Proceedings of Int'l Conf. on Computer Vision and Pattern Recognition, South Carolina, June, 2000.

Learning in Gibbsian Fields: How Accurate and How Fast Can It Be?

Song-Chun Zhu and Xiuwen Liu

Department of Computer and Information Sciences
The Ohio State University, Columbus, OH 43210
$\{szhu, liux\}$@cis.ohio-state.edu

Abstract

In this article, we present a unified framework for learning Gibbs models from training images. We identify two key factors that determine the accuracy and speed of learning Gibbs models: 1). Fisher information, and 2). the accuracy of Monte Carlo estimate for partition functions. We propose three new learning algorithms under the unified framework. I). The maximum partial likelihood estimator*. II). The* maximum patch likelihood estimator*, and III). The* maximum satellite likelihood estimator*. The first two algorithms can speed up the minimax entropy algorithm by about 20 times without losing much accuracy. The third one makes use of a set of known Gibbs models as references – dubbed "satellites" and can approximately estimate the minimax entropy model in the speed of* 10 *seconds, which may take many hours for the minimax entropy method in[14].*

1 Introduction

Gibbsian fields or Markov random fields are widely used in Bayesian image analysis. However, learning accurate Gibbs models from training images is hard and time consuming. In the literature, most existing work focus on simplified auto-models[6, 5], such as auto-binomial models and Gaussian random fields, for which the parameters are analytically computable.

For Gibbs models beyond the auto-model families, there are three basic solutions. 1). The *maximum pseudo-likelihood estimator* (MPLE) [3]. 2). A *stochastic gradient* algorithm[10, 12, 14]. 3). A *Markov chain Monte Carlo MLE* (MCMCMLE) method[8, 7], which is, in practice, the same as the second one. These algorithms are originally developed from statistical physics with pair cliques. The MPLE is often *inefficient*, i.e. the estimated parameters have large variance, while the second and third methods are very slow in convergence.

The computational difficulties are pronounced by more realistic Gibbs models (FRAME) derived by the minimax entropy learning theory[14]. For example, learning a typical texture model may take many hours. Since the publication of the minimax entropy learning theory, four new algorithms are proposed to improve its speed with mixed success: 1). a minuteman minimax[4], 2). a variational method[2, 1], 3). learning by diffusion[11], 4). a generalized MPLE (private communication with Wu).

In this article, we study a common statistical framework for learning general Gibbs models with an emphasis on the computational complexity. We identify two key factors that determine the accuracy and speed of learning Gibbs models. I). The variance of the estimated parameters or the inverse *Fisher information*. II). The accuracy of estimating partition functions by Monte Carlo integration. As it is illustrated in figure 12, existing algorithms can be unified and mapped along these two dimensions.

We propose three new learning algorithms under an unified framework which trades off between accuracy and speed: 1). *maximum partial likelihood estimator*, 2). *maximum patch likelihood estimator*, and 3). *maximum satellite likelihood estimator*. The first two algorithms speed up the minimax entropy algorithm by about 20 times without losing much accuracy. The third algorithm makes use of a set of known Gibbs models as references-dubbed "satellites" which are computed off-line. It can approximately estimate the minimax entropy model in the speed of 10 seconds. The learned models are evaluated in texture synthesis experiments in section (5).

2 Basic ideas of Gibbs learning

Given a stochastic visual signal $\mathbf{I}$, for example, a texture image, the goal of visual learning is to compute a probability model $p(\mathbf{I})$, which approaches an underlying "true" distribution $f(\mathbf{I})$ by minimizing the Kullback-Leibler divergence.

$$\begin{aligned} KL(f||p) &= \int f(\mathbf{I}) \log \frac{f(\mathbf{I})}{p(\mathbf{I})} d\mathbf{I} \\ &= E_f[\log f(\mathbf{I})] - E_f[\log p(\mathbf{I})]. \end{aligned}$$

In practice, the expectation $E_f[\log p(\mathbf{I})]$ is estimated by the sample mean using samples $\{\mathbf{I}_1^{\text{obs}}, \mathbf{I}_2^{\text{obs}}, ..., \mathbf{I}_M^{\text{obs}}\}$

drawn from $f(\mathbf{I})$. Thus minimizing $KL(f||p)$ leads to the conventional *maximum likelihood estimator* (MLE).

$$\hat{p} = \arg\max_{p \in \Omega_p} \sum_{i=1}^{M} \log p(\mathbf{I}_i^{\text{obs}}), \tag{1}$$

where Ω_p is the family of distributions for model $p(\mathbf{I})$.

One choice of Ω_p is the exponential family Ω_k derived from the *maximum entropy principle*, with k indexing the number of statistics in the model.

$$\begin{aligned} p(\mathbf{I}; \boldsymbol{\beta}) &= \frac{1}{Z(\boldsymbol{\beta})} \exp\{-\sum_{j=1}^{k} \beta_j h_j(\mathbf{I})\} \\ &= \frac{1}{Z(\boldsymbol{\beta})} \exp\{- < \boldsymbol{\beta}, \mathbf{h}(\mathbf{I}) >\}, \end{aligned} \tag{2}$$

where $\boldsymbol{\beta} = (\beta_1, \beta_2, ..., \beta_k)$ are parameters, and $\mathbf{h}(\mathbf{I}) = (h_1(\mathbf{I}), h_2(\mathbf{I}), ..., h_k(\mathbf{I}))$ are the feature statistics of $\mathbf{I}$. $p(\mathbf{I}; \boldsymbol{\beta})$ is called a Gibbsian field on a lattice system. For example, in modeling texture and natural images[14], the statistics $h_j(\mathbf{I}), j = 1, 2, ..., k$, are chosen to be the histograms of filter responses, and $\beta_j, j = 1, 2, .., k$, are vector-valued approximation to continuous potential functions. Notice that $\mathbf{h}(\mathbf{I})$ can be other statistics, such as joint histograms, with $p(\mathbf{I}; \boldsymbol{\beta})$ being the same form.

The minimax learning theory proceeds in two phases[14].

