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

Song Chun Zhu and Xiuwen Liu

Abstract—Gibbsian fields or Markov random fields are widely used in Bayesian image analysis, but learning Gibbs models is computationally expensive. The computational complexity is pronounced by the recent minimax entropy (FRAME) models which use large neighborhoods and hundreds of parameters [22]. In this paper, we present a common framework for learning Gibbs models. We identify two key factors that determine the accuracy and speed of learning Gibbs models: The efficiency of likelihood functions and the variance in approximating partition functions using Monte Carlo integration. We propose three new algorithms. In particular, we are interested in a *maximum satellite likelihood estimator*, which makes use of a set of precomputed Gibbs models called "satellites" to approximate likelihood functions. This algorithm can approximately estimate the minimax entropy model for textures in seconds in a HP workstation. The performances of various learning algorithms are compared in our experiments.

Index Terms—Markov random fields, minimax entropy learning, texture modeling, Markov chain Monte Carlo, maximum-likelihood estimate, importance sampling.

1 Introduction

GIBBSIAN fields or Markov random fields (MRF) are widely used in Bayesian image analysis for characterizing stochastic patterns or prior knowledge, but learning accurate Gibbs models is computationally expensive. In the literature, many learning algorithms are focused on auto-models, such as auto-binomial models [7] and Gaussian Markov random fields [5] whose parameters can be computed analytically. For MRF models beyond the auto-model families, there are three algorithms. The first is a *maximum pseudolikelihood estimator* (MPLE) [4]. The second is a *stochastic gradient algorithm* [18], [21], [22]. The third is *Markov chain Monte Carlo maximum-likelihood estimator* (MCMCMLE) [12], [13], [9]. The second and third methods approximate partition functions by importance sampling techniques—a topic studied extensively in the literature [15], [16], [17], [8]. There are also methods [10], [11] that estimate Gibbs parameters with precomputed derivatives of log-partition functions. These algorithms were used primarily for learning MRF models with pair cliques, such as Ising models and Potts models. The computational complexity is pronounced by the recent minimax entropy models (FRAME) for texture [22] which involve large neighborhoods (e.g., filters with window sizes of 33×33 pixels) and large number of parameters (e.g., $100 - 200$).

Recently, four algorithms were proposed to speedup minimax entropy learning with mixed success:

1. A minuteman minimax algorithm [6],
2. A variational method [2], [1],
3. A method of learning by diffusion [19], and
4. A generalized MPLE (Private communication with Y.N. Wu).

Besides the new algorithms, an ensemble equivalence theorem enables the separation of the model selection procedure from the parameter learning step [20].

- S.C. Zhu is with the Departments of Statistics and Computer Science, University of California at Los Angeles, Los Angeles, CA 90095. E-mail: sczhu@stat.ucla.edu.
- X.W. Liu is with the Department of Computer Science, Florida State University, Tallahassee, FL 32306. E-mail: liux@cs.fsu.edu.

Manuscript received 1 Feb. 2000; revised 26 Feb. 2001; accepted 29 Nov. 2001. Recommended for acceptance by C. Bouman.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number 111610.

In this paper, we study a common statistical framework for learning the parameters of Gibbs models with an emphasis on computational efficiency. There are two factors that determine the accuracy and speed of these learning algorithms. The first is the efficiency of the formulated likelihood functions measured by the Fisher's information. The second is the approximation of partition functions by importance sampling. This analysis leads to three new learning algorithms:

1. A maximum partial likelihood estimator,
2. A maximum patch likelihood estimator, and
3. A maximum satellite likelihood estimator.

Our emphasis will be on the third algorithm which approximates partition functions by a set of precomputed reference Gibbs models in a similar spirit to [10], [11]. This algorithm can approximately compute Gibbs models in about 10 seconds on a HP workstation.

The paper is organized as follows: Section 2 presents a common framework for Gibbs learning. Section 3 presents three new algorithms. Section 4 demonstrates experiments on texture synthesis. We conclude with a discussion in Section 5.

2 Learning in Gibbsian Fields—A Common Framework

Let $\mathbf{I}_\Lambda$ be an image defined on a lattice Λ and $\mathbf{I}_{\partial\Lambda}$ its boundary conditions. $\partial\Lambda$ is the neighborhood of Λ. Let $\mathbf{h}(\mathbf{I}_\Lambda|\mathbf{I}_{\partial\Lambda})$ be the feature statistics of $\mathbf{I}_\Lambda$ under boundary conditions $\mathbf{I}_{\partial\Lambda}$. For example, $\mathbf{h}()$ is a vector for the histograms of filtered images [23]. Without loss of generality, a Gibbs model is of the following form (see [22]),

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

In (1), β is a vector valued parameter corresponding to the Julesz ensemble on infinite images [20] and it is invariant to the sizes and shapes of Λ. So, β can be estimated on an arbitrary Λ.

In learning Gibbs models, we are given an observed image $\mathbf{I}_\Lambda^{\text{obs}}$, where Λ may have many disconnected components to account for multiple observations. Equally, we may break Λ into smaller patches $\mathbf{I}_{\Lambda_i}^{\text{obs}}, i = 1,2,\ldots,M$ on lattices Λ_i of arbitrary shapes and sizes. These patches may overlap with each other. Then, β is learned by maximizing a log-likelihood,

$$\beta^* = \arg\max_\beta \mathcal{G}(\beta), \quad \text{with } \mathcal{G}(\beta) = \sum_{i=1}^{M} \log p\Big(\mathbf{I}_{\Lambda_i}^{\text{obs}}|\mathbf{I}_{\partial\Lambda_i}^{\text{obs}};\beta\Big). \quad (2)$$

We show that existing Gibbs learning algorithms are unified as ML-estimators and they differ in the following two choices.

