AD-A247 291

CMU-CS-84-120

1

Variations on the Boltzmann Machine Learning Algorithm

Mark Derthick
Department of Computer Science
Carnegie-Mellon University
Pittsburgh, PA 15213

August 1984

DTIC
S ELECTE D
MAR 0 3 1992
D

DEPARTMENT
of
COMPUTER SCIENCE

This document has been approved for public release and sale; its distribution is unlimited.

92 2 28 090

Carnegie-Mellon University

92-05242

CMU-CS-84-120

Variations on the Boltzmann Machine Learning Algorithm

DTIC
COPY
INSPECTED
6

Accesion For
NTIS CRA&I ✓
DTIC TAB
Unannounced
Justification
By per ltr
Distribution/
Availability Codes
Dist Avail and/or Special
A-1

Mark Derthick
Department of Computer Science
Carnegie-Mellon University
Pittsburgh, PA 15213

August 1984

Abstract

Boltzmann Machines learn to model the structure of an environment by modifying internal weights. The algorithm used for changing a weight depends on collecting statistics about the behavior of the two units that the weight connects. The success and speed of the algorithm depends on the accuracy of the statistics, the size of the weight changes, and the way in which the accuracy of the machine's model varies as the weights are changed. This paper presents theoretical analysis and empirical results that can be used to select more effective parameters for the learning algorithm.

Copyright © 1984 Mark Derthick

This research was sponsored in part by a Graduate Fellowship from the Office of Naval Research, and in part by a grant from the System Development Foundation.

6

Table of Contents

1. Introduction

This paper assumes familiarity with Boltzmann Machines, and the learning algorithm described in (Hinton et al, 1984). As discussed there, the learning parameters affect the accuracy of the estimate of the gradient of the cost function, G, at a point, and how that estimate is used to select the next point to be investigated.

The first section of this report discusses G in general terms and presents some intuitions about its topography. This is followed by two sections devoted to empirical analysis of the results of varying parameters of the learning algorithm in an effort to speed up the process. Sections 5 and 6 discuss specific problems encountered, and the conclusion speculates on modifications to the cost function which may further improve the learning.

2. General Results about G-space

2.1. Description of G-space

The Boltzmann Machine learns to make its model of its environment correspond to the actual environment in which it is placed. An environment is a probability distribution of patterns over a subset of the units called the visible units. The environment can be specified explicitly as a list of ordered pairs, giving a pattern, V_α, over the visible units, and a probability for the occurrence of the pattern, $P(V_\alpha)$. The machine's model is just the probability distribution it would produce over the visible units if it were allowed to run freely without any environmental input. This probability distribution is not stored directly. Instead, it is specified implicitly by the magnitude of the weights in the machine. For a machine architecture with v visible units, there are 2^v possible patterns. A machine whose weights were all zero would implicitly specify an environment where each of these patterns had probability 2^{-v}, since the energy of all states would be the same, namely 0. In all the environments we have investigated, the number of states that occur is very small, on the order of v rather than 2^v.[1]

Sampling the probability distribution implicit in the machine's connections is accomplished by simulated annealing in Energy Space. The points in this space correspond to the 2^{v+h} discrete states of the $v+h$ units; the value of E at each point is determined by the weights, which remain fixed during the annealing process.

The learning procedure attempts to change the weights to minimize the distance between the two probability distributions, as measured by the function

$$G=\sum_\alpha P(V_\alpha)\log\frac{P(V_\alpha)}{P'(V_\alpha)}$$

[1] see section 4.1 for an example of an environment.

where $P(V_\alpha)$ is the environmental probability for state V_α, and $P'(V_\alpha)$ is the probability derived from the machine's model. G is a function of the W weights, and lies on a W dimensional surface within the $W+1$ dimensional space we call G-space. Optimization is done by finding the global minimum (or a good local minimum) over this surface.

In many ways optimization in this continuous space is like the optimization in the discrete energy space. It is important not to confuse the two; one iteration of the optimization process for G requires many complete optimizations in E-space.

2.2. Analytical Results

G is a rather nice function, and several important results can be found analytically. The principle one used by the learning procedure,

$$\frac{\partial G}{\partial w_{ij}} = -\frac{1}{T}(p_{ij}-p'_{ij}) \tag{1}$$

was given in (Hinton et al 1984). p_{ij} is the probability that two units are both on when the visible units are clamped into environmentally specified states, and p'_{ij} is the corresponding probability when the visible units are not clamped. We sample these probabilities by running the machine in two phases. When it is running under the influence of the environment, we metaphorically say it is in the wake phase; when free running it is in the sleep phase. By sampling these two probabilities we can estimate the gradient of G, and use this as the basis for optimization. T is only a scale factor for the weights; doubling T and doubling all the weights will not affect the behavior of the machine, nor will it affect the learning provided that increments in the weights are also scaled appropriately. For simplicity, it will be assumed throughout this paper that the annealing searches in E-space use a final temperature of 1.0.

The following result about the smoothness of G gives us a guarantee that we can change the weights by a fixed proportion of the gradient and continue to descend.[2]

$$\left|\frac{\partial^2 G}{\partial s^2}\right| \leq \frac{W}{2} \tag{2}$$

where W is the number of weights, and s is any unit vector in G-space. To accomplish steepest descent, let s point away from the gradient vector. Then in the worst case, $\frac{\partial G}{\partial s}$ will increase toward zero by $W/2$ for every unit distance moved in direction s. Thus, we can always go a distance of $2\frac{|\nabla G|}{W}$ while continuing to decrease G, and twice this distance without ending up worse than where we started. In practice, this is an extremely conservative bound. When deciding how far to move in G space, one must also take into account the reliability of the estimates of p_{ij} and p'_{ij}; good estimates require longer sampling times. It has been found that

[2]See appendix I for the derivation.

moving a distance of $|\nabla G_{est}|$ works well for the problems we have studied.[3]

Here are two more interesting theorems about the topography of G:

- If all weights but one are fixed, there is exactly one value for the remaining weight which is a local minimum of G.
    - Corollary: There are no local maxima in G-space (though there may be several local minima).
- With no hidden units, G is concave upward in all directions.

The proof of the first theorem involves showing that the curvature of G along an axis, ij, is the variance of the cooccurence function, b_{ij}. That of the second involves showing that the curvature in an arbitrary direction is a quadratic form involving the direction vector and the covariance matrix of b.

2.3. What G Looks Like

For most environments that we have studied there are a few equally probable desired states, and many improbable states. This environmental structure and the previous theorems lead to the following mental image of G: In polar coordinates, G is very much dependent on the angles, but not so much on the distance from the origin. More specifically, for any point in G-space, G will be more or less monotonic on the path from the origin to the point, and on toward infinity. Basically the idea is this: For a given set of weights, if we increase them all proportionally, they will increase the probability difference between high and low probability states. If they lowered G with respect to its value at the origin, increasing them will further decrease G, and vice versa.

To be more precise, it can be shown that

$$\frac{\partial G}{\partial r} = \langle E \rangle - \langle E' \rangle$$

where r is the polar coordinate radius, $\langle E \rangle$ is the average energy when the environment clamps the visible units, and $\langle E' \rangle$ is the average energy when the machine is free running.

Thus claiming that scaling the weights will enable them to do whatever they were doing more effectively entails claiming that $\langle E \rangle - \langle E' \rangle$ doesn't change sign with the scaling. Empirical investigation of G for a very simple problem corroborates this conclusion (see figure 2-1).

When examining a two dimensional cross section of G, it is important not to over generalize. In figure 2-1 it

[3] Typically $|\nabla G_{est}| \gg |\nabla G|$; see section 3.1.

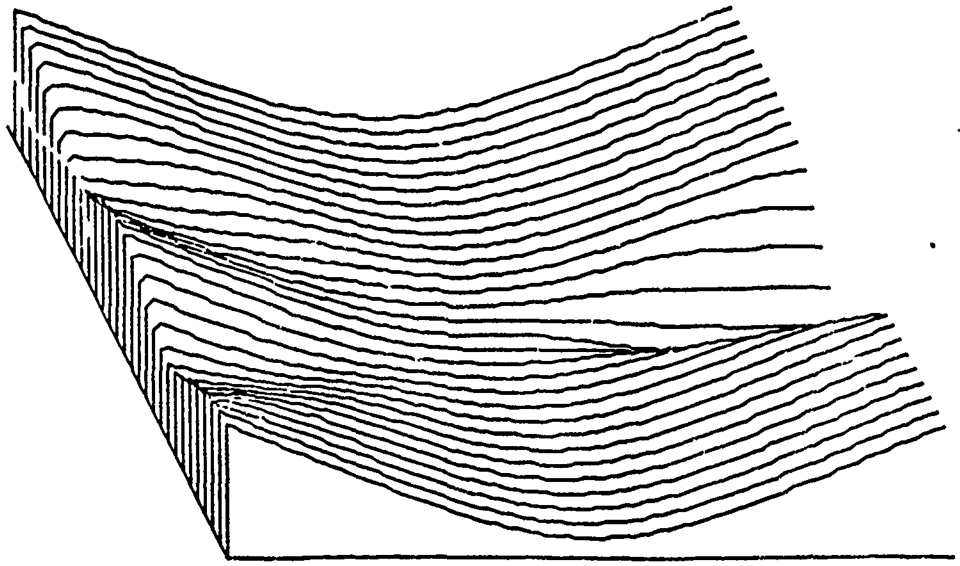


Figure 2-1:
Two dimensional cross section of G for the 1-1-1-1 Encoder problem.[4]

appears that the solution lies in the ravine sloping down to the right. In fact, one should only conclude that the best trade-off between the independent vectors X and Y in G-space given that everything else must remain fixed is to have Y small and negative and X large and positive. It is quite possible that by changing some other vector, a better ravine could be found.