Phase I augments the dimension of the spaces through newly selected features and generates a sequence of exponential families,

$$\Omega_0, \Omega_1, ..., \Omega_K \longrightarrow \Omega_f \ni f.$$

With more and more information bearing features included in $p(\mathbf{I}; \boldsymbol{\beta})$, it is proved that the exponential family can approach the space Ω_f which contains the true model $f(\mathbf{I})$ [14].

Phase II computes the parameters $\boldsymbol{\beta}$ by maximizing the log-likelihood (see equation (1)) in space $\Omega_p = \Omega_k$.

In [14], these two phases are computed iteratively. So a Gibbs model is learned for each family in the sequence. This process can take many hours when large filters are used.

Recently a technical breakthrough has been achieved, which shows that the two learning phases are separable[13]. Thus the model selection phase can be done using the so-called Julesz ensembles[15] and this has largely reduced the computational complexity.

In this paper, we focus on phase II – the maximum likelihood estimation of a general Gibbs model.

3 A unifying framework

To tight up our notations, we denote by $\mathbf{I}_\Lambda$ an image defined in a lattice Λ. It was shown in [13, 15] that for a given set of observed feature statistics $\mathbf{h}(\mathbf{I})$ all Gibbs models on finite lattices Λ originate from a uniform distribution on the 2D plane $\mathbf{Z}^2$, which is called the Julesz ensembles[13]. Gibbs models are conditional distributions of this uniform distribution. So the boundary conditions must be included in Gibbs distributions,

$$p(\mathbf{I}_\Lambda | \mathbf{I}_{\partial\Lambda}) = \frac{1}{Z(\mathbf{I}_{\partial\Lambda}, \boldsymbol{\beta})} \exp\{- < \boldsymbol{\beta}, \mathbf{h}(\mathbf{I}_\Lambda | \mathbf{I}_{\partial\Lambda}) >\}.$$

The boundary condition is often dropped for convenience when the area of boundary condition is negligible in comparison with the size of the lattice. For example, $\frac{|\partial\Lambda|}{|\Lambda|} \approx 1/20$. The width w of $\partial\Lambda$ is determined by the window size of feature extractors such as Gabor filters.

One remarkable property of the Gibbs model is that $\boldsymbol{\beta}$ *is invariant to the sizes and shapes of* Λ. *Therefore it can be estimated on arbitrary* Λ.

Without loss of generality, we assume we have only one observed image $\mathbf{I}_\Lambda^{\text{obs}}$, with Λ being $N \times N$ pixels. We may view $\mathbf{I}_\Lambda^{\text{obs}}$ as a patch from an infinity texture image on $\mathbf{Z}^2$. Equally we may break it into smaller patches $\mathbf{I}_{\Lambda_i}^{\text{obs}}, i = 1, 2, ..., M$ on lattices Λ_i of arbitrary shapes and sizes[1]. These patches may overlap with each other. Then $\boldsymbol{\beta}$ is learned by maximizing the following log-likelihood,

$$\mathcal{G}(\boldsymbol{\beta}) = \sum_{i=1}^{M} \log p(\mathbf{I}_{\Lambda_i}^{\text{obs}} | \mathbf{I}_{\partial\Lambda_i}^{\text{obs}}; \boldsymbol{\beta}). \tag{3}$$

Now we show that existing learning algorithms are unified as MLE maximizing $\mathcal{G}(\boldsymbol{\beta})$ and differ in the following two choices.

Choice I: the number, sizes and shapes of the *foreground* **patches** $\Lambda_i, i = 1, 2, ..., M$**.**

Figure 1 displays four typical choices for Λ_i. The bright pixels are in the *foreground* $\Lambda_i, i = 1, 2, ..., M$, which are surrounded by dark pixels in the *background* $\partial\Lambda_i, i = 1, 2, ..., M$.

In the first three cases, Λ_i are square patches with $m \times m$ pixels. In one extreme at figure 1.a, the stochastic gradient[14] and MCMCMLE[7] methods choose one largest patch denoted by Λ_1, i.e. $M = 1$ and $m = N - 2w$ with w being the width of the boundary. In the other extreme in figure 1.c, the MPLE method[3] chooses the minimum patch size $m = 1$ and maximum number of patches $M = N - 2w$. That is, it chooses one patch for each pixel. Figure 1.b is an

[1] $\Lambda_i, i = 1, 2, ..., M$ are chosen so that the boundary conditions $\partial\Lambda_i$ are observed as well, in order to avoid the complication of learning with missing data.

example between the two extremes. Another choice in figure 1.d chooses only one ($M = 1$) irregular-shaped patch, denoted by Λ_1, where Λ_o is a set of randomly selected pixels with the rest pixels being the boundary condition $\partial\Lambda_1$.

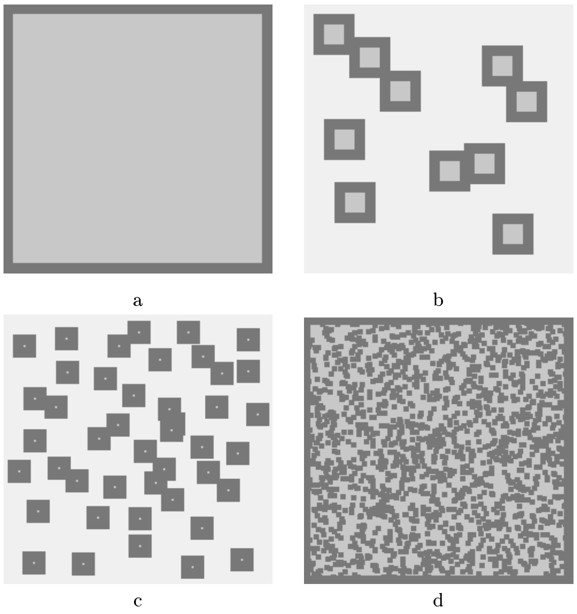


Figure 1: Choices of boundary conditions. The bright pixels are in the foreground which are surrounded by dark pixels in the background. a). Likelihood. b). Patch likelihood and satellite likelihood, c). Pseudo likelihood. d). Partial likelihood.