Choice 1. The number, sizes, and shapes of the *foreground* patches $\Lambda_i, i = 1,\ldots,M$.

Fig. 1 displays four typical choices for Λ_i. The bright pixels are in the *foreground* $\Lambda_i, i = 1,2,\ldots,M$, which are surrounded by dark pixels in the *background* $\partial\Lambda_i, i = 1,2,\ldots,M$. In the first three cases, Λ_i are square patches with $m \times m$ pixels. In one extreme, Fig. 1a chooses one largest patch denoted by Λ_1, i.e., $M = 1$ and $m = N - 2w$ with w being the width of the boundary. $\mathcal{G}(\beta)$ is called the *log-likelihood*, and it is adopted by the stochastic gradient [21], [22] and MCMCMLE [12], [13], [9] methods. In the other extreme, Fig. 1c chooses the minimum patch size $m = 1$ and $\mathcal{G}(\beta)$ is called the *log-pseudolikelihood*, used in the maximum pseudolikelihood estimation (MPLE) [4]. Fig. 1b is an example between the two extremes and $\mathcal{G}(\beta)$ is called the *log-patch-likelihood*. In the fourth case, Fig. 1d chooses only one ($M = 1$) irregular-shaped patch, denoted by Λ_1, where Λ_1 is a set of randomly selected pixels with the rest of the pixels being the background $\partial\Lambda_1$, and $\mathcal{G}(\beta)$ is called the *log-partial-likelihood*. In Figs. 1b and 1c, a foreground pixel may serve as background in

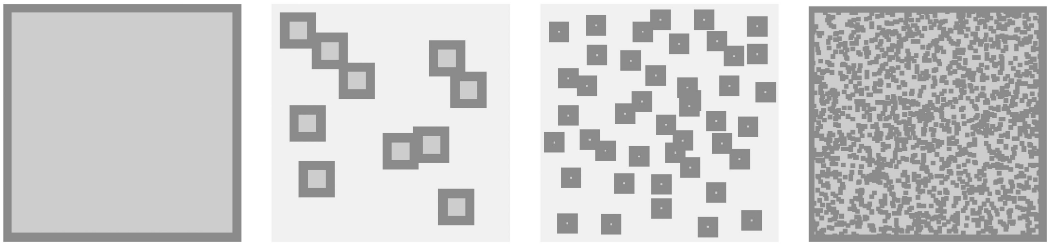

Fig. 1. Various choices of Λ_i, $i = 1, 2, \ldots, M$. The bright pixels are in foreground Λ_i which are surrounded by dark background pixels in $\partial\Lambda_i$. (a) Likelihood, (b) patch likelihood (or satellite likelihood), (c) pseudolikelihood, and (d) partial likelihood.

different patches. It is straightforward to prove that maximizing $\mathcal{G}(\beta)$ leads to a consistent estimator for all four choices [14].

The flexibility of likelihood function distinguishes Gibbs learning from the problem of estimating partition functions [15], [16], [17]. The latter computes the "pressure" on a large lattice in order to overcome boundary effects.

Choice 2. The reference models used for estimating the partition functions. For a chosen foreground and log-likelihood function, the second step is to approximate the partition functions $Z(\mathbf{I}^{\rm obs}_{\partial\Lambda_i}, \beta)$ for each $\Lambda_i, i = 1, \ldots, M$ by Monte Carlo integration using a reference model at β_o.

$$
\begin{aligned}
&Z\Big(\mathbf{I}^{\rm obs}_{\partial\Lambda_i}, \beta\Big)\\
&= \int \exp\Big\{- < \beta, \mathbf{h}\Big(\mathbf{I}_{\Lambda_i} | \mathbf{I}^{\rm obs}_{\partial\Lambda_i}\Big) > \Big\} d\mathbf{I}_{\Lambda_i},\\
&= Z(\mathbf{I}_{\partial\Lambda_i}, \beta_o) \int \exp\Big\{- < \beta - \beta_o, \mathbf{h}\Big(\mathbf{I}_{\Lambda_i} | \mathbf{I}^{\rm obs}_{\partial\Lambda_i}\Big) > \Big\} dp\Big(\mathbf{I} | \mathbf{I}^{\rm obs}_{\partial\Lambda_i}; \beta_o\Big)\\
&\approx \frac{Z\Big(\mathbf{I}^{\rm obs}_{\partial\Lambda_i}, \beta_o\Big)}{L} \sum_{j=1}^{L} \exp\Big\{- < \beta - \beta_o, \mathbf{h}\Big(\mathbf{I}^{\rm syn}_{ij} | \mathbf{I}^{\rm obs}_{\partial\Lambda_i}\Big) > \Big\}.
\end{aligned} \tag{3}
$$

$\mathbf{I}^{\rm syn}_{ij}, j = 1, 2, \ldots, L$ are *typical* samples from the reference model $p(\mathbf{I}_{\Lambda_i} | \mathbf{I}^{\rm obs}_{\partial\Lambda_i}; \beta_o)$ for each patch $i = 1, 2, \ldots, M$.

