Learning in Markov Random Fields
An Empirical Study

Sridevi Parise, Max Welling
University of California, Irvine
sparise,welling@ics.uci.edu

Abstract

Learning the parameters of an undirected graphical model is particularly difficult due to the presence of a global normalization constant. For large unstructured models computing the gradient of the log-likelihood is intractable and approximations become necessary. Several approximate learning algorithms have been proposed in the literature but a thorough comparative study seems to be absent. In this paper we report on the results of a series of experiments which compare a number of learning algorithms on several models. In our experimental design we use perfect sampling techniques in order to be able to assess quantities such as (asymptotic) normality, bias and variance of the estimates. We envision this effort as a first step towards a more comprehensive open source testing environment where researchers can submit learning algorithms and benchmark problems.

Keywords: Markov Random Fields, Learning, Maximum Likelihood

1. Introduction

In this paper we report on experimental evaluation of learning algorithms for Markov random field models. We propose to evaluate these algorithms by comparing bias and variance of the obtained estimates. Importantly, we insist that the data was generated by a model in the parameterized family of models under consideration. This to make sure that unbiased learning algorithms, will return unbiased estimates, at least asymptotically. As a result, we are restricted to models for which we can generate perfect samples. In light of modern techniques to generate these perfect samples this condition is less restrictive than might be thought at first. In particular, we look at MRFs with arbitrary connectivity but with attractive interactions and at rectangular grid models with arbitrary interactions. The main reason we have chosen not to evaluate on indirect performance measures such as classification performance is that there are a number of confounding factors that are hard to disentangle from the real question that we want to answer: how well can a learning algorithm identify the true value of parameters from data?

Our software is publicly online[1] and we hope that by soliciting other researchers to submit their code, data-sets and learning algorithms this may represent the first step towards a more centralized and objective manner to evaluate learning algorithms in MRF models.

2. Experimental Design

In this section we describe the experimental setup for the experiments we report on in section 3..

Given a probabilistic model $\mathcal{M}$ that depends on a number of parameters $\boldsymbol{\theta}$ we provide M data-sets of N data cases each. We have restricted ourselves to models for which we can produce perfect samples, i.e. samples that are guaranteed to come from the distribution under consideration. The reason we insist on this property is that it is notoriously difficult to assess convergence of a Markov chain. In the absence of such guarantees it becomes impossible to disentangle bias produced by the approximate learning algorithm and from the fact that the samples are not from the distribution we are trying to infer.

Given an algorithm $\mathcal{A}$, we estimate the parameters of the model under consideration on each of the M data-sets of size N producing $\{\hat{\boldsymbol{\theta}}_1, \hat{\boldsymbol{\theta}}_2, ..., \hat{\boldsymbol{\theta}}_M\}$. Next, we determine the bias and the variance of these estimates according to,

$$\text{bias}(\hat{\boldsymbol{\theta}}) = |\boldsymbol{\theta} - \mathbb{E}(\hat{\boldsymbol{\theta}})| \ , \ \mathbb{E}(\hat{\boldsymbol{\theta}}) = \frac{1}{M}\sum_{m=1}^{M} \hat{\boldsymbol{\theta}}_m \quad (1)$$

$$\text{var}(\hat{\boldsymbol{\theta}}) = \frac{1}{M}\sum_{m=1}^{M} \Big(\hat{\boldsymbol{\theta}}_m - \mathbb{E}(\hat{\boldsymbol{\theta}})\Big)^2 \quad (2)$$

Estimation can often be interpreted as a trade-off between bias and variance. The total mean square error is an easy function of bias and variance $\text{MSE} = \text{bias}(\hat{\boldsymbol{\theta}})^2 + \text{var}(\hat{\boldsymbol{\theta}})$. To assess whether the bias is statistically significant we first subject the estimates

[1] URL:http://www.ics.uci.edu/∼sparise/software

$\{\hat{\boldsymbol{\theta}}_m\}$ to a hypothesis test to determine if they follow a normal distribution. We have used the following univariate normality tests: Bera-Jarque and Lilliefors tests at significance levels $\alpha = 0.05$ (both in MATLAB).

