Journal of Machine Learning Research 11 (2010) 2855-2900 Submitted 6/09; Revised 8/10; Published 10/10

Expectation Truncation and the Benefits of Preselection In Training Generative Models

Jörg Lücke LUECKE@FIAS.UNI-FRANKFURT.DE
Frankfurt Institute for Advanced Studies (FIAS)
Goethe-Universität Frankfurt
Ruth-Moufang-Str. 1
60438 Frankfurt am Main
Germany

Julian Eggert JULIAN.EGGERT@HONDA-RI.DE
Honda Research Institute Europe
Carl-Legien-Str. 30
63073 Offenbach am Main
Germany

Editor: Aapo Hyvärinen

Abstract

We show how a preselection of hidden variables can be used to efficiently train generative models with binary hidden variables. The approach is based on Expectation Maximization (EM) and uses an efficiently computable approximation to the sufficient statistics of a given model. The computational cost to compute the sufficient statistics is strongly reduced by selecting, for each data point, the relevant hidden causes. The approximation is applicable to a wide range of generative models and provides an interpretation of the benefits of preselection in terms of a variational EM approximation. To empirically show that the method maximizes the data likelihood, it is applied to different types of generative models including: a version of non-negative matrix factorization (NMF), a model for non-linear component extraction (MCA), and a linear generative model similar to sparse coding. The derived algorithms are applied to both artificial and realistic data, and are compared to other models in the literature. We find that the training scheme can reduce computational costs by orders of magnitude and allows for a reliable extraction of hidden causes.

Keywords: maximum likelihood, deterministic approximations, variational EM, generative models, component extraction, multiple-cause models

1. Introduction

In many applications of artificial and biological systems, data interpretation is challenging because of noise, the complexity of the input and because of its ambiguity. Optimal inference based on probabilistic generative models is in general intractable in such situations because it involves the evaluation of all potential interpretations of the input. To approximate optimal inference, a fast initial stage of processing has therefore long since been suggested. In applications to visual data, a first processing stage can select candidate objects or components that are potential causes of the given input (see, e.g., Yuille and Kersten, 2006). Based on these candidates the meaning of a data point is subsequently inferred in a second recurrent stage. The strategy of candidate preselection has indeed been suggested and applied in different contexts (e.g., Körner et al., 1999; Lee and

Mumford, 2003; Yuille and Kersten, 2006; Westphal and Würtz, 2009) and prominent systems of feed-forward processing (such as Riesenhuber and Poggio, 1999, and van Rullen and Thorpe, 2001) are sometimes interpreted as sophisticated preprocessing stages for a subsequent recurrent stage. The general idea of candidate preselection followed by recurrent recognition will in this paper be formulated in terms of a variational approximation that allows for an efficient training of probabilistic generative models.

In Section 2 the approximation scheme will be introduced as an approximation to Expectation Maximization (EM). In Section 3 we systematically derive it as a variational EM approach. In Section 4 the training scheme is applied to a number of different generative models and a number of different data types. Section 5 discusses the features of the novel approach and the obtained results.

2. Expectation Maximization and Expectation Truncation

Our approach is based on Expectation Maximization (EM; Dempster et al., 1977) which is used to maximize the data likelihood under a given generative model:

$$\Theta^* \;=\; \mathrm{argmax}_\Theta\{L(\Theta)\} \quad \text{with} \quad L(\Theta) = \log\Big(p(\vec{y}^{(1)},\dots,\vec{y}^{(N)}\,|\,\Theta)\Big)\,, \tag{1}$$

where Θ are the parameters of a given generative model and where the N data points, $\{\vec{y}^{(n)}\}_{n=1,\dots,N}$, will be taken to be generated independently from a stationary process.

To find the parameters Θ^* at least approximately, we use the EM approach as it was formalized, for example, by Neal and Hinton (1998) and introduce the free-energy function $\mathcal{F}(q,\Theta)$ which is a function of Θ and an unknown distribution $q(\vec{s}^{(1)},\dots,\vec{s}^{(N)})$ over the hidden variables. $\mathcal{F}(q,\Theta)$ can be shown to be a lower bound of the likelihood evaluated at the same parameter values. For our purposes we assume independently generated data vectors $\vec{y}^{(n)}$ and use (without loss of generality) a distribution q which is factored over the data points, $q(\vec{s}^{(1)},\dots,\vec{s}^{(N)}) = \prod_n q^{(n)}(\vec{s}^{(n)};\Theta^{\mathrm{old}})$. Note that we take q to be parameterized by Θ^{old}. The free-energy can thus be written as:

$$\mathcal{F}(q,\Theta) = \sum_{n=1}^{N}\Big[\sum_{\vec{s}} q^{(n)}(\vec{s};\Theta^{\mathrm{old}})\;\Big[\log\Big(p(\vec{y}^{(n)}\,|\,\vec{s},\Theta)\Big) + \log\big(p(\vec{s}\,|\,\Theta)\big)\Big]\Big] + H(q)\,, \tag{2}$$

where $H(q) = -\sum_n\sum_{\vec{s}} q^{(n)}(\vec{s};\Theta^{\mathrm{old}})\log(q^{(n)}(\vec{s};\Theta^{\mathrm{old}}))$ is a function (the Shannon entropy) that is independent of Θ. The sum over all states of $\vec{s}$ becomes an integral if the values of $\vec{s}$ are continuous. In the EM scheme $\mathcal{F}(q,\Theta)$ is maximized alternately with respect to q in the E-step (while Θ is kept fixed) and with respect to Θ in the M-step (while q is kept fixed). It can be shown that the EM iterations increase the likelihood or keep it constant. In practical applications EM is found to increase the likelihood to (at least local) likelihood maxima.

The free-energy function (2) can be used to derive update rules for the parameters Θ of a given model. Such a derivation can in some cases be challenging but one usually arrives at expressions in which the new set of parameters Θ is a function of the old set Θ^{old} and the data $\{\vec{y}^{(n)}\}_{n=1,\dots,N}$. The update rules derived contain what is often referred to as the *sufficient statistics* of the model, that is, they contain expressions of the form

$$\langle g(\vec{s})\rangle_{q^{(n)}} = \sum_{\vec{s}} q^{(n)}(\vec{s};\Theta^{\mathrm{old}})\,g(\vec{s})\,, \tag{3}$$

where $g(\vec{s})$ is a function of the hidden variables. The functions $g(\vec{s})$ are often relatively simple, for example, $g(\vec{s}) = s_i$ or $g(\vec{s}) = s_i s_j$, but can for some models be more elaborate and may also include parameter dependencies. For an exact E-step in the EM scheme the functions $q^{(n)}(\vec{s};\Theta^{\mathrm{old}})$ are given by

$$q^{(n)}(\vec{s};\Theta^{\mathrm{old}}) = p(\vec{s}\,|\,\vec{y}^{(n)},\Theta^{\mathrm{old}}) = \frac{p(\vec{s},\vec{y}^{(n)}\,|\,\Theta^{\mathrm{old}})}{\sum_{\vec{s}\,'} p(\vec{s}\,',\vec{y}^{(n)}\,|\,\Theta^{\mathrm{old}})}, \tag{4}$$

where $p(\vec{s},\vec{y}^{(n)}\,|\,\Theta^{\mathrm{old}}) = p(\vec{s}\,|\,\Theta^{\mathrm{old}})\,p(\vec{y}^{(n)}\,|\,\vec{s},\Theta^{\mathrm{old}})$ with the latter distributions being defined by the used generative model. To train models with multiple causes, the computation of the exact sufficient statistics is usually avoided because it involves summing or integrating over a large space of hidden states in (3) and (4). To reduce computational costs, training schemes therefore use approximations to these intractable sums (or integrals) or approximations to the exact posterior $p(\vec{s}\,|\,\vec{y}^{(n)},\Theta^{\mathrm{old}})$. The approximation method discussed in this paper will be introduced as an approximation to the sufficient statistics.

Let us consider the sufficient statistics of a function g given by the combination of (3) and (4):

$$\langle g(\vec{s})\rangle_{q^{(n)}} = \frac{\sum_{\vec{s}} p(\vec{s},\vec{y}^{(n)}\,|\,\Theta^{\mathrm{old}})\; g(\vec{s})}{\sum_{\vec{s}\,'} p(\vec{s}\,',\vec{y}^{(n)}\,|\,\Theta^{\mathrm{old}})}. \tag{5}$$

Again, we have to sum over a very large space of hidden states. Let us for instance assume that we have already found the optimal or approximately optimal parameters $\Theta^{\mathrm{old}} \approx \Theta^{*}$, that is, let us assume that any given input vector is well represented by a distribution over hidden states. A given $\vec{y}^{(n)}$ is in this case usually well represented by a distribution over just a small set of hidden vectors. For the sums in (5) this means that just some summands contribute significantly while the others are negligible. Thus, if we could find the right summands for a given $\vec{y}^{(n)}$, we could expect a good approximation of $\langle g(\vec{s})\rangle_{q^{(n)}}$ in (5) without having to sum over the entire state space of $\vec{s}$.

More formally, if $\mathcal{K}_n$ denotes the set of all states that contain significant contributions to the sums in (5), it applies:

$$\langle g(\vec{s})\rangle_{q^{(n)}} \approx \frac{\sum_{\vec{s}\in\mathcal{K}_n} p(\vec{s},\vec{y}^{(n)}\,|\,\Theta^{\mathrm{old}})\; g(\vec{s})}{\sum_{\vec{s}\,'\in\mathcal{K}_n} p(\vec{s}\,',\vec{y}^{(n)}\,|\,\Theta^{\mathrm{old}})}. \tag{6}$$

A potential subset containing the relevant summands could be found by exploiting specific data properties. If the data was generated by few hidden units on average, for instance, most data points are well approximated by only considering combinations of few active causes. A subset for the approximation in (6) for binary causes s_j could thus be given by $\mathcal{K} = \{\vec{s}\mid \sum_j s_j \leq \gamma\}$, where γ is the maximal number of active causes. Such a choice can significantly reduce the number of states that have to be evaluated. Depending on γ the combinatorics can still be considerable, however, and still, just a few of the summands might contribute.

To further constrain the state space, let us suppose that we can in some way find functions $\mathcal{S}_h : \mathbb{R}^D \to \mathbb{R}$ that give estimates of how likely it is for the hidden causes $h = 1,\ldots,H$ to have contributed to the generation of a specific input $\vec{y}^{(n)}$. If we had such functions, we could approximate the sufficient statistics (5) by neglecting all causes that are unlikely to have contributed. In other words, we could just sum over a subset $\mathcal{K}_n \subseteq \mathcal{K}$ that contains the combinations of all hidden variables that are likely to have caused the input $\vec{y}^{(n)}$. To define such a set $\mathcal{K}_n$ more formally, consider the $h = 1, \dots, H$ values $\mathcal{S}_h(\vec{y}^{(n)})$ for a given data point $\vec{y}^{(n)}$. To select H' candidates, we define the index set I to contain those latent indices h with the H' largest values $\mathcal{S}_h(\vec{y}^{(n)})$. The set $\mathcal{K}_n$ is then given by:

$$\mathcal{K}_n = \{\vec{s} \mid \textstyle\sum_j s_j \leq \gamma \text{ and } (\forall i \notin I : s_i = 0)\}. \tag{7}$$

Equation 6 represents a good approximation if the contributions of the states not in $\mathcal{K}_n$ are indeed negligible compared to the contributions of the states in $\mathcal{K}_n$. Much depends, of course, on the functions $\mathcal{S}_h$. From these functions, which we will term *selection functions*, we demand the following properties: (A) they have to be efficiently computable and (B) they have to give high values for hidden variables that actually are responsible for a given $\vec{y}^{(n)}$. Note that for the selection functions it is merely important to select candidates that can *potentially* explain the input. Neither are their exact values used in (6) nor is it disadvantageous for the accuracy of the approximation if some candidates are selected that turn out to contribute very little. We will see examples of such selection functions for different generative models in Section 4. Before let us summarize the approximation discussed above in the form of the pseudo-code given in Algorithm 1. As the approximation scheme resides on a truncation of the sums in the expectation value computations in (5), we will refer to it as *Expectation Truncation* (ET).

Algorithm 1: Expectation Truncation - Pseudo Code

```
1 Choose approximation parameters $H'$ and $\gamma$ ($\gamma \leq H' \leq H$) and randomly initialize the
  parameters of the generative model.
2 while parameters have not converged do
3     for all data points $n = 1, \dots, N$ do
4         Compute the selection function value $\mathcal{S}_h$ for each $h = 1, \dots, H$ and determine the
          index set $I$ of the $H'$ hidden variables with the $H'$ highest values for $\mathcal{S}_h$.
5         Compute the set of binary vectors $\mathcal{K}_n = \{\vec{s} \mid \sum_j s_j \leq \gamma \text{ and } (\forall i \notin I : s_i = 0)\}$
6         Compute the approximate sufficient statistics (6).
7     Update the parameters in the M-step using the approximate sufficient statistics.
```

The two parameters H' and γ control the size of $\mathcal{K}_n$. H' determines how many candidates are selected and γ fixes the maximal number of non-zero hidden units s_h. For instance, if we choose $H' = 4$ and $\gamma = 2$, the summation over $\vec{s}$ considers four candidates of which either none, one, or two are simultaneously active (compare Figure 2). The size of $\mathcal{K}_n$ is thus given by $\sum_{\gamma'=0}^{\gamma} \binom{H'}{\gamma'}$. The approximation's accuracy increases with increasing values of H' and γ but its computational demand increases as well. For the highest possible values, $H' = \gamma = H$, we drop, for any selection function $\mathcal{S}_h$, back to the case of the exact sufficient statistics (5).

Note that while the approximation scheme presumably results in good approximations for many data points it can be expected to give poor results for data points generated by more than γ hidden causes (i.e., for data points not in $\mathcal{K}$). To avoid learning from such data points with inappropriately estimated sufficient statistics let us again assume that we have already found close to optimal parameters Θ^*. In such a situation and for any data point $\vec{y}^{(n)}$ generated by less or equal γ hidden causes we get:

$$p(\vec{y}^{(n)}\,|\,\Theta^*) = \sum_{\vec{s}} p(\vec{s},\vec{y}^{(n)}\,|\,\Theta^*) \approx \sum_{\vec{s}\in\mathcal{K}_n} p(\vec{s},\vec{y}^{(n)}\,|\,\Theta^*)\,, \tag{8}$$

where we have assumed appropriate selection functions and relatively low noise levels. In the same situation but for data points $\vec{y}^{(n)}$ which were generated by more than γ hidden causes we obtain:

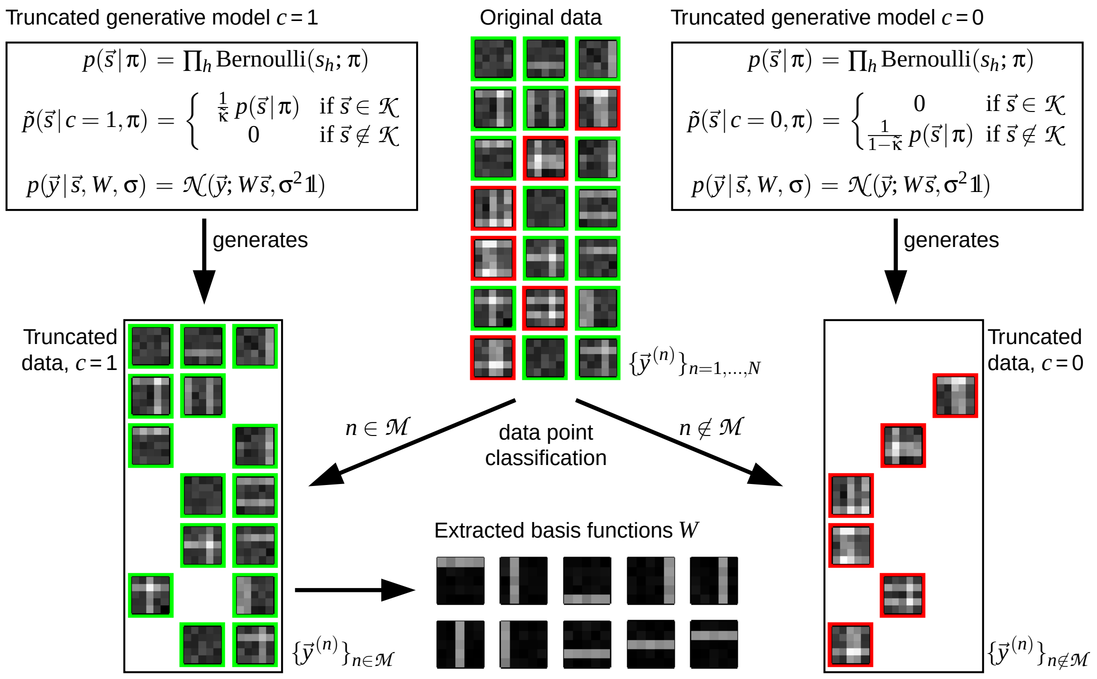
$$p(\vec{y}^{(n)}\,|\,\Theta^*) = \sum_{\vec{s}} p(\vec{s},\vec{y}^{(n)}\,|\,\Theta^*) \approx \sum_{\vec{s}\notin\mathcal{K}_n} p(\vec{s},\vec{y}^{(n)}\,|\,\Theta^*) \;\;\Rightarrow\;\; \sum_{\vec{s}\in\mathcal{K}_n} p(\vec{s},\vec{y}^{(n)}\,|\,\Theta^*) \approx 0\,. \tag{9}$$

The values of the sums $\sum_{\vec{s}\in\mathcal{K}_n} p(\vec{s},\vec{y}^{(n)}\,|\,\Theta^*)$ for the different data points can thus serve as indicators for finding data points that were presumably generated by less than γ causes. For learning we aim to include only those data points that are approximated well. We therefore define a set $\mathcal{M}$ of the $N^{\mathrm{cut}} \leq N$ data points with largest sums. In the beginning of learning, the estimation of such data points is too imprecise, however. We therefore start by taking all data points into account $N^{\mathrm{cut}} = N$ and decrease N^{cut} to values close to $N^{\leq\gamma}$ within the last third of the iterations. $N^{\leq\gamma}$ is hereby the expected number of data points generated by less or equal γ causes. For a given generative model, this number can usually be computed tractably. The number of data points generated by $\leq\gamma$ causes can, for a given data set, be smaller than $N^{\leq\gamma}$ because of finitely many data points. It can therefore be beneficial to finally use an N^{cut} slightly smaller than $N^{\leq\gamma}$. This potentially avoids the consideration of data points that are not well approximated. In numerical experiments in Section 4 we will therefore use final values of $N^{\mathrm{cut}} = 0.9\,N^{\leq\gamma}$ although $N^{\mathrm{cut}} = N^{\leq\gamma}$ gives similar results especially for large N.

Considering Algorithm 1 what is still left to specify are concrete expressions for the selection functions $\mathcal{S}_h$ and expressions for parameter update rules (M-step equations). These equations do, however, depend on the particular generative model the method is applied to. We will therefore discuss selection functions and update rules individually for the different generative models we investigate in Section 4. Given selection functions and update rules, Algorithm 1 describes an approximation scheme applicable to generative models with binary hidden variables. The scheme has been introduced and defined as an approximation to the sufficient statistics (5). In the next section it will be systematically derived as a variational EM approach. The assumptions used for the approximation will thus become explicit, allowing a generalization of the scheme and a specification of its potential limitations. The computational complexity of the method will be discussed in Appendix C.