Since $\frac{Z(\mathbf{I}^{\rm obs}_{\partial\Lambda_i}, \beta_o)}{L}, i = 1, 2, \ldots M$ are independent of β, we can maximize the estimated log-likelihood $\mathcal{G}(\beta)$ by gradient descent. This leads to

$$
\frac{d\beta}{dt} = \sum_{i=1}^{M} \left\{ \sum_{j=1}^{L} \omega_{ij} \mathbf{h}\Big(\mathbf{I}^{\rm syn}_{ij} | \mathbf{I}^{\rm obs}_{\partial\Lambda_i}\Big) - \mathbf{h}\Big(\mathbf{I}^{\rm obs}_{\Lambda_i} | \mathbf{I}^{\rm obs}_{\partial\Lambda_i}\Big) \right\}. \tag{4}
$$

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

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

The selection of the reference models $p(\mathbf{I}_{\Lambda_i} | \mathbf{I}^{\rm obs}_{\partial\Lambda_i}; \beta_o)$ depends on the sizes of the patches $\Lambda_i, i = 1, \ldots, M$. In general, importance sampling is only valid when the two distributions $p(\mathbf{I}_{\Lambda_i} | \mathbf{I}^{\rm obs}_{\partial\Lambda_i}; \beta_o)$ and $p(\mathbf{I}_{\Lambda_i} | \mathbf{I}^{\rm obs}_{\partial\Lambda_i}; \beta)$ heavily overlap. In one extreme case $m = 1$, the MPLE method [4] selects $\beta_o = 0$ and $p(\mathbf{I}_{\Lambda_i} | \mathbf{I}^{\rm obs}_{\partial\Lambda_i}; \beta_o)$ a uniform distribution. In this case, $Z(\mathbf{I}^{\rm obs}_{\partial\Lambda_i}, \beta)$ can be computed exactly. In the other extreme case for a large foreground $m = N - 2w$, the stochastic gradient and the MCMCMLE methods have to choose $\beta_o = \beta$ in order to obtain sensible approximations. Thus, both methods must sample $p(\mathbf{I}; \beta)$ iteratively starting from $\beta_0 = 0$. This is the algorithm adopted in learning the FRAME models [22].

To summarize, Fig. 2 illustrates two factors that determine the accuracy and speed of learning β. These curves are verified through experiments in Section 4 (see Fig. 7). The horizontal axis is the size of an individual foreground lattice $|\Lambda_i|$.

1. *The variances of MLE or inverse Fisher information*. Let $\hat{\beta}(\mathbf{I}^{\rm obs})$ be the estimator maximizing $\mathcal{G}(\beta)$ and let β^* be the optimal solution. The dashed curve in Fig. 2 illustrates the variance
$$E_f\left[\left(\hat{\beta}(\mathbf{I}^{\rm obs}) - \beta^*\right)^2\right],$$
where $f(\mathbf{I})$ is a underlying distribution representing the Julesz ensemble. For choices shown in Fig. 1, if we fix the total number of foreground pixels $\sum_{i=1}^{M} |\Lambda_i|$, then the variance (or estimation error) decreases as the patch size (diameter of the hole) increases.
2. *The variance of estimating Z by Monte Carlo integration* $E_p[(\hat{Z} - Z)^2]$. For a given reference model $\beta_o = \beta_i, i = 1, 2, \ldots, k$ (see solid curves in Fig. 2), this estimation error increases with the lattice sizes. Therefore, for very large patches, such as $m = 200$, we must construct a sequence of reference models to approach β, $\beta_0 = 0 \rightarrow \beta_1 \rightarrow \beta_2 \rightarrow \ldots, \rightarrow \beta_k \rightarrow \beta$. This is the major reason why the stochastic gradient algorithm was so slow in FRAME [22].

3 Three New Algorithms

The analysis in the previous section leads to three new algorithms by selecting likelihoods that trade-off between the two factors and the third algorithm improves accuracy by precomputed reference models.

Algorithm 1: Maximizing partial likelihood. We choose a lattice shown in Fig. 1d by choosing at random a certain

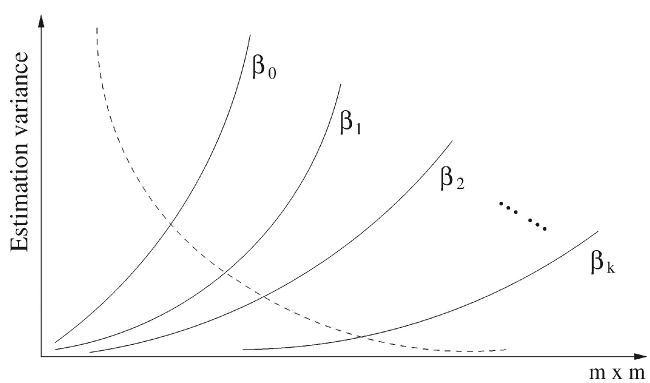

Fig. 2. Estimation variances for various selections of patch sizes $m \times m$ and reference models β_o. The dashed curve shows the inverse Fisher's information which decreases as $m \times m$ increases. The solid curves show the variances in the importance sampling for a sequence of models approaching β.

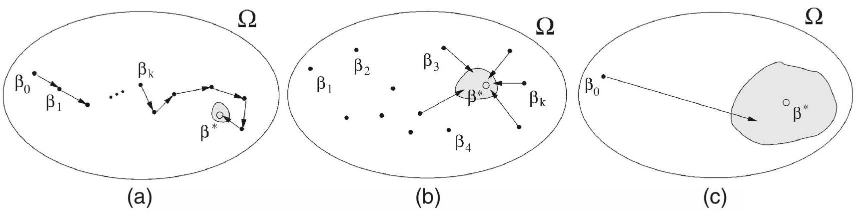

Fig. 3. The shadow areas around β^* illustrate the variance of the estimated β or efficiency of the log-likelihood functions. (a) Stochastic gradient and Algorithms 1 and 2 generate a sequence of satellites online to approach β closely, m can be small or large. (b) The maximum satellite likelihood estimator uses a general set of satellites computed offline and can be updated incrementally. This can be used for small size m. (c) MPLE uses a single satellite: $\beta_o = 0$.

percentage (say, 30 percent) of pixels as foreground Λ_1 and the rest are treated as background Λ/Λ_1.

