A Tighter Bound for Graphical Models

M.A.R. Leisink* and H.J. Kappen†
Department of Biophysics
University of Nijmegen, Geert Grooteplein 21
NL 6525 EZ Nijmegen, The Netherlands ‡

August 4, 2000

Abstract

We present a method to bound the partition function of a Boltzmann machine neural network with any odd order polynomial. This is a direct extension of the mean field bound, which is first order. We show that the third order bound is strictly better than mean field. Additionally we derive a third order bound for the likelihood of sigmoid belief networks. Numerical experiments indicate that an error reduction of a factor two is easily reached in the region where expansion based approximations are useful.

1 Introduction

Graphical models have the capability to model a large class of probability distributions. The neurons in these networks are the random variables, whereas the connections between them represent conditional independencies. Usually,

*http://www.mbfys.kun.nl/~martijn

†http://www.mbfys.kun.nl/~bert

‡This research is supported by the Technology Foundation STW, applied science devision of NWO and the technology programme of the Ministry of Economic Affairs.

some of the nodes have a direct relation with the random variables in the problem and are called 'visibles'. The other nodes, known as 'hiddens', are used to model more complex probability distributions.

Learning in graphical models, defined as maximising the likelihood, can be done as long as the likelihood that the visibles correspond to a pattern in the data set, is tractable to compute. For a lot of special structures (like trees or other sparse networks) this can be done efficiently, but in general the time it takes, scales exponentially with the number of hidden neurons. For such architectures one has no other choice than using an approximation of the likelihood.

A well known approximation technique from statistical mechanics, called Gibbs sampling, was applied to graphical models in (Pearl, 1988). More recently, the mean field approximation was derived for sigmoid belief networks (Saul et al., 1996). For this type of graphical model the parental dependency of a neuron is modelled by a non-linear (sigmoidal) function of the weighted parent states (Neal, 1992). It turns out that the mean field approximation has the nice feature that it bounds the likelihood from below. This is useful for learning, since a maximisation of the bound either increases its accuracy or increases the likelihood for a pattern in the data set. Other work on bounds for neural networks can be found in (Jaakkola et al., 1996) and (Neal and Hinton, 1998).

In this article we show that it is possible to improve the mean field approximation without losing the bounding properties. In section 2 we show the general theory for creating a new bound using an existing one. If we start with a polynomial bound of degree k (mean field corresponds to $k = 1$), it turns out that the new bound is of degree $k + 2$. The procedure leads, however, to quite complicated formulae for belief networks. Therefore we first focus on Boltzmann machines in section 3. These networks are stochastic as well, but the connections are symmetric and not directed (Ackley et al., 1985). A mean field approximation for this type of neural networks was already described in (Peterson and Anderson, 1987). An improvement of this approximation was found by Thouless, Anderson and Palmer in (Thouless et al., 1977), which was applied to Boltzmann machines in (Kappen and Rodríguez, 1999). Unfortunately, this so called TAP approximation is not a bound. We apply our method to the mean field approximation, which results in a third order bound. We

prove the latter is always tighter than the standard mean field bound.

In section 4 the procedure is extended to sigmoid belief networks. In contrast to Boltzmann machines, we need an additional bound for this type of graphical model to make the final approximation tractable to compute. This is analogous to the mean field case, which was described in (Saul et al., 1996).

For both, sigmoid belief networks and Boltzmann machines, the combination of a lower and upper bound is important for inference, since conditional probabilities (which are ratio's of likelihoods) can be bounded as well. This article focuses solely on lower bounds, but more information about upper bounds can be found in (Jaakkola and Jordan, 1996a) and (Jaakkola and Jordan, 1996b).

In section 5 we present some numerical results and compare the third order bound with several existing approximation techniques. Finally, in section 6, we present our conclusions.

2 Higher order bounds

This section is divided in two subsections: The first shows the general procedure to obtain a $k+2$ order bound given a polynomial bound of order k; the second subsection applies this method to the known straight line bound of the exponential function which results in a third order bound.

2.1 Upgrading a bound

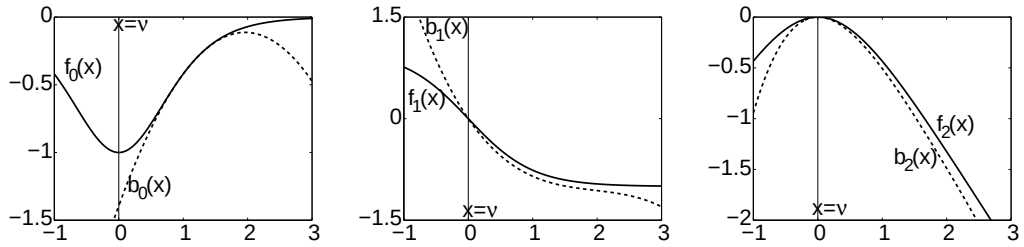
Suppose we have a function $f_0(x)$ and a bound $b_0(x)$ such that

$$\forall_x \quad f_0(x) \geq b_0(x) \tag{1}$$

Let $f_1(x)$ and $b_1(x)$ be two primitive functions of $f_0(x)$ and $b_0(x)$

$$f_1(x) = \int \mathrm{d}x \; f_0(x) \quad \text{and} \quad b_1(x) = \int \mathrm{d}x \; b_0(x) \tag{2}$$

This defines the functions $f_1(x)$ and $b_1(x)$ upto a constant. We choose this constant such that for some ν: $f_1(\nu) = b_1(\nu)$.



$f_0(x) \geq b_0(x)$ $\quad$ $f_1(x) \lesseqgtr b_1(x)$ $\quad$ $f_2(x) \geq b_2(x)$

Figure 1: The three stages in deriving a new bound. In this case $f_2(x) = -\log\cosh x$ and $\nu = 0$

Since the surface under $f_0(x)$ at the left as well as at the right of $x = \nu$ is obviously greater than the surface under $b_0(x)$ (which follows from equation 1) and the primitive functions are equal at $x = \nu$ (by construction), we know

$$\begin{cases} f_1(x) \leq b_1(x) & \text{for } x \leq \nu \\ f_1(x) \geq b_1(x) & \text{for } x \geq \nu \end{cases} \tag{3}$$

or in shorthand notation $f_1(x) \lesseqgtr b_1(x)$. It is important to understand that even if $f_0(\nu) > b_0(\nu)$ the above result holds. Therefore, we are completely free to choose ν.

If we repeat this and let $f_2(x)$ and $b_2(x)$ be two primitive functions of $f_1(x)$ and $b_1(x)$, again such that $f_2(\nu) = b_2(\nu)$, one can easily verify that

$$\forall_x \quad f_2(x) \geq b_2(x) \tag{4}$$

Note that by construction ν is the point where $f_1(\nu) = b_1(\nu)$, which is necessary for the above result to hold. This procedure is illustrated in figure 1.

Thus given a lower bound of $f_0(x)$ we can create another lower bound. In case the given bound is a polynomial of degree k, the new bound is a polynomial of degree $k + 2$ with one additional free parameter. In the next subsection we apply this procedure to the exponential function.

2.2 The exponential function

Starting with the trivial bound $\mathrm{e}^x \geq 0$ and applying the procedure of section 2.1, we derive

$$f_0(x) = \mathrm{e}^x \geq 0 = b_0(x) \tag{5}$$

$$f_1(x) = \mathrm{e}^x \lesseqgtr \mathrm{e}^\nu = b_1(x) \tag{6}$$

$$\forall_\nu \quad f_2(x) = \mathrm{e}^x \geq \mathrm{e}^\nu \left(1 + x - \nu\right) = b_2(x) \tag{7}$$

One can verify that the condition that $f_1(\nu) = b_1(\nu)$ and $f_2(\nu) = b_2(\nu)$ indeed is met and therefore we can conclude that $b_2(x)$ is also a lower bound on the exponential function. The function $b_2(x)$ here is the tangent of e^x at the point $x = \nu$ and due to the convexity of the exponential function, that is indeed a lower bound.