3. Expectation Truncation and Variational EM

In this section we will show that Expectation Truncation corresponds to a variational EM approximation. The approximation, as introduced in Section 2, consists of two parts: (A) an approximation to the sufficient statistics in Equation 6, and (B) a selection of data points that are well approximated by Equation 6. Both of these parts can be formulated as variational EM steps. For the derivation we will start with (B), that is, with the selection of data points. The selection will take the form of a classification of data points into two classes: one class that contains the data points that can be well approximated, and another class that contains the remaining points. The corresponding variational step will be referred to as the first variational step (Section 3.1). The step (A) corresponding to approximation (6) will only be the second variational step (Section 3.2). Both steps combine to form the approximation scheme of Expectation Truncation. The scheme is compactly summarized in Section 3.3, and the basic procedural steps are listed in Algorithm 2 which represents a generalization of Algorithm 1.



Truncated generative model $c=1$
$p(\vec{s}\,|\,\pi) = \prod_h \mathrm{Bernoulli}(s_h;\pi)$
$\tilde{p}(\vec{s}\,|\,c=1,\pi) = \begin{cases} \frac{1}{\kappa}\,p(\vec{s}\,|\,\pi) & \text{if } \vec{s}\in\mathcal{K} \\ 0 & \text{if } \vec{s}\notin\mathcal{K}\end{cases}$
$p(\vec{y}\,|\,\vec{s},W,\sigma) = \mathcal{N}(\vec{y};W\vec{s},\sigma^2\mathbb{1})$
Original data
Truncated generative model $c=0$
$p(\vec{s}\,|\,\pi) = \prod_h \mathrm{Bernoulli}(s_h;\pi)$
$\tilde{p}(\vec{s}\,|\,c=0,\pi) = \begin{cases} 0 & \text{if } \vec{s}\in\mathcal{K} \\ \frac{1}{1-\kappa}\,p(\vec{s}\,|\,\pi) & \text{if } \vec{s}\notin\mathcal{K}\end{cases}$
$p(\vec{y}\,|\,\vec{s},W,\sigma) = \mathcal{N}(\vec{y};W\vec{s},\sigma^2\mathbb{1})$
generates
generates
Truncated data, $c=1$
$\{\vec{y}^{(n)}\}_{n=1,\ldots,N}$
Truncated data, $c=0$
$n\in\mathcal{M}$
data point classification
$n\notin\mathcal{M}$
Extracted basis functions W
$\{\vec{y}^{(n)}\}_{n\in\mathcal{M}}$
$\{\vec{y}^{(n)}\}_{n\notin\mathcal{M}}$

Figure 1: First variational approximation: data classification. The figure shows Expectation Truncation (without preselection) for a concrete generative model. In this example, the generative model generates data by linearly superimposing basis functions in the form of horizontal and vertical bars. Data generated by the original model contains up to ten bars chosen with a Bernoulli prior (example data points are shown in the center). Data generated by the truncated generative model with $c=1$ contains up to two bars (we set $\mathcal{K}=\{\vec{s}\mid\sum_j s_j\leq\gamma\}$ with $\gamma=2$). Data generated by the truncated generative model with $c=0$ contains at least three bars. If we train the truncated generative model with $c=1$ on data from which data points with $\sum_j s_j>\gamma$ were removed, we can expect to approximately recover the true generating basis functions W of the original model.

3.1 First Variational Approximation: Data Classification

As in Section 2, let us consider a generative model with a set of hidden variables denoted by $\vec{s}$, a set of observed variables denoted by $\vec{y}$, and a set of parameters denoted by Θ. Let us denote the prior distribution of the model by the (not further specified) function $p(\vec{s}\,|\,\Theta)$, and the noise distribution by the (not further specified) function $p(\vec{y}\,|\,\vec{s},\Theta)$. To distinguish this generative model from models introduced later, it will from now on be referred to as the *original* generative model.

We will formalize the classification of data points by introducing two new generative models defined based on the original model. The two models will correspond to two classes of data points: one class of those points that can be well approximated, and one class of points that can not. Let $\mathcal{K}$ be a subset of the space of all possible values of $\vec{s}$. Given such a set, we define the two generative models by introducing two new prior distributions that are based on the original prior:

$$p(\vec{s}\,|\,c=1,\Theta) \;=\; \begin{cases} \frac{1}{\tilde{\kappa}}\,p(\vec{s}\,|\,\Theta) & \text{if } \vec{s}\in\mathcal{K} \\ 0 & \text{if } \vec{s}\notin\mathcal{K} \end{cases} \text{ and } \; p(\vec{s}\,|\,c=0,\Theta) \;=\; \begin{cases} 0 & \text{if } \vec{s}\in\mathcal{K} \\ \frac{1}{1-\tilde{\kappa}}\,p(\vec{s}\,|\,\Theta) & \text{if } \vec{s}\notin\mathcal{K} \end{cases} \tag{10}$$

where $\tilde{\kappa} = \sum_{\vec{s}\in\mathcal{K}} p(\vec{s}\,|\,\Theta)$. We take the noise distribution $p(\vec{y}\,|\,\vec{s},\Theta)$ of the new models to be identical to the original noise distribution. We will refer to the new generative models as *truncated* models because their prior distributions are truncated to be zero outside of specific subsets. Note that the generation of data according to the truncated model with $c=1$ corresponds to generating data according to the original model while only accepting data points generated by $\vec{s}\in\mathcal{K}$. Analogously, generating data according to the truncated model with $c=0$ is equivalent to generating data according to the original model while accepting only data points generated by $\vec{s}\notin\mathcal{K}$. Figure 1 shows the truncated generative models and data they generate for a concrete example (a model that linearly combines bar-like generative fields; compare Section 4.1). For the example, $\mathcal{K}$ is the set of binary states $\vec{s}$ with less or equal γ non-zero entries. In general, $\mathcal{K}$ can be any subset, however.

Let us now mix the two truncated models in Equation 10 by introducing $c\in\{0,1\}$ as additional hidden variable and by drawing $c=1$ with probability κ. The prior distribution of this mixed model is thus given by:

$$p(c\,|\,\kappa) \;=\; \kappa^{c}\,(1-\kappa)^{1-c}\,, \tag{11}$$

$$p(\vec{s}\,|\,c,\Theta) \;=\; \Big(\frac{c}{\tilde{\kappa}}\,\delta(\vec{s}\in\mathcal{K}) + \frac{1-c}{1-\tilde{\kappa}}\,\delta(\vec{s}\notin\mathcal{K})\Big)\,p(\vec{s}\,|\,\Theta)\,. \tag{12}$$

where we have introduced $\delta(\vec{s}\in\mathcal{K})=1$ if $\vec{s}\in\mathcal{K}$ and zero otherwise, and $\delta(\vec{s}\notin\mathcal{K})=1$ if $\vec{s}\notin\mathcal{K}$ and zero otherwise. We will refer to this model as the *mixed* generative model. Note that the mixed model is identical to the original generative model if we choose $\kappa=\tilde{\kappa}=\sum_{\vec{s}\in\mathcal{K}} p(\vec{s}\,|\,\Theta)$ as mixing proportion. The mixed model thus contains the original model as a special case.

Now, consider a set of N data points $\{\vec{y}^{(n)}\}_{n=1,\ldots,N}$ generated according to the original generative model. Let us maximize the likelihood of the data under the mixed model (11) and (12). If we use EM for optimization (compare Section 2), we obtain the free-energy

$$\begin{aligned} \tilde{\mathcal{F}}(q,\Theta,\kappa) \;&=\; \sum_{n}\sum_{c} q^{(n)}(c;\Theta^{\text{old}})\,\log\big(p(\vec{y}^{(n)}\,|\,c,\Theta)\big) \\ &\quad + \log(\kappa)\sum_{n} q^{(n)}(c=1;\Theta^{\text{old}}) + \log(1-\kappa)\sum_{n} q^{(n)}(c=0;\Theta^{\text{old}}) + H(q)\,, \end{aligned} \tag{13}$$

where $H(q)$ is the entropy w.r.t. $q^{(n)}(c;\Theta^{\text{old}})$ (summed over all n and c). The free-energy (13) can be optimized iteratively by maximizing q in the E-step and (Θ,κ) in the M-step. For the E-step, choosing the exact posterior, $q^{(n)}(c\,;\Theta^{\text{old}}) = p(c\,|\,\vec{y}^{(n)},\Theta^{\text{old}})$, represents the optimal choice. Unfortunately, it is computationally intractable in general because

$$p(c=1\,|\,\vec{y}^{(n)},\Theta^{\text{old}}) = \frac{\sum_{\vec{s}\in\mathcal{K}} p(\vec{y}^{(n)},\vec{s}\,|\,\Theta)}{\sum_{\vec{s}} p(\vec{y}^{(n)},\vec{s}\,|\,\Theta)} \tag{14}$$

requires a summation over the entire state space of $\vec{s}$ (similarly for $c=0$). We thus choose a variational approximation to the true posterior by setting $q^{(n)}(c\,;\Theta^{\text{old}})$ to zero or one. This approximation reduces the free-energy (13) to:

$$\tilde{\mathcal{F}}(q,\Theta) = \overbrace{\sum_{n\in\mathcal{M}} \log\left(p(\vec{y}^{(n)}\,|\,c=1,\Theta)\right)}^{\approx L_1(\Theta)} + \overbrace{\sum_{n\notin\mathcal{M}} \log\left(p(\vec{y}^{(n)}\,|\,c=0,\Theta)\right)}^{\approx L_0(\Theta)} + \log(\kappa)|\mathcal{M}| + \log(1-\kappa)(N-|\mathcal{M}|) + H(q)\,. \quad (15)$$

where $\mathcal{M} = \{n\,|\,q^{(n)}(c=1\,;\Theta^{\mathrm{old}})=1\}$. Note that κ can be optimized independently of Θ because the first two summands in (15) only depend on Θ. As we also know its final optimal value ($\kappa=\tilde{\kappa}$), we will treat the mixing proportion as implicitly known. κ can thus be omitted as a parameter of the free-energy (15) and will not play a role for our further considerations.

For the data set $\mathcal{M}$ note that the best choice for $q^{(n)}(c\,;\Theta^{\mathrm{old}})$ under the constraint $q^{(n)}(c\,;\Theta^{\mathrm{old}})\in\{0,1\}$ is given by the assignment $q^{(n)}(c=1\,;\Theta^{\mathrm{old}})=1$ if $\vec{y}^{(n)}$ was generated by class $c=1$ (and zero otherwise). This would amount to setting $\mathcal{M}$ to

$$\mathcal{M}^{\mathrm{opt}} = \{n\,|\,\vec{y}^{(n)} \text{ generated by class } c=1\}\,. \quad (16)$$

In general this best choice can not be computed exactly. We can, however, derive tractable approximations to $\mathcal{M}^{\mathrm{opt}}$. Choosing a set $\mathcal{M}$ is equivalent to choosing a distribution $q^{(n)}(c\,;\Theta^{\mathrm{old}})$ with binary values (as an approximation to Equation 14). Any choice of $\mathcal{M}$ thus represents a variational approximation. In Appendix A.2 it is shown that one such approximation is obtained via sorting according to the denominator values in Equation 6. The selection of N^{cut} data points obtained in this way is thus equivalent to a variational E-step.

An interesting aspect of Equation 15 is that the first two summands take the form of two log-likelihoods. The first summand is the likelihood $L_1(\Theta)$ of the truncated generative model with $c=1$, and the second is the likelihood $L_0(\Theta)$ of the model with $c=0$. If $\mathcal{M}=\mathcal{M}^{\mathrm{opt}}$, both likelihoods are evaluated on the set of data points they can generate (see Figure 1). Considering these properties of Equation 15 the question arises how the maximum of $\tilde{\mathcal{F}}(q,\Theta)$ and the maxima of $L_1(\Theta)$ and $L_0(\Theta)$ are related. It could, for instance, be asked if all three functions have a maximum for the same parameter values. From the structure of the equation this can not be concluded, and, indeed, the question must be answered negatively because it can be shown that in general the maxima do not coincide. However, under assumptions that are usually fulfilled at least approximately, we can show that any global maximum of $\tilde{\mathcal{F}}(q,\Theta)$ is an approximate global maximum of $L_1(\Theta)$ and of $L_0(\Theta)$. A necessary condition for an approximate global maximum of $L(\Theta)$ (the likelihood of the original model) is thus a global likelihood maximum of $L_1(\Theta)$ (or of $L_0(\Theta)$). The technical derivation of this observation is given in Appendix A.1.

Note that, intuitively, it makes sense that the maximization of the likelihood $L_1(\Theta)$ results in parameters that can approximately maximize $L(\Theta)$. To see this consider the example of Figure 1. If the truncated generative model with $c=1$ is optimized on the truncated data class $c=1$, the displayed generative fields W are learned. These parameter values can be expected to also result in close to maximum likelihood values for the generative model with $c=0$ on data class $c=0$, and also correspond to approximately optimal likelihood values of the original generative model on the original data. The approximation improves with increasingly many data points. For this example, all parameters to approximately maximize $L(\Theta)$ can be recovered based on the necessary condition of maximizing $L_1(\Theta)$. The example of Figure 1 also demonstrates that the first variational step already significantly reduces computational costs. The truncated model with $c=1$ only requires the evaluation of 56 states per data point instead of $2^{10}=1024$ evaluations for the original model.

3.2 Second Variational Approximation: Preselection

Starting point for the second variational approximation will be the likelihood $L_1(\Theta)$ of the truncated generative model $c=1$. We have seen in the previous section (and Appendix A.1) that a global maximum in $L_1(\Theta)$ is a necessary condition for an approximate global maximum of the likelihood $L(\Theta)$ of the original model. To find the maximum of $L_1(\Theta)$ we optimize the lower bound $\mathcal{F}_1(q,\Theta)$ given by:

$$\mathcal{F}_1(q,\Theta) \;=\; \overbrace{\sum_{n\in\mathcal{M}}\sum_{\vec{s}\in\mathcal{K}} q^{(n)}(\vec{s};\Theta^{\mathrm{old}})\;\log\Big(p(\vec{y}^{(n)}\,|\,\vec{s},\Theta)\,\frac{1}{\tilde{\kappa}}\,p(\vec{s}\,|\,\Theta)\Big)}^{Q_1(q,\Theta)} \;+H(q)\,, \tag{17}$$

with $\sum_{\vec{s}\in\mathcal{K}} q^{(n)}(\vec{s};\Theta^{\mathrm{old}})=1$. $\mathcal{F}_1(q,\Theta)$ is derived by a variational approximation, this time w.r.t. the hidden variables $\vec{s}$. The free-energy equals the likelihood $L_1(\Theta)$ after each E-step if the distributions $q^{(n)}(\vec{s};\Theta^{\mathrm{old}})$ are given by:

$$q^{(n)}(\vec{s};\Theta^{\mathrm{old}}) \;=\; p(\vec{s}\,|\,\vec{y}^{(n)},c=1,\Theta^{\mathrm{old}}) \;=\; \frac{p(\vec{s}\,|\,\vec{y}^{(n)},\Theta^{\mathrm{old}})}{\sum_{\vec{s}\,'\in\mathcal{K}} p(\vec{s}\,'\,|\,\vec{y}^{(n)},\Theta^{\mathrm{old}})}\delta(\vec{s}\in\mathcal{K}). \tag{18}$$

M-step rules can be derived by setting the derivatives of $\mathcal{F}_1(q,\Theta)$ w.r.t. all parameters to zero. As the entropy term in (17) is independent of Θ if q is held fixed, we obtain

$$\frac{d}{d\Theta}\mathcal{F}_1(q,\Theta)=\frac{d}{d\Theta}Q_1(q,\Theta)=0 \tag{19}$$

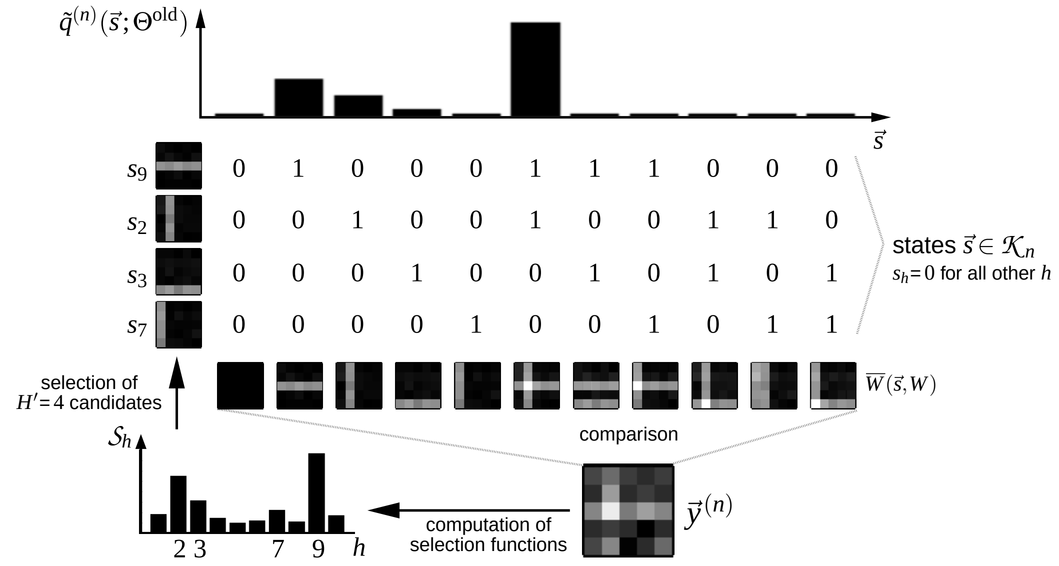
as necessary condition. The derivative $\frac{d}{d\Theta}$ hereby stands for derivatives w.r.t. all the individual parameters.