We define a *log-partial-likelihood*

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

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

$$\frac{d\boldsymbol{\beta}}{dt} = E_{p\left(\mathbf{I}_{\Lambda_1} | \mathbf{I}^{\text{obs}}_{\Lambda/\Lambda_1}; \boldsymbol{\beta}\right)} \Big[\mathbf{h}\Big(\mathbf{I}_{\Lambda_1} | \mathbf{I}^{\text{obs}}_{\Lambda/\Lambda_1}\Big)\Big] - \mathbf{h}\Big(\mathbf{I}^{\text{obs}}_{\Lambda_1} | \mathbf{I}^{\text{obs}}_{\Lambda/\Lambda_1}\Big). \tag{5}$$

This algorithm follows the same procedure as the original method in FRAME [22]. It trades off between accuracy and speed in a better way than the original algorithm in FRAME [22]. The log-partial-likelihood has lower Fisher information than the log-likelihood; however, our experiments demonstrate that it is about 25 times faster than the original minimax learning method without losing much accuracy. We observed that the reason for this speedup is that the original sampling method [22] spends a major portion of its time synthesizing $\mathbf{I}^{\text{syn}}_{\Lambda_1}$ under "nontypical" boundary conditions starting with white noise images. In contrast, the new algorithm works on typical boundary condition $\mathbf{I}^{\text{obs}}_{\Lambda/\Lambda_1}$ where the probability mass of the Gibbs model $p(\mathbf{I}; \boldsymbol{\beta})$ is focused on. The speed appears to be decided by the diameter of the foreground lattice measured by the maximum circle that can fit in the foreground lattice.

Algorithm 2. Maximizing patch likelihood. Algorithm 2 chooses a set of M overlapping patches from $\mathbf{I}^{\text{obs}}_{\Lambda}$ and "digs" a hole Λ_i on each patch, as Fig. 1b shows. Thus, we define a *patch log-likelihood*

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

Maximizing $\mathcal{G}_2(\boldsymbol{\beta})$ by gradient descent, we update $\boldsymbol{\beta}$ iteratively as Algorithm 1 does.

$$\frac{d\boldsymbol{\beta}}{dt} = \sum_{i=1}^{M} \mathbf{h}\Big(\mathbf{I}^{\text{syn}}_{\Lambda_i} | \mathbf{I}^{\text{obs}}_{\Lambda/\Lambda_i}\Big) - \sum_{i=1}^{M} \mathbf{h}\Big(\mathbf{I}^{\text{obs}}_{\Lambda_i} | \mathbf{I}^{\text{obs}}_{\Lambda/\Lambda_i}\Big). \tag{6}$$

In comparison with Algorithm 1, the diameters of the lattices are evenly controlled. Algorithm 1 has similar performance as Algorithm 1.

Algorithm 3. Maximizing satellite likelihood. Both Algorithms 1 and 2 still need to synthesize images online, which is a computationally intensive task. Now, we propose an third algorithm which may approximately compute $\boldsymbol{\beta}$ in the speed of a few seconds without synthesizing images online.

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

$$\mathcal{R} = \big\{p(\mathbf{I}; \boldsymbol{\beta}_j) \ : \ \boldsymbol{\beta}_j \in \Omega, j = 1, 2, \ldots, s.\big\}$$

We sample (or synthesize) one large typical image $\mathbf{I}^{\text{syn}}_j \sim p(\mathbf{I}; \boldsymbol{\beta}_j)$ for each reference model offline. These reference models estimate $\boldsymbol{\beta}$ in Ω from different "viewing angles." By analogy to the global positioning system, we call the reference models the "satellites."

The *log-satellite-likelihood* is defined as

$$\mathcal{G}_3(\boldsymbol{\beta}) = \mathcal{G}^{(1)}_3(\boldsymbol{\beta}; \boldsymbol{\beta}_1) + \mathcal{G}^{(2)}_3(\boldsymbol{\beta}; \boldsymbol{\beta}_2) + \cdots + \mathcal{G}^{(s)}_3(\boldsymbol{\beta}; \boldsymbol{\beta}_s), \tag{7}$$

where each satellite contributes one log-likelihood approximation,

$$\mathcal{G}^{(j)}_3(\boldsymbol{\beta}; \boldsymbol{\beta}_j) = \sum_{i=1}^{M} \log \frac{1}{\hat{Z}^{(j)}_i} \exp\Big\{- < \boldsymbol{\beta}, \mathbf{h}\Big(\mathbf{I}^{\text{obs}}_{\Lambda_i} | \mathbf{I}^{\text{obs}}_{\partial\Lambda_i}\Big) >\Big\}. \tag{8}$$

Following the importance sampling method in (3), we estimate $Z(\mathbf{I}^{\text{obs}}_{\partial\Lambda_i}, \boldsymbol{\beta})$ by Monte Carlo integration.

$$\hat{Z}^{(j)}_i = \frac{Z(\mathbf{I}^{\text{obs}}_{\partial\Lambda_i}, \boldsymbol{\beta}_j)}{L} \sum_{\ell=1}^{L} \exp\Big\{- < \boldsymbol{\beta} - \boldsymbol{\beta}_j, \mathbf{h}\Big(\mathbf{I}^{\text{syn}}_{ij\ell} | \mathbf{I}^{\text{obs}}_{\partial\Lambda_i}\Big) >\Big\}. \tag{9}$$

Notice that, for every hole Λ_i and for every 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:

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

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 and has to be computed online.
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 synthesized image $\mathbf{I}^{\text{syn}}_j$ computed offline.

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

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

$$\frac{d\boldsymbol{\beta}}{dt} = \sum_{j=1}^{s} \left\{ \sum_{i=1}^{M} \left[\sum_{\ell=1}^{L} \omega_{ij} \mathbf{h}\Big(\mathbf{I}^{\text{syn}}_{ij\ell} | \mathbf{I}^{\text{obs}}_{\partial\Lambda_i}\Big) - \mathbf{h}\Big(\mathbf{I}^{\text{obs}}_{\Lambda_i} | \mathbf{I}^{\text{obs}}_{\partial\Lambda_i}\Big) \right] \right\} \tag{10}$$

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

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

Equation (10) converges in the speed of seconds for an average texture model.