$\mathcal{G}(\boldsymbol{\beta})$ is called likelihood, patch likelihood, pseudo likelihood, and partial likelihood for the four cases respectively. It is trivial to prove that maximizing $\mathcal{G}(\boldsymbol{\beta})$ leads to a consistent estimator for all four choices. The proof follows directly from[9].

Choice II: the reference models used for estimating the partition function $Z(\boldsymbol{\beta})$.

Once we have chosen the foreground lattices, the second step for estimating $\boldsymbol{\beta}$ is to approximate the partition function $Z(\mathbf{I}^{\text{obs}}_{\partial\Lambda_i}, \boldsymbol{\beta})$ for each $\Lambda_i, i = 1, ..., M$ by Monte Carlo integration using a reference model at $\boldsymbol{\beta}_o$.

$$\begin{aligned}
& Z(\mathbf{I}^{\text{obs}}_{\partial\Lambda_i}, \boldsymbol{\beta}) \\
= & \int \exp\{- < \boldsymbol{\beta}, \mathbf{h}(\mathbf{I}_{\Lambda_i}|\mathbf{I}^{\text{obs}}_{\partial\Lambda_i}) >\} d\mathbf{I}_{\Lambda_i}, \\
= & Z(\mathbf{I}_{\partial\Lambda_i}, \boldsymbol{\beta}_o) \int \exp\{- < \boldsymbol{\beta} - \boldsymbol{\beta}_o, \mathbf{h}(\mathbf{I}_{\Lambda_i}|\mathbf{I}^{\text{obs}}_{\partial\Lambda_i}) >\} dp(\mathbf{I}|\mathbf{I}^{\text{obs}}_{\partial\Lambda_i}; \boldsymbol{\beta}_o) \\
\approx & \frac{Z(\mathbf{I}^{\text{obs}}_{\partial\Lambda_i}, \boldsymbol{\beta}_o)}{L} \sum_{i=1}^{L} \exp\{- < \boldsymbol{\beta} - \boldsymbol{\beta}_o, \mathbf{h}(\mathbf{I}^{\text{syn}}_{ij}|\mathbf{I}^{\text{obs}}_{\partial\Lambda_i}) >\}.
\end{aligned} \tag{4}$$

$\mathbf{I}^{\text{syn}}_{ij}, j = 1, 2, ..., L$ are *typical* samples from the reference model $p(\mathbf{I}_{\Lambda_i}|\mathbf{I}^{\text{obs}}_{\partial\Lambda_i}; \boldsymbol{\beta}_o)$.

Therefore, maximizing the estimated log-likelihood $\mathcal{G}(\boldsymbol{\beta})$ by gradient descent, we have,

$$\frac{d\boldsymbol{\beta}}{dt} = \sum_{i=1}^{M}\{\sum_{j=1}^{L} \omega_{ij}\mathbf{h}(\mathbf{I}^{\text{syn}}_{ij}|\mathbf{I}^{\text{obs}}_{\partial\Lambda_i}) - \mathbf{h}(\mathbf{I}^{\text{obs}}_{\Lambda_i}|\mathbf{I}^{\text{obs}}_{\partial\Lambda_i})\}. \tag{5}$$

ω_{ij} is the weight for sample $\mathbf{I}^{\text{syn}}_{ij}$,

$$\omega_{ij} = \frac{\exp\{- < \boldsymbol{\beta} - \boldsymbol{\beta}_o, \mathbf{h}(\mathbf{I}^{\text{syn}}_{ij}|\mathbf{I}^{\text{obs}}_{\partial\Lambda_i}) >\}}{\sum_{j'=1}^{L} \exp\{- < \boldsymbol{\beta} - \boldsymbol{\beta}_o, \mathbf{h}(\mathbf{I}^{\text{syn}}_{ij'}|\mathbf{I}^{\text{obs}}_{\partial\Lambda_i}) >\}}$$

The choices of the reference models $p(\mathbf{I}_{\Lambda_i}|\mathbf{I}^{\text{obs}}_{\partial\Lambda_i}; \boldsymbol{\beta}_o)$ depend on the size of the patches Λ_i. In general the importance sampling is only valid when the two distributions $p(\mathbf{I}_{\Lambda_i}|\mathbf{I}^{\text{obs}}_{\partial\Lambda_i}; \boldsymbol{\beta}_o)$ and $p(\mathbf{I}_{\Lambda_i}|\mathbf{I}^{\text{obs}}_{\partial\Lambda_i}; \boldsymbol{\beta})$ heavily overlap.

In one extreme case $m = 1$, the MPLE method[3] selects $\boldsymbol{\beta}_o = 0$ and $p(\mathbf{I}_{\Lambda_i}|\mathbf{I}^{\text{obs}}_{\partial\Lambda_i}; \boldsymbol{\beta}_o)$ is a uniform distribution. In the other extreme cases ($m = N - 2w$), the stochastic gradient and the MCMCMLE methods must choose $\boldsymbol{\beta}_o = \boldsymbol{\beta}$, otherwise the two ensembles do not overlap.

To summarize, figure 2 illustrates the two factors that determine the accuracy and speed of learning $\boldsymbol{\beta}$. These curves are verified through experiments in section 5. The horizontal axis is the size of an individual foreground lattice $|\Lambda_i|$.

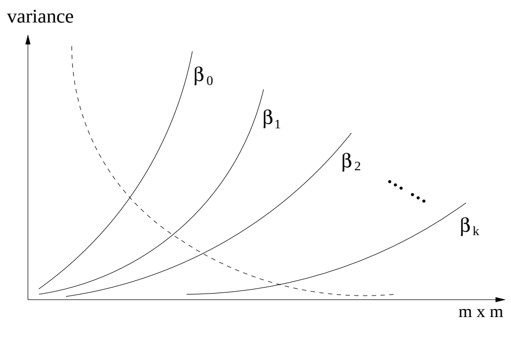


Figure 2: Estimation variances for choices of the patch size $m \times m$ and reference models.