Once we have established normality, we test if the bias for each parameter is statistically significant using a standard univariate one-sample t-test at a significance level of $\alpha = 0.05$. In our experiments we report bias, variance (both averaged over all parameters) and the fraction of bias estimates that were tested significant.

2.1 Models

In all our experiments we work with binary valued ($\{0,1\}$, pairwise MRF models also known as Boltzmann machines. Due to the constraints imposed by the requirement to obtain perfect samples we looked at two architectures in particular: $K \times L$ square grids which are either fully observed or have unobserved nodes interspersed with observed nodes in such a way that the neighbors of observed nodes are all unobserved nodes and vice versa. We also looked at models with only non-negative weights but arbitrary biases[2]. With this constraint we allow arbitrary architectures if all variables are observed while in the presence of unobserved nodes we restrict to bipartite architectures, i.e. the neighbors of unobserved nodes must be observed and vice versa.

The above fully observed (FO) models and bipartite partly observed (PO) models follow the following PDFs,

$$P^{\mathrm{FO}}(\mathbf{x}) = \frac{e^{\sum_{i>j} W_{ij} x_i x_j + \sum_i \alpha_i x_i}}{Z(W, \boldsymbol{\alpha})} \quad (3)$$

$$P^{\mathrm{PO}}(\mathbf{x}, \mathbf{h}) = \frac{e^{\sum_{i,j} J_{ij} x_i h_j + \sum_i \alpha_i x_i + \sum_j \beta_j h_j}}{Z(W, \boldsymbol{\alpha}, \boldsymbol{\beta})} \quad (4)$$

where we note that W_{ij} and J_{ij} are sometimes constrained to be non-negative.

2.2 Perfect Sampling Techniques

We used the following procedures to generate perfect samples for the models discussed in the previous section.

For grids we first build a junction tree (JT). In this case it turns out to be a chain where the n^{th} 'super-node' in the JT chain contains $(R+1)$ consecutive nodes starting from node n (assuming nodes are numbered column-wise and R is the number of rows). To generate perfect samples we extended the "forward-filtering-backward-sampling" algorithm described in Scott (2002) for HMMs to the JT constructed for the square grid models. To generate perfect samples for the MRFs with non-negative interactions we use "coupling-from-the-past" as described in Propp and Wilson (1996).

[2]Note that the term bias has two meanings in this paper: the interaction at a single node and the statistical bias in estimating parameters.

2.3 Learning Algorithms

For all learning algorithms we follow the gradient or the approximate gradient of the log-likelihood of the data. The exact gradients of the weights for the distributions above are given by,

$$\nabla_{W_{ij}} \ell^{\mathrm{FO}} = \mathbb{E}[x_i x_j]_{\hat{P}(x_i x_j)} - \mathbb{E}[x_i x_j]_{P(x_i, x_j)} \quad (5)$$

$$\nabla_{J_{ij}} \ell^{\mathrm{PO}} = \mathbb{E}[x_i h_j]_{\hat{P}(x_i) P(h_j|\mathcal{D})} - \mathbb{E}[x_i h_j]_{P(x_i h_j)} \quad (6)$$

where $\mathcal{D}$ denotes the data, $\hat{P}(\cdot)$ is the empirical distribution of the data and $P(\cdot)$ is the model distribution. Similar expressions hold for $\boldsymbol{\alpha}, \boldsymbol{\beta}$. Learning by following these derivatives will be denoted by "ML-exact". We use JTs to perform exact inference for the second term.

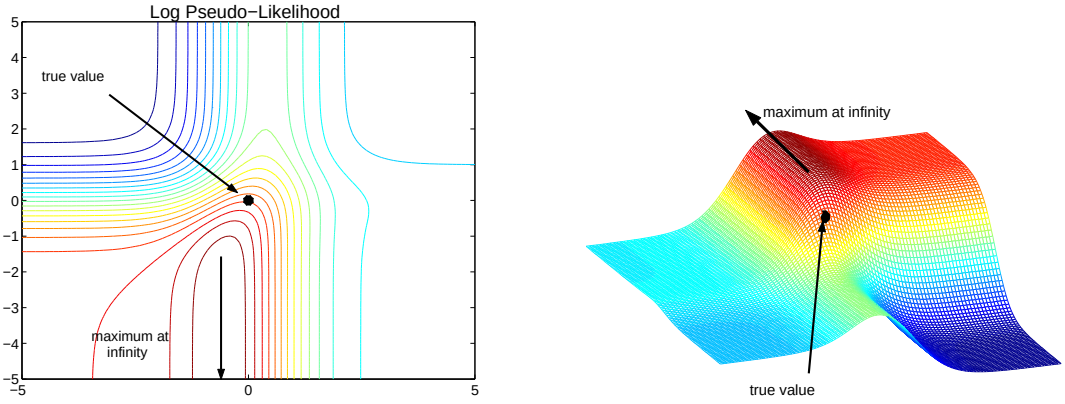
While computing the first term of these derivatives is straightforward, it is the second term that often needs to be approximated. One option is to approximate it by some Markov chain Monte Carlo procedure. This turned out to be too computationally taxing for the experiments we conducted.