Based on condition (19) we can now introduce candidate preselection as a variational approximation. As described in Section 2, preselection amounts to selecting, for a given $\vec{y}^{(n)}$, a subset $\mathcal{K}_n$ of the state space. Section 2 gives an example of how to define the set $\mathcal{K}$ and how to construct subsets $\mathcal{K}_n$ using selection functions. Figure 2 shows a concrete example of how a set $\mathcal{K}_n$ is constructed using selection function values $\mathcal{S}_h(\vec{y}^{(n)})$. In Section 4 and Appendix B different instances of selection functions can be found. More generally, we here require from the sets $\mathcal{K}_n$ that for all data points generated by $\vec{s}\in\mathcal{K}$, they finally contain most of the posterior mass in $\mathcal{K}$. If this applies, we obtain an approximation to the posterior $q^{(n)}$ in (18) given by:

$$\tilde{q}^{(n)}(\vec{s};\Theta^{\mathrm{old}}) \;=\; \frac{p(\vec{s}\,|\,\vec{y}^{(n)},\Theta^{\mathrm{old}})}{\sum_{\vec{s}\,'\in\mathcal{K}_n} p(\vec{s}\,'\,|\,\vec{y}^{(n)},\Theta^{\mathrm{old}})}\,\delta(\vec{s}\in\mathcal{K}_n) \;=\; \frac{p(\vec{s},\vec{y}^{(n)}\,|\,\Theta^{\mathrm{old}})}{\sum_{\vec{s}\,'\in\mathcal{K}_n} p(\vec{s}\,',\vec{y}^{(n)}\,|\,\Theta^{\mathrm{old}})}\,\delta(\vec{s}\in\mathcal{K}_n). \tag{20}$$

Note that $\tilde{q}^{(n)}(\vec{s};\Theta^{\mathrm{old}})$ sums to one in $\mathcal{K}$ as $\mathcal{K}_n\subseteq\mathcal{K}$. It thus fulfils the condition on $q^{(n)}$ required for (17). If preselection finds, at least finally, appropriate sets $\mathcal{K}_n$, we obtain with (20) the necessary condition: $\frac{d}{d\Theta}Q_1(\tilde{q},\Theta)\approx\frac{d}{d\Theta}Q_1(q,\Theta)=0$. Parameter update rules derived from $\frac{d}{d\Theta}Q_1(\tilde{q},\Theta)=0$ can therefore be expected to (at least approximately) optimize the free-energy (15) and thus $L_1(\Theta)$. The update rules derived will contain expectation values (the sufficient statistics) of the form $\langle g(\vec{s})\rangle_{\tilde{q}^{(n)}}$. If we use (20) for these expectations we obtain:

$$\langle g(\vec{s})\rangle_{\tilde{q}^{(n)}} \;=\; \sum_{\vec{s}}\tilde{q}^{(n)}(\vec{s};\Theta^{\mathrm{old}})\,g(\vec{s}) \;=\; \frac{\displaystyle\sum_{\vec{s}\in\mathcal{K}_n} p(\vec{s},\vec{y}^{(n)}\,|\,\Theta^{\mathrm{old}})\;g(\vec{s})}{\displaystyle\sum_{\vec{s}\,'\in\mathcal{K}_n} p(\vec{s}\,',\vec{y}^{(n)}\,|\,\Theta^{\mathrm{old}})}\,,$$



$\tilde{q}^{(n)}(\vec{s};\Theta^{\text{old}})$
$\vec{s}$
s_9 0 1 0 0 0 1 1 1 0 0 0
s_2 0 0 1 0 0 1 0 0 1 1 0
s_3 0 0 0 1 0 0 1 0 1 0 1
s_7 0 0 0 0 1 0 0 1 0 1 1
states $\vec{s} \in \mathcal{K}_n$
$s_h = 0$ for all other h
$\overline{W}(\vec{s}, W)$
selection of $H' = 4$ candidates
comparison
$\mathcal{S}_h$
2 3 7 9 h
computation of selection functions
$\vec{y}^{(n)}$

Figure 2: Second variational approximation: preselection. The figure illustrates how the variational approximation $\tilde{q}^{(n)}(\vec{s};\Theta^{\text{old}})$ in Equation 20 is computed. The selection of hidden states $\vec{s} \in \mathcal{K}_n$ and the computation of $\tilde{q}^{(n)}(\vec{s};\Theta^{\text{old}})$ are shown for the example of Figure 1 close to the optimal parameters W. Given the data point $\vec{y}^{(n)}$ a selection function value $\mathcal{S}_h$ for each hidden variable h is computed. The H' largest values are selected ($H' = 4$ for this example). The approximation $\tilde{q}^{(n)}(\vec{s};\Theta^{\text{old}})$ is then computed based on the combinatorics of these H' candidates (with $\sum_j s_j \leq \gamma = 2$). For the displayed data point and parameters all values of $\tilde{q}^{(n)}(\vec{s};\Theta^{\text{old}})$ except for one lie close to zero. For visualization purposes these values have been increase in the figure.

that is, precisely expression (6) in Section 2. Expectation Truncation, introduced as an approximation to Equation 5, can thus been derived as a variational approximation. Importantly, this approximation is tractable if $\mathcal{K}_n$ is small. The computational gain of preselection compared to an approximation without preselection is reflected by the reduced size of $\mathcal{K}_n$ compared to $\mathcal{K}$ (see Figure 2 for an example and Appendix C for a detailed complexity analysis).

3.3 Summary and Numerical Controls

We have seen that the approximation procedure introduced in Section 2 can be derived as a variational EM approach. This approach consists of two variational approximation steps: First, an approximation that assigns the data points to two classes (Section 3.1). Second, a variational step that approximates the posterior (18) by an approximate posterior (20) defined through preselection (Section 3.2). Although the derivation of ET as a variational approach requires in parts rather technical steps, the final result is intuitive (see Figure 1 and Figure 2) and can be stated very compactly. Algorithm 2 summarizes all required steps of the approximation scheme.

Note that also with preselection, the ET approximation still requires a summation over $\mathcal{K}$, namely $\sum_{\vec{s}\in\mathcal{K}} p(\vec{s}\,|\,\Theta)$. This sum has to be computed to determine N^{cut}, $N^{\text{cut}} = N\sum_{\vec{s}\in\mathcal{K}} p(\vec{s}\,|\,\Theta)$,

```
Algorithm 2: Expectation Truncation
    Preselection:          select a state space volume $\mathcal{K}_n$ for each data point $\vec{y}^{(n)}$
    Data classification:   find a data set $\mathcal{M}$ that approximates $\mathcal{M}^{\rm opt}$ in Equation 16
    E-step:                compute $\tilde{q}^{(n)}(\vec{s};\Theta^{\rm old}) = \frac{p(\vec{s},\vec{y}^{(n)}\,|\,\Theta^{\rm old})}{\sum_{\vec{s}\in\mathcal{K}_n} p(\vec{s},\vec{y}^{(n)}\,|\,\Theta^{\rm old})}$ for all $\vec{y}^{(n)}$ and $\vec{s}\in\mathcal{K}_n$
    M-step:                find parameters $\Theta$ such that
                           $\frac{d}{d\Theta}\sum_{n\in\mathcal{M}}\sum_{\vec{s}\in\mathcal{K}_n}\tilde{q}^{(n)}(\vec{s};\Theta^{\rm old})\,\log\Big(p(\vec{y}^{(n)}\,|\,\vec{s},\Theta)\,\frac{p(\vec{s}\,|\,\Theta)}{\sum_{\vec{s}\,'\in\mathcal{K}} p(\vec{s}\,'\,|\,\Theta)}\Big) = 0$
```

and it appears in the M-step equation (Algorithm 2). Because of symmetries in the usual priors for generative models, this sum can, however, be computed without summing over all $\vec{s}$ explicitly (an example is given in the next section). Even if symmetries can not be exploited, note that the sum over $\vec{s} \in \mathcal{K}$ has to be computed at most once per EM iteration and not once per data point and per EM iteration (as it is the case for the sums over $\mathcal{K}_n$).

Algorithm 2 summarizes ET as a variational approximation and represents a generalization of Algorithm 1. Algorithm 1 in Section 2 contains, for example, one specific way of how to select an appropriate set $\mathcal{K}$ and appropriate sets $\mathcal{K}_n$: we defined $\mathcal{K}$ based on sparseness and selected $\mathcal{K}_n$ using selection functions $\mathcal{S}_h$. In the variational derivation of ET we, however, only specified the properties required from $\mathcal{K}$ and $\mathcal{K}_n$. In general, $\mathcal{K}$ does not have to be defined based on a sparseness assumption and there are potentially alternative ways to define the sets $\mathcal{K}_n$. Importantly, the variational derivation of ET allows for a comparison with other variational approaches. We can thus observe that ET is qualitatively different from the standard variational approach. Concrete instances of variational EM usually approximate the exact posteriors by fully or partly factored distributions over the hidden variables (compare Jordan et al., 1999; Jaakkola, 2000; MacKay, 2003; Bishop, 2006). Such approximations become the more severe the stronger dependencies between the hidden variables in the posterior are. In the derivation of the ET approximation, no independence assumption for the posterior has been used. Strong dependencies are therefore not expected to negatively affect the approximation quality of ET. On the other hand, the ET approximations can get more severe if the approximate classification of data points becomes imprecise (first variational step), if the preselection step does not include the relevant candidates (second variational step), or if too few data points are used.

ET has in common with all variational EM approaches that there is no guarantee for the likelihood to finally increase to values close to the global optimum. This can in general be due to the approximations being too severe or due to many local optima in the likelihood landscape. Variational approximations therefore have to be verified numerically. In the following section, ET will be evaluated based on different generative models and different data sets. During likelihood maximization we will monitor different values relevant for the approximation. We will thus control if the likelihood is indeed increased during learning and, given ground-truth, if it approaches values close to the global likelihood maximum. Ground-truth data will also allow us to quantify how well the generating parameters are recovered by the derived algorithm. Furthermore, we will monitor values that give evidence about the quality of the specific approximations used by ET. We will thus monitor the quality of the data point classification (first variational approximation) and the quality of the approximation by preselection (second variational approximation). For the first approximation we compare the obtained classification with ground-truth of all data points. For the second approximation, we will monitor the differences between the exact posterior $p(\vec{s}\,|\,\vec{y}^{(n)},\Theta)$ and the approximate posterior $\tilde{q}^{(n)}(\vec{s};\Theta)$ in (20). If we evaluate the differences between these distributions using the Kullback-Leibler divergence, we obtain:

$$D_{KL}(\tilde{q}^{(n)},p) = -\log(Q^{(n)}) \quad \text{with} \quad Q^{(n)} = \frac{\sum_{\vec{s}\in\mathcal{K}_n} p(\vec{s},\vec{y}^{(n)}\,|\,\Theta)}{\sum_{\vec{s}'} p(\vec{s}',\vec{y}^{(n)}\,|\,\Theta)}\,. \tag{21}$$

The preselection approximation has a high quality if for the data points in $\mathcal{M}$ the values $D_{KL}(\tilde{q}^{(n)},p)$ are close to zero. For data points not in $\mathcal{M}$ the differences between the distributions should be large. For practical reasons, we have introduced the quality values $Q^{(n)}$ in (21). The values $Q^{(n)}$ measure the percentage of the posterior probability mass concentrated in $\mathcal{K}_n$. In terms of these values the approximation quality is high if $Q^{(n)}$ is close to one for data points in $\mathcal{M}$, and close to zero for data points not in $\mathcal{M}$.

4. Training Generative Models

The approximation scheme defined and discussed in the previous sections is so far independent of the specific choice of a generative model except for the assumption of binary hidden variables. To demonstrate the applicability of the method and to investigate its properties, in this section we will apply it to a number of different generative models. The investigated models are all multiple-cause models that require tractable approximations. The three major classes investigated here are non-negative matrix factorization (NMF; Section 4.1), maximal causes analysis (MCA; Section 4.2), and a sparse-coding-like model termed LinCA (Section 4.3). For all these models we will assume independent hidden variables distributed according to a Bernoulli prior:

$$p(\vec{s}\,|\,\pi) = \prod_{h=1}^{H} p(s_h\,|\,\pi), \quad p(s_h\,|\,\pi) = \pi^{s_h}\,(1-\pi)^{1-s_h}\,, \tag{22}$$

where $\pi \in [0,1]$ parameterizes the sparseness of the distribution. For binary hidden variables the Bernoulli prior represents the most straightforward choice (compare, e.g., Berkes et al., 2009; Lücke and Sahani, 2008). Given the prior (22) the expected number of data points generated by less or equal γ causes is given by:

$$N^{\leq\gamma} = N \sum_{\vec{s},\,|\vec{s}|\leq\gamma} p(\vec{s}\,|\,\pi) = N \sum_{\gamma'=0}^{\gamma} \binom{H}{\gamma'} \pi^{\gamma'}\,(1-\pi)^{H-\gamma'}\,. \tag{23}$$

$N^{\leq\gamma}$ is required to find the set of N^{cut} data points $\mathcal{M}$ considered for a parameter update, and Equation 23 shows that this number is tractably computable. For all generative models we will use ET as described in Section 2 and Algorithm 1. That is, we will use a set $\mathcal{K}$ that constrains the number of simultaneously active causes, and use selection functions $\mathcal{S}_h$ to obtain sets $\mathcal{K}_n$.

4.1 Non-negative Matrix Factorization (NMF)

The first generative model considered uses prior (22) and combines the generative fields $\vec{W}_h = (W_{1h},\ldots,W_{Dh})^T$ of all latents with $s_h = 1$ linearly. We use a Gaussian noise model such that the probability of a data vector $\vec{y}$ given $\vec{s}$ is defined by:

$$p(\vec{y}\,|\,\vec{s},\Theta) = \prod_{d=1}^{D} p(y_d\,|\,\overline{W}_d(\vec{s},W),\sigma), \quad p(y_d\,|\,w,\sigma) = \mathcal{N}(y_d;w,\sigma^2) \tag{24}$$

$$\text{with} \quad \overline{W}_d(\vec{s},W) = \sum_h W_{dh} s_h\,, \tag{25}$$

with $W \in \mathbb{R}^{D\times H}$. $\mathcal{N}(y_d;w,\sigma^2)$ is a scalar Gaussian density function with mean w and variance σ^2. The introduction of $\overline{W}$ will turn out to be convenient for the analytical treatment below. Note that we otherwise could have written $p(\vec{y}\,|\,\vec{s},\Theta) = \mathcal{N}(\vec{y};W\vec{s},\sigma^2\mathbb{1})$.

We then use (24) to compute the update equation (the M-step) for the weight matrix W as described in Section 2. The update equation is gained from the necessary condition for an optimum of the free-energy in (2). The W-update is consequently a function of the sufficient statistics $\langle\vec{s}\rangle^T_{q^{(n)}}$ and $\left\langle\vec{s}\vec{s}^T\right\rangle_{q^{(n)}}$. Following Section 2, these expectation values are approximated by using (6) instead of the exact sufficient statistics (5), that is, we use a subset of cause combinations as selected by the truncation approach. Furthermore, we sum only over the N^{cut} data points in $\mathcal{M}$ as explained at the end of Section 2. The update equation is then given by:

$$W \;=\; \left(\sum_{n\in\mathcal{M}} \vec{y}^{(n)}\,\langle\vec{s}\rangle^T_{q^{(n)}}\right)\left(\sum_{n'\in\mathcal{M}}\left\langle\vec{s}\vec{s}^T\right\rangle_{q^{(n')}}\right)^{-1}. \tag{26}$$

Note that the equation can consistently be gained from the necessary condition for a free-energy optimum in Equation 19 (see M-step of Algorithm 2).

Equations 24 to 26 are valid irrespective of the sign of the entries of the generative fields and the input data. Generative models corresponding to the class of non-negative Matrix Factorization (NMF) methods are based on a linear combination of generative fields but rely on non-negative data points and generative fields. In Equation 26, non-negativity can be ensured for the generative fields by clamping small appearing weights at zero.

A more direct way to ensure non-negativity for the parameters of the model defined by (22) and (24) is to rely on convergence proofs similar to those used for classical NMF (Lee and Seung, 2001), that is, to ensure non-negativity by deriving a multiplicative update rule for the generative fields. For the EM algorithm used as a basis for ET, it can be shown that the parameter-free update rule

$$\vec{W}_h \leftarrow \vec{W}_h \odot \frac{\langle\vec{y}\,s_h\rangle_{\text{ET}}}{\sum_{h'}\vec{W}_{h'}\,\langle s_{h'}\,s_h\rangle_{\text{ET}}} \tag{27}$$

provides monotonic convergence towards the M-step solution of the generative fields, where we have introduced the ET-averaging

$$\langle f(\vec{y},s_h)\rangle_{\text{ET}} := \sum_{n\in\mathcal{M}}\sum_{\vec{s}\in\mathcal{K}_n} p(\vec{s}|\vec{y}^{(n)},W)\, f(\vec{y}^{(n)},s_h) = \sum_{n\in\mathcal{M}}\left\langle f(\vec{y}^{(n)},s_h)\right\rangle_{q^{(n)}}.$$

Equation 27 corresponds to a partial M-step of ET-NMF ('Expectation-Truncation-NMF'). In simulations, it is therefore applied iteratively with 20 partial M-steps after each E-step.

For the introduced generative model, Equation 27 converges towards the M-step solutions of the EM algorithm under non-negativity constraints. The update rule can be understood as a diagonally rescaled gradient descent derived from the EM algorithm, with a rescaling factor that is "optimally chosen to ensure convergence" (Lee and Seung, 2001). A thorough derivation of the update Equation 27 and its relation to the classical NMF algorithm known from the literature can be found in Appendix D.

The E-step of ET-NMF is based on the truncated expectation values (6) to calculate the averaged quantities in the W update equations according to

$$\langle \vec{y} s_h \rangle_{\mathrm{ET}} = \sum_{n \in \mathcal{M}} \vec{y}^{(n)} \langle s_h \rangle_{q^{(n)}} \quad \text{and} \quad \langle s_{h'} s_h \rangle_{\mathrm{ET}} = \sum_{n \in \mathcal{M}} \langle s_{h'} s_h \rangle_{q^{(n)}}$$

so that the sufficient statistics of the ET-NMF model that have to be computed for the M-step will be given by the first and second order moments $\langle s_h \rangle_{q^{(n)}}$ and $\langle s_{h'} s_h \rangle_{q^{(n)}}$ of the (approximate) posterior.

For our generatively formulated version of NMF with M-step equations (26) or (27) we can now apply ET as described in Section 2. We ran the ET-NMF algorithm with both M-step versions (26) and (27) and observed, at least for the data used, a qualitatively and quantitatively comparable behavior. Note that the probability density $p(\vec{y}\,|\,\Theta)$ of the model is invariant under the exchange of any two generative fields (or basis functions), $\vec{W}_h \rightarrow \vec{W}_{h'}$, because of symmetric priors. By the definition of the truncated generative models (Section 3), it can instantly be seen that their probability densities $p(\vec{y}\,|\,c=1,\Theta)$ and $p(\vec{y}\,|\,c=0,\Theta)$ are also invariant under these transformations (the same will apply for the other models considered).

As indicated, the sufficient statistics for ET-NMF are given by the first and second order moments, $\langle s_h \rangle_{q^{(n)}}$ and $\langle s_{h'} s_h \rangle_{q^{(n)}}$, of the exact posterior; to find approximations to these intractable expectation values we first have to find appropriate selection functions $\mathcal{S}_h$. A natural starting point for finding such functions is to consider the joint probability $p(s_h=1,\vec{y}^{(n)}\,|\,\Theta) = \sum_{\vec{s}\,(s_h=1)} p(\vec{s},\vec{y}^{(n)}\,|\,\Theta)$. If we knew that the joint probability was small for a given h, we would know that the sum over all $\vec{s}$ in (6) which contains $\vec{s}$ with $s_h=1$ is small as well. Furthermore, note that for a given data point the joint represents the part of $p(s_h=1\,|\,\vec{y}^{(n)},\Theta)$ which depends on h, $p(s_h=1\,|\,\vec{y}^{(n)},\Theta) = \frac{p(s_h=1,\vec{y}^{(n)}\,|\,\Theta)}{p(\vec{y}^{(n)}\,|\,\Theta)}$. $p(s_h=1\,|\,\vec{y}^{(n)},\Theta)$, on the other hand, directly reports the probability of unit h to have contributed to the data point. It could thus be regarded as the optimal selection function. Unfortunately, neither $p(s_h=1,\vec{y}^{(n)}\,|\,\Theta)$ nor $p(s_h=1\,|\,\vec{y}^{(n)},\Theta)$ are computationally tractable and, thus, neither function fulfils one of the properties demanded from a selection function. Therefore, we will compute an upper bound of $p(s_h=1,\vec{y}^{(n)}\,|\,\Theta)$ which is tractable and still serves well in selecting a subset of hidden units that can explain a given $\vec{y}^{(n)}$. Let us for this purpose consider the data point dependent weight matrix defined by: $W^{\mathrm{ub}}_{dh} := W^{\mathrm{ub}}_{dh}(\vec{y}^{(n)},W) = \max\{y^{(n)}_d, W_{dh}\}$ and $\vec{W}^{\mathrm{ub}}_h := \big(W^{\mathrm{ub}}_{1h},\ldots,W^{\mathrm{ub}}_{Dh}\big)^T$, where 'ub' refers to 'upper bound'. This formal definition of W^{ub} allows for a compact notation of an upper bound of $p(s_h=1,\vec{y}^{(n)}\,|\,\Theta)$. Because of the non-negativity of the entries in W and the mono-modality of the Gaussian distribution w.r.t. the mean, we can show (see Appendix B for details):

$$p(s_h=1,\vec{y}^{(n)}\,|\,\Theta) = \sum_{\vec{s}\,(s_h=1)} p(\vec{y}^{(n)}\,|\,\overline{W}_d(\vec{s},W),\sigma)\,p(\vec{s}\,|\,\pi) \;\leq\; \pi\, p(\vec{y}^{(n)}\,|\,\vec{W}^{\mathrm{ub}}_h,\sigma) =: \mathcal{S}_h(\vec{y}^{(n)})\,. \tag{28}$$

The upper bound $\mathcal{S}_h$ is tractable (also compare Appendix C) because we have removed the summation over the $\vec{s}$. The price we pay is that the selection function $\mathcal{S}_h$ can be a relatively coarse estimate in some cases. Importantly, however, we know that if $\mathcal{S}_h$ is sufficiently small, then the contribution of all joint probabilities $p(\vec{s},\vec{y}^{(n)}\,|\,\Theta)$ with $s_h=1$ can be neglected. The E-step given by the approximation of the sufficient statistics in Section 2 with (28) as selection function together with the M-step in (27) or (26) represents the learning algorithm for the NMF generative model defined by (22), (24) and (25) with non-negativity constraint.