1. *The variances of MLE or inverse Fisher information.* Let $\hat{\boldsymbol{\beta}}(\mathbf{I}^{\text{obs}})$ be the estimator maximizing $\mathcal{G}(\boldsymbol{\beta})$ and let $\boldsymbol{\beta}^* = \arg\min KL(f||p)$ be the optimal solution in Ω_k. The dashed curve in figure 2 illustrates the variance,

$$E_f[(\hat{\boldsymbol{\beta}}(\mathbf{I}^{\text{obs}}) - \boldsymbol{\beta}^*)^2].$$

For choices shown in figure 1, if we fix the total number of pixels $\sum_{i=1}^{M} |\Lambda_i|$ in the foreground, then the variance decreases as the patch size (diameter of the hole) increases.

2. *The variance of estimating Z by Monte Carlo integration,*

$$E_p[(\hat{Z}-Z)^2]$$

increases with the lattice sizes for a given reference model β_o (see the solid curves). Therefore, for very large patches, such as $m = 200$, we must construct a sequence of reference models to approach β,

$$\beta_0 = 0,\ \beta_1,\ \beta_2,\ ...,\ \beta_k\ \rightarrow\ \beta.$$

This is the major reason for why the stochastic gradient algorithm was so slow in [14]!

4 New algorithms

We propose three algorithms which trade-off between accuracy and speed based on our analysis in section 3.

Algorithm I: maximizing partial likelihood.

We choose the random boundary condition shown in figure 1.d by masking a certain percentage (say 30%) of pixels on Λ_1. This partitions the observed image into

$$\mathbf{I}^{\rm obs}_{\Lambda} = \mathbf{I}^{\rm obs}_{\Lambda_1} \cup \mathbf{I}^{\rm obs}_{\Lambda/\Lambda_1},$$

with $\mathbf{I}^{\rm obs}_{\Lambda/\Lambda_1}$ being the boundary condition.

We define a *partial likelihood*

$$\mathcal{G}_1(\beta) = \log p(\mathbf{I}^{\rm obs}_{\Lambda_1} | \mathbf{I}^{\rm obs}_{\Lambda/\Lambda_1}; \beta).$$

Maximizing $\mathcal{G}_1(\beta)$ by gradient descent, we update β iteratively.

$$\frac{d\beta}{dt} = \mathbf{h}(\mathbf{I}^{\rm obs}_{\Lambda_1} | \mathbf{I}^{\rm obs}_{\Lambda/\Lambda_1}) - \mathbf{h}(\mathbf{I}^{\rm syn}_{\Lambda_1} | \mathbf{I}^{\rm obs}_{\Lambda/\Lambda_1}). \qquad (6)$$

It starts with $\beta = 0$ and synthesizes $\mathbf{I}^{\rm syn}_{\Lambda_1}$ for each β by filling the masked pixels, until β converges.

This algorithm is virtually the same as the original algorithm in [14]. It trades off between accuracy and speed in an interesting way. The partial likelihood has lower Fisher information than the likelihood, and our experiments demonstrate that it is about 25 times faster than the original minimax learning method[14] without losing much accuracy. Our observation is that the original sampling method[14] spends a major portion of its time synthesizing $\mathbf{I}^{\rm syn}_{\Lambda_1}$ under "non-typical" boundary conditions $\mathbf{I}_{\Lambda/\Lambda_1}$ starting with white noise images. In contrast, the new algorithm works on typical boundary condition $\mathbf{I}^{\rm obs}_{\Lambda/\Lambda_1}$ where the probability mass of the Gibbs model $p(\mathbf{I};\beta)$ is focused on.

Algorithm II: maximizing patch likelihood.

We choose a set of M overlapping patches from $\mathbf{I}^{\rm obs}_{\Lambda}$ and "dig" a hole Λ_i on each patch (see figure 1.b).

Thus we define a *patch log-likelihood*

$$\mathcal{G}_2(\beta) = \sum_{i=1}^{M} \log p(\mathbf{I}^{\rm obs}_{\Lambda_i} | \mathbf{I}^{\rm obs}_{\partial\Lambda_i}; \beta).$$

Maximizing $\mathcal{G}_2(\beta)$ by gradient descent, again we update β iteratively.

$$\frac{d\beta}{dt} = \sum_{i=1}^{M} \mathbf{h}(\mathbf{I}^{\rm syn}_{\Lambda_i} | \mathbf{I}^{\rm obs}_{\Lambda/\Lambda_i}) - \sum_{i=1}^{M} \mathbf{h}(\mathbf{I}^{\rm obs}_{\Lambda_i} | \mathbf{I}^{\rm obs}_{\Lambda/\Lambda_i}). \qquad (7)$$

It starts with $\beta = 0$ and synthesizes $\mathbf{I}^{\rm syn}_{\Lambda_i}, i = 1, 2, ..., M$ for each sequential β by filling the holes, until β converges. Like algorithm I, it trades off between speed and accuracy in a similar way.

Algorithm III: maximizing satellite likelihood

Algorithms I and II can compute β accurately, however, both still need to synthesize images on line, which is computationally intensive. Now we propose an algorithm which may approximately compute β in the speed of a few seconds without synthesizing images on-line!

We select a set of reference models in the exponential family Ω_k to which the Gibbs model $p(\mathbf{I};\beta)$ belongs,

$$\mathcal{R} = \{p(\mathbf{I};\beta_j)\ :\ \beta_j \in \Omega_k, j = 1, 2, ..., s.\}$$

We drop the notation of lattice as we study models on large images. We sample one large typical image $\mathbf{I}^{\rm syn}_j \sim p(\mathbf{I};\beta_j)$ for each reference model off-line. These reference models estimate β in Ω_k from different "viewing angles". By analogy to the global positioning system (GPS), we call the reference models the "satellites".