However, we should be aware of the risk that the log-satellite-likelihood $\mathcal{G}_3(\boldsymbol{\beta})$ may not be upper bounded. It is almost surely not upper bounded for the MCMCMLE method. This case occurs when $\mathbf{h}(\mathbf{I}^{\text{obs}}_{\Lambda_i} | \mathbf{I}^{\text{obs}}_{\partial\Lambda_i})$ cannot be described by a linear combination of the statistics of the sampled patches $\sum_{\ell=1}^{L} \omega_{ij} \mathbf{h}(\mathbf{I}^{\text{syn}}_{ij\ell} | \mathbf{I}^{\text{obs}}_{\partial\Lambda_i})$. When this occurs, $\boldsymbol{\beta}$ does not converge. 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, let $\mathbf{I}^{\text{syn}}_{ij1} = \mathbf{I}^{\text{obs}}_{\Lambda_i}$, $\boldsymbol{\beta}$ is learned so that the conditional probabilities $\omega_{ij1} \to 1$ and $\omega_{ij\ell} \to 0$, $\forall \ell \neq 1$.

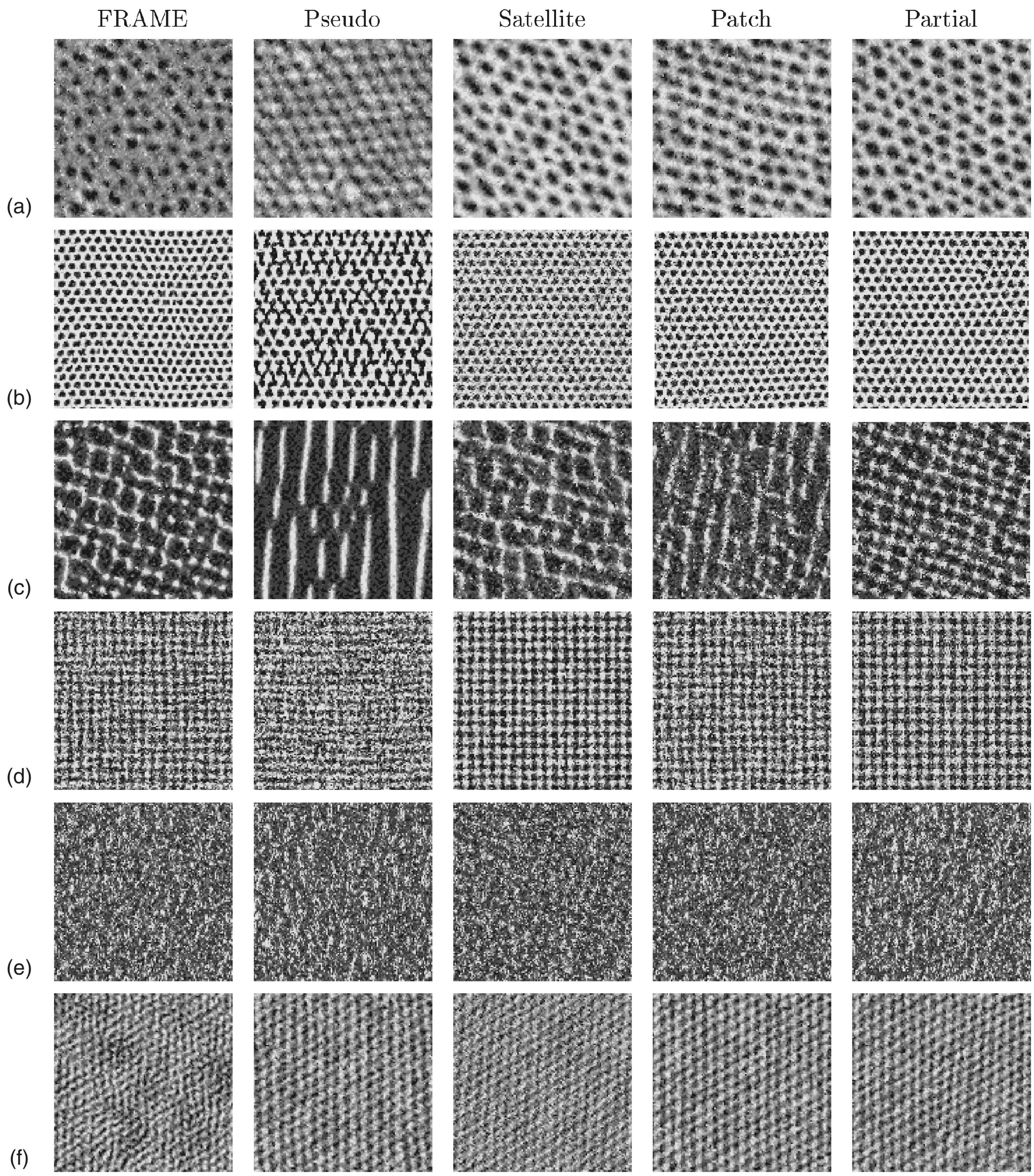

Fig. 4. Synthesized texture images using β learned from various algorithms. For each column from left to right: 1: stochastic gradient algorithm as the ground truth, 2: pseudolikelihood, 3: satellite likelihood, 4: patch likelihood, 5: partial likelihood.

Since ℓ is often very large, say $\ell = 20$, adding one extra sample will not disort the sample set.

To summarize, we compare existing algorithms and the newly proposed algorithms from the perspective of estimating β^* in Ω, and divide them into three groups. Fig. 3 illustrates the comparison where the ellipse stands for the space Ω and each Gibbs model is represented by a single point.