In addition we add after each M-step a small parameter noise to the basis vectors W (iid Gaussian, standard deviation 0.05) and we use a standard relaxation scheme in order to avoid local optima of the potentially multi-modal likelihood landscape. Annealing (see, e.g., Ueda and Nakano, 1998; Sahani, 1999) amounts to the replacements: $(1/\sigma^2) \to (\beta/\sigma^2)$, $\pi \to \pi^\beta$ and $(1-\pi) \to (1-\pi)^\beta$. The constant β is an inverse 'temperature', $\beta = 1/T$, where T is decreased from a high value T^{init} to a value T^{final} equal or close to one.

4.1.1 EXPERIMENTS - ARTIFICIAL DATA

Let us consider data as displayed in Figure 3A. That is, we consider hidden causes in the form of horizontal and vertical bars (five pixels each) on a 5×5 grid. Each bar appears with probability $\frac{2}{10}$ such that there are on average two bars per data point. We use $N = 500$ data points. The grey-value of a bar is taken to be 10, background pixels are zero. The bars superimpose linearly (pixel values in regions of overlap are 20) and are subject to Gaussian noise with standard deviation $\sigma = 2.0$. Data as in Figure 3A are well-suited to study the functioning of the approximation scheme because we know the underlying generating process and have ground-truth for each data point. We will later use this knowledge to illustrate the influence of each data point on the update of the model parameters. For this reason data points such as displayed in Figure 3A or versions without noise are frequently used in the recent literature (see, e.g., Hinton et al., 1995; Hoyer, 2002; Spratling, 2006).

The model which is applied to the data uses $H = 10$ hidden units and $D = 25$ observed units. The entries of W are initialized by drawing iid from a Gaussian distribution with a mean of 4 and a standard deviation of $\frac{4}{3}$. The standard deviation of the generative model is set to $\sigma = 2.0$ and the value of π is set to $\frac{2}{10}$. Small deviations from these values did not have significantly negative effects on the algorithms performance in extracting data components. Figure 4B shows the cooling schedule for annealing. We linearly decrease T from $T^{\text{init}} = 13$ to $T^{\text{final}} = 1$ during 100 EM-iterations (including ten initial iterations at T^{init} and twenty final iterations at T^{final}). Figure 3B shows a typical time-course of the parameters W if trained as described above. As approximation parameters we used $H' = 5$ and $\gamma = 3$. Although approximate EM schemes do in general not guarantee the increase of the data likelihood, we find that the learning algorithm increases the likelihood to values close to the one for the generating parameters (dashed horizontal line in Figure 4A). This behavior is reflected by the convergence of the model parameters to values close to the generating ones (compare Figure 3). In most trials the parameters converged to approximately optimal values relatively early but in some trials they converged relatively late during learning (compare likelihood values in Figure 4A). We ran 50 trials with 50 different sets of data points generated as described above. The algorithm extracted all bars in all of the trials (see Appendix E for measurement details). To quantify the quality of parameter reconstruction we computed, for each trial, the mean absolute error (MAE) between the obtained generative fields, $\vec{W}_h$, and the corresponding generating causes: $\text{MAE} = \frac{1}{HD} \sum_{hd} |W_{dh} - W^{\text{gen}}_{dh'}|$ where $W^{\text{gen}}_{dh'}$ denotes the cause represented by $\vec{W}_h$ (compare Appendix E). For all trials the MAE was smaller than 0.24 with a mean of 0.20 (note that pixels of bars were set to 10.0 and background to zero).

In a second series of measurements we used the model with the same parameters and the same data as above except that the generating noise variance was set to zero. In 50 trials on this non-noisy data the algorithm extracted all bars in 46 of the trials (92% reliability) and extracted nine of the ten

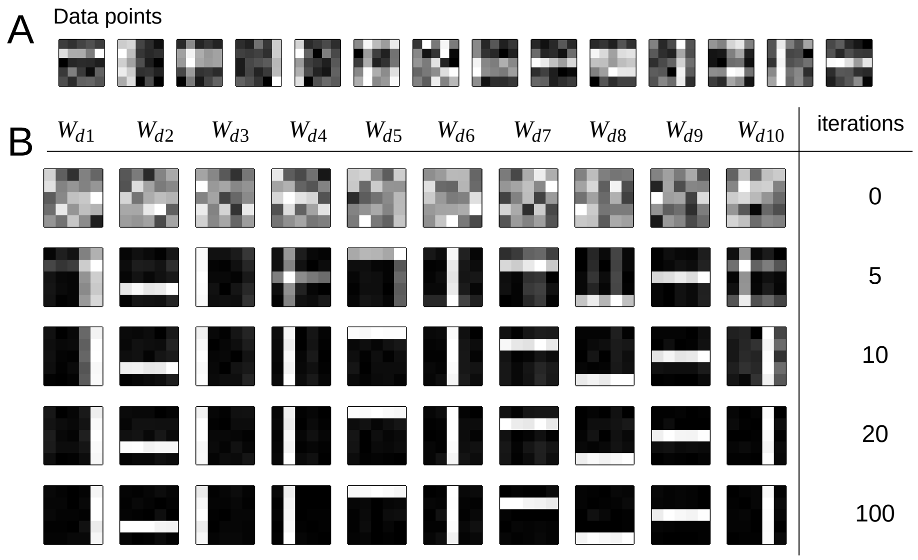


Figure 3: **A** 14 data points of the linear bars test with Gaussian noise. **B** Time course of the generative fields W of the NMF-like generative model if Expectation Truncation is used for learning.

single bars in the four remaining cases. All successful trials had a MAE of below 0.20 with a mean MAE of 0.05. Parameter recovery was more precise than in the previous trial series because of the non-noisy data. A higher initial temperature or a longer cooling increased reliability to values close to 100% for this data. E.g., when we used $T^{\text{init}} = 15$ and stretched the cooling schedule in Figure 4B to 200 iterations, the algorithm found all bars in all of 50 trials. Likewise, increasing the number of data points increased reliability ($> 94\%$ reliability for $N > 1000$ data points with $T = 13$ and 100 iterations cooling).

High reliability (i.e., a high probability to extract all causes) in this linear bars test has also been observed for other learning algorithms (see, e.g., algorithms investigated in Spratling, 2006). Note, however, that the standard bars benchmark test (Földiák, 1990) uses non-linearly overlapping bars (we will come to the standard version of the bars test in the next section). For the presented NMF algorithms we have (as is usual in the literature) only inferred the parameters W. In our generative setting it is in principle also possible to learn the model parameters σ and π (compare Lücke and Sahani, 2008). However, for comparison with other approaches and for brevity, we focused on W.

To investigate the quality of the ET approximation more directly, the values $Q^{(n)}$ (Equation 21) were computed for 40 data points during learning. Figure 4C shows time courses of $Q^{(n)}$ during a trial on the noisy linear bars test using the parameters given above. The data points were randomly selected but it was made sure that twenty of them were generated by less or equal γ causes (bright green lines) and the other twenty by more than γ causes (dark red lines). As can be observed, the approximation quality of the twenty data points generated by $\leq \gamma$ causes quickly increases whereas the approximation for the other twenty data points approaches zero. From the used approximation this could have been expected as we have restricted the summation in (6) to hidden states with less

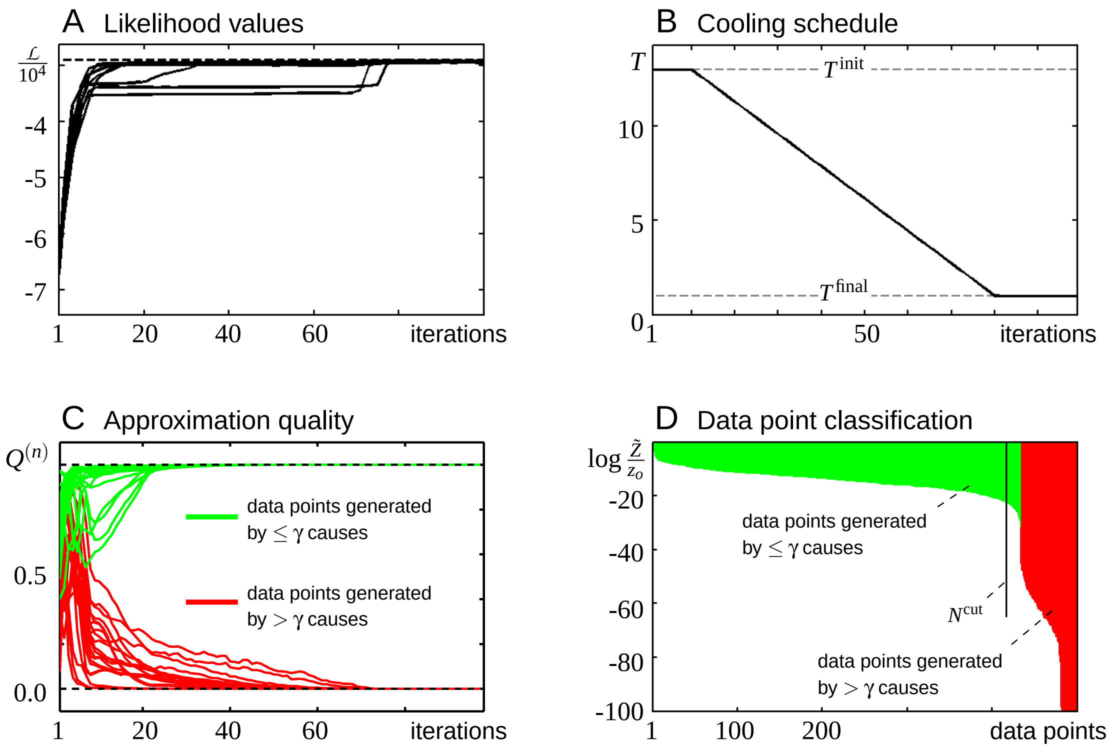


Figure 4: **A** Data likelihood during ten trials using the same set of $N = 500$ data points but different random initializations. The likelihood value that results from using the generating parameters is marked by the dashed horizontal line. **B** Cooling schedule during learning. **C** Approximation qualities $Q^{(n)}$ of 40 data points during learning. Twenty of the data points were generated by $\leq \gamma$ causes (bright green lines) and the other twenty by more than γ causes (dark red lines). **D** Sorted values of the sums $\tilde{Z} = \sum_{\vec{s}\in\mathcal{K}_n} p(\vec{s},\vec{y}^{(n)}\,|\,\Theta)$ at the end of learning. Bright green bars were used to mark the data points generated by $\leq \gamma$ causes, dark red bars to mark data points generated by more than γ causes. The N^{cut} data points left of the black vertical line were used in the final M-steps.

or equal $\gamma = 3$ active causes. For other trials, the values $Q^{(n)}$ show the same qualitative behavior. However, the exact time-courses can differ quantitatively from trial to trial.

Note that the poorly approximated data points do finally not negatively affect the parameter updates because they are not taken into account for the M-step. This is illustrated in Figure 4D which shows the logarithms of the sum $\tilde{Z} = \sum_{\vec{s}\in\mathcal{K}_n} p(\vec{s},\vec{y}^{(n)}\,|\,\Theta)$ at the end of learning and for each of the $N = 500$ data points used (divided by a common factor z_o). Bright green bars display the values of all data points generated by less or equal γ causes, dark red bars display the values of all other data points. The data points are ordered descendingly. According to the approximation used (see Section 2 resp. Equation 23), we only consider the N^{cut} data points left of the black bar, that is, we finally only use data points with quality values $Q^{(n)}$ close to one. The relation of the quality values to the KL-divergence in (21) directly shows that the ET approximation for these data points is virtually optimal in this case.

4.1.2 EXPERIMENTS - MORE REALISTIC DATA

As a second example, we applied the learning algorithm to gray-valued images of the postal digits database MNIST (http://yann.lecun.com/exdb/mnist/). This data is frequently used in the NMF literature, which makes it well suited as a means for comparison of our algorithm to standard NMF approaches. Note that we do not have ground-truth about the hidden components in this case.

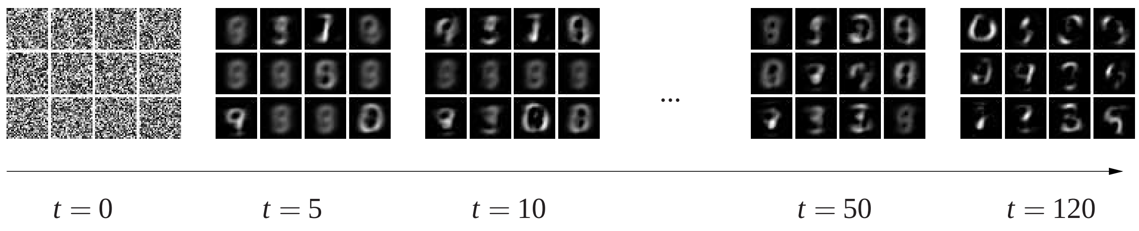


Figure 5: Resulting basis vectors $\vec{W}_h$ for the MNIST database. The probabilistic version of NMF trained with ET converges to a parts-based decomposition.

Figure 5 shows the application of the algorithm using $H = 12$ hidden variables and approximation parameters for ET set to $H' = 10$ and $\gamma = 5$. The prior parameter was set to $\pi = 0.3$ such that three to four of the 12 latent variables do explain a data point on average. The noise parameter σ in (24) was set to $\sigma = 0.73$ after screening through values between zero and one. The dimensionality of each data point is $D = 28 \times 28$ and we used a subset of $N = 1000$ data points, some of which are shown in Figure 6A. We linearly decreased the temperature as in Figure 4B but used a slightly longer learning time (120 iterations) to provide more time for convergence. In Figure 5 the time course of W displayed as basis vector sets is shown. As can be observed, the parameters W converge to basis vectors that represent digit parts.

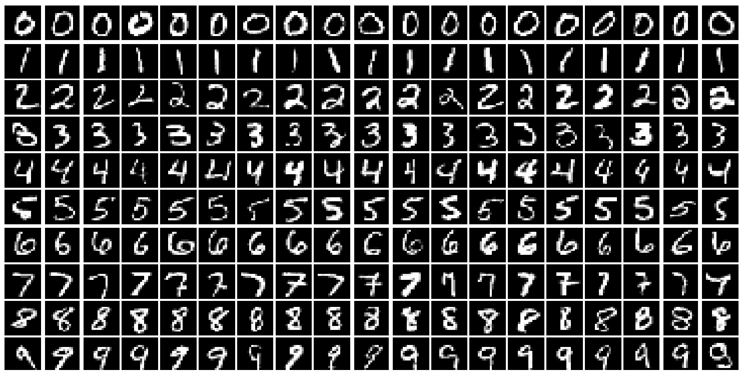
To assess the quality of the basis vectors and for comparison with standard NMF, we show average reconstructions of probabilistic NMF and standard NMF in Figure 6. In Figure 6B it can be observed that already for the small subset of 12 basis vectors in Figure 5 the reconstructions match the inputs in Figure 6A relatively well. Despite the constraint to binary hidden variables in our generative version of NMF, the resulting reconstructions are very similar to those of standard NMF as shown in Figure 6C. For these data, the overall average reconstruction error, $\frac{1}{ND}\sum_n ||\vec{y}^{(n)} - \sum_h \vec{W}_h \langle s_h \rangle_{q^{(n)}}||^2$, of the generative version is less than 5% larger than the reconstruction error of standard NMF.

4.2 Maximal Causes Analysis (MCA)

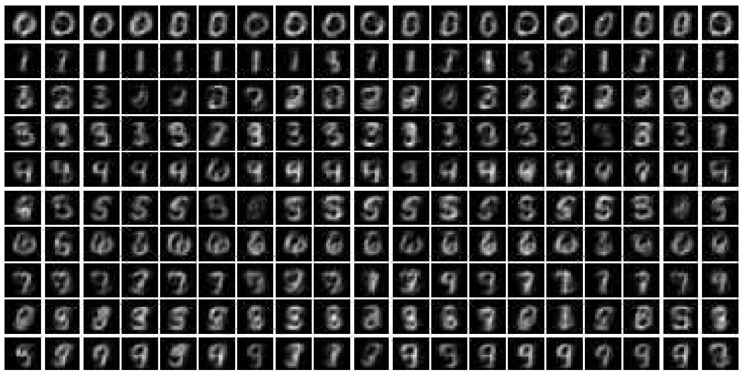
The second generative model we consider was suggested to extract the hidden causes from data whose components combine non-linearly. It uses a maximum rule in the place where NMF, sparse coding (Olshausen and Field, 1996), independent component analysis (ICA; Comon, 1994) and many other methods assume a linear superposition of hidden components:

$$p(\vec{y}\,|\,\vec{s},W) = \prod_{d=1}^{D} p(y_d\,|\,\overline{W}_d(\vec{s},W),\sigma), \quad p(y_d\,|\,w) = \mathcal{N}(y_d;w,\sigma^2) \qquad (29)$$
$$\text{with} \quad \overline{W}_d(\vec{s},W) = \max_h\{s_h W_{dh}\},$$

A Digit data used for testing the ET-version of NMF



B ET-NMF average reconstruction



C Standard NMF reconstruction

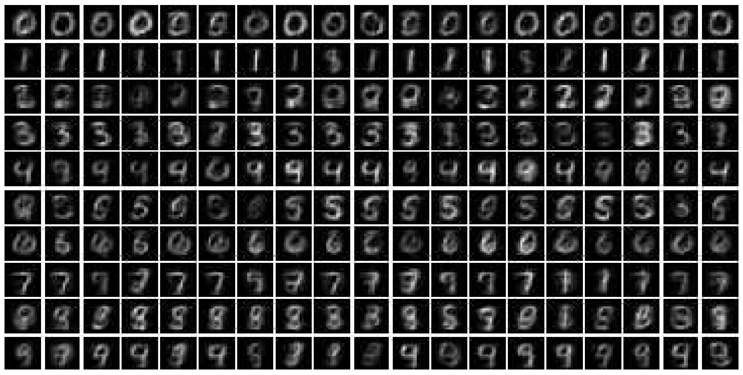


Figure 6: Reconstruction of data points representing hand-written digits. **A** Subset of the $N = 1000$ data points used for the application of probabilistic and standard NMF to the MNIST data base. **B** Average reconstruction of the digit data in **A** on the basis of the basis vectors in Figure 5. **C** Reconstruction of the digit data in **A** using standard NMF with the same number of basis vectors.

where we have used a Gaussian noise model with w as a place-hoder for $\overline{W}_d(\vec{s},W)$ (note the difference to Poisson noise used by Lücke and Sahani, 2008). For the activities of the binary hidden variables s_h we use the same prior as for the NMF model in Section 4.1 (see Equation 22). As in the previous model, the weight matrix $W \in \mathbb{R}^{D\times H}$ parameterizes the influence of the hidden causes on the distribution of $\vec{y}$. The function $\overline{W}_d(\vec{s},W)$ in (29) gives the *effective weight* on y_d, resulting from a particular instance of the state vector $\vec{s}$. An update rule for the weight matrix W of this model was derived in Lücke and Sahani (2008) and is given by:

$$W_{dh} = \frac{\sum\limits_{n\in\mathcal{M}} \left\langle \mathcal{A}^{\rho}_{dh}(\vec{s},W)\right\rangle_{q^{(n)}} y_d^{(n)}}{\sum\limits_{n\in\mathcal{M}} \left\langle \mathcal{A}^{\rho}_{dh}(\vec{s},W)\right\rangle_{q^{(n)}}}, \quad \text{where} \qquad \mathcal{A}^{\rho}_{dh}(\vec{s},W) = \left(\frac{\partial}{\partial W_{dh}}\overline{W}^{\rho}_d(\vec{s},W)\right) \tag{30}$$

$$\overline{W}^{\rho}_d(\vec{s},W) = \left(\sum_{h=1}^{H}(s_h W_{dh})^{\rho}\right)^{\frac{1}{\rho}}. \tag{31}$$

The parameter ρ controls the nonlinearity and is increased to large values during learning. Again, the entries in W are non-negative. To derive a selection function we can therefore apply the same arguments as for NMF and thus arrive at the very same functions $\mathcal{S}_h$ as given in Equation 28. The selection function (28), M-step equations (30) and (31), and the E-step approximation described in Section 2 represent a full learning algorithm for the extraction of non-linearly combining components, which will be referred to as MCA_{ET}.

