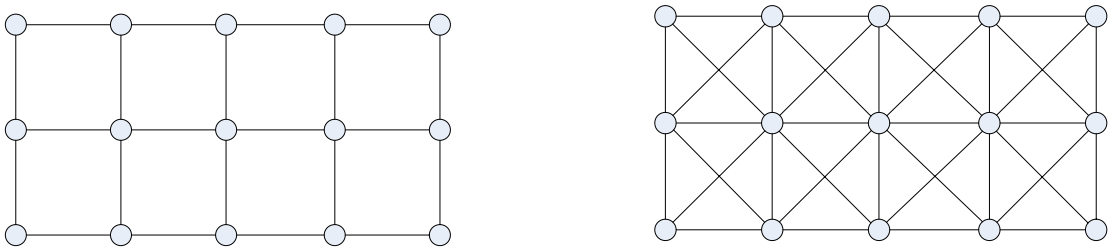
7.6 Undirected Graphs

Note that a MRF with neighborhood system $\mathscr{N} = \{\mathcal{N}_i,\, i \in \mathcal{I}\}$ may be viewed as an *undirected graph* with vertex set $\mathcal{I}$ and edge set $\mathcal{E}$ linking all pairs of neighbors $i \sim j$. Conversely, any undirected graph may be viewed as a representation of a neighborhood system.

A graphical representation of first- and second-order neighborhood systems is given in Fig. 13. Note the presence of loops. The cliques associated with $\mathscr{N}$ are also cliques of the graph $(\mathcal{I}, \mathcal{E})$.

More generally, consider any undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and a collection $\mathbf{X} = \{X_v,\, v \in \mathcal{V}\}$ of random variables indexed by the nodes of $\mathcal{G}$. The pmf of $\mathbf{X}$ takes the form

$$p(\mathbf{x}) = \frac{1}{Z} \prod_{\mathcal{C} \in \mathscr{C}} \psi_{\mathcal{C}}(x_{\mathcal{C}}) \tag{32}$$



(a) (b)

Figure 13: Graph representation of MRFs with (a) first-order neighborhood and (b) second-order neighborhood.

where Z is a normalization constant, $\mathscr{C}$ is a collection of cliques of the graph, and

$$\psi_{\mathcal{C}}(x_{\mathcal{C}}) = \exp\{-V_{\mathcal{C}}(x_{\mathcal{C}})\}$$

are exponentials of the clique potentials for a Gibbs random field.

7.6.1 Trees

Consider the tree (acyclic graph) of Fig. 14, which has 5 nodes and edges $1 \sim 2 \sim 3$ and $4 \sim 3 \sim 5$. We have

$$p(\mathbf{x}) = \frac{1}{Z}\psi_{12}(x_1, x_2)\psi_{23}(x_2, x_3)\psi_{34}(x_3, x_4)\psi_{35}(x_3, x_5).$$

As an example of inference on trees, consider the problem of evaluating the marginal pmf $p(x_5)$. We explore two approaches: the direct approach, which is computationally infeasible for large graphs; and the sum-product algorithm, which exploits the graph structure.

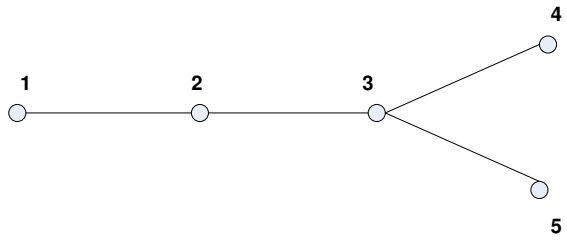


Figure 14: Tree (an undirected, acyclic graph).

Direct approach:

$$p(x_5) = \sum_{x_1,\cdots,x_4} \frac{1}{Z}\psi_{12}(x_1, x_2)\psi_{23}(x_2, x_3)\psi_{34}(x_3, x_4)\psi_{35}(x_3, x_5)$$

The number of terms in the sum is $|\Lambda|^4$. Clearly this approach becomes infeasible if the number of nodes is large (recall our previous discussion of HMMs).