Getting out of local minima presumably involves getting out of the wrong ravine and into the right one. Obviously this is much easier if the errant weights are small, since the ravines seem to converge at the origin. Since the ridges between ravines rise monotonically, for moderate sized weights it is almost certain that the machine will be in one of the ravines.[5] The differences in the heights of the floors of the ravines are small enough, however, that it is difficult to select the right one; generally there are more wrong than right choices.

3. Search Methods

[4]The 1-1-1-1 Encoder is described in section 4.1.

[5]The amount of noise in the search determines how close to the bottom of a ravine the machine generally stays.

3.1. Steepest Descent with Noise

In section 2, optimization was discussed as if the goal was to descend in G as fast as possible. Since G is not, in general, concave upward, and globally rather than merely locally optimal combinations of weights are desired, occasional uphill steps in G must be allowed, just as occasional uphill steps in E were allowed. It is less straightforward to define a temperature for searching G and to use it for annealing than it is for E. Currently the possibility of going uphill is provided in two ways. First, there is inevitably some noise introduced when estimating the gradient of G by sampling cooccurrence rates, so G may actually *increase* in the *estimated* direction of steepest descent. Second, we move farther in G than is guaranteed by the smoothness results. As will be shown, the estimates we have been using are poor, and lead to gross overestimates of the magnitude of the gradient; the fact that we use a step size that would exceed the smoothness guarantee even with perfect estimates is largely irrelevant.

Many of the various search techniques we are experimenting with can be viewed as attempts to modify the characteristics of the noise so as to facilitate rapid descent in G, while preserving the ability to escape from merely local optima. Before analyzing how we can improve our noise, let us examine what it looks like to begin with:

The following assumptions lead to a simple expression for the variance of the magnitude of the estimated gradient:

1. The number of weights is large

2. The samples are independent

3. All units independently come on half the time

The derivation in appendix II gives the following result:

$$\sigma^2(\| \nabla G_{est} \|) = \frac{W}{2N\sqrt{\pi}}$$

where W is the number of weights, and N is the number of samples. For a simple knowledge representation problem, I used $W=271$, $N=7$. A typical value for the magnitude of the gradient was probably one or two. Thus the variance, 11, was much larger than the actual value.

Contrary to assumption 2 above, the samples used to estimate p_{ij} and p'_{ij} are highly correlated since sequential samples of the global state differ by at most one unit. To combat this, we usually anneal several times in each sleep or wake cycle, producing several independent sequences of samples.

3.2. Moving a Constant Distance in G-space

One way to avoid making very large moves when the estimated gradient is very large is to always move a constant distance in each learning cycle. The large variance in the estimate of the magnitude of ∇G suggests that this may perform better than the previous method. If minima tend to have diameters of order 10 (with $T = 1$), moving a distance of 1 every time may be a good compromise between the possibility of not finding a minimum when close to it and the possibility of remaining too long in a sub-optimal minimum. In contrast, the proportional technique can produce the wrong sort of feedback: when at the bottom of a minimum, the gradient will be small, and little searching will be done. When far from a minimum, the gradient will be large, leading to overshoot. In fact, unbounded oscillations were the reason for originally trying the constant distance technique.

The technique requires non-local information, because the change to any weight is dependent both on its p_{ij} and p'_{ij}, and on the magnitude of the gradient estimate, which depends on all of the p_{ij} and p'_{ij}. Further, results in the next section show that this performs significantly *worse* than the proportional technique for a very small problem.

3.3. Changing Each Weight by a Fixed Increment

A related technique is to increment or decrement each weight by a fixed amount,[6] depending only on the sign of $\frac{\partial G}{\partial w_{ij}}$. This will also result in moving a constant distance in G on each step, however it has the advantage that all weights are modified every learning cycle and it does not require non-local information. With the previous technique, which only moves in the direction of the gradient, the machine could spend all its effort sloshing back and forth up the sides of a ravine. With the fixed increment method, it will also make progress *along* the ravine.

The fixed increment technique also has the advantage that it is compatible with two valued (or small integer valued) weights, for which simpler, and hence faster, simulation techniques suffice. In addition, it forces the machine to pick a representation that does not require very precise coordination between the numerical values of weights. An example of this undesirable behavior is where a positive and a negative weight must differ in magnitude by a precise amount to achieve the desired behavior. Such combinations of values are hard to learn and often lead to suicidal behavior[7] of the units involved.

Note that the fixed increment method is not a *steepest* descent technique.

---

[6] Hinton et al. 1984

[7] see section 5.1.1

3.4. Ravine Search

A class of non-steepest descent techniques are ravine search methods.[8] The existence of ravines is apparent from figure 2-1. Figure 3-1 shows the path of the machine in G space as it descends a ravine (the learning algorithm was modified so that motion was restricted to this cross section of the particular 7 dimensional G space of the 1-1-1-1 Encoder in order to produce this figure). It is apparent that there is more side-to-side movement than movement downhill. This sloshing results because the sides of a ravine are much steeper than the bottom, and the "sideways gradient" dominates the desired gradient along the ravine except at the very bottom.

3.5. Temporal Filtering: Giving the Weights Momentum

One of the ravine search techniques involves low pass filtering in time by averaging current estimates for p_{ij} and p'_{ij}, with previous estimates, some of which were taken on one side of the ravine, some on the other. Thus the sideways gradient will tend to cancel out, lessening the sloshing.

In addition to the improvements in ravines, where successive learning cycles have very different statistics due to sloshing, it also improves learning of weights whose coocurrences change little between successive cycles. In these cases, the effect is similar to increasing the sampling time within cycles, and reduces the variance. Excessive variance is the cause of two serious problems: getting lost (see section 4.2.4), and suicide (see section 6).

3.6. Spatial Filtering: Smoothing G-Space

An interesting possibility is to low pass filter G before searching it. Since the sides of ravines become higher farther from the origin in G, the ravines themselves will slope up in filtered G. This will keep the weights small, which is a big advantage: At large radii, it is a long way from one ravine to another, and the high ridges between ravines (caused by the large weights) also preclude getting equilibrium statistics, upon which the whole optimization procedure is based.

Further, it is easy to simulate the filtering. Adding gaussian noise to each weight will result in behavior which is an average of that obtained with nearby combinations of weights. If the noise is proportional to the magnitude of the weight, it will have the following effect: if two weights must differ by a constant, the variance of the difference, and hence G, will be lower when the magnitudes of the weights are small.[9]

[8]Several techniques for searching topographies with ravines are discussed in (Gelfand and Tsetlin, 1966).

[9]Geoff Hinton, personal communication

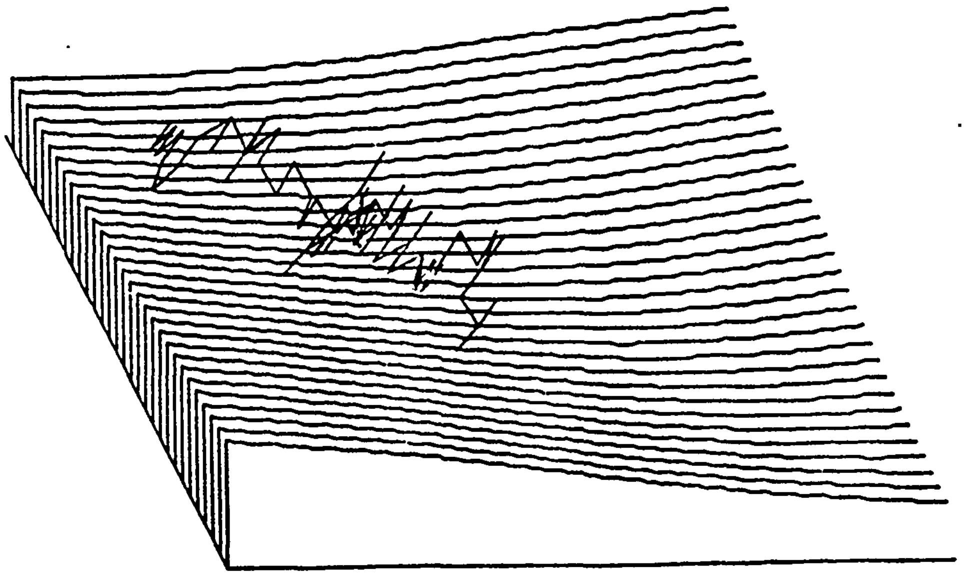


Figure 3-1: Path of Machine down a Ravine

3.7. Explicitly Keeping the Weights Small

We can also keep the search confined to the area near the origin by adding a term to the cost function which penalizes large weights.[10] For instance

$$G_{new} = G + \frac{h}{2} \sum_{ij} w_{ij}^2$$

$$\frac{\partial G_{new}}{\partial w_{ij}} = \frac{\partial G}{\partial w_{ij}} + h w_{ij}$$

To do gradient descent in this new space, we proceed as before, and in addition subtract from each weight a fraction of its value.