4.2.1 EXPERIMENTS - ARTIFICIAL DATA

To study the properties of MCA_{ET} let us first apply it to data with ground-truth. A well-suited type of data for the algorithm is the so-called bars test introduced by Földiák (1990). The bars test has become a standard benchmark for component extraction algorithms (see, e.g., Saund, 1995; Dayan and Zemel, 1995; Hochreiter and Schmidhuber, 1999; Charles et al., 2002; Lücke and von der Malsburg, 2004; Spratling, 2006; Butko and Triesch, 2007; Lücke and Sahani, 2008) and thus allows for quantitative comparison with other systems. To generate data according to the bars test we use the same parameter settings as for the artificial data in Figure 3A, that is, $D = 5\times 5$, bars pixel value 10 and other pixels zero, Gaussian generating noise with standard deviation 2.0, and the probability for each bar to occur is $\frac{2}{10}$. In contrast to the data used in the experiment of Figure 3, however, the standard form of the bars test uses a non-linear superposition of the causes (overlapping bar regions have pixel values 10 instead of 20 for NMF). Figure 7A shows a random selection of 12 of the $N = 500$ data points used.

We apply MCA_{ET} to the data using the same model parameters and the same approximation parameters ($H' = 5$ and $\gamma = 3$) as for the linear bars test in Section 4.1.1. Annealing for MCA_{ET} amounts to the same replacements as for NMF: $(1/\sigma^2) \to (\beta/\sigma^2)$, $\pi \to \pi^{\beta}$ and $(1-\pi) \to (1-\pi)^{\beta}$. Additionally, ρ in (30) and (31) is increased from a relatively small value at T^{init} to a large value at T^{final} by making ρ temperature dependent: $\rho = \frac{1}{1-\beta} = \frac{T}{T-1}$. As cooling schedule we use the same one as in Section 4.1.1 (see Figure 4B) but with $T^{\text{final}} = 1.05$ to avoid a singularity for ρ. For MCA we found it beneficial to add to the set $\mathcal{K}_n$ (Equation 7), the set of all vectors with just one non-zero entry: $\overline{\mathcal{K}}_n = \mathcal{K}_n \cup \{\vec{s} \mid \sum_i s_i = 1\}$. Making $\mathcal{K}_n$ larger can in general only increase the accuracy of the approximation. At the same time, using $\overline{\mathcal{K}}_n$ instead of $\mathcal{K}_n$ does not change the scaling behavior with H of the algorithm (see Appendix C for a discussion).

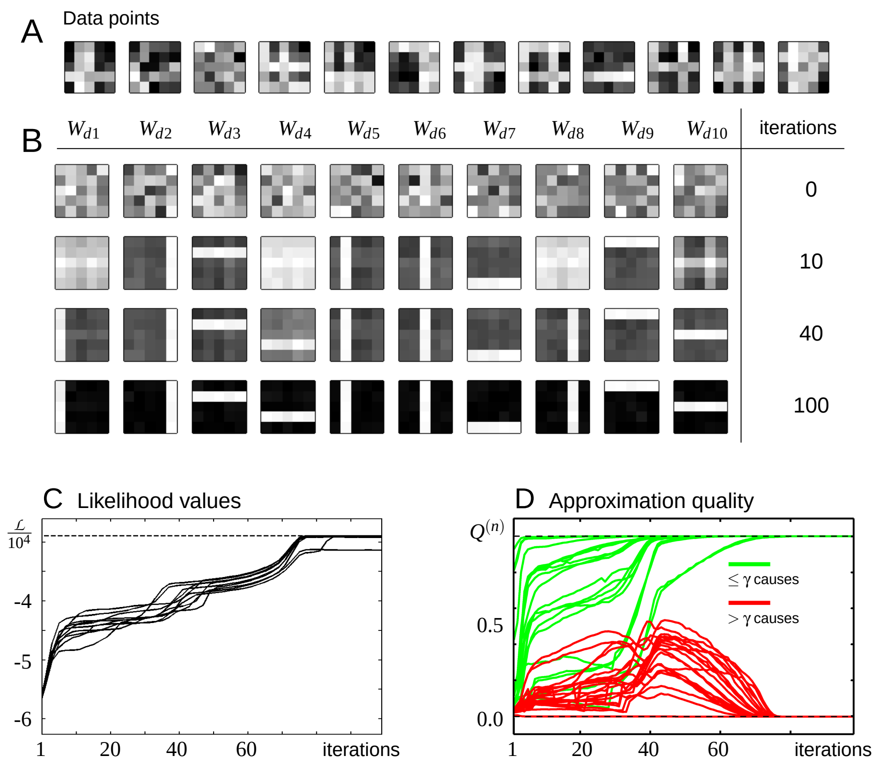


Figure 7: **A** Example data of a bars test with $D = 5 \times 5$ and additive Gaussian noise. **B** Time course of W for the MCA model trained with ET. **C** Likelihood values for ten trials with the same set of training patterns and different initial conditions for each trial. **D** Time course of the quality values of 40 data points during one trial. Values are plotted for twenty randomly selected data points generated by less or equal γ causes (bright green lines) and twenty randomly selected data points generated by greater than γ causes (dark red lines).

Figure 7B shows a typical time course of W during learning. Figure 7C shows time courses of the data likelihood for ten different runs using the same data set. The behavior of the likelihood values results from the specific form of annealing which includes the annealed nonlinearity in Equation 31. In Figure 7D typical time courses for the quality values $Q^{(n)}$ (Equation 21) for 40 data points are shown. For the 20 data points which were generated by $\leq \gamma$ causes (bright green lines) the quality values increased to one. For the data points generated by $> \gamma$ causes (dark red lines) the quality values finally decreased to zero. As for NMF, only the data points which were well approximated were finally taken into account for learning.

To probe the reliability of MCA$_{\text{ET}}$, we ran 50 trials of the bars test with the bars test parameters as given above. In each trial we used a new set of $N = 500$ data points. In 46 of the 50 trials MCA$_{\text{ET}}$ extracted all bars (92% reliability), and in four of the trials 9 of 10 bars were extracted. Reconstruction of the generating parameters was high with a maximal MAE of 0.35 and a mean

MAE of 0.29 (bars had value 10). We observed that the convergence to local optima in 8% of the trials was mainly due to effects of finite sample size. When we ran 100 trials using the same parameters but $N = 2000$ data points instead of $N = 500$, the algorithm extracted all bars in all trials.

For comparison with other methods, we ran additional trials using the same parameters for bars generation but with Gaussian noise for data pixels set to variance zero. This version of the bars test is presumably the one most commonly used in the literature (e.g., Saund, 1995; Dayan and Zemel, 1995; Hochreiter and Schmidhuber, 1999; Lücke and Sahani, 2008). In 41 of the 50 trials MCA_{ET} extracted all bars (82% reliability) and found 9 of 10 bars in the other nine trials. Again, reconstruction of the generating parameters in the successful trials was high with a maximal MAE of 0.14 and a mean MAE of 0.05. Also for the non-noisy bars test, reliability of the algorithm increased when we increased the sample size. For instance, we obtained reliabilities of more than 90% for N larger than about 2000. With $N = 10\,000$ and slower cooling (same cooling schedule as in Figure 4B but stretched to 200 iterations), the algorithm found all bars in all of the 50 trials. Achieving close to 100% reliability is thus more difficult for non-noisy bars.[1]

In earlier work, generative modelling approaches to the bars test merely achieved relatively low reliability values. For instance, the model of Saund (1995) achieved 27% reliability, and the model of Dayan and Zemel (1995) (although trained without overlapping bars) achieved 69%. Approaches such as PCA or ICA that assume linear superposition have been reported to fail in this task (see Hochreiter and Schmidhuber, 1999). Other objective function approaches and different types of neural network approaches (e.g., Charles et al., 2002; Lücke and von der Malsburg, 2004; Spratling, 2006, and references therein) have been more successful in terms of reliability. They do, however, often use hidden assumptions and constraints which make an objective comparison difficult (see, e.g., Spratling, 2006, or Lücke and Sahani, 2008, for discussions). The more recently suggested approach of MCA (Lücke and Sahani, 2008) represents a fully generatively interpretable approach which achieves high reliability values. The unrestricted version of MCA_3 extracts all bars in 90% of the trials with noisy bars (with Poisson noise) and in 81% of the cases for the noiseless bars. MCA_{ET} slightly improves on these results with 92% vs. 90% for noisy bars and 82% vs. 81% in the non-noisy case (experiments with $N = 500$). If more data points are used, MCA_{ET} shows close to 100% reliability (50 of 50 trials successful, see above).

Other than the standard bars test, there has recently been an increasing interest in bars with more pronounced overlap. We therefore used a version of the bars test as suggested by Lücke (2004). For this data, bars of the same orientation can overlap (two neighboring vertical bars are not disjoint but overlap). For the test we adopted the same parameter setting for such input as used by Spratling (2006) and Lücke and Sahani (2008), that is, $N = 400$ example patterns, 16 bars, $D = 9 \times 9$, bars appear with probability $\frac{2}{16}$, and number of hidden units is $H = 32$. Bars are two pixels wide such that parallel neighboring bars have a one pixel wide region of overlap. Figure 8A shows some examples of the data points used. We applied MCA_{ET} to the data using the same parameters as for the standard bars test except of a higher initial temperature ($T = 23$) and longer cooling time (the cooling schedule in Figure 4B was stretched by a factor four to 400 iterations). In 21 of the 25 trials the system extracted all bars (see Figure 8B). In four trials 15 of the 16 bars were represented. The average number of extracted bars was thus 15.84. In all the successful trials, reconstruction of the generating parameters was high with a maximal MAE of 0.05 and a mean MAE of 0.04 (bars have

1. Note that alternatively to using $N = 10\,000$ and 200 iterations, we could, for non-noisy data, simply add Gaussian pixel noise. This would take us back to the noisy-bars test and we would obtain close to 100% reliability with $N = 1000$ data points.

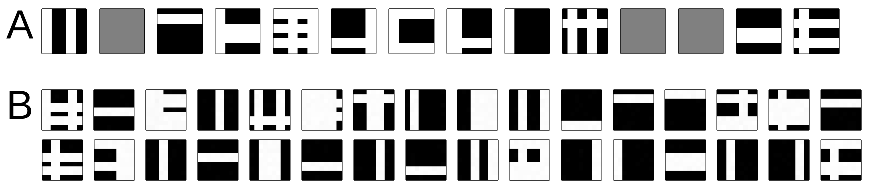


Figure 8: **A** Selection of 14 data points of a bars test with increased overlap. **B** Typical parameters W after learning if twice as many hidden units as bars are used. The supernumerous units are used to represent composite patterns.

value 10). As for the standard bars test we found that convergence to local optima was caused by effects of the finite sample size. When we repeated the experiment with the same generating and model parameters but with $N = 800$ instead of $N = 400$ data points, the algorithm extracted all bars in all of 50 trials (mean number of bars extracted equal to 16.0). In work by Spratling (2006) state-of-the-art systems were quantitatively compared using the mean number of extracted bars. From the evaluated systems only few achieved values close or equal to the optimal value of 16 for $N = 400$. From the systems with high values, many required additional constraints on the parameters (e.g., constrained forms of NMF) that had to be set by hand (see Spratling, 2006; Lücke and Sahani, 2008, for discussions).

In general, the component extraction performance of MCA_{ET} on the different bars test tasks is similar to the performance of MCA_3 suggested by Lücke and Sahani, 2008. In terms of computational cost, MCA_{ET} represents a substantial improvement, however (even compared to R-MCA_2, a constrained form of MCA; see Lücke and Sahani, 2008). This allows for applications with large H as demonstrated, for example, in the following section.

4.2.2 EXPERIMENTS - MORE REALISTIC DATA

As an example for an application to more realistic data, we applied MCA_{ET} to visual data in the form of image patches. We used the same image, Figure 9A, as in work by Lücke and Sahani (2008) to allow for a comparison. We randomly selected $N = 40\,000$ patches of 10×10 pixels as data points (see Figure 9D for ten examples). The data points were globally scaled to lie in the interval $[0, 10]$. However, just very few pixels had values close to 10 after scaling. The mean pixel value was 1.6 and thus smaller than for the bars test. We therefore used a smaller assumed Gaussian noise for the model (standard deviation $\sigma = 1.0$ instead of $\sigma = 2.0$) and started cooling at a lower temperature of $T^{\text{init}} = 4.0$. Also the small noise term on the model parameters was scaled down (0.01 instead of 0.05). During learning, we cooled for 400 iterations (cooling schedule of Figure 4B stretched by a factor four) and allowed for additional 400 iterations at T^{final} to guarantee full convergence (although changes after iteration 400 were small).

Figures 9B and 9C show the resulting parameters W after applying MCA_{ET} with $H = 50$ and $H = 100$ hidden units, respectively. For the approximations we again used $H' = 5$ and $\gamma = 3$. As can be observed, the extracted generative fields represent typical components of the training patches (compare Figure 9B-D). Data generated according to the MCA generative model using the

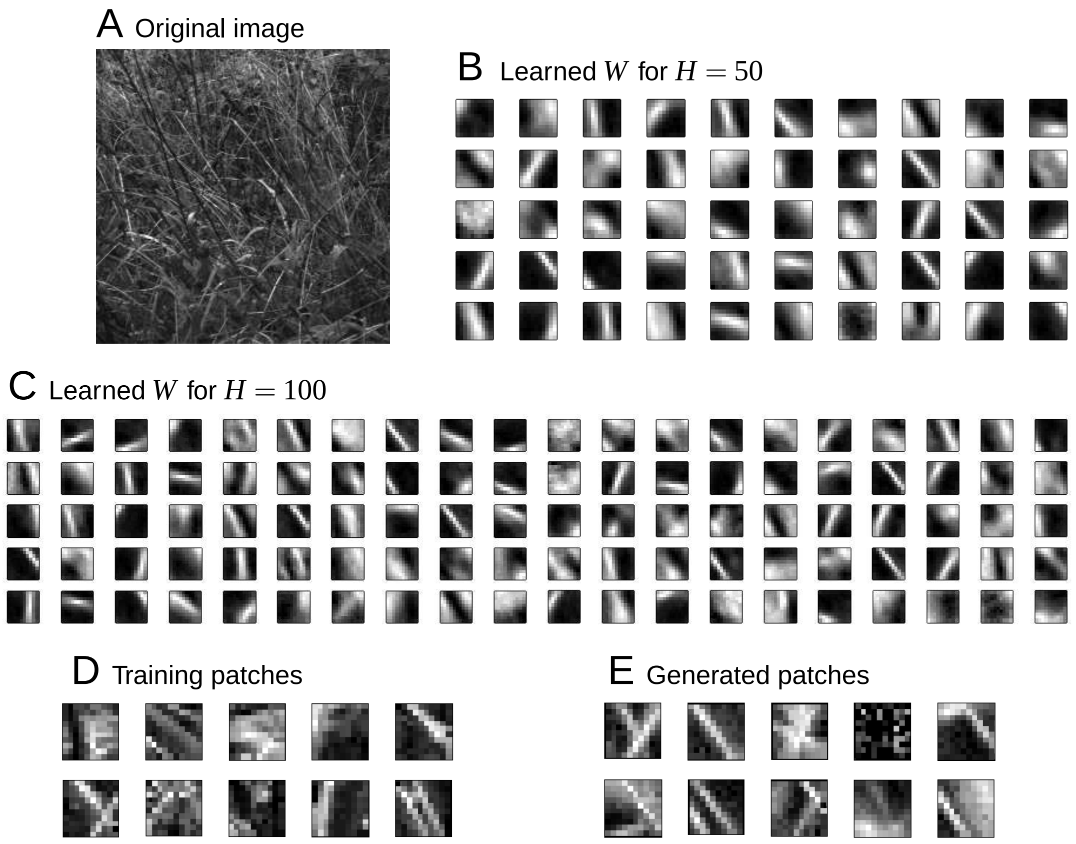


Figure 9: Application to visual data. **A** The 250-by-250 pixel image used as basis for the experiment. The image is taken from the van Hateren database of natural images (brightened for visualization). **B** Parameters $\vec{W}_h = (W_{h1}, \ldots, W_{hD})^T$ of the MCA generative model with $H = 50$ trained by ET with $H' = 5$ and $\gamma = 3$. **C** Parameters $\vec{W}_h$ of MCA with $H = 100$ ($H' = 5$ and $\gamma = 3$). **D** A selection of 10 typical training patches. **E** Ten examples of patches generated according to the MCA generative model using the generative fields in **C**. To reduce the apparent noise level, the patches were generated with a smaller σ than for training.

extracted generative fields thus resemble the structure of the training patches (Figure 9E). Note that the components of the data (e.g., images of grass blades and stems) superimpose non-linearly which motivated the application of MCA.

Due to impractically long computation times, no previous version of MCA could be applied to numbers of hidden units much larger than 50. The maximum achievable so far, was the application of the constrained MCA version (R-MCA$_2$) with $H = 50$ to $N = 5000$ patches of 10×10 pixels. As shown in the examples of Figure 9, Expectation Truncation allows for larger scale applications of unconstrained MCA with $H = 100$ and beyond. Compared to MCA$_3$ the number of states that have to be evaluated by MCA$_{\mathrm{ET}}$ for $H = 100$ is reduced by more than three orders of magnitude (see Section 5.2 and Appendix C for discussions).

4.3 Linear Components Analysis (LinCA)

As third and final example let us discuss a generative model whose components can be negative as well as positive. Consider, therefore, the model given by (22), (24) and (25) without the restriction to non-negative weights. For small π in (22) such a generative model is reminiscent of sparse coding (SC; Olshausen and Field, 1996): its hidden variables are sparsely active, its basis functions are combined linearly, and its observed variables are, given the latents, independently drawn from standard Gaussian distributions. Instead of defining explicit prior and noise distributions, SC-like models are often, more informally, written in the form:

$$\vec{y} = \sum_{h=1}^{H} s_h \vec{W}_h + \sigma\vec{\eta} = W\vec{s} + \sigma\vec{\eta}\,, \tag{32}$$

where the entries of $\vec{\eta}$ are independently and identically drawn from a Gaussian distribution with zero mean and unit variance.