Nothing will stop us, however, from applying the procedure again, but this time to the lower bound $f_2(x) \geq b_2(x)$, which yields

$$f_2(x) = \mathrm{e}^x \geq \mathrm{e}^\nu \left(1 + x - \nu\right) = b_2(x) \tag{8}$$

$$f_3(x) = \mathrm{e}^x \lesseqgtr \mathrm{e}^\mu + \mathrm{e}^\nu \left(\left(1 + \mu - \nu\right)\left(x - \mu\right) + \frac{1}{2}\left(x - \mu\right)^2\right) = b_3(x) \tag{9}$$

$$f_4(x) = \mathrm{e}^x \geq \mathrm{e}^\mu \left\{1 + x - \mu + \mathrm{e}^{\nu-\mu}\left(\frac{1 - (\nu - \mu)}{2}\left(x - \mu\right)^2 + \frac{1}{6}\left(x - \mu\right)^3\right)\right\} = b_4(x) \tag{10}$$

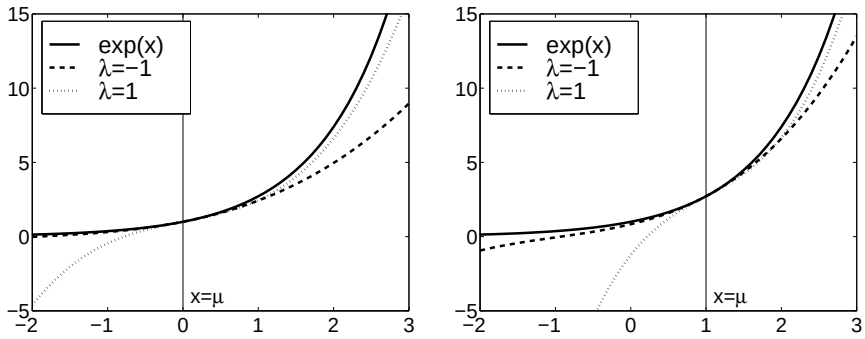
or (substituting $\lambda = \nu - \mu$)

$$\forall_{x,\mu,\lambda} \quad f_4(x) = \mathrm{e}^x \geq \mathrm{e}^\mu \left\{1 + x - \mu + \mathrm{e}^{\lambda}\left(\frac{1 - \lambda}{2}\left(x - \mu\right)^2 + \frac{1}{6}\left(x - \mu\right)^3\right)\right\} = b_4(x) \tag{11}$$

In figure 2 the derived bound is shown for some values of μ and λ. The role of μ is clearly to determine at which point the bound equals the exponential function. The role of λ can be seen as tightening the bound at the left of $x = \mu$ for negative and at the right for positive λ. The price we pay, however, is a less accurate approximation at the opposite side.

3 Boltzmann machines

In this section we derive a third order lower bound on the partition function of a Boltzmann machine neural network using the results from the previous



$\mu = 0$ $\mu = 1$

Figure 2: Examples of the bound as in equation 11 for some values of μ and λ.

section. The probability to find a Boltzmann machine in a state $\vec{s} \in \{-1, +1\}^N$ is given by

$$P(\vec{s}) = \frac{1}{Z} \exp(-E(\vec{s})) \tag{12}$$

where

$$-E(\vec{s}) = \frac{1}{2}\theta^{ij} s_i s_j + \theta^i s_i \tag{13}$$

There is an implicit summation over all repeated indices (Einstein's convention), unless stated otherwise. Z is the normalisation constant known as the partition function

$$Z = \sum_{\text{all } \vec{s}} \exp(-E(\vec{s})) \tag{14}$$

which requires a sum over all, exponentially many states. Therefore this sum is intractable to compute even for rather small networks.

3.1 A third order approximation

To compute the partition function approximately, we use the third order bound[1] from equation 11. We obtain

$$Z = \sum_{\text{all } \vec{s}} \exp(-E(\vec{s})) \geq \sum_{\text{all } \vec{s}} \mathrm{e}^{\mu(\vec{s})} \left\{ 1 - \Delta E + \mathrm{e}^{\lambda(\vec{s})} \left(\frac{1 - \lambda(\vec{s})}{2} \Delta E^2 - \frac{1}{6} \Delta E^3 \right) \right\} \tag{15}$$

[1]If we use the first order bound from equation 7, the result would be the well known mean field bound.

where $\Delta E = \mu(\vec{s}) + E$. Note that the former constants μ and λ are now functions of $\vec{s}$, since we may take different values for μ and λ for each term in the sum. In principle these functions can take any form. If we take, for instance, $\mu(\vec{s}) = -E(\vec{s})$ the approximation is exact. This would lead, however, to the same intractability as before and therefore we must restrict our choice to those that make equation 15 tractable to compute. We choose $\mu(\vec{s})$ and $\lambda(\vec{s})$ to be linear with respect to the neuron states s_i.

$$\mu(\vec{s}) = \mu^i s_i + \mu^0 \tag{16}$$

$$\lambda(\vec{s}) = \lambda^i s_i + \lambda^0 \tag{17}$$

One may view $\mu(\vec{s})$ and $\lambda(\vec{s})$ as (the negative of) the energy functions for the Boltzmann distribution $P \sim \exp(\mu(\vec{s}))$ and $P \sim \exp(\lambda(\vec{s}))$. Therefore we will sometimes speak of 'the distribution $\mu(\vec{s})$'. Since these linear energy functions correspond to factorised distributions[2], we can compute the right hand side of equation 15 in a reasonable time (i.e. polynomial increasing with the network size). For instance

$$Z_\mu = \sum_{\text{all } \vec{s}} \mathrm{e}^{\mu(\vec{s})} = \sum_{\text{all } \vec{s}} \exp\left(\mu^i s_i + \mu^0\right) = \mathrm{e}^{\mu^0} \prod_i 2\cosh\mu^i \tag{18}$$

or averages with respect to the distribution $\mu(\vec{s})$

$$\langle s_i s_j \rangle = \frac{1}{Z_\mu} \sum_{\text{all } \vec{s}} \mathrm{e}^{\mu(\vec{s})} s_i s_j = \tanh\mu^i \tanh\mu^j \tag{19}$$

Since (15) is a lower bound, we may maximise it with respect to its variational parameters μ^0, μ^i, λ^0 and λ^i to obtain the tightest bound.

3.2 A special case of the third order bound

Although $\lambda^i = 0$ does certainly not correspond to a maximum of (15), we choose to set them to zero, because numerical experiments indicate (not presented in this article) that for the real optimum of (15) the value of λ^i is close to zero for Boltzmann machines (given a Gaussian distribution of the weights) and, more importantly, this simplifies (15) enormously. This enables us to compare the third order bound with the mean field bound. The reader should

[2]This is the simplest choice. See however (Barber and Wiegerinck, 1998).

keep in mind, however, that the calculations in the rest of this article could be done for $\lambda^i \neq 0$, which would have tightened the bound even further.