A standard method in the statistics literature, Besag (1977), is the "pseudo-likelihood" estimate (PL). Here we condition each observed variable on its (observed) neighbors and average the result. This results in the following approximation for the second term in Eqns.5,

$$\mathbb{E}[x_i x_j]_{P(x_i, x_j)} \approx \frac{1}{2} \mathbb{E}[x_i \sigma(W\mathbf{x} + \boldsymbol{\alpha})_j + \sigma(W\mathbf{x} + \boldsymbol{\alpha})_i x_j]_{\hat{P}(\mathbf{x})} \quad (7)$$

where we note that $\sigma(W\mathbf{x} + \boldsymbol{\alpha})_i = \sigma(\sum_{k \in \mathcal{N}_j} W_{ik} x_k + \alpha_i)$ with $\sigma(\cdot)$ a sigmoid function and $\mathcal{N}_j$ the neighbors of node i. The PL estimate is known to be a consistent estimator, but is statistically less efficient than the ML. We have also tried to maximize the PL objective function for the partially observed problems, i.e. we marginalize out the unobserved variables from the conditional distributions. However, as illustrated in figures 1 this often led to infinite values of certain parameters.

Another approach we tested is contrastive divergence (CD) described in Hinton (2002). In this approach we use N Markov chain Monte Carlo sam-



(a) (b)

Figure 1: Log-pseudo-likelihood surface for bias terms $\beta_{1,2}$ for partially observed bipartite model (as in Eqn.4) with two observed and two unobserved variables. We set $\alpha_{1,2} = \beta_{1,2} = 0$, sample J_{ij} from a Gaussian with std.=0.5 and use 1000 data-cases. In most cases (as in the figure) the optimal estimate is located at infinity.

plers initialized at each data-case in the data-set[3]. These Markov chains are run for k steps after which we record the samples $\{s_n\}$. In the experiments for the FO case we used Gibbs sampling where one step updates a total of V variables (where V is the number of variables in the problem) chosen at random with replacement. For the PO case we used the 2-phase Gibbs sampler proposed in Hinton (2002). The negative terms in the gradients are now approximated by the "empirical" distribution of these k-step samples

$$\begin{aligned} \mathbb{E}[x_i x_j]_{P(x_i,x_j)} &\approx \mathbb{E}[x_i x_j]_{\hat{P}(s_i,s_j)} \\ &= \frac{1}{N}\sum_n s_{i,n}s_{j,n} \qquad (8) \\ \mathbb{E}[x_i h_j]_{P(x_i h_j)} &\approx \mathbb{E}[x_i h_j]_{\hat{P}(s_i)P(h_j|\mathcal{S})} \\ &= \frac{1}{N}\sum_n s_{i,n}\sigma(J^T \mathbf{s}_n + \beta)_j \qquad (9) \end{aligned}$$

where $\mathcal{S}$ is the set of k-step samples.

A popular approach to approximating marginals over single nodes and neighboring nodes is loopy belief propagation (see Pearl (1988)). One can use these marginals to approximate the required expectations in the learning rules (5) and (6). We call this "BP learning".