The model (32) with binary factors s_h distributed according to the Bernoulli prior (22) can be trained with Expectation Truncation. As M-step we can directly use Equation 26. However, in contrast to the previous generative models, we can not use Equation 28 as a selection function because it was derived assuming non-negative entries W_{dh}. To find appropriate selection functions, let us first consider a special case: let us assume the number of observed and hidden variables to be identical, $H = D$, and let us assume zero observation noise ($\sigma = 0$ in Equation 32; also compare Teh et al., 2003). For continuous factors s_h with non-Gaussian priors this special case represents the standard version of ICA (see, e.g., Comon, 1994; Hyvärinen et al., 2009). ICA is deterministic in the sense that for any given data point $\vec{y}^{(n)}$ the generating hidden vector $\vec{s}^{(n)}$ is known exactly. If W is invertible, the generating hidden vector is given by $\vec{s}^{(n)} = W^{-1}\vec{y}^{(n)}$ (as can directly be deduced from Equation 32 with $\sigma = 0$). ICA is most frequently applied to (PCA-)whitened data (compare, e.g., Bishop, 2006; Hyvärinen et al., 2009), in which case W is an orthogonal matrix ($W^{-1} = W^T$). For ICA on whitened data, the generating hidden units of a data point $\vec{y}^{(n)}$ are thus given by $\vec{s}^{(n)} = W^T\vec{y}^{(n)}$. In other words, the conditional distribution $p(\vec{y}\,|\,\vec{s},\Theta)$ and thus the posterior $p(\vec{s}\,|\,\vec{y}^{(n)},\Theta)$ become equal to the delta function $\delta(\vec{s} - W^T\vec{y}^{(n)})$. For $\sigma > 0$ in Equation 32, the conditional distribution $p(\vec{y}\,|\,\vec{s},\Theta)$ is, in hidden space, a Gaussian function with mean $W^T\vec{y}^{(n)}$, and the posterior is proportional to the product of this function with the prior. A multiplication with a sparse prior has the effect of moving the Gaussian function closer to those axes h with large values $\vec{W}_h^T\vec{y}^{(n)}$. The scalar products $\vec{W}_h^T\vec{y}^{(n)}$ can thus serve to select those hidden dimensions which span the space most posterior mass lies near to.

Knowing where most posterior mass is concentrated is the crucial prerequisite for finding selection functions for ET. In the binary case of model (32) with Bernoulli prior (22), selecting the hidden dimensions corresponds to selecting the hidden variables which are most likely to have non-zero entries. In analogy to the ICA case, we thus use the scalar product to define selection functions:

$$\mathcal{S}_h(\vec{y}^{(n)}) = \frac{\vec{W}_h^{\mathrm{T}}\vec{y}^{(n)}}{|\vec{W}_h|\,|\vec{y}^{(n)}|}, \quad \text{with} \;\; |\vec{v}| = \sqrt{\textstyle\sum_{i=1}^{D}(v_i)^2} \quad \text{and} \quad \vec{W}_h^T = (W_{1h},\ldots,W_{Dh}). \tag{33}$$

We use normalized scalar products because we are not constrained to an orthogonal matrix W and want to prevent the lengths of $|\vec{W}_h|$ from having a strong influence on the selections. Note, however, that if the matrix W is orthogonal, the selection is equivalent to one based on non-normalized scalar products. The normalization by $|\vec{y}^{(n)}|$ does not affect the selection because $\vec{y}^{(n)}$ is independent of h.

Selection functions (33), M-step equation (26), and the ET approximation of the E-step represent a learning algorithm that will be referred to as Linear Components Analysis (LinCA).

4.3.1 EXPERIMENTS - ARTIFICIAL DATA

To study the LinCA learning algorithm it is first applied to artificial bars data. We use the same linear bars test setup as for the noisy bars test in Section 4.1.1 (compare Figure 3A) but invert five of the ten bars to negative values. Figure 10A shows 12 examples of the $N = 500$ data points used. The observed variables (or 'pixels') of positive bars have value 10.0, observed variables of negative bars have value -10.0, and all other observed variables have value zero. Consequently, we initialize the parameters W with positive and negative values by drawing iid from a Gaussian with zero mean and standard deviation 2.0. All other parameters (for initialization, approximation, annealing, and data generation) are chosen as in Section 4.1.1 for the NMF model. Figure 10B shows a typical time course of the parameters W for noisy data points. As can be observed, the parameters converge to the true generating causes relatively early. Figure 10C shows the likelihood values of ten trials with the same set of $N = 500$ data points. In most trials, likelihoods converged quickly to values close to the likelihood values of the generating parameters (dashed horizontal line). In some trials convergence took longer, however. Figure 10D shows the quality values $Q^{(n)}$ (Equation 21) during a typical trial. After an initially relatively low approximation quality, the quality values of the data points generated by less or equal γ causes (bright green lines) quickly increase. As previously, only these data points are finally used for learning. Data points generated by greater than γ causes (dark red lines) are finally discarded.

To measure the reliability of the system, we ran 50 trials with 50 different data sets of $N = 500$ data points. In all trials all bars were extracted. Parameter reconstruction in all the trials was high with all MAE smaller than 0.28 and a mean MAE of 0.21. For bars without noise we extract all bars in 49 trials (98%) and nine of ten bars in one trial. Due to the non-noisy data, parameter reconstruction was higher than for noisy data. The MAE of all successful trials was smaller than 0.09 and the mean MAE was 0.04. The reliability for non-noisy data increased to still higher reliability values when we cooled longer. If the cooling schedule in Figure 4B was stretched to 200 iterations, all bars were found in all of 100 trials. Alternatively, reliability increased when we increased the sample size: all bars were found in 100 of 100 trials if $N = 1000$ data points were used.

4.3.2 EXPERIMENTS - MORE REALISTIC DATA

As an example of more realistic data, we applied LinCA to sound waveforms. Sound waves have positive and negative parts and their components superimpose linearly, which is consistent with the assumptions of the LinCA model. As data we used short sound intervals obtained from recordings of ten different male voices uttering the sentence: "Don't ask me to carry an oily rag like that". Data was taken from the TIMIT database with voices sampled at 16kHz. To only learn from the spoken text, we cut off the silent initial part and the silent final part of each recorded sentence. Furthermore, we multiplied each voice recording by a different factor such that each recording filled the interval $[-5, 5]$ (maximal absolute amplitude of each recording 5.0). This compensated for different sound levels of the different speakers and made the data range comparable to those of

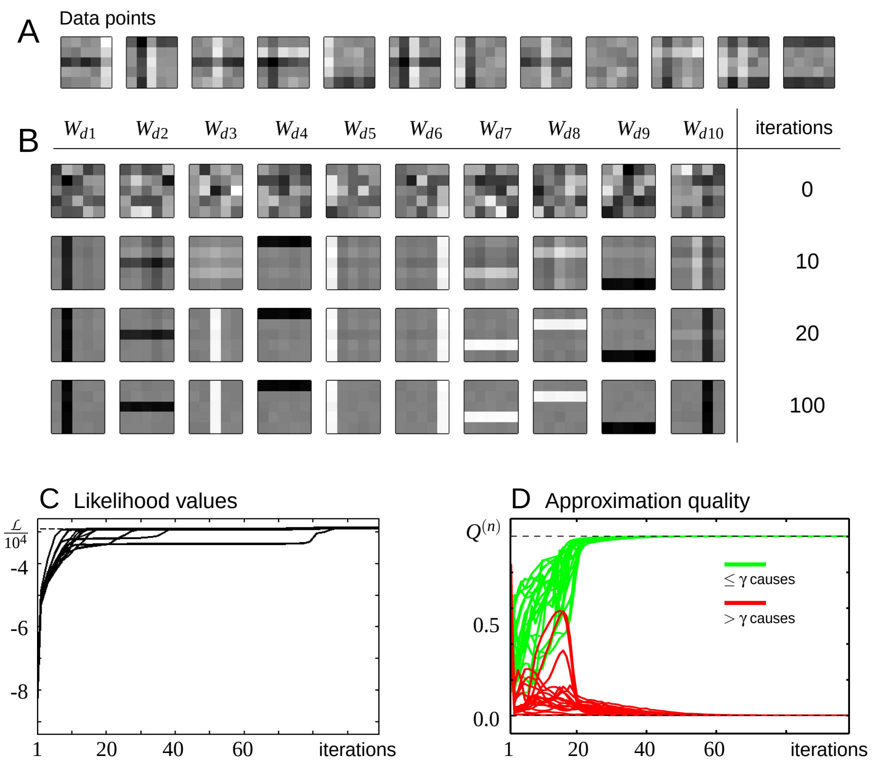


Figure 10: **A** Example data of a bars test with components of positive and negative values, $D = 5\times5$, and additive Gaussian noise. **B** Time course of W for the LinCA model trained with ET. **C** Likelihood values for ten trials with the same set of training patterns and different initial conditions for each trial. **D** Time course of the quality values of 40 data points during one trial. Values are plotted for twenty randomly selected data points generated by less or equal γ causes (bright green lines) and twenty randomly selected data points generated by greater than γ causes (dark red lines).

previous experiments. After the multiplication only few data values lay close to -5 or 5. As in Section 4.2.2 we therefore assumed a lower data noise than for the bars data ($\sigma = 0.5$ in this case). As data points we used all possible segments of 12.5ms length. This amounts to $N = 389\,510$ data points with $D = 200$ observed variables.

We applied LinCA with $H = 200$ hidden units to the data and used a prior parameter of $\pi = 0.01$ (on average $\pi H = 2.0$ causes per data point as in previous experiments). The approximation parameters for ET we set to $\gamma = 4$ and $H' = 10$ (as in Section 4.2, we found it beneficial to use $\overline{\mathcal{K}}_n$, which increases robustness of learning without changing the complexity; compare Appendix C). During training we annealed according to the cooling schedule in Figure 4B stretched by a factor two to 200 iterations. The initial temperature was set to $T^{\text{init}} = 5.0$ and the final temperature to $T^{\text{final}} = 1.0$. An additional 200 iterations at $T^{\text{final}} = 1$ were used to guarantee full parameter convergence (although

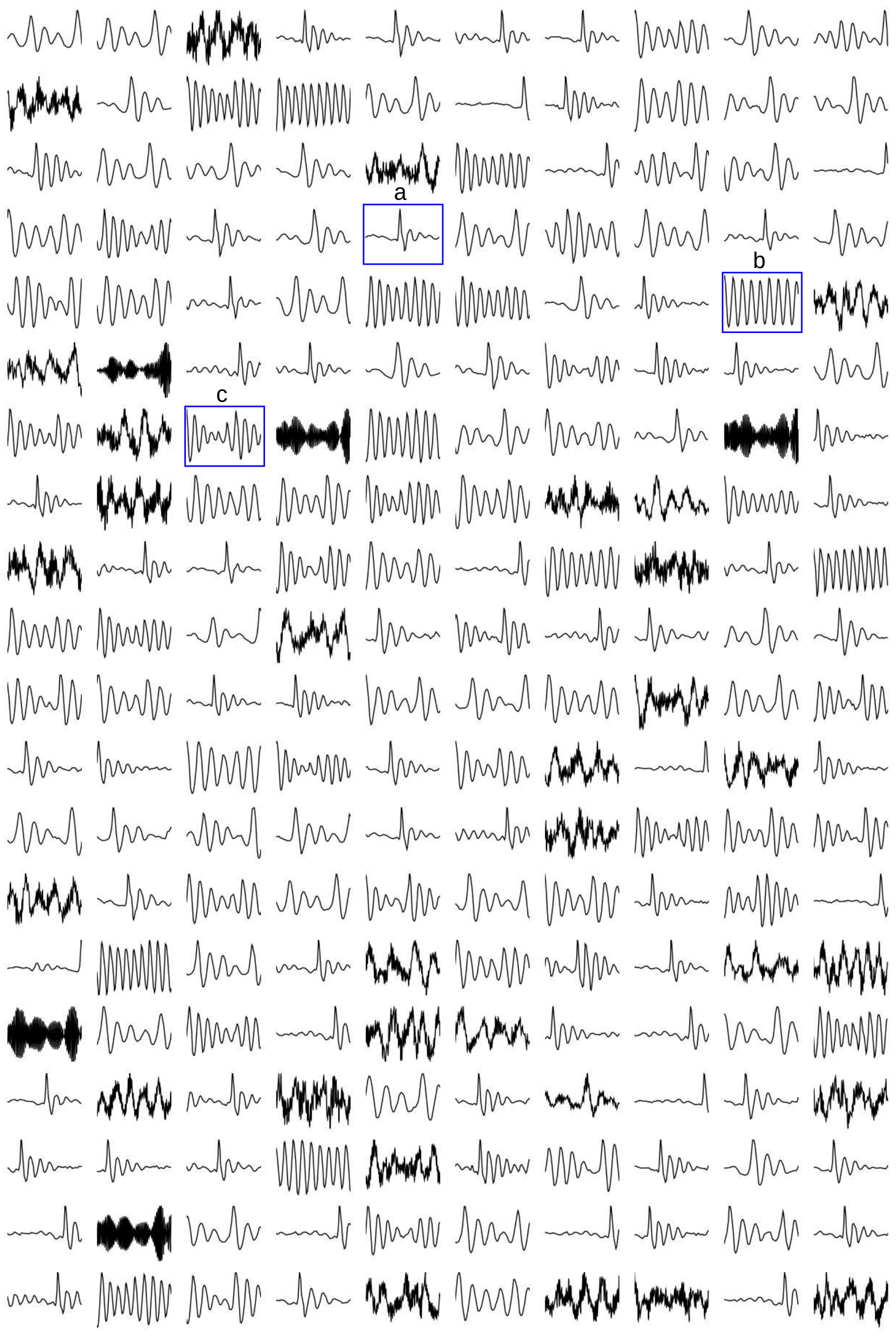


Figure 11: Parameters W after training LinCA on acoustic data. Data points were 12.5ms long intervals of voice recordings. For visualization, the $H = 200$ generative fields were individually scaled. Fields **a-c** are examples of different types of generative fields.

changes after 200 iterations were small; compare Section 4.2.2). Figure 11 shows the learnt parameters W after a typical run. Many of the generative fields (or basis functions) $\vec{W}_h$ are localized in time and frequency space (e.g., field **a**). Others are only localized in frequency space (e.g., field **b**). Still others resemble amplitude modulated waveforms (e.g., field **c**) or represent combinations of low and very high frequencies. We thus recover generative field properties similar to those obtained by other approaches (compare Teh et al., 2003, or Köster and Hyvärinen, 2007). Note, however, that we applied the algorithm to the data without preprocessing such as whitening. As for the previous experiments, no preprocessing except for data scaling was used. This is unlike SC or ICA approaches that usually require specifically preprocessed data. We also ran LinCA on a subset of the $N = 389\,510$ data points. Results with, for example, $N = 100\,000$ showed no significant differences. Similarly, a run of LinCA with $H = 400$ and $\pi H = 2.0$ showed comparable results (although we observed a tendency towards more generative fields with low absolute amplitudes in this case).

The experiments on acoustic data show that LinCA can be trained with ET also for large-scale applications. As the number of preselected candidates H' can be much smaller than the total number of hidden units, ET scales very favorably with H (see discussion in Section 5.2 and Appendix C). For the experiment of Figure 11, ET evaluates just a couple of hundreds of hidden states per data point instead of 2^{200} required for an exact E-step. Without preselection (i.e., $H' = H$) and same γ, the number of hidden states per data point is still larger than 200×10^6 (compare Appendix C), which would by far exceed currently available computational resources.

5. Discussion

We have studied an approximation to EM to train multiple-cause generative models with binary hidden variables. Training in the scheme is based on a candidate preselection to reduce the computational cost of intractable exact EM learning.

5.1 Properties of Expectation Truncation

The approximation scheme introduced in Section 2 and systematically derived in Section 3 is an example of a deterministic approximation to EM. In contrast to exact EM, there is no guarantee that the data likelihood under a given generative model is always increased or remains unchanged. In numerical experiments on a number of different generative models, we recovered, however, very close to optimal parameters for different types of data. This reflects the property of the approximation to become increasingly optimal the more it approaches the likelihood optimum. To quantify the approximation quality, we have, in different applications, monitored quality values, $Q^{(n)}$, for a selection of data points (see Section 4.1.1, Section 4.2.1, and Section 4.3.1). The quality values are themselves measures for the KL-divergence between the exact posterior (given a data point n) and its approximation (compare Section 3). The values for $Q^{(n)}$ as monitored during the experiments show that the KL-divergence for those data points, which are finally taken into account for learning, becomes virtually zero. Further away from the optimum, the approximation of ET is usually poorer than close to the optimum. However, when measured, also the initial iteration steps did increase the data likelihood in numerical experiments.

5.2 Applicability and Complexity

For the ET approximation to work, different requirements for the generative model and the data have to be fulfilled. These requirements have been made explicit in Section 3. For practical reasons, hidden variables with a small number of discrete values are the most salient requirement. In our formulation and in the experiments, we have used binary values, but ET can be applied without modifications to hidden variables with more than two discrete states. For hidden variables with continuous values the E-step can, however, not be evaluated directly (compare Equation 6 or Algorithm 2). Thus, generative models with latents of continuous values cannot be trained by the presented method. By combining ET with other approximation schemes, it could, however, be generalized to latents with continuous values. For this note that the derivation of ET in Section 3 in large parts does not rely on the assumption of discrete latents. Indeed, the derivations would still hold when all sums over $\vec{s}$ in Section 3 were replaced by integrals over the corresponding states. We are then, however, left with integrals over $\vec{s}$ (see Algorithm 2) that have to be evaluated. By applying approximation methods to these integrals over relatively small state spaces, learning algorithms for generative models with latents of continuous values could be derived.

As shown in Section 4, ET can be applied directly to data that can be generated well by combinations of binary causes. In numerical experiments we have seen examples of its application to NMF, to non-linear component superpositions (MCA), and to a form of sparse coding (LinCA). But also the applicability of ET to generative models with binary hidden variables is not without limits. From the inspection of the methods' functioning (compare Section 3) it becomes clear that it is based on the following assumption: for a sufficiently large number of data points the posterior probability mass has to be concentrated in a relatively small subset of the latents' state space. In other words, a large number of data points must be well-explicable by considering few configurations of the potential latent states. If this assumption is not fulfilled, good approximations can only be achieved by considering sums over large sets of states. We could still apply ET but the required approximation parameters would result in computational costs comparable to the ones for exact EM. Models with large numbers of hidden variables would thus not be computationally tractable in such cases. Even if the probability mass of most posteriors is finally concentrated within small regions of the state space, the applicability of ET may still be limited as it additionally requires a mechanism to locate these regions. For the generative models discussed in Section 4, the tractable selection functions (28) and (33) perform well in this respect, and successfully maximize the likelihood and recover close to optimal parameter values. However, for more complicated models the definition of tractable selection functions (or more generally of sets $\mathcal{K}_n$) may be challenging and can limit the applicability of ET. Again, the summation over more states would reduce the problem. For instance, by choosing $\mathcal{K}_n = \mathcal{K}$ (no second variational step) ET could be applied without a preselection process. However, without this crucial reduction of states, the application scale would be much more limited. In summary, applicability of ET for models with binary latents is thus not a question of principal nature but a question about the trade-off between computational cost and approximation quality.

Note that the space of models ET can be applied to is much larger than the space of models with sparsely active binary latents. ET only requires that, on average, few states can represent a data point well. These states do not have to be sparse nor do the hidden variables have to be independent. Examples are, for instance, the generative models discussed in Section 4 but with priors that fix the number of active causes (e.g., always five causes active). The requirements for ET can still be satisfied in such situations. In preliminary numerical experiments using such non-sparse and dependent priors, ET still efficiently increased the likelihood to approximately optimal values.