Sum-Product Algorithm: Exploiting distributivity, we write

$$\begin{aligned}
p(x_5) &= \frac{1}{Z}\sum_{x_3}\psi_{35}(x_3,x_5)\sum_{x_4}\psi_{34}(x_3,x_4)\sum_{x_2}\psi_{23}(x_2,x_3)\underbrace{\sum_{x_1}\psi_{12}(x_1,x_2)}_{m_{12}(x_2)} \\
&= \frac{1}{Z}\sum_{x_3}\psi_{35}(x_3,x_5)\sum_{x_4}\psi_{34}(x_3,x_4)\underbrace{\sum_{x_2}\psi_{23}(x_2,x_3)m_{12}(x_2)}_{m_{23}(x_3)} \\
&= \frac{1}{Z}\sum_{x_3}\psi_{35}(x_3,x_5)m_{23}(x_3)\underbrace{\sum_{x_4}\psi_{34}(x_3,x_4)}_{m_{43}(x_3)} \\
&= \frac{1}{Z}\underbrace{\sum_{x_3}\psi_{35}(x_3,x_5)m_{23}(x_3)m_{43}(x_3)}_{m_{35}(x_5)}
\end{aligned}$$

In this derivation, nodes 1, 2, 4, 3 are eliminated in that order. We think of each term $m_{ij}(x_j)$ as a message conveyed from node i to node j, just before elimination of j. Computing $m_{ij}(x_j)$ involves a summation over all possible values x_i. This interpretation will be helpful in more complex problems.

For a general tree, upon choosing an elimination order, we evaluate the following messages in the corresponding order:

$$m_{ij}(x_j) = \sum_{x_i}\psi_{ij}(x_i,x_j)\prod_{k\in\mathcal{N}(i)\setminus\{j\}} m_{ki}(x_i) \tag{33}$$

A node can send a message to a neighbor once it has received messages from all of its other neighbors. The marginal probability at any node i is the product of all incoming messages:

$$p(x_i) = \frac{1}{Z}\prod_{k\in\mathcal{N}(i)} m_{ki}(x_i). \tag{34}$$

The computations (33) and (34) define the sum-product algorithm, or *belief propagation* algorithm. The choice of the elimination order has an impact on the total number of computations. The elimination order can be selected by inspection for simple graphs, but finding the optimal elimination order is NP-hard for general graphs.

7.6.2 Graphs with Cycles

Consider now the graph of Fig. 15, which has 5 nodes and 6 edges: $(1,2),(1,3),(2,4),(3,4),(3,5),(4,5)$. Notice the two cycles (1,2,3,4) and (3,4,5). The maximal clique for this graph is (3,4,5).

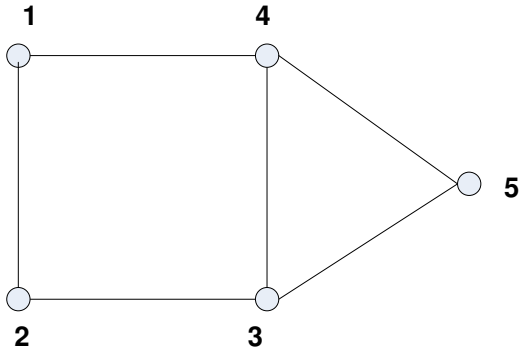


Figure 15: Undirected graph with two loops.

We have

$$p(\mathbf{x}) = \frac{1}{Z}\psi_{12}(x_1,x_2)\psi_{23}(x_2,x_3)\psi_{14}(x_1,x_4)\psi_{34}(x_3,x_4)\psi_{35}(x_3,x_5)\psi_{45}(x_4,x_5).$$

Let's find an efficient way to evaluate the marginal pmf $p(x_1)$. The direct approach is again infeasible for large graphs, but the sum-product algorithm works well.