The *satellite log-likelihood* is defined as

$$\mathcal{G}_3(\beta) = \mathcal{G}^{(1)}_3(\beta;\beta_1) + \mathcal{G}^{(2)}_3(\beta;\beta_2) + \cdots + \mathcal{G}^{(s)}_3(\beta;\beta_s), \qquad (8)$$

where each satellite contributes one likelihood approximation,

$$\mathcal{G}^{(j)}_3(\beta;\beta_j) = \sum_{i=1}^{M} \log \frac{1}{\hat{Z}^{(j)}_i} \exp\{- < \beta, \mathbf{h}(\mathbf{I}^{\rm obs}_{\Lambda_i} | \mathbf{I}^{\rm obs}_{\partial\Lambda_i}) >\}. \qquad (9)$$

Following the importance sampling method in equation (4), we estimate $Z(\mathbf{I}^{\rm obs}_{\partial\Lambda_i}, \beta)$ by Monte Carlo integration.

$$\hat{Z}^{(j)}_i = \frac{Z(\mathbf{I}^{\rm obs}_{\partial\Lambda_i}, \beta_j)}{L} \sum_{\ell=1}^{L} \exp\{- < \beta - \beta_j, \mathbf{h}(\mathbf{I}^{\rm syn}_{ij\ell} | \mathbf{I}^{\rm obs}_{\partial\Lambda_i}) >\}. \qquad (10)$$

Notice that for each hole Λ_i and for each reference model $p(\mathbf{I}; \boldsymbol{\beta}_j)$, we have a set of L synthesized patches $\mathbf{I}^{\text{syn}}_{ij\ell}$ to fill the hole:

$$H^{\text{syn}}_{ij} = \{\mathbf{I}^{\text{syn}}_{ij\ell}; \ell = 1, 2, ..., L, \forall i, j\}.$$

There are two ways for generating H^{syn}_{ij}.

1. Sampling $\mathbf{I}^{\text{syn}}_{ij\ell} \sim p(\mathbf{I}_{\Lambda_i} | \mathbf{I}^{\text{obs}}_{\partial\Lambda_i}; \boldsymbol{\beta}_j)$ – using the conditional distribution. This is expensive.
2. Sampling $\mathbf{I}^{\text{syn}}_{ij\ell} \sim p(\mathbf{I}_{\Lambda_i}; \boldsymbol{\beta}_j)$ – using the marginal distribution. In practice, this is just to fill the holes with randomly selected patches from the $\mathbf{I}^{\text{syn}}_j$ which was synthesized off-line. Thus no sampling is needed for on-line computation.

In our experiments, we tried both cases and we found that differences are very little for middle sizes lattices, say $4 \leq m \leq 13$.

Maximizing $\mathcal{G}_3(\boldsymbol{\beta})$ by gradient ascent, we have,

$$\frac{d\boldsymbol{\beta}}{dt} = \sum_{j=1}^{s} \{ \sum_{i=1}^{M} [\sum_{\ell=1}^{L} \omega_{ij} \mathbf{h}(\mathbf{I}^{\text{syn}}_{ij\ell} | \mathbf{I}^{\text{obs}}_{\partial\Lambda_i}) \; - \; \mathbf{h}(\mathbf{I}^{\text{obs}}_{\Lambda_i} | \mathbf{I}^{\text{obs}}_{\partial\Lambda_i})] \} \tag{11}$$

ω_{ij} is the weight for sample $\mathbf{I}^{\text{syn}}_{ij\ell}$,

$$\omega_{ij\ell} = \frac{\exp\{- < \boldsymbol{\beta} - \boldsymbol{\beta}_o, \mathbf{h}(\mathbf{I}^{\text{syn}}_{ij\ell} | \mathbf{I}^{\text{obs}}_{\partial\Lambda_i}) >\}}{\sum_{\ell'=1}^{L} \exp\{- < \boldsymbol{\beta} - \boldsymbol{\beta}_o, \mathbf{h}(\mathbf{I}^{\text{syn}}_{ij\ell'} | \mathbf{I}^{\text{obs}}_{\partial\Lambda_i}) >\}}$$

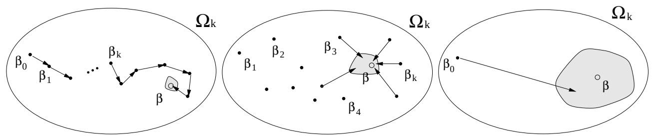
Equation (11) converges in the speed of seconds for an average texture model.

However, we should be also aware of the risk that the satellite likelihood $\mathcal{G}_3(\boldsymbol{\beta})$ may not be upper bounded. It is almost surely not upper bounded for the MCMCMLE method[7]. This case occurs when $\mathbf{h}(\mathbf{I}^{\text{obs}}_{\Lambda_i} | \mathbf{I}^{\text{obs}}_{\partial\Lambda_i})$ cannot be described by the linear combination of the statistics of the sampled patches $\sum_{\ell=1}^{L} \omega_{ij} \mathbf{h}(\mathbf{I}^{\text{syn}}_{ij\ell} | \mathbf{I}^{\text{obs}}_{\partial\Lambda_i})$. We handle this problem by including the observed patch $\mathbf{I}^{\text{obs}}_{\Lambda_i}$ in H^{syn}_{ij}, therefore the satellite likelihood is always upper bounded.

Intuitively, since $\mathbf{I}^{\text{syn}}_{ij1} = \mathbf{I}^{\text{obs}}_{\Lambda_i}$, the $\boldsymbol{\beta}$ is learned so that the conditional probabilities $\omega_{ij1} \to 1$ and $\omega_{ij\ell} \to 0$, $\forall \ell \neq 1$. Therefore $\mathbf{I}^{\text{obs}}_{\Lambda_i}$ is treated as a positive example and others are negative examples.

To summarize, figure 3 compare the three groups of algorithms in a second unified view:

Group 1: the maximum likelihood estimators (including stochastic gradient and MCMCMLE) and maximum partial/patch likelihood estimators. As figure 3.a illustrates generate and sample a sequence of "satellites" $\boldsymbol{\beta}_0, \boldsymbol{\beta}_1, ..., \boldsymbol{\beta}_k$ on-line. These satellites get closer and closer to $\boldsymbol{\beta}$ in order to locate $\boldsymbol{\beta}$ precisely. The shadow area represents the uncertainty in computing $\boldsymbol{\beta}$, whose size can be measured by the Fisher information.



a any m b $m \sim 10$ c $m = 1$

Figure 3: In space Ω_k, each probability model corresponds to a point θ. The shadow areas illustrates the variance of the estimated $\boldsymbol{\beta}$. MLE and MPLE are considered unified in satellite approximation.