In general, a reduction of the computational cost using ET is paid for by a reduction of the approximation's quality. Hereby, ET benefits from the fact that the reduction in quality is very limited compared to a vast reduction in required computations (compare Section 4). For the data considered, the computational cost can be reduced by many orders of magnitude with only minimal losses for the approximation's accuracy. For a given set of data, a good approximation setting for ET usually arises naturally. The Bernoulli prior (22), for instance, results in many data points requiring the consideration of at most πH components for good approximations. The overall computational complexity of ET thus depends on the complexity of the data and the generative model used. More specifically, it depends on how the subsets $\mathcal{K}_n$ in (6) are selected. For the generative models and data considered in this paper, a detailed discussion of ET's computational complexity can be found in Appendix C. The crucial advantage of preselection is that the computational cost is split up into essentially two parts (see Equation 38). Importantly, only the preselection part depends hereby on the total number of hidden variables H. The second and computationally more intensive part becomes independent of the total number of latents, depending only on the number of selected candidates. The preselection part is computationally much less costly: for the examples considered here, its complexity scales just linearly with H (also compare Figure 13). ET thus becomes especially efficient for large numbers of hidden variables. An example is the application of $\mathrm{MCA}_{\mathrm{ET}}$ to the data of Section 4.2.2. Another example is the application of LinCA to the data of Section 4.3.2. In the latter case ET computes approximations by evaluating less than 1000 states for $H = 200$ instead of 2^{200} required for an exact E-step.

5.3 Relations to Other Approximation Schemes

A central role for optimal learning and inference in probabilistic models is played by the posterior probability $p(\vec{s}\,|\,\vec{y}^{(n)}, \Theta)$. Computing expectation values w.r.t. to the posterior or computing the posterior directly is usually intractable for multiple-cause models. Approximation schemes find computationally tractable approximations to expectation values w.r.t. $p(\vec{s}\,|\,\vec{y}^{(n)}, \Theta)$ or they approximate the posterior directly. Many examples of such approximations can be found in the literature, and they usually fall into two major classes: sampling methods and deterministic approaches.

Sampling methods are in a sense more general because they usually rely on fewer assumptions than deterministic approaches. Furthermore, there are no principled limits to most of them. In general, the approximations obtained recover the exact solutions in the limit of infinite computational resources. However, sampling methods can be computationally very demanding and may limit applications to relatively small scale problems. In contrast to sampling, deterministic approaches are based on analytical approximations to the posterior or its expectations computed w.r.t. it. By definition, they do in general not find the exact solutions and often rely on particular assumptions about the model they are applied to. However, if these assumptions are fulfilled, they are often computationally much less demanding. For these reasons, deterministic and sampling approaches are often regarded as complementary (see, e.g., MacKay, 2003, or Bishop, 2006, for discussions).

Major and frequently applied examples of deterministic approaches are *variational EM* approaches (e.g., Jordan et al., 1999; Jaakkola, 2000; MacKay, 2003; Bishop, 2006) and *expectation propagation* (EP) approaches (e.g., Minka, 2001; Bishop, 2006). The method of Expectation Truncation discussed in this paper is an example of a variational EM approach. Although introduced as an approximation to expectation values (see Equation 6), we have shown in Section 3 that ET can be derived from variational approximations to exact posterior distributions. Thus, much of the typical properties of variational approaches and their differences to other approximation methods (discussed, e.g., by MacKay, 2003; Bishop, 2006) such as expectation propagation carry over to ET. A crucial difference between ET and standard variational EM is that ET does not use factorizing distributions to approximate the exact posterior. Instead ET uses two variational steps: one that takes the form of a data point selection, and a second that is based on a preselection of hidden variables (see Section 3 for more details). A consequence, for example, for the data discussed in this paper, was that we finally only learned from data points whose KL-divergence between exact and approximated posterior was virtually zero. In numerical simulations this resulted in a virtually optimal parameter recovery. At the same time, the preselection of relevant hidden variables per data point allowed a massive reduction of the computational cost.

Expectation Truncation was motivated by earlier approaches which were developed in the context of non-linear component extraction models (Lücke and Sahani, 2008; Lücke et al., 2009). For efficient learning truncated sums were used to optimize the model parameters (also see Lücke and Sahani, 2007). Optimization was partly performed under additional constraints to correct for imprecise approximations. Furthermore, the approach was developed for specific non-linear generative models and no explicit data point selection was used. As a consequence no clean relation to variational EM could be derived (but see discussion in Lücke and Sahani, 2008). Most significantly, however, no candidate preselection was used. This could result in impractically long computation times already for relatively small numbers of hidden variables. With the ET framework developed in this paper we can interpret the earlier approaches as approximations to special cases of ET. The approximations for MCA$_3$ (Lücke and Sahani, 2008) can thus be regarded as ET approximation without the second variational step, that is, without selecting subsets $\mathcal{K}_n$ of $\mathcal{K}$ (no preselection). Without preselection, the computational cost of learning algorithms can scale unfavorably with the number of hidden units H (cubically in the case of MCA$_3$; compare cases $H' = H$ in Appendix C). Data point selection in MCA$_3$ can be seen as implicitly accomplished by an E-step that considers more terms in the denominator than in the numerator. Likewise, the approximation for occlusive components analysis (OCA; Lücke et al., 2009) can be seen as ET without preselection (instead of implicit data point selection more terms of the truncated sums were used here). Note that instead of omitting the second variational approximation, another special case of ET is obtained if the first variational step is omitted (no data point classification). This amounts to setting $\mathcal{K}$ equal to the whole state space (the sums over $\mathcal{K}$ in Algorithm 2 equal one in this case). Efficient approximations can still be obtained if proper subsets $\mathcal{K}_n$ are selected. On different generative models preliminary experiments showed results similar to the ones reported in Section 4. Depending on the model, the approximations can be much less efficient or much less accurate, however.

An extreme case for selecting subsets $\mathcal{K}_n$ is to select them to contain just one element, $\mathcal{K}_n = \{\vec{s}^{(n)}\}$. In this case the approximated expectation value of a function g in (6) becomes equal to the functions value at $\vec{s}^{(n)}$: $\langle g(\vec{s})\rangle_{q^{(n)}} = g(\vec{s}^{(n)})$. This choice relates ET to energy models (compare, e.g., Teh et al., 2003, or Hyvärinen et al., 2009) and maximum a posteriori (MAP) approximations as, for example, used in the original sparse coding model (Olshausen and Field, 1996). Indeed, if we set $\mathcal{K}$ to be the entire state space and choose $\mathcal{K}_n = \{\vec{s}^{(n)}\}$ with $\vec{s}^{(n)}$ being the MAP estimate of the posterior, we obtain an update rule for basis functions W proportional to $\sum_n \vec{y}^{(n)} \, (\vec{s}^{(n)})^T$ (compare Equation 26). That is, a learning rule for W as used in sparse coding can be obtained (compare Olshausen and Field, 1996, or Olshausen, 2002). If instead of a MAP estimate the scalar product $\vec{s}^{(n)} = \vec{W}_h^T \vec{y}^{(n)}$ is used to select $\vec{s}^{(n)}$, ET for linear generative models can be related to ICA approaches (also compare Section 4.3). Note that MAP estimate and scalar product play the role of the selection functions $\mathcal{S}_h$ in the case of SC and ICA, respectively.

More generally, MAP estimates or scalar products can be starting points to derive less basic selection functions for a generative model. Regarding a MAP estimate, it is likely that a significant amount of posterior mass is located in the vicinity of the MAP state. If the posterior is additionally known to be monomodal, $\mathcal{K}_n$ can be chosen as a region around the MAP estimate. For multi-modal posteriors, generalizations of MAP estimates could be used (see, e.g., Fromer and Globerson, 2009, and references therein). Alternatively, selection functions can be derived by considering the limit of no observation noise for a given model. For data with relatively low amounts of noise, most of the posterior mass will be located close to the estimated state for zero noise (compare the ICA case).

In numerical experiments we have compared the tractable selection functions (28) and (33) with the respective optimal selection function given by $\mathcal{S}_h^{\text{opt}}(\vec{y}^{(n)}) = p(s_h = 1 \,|\, \vec{y}^{(n)}, \Theta)$ (see Section 4.1). For low numbers of hidden variables, $p(s_h = 1 \,|\, \vec{y}^{(n)}, \Theta)$ can still be computed. Numerical experiments using the tractable selection functions, experiments using the optimal selection function, and experiments using exact EM hereby resulted in virtually identical final parameter values close to the optimum. We observed some differences in the convergence behavior. However, these differences were small compared, for example, to differences in using different annealing schemes or annealing parameters.

5.4 Results on Different Data Sets

In numerical experiments we have applied ET to three types of generative models: NMF, MCA, and LinCA. The experiments were aimed at demonstrating the method itself. However, as a by-product, we obtained some results that are closely related to recent developments in component extraction algorithms: The application of the probabilistic NMF algorithm to the MNIST data base showed that, for this data, potentially only very little is gained by considering continuous hidden units instead of binary ones. As could be observed by comparing reconstruction errors obtained for standard vs. probabilistic NMF, continuous hidden variables improved reconstructions by less than 5%. Regarding MCA, we showed that the algorithm derived with ET is competitive to the best performing systems in standard benchmarks. We applied the algorithm to the standard bars test with 10 bars and to a more recent benchmark with 16 bars and larger overlap. In both cases the algorithm extracted all causes with close to 100% reliability (50 successful trials out of 50) provided that we used sufficiently many data points. For fewer data points the algorithm was still competitive but reliability was lower. Applications of the algorithm to image patches and acoustic data showed robust applicability to large scale problems.

For all experiments in the paper we have used deterministic annealing (e.g., Ueda and Nakano, 1998; Sahani, 1999) to avoid the convergence to local optima (compare Section 4). The initial temperature for annealing can be chosen by observing that for a given model and application a critical temperature exists above which all generative fields converge to the same average field (no differentiation to different components). The initial temperature is then chosen to lie below this critical temperature. Note that for Gaussian noise, the annealing temperature changes the standard deviation σ. It is thus closely connected to the noise parameter. For many types of data, component extraction is robust to different values of σ. For instance, for the noiseless bars tests in Section 4.1, Section 4.2, or Section 4.3, $\sigma = 2.0$ was used while the ground-truth would be $\sigma = 0.0$. For some data/model combinations the dependency on σ can be more sensitive, however.

In addition to the data and models discussed in Section 4 we also ran experiments on models with Poisson instead of Gaussian noise, used other types of data including, for example, sound spectrograms instead of sound waveforms, and used data and models in different combinations with different priors. The obtained results were all comparable to the ones reported for the other experiments: the likelihood (when computationally tractable) usually increased to close to optimal values, the generating parameters in successful trials were well recovered, and the reliability to recover the generating causes was high.

In general, the derivation of new learning algorithms based on ET is straightforward. If the data is well-explicable by combinations of binary hidden causes, a generative model should be chosen that appropriately reflects the data generation process. Once the parameter update rules for such a model are derived, the E-step can be computed with ET. This involves the definition of selection functions, the choice of an appropriate constraint for $\mathcal{K}_n$ (compare Equation 7), and the choice of approximation parameters. Depending on the data, a preselection function can take the form of an upper-bound on the joint probability as shown, for example, for NMF in Section 4.1, or it can take the form of a scalar product as for LinCA in Section 4.3. More generally, any discriminative method represents a potential choice for a selection function.

5.5 Conclusion

Motivated by earlier work that discusses the benefits of a candidate preselection (e.g., Körner et al., 1999; Lee and Mumford, 2003; Yuille and Kersten, 2006), we have defined and studied a novel approximation scheme for probabilistic generative models. This scheme is formulated as a deterministic variational EM approximation to maximize the data likelihood under a given generative model. The formulation in terms of a grounded probabilistic approach allowed us to quantify the gain in efficiency that is achievable by preselection. To study the approximation scheme empirically, it has been applied to different types of generative models with different combination rules. In standard benchmarks on artificial data we found that the derived algorithms increased the likelihood to values very close to the optimum, extracted hidden causes with high reliability, and reduced the computational cost potentially by orders of magnitude. We reported quantitative results on data with ground-truth and, where standard benchmarks were available, showed that the derived learning algorithms are competitive with the best performing systems so far. Applications to more realistic data demonstrated robustness and applicability to larger scale problems.

In conclusion, the contribution of the novel method is thus two-fold: (1) it relates the intuitive and frequently discussed benefits of preselection to the grounded framework of an EM-based approximation, and (2) it defines an approximation scheme that allows to efficiently train standard and novel types of generative models.

Acknowledgments. We would like to thank Marc Henniges for his help with the experiment of Figure 11 and Christian Keck for his help in generating animations (Online Appendix). Furthermore, we gratefully acknowledge funding by the German Research Foundation (DFG) in the project LU 1196/4-1, by the Honda Research Institute in Europe (HRI Europe), and by the German Federal Ministry of Education and Research (BMBF) in the project 01GQ0840 (BFNT Frankfurt). Finally, we would like to acknowledge support by the Frankfurt Center for Scientific Computing (CSC Frankfurt).

Appendix A. ET as Variational EM: Detailed Derivations

In Appendix A.1 we discuss the consistency of maximum likelihood parameters of original and truncated generative models. Appendix A.2 discusses details about the classification of data points.

A.1 Necessary Conditions for Global Likelihood Maxima

In Section 3.1 we have seen that $\tilde{F}(q,\Theta)$ in (15) is a lower bound of the likelihood $L(\Theta)$ in (1). If the used variational distributions $q^{(n)}(c;\Theta)$ are good approximations to the exact posteriors $p(c\,|\,\vec{y}^{(n)},\Theta)$ in (14), then $L(\Theta) \approx \tilde{F}(q,\Theta)$ after each E-step. Because of the variational approximation in Section 3.1 the equality $\tilde{F}(q,\Theta) = L(\Theta)$ holds if $\mathcal{M}$ is equal to $\mathcal{M}^{\mathrm{opt}}$ (16) and if the true posterior values in (14) are equal to zero or one for all n. Although the latter condition is fulfilled only in boundary cases, we will, in this section, assume the equality to hold (while keeping in mind that it is almost always an approximation). If the equality holds, $\tilde{F}(q,\Theta)$ in (15) is in its global maximum equal to the likelihood $L(\Theta)$.

Let us, further, assume that there exist parameters Θ^* such that the original generative model reproduces the underlying distribution of the data points, $p(\vec{y}) = p(\vec{y}\,|\,\Theta^*)$. From Section 3.1 we then know that the mixed model with prior (11) and (12) and $\kappa = \tilde{\kappa}$ also reproduces the original distribution for these parameters. Using the mixed model, the data points $\{\vec{y}^{(n)}\}_{n=1,\ldots,N}$ can thus be taken to have been generated by the truncated generative models. That is, the data set can be subdivided into the two disjoint sets $\{\vec{y}^{(n)}\}_{n\in\mathcal{M}^{\mathrm{opt}}}$ and $\{\vec{y}^{(n)}\}_{n\notin\mathcal{M}^{\mathrm{opt}}}$. If $p(\vec{y}\,|\,\Theta^*)$ is the underlying distribution of the whole data set, then $p(\vec{y}\,|\,c=1,\Theta^*)$ and $p(\vec{y}\,|\,c=0,\Theta^*)$ are the underlying distributions of the two disjoint parts (compare Figure 1).

We can approximately recover the distribution $p(\vec{y}\,|\,\Theta^*)$ by (globally) maximizing the data likelihood under the mixed generative model on $\{\vec{y}^{(n)}\}_{n=1,\ldots,N}$. Furthermore, we can recover the distributions $p(\vec{y}\,|\,c=1,\Theta^*)$ and $p(\vec{y}\,|\,c=0,\Theta^*)$ by (globally) maximizing the data likelihoods of the truncated generative models on $\{\vec{y}^{(n)}\}_{n\in\mathcal{M}^{\mathrm{opt}}}$ and $\{\vec{y}^{(n)}\}_{n\notin\mathcal{M}^{\mathrm{opt}}}$, respectively. Let us denote the pa-

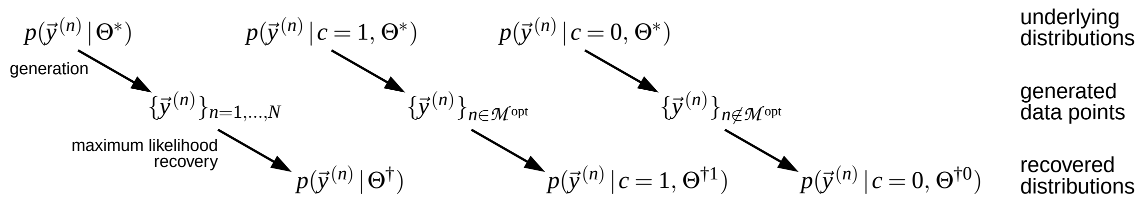

$p(\vec{y}^{(n)}\,|\,\Theta^*)$
$p(\vec{y}^{(n)}\,|\,c=1,\Theta^*)$
$p(\vec{y}^{(n)}\,|\,c=0,\Theta^*)$
underlying distributions
generation
$\{\vec{y}^{(n)}\}_{n=1,\ldots,N}$
$\{\vec{y}^{(n)}\}_{n\in\mathcal{M}^{\mathrm{opt}}}$
$\{\vec{y}^{(n)}\}_{n\notin\mathcal{M}^{\mathrm{opt}}}$
generated data points
maximum likelihood recovery
$p(\vec{y}^{(n)}\,|\,\Theta^\dagger)$
$p(\vec{y}^{(n)}\,|\,c=1,\Theta^{\dagger 1})$
$p(\vec{y}^{(n)}\,|\,c=0,\Theta^{\dagger 0})$
recovered distributions

Figure 12: Recovery of the generating distributions through the original and the truncated generative models. The original distributions can be recovered from the data sets if the corresponding likelihoods are maximized.

rameters recovered by maximizing $L(\Theta)$ by $\Theta^\dagger$, and the parameters recovered by maximizing $L_1(\Theta)$ and $L_0(\Theta)$ by $\Theta^{\dagger 1}$ and $\Theta^{\dagger 0}$, respectively (compare Figure 12). In general, $\Theta^\dagger$, $\Theta^{\dagger 1}$, and $\Theta^{\dagger 0}$ are different. If the variational approximation $\mathcal{M} = \mathcal{M}^{\mathrm{opt}}$ is exact, we know, however, that in the limit of infinitely many data points (and by still assuming $p(\vec{y}) = p(\vec{y}\,|\,\Theta^*)$) applies:

$$p(\vec{y}\,|\,\Theta^*) = p(\vec{y}\,|\,\Theta^\dagger),\ p(\vec{y}\,|\,c{=}1,\Theta^*) = p(\vec{y}\,|\,c{=}1,\Theta^{\dagger 1}),\ \text{ and } p(\vec{y}\,|\,c{=}0,\Theta^*) = p(\vec{y}\,|\,c{=}0,\Theta^{\dagger 0}). \quad (34)$$

The equalities hold because for $N \to \infty$ and $p(\vec{y}) = p(\vec{y}\,|\,\Theta^*)$ it follows from $L(\Theta^*) = L(\Theta^\dagger)$ that $D_{KL}(p(\vec{y}\,|\,\Theta^*), p(\vec{y}\,|\,\Theta^\dagger)) = 0$. As the Kullback-Leibler divergence between two distributions q and

p is zero if and only if the distributions are identical, (34) has to hold. Note, however, that the recovered parameters can still be different from Θ^*. For instance, if there exist transformations $\mathcal{T}$ of Θ^* that do not change the distribution, then any Θ obtained from Θ^* through such a transformation, $\Theta = \mathcal{T}(\Theta^*)$, is a likelihood maximum as well. Such multiple global maxima are the norm rather than the exception. Global maxima of models such as sparse coding (Olshausen and Field, 1996) or independent component analysis (e.g., Comon, 1994) remain global maxima under the exchange of any two basis functions or the negation of any of them.