Finally, we tested "pseudo-moment matching" (PMM) as proposed in Wainwright, Jaakkola, and Willsky (2003). The idea is to choose the parameters such that if we run loopy belief propagation on these parameters we obtain the observed frequency counts on edges and nodes. For binary random variables the equations were derived in Welling and Teh (2003),

$$W_{ij} = \log\left(\frac{\xi_{ij}(\xi_{ij}+1-p_i-p_j)}{(p_i-\xi_{ij})(p_j-\xi_{ij})}\right) \qquad (10)$$

$$\alpha_i = \log\left(\frac{(1-p_i)^{|\mathcal{N}_i|-1}\prod_{j\in\mathcal{N}_i}(p_i-\xi_{ij})}{p_i^{|\mathcal{N}_i|-1}\prod_{j\in\mathcal{N}_i}(\xi_{ij}+1-p_i-p_j)}\right) \qquad (11)$$

where $p_i = \hat{P}(x_i = 1)$ and $\xi_{ij} = \hat{P}(x_i = 1, x_j = 1)$ are given by empirical estimates (i.e. frequency tables). $\mathcal{N}_i$ denotes the neighbors of node i. As it stands, this method is only defined for fully observed models. Note that PMM learning is equivalent to BP learning for FO case. Hence we tested BP learning only for the PO case.

3. Experiments and Results

We performed a series of experiments to analyze absolute and relative performance of the various methods described in section 2.. In each experiment we evaluated a single method or a subset of methods by varying some "independent property" such as: (1) N, the number of samples in each dataset, (2) d, a measure of the interaction strengths between nodes or (3) s, a measure of the graph connectivity (or sparsity).

Given a model family (with specified structure), we first sample the parameters (see below) and then given those, we generate the dataset using perfect sampling. Unless stated otherwise, we generate weights W and J from $\mathcal{U}[-d,d]$ and biases $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ from $\mathcal{U}[-d/2,d/2]$ for the unconstrained case

[3] For computational efficiency one often performs updates on mini-batches.

Table 1: Summary of Experiments

Experiment	Model	Alg. evaluated	property varied	other settings	figures
FO-grid-N	FO grid	ML-exact,PL,CD,PMM	N	grid size:$7X10$, d=0.1	figure (2)
FO-grid-d	FO grid	ML-exact,PL,CD,PMM	d	grid size:$7X10$, N=20000	figure (3)
FO-FC-s	FO with arbitrary connectivity, +ve interactions	PL,CD,PMM	s	#nodes=49, s starts with $7X7$ grid,N=10∗# model parameters,d=0.1	figure (4)
FO-grid-N-real	FO grid using parameters learnt from real data	ML-exact,PL,CD,PMM	N	grid size:$6X6$	figure (5)
FO-FC-s-real	FO arbitrary connectivity, +ve interactions	PL,CD,PMM	s	#nodes=36,s starts with $6X6$ grid, N=10∗#parameters	figure (6)
PO-FC-d	PO,bipartite,all hidden-visible edges, +ve interactions	CD	d	#hidden nodes=#visible nodes=10,N=12000	figures (7),(9)
PO-grid-d	PO bipartite grid	CD	d	grid size:$7X10$, N=20000	figures (7),(9)
PO-FC-N	PO,bipartite,all hidden-visible edges, +ve interactions	CD	N	#hidden nodes=#visible nodes=10, d=0.1	figures (8),(9)
PO-grid-N	PO bipartite grid	CD	N	grid size:$7X10$, d=0.1	figures (8),(9)

and correspondingly from $\mathcal{U}[0, d]$ and $\mathcal{U}[-d/2, d/2]$ for the models with non-negative weight constraint. Here $d > 0$ is some constant that acts as a measure of the interaction strengths. We also performed experiments where instead of using the uniform distribution, we use parameters learnt from real-world data (NIST handwritten digits)

In the experiments where we vary the underlying graph structure (holding the number of nodes fixed), we use s to denote the fraction of all possible edges that are present in the model. Hence, a low s indicates sparse graphs. To vary s we start with a grid connectivity and randomly add edges until full connectivity.

Given the various learning algorithms, model families, parameter distributions and "independent properties", there are a plethora of possible experiments that can be done. Here we report only on a subset of these. In some cases this choice was dictated by factors such as applicability of a method, practical convergence times, avoiding possible degeneracies etc.