Given $\lambda^i = 0$ we can rewrite the bound as

$$Z \geq Z_\mu \left\{ 1 - \langle \Delta E \rangle + \mathrm{e}^{\lambda^0} \left(\frac{1-\lambda^0}{2} \langle \Delta E^2 \rangle - \frac{1}{6} \langle \Delta E^3 \rangle \right) \right\} = B\left(E, \mu, \lambda\right) \tag{20}$$

where Z_μ is the partition function of the distribution $\mu\left(\vec{s}\right)$ as defined in equation 18 and $\langle \cdot \rangle$ denotes an average over that (factorised) distribution as defined in equation 19.

To find the tightest bound, we set all derivatives with respect to the variational parameters to zero. In appendix A this is done explicitly for μ^0 and λ^0, which yields

$$\begin{cases} \mu^0 = -\left\langle E + \mu^i s_i \right\rangle \\ \lambda^0 = -\frac{1}{3} \left\langle \Delta E^3 \right\rangle / \left\langle \Delta E^2 \right\rangle \end{cases} \tag{21}$$

Using this solution the bound reduces to

$$\log Z \geq \log Z_\mu + \log \left\{ 1 + \frac{1}{2} \mathrm{e}^{\lambda^0} \left\langle \Delta E^2 \right\rangle \right\} \tag{22}$$

where the term $\left\langle \Delta E^2 \right\rangle$ corresponds to the variance (or second order moment) of $E + \mu^i s_i$ with respect to the distribution $\mu\left(\vec{s}\right)$, since $\mu^0 = -\left\langle E + \mu^i s_i \right\rangle$. λ^0 on the other hand is proportional to the third order moment according to (21). Explicit expressions for these moments can be found in appendix B.

We still haven't set the variational parameters μ^i to an appropriate value. Taking the derivative with respect to these parameters yields

$$\frac{\partial B}{\partial \mu^i} = Z_\mu \left\{ -\langle \Delta E s_i \rangle + \mathrm{e}^{\lambda^0} \left(\left(1 - \lambda^0\right) \langle \Delta E s_i \rangle - \frac{\lambda^0}{2} \left\langle \Delta E^2 s_i \right\rangle - \frac{1}{6} \left\langle \Delta E^3 s_i \right\rangle \right) \right\} = 0 \tag{23}$$

which is an implicit equation for μ^i, analogously to the standard mean field equations. We can solve μ^i numerically by iteration. Wherever we speak of 'fully optimised', we refer to the solution of μ^i given by equation 23.

Although the fully optimised μ^i give the tightest bound, we like to focus for a moment on the suboptimal case where μ^i correspond to the mean field solution, given by

$$\forall_i \quad m_i \stackrel{\mathrm{def}}{=} \tanh \mu^i = \tanh\left(\theta^i + \theta^{ij} m_j\right) \tag{24}$$

For this choice for μ^i the $\log Z_\mu$ term in equation 22 is equal to the optimal mean field bound on the partition function. Since the last term in equation 22 is always positive, we conclude that the third order bound is always tighter than the mean field bound. Additionally, a real optimisation of μ^i using equation 23 would tighten the third order bound further.

The relation between TAP and the third order bound is clear in the region of small weights. If we assume that the terms of $\mathcal{O}\left(\theta^{ij^3}\right)$ are negligible, a small weight expansion of equation 22 yields (see also appendix B)

$$\log Z \geq \log Z_\mu + \log\left\{1 + \frac{1}{2}\mathrm{e}^{\lambda^0}\left\langle \Delta E^2 \right\rangle\right\} \approx \log Z_\mu + \frac{1}{4}\theta^{ij^2}\left(1 - m_i^2\right)\left(1 - m_j^2\right) \tag{25}$$

where the last term is equal to the TAP correction term (Kappen and Rodríguez, 1999). Thus the third order bound tends to the TAP approximation for small weights. For larger weights, however, the TAP approximation overestimates the partition function, whereas the third order approximation is still a bound.

4 Sigmoid belief networks

In the previous section we saw how to derive a third order bound on the partition function. For sigmoid belief networks we can use the same strategy to obtain a third order bound on the likelihood of the visible neurons of the network to be in some particular state. We start with a short description of sigmoid belief networks (see also (Neal, 1992)). After that we derive analogously to the previous section a third order bound.

A sigmoid belief network has connections θ^{ij} such that $\theta^{ij} = 0$ for $i \leq j$. For $i > j$ it is zero if neuron j is not a parent of i. The probability to find a neuron in state $+1$ given all its parents can be written as

$$P\left(s_i = +1 \mid \mathrm{pa}\left(s_i\right)\right) = \sigma\left(\theta^{ij}s_j + \theta^i\right) = \frac{1}{1 + \exp\left(-2\theta^{ij}s_j - 2\theta^i\right)} \tag{26}$$

and hence

$$P\left(s_i \mid \mathrm{pa}\left(s_i\right)\right) = \frac{\exp\left(\theta^{ij}s_i s_j + \theta^i s_i\right)}{2\cosh\left(\theta^{ij}s_j + \theta^i\right)} \tag{27}$$

where there is no implicit summation over i. The joint probability is given by

$$P(\vec{s}) = \prod_i P(s_i|\mathrm{pa}(s_i)) = \exp(-E(\vec{s})) \tag{28}$$

with

$$-E(\vec{s}) = \theta^{ij} s_i s_j + \theta^i s_i - \sum_p \log 2\cosh\left(\theta^{pi} s_i + \theta^p\right) \tag{29}$$

The last term, known as the local normalisation, does not appear in the Boltzmann machine energy function.

We have similar difficulties as in equation 14, if we want to compute the log-likelihood given by

$$\log \mathcal{L} = \log \sum_{\vec{s}\in H} P(\vec{s}) = \log \sum_{\vec{s}\in H} \exp(-E(\vec{s})) \tag{30}$$

The sum is taken only over the hidden units; the visible units are clamped to some given pattern. Using greek indices for the visible units, the clamped energy is given by

$$\begin{aligned} -E(\vec{s}) = \theta^{ij} s_i s_j + \left(\theta^i + \left(\theta^{i\alpha} + \theta^{\alpha i}\right) s_\alpha\right) s_i + \left(\theta^{\alpha\beta} s_\alpha s_\beta + \theta^\alpha s_\alpha\right) \\ - \sum_p \log 2\cosh\left(\theta^{pi} s_i + \left(\theta^{p\alpha} s_\alpha + \theta^p\right)\right) \end{aligned} \tag{31}$$

where the index p runs over both, hidden and visible units.

4.1 The problem of local normalisation

As said this problem has certain similarities with the Boltzmann machine. However, it is well known that due to the non-linear $\log 2\cosh$ term in the sigmoid belief energy, the bound as in equation 15 is intractable for all choices of $\mu(\vec{s})$ and $\lambda(\vec{s})$. Therefore it is necessary to derive an additional bound such that the approximated likelihood is tractable to compute.

We make use of the concavity of the log function to find a straight line upper bound[3] given by

$$\forall_\xi \quad \log x \leq \mathrm{e}^\xi x - \xi - 1 \tag{32}$$

[3]Note that this bound is also derivable using the method from section 2 starting with $-\frac{1}{x^2} \leq 0$.