Very small values of h seem to be sufficient to keep the weights small.

3.8. Adding noise to the environmental input

In addition to altering the search techniques, there are alternative methods of gathering the statistics from which the gradient of G is estimated. A technique designed to keep the weights small is to occasionally clamp patterns during wake which do not occur in the environment (more accurately, the environment is modified so that many or all of the patterns have a finite probability). This avoids the following problem: If certain

[10] Barak Pearlmutter, personal communication

patterns never occur in the environment, they must be given infinitely higher energy than the ones that do occur. This requires infinite weights. One must be careful to modify the environment in a way that the Boltzmann Machine can easily model; if it has to devote some of its representation capacity to modelling the structure of this environmental noise, the technique will be counter-productive.

3.9. Choosing Environmental Patterns

Originally, we clamped all the environmental patterns for a period proportional to their probabilities. For environments with many patterns, however, this becomes prohibitively slow, since an annealing must be done every time a new pattern is clamped. Randomly choosing patterns for clamping introduces a lot of noise into the statistics, but seems unavoidable. Using temporal filtering in conjuction with random pattern selection may reduce the variance back to an acceptable level.[11]

3.10. Partial Clamping during Sleep

When we are sampling the machine's model of its environment, the visible units are not clamped. Thus p'_{ij} is always estimated from random samples, and will be noisier than estimates of p_{ij}. To combat this, we can encourage the correct distribution of samples by clamping a subset of the visible units during sleep. The clamped units will then behave according to the environment, and the sleep and wake statistics of the weights connecting them will be identical. Thus these weights will not change. However, the rest of the weights will obtain statistics with a lower variance.

In some situations, the Boltzmann Machine will have some visible units, O, dedicated to output, and some, I, to input. An environment then specifies a set of conditional probabilities of the form $P(O_\beta | I_\alpha)$. In this case, there is no need for the machine to learn the structure among the input units, since they will always be clamped. All the weights can be used for modelling the structure *between* the inputs and outputs. Thus sleep clamping can produce faster learning for this special case. The appropriate G measure in this case is

$$G = \sum_{\alpha\beta} P(I_\alpha \wedge O_\beta) \ln \frac{P(O_\beta | I_\alpha)}{P'(O_\beta | I_\alpha)}$$

Similar mathematics apply in this formulation and $\partial G / \partial w_{ij}$ is the same as before.[12]

For a knowledge representation problem with 32 environmental patterns, 10 of which are clamped during each learning cycle, the machine was not able to learn the environmental distribution due to the high noise. Using sleep clamping the learning proceded smoothly.

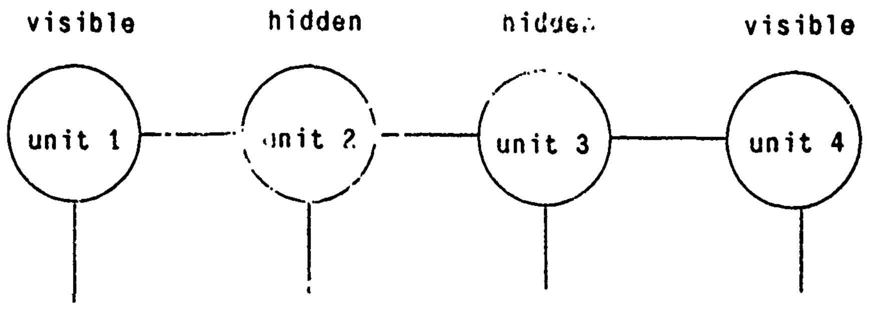
[11] see section 3.5

[12] Hinton et al. 1984

4 Empirical Tests of the Search Methods

4.1. Problem Description

To check our analytic results and compare search methods, a very small problem was investigated. There are four units, two visible and two hidden, and seven weights. In the environment, the two visible units are always either both on or both off. The hidden units must encode the state of the visible units. They form a channel through which the visible units can communicate their state. Figure 4-1 shows the layout of the problem.



PATTERN				PROBABILITY
Unit 1	Unit 2	Unit 3	Unit4	
on	x	x	on	.50
off	x	x	off	.50

Figure 4-1: Architecture and Environment for the 1-1-1-1 Encoder

This problem is the minimal case of the encoders described in (Hinton, et al, 1984) since each visible group consists of only one unit. The channel, however, consists of two groups of hidden units, neither of which can directly perceive *both* visible groups. This makes it much harder for random weight changes to increase the correlation between visible units and be reinforced. With less environmental influence, more of the undesirable tendencies toward suicidal behavior are exhibited (see section 6). This is proving useful for gaining a better understanding of the previously unexplained behavior on larger problems.

For this size system, it was possible to analytically determine p_{ij} and p'_{ij}, and do error analysis. Comparing statistics derived analytically with those found by a simulated real machine turned up some subtle bugs due to round-off error. Boltzmann Machines are extremely sensitive to small consistent errors, even when buried in large amounts of statistical noise. This makes such errors difficult to detect except by careful analysis.

In addition we were able to investigate the adequacy of our annealing schedules by comparing the analytic statistics, which reflect the true equilibrium distribution of states as given by the Boltzmann distribution, to those of a simulated real machine, which only asymptotically approach equilibrium. The results of this investigation are discussed in section 5.1.

Sets of weights which model the environment must include either three positive weights between the four units, or one positive weight and two negative weights. The bias weights must then be adjusted so that their unit is on about half the time (the environmental probability).

4.2. Results

Due to the large number of parameters, comparisons were done by varying only one parameter at a time. The control values were as follows:

- Ten independent samples were taken during both sleep and wake in order to estimate p_{ij} and p'_{ij}.
- At each learning cycle, the weights were modified so as to travel a distance of .5 in G-space in the direction of the estimated gradient.
- Each of the two patterns was shown in its correct form during wake for an equal period, and there was no clamping during sleep.
- The annealing schedule is shown in figure 4-2.[13] After each annealing, a single sample was taken at the final temperature, 1.0
- The weights were each initialized to random values uniformly distributed on [-2.0, 2.0], prior to each run.

TIME	TEMPERATURE
4.0	2.0
6.0	1.5
8.0	1.2
10.0	1.0

Figure 4-2: Annealing Schedule for 1-1-1-1 Encoder Problem

[13]In one unit of time all of the units are probed once each (on average). When probed, a unit decides whether to be on or off based on its energy gap.

Each of the following graphs represent ten runs of 500 learning cycles. For this problem,

$$G = P_{off} \log \frac{P'_{off}}{P_{off}} + P_{on} \log \frac{P'_{on}}{P_{on}} = .69$$

for uncorrelated behavior between the two visible units.[14] Because G varies tremendously from one learning cycle to the next, the data was smoothed before plotting, by averaging with nearby values.[15] The values for p_{ij} and p'_{ij} used to calculate G were found analytically, since the variance of the estimate from the simulated machine, using only 10 samples, is extremely high.

Figure 4-3 shows the results obtained with the control paramenter values, to which each of the other graphs should be compared. In each graph, it is clear when the machine finds the basic structure of the environment. The main things to note are how many successful runs there are out of ten trials, and how early the success becomes evident.

4.2.1. Number of Samples

The number of samples, N, used to estimate p_{ij} and p'_{ij} was varied between five and twenty. As can be seen from figure 4-4, the number of successes generally increases slightly with increase in number of samples, while the variablity in G over short periods decreases. Since time to estimate p_{ij} and p'_{ij} varies directly with N, ten looks like a good compromise between number of successes and running time. I think ten will remain a good value for larger problems. Since getting independent samples requires an annealing for each sample, and hence takes longer than getting dependent samples, there is a tradeoff between having many dependent samples or few independent samples, given a fixed sampling time. Gail Gong (personal communication) has determined the optimal number of annealings per sample period in terms of the variances of dependent and independent samples, but no empirical work has been done to see how accurately these variances can be estimated.

4.2.2. Choosing patterns

The machine has a much harder time when the environmental patterns are picked randomly, albeit with the correct probability (figure 4-5a). Unfortunately, for large problems, there are too many patterns for them all to be shown on each learning cycle. The techniques of clamping some units during sleep and temporal filtering[16] both tend to reduce the variance and may alleviate this problem. Figure 4-5b suggests that sleep clamping helps when each pattern is shown every cycle. It may have even more effect in the case of random

[14] It is lower when randomness is introduced in the clamping. G becomes .37 and .22 respectively for a clamping noise (explained in section 4.2.3) of .05 and .10 when the visible units are uncorrelated.