Direct approach:

$$p(x_1) = \sum_{x_2,\cdots,x_6} \frac{1}{Z}\psi_{12}(x_1,x_2)\psi_{13}(x_1,x_3)\psi_{24}(x_2,x_4)\psi_{34}(x_3,x_4)\psi_{35}(x_3,x_5)\psi_{45}(x_4,x_5)$$

The number of terms in the sum is $|\Lambda|^5$.

Sum-Product Algorithm: Exploiting distributivity, write

$$\begin{aligned}
p(x_1) &= \frac{1}{Z}\sum_{x_2}\psi_{12}(x_1,x_2)\sum_{x_3}\psi_{23}(x_2,x_3)\sum_{x_4}\psi_{14}(x_1,x_4)\psi_{34}(x_3,x_4)\underbrace{\sum_{x_5}\psi_{35}(x_3,x_5)\psi_{45}(x_4,x_5)}_{m_{34}(x_3,x_4)} \\
&= \frac{1}{Z}\sum_{x_2}\psi_{12}(x_1,x_2)\sum_{x_3}\psi_{23}(x_2,x_3)\underbrace{\sum_{x_4}\psi_{14}(x_1,x_4)\psi_{34}(x_3,x_4)m_{34}(x_3,x_4)}_{m_{13}(x_1,x_3)} \\
&= \frac{1}{Z}\sum_{x_2}\psi_{12}(x_1,x_2)\underbrace{\sum_{x_3}\psi_{23}(x_2,x_3)m_{13}(x_1,x_3)}_{m_{12}(x_1,x_2)} \\
&= \frac{1}{Z}\underbrace{\sum_{x_2}\psi_{12}(x_1,x_2)m_{12}(x_1,x_2)}_{m_1(x_1)} \\
&= \frac{1}{Z}m_1(x_1).
\end{aligned}$$

Evaluating sums such as $m_{34}(x_3, x_4)$ for all possible values of (x_3, x_4) requires $|\Lambda|^3$ operations; evaluating $m_1(x_1)$ for all values of x_1 requires only $|\Lambda|^2$ operations.

For trees, all messages were of the form $m_{ij}(x_j)$ and could be evaluated using $|\Lambda|^2$ operations. For graphs with cycles, the messages may involve more variables. For a general problem with n nodes and maximum clique size c, the computational complexity of the sum-product algorithm is $O(n|\Lambda|^c)$ as opposed to $O(|\Lambda|^n)$ for the direct approach.

7.6.3 Junction-Tree Algorithm

There exists an interesting extension of the sum-product algorithm which converts the graph-based inference problem into a tree-based inference problem. The new tree is called *junction tree*, and its nodes are cliques of the original graph. The *junction-tree algorithm* may be used to compute marginals for these cliques (instead of single nodes of the original graph). The algorithm passes messages between nodes of the junction tree, which may be thought of as message passing on a tree of cliques.

The cliques are formed by triangulating the original graph as follows. Given a directed graph $\mathcal{G}$, the associated *moral graph* $\mathcal{G}^{\mathrm{m}}$ is defined as the undirected graph obtained by connecting all parents of each node in $\mathcal{G}$ and removing all arrowheads. Therefore, $\mathcal{C}_i \triangleq \{i, \pi(i)\}$ is a clique of $\mathcal{G}^{\mathrm{m}}$ for each $i \in \mathcal{V}$. Moreover, the conditional probability $p(x_i|x_{\pi(i)})$ is a potential function for the clique $\mathcal{C}_i$, and therefore (31) reduces to a special case of (32). The cliques of $\mathcal{G}^{\mathrm{m}}$ are the nodes of the junction tree, and cliques that have common vertices in $\mathcal{G}^{\mathrm{m}}$ are neighbors on the junction tree.

7.6.4 More on Distributivity

A variation of the sum-product algorithm is the max-product algorithm, which is useful to solve maximization problems such as the following.