We use this inequality to bound the $\log 2\cosh$ term in equation 31 for each p separately, where we choose ξ^p to be

$$\xi^p\left(\vec{s}\right) = \xi^{pi} s_i + \xi^p \tag{33}$$

We derive

$$-E\left(\vec{s}\right) \geq -\tilde{E}\left(\vec{s}\right) = \theta^{ij} s_i s_j + \tilde{\theta}^i s_i + \tilde{\theta} - \sum_p \left\{ \exp \xi_+^p(\vec{s}) + \exp \xi_-^p(\vec{s}) \right\} \tag{34}$$

with

$$\tilde{\theta}^i = \theta^i + \left(\theta^{i\alpha} + \theta^{\alpha i}\right) s_\alpha + \sum_p \xi^{pi} \tag{35}$$

$$\tilde{\theta} = \theta^{\alpha\beta} s_\alpha s_\beta + \theta^\alpha s_\alpha + \sum_p \xi^p + N \tag{36}$$

$$\xi_+^p\left(\vec{s}\right) = \left(\xi^{pi} + \theta^{pi}\right) s_i + \left(\xi^p + \theta^{p\alpha} s_\alpha + \theta^p\right) \tag{37}$$

$$\xi_-^p\left(\vec{s}\right) = \left(\xi^{pi} - \theta^{pi}\right) s_i + \left(\xi^p - \theta^{p\alpha} s_\alpha - \theta^p\right) \tag{38}$$

By introducing this second bound, we have rewritten the likelihood in an already solved form (see equation 15), since

$$\mathcal{L} = \sum_{\vec{s}\in H} \exp\left(-E\left(\vec{s}\right)\right) \geq \sum_{\vec{s}\in H} \exp\left(-\tilde{E}\left(\vec{s}\right)\right) \geq B\left(\tilde{E}, \mu, \lambda\right) \tag{39}$$

where the bound $B\left(\tilde{E}, \mu, \lambda\right)$ is tractable to compute. For instance

$$-\left\langle \tilde{E} \right\rangle = \left\langle \theta^{ij} s_i s_j + \tilde{\theta}^i s_i + \tilde{\theta} \right\rangle - \sum_p \left\langle \exp \xi_+^p\left(\vec{s}\right) \right\rangle - \sum_p \left\langle \exp \xi_-^p\left(\vec{s}\right) \right\rangle \tag{40}$$

Since $\xi_+^p\left(\vec{s}\right)$ and $\xi_-^p\left(\vec{s}\right)$ correspond to factorised distributions, the last two terms are easily computed. Unfortunately, due to the summation over p, the complexity of the third order bound increases[4] from $\mathcal{O}\left(N^3\right)$ to $\mathcal{O}\left(N^4\right)$.

In (Saul et al., 1996) the bound on the $\log 2\cosh$ term was derived in a different way. Their result is the special case that

$$\begin{cases} \xi^{pi} = \alpha_p \theta^{pi} \\ \xi^p = \alpha_p \left(\theta^{p\alpha} s_\alpha + \theta^p\right) \end{cases} \tag{41}$$

[4]With the assumption of not too many neurons in each layer, we can reduce the complexity to $\mathcal{O}\left(N^3\right)$. In fact, for a layered network, the order of the computational complexity is $\max\left(N^3, N^2 P^2\right)$ where P is the number of neurons in the largest layer.

where the α_p are N variational parameters. One might argue that the number of variational parameters in our approach (proportional to N^2) is too high for practical applications. Fortunately, we can reduce this number. It turns out that the optimal choice for ξ^{pi} is zero if the corresponding weight θ^{pi} is zero, as one can easily verify by investigating the derivative of the bound with respect to those parameters. If the network has not too many parents for each node (which is often the case), this corresponds mathematically to a few non-zero weights for each node. Therefore the number of variational parameters is rather linear in N than quadratic.

The attentive reader might have thought about taking the quadratic bound[5]

$$\forall_\nu \quad \log 2\cosh x \leq \frac{1}{2}(x-\nu)^2 + (x-\nu)\tanh\nu + \log 2\cosh\nu \qquad (42)$$

instead of (32). Although this choice indeed leads to a tractable function which obeys the bounding property, we have experimental evidence that it gives a worse approximation generally, mainly for the larger weights and thresholds.

5 Results

In this section we compare the third order bound with the mean field bound. In section 5.1 this is done for Boltzmann machines, in section 5.2 for sigmoid belief networks.

5.1 Boltzmann machines

In section 3 we derived the third order bound for Boltzmann machines. We distinguish three bounds on the partition function: (1) the mean field bound, $B_{\rm mf}$, (2) the third order bound using the (easy to obtain) mean field solution (equation 24) for μ^i, $B_{\rm tm}$, and (3) the fully optimised (equation 23) third order bound, $B_{\rm to}$. The reason that we consider $B_{\rm tm}$ apart from $B_{\rm to}$ is that the first has a lower computational complexity, which is especially important for sigmoid belief networks. Note that $B_{\rm mf} \leq B_{\rm tm} \leq B_{\rm to} \leq Z$. We created networks of $N = 20$ neurons with thresholds drawn from a Gaussian $\mathcal{N}(\mu = 0, \sigma = \sigma_1)$ and

[5]This can be derived using the method from section 2, starting with $1 - \tanh^2 x \leq 1$

weights drawn from $\mathcal{N}\left(\mu = 0, \sigma = \sigma_2/\sqrt{N}\right)$ for various σ_1 and σ_2, which is a so called SK-model (Sherrington and Kirkpatrick, 1975).

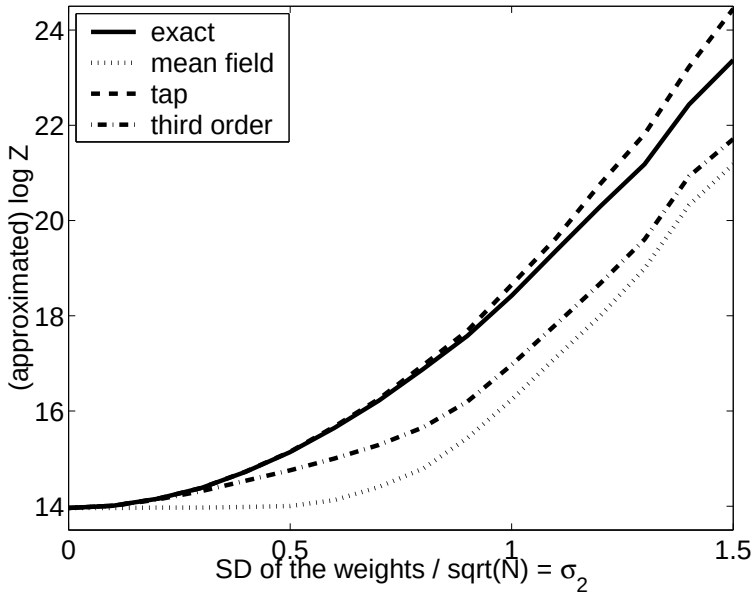


Figure 3: The exact partition function and three approximations: (1) Mean field, (2) TAP and (3) Fully optimised third order. The standard deviation of the thresholds is 0.1. Each point was averaged over a hundred randomly generated networks of 20 neurons. The inner plot shows the behaviour of the approximating functions for small weights.

In figure 3 the exact partition function versus σ_2 is shown with $\sigma_1 = 0.1$. In the same figure the mean field and fully optimised third order bound are shown together with the TAP approximation. For large σ_2 the exact partition function is linear in σ_2, whereas this is not necessarily the case for small σ_2 (see figure 3). In fact, in the absence of thresholds, the partition function is quadratic for small σ_2. Since TAP is based on a Taylor expansion in the weights up to second order, it is very accurate in the small weight region. However, as soon as the size of the weights exceeds the radius of convergence of this expansion (this occurs approximately at $\sigma_2 = 1$), the approximation rapidly diverges from the true value (Leisink and Kappen, 1999) (Plefka, 1981). Although the figure might suggest otherwise, the TAP approximation is neither an upper nor a lower bound.