[15] specifically, the data was convolved with a triangle function of length seven

[16] see section 3.5

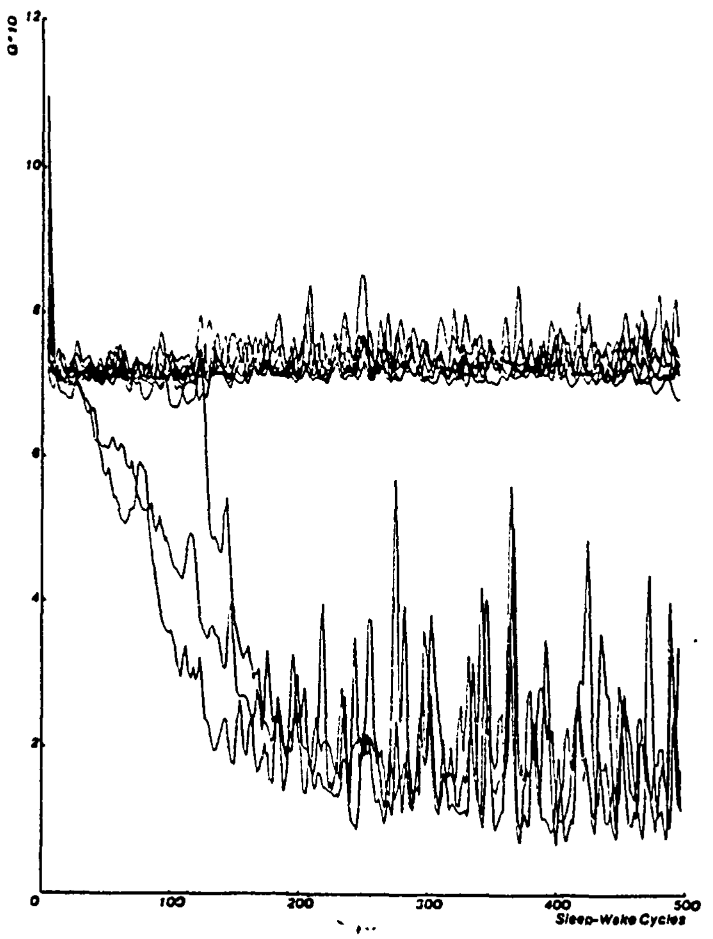


Figure 4-3: Performance Obtained with the Control Parameters

pattern selection.

4.2.3. Environmental Noise

One model for introducing noise in the environment, in the quest to combat unbounded weight growth[17], is to give each visible unit a probability to be clamped incorrectly. This probability is termed the clamping noise.

Boltzmann machines represent ratios of probabilities between states as energy differences between those states. With clamping noise, the probability ratios are relatively small, and change rather slowly as we move to patterns farther away in Hamming distance. Thus small energy gaps between adjacent patterns will suffice. With small energy gaps, the machine moves rapidly through E-space, and it is easy to get a good sample of the equilibrium values of p_{ij} and p'_{ij}. In geographic terms, the bottoms of the ravines flatten out and begin to rise again, rather than continuing to descend in the direction of ever larger weights. Thus we can argue that a Boltzmann machine performs better with clamping noise.

[17] see section 3.8

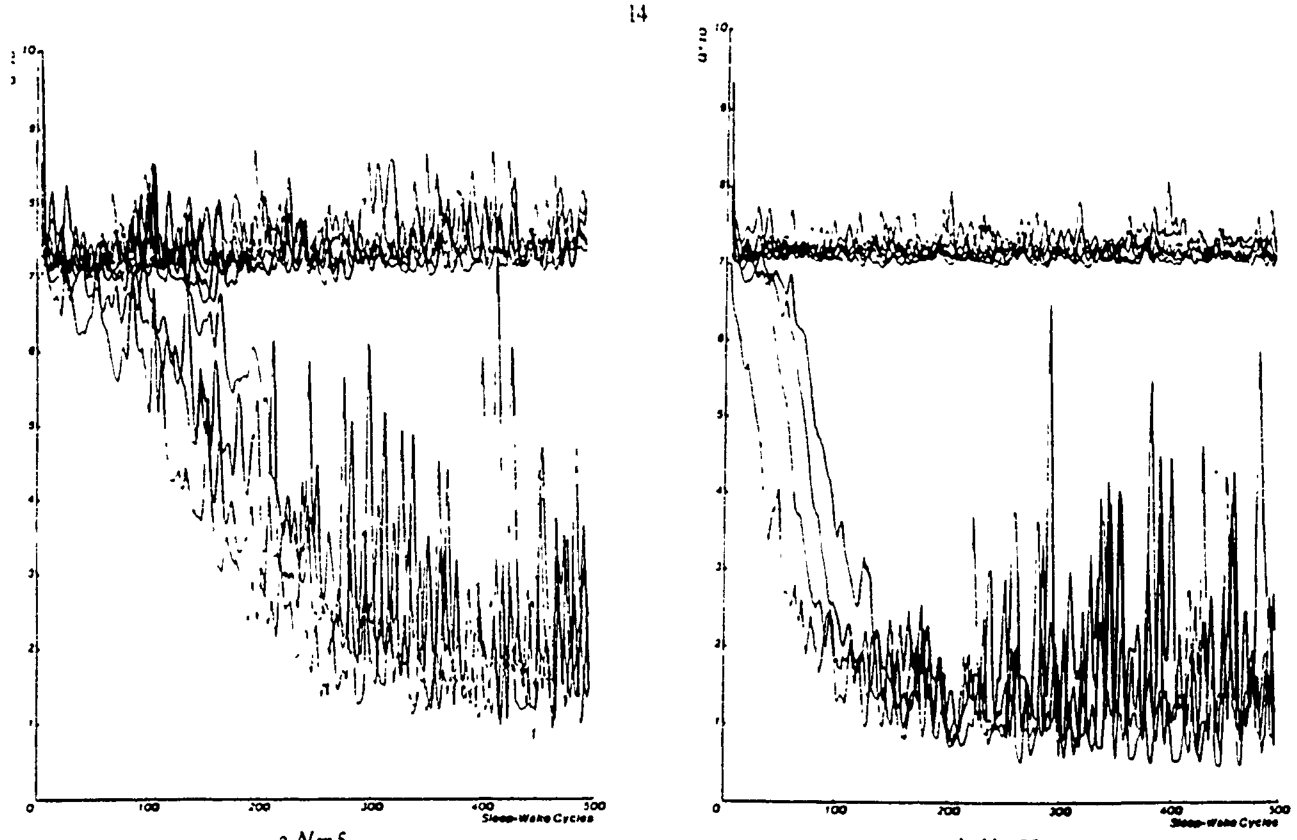
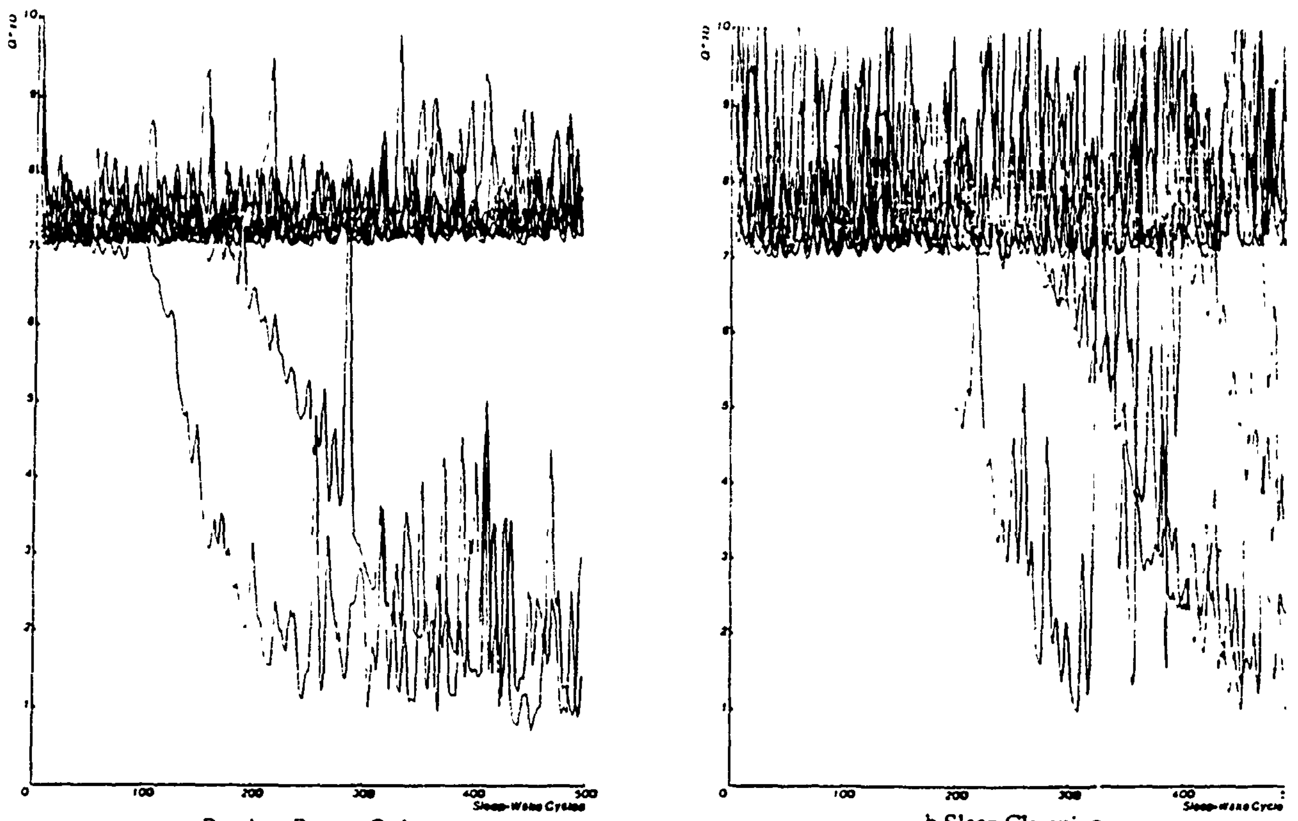


Figure 4-4: Effect of Number of Samples on Performance

The following clamping noises were compared. For each, the corresponding environment is given.