Problem: Evaluate $M(x_1) = \max_{x_2,\cdots,x_6} p(x_1, x_2, \cdots, x_6)$.
This problem is very similar to that of evaluating the marginal $p(x_1)$, in that the direct calculation has complexity $|\Lambda|^5$, and a more efficient solution can be obtained exploiting distributivity and using the *max product algorithm*:

$$\begin{aligned} M(x_1) &= \frac{1}{Z} \max_{x_2} \psi_{12}(x_1, x_2) \max_{x_3} \psi_{13}(x_1, x_3) \max_{x_4} \psi_{24}(x_2, x_4)\psi_{34}(x_3, x_4) \underbrace{\max_{x_5} \psi_{35}(x_3, x_5)\psi_{45}(x_4, x_5)}_{m_{34}(x_3,x_4)} \\ &= \cdots \\ &= \frac{1}{Z} m_1(x_1). \end{aligned}$$

See [27] for a general approach to such problems.

Another related problem is to evaluate $\max_{x_2} p(x_2|x_1)$. This will be useful later to compute MAP estimates of an image. The solution is based on the fact that

$$\begin{aligned} p(x_2|x_1) &= \frac{p(x_1,x_2)}{p(x_1)} \\ \max_{x_2} p(x_2|x_1) &= \frac{\max_{x_2} p(x_1,x_2)}{\sum_{x_2} p(x_1,x_2)} \end{aligned}$$

i.e., conditioning (on the variable x_2) introduces no special difficulty.

7.7 Pattern Theory

To be continued...

References

[1] H. V. Poor, *An Introduction to Signal Detection and Estimation*, Springer-Verlag, 1994.

[2] B. Porat, *Digital Processing of Random Signals*, Prentice-Hall, Englewood Cliffs, NJ, 1994.

[3] R. L. Dobrushin, "The Description of a Random Field by Means of Conditional Probabilities and Conditions of Its Regularity," *Theory Prob. Appl.*, Vol. 13, pp. 197—224, 1968.

[4] J. Besag, "Spatial Interaction and the Statistical Analysis of Lattice Systems" (with discussion), J. Royal Statistical Society B, vol. 36, pp. 192-236, 1974.

[5] R. Kindermann and J. L. Snell, *Markov Random Fields and Their Applications*, American Mathematical Society, Providence, RI, 1980.

[6] S. Geman and D. Geman, "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images," *IEEE Trans. on PAMI*, Vol. 6, No. 6, pp. 721-741, Nov. 1984.

[7] K. Abend, T. J. Harley and L. N. Kanal, "Classification of Binary Random Patterns," *IEEE Trans. Information Theory*, Vol. 11, pp. 538—544, 1965.

[8] P. Brémaud, *Markov Chains, Gibbs Fields, Monte Carlo Simulations, and Queues*, Springer-Verlag, New York, 1999.

[9] H. Tamura, S. Mori and T. Yamawaki, "Textural Features Corresponding to Visual Perception," *IEEE Trans. Systems, Man and Cybernetics*, Vol. 8, pp. 460—473, 1978.

[10] G. R. Cross and A. K. Jain, "Markov Random Field Texture Models," *IEEE Transactions on PAMI*, Vol. 5, No. 1, pp. 44-58, Jan. 1983.

[11] C. A. Bouman and K. Sauer, "A Generalized Gaussian Image Model for Edge-Preserving MAP Estimation," *IEEE Transactions on Image Processing*, Vol. 2, pp. 296—310, 1993.

[12] A. Rosenfeld, *Multiresolution Image Processing and Analysis*, A. Rosenfeld, Ed., Springer–Verlag, 1984.

[13] P. Burt, "Multiresolution Techniques for Image Representation, Analysis, and 'Smart' Transmission," *SPIE* Vol. 1199, 1989.

[14] S. G. Mallat, "A Theory for Multiresolution Signal Decomposition: The Wavelet Transform," *IEEE Trans. Patt. Anal. and Mach. Intell.*, Vol. 11, No. 7, pp. 674—693, 1989.

[15] I. Daubechies, *Ten Lectures on Wavelets*, CBMS–NSF Regional Conference Series in Applied Mathematics, Vol. 61, SIAM, Philadelphia, 1992.

[16] M. Vetterli and J. Kovačević, *Wavelets and Subband Coding*, Prentice-Hall, Englewood Cliffs, NJ, 1995.