The mean field and third order approximation are both linear for large σ_2,

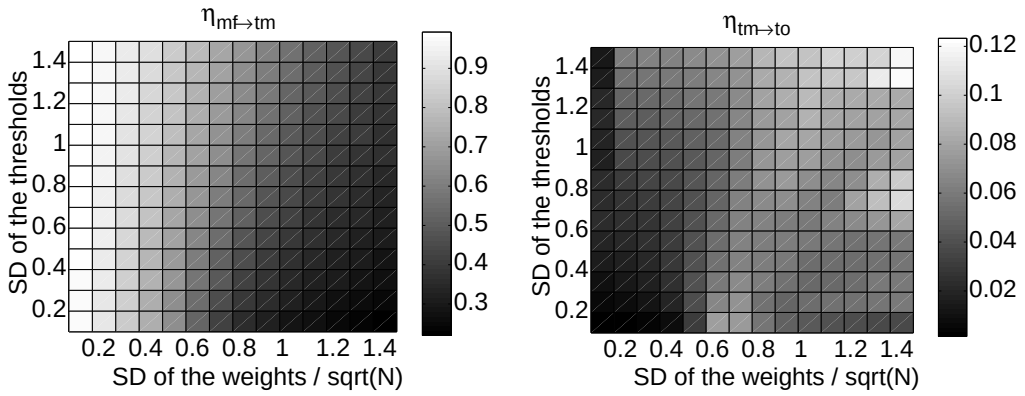


Figure 4: Relative improvement of the bound. Left: Comparison between third order and mean field both with the mean field solution for μ^i. Right: Comparison between the third order bound with the mean field solution for μ^i and the optimal solution. Zero is no improvement, one is maximal improvement. Each point was averaged over twenty randomly generated networks.

which prevents them from crossing the true partition function and violating the bound. For small weights ($\sigma_2 < 1$) we see that the third order bound is much closer to the exact curved form than mean field is. Thus if one wants to preserve the bounding property, but finds mean field too poor to work with, the third order approximation is worth to be considered.

We define the relative improvement from bound B_x to B_y by

$$\eta_{\mathrm{x}\to\mathrm{y}} = \frac{\log B_y - \log B_x}{\log Z - \log B_x} \tag{43}$$

This quantity takes on values from zero to one, for minimal to maximal improvement, respectively. We consider $\eta_{\mathrm{mf}\to\mathrm{tm}}$ and $\eta_{\mathrm{tm}\to\mathrm{to}}$. For several values of σ_1 and σ_2 we computed the three bounds, B_{mf}, B_{tm} and B_{to}, and the exact partition function. In figure 4 the relative improvements are shown. We conclude that a 30%-100% improvement is due to the use of the third order bound. Using the fully optimised μ^i instead of the (easier to obtain) mean field solution has only a minor effect (about 10%). We should mention, however, that the full optimisation becomes relatively more important for large σ_2. However, in this regime any expansion based approximation is too inaccurate for practical purposes.

Although the partition function is approximated more accurate, this is not necessarily the case for the mean firing rates and correlations in the system.

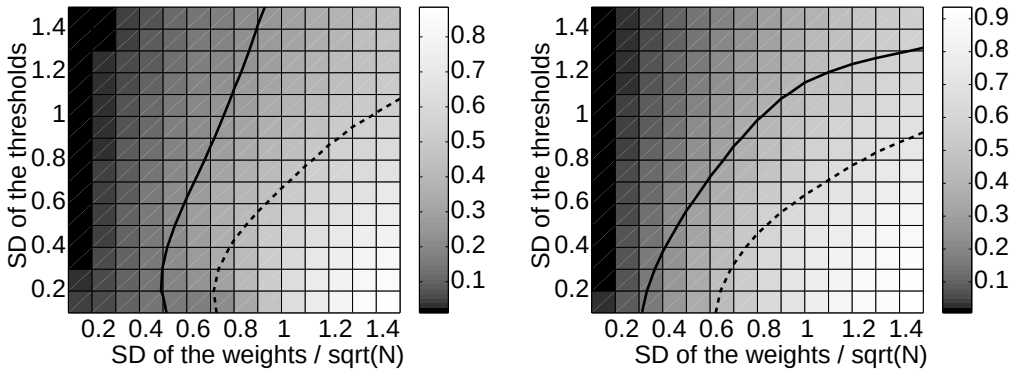


Figure 5: The ratio between third order and mean field approximation. Left: The ratio of the mean firing rates. Right: The ratio of the correlations. The solid line indicates a mean field error of 0.1; the dashed line a third order error of 0.1.

For a Boltzmann machine the mean firing rates are equal to $\partial \log Z / \partial \theta^i$ and, in the same way, the correlations are given by $\partial \log Z / \partial \theta^{ij}$. In figure 5 we plotted the ratio of these statistics obtained by the third order bound and the mean field approximation. In that way, a number smaller than one indicates that the third order approximation is better. This is done for 25 networks of ten neurons for all values for σ_1 and σ_2. If we define a sum squared error measure

$$\text{Error}^2 = \frac{1}{n} \sum_i \left(\langle s_i \rangle_{\text{exact}} - \langle s_i \rangle_{\text{approx}} \right)^2 \tag{44}$$

and similarly for the correlations, then the solid and dashed line in the figures correspond to an error of 0.1 for mean field and third order, respectively. We conclude that the third order method is better than mean field, at least in this region of weights and thresholds.

5.2 Sigmoid belief networks

As we saw in the previous subsection, we have two types of third order bounds. One for which we fully optimise all variational parameters and one which we compute using the mean field solution for μ^i. We have seen in figure 4 that the major improvement is due to the third order bound and not to the choice of optimisation. Therefore we propose to consider only the mean field solution, since the computation of the mean field parameters is considerably less complex

than a full optimisation. In the following we explore the computational quality of this bound.

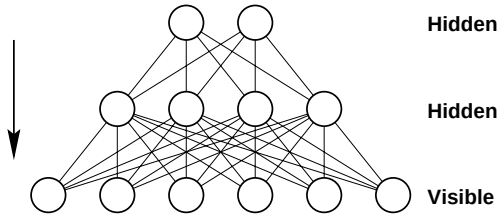


Figure 6: A toy problem.

Given this choice the following options are left

- using the mean field bound or the third order bound;
- using all ξ^{pi} as variational parameters or restrict them to the choice of Saul et al. as in equation 41.

To assess the error made by the various approaches, we use the same toy problem as in (Saul et al., 1996) and (Barber and Wiegerinck, 1998). The network has a top layer of two neurons, a middle layer of four neurons and a lower layer of six visibles (figure 6). All neurons of two successive layers are connected with weights pointing downwards and drawn from a uniform distribution over $[-b, b]$. Each neuron has a threshold drawn from a uniform distribution over $[-a, a]$. Since Saul used a 0/1-coding for the neuron activity, we transformed the randomly generated weights and thresholds from their representation to $-1/+1$-coding, which is used in this article. This makes our results comparable with those of Saul. We want to compute the likelihood when all visibles are clamped to -1. Since the network is rather small, we can compute the exact likelihood to compare the lower bound with. In figure 7 we show a typical example of the relative error, defined as $\log B / \log \mathcal{L} - 1$, for the four possible bounds given $a = b = 1$. It is clear from the figure that the use of the third order bound reduces the error enormously. In the regime of such a small error it is even helpful to use a full optimisation of the parameters ξ^{pi}.