Group 2: the maximum pseudo likelihood estimator. As figure 3.c shows, it uses the uniform model $\boldsymbol{\beta}_o = 0$ as "satellite" and has large variation in estimating $\boldsymbol{\beta}$.

Group 3: the maximum satellite likelihood estimators. These methods use a general set of satellites which are computed and sampled off-line. The shadow area is relatively larger because of the Monte Carlo integration. It can improve by adding new satellites through the learning process. To save time it may select a small subset of satellites for computing a given model. One can choose a satellite based on the difference $\mathbf{h}(\mathbf{I}^{\text{syn}}_j)$ and $\mathbf{h}(\mathbf{I}^{\text{obs}})$. The closer the statistics are, the better the satellite is. Another point is that these satellite should be distributed evenly around $\boldsymbol{\beta}$ to obtain good estimation.

5 Experiments

In the experiment, we choose a set of 15 texture images, shown in figure 4, and $K = 12$ filters, including the intensity filter, two gradient filters, three LoG filters and six Gabor filters with a fixed scale but different orientations. We learn the parameter in Ω_{12} by the original stochastic gradient algorithm[14] as ground truth $\boldsymbol{\beta}^*$. For each of the 15 textures, we compute $\boldsymbol{\beta}^*_j$ and a synthesized image $\mathbf{I}^{\text{syn}}_j \sim p(\mathbf{I}; \boldsymbol{\beta}^*_j)$. It is true that the synthesized images can be improved by choosing better filters, but in our experiments we fix a set of filters for computational convenience.

As it is unclear how we measure the distance between $\boldsymbol{\beta}$ and $\boldsymbol{\beta}^*$, we evaluate $\boldsymbol{\beta}$ by sampling from $p(\mathbf{I}; \boldsymbol{\beta})$, and compare the sampled image with the one from $p(\mathbf{I}; \boldsymbol{\beta}^*)$ perceptually.

Experiment I: maximum satellite likelihood. Our first experiment is to evaluate the performance of the maximum satellite likelihood estimators.

Suppose we are learning a texture model chosen from the 15 images. We use five satellites out of the rest 14 learned models $p(\mathbf{I}; \boldsymbol{\beta}^*_j)$. For example, figure 5.a displays a texture image as $\mathbf{I}^{\text{obs}}$. We run

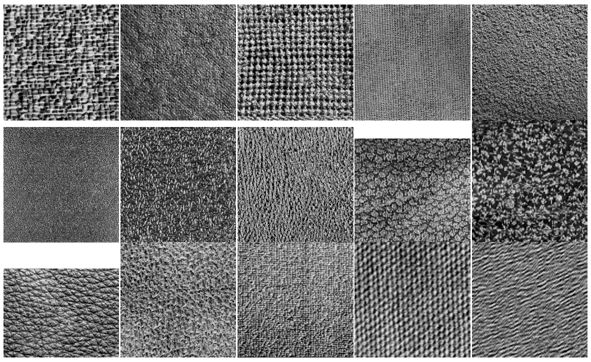
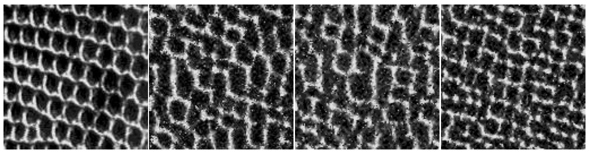


Figure 4: The texture images used in the experiments.

three algorithms for comparison. Figure 6 plots the $\boldsymbol{\beta} = (\beta_1, ..., \beta_{12})$ of the 12 filters learned by the three methods respectively. 1). The dotted curves in Figure 6 are $\boldsymbol{\beta}^*$ learned by the stochastic gradient method, and the synthesized image using $\boldsymbol{\beta}^*$ is shown in figure 5.d. 2). The dashed curves are $\boldsymbol{\beta}$ by the maximum satellite likelihood estimator with samples H_{ij}^{syn} generated from the conditional probabilities $p(\mathbf{I}_{\Lambda_i} | \mathbf{I}^{\text{obs}}_{\partial \Lambda_i}; \boldsymbol{\beta}_j)$. The synthesized image is shown in figure 5.c. 3). The solid curves are $\boldsymbol{\beta}$ by maximum satellite likelihood with samples H_{ij}^{syn} generated from the marginal probabilities $p(\mathbf{I}_{\Lambda_i}; \boldsymbol{\beta}_j)$. The synthesized image is shown in figure 5.b.



a b c d

Figure 5: a). $\mathbf{I}^{\text{obs}}$. b) $\mathbf{I}^{\text{syn}}$ from a satellite algorithm without boundary conditions. c) $\mathbf{I}^{\text{syn}}$ from a satellite algorithm with boundary conditions d) $\mathbf{I}^{\text{syn}}$ from $\boldsymbol{\beta}^*$.

We can see that the learned $\boldsymbol{\beta}$ from the maximum satellite likelihood are quite accurate for this texture pattern. For all the following results using satellite algorithms, we use the marginal probability to fill the holes.

We fix the total number of pixels $\sum_i |\Lambda_i|$ and study the influences of the hole size $m \times m$ on the accuracy of estimating $\boldsymbol{\beta}$. Figure 7.a, b, c show three synthesized images using the $\boldsymbol{\beta}$ learned with three hole sizes $m = 3, 6, 9$ respectively. In comparison, the synthesized image using the maximizing pseudo likelihood

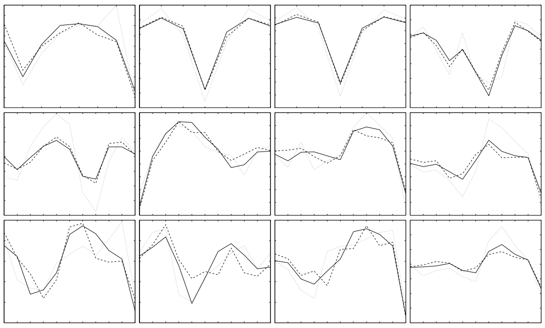
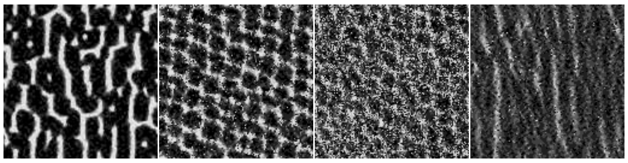