[17] S. G. Mallat, *A Wavelet Tour of Signal Processing*, Academic Press, San Diego, CA, 1998.

[18] B. J. Frey, *Graphical Models for Machine Learning and Digital Communication*, MIT Press, Cambridge, MA, 1998.

[19] M. I. Jordan, "Graphical Models," *Statistical Science*, Special issue on Bayesian statistics, Vol. 19, pp. 140—155, 2004.

[20] S. L. Lauritzen, *Graphical Models*, Oxford University Press, Oxford, UK, 1996.

[21] U. Grenander, *Elements of Pattern Theory*, Johns Hopkins U. Press, Baltimore, 1996.

[22] L. E. Baum and T. Petrie, "Statistical Inference for Probabilistic Functions of Finite State Markov Chains," *Ann. Math. Stat.*, Vol. 37, pp. 1554—1563, 1966.

[23] L. E. Baum, T. Petrie, G. Soules and N. Weiss, "A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains," *Ann. Math. Stat.*, Vol. 41, No. 1, pp. 164—171, 1970.

[24] L. E. Rabiner and B. H. Juang, "An Introduction to Hidden Markov Models," *IEEE ASSP Magazine*, Vol. 3, No. 1, pp. 4—16, 1986.

[25] L. E. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," *Proc. IEEE*, Vo. 72, No. 2, pp. 257—286, 1989.

[26] A. S. Willsky, "Multiresolution Markov Models for Signal and Image Processing," *Proc. IEEE*, Vol. 90, No. 8, pp. 1396—1458, Aug. 2002.

[27] S. A. Aji and R. J. McEliece, "The Generalized Distributive Law," *IEEE Trans. on Information Theory*, Vol. 46, No. 2, pp. 325—343, 2000.

[28] E. Simoncelli, "Statistical Models for Images: Compression, Restoration and Synthesis," *31st Asilomar Conf.*, Nov. 1997.

[29] R. W. Buccigrossi and E. P. Simoncelli, "Image Compression via Joint Statistical Characterization in the Wavelet Domain," *IEEE Trans. on Image Processing*, Vol. 8, No. 12, pp. 1688–1701, 1999.

[30] J. Portilla and E. P. Simoncelli, "A Parametric Texture Model based on Joint Statistics of Complex Wavelet Coefficients," *Int. J. Computer Vision*, Vol. 40, No. 1, pp. 49—71, Dec. 2000.

[31] S. LoPresto, K. Ramchandran and M. T. Orchard, "Image Coding Based on Mixture Modeling of Wavelet Coefficients and a Fast Estimation-Quantization Framework," *Proc. Data Compression Conference*, pp. 221—230, Snowbird, UT, 1997.

[32] M. Malfait and D. Roose, "Wavelet-Based Image Denoising Using a Markov Random Field a Priori Model," *IEEE Trans. on Image Processing*, Vol. 6, pp. 549—565, 1997.

[33] J. Liu and P. Moulin, "Information-Theoretic Analysis of Interscale and Intrascale Dependencies Between Image Wavelet Coefficients," *IEEE Trans. on Image Processing*, Vol. 10, No. 10, pp. 1647—1658, Nov. 2001.

[34] M. S. Crouse, R. D. Nowak and R. G. Baraniuk, "Wavelet-Based Statistical Signal Processing Using Hidden Markov Models," *IEEE Trans. on Signal Processing*, Vol. 46, No. 4, pp. 886—902, Apr. 1998.

[35] J. K. Romberg, H. Choi and R. G. Baraniuk, "Bayesian Tree-Structured Modeling Using Wavelet-Domain Hidden Markov Models," *IEEE Trans. on Image Processing*, Vol. 10, No. 7, pp. 10566—1068, July 2001.

[36] M. J. Wainwright, A. S. Willsky and E. P. Simoncelli, "Random Cascades on Wavelet Trees and Their Use in Analyzing and Modeling Natural Images," *J. Applied Computational and Harmonic Analysis*, special issue on wavelets, Spring 2001.