The experiments performed are summarized in table 1 and results are shown in figures (2) to (9). In all experiments, $M = 100$ datasets were used to compute performance measures. For experiments *FO-grid-N-real* and *FO-FC-s-real*, parameters were first learnt using PL on a real-world dataset and then M datasets were generated in the usual manner using these parameters. For both FC nets and grids the weights ended up between $[0, 3]$ while the biases were all negative between $[-6, 0]$. The data used were $6X6$ cropped binary images of handwritten digits (NIST dataset).

In the fully observed cases, datasets were checked to ensure that there are no empty cells, that is, all possible node and edge combinations occur at least once in the data. This avoids potential degeneracies in the likelihood surface and also ensures that the PMM equations (10) and (11) are well defined.

The log-likelihood for fully observed problems is convex and all parameters are identifiable, hence we initialized all parameters at small random values. On the other hand, the log-likelihood in the PO case has many local minima and the parameters are only identifiable up to a permutation of the unobserved variables. This led us to initialize the parameters at their true value.

4. Discussion

The experimental results lead us to the following conclusions.

– On the FO problems both PL and CD perform at the same level as ML-exact and very few parameters seem significantly biased. In general we would therefore recommend PL for FO problems because PL is faster than CD.

– For FO problems PMM was only accurate when

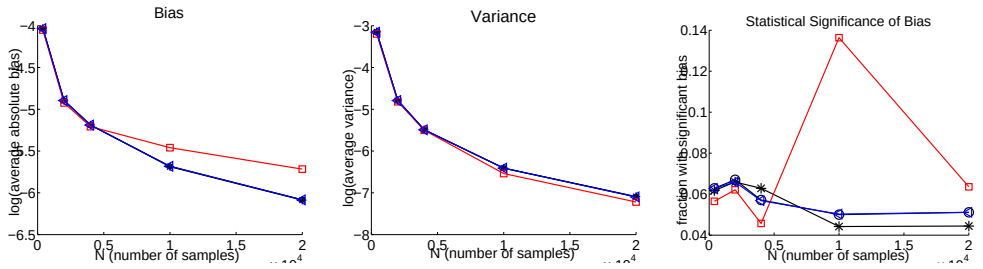


Figure 2: *FO-grid-N* Experiment (∘,∗ , □, ◁ indicate ML-exact, PL, CD (K=5), PMM respectively)

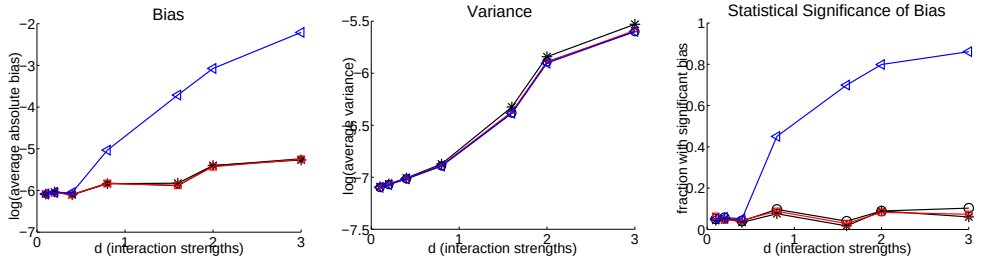


Figure 3: *FO-grid-d* Experiment (∘,∗ , □, ◁ indicate ML-exact, PL, CD (K=5), PMM respectively)

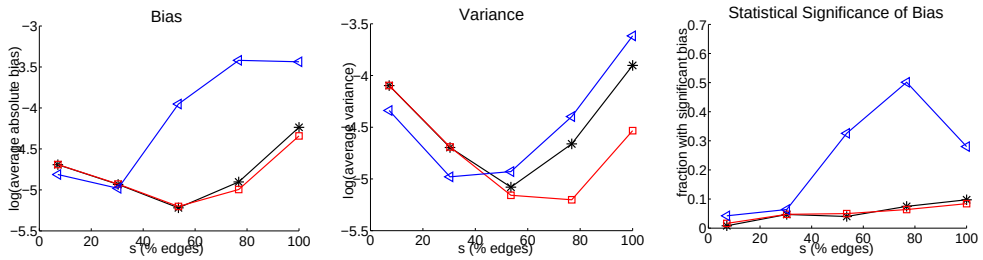


Figure 4: *FO-FC-s* Experiment (∗ , □, ◁ indicate PL, CD (K=5), PMM respectively)