Figure 6: $\boldsymbol{\beta}$ learned using satellite algorithms and FRAME. Solid - Satellite algorithms without boundary conditions. Dashed - Satellite algorithms with synthesized samples of boundary conditions. Dotted - The ground truth - FRAME model.

method is shown in figure 7.d. The observed image is in figure 5.a. Obviously the hole size of 6×6 gives the best result and it takes only about 7 seconds to learn the Gibbs model in a HP workstation (no synthesis).



a b c d

Figure 7: Synthesized images using $\boldsymbol{\beta}$ learned using satellite algorithms with different hole sizes and MPLE. a) Satellite with $m = 3$. b) Satellite with $m = 6$. c) Satellite with $m = 9$. d) MPLE.

Now we discuss how the optimal hole size m is decided.

Figure 2 explains the theory that as m increases, the variance $E_f[(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*)^2]$ decreases, whereas the variance of Monte Carlo integration $E_p[(\hat{Z} - Z)^2]$ increases for a fixed satellite model $\boldsymbol{\beta}_o$. This figure is verified in our experiments, which are shown in figure 8.a. The optimal choice of the hole size m is approximately the intersection of the two curves. Given that the reference models used are close to the dash-dotted line shown in figure 8.a, the predicted optimal hole size is between 5×5 and 6×6. Figure 8.b shows the average filter error of synthesized images $\mathbf{I}^{\text{syn}}$

$$\text{err} = \frac{1}{12} |\mathbf{h}(\mathbf{I}^{\text{obs}}) - \mathbf{h}(\mathbf{I}^{\text{syn}})|$$

using satellite algorithms for different hole sizes. It has minimum error at 6×6.

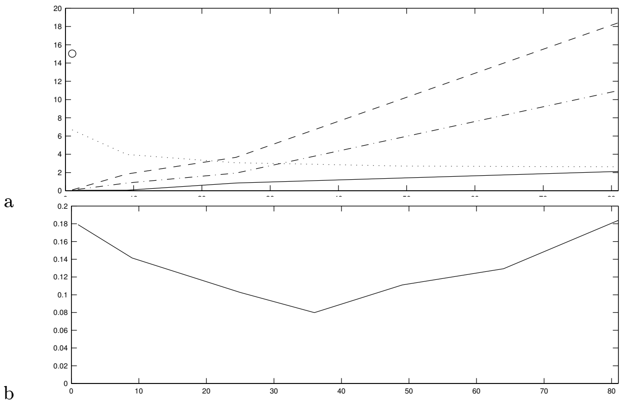


Figure 8: a) The dotted curve is $E_f[(\hat{\beta} - \beta^*)^2]$ The solid, dash-dotted, and dashed curves are for $E_p[(\hat{Z} - Z)^2]$ for three different reference models. b) Average synthesis error per filter err plotted against the hole sizes $|\Lambda_i|$.

We show two more texture synthesis experiments using the maximum satellite likelihood estimator. Figure 9 shows an image with a periodic structure. With different hole sizes, all the satellite algorithms can capture the basic elements and structures among them. With hole size of 3×3, the synthesized image is almost identical to the observed perceptually.

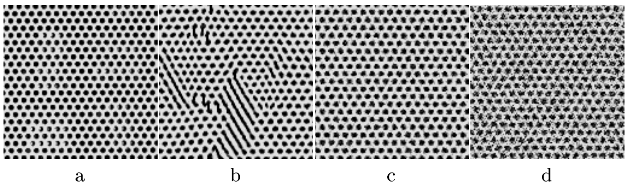


Figure 9: Synthesized images from maximum satellite likelihood: a) $\mathbf{I}^{\text{obs}}$, b) $m = 1$, c) $m = 3$, and d) $m = 5$.

Figure 10 shows a cheetah skin pattern. Even the observed image is not homogeneous, the synthesized images shown in figure 10.c can still capture the perceptual structures and reproduce the major characteristics of the observed images. This result is even comparable with the result shown in [14] using the original algorithm.

Experiment II: maximizing partial likelihood.

The maximizing partial likelihood estimator is similar to the original algorithm used in [14]. However, because the initial boundary condition for the masked pixels is close to the true typical condition, this estimator takes only 1/10th of the number of sweeps of the original algorithm to convergence. In addition, only the foreground pixels need to be synthesized, which can save computation further. Figure 11.c and d show the synthesized images using the learned β using a maximum partial likelihood estimator with only 30 % and 7.5 % pixels in the foreground $\Lambda_i, i = 1, 2, ..., M$. With 30 % foreground pixels, the synthesized result is close to the synthesized image from β^*, and the latter is shown in figure 11.b for comparison. The algorithm is about 25 times faster than the original algorithm[14]. With 7.5 % marked pixels, the synthesized image still captures some structures of the observed image.

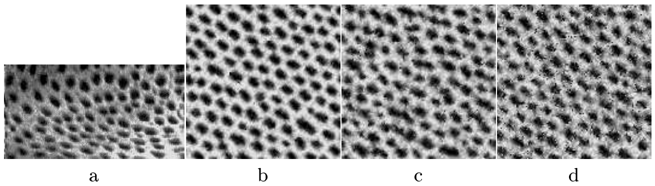


Figure 10: Synthesized images from maximum satellite likelihood: a) $\mathbf{I}^{\text{obs}}$. b) $m = 3$, c) $m = 4$, and d) $m = 5$.