We computed the relative improvement as defined in equation 43 (but with the log-likelihood instead of $\log Z$) between the first order bound with partial optimisation of $\xi^p(\vec{s})$ (as in (Saul et al., 1996)) and the third order bound with a full optimisation. The value of the improvement η for several a and b

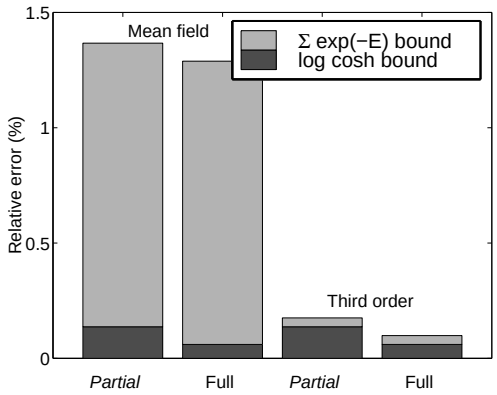


Figure 7: A typical example of the error made by the various bounds. The left two bars are mean field, the right two bars third order bounds. The word 'full' refers to a full optimisation of all ξ^{pi}, whereas the word 'partial' stands for the special choice for $\xi^p(\vec{s})$ as in equation 41. The grey surface is the error due to the bound on the exponential function, whereas black refers to the bound on the $\log 2\cosh$ term.

a\b	0.5	1.0	1.5	2.0
0.5	0.968	0.903	0.837	0.767
1.0	0.968	**0.906**	0.839	0.769
1.5	0.969	0.907	0.840	0.771
2.0	0.971	0.911	0.842	0.774

Table 1: The relative improvement as defined in equation 43 of the third order bound with full optimisation of $\xi^p(\vec{s})$ compared to the mean field bound with partial optimisation. The bold font denotes the toy problem as in (Saul et al., 1996).

is shown in table 1. The numbers are averaged over 981 networks (in 19 cases the optimiser did not converge). It is clear from the table that the gain by using the third order bound is almost independent of the size of the thresholds. The size of the weights, however, plays an important role and we see that the relative error decreases by more than a factor 20 (the relative improvement is about 95%) for small weights to a factor 4 for $b = 2$. Keep in mind that the third order bound is always better (i.e. $\eta > 0$), also for large weights. In figure 8 we show the histograms of the relative error for the case $a = b = 1$.

Besides this toy problem we address the quality of the approximation and the computation time for larger networks. Let us first define the cascade network

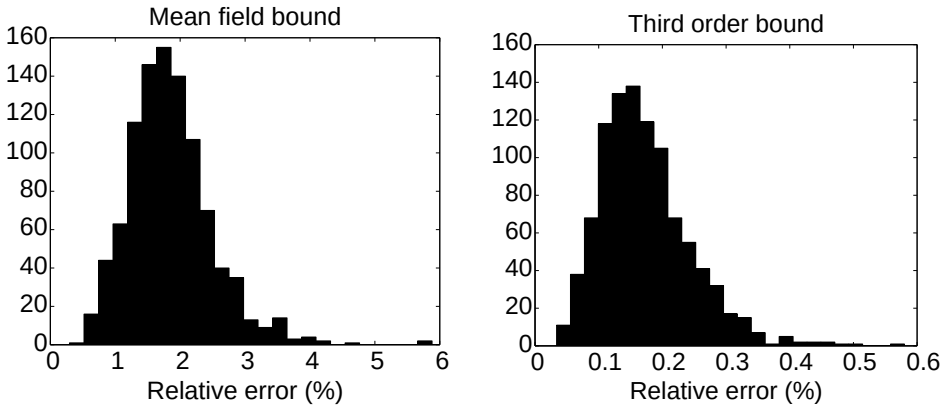


Figure 8: Histograms of the relative error for $a = b = 1$. The error of the third order bound is roughly ten times smaller than the error of the mean field bound.

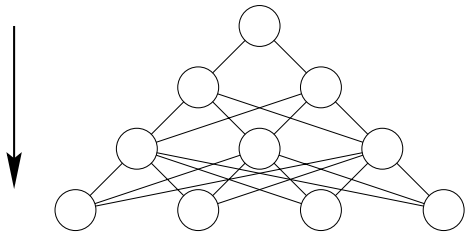


Figure 9: The cascade network. The lth layer has l neurons. Thresholds are initialised from a Gaussian with zero mean and standard deviation σ_1; weights between layer l and $l+1$ are from a Gaussian with zero mean and standard deviation $\sigma_2/\sqrt{l}$. The $\sqrt{l}$ makes σ_1 and σ_2 comparable in magnitude.

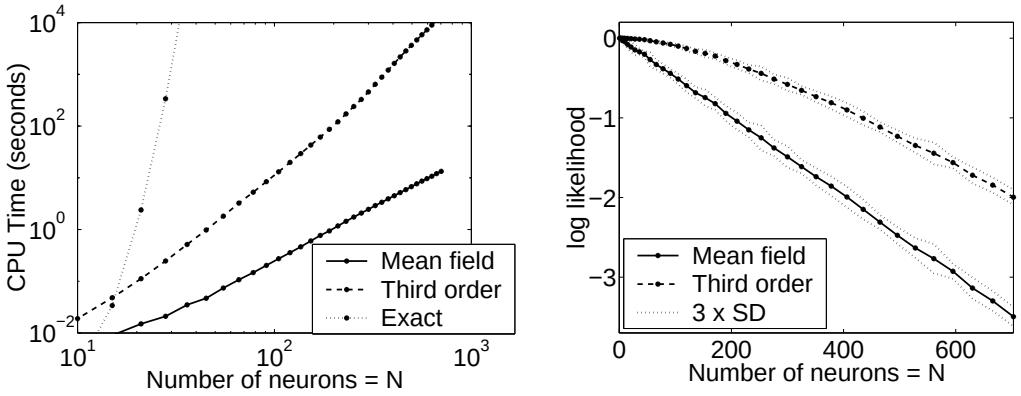


Figure 10: Computation of the likelihood for the cascade network. Left: Computation time on a double log scale for exact, mean field and third order. Right: Estimated log likelihood per neuron for mean field and third order. Since no neurons are clamped, the exact log likelihood is zero. Each point was averaged over ten networks. The dotted lines are three standard deviations from the mean.

as shown in figure 9. This is a layered network with L layers. The l-th layer ($l = 1 \ldots L$) has l neurons, which are fully connected to the previous and the next layer. A cascade network with L layers has $N = L(L+1)/2$ neurons. For each L up to 37 we initialised ten networks with random weights and thresholds ($\sigma_1 = \sigma_2 = 0.1$) and computed the mean field and third order bound on the log likelihood. Since no units were clamped, the exact log likelihood is zero. From figure 10 we conclude that the third order bound gives a significant error reduction, although computation time may be a drawback for large networks.

Although a better bound on the likelihood is nice, a more important aspect is whether this results in a better trained network. First of all we have to define what we mean by 'a better trained network'. Our goal can be to maximise the likelihood of a given data set by setting all the weights and thresholds to an appropriate value. In that case we view exact, mean field and third order just as three different models, each learning the data set as good as possible allowing a totally different set of weights for each model. On the other hand our goal can be to approximate the exact method as good as possible such that the weights, thresholds and mean activities obtained with our approximation method closely resemble the exact values. An accurate approximation in this sense makes it possible to reveal the hidden structure of a particular data set.