Group 1. As Fig. 3a illustrates, the maximum likelihood estimators (including stochastic gradient and MCMCMLE) and the maximum partial/patch likelihood estimators generate and sample a sequence of "satellites" $\beta_0, \beta_1, \ldots, \beta_k$ online. These satellites get closer and closer to β^* (supposed truth). The shadow area around β^* represents the uncertainty in computing β, whose size can be measured by the Fisher information.

Group 2. As Fig. 3c shows, the maximum pseudolikelihood estimator uses a uniform model $\beta_o = 0$ as a "satellite" to estimate any model and, thus, has large variance.

Group 3. The maximum satellite likelihood estimators in Fig. 3b use a general set of satellites which are precomputed and sampled offline. To save time, one may select a small subset of satellites for computing a given model. One can choose satellites based on the differences $\mathbf{h}(\mathbf{I}_j^{\text{syn}})$ and $\mathbf{h}(\mathbf{I}^{\text{obs}})$. The smaller the differences are, the closer the satellite is to the estimated model and, thus, better approximation. Another criterion is that these satellite should be distributed evenly around β^* to obtain good estimation.

4 Experiments

In this section, we evaluate the performance of various algorithms in the context of learning Gibbs models for textures. We use 12 filters including an intensity filter, two gradient filters, three Laplacian of Gaussian filters, and six Gabor filters at a fixed scale and different orientations. Thus, $\mathbf{h}(\mathbf{I})$ includes 12 histograms of filter responses and each histogram has 12 bins. So, β has 12×11 free parameters.

We choose 15 natural texture images. For each texture, we use the stochastic gradient algorithm [22] to learn β which is treated as ground truth β^* for comparison. In this way, we also obtained 15 satellites with 15 synthesized images $\mathbf{I}_j^{\text{syn}}$ computed offline.

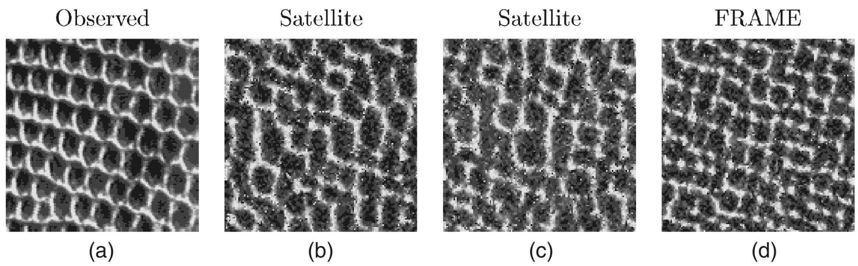

Fig. 5. Performance evaluation of the satellite algorithm. (a) Observed texture image. (b) Synthesized image using β learned without boundary conditions. (c) Synthesized image using β learned with boundary conditions. (d) Synthesized image using β learned by stochastic gradient.

Experiment 1. Comparison of five algorithms. In the first experiment, we compare the performance of five algorithms in texture synthesis. Fig. 4 demonstrates 6 texture patterns of 128×128 pixels. For each row, the first column is the synthesized image (ground truth) using a stochastic gradient method used in the FRAME model [22], the other four images are, respectively, synthesized images using maximum pseudolikelihood, maximum satellite likelihood, maximum patch likelihood, and maximum partial likelihood. For the last three algorithms, we fixed the total number of foreground pixels to $5,000$. The patch size is fixed to 5×5 pixels for patch likelihoods and satellite likelihoods. We select 5 satellites out of the rest of the 14 precomputed models for each texture.

Since for different textures the model $p(\mathbf{I};\boldsymbol{\beta})$ may be more sensitive to some elements of $\boldsymbol{\beta}$ (such as tail bins) than to the rest of the parameters and the β vectors are highly correlated between its components, it is not very meaningful to compare the accuracy of the learned β using an error measure $|\boldsymbol{\beta} - \boldsymbol{\beta}^*|$. Instead, we sample each learned model $\mathbf{I}^{\rm syn} \sim p(\mathbf{I};\boldsymbol{\beta})$ and compare the histogram errors of the synthesized image against the observed, i.e., $|\mathbf{h}(\mathbf{I}^{\rm syn}) - \mathbf{h}(\mathbf{I}^{\rm obs})|$, summed over 12 pairs of histograms each being normalized to 1. The table below shows the errors for each algorithms for the synthesized images in Fig. 4. The numbers are subject to some computational fluctuations including the sampling process.

The experimental results show that the four algorithms work reasonably well. In comparison, the satellite method is often close to the patch and partial likelihood methods. Though it sometimes may yield slightly better results than other methods depending on the similarity between the reference models and the model to be learned. The pseudolikelihood method can also capture some large image features. In particular, it works well for textures of stochastic nature. For example, on the three textures in Figs. 4d, 4e, and 4f.

In terms of computational complexity, the satellite algorithm is the fastest, and it computes β in the order of 10 seconds on a HP-workstation. The second fastest is the pseudolikelihood. It takes a few minutes. However, the pseudolikelihood method consumes a large amount of memory, as it needs to remember all the k histograms for the g gray levels in $N \times N$ pixels. The space complexity is $O(N^2 \times g \times k \times B)$ with B being the number of bins. It often needs more than one Gigabyte of memory. The partial likelihood and patch likelihood are very similar to the stochastic gradient algorithm [22]. Since the initial boundary condition is typical, these two estimators take only, in general, 1/10th of the number of sweeps to convergence. In addition, only a portion of pixels need to be synthesized, which can save further computation. The computation time is only about 1/20th of the stochastic gradient algorithm.

Experiment 2. Analysis of the maximum satellite likelihood estimator. In the second experiment, we study how the performance of the satellite algorithm is influenced by 1) the boundary condition, and 2) the size of patch $m \times m$.