CLAMPING NOISE	STATES OF VISIBLE UNITS		PROBABILITY
.00	on	on	.50
	off	off	.50
.05	on	on	.45
	on	off	.05
	off	on	.05
	off	off	.45
.10	on	on	.41
	on	off	.09
	off	on	.09
	off	off	.41



a Random Pattern Order

b Sleep Clamping

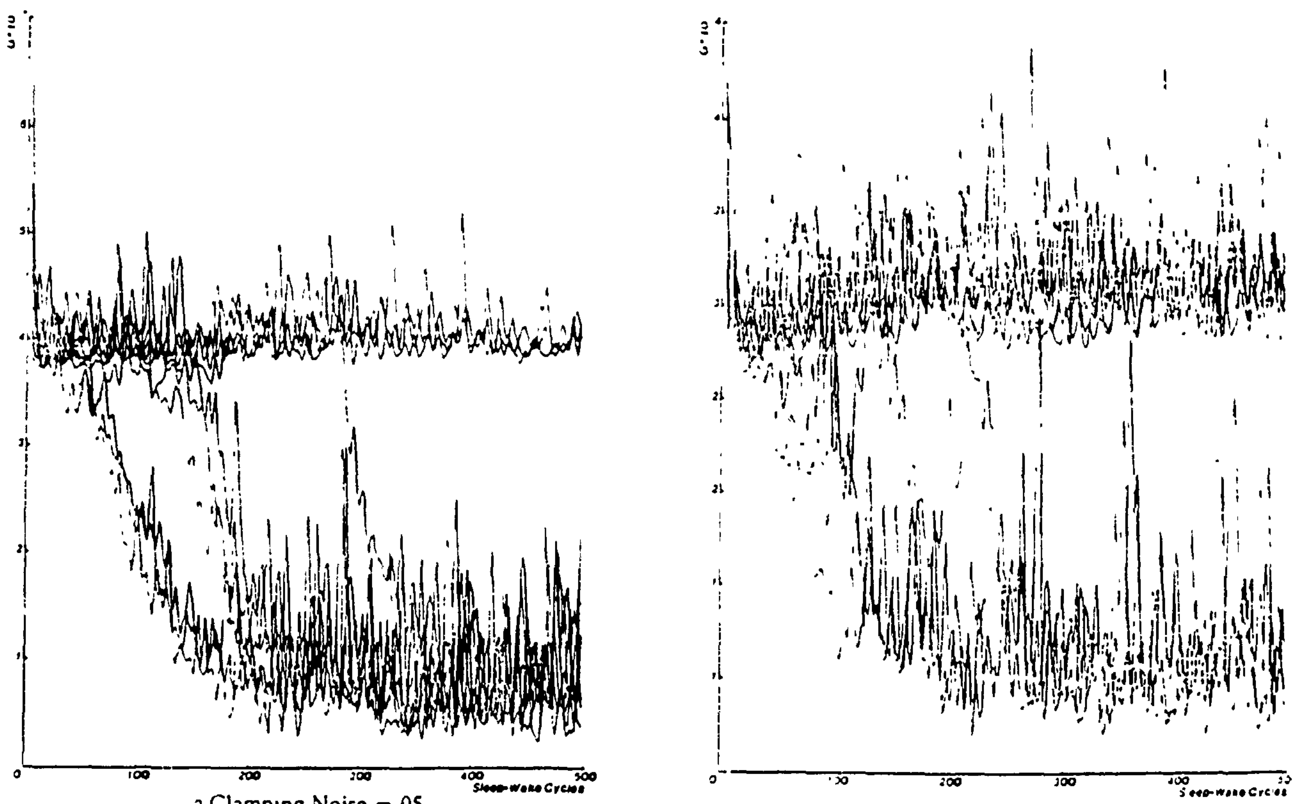
Figure 4-5: Effect of Clamping Strategy on Performance

The results in figure 4-6(a)[18] seem to indicate a second advantage of clamping noise. As the hidden units randomly modify their weights and form constraints between the visible units, they will initially not get them right, quantitatively, even if the signs are correct. It seems that the clamping noise reduces the penalty for these attempts to use the hidden units by allowing for a certain number of atypical states. It thus reduces the tendency toward dissociation (see section 5.1.1). Figure 4-6(b) suggests that too much clamping noise obscures the pattern to the point that it is not learned in the number of cycles given.

4.2.4. Size of Weight Step

Changing the size of the weight step seems to be the most sensitive parameter (figure 4-7), and is one whose optimal value may be expected to change drastically from one problem to another. There are two sets of data here; data when the weight step is fixed, and data when it is proportional to the gradient. The magnitude of

[18]Note that the y axis scales have been adjusted so that *G* for uncorrelated behavior is at about the same height.

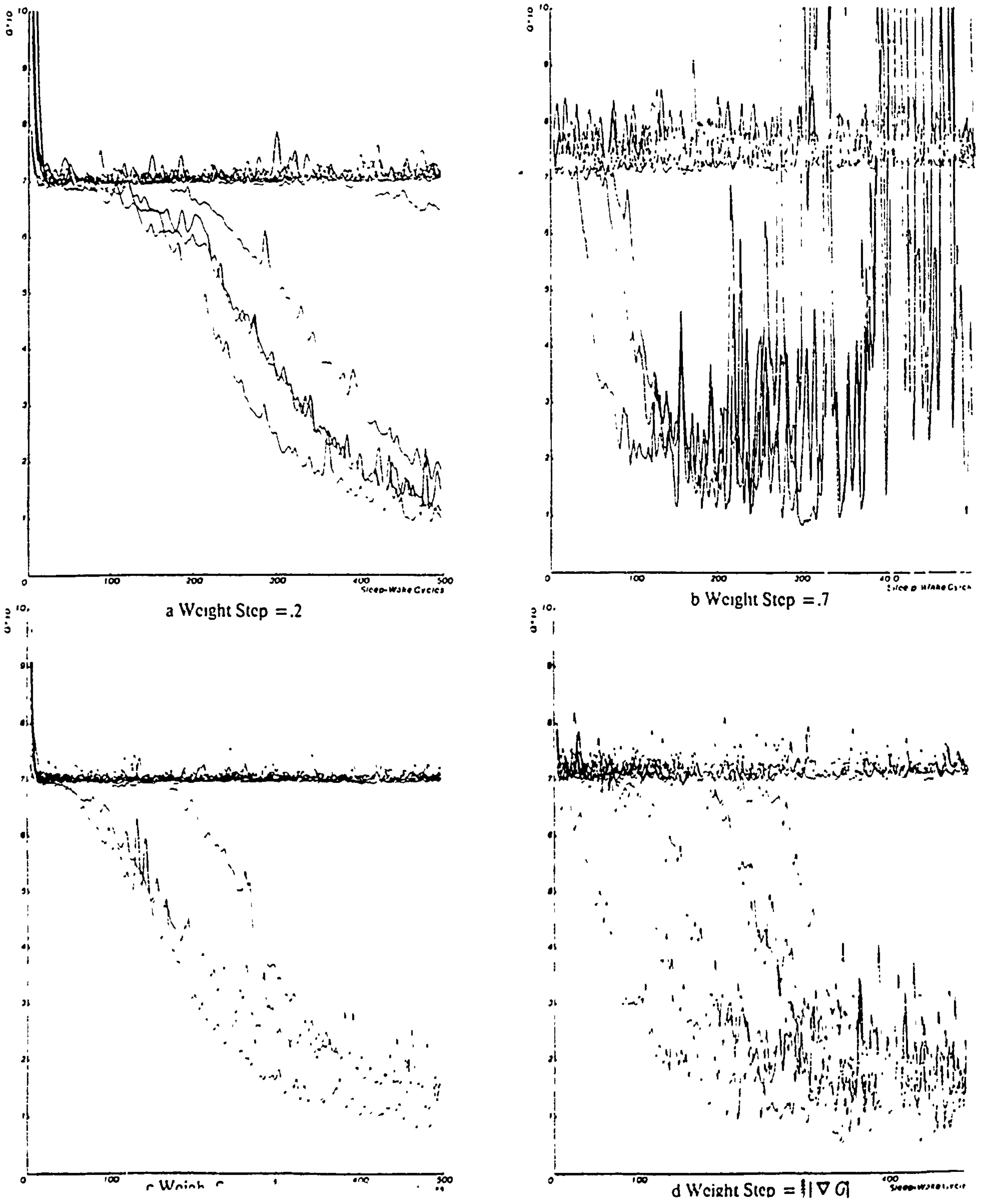


a Clamping Noise =.05

b Clamping Noise =.10

Figure 4-6: Effect of Environmental Noise on Performance

Figure 4-7. Effect of Weight Step on Performance



the gradient is less than the square root of the number of weights.[19] With 7 weights, $|\nabla G| \leq 2.6$. Usually, it was observed to be much lower, around 1. Thus, the distances moved on runs with a fixed step of .5 (figure 4-3) and on runs with a proportional step of $\frac{4}{7}|\nabla G|$ (figure 4-7c) are about the same, on average.[20] Reasonably enough, these two did about the same overall. With proportional steps, the machine is quite happy to stay pretty much where it is when the gradient is small. Thus the variance in G is smaller for runs which haven't yet got the structure of the environment than it is for the case of fixed steps. Conversely, when it is obvious what to do, the proportional step runs show a higher variance for G. The fact that fixed steps lead to wider searching when the gradient is small and there is nothing obvious to do seems to be an advantage for this technique.