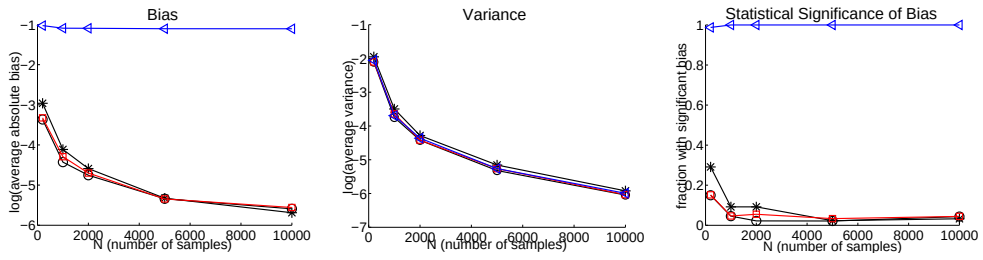


Figure 5: *FO-grid-N-real* Experiment (∘,∗ , □, ◁ indicate ML-exact, PL, CD (K=5), PMM respectively)

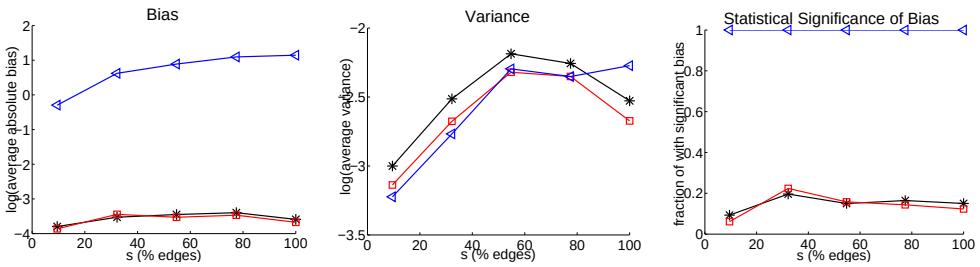


Figure 6: *FO-FC-s-real* Experiment ($*$, $\square$, $\triangleleft$ indicate PL, CD (K=5), PMM respectively)

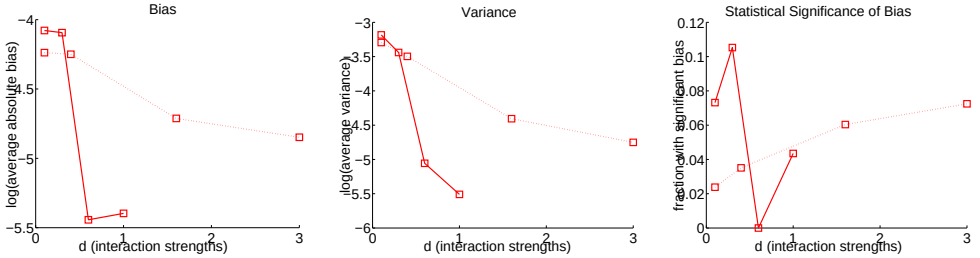


Figure 7: *PO-FC-d and PO-grid-d* Experiments ($\square$ indicates CD (K=5). The solid line is for PO-FC-d and the dotted line is for PO-grid-d)

the interactions are weak and the graph is not densely connected. It is very fast, but we would only recommend it as an initialization procedure to obtain a rough first guess.

– For PO problems we found that PL could not be made to work satisfactorily because often parameters would run off to infinity. This phenomenon was graphically checked for a simple 2×2 bipartite graph (see figure 1). Initial experiments with BP on PO cases had similar problems with parameter convergence. Also, PMM is not defined for unobserved variables which left us with no real competition for CD. Note that running Gibbs sampling to convergence was not feasible because it was too computationally taxing on the problems we looked at.

– CD seems to work well for PO bipartite problems. It is interesting to note that unlike the FO case, performance improved when the interaction strengths *increased*. This is probably due to the fact that the biases on the unobserved variables are easier to identify with strong interactions. Also observe that for the PO-grid-d experiment the variance decreased more than the bias resulting in more parameters being statistically significantly biased. Our results on CD are consistent with those obtained on much smaller problems in Carreira-Perpinan and Hinton (2005).