1. Influence of boundary conditions. Fig. 5a displays a texture image as $\mathbf{I}^{\rm obs}$. We run three algorithms for comparison. Fig. 5d is a result from the FRAME (stochastic gradient method). Figs. 5b and 5c are results using the satellite algorithms. The difference is that Fig. 5c uses observed boundary condition for each patch and does online sampling, while Fig. 5b ignores the boundary condition. For all the following results of satellite likelihood method (Algorithm 3), $H_{ij}^{\rm syn}$ are generated from the marginal probabilities without online sampling.
2. Influences of the hole size $m \times m$. We fix the total number of foreground pixels $\sum_i |\Lambda_i|$ and study the performance of satellite algorithm with difference hole sizes m. Figs. 6a, 6b, and 6c show three synthesized images using β learned by satellite algorithm with different hole sizes $m = 2, 6, 9$, respectively. It is clear from Figs. 6a, 6b, and 6c that the hole size with 6×6 pixels gives better result.

To explain why the hole size of $m = 6$ gives better satellite approximation, we compute the two key factors that determine performance. Fig. 7a shows the numeric results in correspondence to the theoretical analysis displayed in Fig. 2.

When the hole size is small, the partition function can be estimated accurately as shown by the small $E_p[(\hat{Z} - Z)^2]$ in solid, dash-dotted, and dashed curves in Fig. 7. However, the variance $E_f[(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*)^2]$ is large for small holes, which is shown by the dotted curve in Fig. 7a. The optimal choice of the hole size thus is approximately the intersection of the two curves. Since the reference models that we used are close to the dash-dotted line shown in Fig. 7a, we predict optimal hole size is between 5×5 and 6×6. Fig. 7b shows the average error between the statistics of synthesized image $\mathbf{I}^{\rm syn} \sim p(\mathbf{I};\boldsymbol{\beta})$ and the observed statistics ${\rm err} = \frac{1}{12}|\mathbf{h}(\mathbf{I}^{\rm obs}) - \mathbf{h}(\mathbf{I}^{\rm syn})|$, where β is learned using the satellite method for $m = 1, 2, \ldots, 9$. Here, the hole size of 6×6 pixels gives better result.

5 Conclusion

To conclude our study, we qualitatively compare 10 Gibbs learning algorithms in Fig. 8 along three factors (or dimensions):

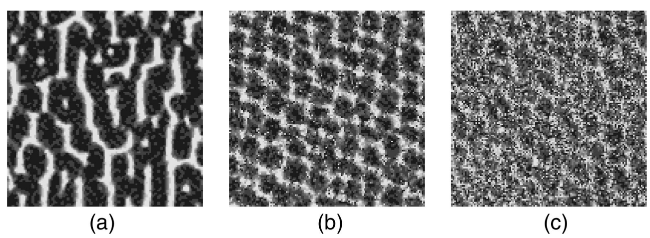

Fig. 6. Synthesized images using β learned by the satellite method with different hole sizes. (a) $m = 2$. (b) $m = 6$. (c) $m = 9$.

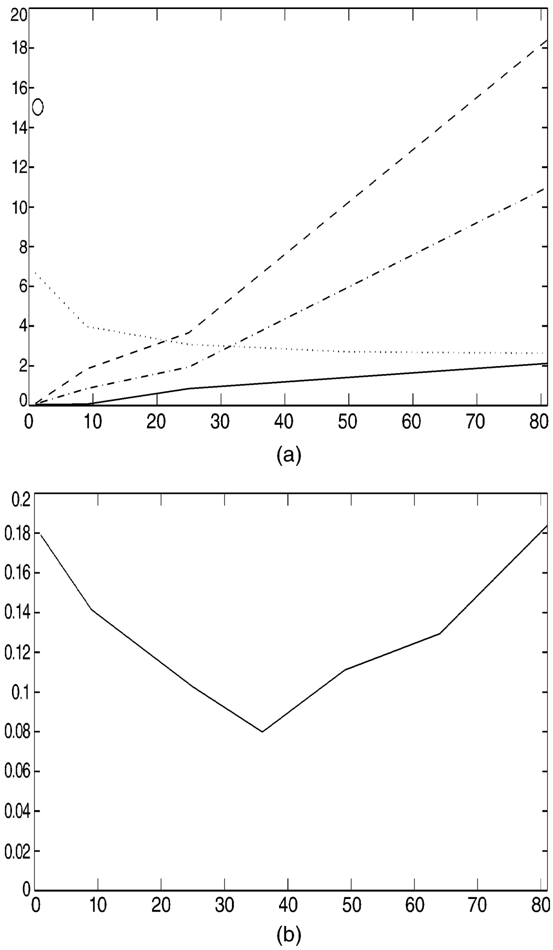

Fig. 7. The x-axes are the hole size m^2. (a) Dotted curve is $E_f[(\hat{\beta} - \beta^*)^2]$ plotted against the hole size m^2. The solid, dash-dotted, and dashed curves are $E_p[(\hat{Z} - Z)^2]$ for three different reference models. (b) Average synthesis error per filter with respect to the hole size m^2.

1. Accuracy in approximating $\log Z$.
2. The diameter of foreground lattices and, thus, efficiency of the likelihood.
3. Computational complexity.

The 10 algorithms are:

1. Stochastic gradient MLE [22],
2. Maximum pseudolikelihood (MPLE) [3], [4],
3. MCMCMLE [12], [13], [9], [16],
4. Maximum patch likelihood,
5. Maximum partial likelihood,
6. Maximum satellite likelihood,
7. Minuteman minimax [6],
8. Variational method [2], [1],
9. Learning by diffusion [21],
10. Generalized maximum pseudolikelihood (Y.N. Wu, private communication).

ACKNOWLEDGMENTS

The project was supported by a US National Science Foundation grant NSF-9877127 and a US National Science Foundation Career award IIS-00-92664. S.C. Zhu would like to thank Yingnian Wu for valuable discussions. An earlier version of this paper appear in the Proceedings of Computer Vision and Pattern Recognition, 2000.