We started with a cascade network (figure 9) with $L = 5$ layers and initialised the network with zero thresholds and Gaussian distributed weights with standard deviation $\sigma_2/\sqrt{l}$. Then we computed the exact probabilities, $p(\vec{s})$, for all $2^5 = 32$ states of the bottom layer. Finally we learned this probability distribution by maximising the total likelihood

$$\log \mathcal{L}_{\text{total}} = \sum_{\vec{s}} p(\vec{s}) \log \mathcal{L}(\vec{s}) \tag{45}$$

with a standard gradient ascent procedure. The log likelihood is given by equation 30 for exact and by equation 39 for mean field and the third order method. The thresholds were initialised with zero and not adapted. This results in three sets of weights: (1) the exact weights, θ_{ex} (used to generate the probability distribution), (2) the mean field weights, θ_{mf}, and (3) the third order weights, θ_{th}.

The initialisation of the weights was random, but identical for the three methods. In this way we force them to learn the same hidden representation, which enables us to compare the weights directly. This is necessary, since any permutation of the hidden nodes would result in same maximum likelihood. Additionally, we repeated the experiments, where we initialised with the exact weights and did the maximisation. The exact method stopped immediately, but mean field and third order adapted the weights slightly and came up with comparable results as for the random initialisation.

When our goal is to maximise the likelihood regardless of the underlying model, the three methods does not differ very much. For $\sigma_2 \leq 1$ the relative error for the approximation versus the exact method is usually smaller than 0.1% When our goal is, however, to approximate the exact model and to find the weights that explain the hidden structure of the data set, the third order method turns out to be much better than mean field.

We ran twenty learning problems starting with random initialisation. In figure 11 we show the histograms of the difference between the exact and approximated weights for mean field and for third order. It is striking that mean field only manages to learn the weights between layer 4 and 5, whereas the third order method is capable to learn all weights up to the top quite accurately. If

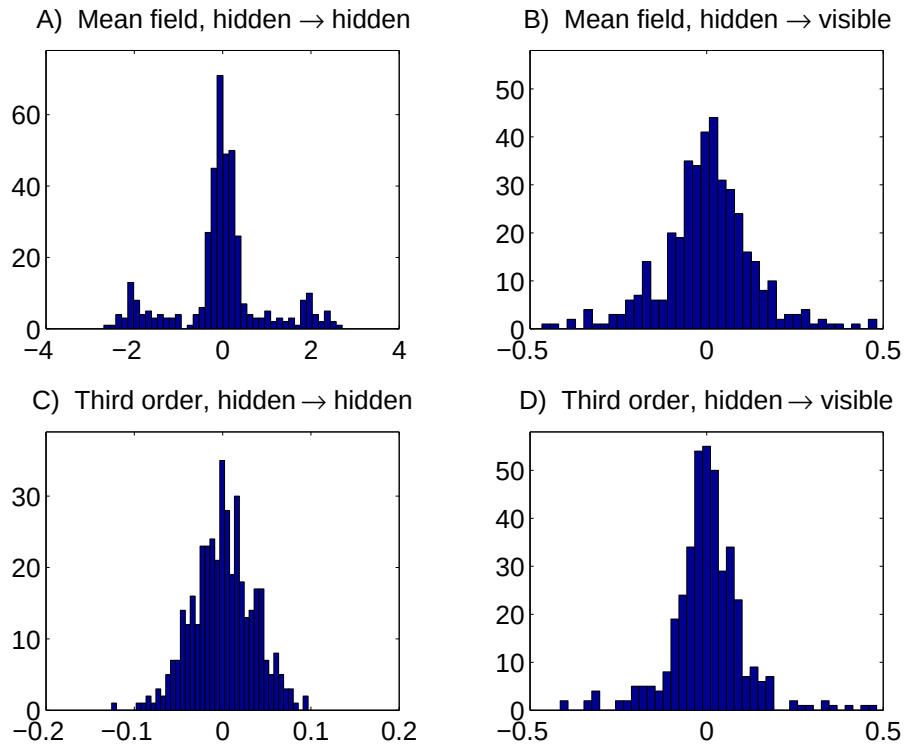


Figure 11: The difference between the exact and approximated weights after training a cascade network with five layers, $\sigma_1 = 0$ and $\sigma_2 = 0.5$ The upper two histograms correspond to mean field, the lower two to third order. Left is the histogram of weights from a hidden node to another hidden (upper layers); right is from a hidden to a visible (layer 4 to 5).

we define the error in the weights by

$$\text{Error}^2 = \sum_{ij} \left({\theta^{ij}}_{\text{exact}} - {\theta^{ij}}_{\text{approx}} \right)^2 \tag{46}$$

the average error for mean field was 0.566, for third order 0.073. In all runs, the third order error was less than mean field.

We conclude that both the mean field and third order method are capable to find a good log likelihood, although third order is still slightly better. Mean field, viewed as an approximation of the exact method, however, fails to learn the hidden structure accurately.

6 Conclusions

We showed a procedure to find any odd order polynomial bound for the exponential function. A $2k-1$ order polynomial bound has k free parameters per binary variable. For the third order bound these are μ and λ. We can apply this bound to the exponential function to derive a bound on the partition function. In the simplest case, where the free paramaters define the energy function of a factorised distribution, we have $(N+1)k$ free parameters. It is certainly possible to use other choices as was done in (Barber and Wiegerinck, 1998). Since the approximating function is a bound, we may maximise it with respect to all its free parameters.

In this article we restricted ourselves to the third order bound, although an extension to any odd order bound is possible. Third order is the next higher order bound to naive mean field. We showed that this bound is strictly better than the mean field bound and tends to the TAP approximation for small weights. For larger weights, however, the TAP approximation crosses the partition function and violates the bounding properties.

We saw that the third order bound gives an enormous improvement of the quality of the bound which gradually becomes less in the region of large weights and small thresholds, where almost all expansion based approximations are bad. We conclude that third order bounds are helpful in general, since they are always tighter than the mean field bound. In practice, however, third order bounds are most useful for problems just outside the scope of the mean field approximation.

Besides the partition function itself, we compared the approximated mean firing rates and correlations. There we saw an improvement of the approximation in the whole range, especially for small weights. A promising direction for further research is to combine the third order lower bound with upper bounds on Boltzmann machines and sigmoid belief networks to obtain better bounds on conditional probabilities.

Also in training the third order method is better than mean field. The differences are quite small, when we are simply interested in maximising the log likelihood. However, the hidden structure found by mean field is totally differ-

ent compared to the exact method, especially for the weights between hidden nodes. Third order, on the other hand, manages to find a highly comparable structure.

A full optimisation of the bound is computationally expensive (especially for the sigmoid belief networks). Therefore we suggest using the mean field solution for μ^i instead of solving equation 23. This avoids an $\mathcal{O}(N^4)$ for Boltzmann machines and a worst-case $\mathcal{O}(N^6)$ for belief networks to find the tightest bound, whereas the approximation is almost as good as in the fully optimised case. The computational complexity for Boltzmann machines is then $\mathcal{O}(N^3)$ to optimise and compute the bound; for sigmoid belief networks the worst-case complexity is $\mathcal{O}(N^4)$, although an $\mathcal{O}(N^3)$ is more likely for layered networks.

References

Ackley, D., Hinton, G., and Sejnowski, T. (1985). A learning algorithm for Boltzmann machines. *Cognitive Science*, 9:147–169.

Barber, D. and Wiegerinck, W. (1998). Tractable undirected approximations for graphical models. In Niklasson, L., Bodén, M., and Ziemke, T., editors, ICANN *98*, volume 1, pages 93–98, ISBN 3 540 76263 9. Springer-Verlag.

Jaakkola, T. and Jordan, M. (1996a). Recursive algorithms for approximating probabilities in graphical models. *MIT Comp. Cogn. Science Technical Report 9604*.