– For FO problems, 80% or more of the parameters passed both normality tests in all experiments. For PO models however the numbers are significantly lower as can be seen from figure 9.

– For all problems we considered, estimation accuracy improves when we increase N and decrease d or s. This is consistent with common wisdom in the field.

In future experiments we plan to look more closely into the problems with learning in PO models. More challenging is a comparative study to train MRF models with large connected subgraphs of unobserved variables. In this case, even CD needs to run a Markov chain to convergence and becomes prohibitively slow. Using mean field dynamics on the unobserved variables was proposed as a possible solution in Welling and Hinton (2001).

References

Besag, J. (1977), "Efficiency of pseudo-likelihood estimation for simple Gaussian fields," *Biometrika*, 64, 616–618.

Carreira-Perpinan, M. and Hinton, G. (2005), "On Contrastive Divergence Learning," in *Tenth International Workshop on Artificial Intelligence and Statistics*, Barbados.

Hinton, G. (2002), "Training products of experts

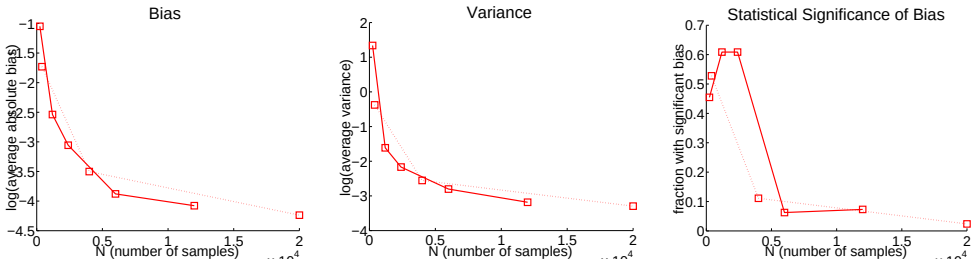
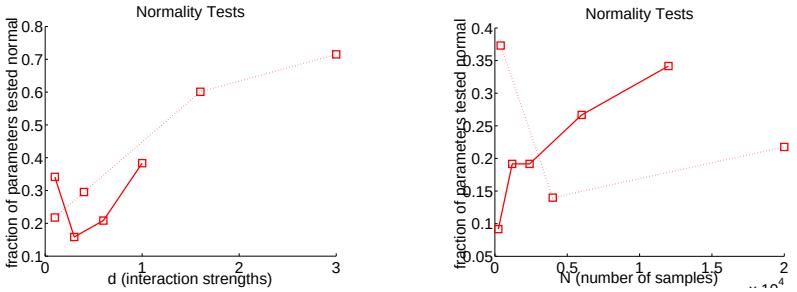


Figure 8: *PO-FC-N and PO-grid-N* Experiments (□ indicates CD (K=5). The solid line is for PO-FC-N and dotted line is for PO-grid-N)



(a) (b)

Figure 9: Normality Tests for PO models. (a) Solid line is for PO-FC-d and dotted line is for PO-grid-d. (b) Solid line is for PO-FC-N and dotted line is for PO-grid-N.

by minimizing contrastive divergence," *Neural Computation*, 14, 1771–1800.

Pearl, J. (1988), *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, San Mateo, California: Morgan Kaufmann Publishers.

Propp, J. G. and Wilson, D. B. (1996), "Exact sampling with coupled Markov chains and applications to statistical mechanics," in *Random Structures and Algorithms*, volume 9, pp. 223–252.

Scott, S. L. (2002), "Bayesian Methods for Hidden Markov Models, Recursive Computing in the 21st Century," in *Journal of the American Statistical Association*, volume 97, pp. 337–351.

Wainwright, M., Jaakkola, T., and Willsky, A. (2003), "Tree-reweighted belief propagation algorithms and approximate ML estimation via pseudo-moment matching," in *AISTATS*.

Welling, M. and Hinton, G. (2001), "A new learning algorithm for mean field Boltzmann machines," in *Proc. of the Int'l Conf. on Artificial Neural Networks*, Madrid, Spain.

Welling, M. and Teh, Y. (2003), "Approximate Inference in Boltzmann Machines," *Artificial Intelligence*, 143, 19–50.