Unfortunately, searching widely with little guidance from the gradient is likely to lead the machine far from the origin. In this never-never land of large weights, the machine can no longer reach equilibrium, and the nice monotonic behavior of G, even for large weights, evidenced in figure 2-1 is for naught.[21] This behavior can be seen in the graph for weight step = .7. If a run doesn't luck into a good combination of weights early on, it gets lost and never finds one. The basic result is this: In a problem where one need only go down in G-space, it is best not to go too far, lest one's estimate be wrong, or the gradient change too much along the way.

5. Reaching Equilibrium

5.1. Necessity of getting Equilibrium Statistics

Equation 1 is based on the assumption that p_{ij} and p'_{ij} follow from Boltzmann distributed global probabilities. If we do not anneal long enough before sampling,[22] $P_{\alpha_{est}}$ will be dependent on the starting state as well as the true P_α, resulting in erroneous p_{ij} and p'_{ij}. An example which comes up often in simulations is the test tube space (see figure 5-1).

The annealing process begins with the machine in a random state. It is equally likely to be within the collecting area for minimum A or for minimum B, since we hypothesize the same number of states in each. Assuming that annealing proceeds too fast to reach equilibrium, the machine will fall to the bottom of whichever minimum it began in and stay there. In this case, we are measuring the proportion of states with

[19] see appendix I

[20] $\frac{4}{7}$ is twice the bound given by equation 2 with $W=7$.

[21] Section 5.1 contains more discussion of the necessity of reaching equilibrium.

[22] Long enough is approximately the recurrence time for a random global state.

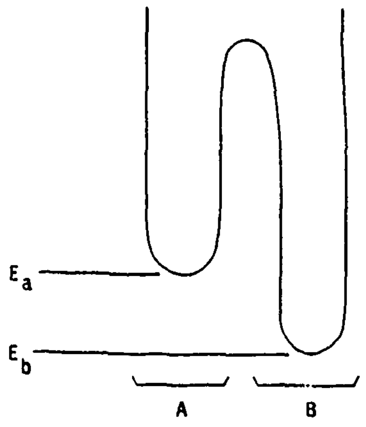


Figure 5-1: Energy Space which Demonstrates the Effect of Insufficient Annealing

downhill paths into each minimum, rather than the relative depths of each minimum.

This effect is evident in figure 5-2. The upper curve is G calculated analytically under the assumption of equilibrium statistics. The lower curve is G estimated from the statistics actually collected by the machine. Even where the analytic G is very high, the sleeping and waking statistics are nearly identical. The machine would be able to complete environmental patterns quite well.

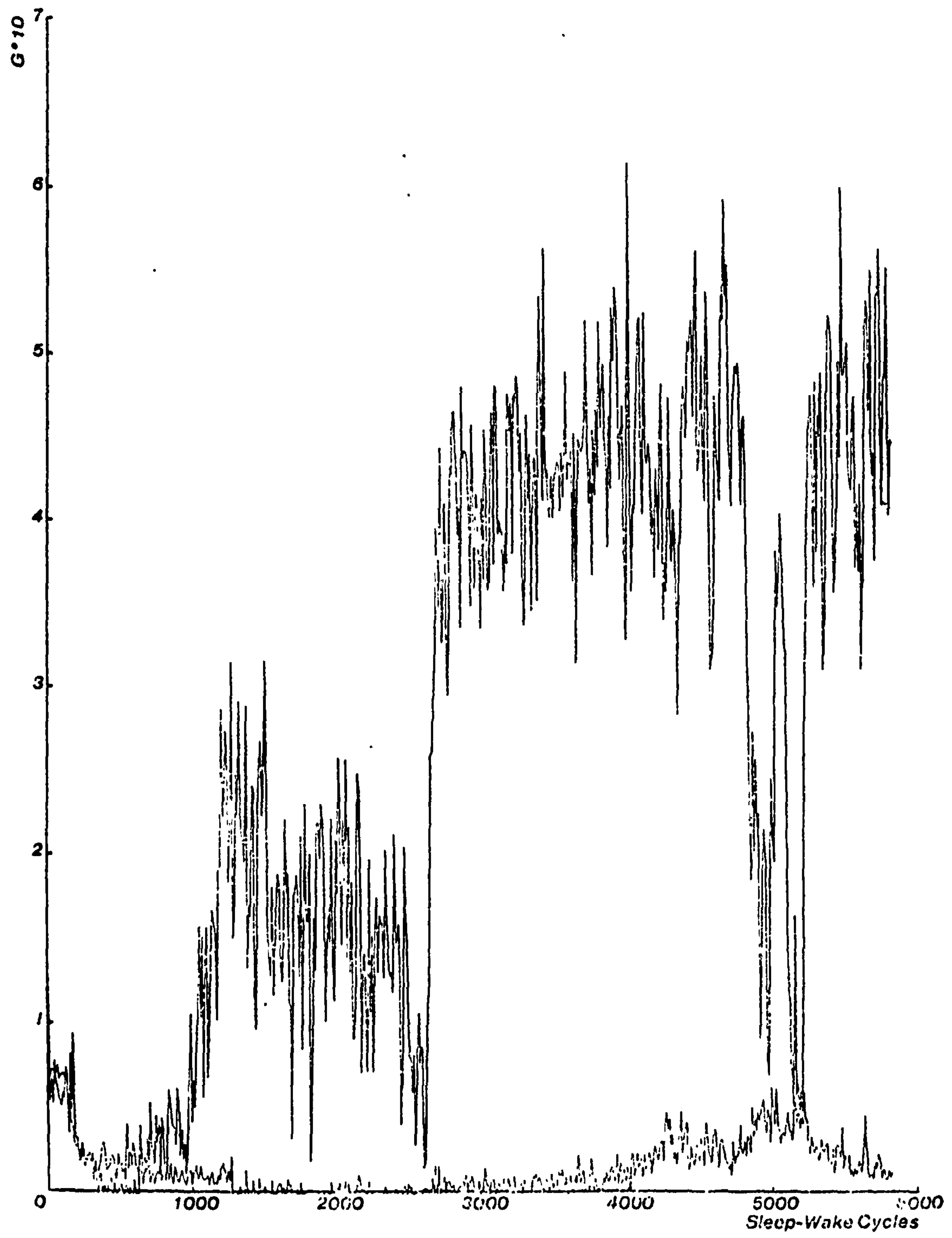
One may therefore ask whether it is essential to reach equilibrium; all we really want is for the sleeping and waking statistics to converge so we have a learning associative memory device. We can use the fact that the collecting areas of and Hamming distances between minima are important, and learn things that the theoretical Boltzmann Machine can't.[23] The answer seems to be no, though problems show up only under extreme conditions where the weights are large. Table 5-1 shows two sets of coocurrence statistics for a machine which is on the borderline of trouble. Due to the large weights, the machine's statistics are very close to the environmental ones, though the true equilibrium statistics are not.

5.1.1. Case Analysis

The problem has to do with the way non-equilibrium statistics compare with equilibrium ones. Consider the energy space for the encoder problem. There are two minima, one for each of the environmental states as in figure 5-1. Due to the symmetry of the architecture, the areas of the minima are the same; thus as the machine's statistics diverge from equilibrium, they will approach 50% in each minimum, independent of the relative depths. By happy accident, this is the same percentage in each state that occurs in the environment. To investigate how the machine handles itself when it must actively maintain the correct ratio between patterns, the environment was changed so that the visible units are off 60% and on 40%. Now, as the weights increase, the machine's statistics become closer to 50% than the equilibrium statistics. As a result, the difference in depth of the minima increases. Eventually the space looks like figure 5-3.

[23] David Ackley, personal communication.