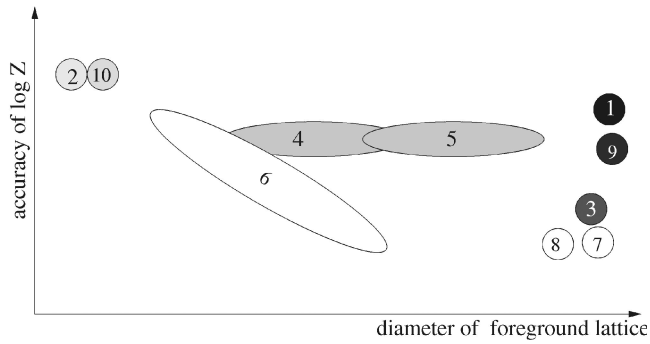

Fig. 8. A common framework for learning Gibbs models. The horizontal axis is the size of foreground patches which is proportional to Fisher's information. The vertical axis is the accuracy in estimating $\log Z$. The brightness of the ellipses implies the learning speed, and darker is slower. This figure is intended only for a qualitative comparison.

REFERENCES

[1] M.P. Almeida and B. Gidas, "A Variational Method for Estimating the Parameters of MRF from Complete and Incomplete Data," *The Annals of Applied Statistics,* vol. 3, pp. 103-136, 1993.
[2] C.H. Anderson and W.D. Langer, "Statistical Models of Image Texture," Unpublished preprint, Washington Univ., St Louis, Mo., 1996.
[3] J. Besag, "Spatial Interaction and the Statistical Analysis of Lattice Systems (with discussion)," *J. Royal Statistical Soc. B,* vol. 36, pp. 192-236, 1973.
[4] J. Besag, "Efficiency of Pseudo-Likelihood Estimation for Simple Gaussian Fields, " *Biometrika,* vol. 64, pp. 616-618, 1977.
[5] R. Chellappa and A.K. Jain, *Markov Random Fields: Theory and Applications.* Academic Press, 1993.
[6] J. Coughlan and A.L. Yuille, "Minutemax: A Fast Approximation for Minimax Learning," *Proc. Neural Information Processing Systems,* 1998.
[7] G.R. Cross and A.K. Jain, "Markov Random Field Texture Models," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 5, pp. 25-39, 1983.
[8] H. Derin and H. Elliott, "Modeling and Segmentation of Noisy and Textured Images Using Gibbs Random Fields," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 9, no. 1, pp. 39-55, 1987.
[9] X. Descombes, R. Morris, J. Zerubia, and M. Berthod, "Maximum Likelihood Estimation of Markov Random Field Parameters Using Markov Chain Monte Carlo Algorithms," *Proc. Int'l Conf. Energy Minimization Methods in Computer Vision and Pattern Recognition,* May 1997.
[10] S. Geman and D. McClure, "Bayesian Images Analysis: An Application to Single Photon Emission Tomography," *Proc. Statistical Computer Section Am. Statistical Assoc.,* pp. 12-18, 1985.
[11] S. Geman and D. McClure, "Statistical Methods for Tomographic Image Reconstruction," *Bull. Int'l Statistical Inst.,* vol. LII-4, pp. 5-21, 1987.
[12] C.J. Geyer and E.A. Thompson, "Constrained Monte Carlo Maximum Likelihood for Dependent Data," *J. Royal Statistical Soc. B,* vol. 54, pp. 657-699, 1992.
[13] C.J. Geyer, "On the Convergence of Monte Carlo Maximum Likelihood Calculations," *J. Royal Statistical Soc. B,* vol. 56, pp. 261-274, 1994.
[14] B. Gidas, "Consistency of Maximum Likelihood and Pseudo-Likelihood Estimators for Gibbs Distributions," *Stochastic Differential Systems, Stochastic Control Theory and Applications,* W. Fleming and P.L. Lions, eds., New York: Springer, 1988.
[15] M. Jerrum and A. Sinclair, "Polynomial-Time Approximation Algorithms for the Ising Model," *SIAM J. Computing,* vol. 22, pp. 1087-1116, 1993.
[16] G.G. Potamianos and J.K. Goutsias, "Partition Function Estimation of Gibbs Random Field Images Using Monte Carlo Simulations," *IEEE Trans. Information Theory,* vol. 39, pp. 1322-1332, 1993.
[17] G.G. Potamianos and J. Goutsias, "Stochastic Approximation Algorithms for Partition Function Estimation of Gibbs Random Fields," *IEEE Trans. Information Theory,* vol. 43, pp. 1948-1965, 1997.
[18] H. Robbins and S. Monro, "A Stochastic Approximation Method," *Annals Math. Statistics,* vol. 22, pp. 400-407, 1951.
[19] J. Shah, "Minimax Entropy and Learning by Diffusion," *Proc. Computer Vision and Pattern Recognition,* 1998.
[20] Y.N. Wu, S.C. Zhu, and X.W. Liu, "Equivalence of Gibbs and Julesz Ensembles," *Proc. Int'l Conf. Computer Vision,* 1999.
[21] L. Younes, "Estimation and Annealing for Gibbsian Fields," *Annales de l'Institut Henri Poincare, Section B, Calcul des Probabilities et Statistique,* vol. 24, pp. 269-294, 1988.
[22] S.C. Zhu, Y.N. Wu, and D.B. Mumford, "Minimax Entropy Principle and Its Application to Texture Modeling," *Neural Computation,* vol. 9, no. 8, Nov. 1997.
[23] S.C. Zhu, X.W. Liu, and Y.N. Wu, "Exploring Julesz Ensembles by Efficient Markov Chain Monte Carlo," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 22, no. 6, pp. 554-569, June 2000.