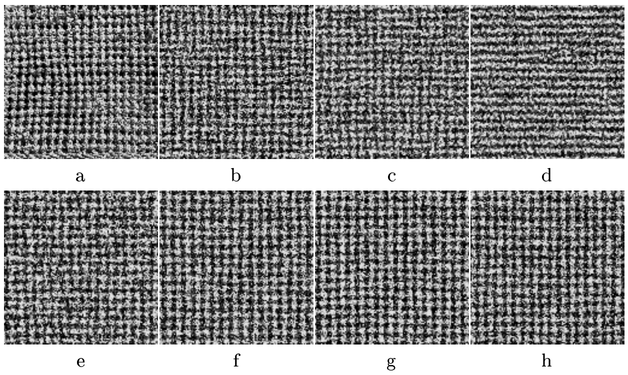


Figure 11: Synthesized images from models estimated by maximizing partial and patch likelihood. a) $\mathbf{I}^{\text{obs}}$. b) Synthesized image using the β^*. c) partial likelihood estimator with 30 % pixels. d) partial likelihood estimator with 7.5 % pixels. e)-h) Synthesized images using the maximizing patch likelihood estimator with 30 % pixels and hole sizes are 1×1, 5×5, 11×11, and 21×21 respectively.

Experiment III: maximizing patch likelihood The maximizing patch likelihood estimator is similar to the maximizing partial likelihood estimator but the boundary condition is always taken from the observed image. With the total number of pixels in the foreground fixed, one can dig many small holes or a few large holes. Figure 11.e-h show the synthesized images with 30 % pixels in the foreground with hole sizes $m = 1, 5, 11, 21$. All the synthesized images capture the structures in the observed image well.

6 Conclusion

There are some four other algorithms proposed for computing the Gibbs models. 1). The minuteman minimax[4]. It is a Newton-Raphson method for estimating β in one step. 2). A variation method[1, 2] which has lower Fisher information than MPLE. 3). A diffusion method[11] synthesizes $\mathbf{I}^{\mathrm{syn}}$ on-line by the stochastic reaction-diffusion equations, and 4). a generalized maximum pseudo likelihood method. Due to space limitation, we don't unfold the discussion, and instead we summarize the ten algorithms in figure 12.

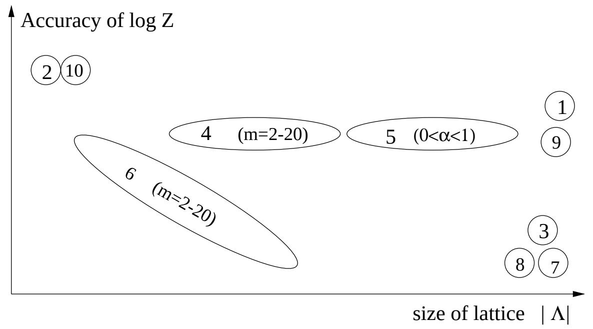


Figure 12: A common framework for learning Gibbs models. The ten algorithms are mapped in two dimensions. 1. Stochastic gradient, 2). maximum pseudo-likelihood, 3). MCMCMLE, 4). maximizing patch likelihood, 5). maximum partial likelihood, 6). maximizing satellite likelihood, 7). minuteman minimax, 8). variational method, 9). learning by diffusion, 10). maximizing generalize pseudo likelihood.

Our conclusion is that if accuracy is the first concern, then the fast algorithm is the maximizing patch likelihood algorithm with $m \sim 10$ or the partial likelihood. If speed is the primary concern, then the maximum satellite likelihood is a good approximation which can be computed in seconds. However, it needs off-line learning.

Acknowledgments

The work is supported by an ARO grant DAAD19-99-1-0012 and an NSF grant IIS-98-77-127. We thank Y. N. Wu for stimulating discussions.

References

[1] M. P. Almeida, B. Gidas, "A variational method for estimating the parameters of MRF from complete and incomplete data", *The Annals of Applied Statistics*, 3:103-136, 1993.

[2] C. H. Anderson and W. D. Langer, "Statistical models of image texture", *Unpublished preprint*, Washington University, St Louis, MO, 1996.

[3] J. Besag, "Efficiency of pseudo-likelihood estimation for simple Gaussian fields, *Biometrika*, **64**: 616-618, 1977.

[4] J. Coughlan, and A. L. Yuille, "Minutemax: a fast approximation for minimax learning", *Proc. of NIPS*, 1998.

[5] R. Chellappa and A. Jain, *Markov random fields: theory and applications*, Academic Press, 1993.

[6] G. R. Cross and A. K. Jain, "Markov random field texture models." *IEEE, PAMI*, **5**, 25-39. 1983.

[7] X. Descombes, R. Morris, J. Zerubia, and M. Berthod, "Maximum likelihood estimation of Markov random field parameters using Markov chain Monte Carlo algorithms", *Int'l Workshop on Energy Minimization Methods for CVPR*, Venice, Italy, May, 1997.

[8] C. J. Geyer and E. A. Thompson, "Annealing Markov chain Monto Carlo with applications to ancestral inference." *J. of Amer. Stat. Assoc.*, **90**, 909-920, 1995.

[9] B. Gidas, "Consistency of maximum likelihood and pseudo-likelihood estimators for Gibbs distributions", in W. Fleming and P.L. Lions (eds.) *Stochastic Differential Systems, Stochastic Control Theory and Applications*, Springer, NY, 1988.

[10] H. Robbins and S. Monro, "A stochastic approximation method." *Annals of Mathematical Statistics*, 22, 400-407, 1951.

[11] J. Shah, "Minimax entropy and learning by diffusion", *Proc. of CVPR*, CA. 1998.

[12] L. Younes, "Estimation and annealing for Gibbsian fields", *Annales de l'Institut Henri Poincare, Section B, Calcul des Probabilities et Statistique*, **24**, 269-294, 1988.

[13] Y. N. Wu, S. C. Zhu, and X. W. Liu, "Equivalence of Gibbs and Julesz ensembles", *Proc. of ICCV*, 1999.

[14] S. C. Zhu, Y. Wu, and D. Mumford. "Minimax Entropy Principle and Its Application to Texture Modeling". Neural Computation, Vol. 9, no 8, Nov.1997.

[15] S. C. Zhu, X. Liu, and Y. Wu, "Exploring Julesz Ensembles by Efficient Markov Chain Monte Carlo", *IEEE Trans. on PAMI* To appear.