Figure 5-2: Effect of Non-Equilibrium on Performance



Connected Units		Weight	p	p'
		ANALYTIC		
1	(bias)	11.4	.50	.90
2	(bias)	1.6	.50	.10
3	(bias)	1.1	.50	.10
4	(bias)	10.3	.50	.90
1	2	-18.2	.00	.00
2	3	16.7	.50	.10
4	3	-18.9	.00	.00
		SIMULATED		
1	(bias)	11.4	.50	.50
2	(bias)	1.6	.50	.50
3	(bias)	1.1	.50	.49
4	(bias)	10.3	.50	.51
1	2	-18.2	.00	.00
2	3	16.7	.50	.49
4	3	-18.9	.00	.00

Table 5-1: Sample Sleeping and Waking Coocurrence Statistics

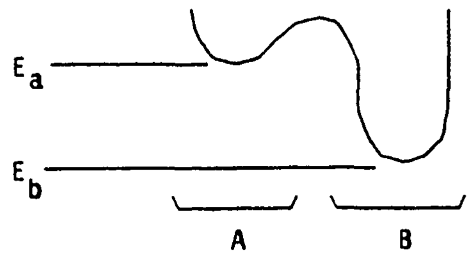


Figure 5-3: Worsening Energy Space

Now a tiny change in the height of the barrier makes a tremendous difference in the states probabilities. We thus have a tense combination of weights; large in magnitude, and precisely coordinated in relative terms. This in itself is not sufficient for disaster, however. Another effect of non-equilibrium prevents the weights from correcting perturbations. Once the hidden units have reached a state where one is on and one is off, they are unlikely to change, even if the positive weight is large(see figure 5-4). The visible unit is much more responsive to changes in the value of the positive weight. This is because there is no barrier to be crossed when the visible unit flips, as there is when both hidden units flip. To generalize, the probability ratio between two states is closer to the equilibrium ratio for states which are close in Hamming distance.

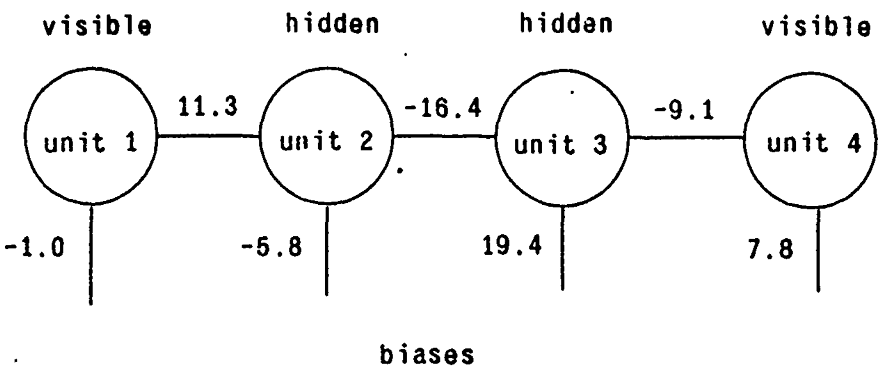


Figure 5-4: Weights Associated with Poor Energy Space

Eventually, random sampling errors will lead to weight changes which cause the more probable state (both off) to occur almost all the time. To correct this, the machine will lower the weights to both visible units and modify the weights to the hidden units so as to equalize the state probabilities. The latter process, however, takes place much more slowly than the former, and the net effect is that the visible units dissociate themselves from the hidden units, which then have no incentive to change; they continue to always be in the same state. This behavior is termed signal driven suicide (to distinguish it from suicide driven by random noise as discussed in section 6). The dissociation is generally stable since during both sleep and wake the statistics will be the same for the hidden units, either 100% or 0% on. Thus the estimated value of $\partial G / \partial w_{ij}$ will be 0.

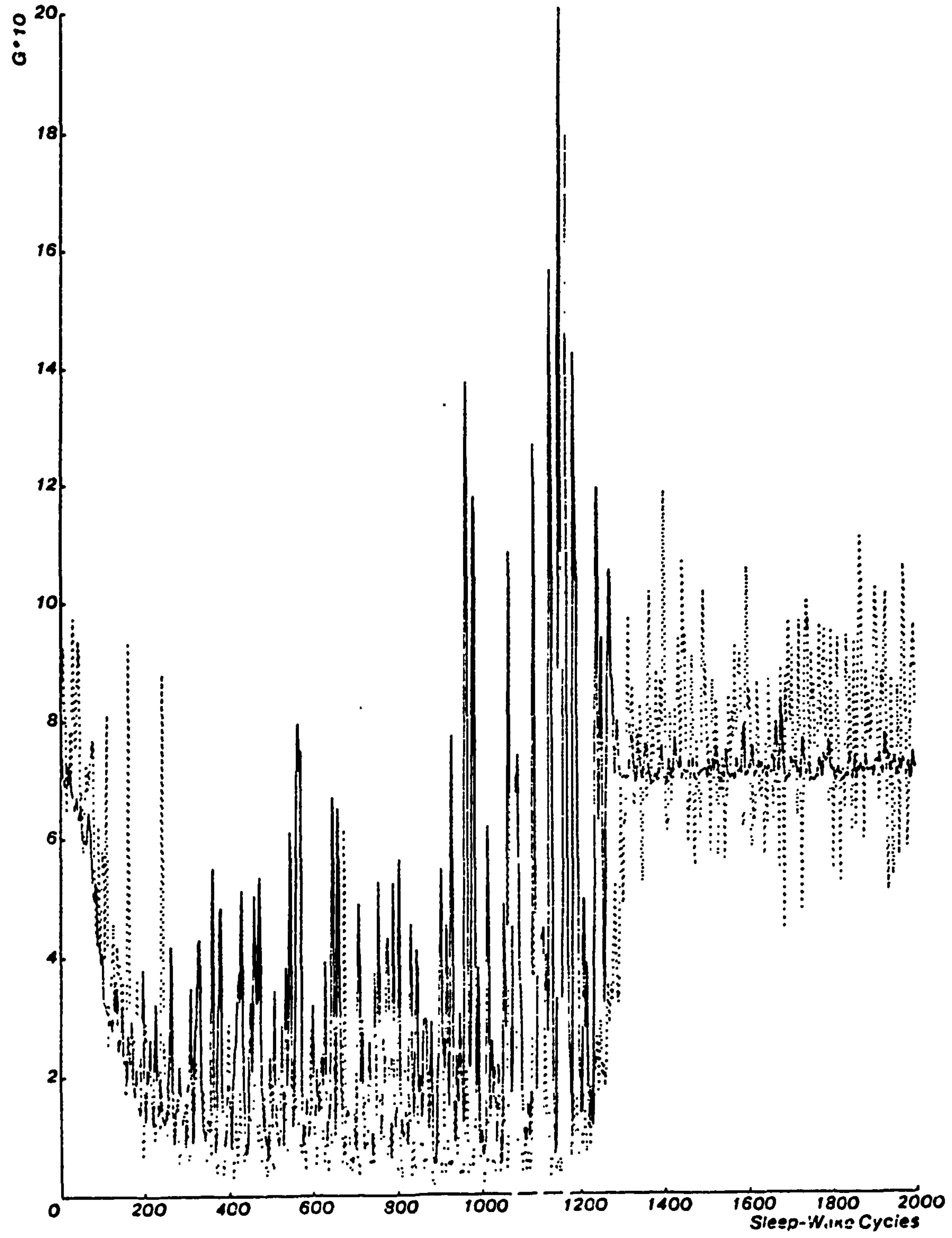
Figure 5-5 shows the behavior of the machine for an asymmetric environment. The dotted curve shows the value of G estimated from the machine's statistics; the solid curve shows the analytic value of G. The behavior is the same as in figure 5-2, until the asymmetry causes the visible units to dissociate from the hidden ones after 1300 cycles.

5.2. Ensuring Equilibrium

A promising technique for discouraging the type of behavior discussed in section 5.1 is to use two annealing schedules. When the weights begin to get large and create a test tube like energy space, the less conservative annealing schedule will begin to generate poor statistics before the more conservative one. The difference between statistics is then used to coax the weights away from combinations of values that make it hard to reach equilibrium.

Suppose we have two deep minima, A and B, and that we have two extreme annealing schedules. One quenches the system, going directly from infinite to zero temperature, and thus measures the relative areas of the minima. The other is slow enough to reach equilibrium, and thus measures the relative depth of the

Figure 5-5: Effect of Asymmetric Environmental Probabilities



minima.

Now we can insert special learning cycles where quenching takes the place of the normal wake cycle, and no units are clamped. After collecting statistics, we modify the weights in the normal manner:

$$\Delta w_{ij} = \frac{\varepsilon}{T}(p'_{ij_{fast}} - p'_{ij_{slow}})$$

If the depth of a minimum is too large for its collecting area, all weights between units that cooccur in that minimum will be reduced, and vice versa for overly shallow minima. The effect is to change the depth of the minima so that the equilibrium statistics match those obtained with the faster schedule. Since the total areas of minima remain relatively constant with time, it prevents any minimum from getting too deep. If we use a moderately fast anneling schedule instead of a very fast quench, the special learning cycle will have no effect on minima until they get too deep for the fast schedule to reach equilibrium. As this point is approached, the special learning cycle will prevent further deepening.