Jaakkola, T., Saul, L., and Jordan, M. (1996). Fast learning by bounding likelihoods in sigmoid-type belief networks. In *Advances in neural information processing systems 8*. MIT Press. 528–534.

Jaakkola, T. S. and Jordan, M. I. (1996b). Computing upper and lower bounds on likelihoods in intractable networks. In *Proceedings of the Twelfth Annual Conference on Uncertainty in Artificial Intelligence (UAI–96)*, pages 340–348, San Francisco, CA. Morgan Kaufmann Publishers.

Kappen, H. and Rodríguez, F. (1999). Boltzmann machine learning using mean field theory and linear response correction. In Kearns, M., Solla, S., and

Cohn, D., editors, *Advances in Neural Information Processing Systems*, volume 11, pages 280–286. MIT Press.

Leisink, M. and Kappen, H. (1999). Validity of TAP equations in neural networks. In ICANN *99*, volume 1, pages 425–430, ISBN 0 85296 721 7. Institution of Electrical Engineers, London.

Neal, R. (1992). Connectionist learning of belief networks. *Artificial intelligence*, 56:71–113.

Neal, R. and Hinton, G. (1998). A view of the EM algorithm that justifies incremental, sparse, and others variants. In Jordan, M., editor, *Learning in Graphical Models.*, volume 89, pages 355–368. Kluwer Academic Publishers.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems*, chapter 8.2.1, pages 387–390. Morgan Kaufmann, San Francisco.

Peterson, C. and Anderson, J. (1987). A mean field theory learning algorithm for neural networks. *Complex systems*, 1:995–1019.

Plefka, T. (1981). Convergence condition of the TAP equation for the infinite-ranged ising spin glass model. *J.Phys.A: Math.Gen.*, 15:1971–1978.

Saul, S., Jaakkola, T., and Jordan, M. (1996). Mean field theory for sigmoid belief networks. *Journal of Artificial Intelligence Research*, 4:61–76.

Sherrington, D. and Kirkpatrick, S. (1975). Solvable model of a spin-glass. *Physical Review Letters*, 35(26):1793–1796.

Thouless, D., Andersson, P., and Palmer, R. (1977). Solution of 'solvable model of a spin glass'. *Philisophical Magazine*, 35(3):593–601.

A Optimal solution for μ^0 and λ^0

The bound from equation 15 is valid for all values of μ^0, λ^0 and μ^i. To find the tightest bound, we set all the derivatives with respect to these variational

parameters to zero. For μ^0 and λ^0 it is possible to obtain explicit results. We derive (keep in mind that ΔE does depend on μ^0)

$$\frac{\partial B}{\partial \mu^0} = Z_\mu \left\{ - \langle \Delta E \rangle + \mathrm{e}^{\lambda^0} \left(\left(1 - \lambda^0\right) \langle \Delta E \rangle - \frac{\lambda^0}{2} \left\langle \Delta E^2 \right\rangle - \frac{1}{6} \left\langle \Delta E^3 \right\rangle \right) \right\} = 0 \tag{47}$$

$$\frac{\partial B}{\partial \lambda^0} = Z_\mu \mathrm{e}^{\lambda^0} \left(-\frac{\lambda^0}{2} \left\langle \Delta E^2 \right\rangle - \frac{1}{6} \left\langle \Delta E^3 \right\rangle \right) = 0 \tag{48}$$

Given equation 48 we can reduce equation 47 to

$$\langle \Delta E \rangle \left(1 - \mathrm{e}^{\lambda^0} \left(1 - \lambda^0\right) \right) = 0 \tag{49}$$

This leads to two possible solutions for μ^0 and λ^0

$$\begin{cases} \mu^0 = -\left\langle E + \mu^i s_i \right\rangle \\ \lambda^0 = -\frac{1}{3} \left\langle \Delta E^3 \right\rangle / \left\langle \Delta E^2 \right\rangle \end{cases} \tag{50a}$$

or

$$\begin{cases} \left\langle \left(E + \mu^i s_i + \mu^0 \right)^3 \right\rangle = 0 \\ \lambda^0 = 0 \end{cases} \tag{50b}$$

The implicit equation at the top of (50b) can easily be made explicit, since it is cubic in μ^0.

We analyse the stability of both solutions by investigating the Hessian

$$H\left(\mu^0, \lambda^0\right) = \begin{bmatrix} \dfrac{\partial^2 B}{\partial {\mu^0}^2} & \dfrac{\partial^2 B}{\partial \mu^0 \partial \lambda^0} \\ \dfrac{\partial^2 B}{\partial \lambda^0 \partial \mu^0} & \dfrac{\partial^2 B}{\partial {\lambda^0}^2} \end{bmatrix} \tag{51}$$

at both solution points. These can be written as

$$H\left(\mu^0, \lambda^0\right) = -Z_\mu \begin{bmatrix} \frac{1}{2} \left\langle \Delta E^2 \right\rangle + X & \frac{1}{2} \left\langle \Delta E^2 \right\rangle \\ \frac{1}{2} \left\langle \Delta E^2 \right\rangle & \frac{1}{2} \left\langle \Delta E^2 \right\rangle \end{bmatrix} \tag{52}$$

where

$$X = 1 - \mathrm{e}^{\lambda^0} \left(1 - \lambda^0\right) \tag{53}$$

which is zero for solution (50b) and positive if $\lambda \neq 0$. Therefore the Hessian is negative definite for solution (50a). Solution (50b), however, has a zero eigenvalue. It turns out that for small fluctuations ϵ in the direction corresponding to this eigenvalue, the bound varies proportional to $\langle \Delta E \rangle \, \epsilon^3$ and therefore this solution corresponds to a saddle point and should be discarded. Thus the optimal choice for μ^0 and λ^0 is given by equation 50a.

B Explicit expressions for $\left\langle \Delta E^2 \right\rangle$ and $\left\langle \Delta E^3 \right\rangle$

It is possible to compute the moments $\left\langle \Delta E^2 \right\rangle$ and $\left\langle \Delta E^3 \right\rangle$ under the factorised distribution $\mu\left(\vec{s}\right)$. Defining

$$m_i = \langle s_i \rangle = \tanh \mu^i \tag{54}$$

we derived

$$\frac{1}{2}\left\langle \Delta E^2 \right\rangle = \frac{1}{4}\theta^{ij\,2}\left(1-m_i^2\right)\left(1-m_j^2\right) + \frac{1}{2}\alpha^{i\,2}\left(1-m_i^2\right) \tag{55}$$

$$\begin{aligned} -\frac{1}{6}\left\langle \Delta E^3 \right\rangle = {} & \frac{1}{6}\theta^{ij}\theta^{jk}\theta^{ki}\left(1-m_i^2\right)\left(1-m_j^2\right)\left(1-m_k^2\right) \\ & + \frac{1}{3}\theta^{ij\,3}m_i m_j\left(1-m_i^2\right)\left(1-m_j^2\right) \\ & - \frac{1}{3}\alpha^{i\,3}m_i\left(1-m_i^2\right) + \frac{1}{2}\alpha^i\alpha^j\theta^{ij}\left(1-m_i^2\right)\left(1-m_j^2\right) \\ & - \alpha^i\theta^{ij\,2}m_i\left(1-m_i^2\right)\left(1-m_j^2\right) \end{aligned} \tag{56, 57}$$

where

$$\alpha^i = \theta^i + \theta^{ij}m_j - \mu^i \tag{58}$$

Note that $\alpha^i = 0$ is equivalent to the well known mean field equations. For the fully optimised third order bound, however, α^i differs from zero in general.