A reasonable implementation for these extra cycles would seem to be to alternate them with standard learning cycles. If we are willing to use the same epsilon for both cycles, we can save time and combine the two. Notice that the net weight change after a standard cycle followed by a special cycle is

$$\Delta w_{ij} = \frac{\varepsilon}{T}(p_{ij} - p'_{ij} + p'_{ij_{fast}} - p'_{ij_{slow}})$$

If we use two fast annealing schedules in the standard cycle, the second and third terms cancel. We can get the effect of the two types of cycles simply by using the fast schedule during the wake phase of the standard cycle, and the slow schedule during the sleep phase.

This technique is not robust in the sense that if a very bad energy space develops, it cannot recover, because even the conservative schedule provides poor statistics. The encoder problem was run with this technique, and the weights remained small. The bound on the weights could be varied by changing the difference in speed between the two annealing schedules. Similar results were obtained on a larger problem involving 37 units and 559 weights.

6. Suicide

One context in which suicide occurs is when hidden units get little feedback from the environment. This difficulty was encountered early on, but was not understood until recently. Hidden units tend to develop all positive or all negative weights and consequently are either always on or always off.

This effect is best explained by an analogy: Nearly all the loose gravel on a busy road accumulates at the side of the road, even if the road is flat. This is not because cars selectively push gravel toward the nearest side. It is because a piece of gravel has a much higher probability of making a move when it is near the middle of the road, so it spends very little time in the middle. Similarly, when a hidden unit comes on about half the time there will be a very high variance in p_{ij} and p'_{ij}, so $|p_{ij}-p'_{ij}|$ will be large and the weights will change a lot. When the unit is almost always on or almost always off, there will be very little variance and the weights will remain fixed. This effect can overpower the systematic effect due to the true value of $\partial G/\partial w_{ij}$.

Since this explanation for suicide depends on the fact that the magnitude of the change to a weight is a function of the weight, it was thought that removing this dependence could solve the problem. This may be accomplished by estimating the standard deviation of the estimates of p_{ij} and p'_{ij}, and dividing by it to determine the change to the weight. Results so far have been negative, however.

The suicide problem may be reduced by using more samples in the estimate of p_{ij} and p'_{ij} than were taken in the current learning cycle as mentioned in section 3.5, as well as by the techniques discussed above for keeping the weights small.

7. Conclusion

7.1. Robustness

The results developed above seem to be generally applicable to Boltzmann Machines. The techniques requiring problem dependent constants, namely constant distance weight modification (section 3.2) and explicitly keeping the weights small (section 3.7), were found to be unnecessary. Work on a 37 unit problem has indicated that the best values for parameters found above can be successfully used unchanged; these include the annealing schedules, number of patterns per learning cycle, epsilon, and amount of temporal filtering. Many techniques used formerly to prevent unbounded weight growth are rendered unnecessary by the single technique of special learning cycles using slow and fast annealing cycles (section 5.2). These old techniques include explicitly keeping the weights small, noise in the environment (section 3.8), and others too specialized for treatment here.

7.2. Future Work

The most important result about learning in Boltzmann Machines is that there exists a global measure, G, of the discrepancy between the machine's model of its environment and the actual environment, and that the partial derivatives of this measure with respect to the weights are locally computable. For problems where the environment can be modeled with pairwise constraints among the visible units, G is concave upward and gradient descent works well.

Upon reflection, it should not be surprising that more complicated environments that need hidden units to model higher order constraints among the visible units are difficult to model without directly considering non-local phenomena. Incorporating meta-knowledge about useful representations into an cost function could perhaps eliminate the need to search many of the local minima found in *G*. For large problems, hierarchical representations will be necessary. "Concepts" at each level can only be formed after those at lower levels upon which they depend. At each level, the influence of the evironment is weaker, and the gradient of the evaluation function will be correspondingly smaller. Additional constraints to select among possible ravines will be invaluable. Modifications to *G* which encourage small weights have been discussed. Encouraging units not to duplicate the behavior of other units may also be necessary. Possibilities such as these will become the center of investigation as the structure of *G*-space and the performance of techniques for searching it become better understood.

Acknowledgements

The members of the Boltzmann Group at CMU have continually guided this research through discussions and suggestions. I am particularly grateful to Geoff Hinton for assistance and encouragement, and to David Ackley for writing the Boltzmann Machine simulator.

Appendix[24]

I. Derivation of the Smoothness Result

The elements of the Hessian of G are given by

$$h_{kl} = \frac{\partial^2 G}{\partial w_k \partial w_l} = \frac{\partial}{\partial w_k}[p'_l - p_l]$$

$$= [\mu(b_k b_l) - \mu(b_k)\mu(b_l)] - \sum_\alpha P(V_\alpha)\mu(b_k b_l|\alpha) - \sum_\alpha P(V_\alpha)\mu(b_k|\alpha)\mu(b_l|\alpha)$$

where $\mu(\cdot)$ is the statistical mean; and b_k is 1 if both units connected by weight a are on, 0 otherwise. Each term is restricted to $[-.25,.25]$, so $|h_{kl}| \leq .5$. Thus the maximum curvature in any direction, $|\frac{\partial^2 G}{\partial s^2}|$, is $\frac{W}{2}$.

Other Smoothness Results

$$|\frac{\partial G}{\partial w_{ij}}| = |p_{ij} - p'_{ij}| \leq 1$$

since p_{ij} and p'_{ij} are probabilities. Thus, the slope along any axis is less than one, and the gradient must satisfy

$$|\nabla G| = \sqrt{(\frac{\partial G}{\partial w_{11}})^2 + \cdots + (\frac{\partial G}{\partial w_{nn}})^2} \leq \sqrt{W}$$

where W is the number of weights. Further,

$$\frac{\partial^2 G}{\partial w_{ij}^2} = (1 - p'_{ij})p'_{ij} - \sum_\alpha P_\alpha(1 - p_{ij}^\alpha)p_{ij}^\alpha$$

where p_{ij}^α is the conditional probability that units i and j are both on (coocurrence probability) given that state α is clamped over the visible units. Both terms are restricted to $[0, .25]$, so

$$|\frac{\partial^2 G}{\partial w_{ij}^2}| \leq \frac{1}{4}$$

II. Derivation of the Variance of the Estimated Gradient

$$\mu(\|\nabla G_{est} - \nabla G\|)^2 = \sum_{i \leq j}[(p_{ij_{est}} - p'_{ij_{est}}) - (p_{ij} - p'_{ij})]^2$$

Let

$$J = (p_{ij_{est}} - p'_{ij_{est}}) - (p_{ij} - p'_{ij})$$

Assuming a zero mean normal distribution for the difference between estimates (this is equivalent to assuming a large number of samples),

[24]Both appendices assume a temperature of 1 in the derivations.

$$\mu(\|\nabla G_{est} - \nabla G\|)^2 = \frac{W}{\sqrt{\pi}}\sigma^2(J)$$

Assuming we have N independent samples,

$$\sigma^2(J) = \frac{p_{ij}q_{ij} + p'_{ij}q'_{ij}}{N} \approx \frac{1}{2N}$$

assuming all units are independently on half the time,

$$\mu(\|\nabla G_{est} - \nabla G\|)^2 = \frac{W}{2N\sqrt{\pi}}$$

Bibliography

The references given here are all relevant to Boltzmann Machines, but they are not all referenced in the text.

Ackley, D.H., Hinton, G.E., & Sejnowski, T.J. A learning algorithm for Boltzmann Machines. *Cognitive Science*, in press.

Fahlman, S.E., Hinton, G.E., & Sejnowski, T.J. Massively parallel architectures for AI: NETL, Thistle, and Boltzmann Machines. *Proceedings of the National Conference on Artificial Intelligence AAAI-83*, Washington, DC, August 1983, 109-113.

Gelfand, I.M. & Tsetlin, M.L. Mathematical Modeling of Mechanisms of the Central Nervous System. In I.M. Gelfand, V.S. Gurfinkel, S.V. Fomin, & M.L. Tsetlin (Eds.) *Models of the Structural-Functional Organization of Certain Biological Systems.* Cambridge, MA: MIT Press, 1971.

Geman, S. & Geman, D. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. Unpublished manuscript, 1983.

Hinton, G.E. Implementing semantic networks in parallel hardware. In G.E. Hinton & J.A. Anderson (Eds.) *Parallel Models of Associative Memory.* Hillsdale, NJ: Erlbaum, 1981.

Hinton, G.E., & Anderson, J.A. *Parallel Models of Associative Memory.* Hillsdale, NJ: Erlbaum, 1981.

Hinton, G.E., & Sejnowski, T.J. Optimal perceptual inference. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition.* Washington, DC, June 1983, 448-453.

Hinton, G.E., Sejnowski, T.J., & Ackley, D.H. Boltzmann Machines: Constraint satisfaction networks that learn. Technical report CMU-CS-84-119, Carnegie-Mellon University, May 1984.

Hopfield, J.J. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences USA*, 1982, *79*, 2554-2558.

Kirkpatrick, S., Gelatt, C.D., & Vecchi, M.P. Optimization by simulated annealing. *Science*, 1983, *220*, 671-680.