Let all transformations $\mathcal{T}$ that map the global maxima of $L(\Theta)$ onto itself define a set that we will refer to as the *transformation set*. We say that the set of global maxima is *invariant* under the transformation set. For any two global maxima Θ^* and $\Theta^\dagger$ of $L(\Theta)$, there now exists a member $\mathcal{T}$ of the transformation set such that $\Theta^* = \mathcal{T}(\Theta^\dagger)$. Let us now demand that all global maxima of $L_1(\Theta)$ and $L_0(\Theta)$ are also invariant under the transformation set. Although this will usually be the case, for example, for the exchange of any two basis functions, it is important to state this requirement explicitly as it is not fulfilled in general. If this property is fulfilled, however, we can infer:

$$\begin{aligned}
&\Theta^\dagger \text{ is maximum likelihood solution on } L(\Theta)\\
\Rightarrow\ &\text{There exists } \mathcal{T} \text{ such that } \Theta^\dagger = \mathcal{T}(\Theta^*) \text{ with } \Theta^* \text{ being the generating parameters.}\\
\Rightarrow\ & p(\vec{y}\,|\,c=1,\Theta^*) \text{ is the actual generating distribution of } \{\vec{y}^{(n)}\}_{n\in\mathcal{M}^{\text{opt}}}\\
\Rightarrow\ &\Theta^* \text{ is maximum likelihood solution of } L_1(\Theta)\\
\Rightarrow\ & p(\vec{y}\,|\,c=1,\Theta^*) = p(\vec{y}\,|\,c=1,\mathcal{T}(\Theta^*)) = p(\vec{y}\,|\,c=1,\Theta^\dagger)\\
\Rightarrow\ &\Theta^\dagger \text{ is maximum likelihood solution of } L_1(\Theta).
\end{aligned}$$

Analogously, $\Theta^\dagger$ is also a maximum likelihood solution of $L_0(\Theta)$ if it is a maximum likelihood solution of $L(\Theta)$. For the free-energy (15) this means that at a global maximum of $L(\Theta)$ both $L_1(\Theta)$ and $L_0(\Theta)$ also have a global maximum (under the stated assumptions). A global maximum, for example, in $L_1(\Theta)$ is thus a *necessary* condition for a global maximum in $L(\Theta)$. We have *not* shown that a maximum in $L_1(\Theta)$ is a sufficient condition for a maximum in $L(\Theta)$. Theoretically, $L_1(\Theta)$ might, for instance, not depend on all parameters, or it might have additional global maxima. Finally, note again that the necessary condition only holds under the introduced assumptions. While, for example, the assumption on invariance under transformations $\mathcal{T}$ can exactly be fulfilled (depending on the generative model), the assumptions that the true data distribution can exactly be matched or that the variational approximation in Section 3.1 is exact are in practice almost never fulfilled. The same applies for the assumption of infinitely many data points. All these assumptions can, however, be fulfilled approximately. By (globally) maximizing $L_1(\Theta)$ we can thus expect to recover parameters that maximize $L(\Theta)$ approximately.

A.2 Details of Data Classification

Starting point for choosing $\mathcal{M}$ is the set $\mathcal{M}^{\text{opt}}$ in Equation 16. Setting $\mathcal{M}$=$\mathcal{M}^{\text{opt}}$ would represent the best choice but without ground-truth information, $\mathcal{M}^{\text{opt}}$ can not be computed exactly. We can, however, try to approximate $\mathcal{M}^{\text{opt}}$. To do so, first note that we can compute an expectation value for the size of $\mathcal{M}^{\text{opt}}$. It is given by $N(\mathcal{K}) = N\sum_{\vec{s}\in\mathcal{K}} p(\vec{s}\,|\,\Theta)$. We can now find an approximation to $\mathcal{M}^{\text{opt}}$ by computing the values $q^{(n)}(c=1;\Theta^{\text{old}})$ for all data points, sort them, and take the data points with the $N(\mathcal{K})$ highest values. This would represent a good approximation to $\mathcal{M}^{\text{opt}}$ but it seems that we have gained very little, since we still have to compute the intractable posteriors

$q^{(n)}(c=1;\Theta^{\mathrm{old}}) = p(c=1\,|\,\vec{y}^{(n)},\Theta^{\mathrm{old}})$ for all n data points. Note, however, that with this procedure, the absolute values of $q^{(n)}(c=1;\Theta^{\mathrm{old}})$ are not used for the approximation anymore. All that is required is a pairwise comparison of the data points based on their values $q^{(n)}(c=1;\Theta^{\mathrm{old}})$.

To derive a tractable approximation of the pairwise comparison, consider two data points, $\vec{y}^{(n)}$ and $\vec{y}^{(n')}$, that are neighbors after a sorting according to $q^{(n)}(c=1;\Theta^{\mathrm{old}})$. For arbitrarily many data points and for non-zero noise, the differences between the two data points become arbitrarily small. In particular, it applies for neighboring data points that the difference between the denominators of $q^{(n)}(c=1;\Theta^{\mathrm{old}})$ become arbitrarily small: $\sum_{\vec{s}} p(\vec{y}^{(n)},\vec{s}\,|\,\Theta) \approx \sum_{\vec{s}} p(\vec{y}^{(n')},\vec{s}\,|\,\Theta)$. The same applies for differences between the numerators. However, as the numerators contain just small sums over $\vec{s}$, their values for neighboring data points can be expected to vary more strongly than those of the denominators. We can thus replace the comparison between $q^{(n)}(c=1;\Theta^{\mathrm{old}})$ by a comparison of their numerators $\sum_{\vec{s}\in\mathcal{K}} p(\vec{y}^{(n)},\vec{s}\,|\,\Theta)$. This is an approximation to the pairwise comparison required for exact sorting. In the limit of infinitely many data points this procedure can be expected to result in sets $\mathcal{M}$ that represent good approximations to $\mathcal{M}^{\mathrm{opt}}$.

If we now take preselection into account (compare Section 3.2), the comparison for sorting can be reduced further. For this note that the posterior in (14) is approximated by $q^{(n)}(c=1;\Theta^{\mathrm{old}}) \approx \frac{\sum_{\vec{s}\in\mathcal{K}_n} p(\vec{y}^{(n)},\vec{s}\,|\,\Theta)}{\sum_{\vec{s}} p(\vec{y}^{(n)},\vec{s}\,|\,\Theta)}$. Following the same arguments as above, an approximation of the sorting by comparing the values $q^{(n)}(c=1;\Theta^{\mathrm{old}})$ is given by sorting based on the values $\sum_{\vec{s}\in\mathcal{K}_n} p(\vec{y}^{(n)},\vec{s}\,|\,\Theta)$ for all n. This shows that the selection of N^{cut} data points as introduced in Section 2 (Equations 8 and 9) corresponds to defining a set $\mathcal{M}$ as an approximation to $\mathcal{M}^{\mathrm{opt}}$.

Appendix B. Selection Function for NMF and MCA

We show that $p(s_h=1,\vec{y}^{(n)}\,|\,\Theta)$ is bounded by $\mathcal{S}_h(\vec{y}^{(n)})$ in (28) from above. This implies that $p(s_h=1\,|\,\vec{y}^{(n)},\Theta)$ is bounded by $\frac{\mathcal{S}_h(\vec{y}^{(n)})}{p(\vec{y}^{(n)}\,|\,\Theta)}$ from above. As $p(\vec{y}^{(n)}\,|\,\Theta)$ is independent of h, candidate selection based on the one bound is equivalent to selection based on the other bound. Now, consider the set $\delta_h := \{d\in\{1,\ldots,D\}\,|\,y_d^{(n)} < W_{dh}\}$ and note that if $y_d^{(n)} < W_{dh}$ and $s_h=1$ then $p(y_d^{(n)}\,|\,W_{dh},\sigma) < p(y_d^{(n)}\,|\,\overline{W}_d(\vec{s},W),\sigma)$. This is because $\overline{W}_d(\vec{s},W)$ can only be larger than W_{dh} for non-negative W and therefore the mono-modal Gaussian distribution $p(y_d^{(n)}\,|\,\overline{W}_d(\vec{s},W),\sigma)$ is further away from the maximum value. $p(y_d^{(n)}\,|\,w,\sigma)$ with $w=y_d^{(n)}$ is on the other hand larger or equal to

$p(y_d^{(n)}\,|\,w',\sigma)$ for any other value w'. It follows:

$$\begin{aligned}
p(s_h=1,\vec{y}^{(n)}\,|\,\Theta) &= \sum_{\substack{\vec{s}\\ s_h=1}} p(\vec{y}^{(n)}\,|\,\vec{s},\Theta)\; p(\vec{s}\,|\,\pi)\\
&= \sum_{\substack{\vec{s}\\ s_h=1}} \left(\prod_{d=1}^{D} p(y_d^{(n)}\,|\,\overline{W}_d(\vec{s},W),\sigma)\right) p(\vec{s}\,|\,\pi)\\
&= \sum_{\substack{\vec{s}\\ s_h=1}} \left(\prod_{d\in\delta_h} p(y_d^{(n)}\,|\,\overline{W}_d(\vec{s},W),\sigma)\right)\left(\prod_{d\notin\delta_h} p(y_d^{(n)}\,|\,\overline{W}_d(\vec{s},W),\sigma)\right) p(\vec{s}\,|\,\pi)\\
&\leq \left(\prod_{d\in\delta_h} p(y_d^{(n)}\,|\,W_{dh},\sigma)\right)\left(\prod_{d\notin\delta_h} p(y_d^{(n)}\,|\,y_d^{(n)},\sigma)\right) \sum_{\substack{\vec{s}\\ s_h=1}} p(\vec{s}\,|\,\pi)\\
&= \left(\prod_{d\in\delta_h} p(y_d^{(n)}\,|\,W_{dh}^{\mathrm{ub}},\sigma)\right)\left(\prod_{d\notin\delta_h} p(y_d^{(n)}\,|\,W_{dh}^{\mathrm{ub}},\sigma)\right)\pi\\
&= \pi\, p(\vec{y}^{(n)}\,|\,\vec{W}_h^{\mathrm{ub}},\sigma) \;=:\; \mathcal{S}_h(\vec{y}^{(n)})\,,
\end{aligned}$$

where $W_{dh}^{\mathrm{ub}} = \max\{y_d^{(n)}, W_{dh}\}$ as in Equation 28.

Appendix C. Computational Complexity

The computational cost of an E-step in the ET approximation scheme results from the computation of the selection functions $\mathcal{S}_h$ (Section 2 and Section 4) and the computation of the approximate sufficient statistics (6). For the sufficient statistics we have to compute the joint probabilities $p(\vec{s},\vec{y}^{(n)}\,|\,\Theta)$ for different arguments. For the generative models of NMF, MCA and LinCA (Section 4.1, Section 4.2, and Section 4.3, respectively) the joint probabilities take the form:

$$p(\vec{s},\vec{y}^{(n)}\,|\,\Theta) = \left(\prod_{d=1}^{D} p(y_d^{(n)}\,|\,\overline{W}_d(\vec{s},W),\sigma)\right)\left(\prod_{h=1}^{H} p(s_h\,|\,\pi)\right). \qquad (35)$$

For the first factor we have to compute D terms. Each term involves the computation of $\overline{W}_d(\vec{s},W)$ which is equal to $\sum_h W_{dh}s_h$ in the cases of NMF and LinCA, and equal to $\max_h\{W_{dh}s_h\}$ in the case of MCA. In either case $\overline{W}_d(\vec{s},W)$ can be computed with a cost proportional to the number of non-zero entries in a given vector $\vec{s}$, $\gamma' = |\vec{s}|$. The first factor in (35) is thus computable in times proportional to $D\gamma'$. The computational cost of the second factor can be neglected because just two terms (occurring γ' and $(H-\gamma')$ times) have to be computed.

The complexity of the selection functions, $\mathcal{S}_h$, used for NMF and MCA results from computing D times the conditional probabilities $p\big(y_d^{(n)}\,|\,W_{dh}^{\mathrm{ub}},\sigma\big)$. The functions $\mathcal{S}_h$ have to be computed H times, which amounts to a computational cost proportional to DH (per data point). The selection functions used for the LinCA generative model (33) have the same computational cost. After preselection, the selection function values are used to determine the H' hidden units with the largest values, a computation that can be executed in times proportional to $\big(H + H'\log(H)\big)$ (see, e.g., Lam and Ting, 2000, for references).

In the approximate sufficient statistics (6) we sum over $|\mathcal{K}_n|$ different state vectors $\vec{s}$. If $\mathcal{K}_n$ is given as in Equation 7, it contains $\sum_{\gamma'=0}^{\gamma} \binom{H'}{\gamma'}$ different states with $\binom{H'}{\gamma'}$ counting the states with γ' non-zero entries. The denominator in (6) can therefore be computed in times proportional to $D\,\mathcal{C}(H',\gamma)$, where

$$\mathcal{C}(H',\gamma) := \sum_{\gamma'=0}^{\gamma} \binom{H'}{\gamma'} \gamma' .$$

For NMF, MCA, and LinCA the numerator can in both cases also be computed in times proportional to $D\,\mathcal{C}(H',\gamma)$ due to the special forms of the sufficient statistics. For NMF and LinCA the sufficient statistics $\langle s_i \rangle_{q^{(n)}}$ and $\langle s_i s_j \rangle_{q^{(n)}}$ do not require the computation of additional terms. For MCA the terms $\mathcal{A}^{\rho}_{di}(\vec{s},W)$ (see Equation 30) are of the form:

$$\mathcal{A}^{\rho}_{di}(\vec{s},W) = (s_i W_{di})^{\rho-1} \Big(\sum_h (s_h W_{dh})^{\rho}\Big)^{\frac{1-\rho}{\rho}} . \qquad (36)$$

The last term in (36) can be computed together with $\overline{W}_d(\vec{s},W)$. Further, the computation of the D terms $p(\vec{s},\vec{y}^{(n)}\,|\,\Theta)\,\mathcal{A}_{di}(\vec{s},W)$ in the denominator is required just γ' times for a given $\vec{s}$. The numerator, like the denominator, is thus computable in times proportional to $D\gamma'$.

Selection functions and sufficient statistics have to be computed for each data point (see Algorithm 1), which results in a computational time proportional to:

$$\alpha_1 NDH \;+\; \alpha_2 N(H + H' \log(H)) \;+\; \alpha_3 ND\,\mathcal{C}(H',\gamma) , \qquad (37)$$

where α_1, α_2, and α_3 are constants that describe potentially different weightings of the three terms. Note that if the size of $\mathcal{K}_n$ is increased by adding unit vectors, $\overline{\mathcal{K}}_n = \mathcal{K}_n \cup \{\vec{s}\,|\, \sum_i s_i = 1\}$ (as was done for MCA and LinCA) the scaling behavior (37) remains unchanged (also note that $\overline{\mathcal{K}}_n$ remains a subset of $\mathcal{K}$ as defined in Section 3.2). The usage of $\overline{\mathcal{K}}_n$ instead of $\mathcal{K}_n$ adds another $(H - H') \leq H$ terms to $\mathcal{C}(H',\gamma)$ and consequently a cost of $\alpha_4 ND(H-H') \leq \alpha_4 NDH$ to the whole expression. With a change of factor α_1, the additional cost thus gets absorbed into the first term in (37).

Considering expression (37) note that in our applications, D is always much larger than $\log(H)$. We can therefore neglect the second term such that the computational cost is approximately:[2]

$$C_{ET(H',\gamma)}(N,D,H) := \alpha_1 NDH \;+\; \alpha_3 ND\,\mathcal{C}(H',\gamma) . \qquad (38)$$

The constants α_1 and α_3 depend on the generative model and the specific implementation of the algorithm. The principle scaling behavior remains the same, however. In Figure 13 we have plotted the scaling behavior for ET with $\alpha_1 = \alpha_3 = \frac{1}{2}$. The different graphs show the H dependence of $C_{ET(H',\gamma)}(N,D,H)$ for different values of H' and γ, $D = 100$ and $N = 1000$. Note that different choices of D and N would just globally shift the curves in the double-log plot. The steepest curve in Figure 13 is the one for $\mathrm{ET}(H,H)$, that is, if we choose $H' = H$ and $\gamma' = H$. In this case we drop back to the exact sufficient statistics with exponential computational cost. Note that for this

2. Note that ET also involves partial sorting of the data points once after each E-step. In a typical experiment (and in all of the experiments discussed in the paper), $\log(N)$ is much smaller than DH. The computational cost of discarding data points can therefore be neglected.

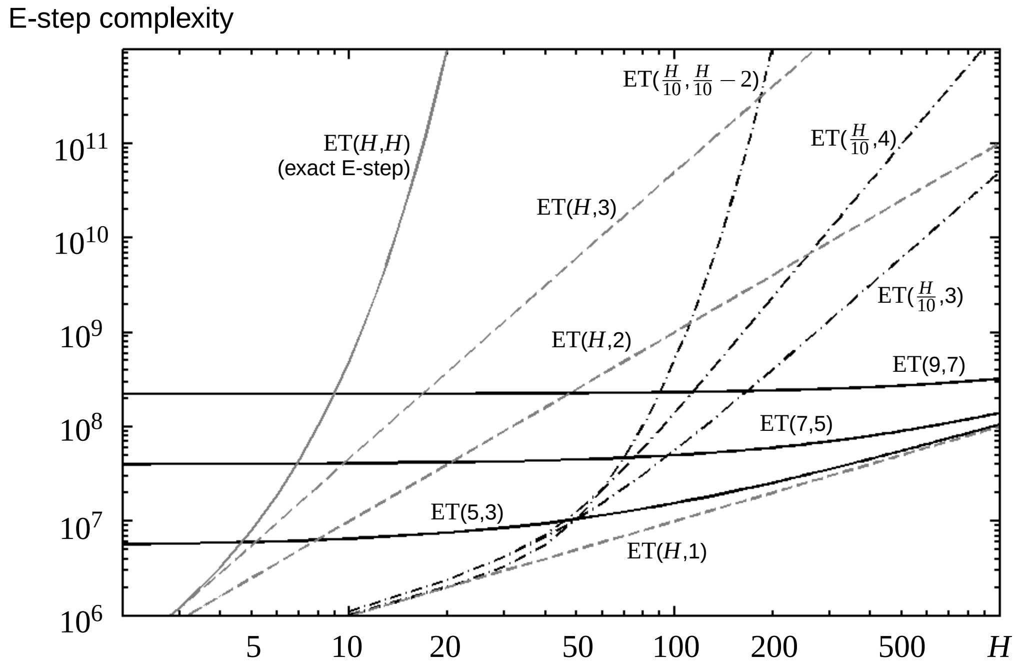


Figure 13: Scaling behavior of ET with (7) for different settings of the parameters H' and γ. The plots show the function (38) and its dependence on H.

and for all other plots with $H' = H$, we set the first term in (38) to zero because no preselection is required. The three dashed grey lines describe the computational costs of $\mathrm{ET}(H,1)$, $\mathrm{ET}(H,2)$, and $\mathrm{ET}(H,3)$. They resemble straight lines because for large H their scaling is proportional to H, H^2, and H^3, respectively. Note that these three approximations are further examples for ET without preselection ($H' = H$). Their scaling behavior relative to the scaling with other parameters can therefore serve as a comparison for the computational gain of candidate preselection. The three solid black lines describe the computational time required for the parameter settings $\mathrm{ET}(5,3)$, $\mathrm{ET}(7,5)$, and $\mathrm{ET}(9,7)$. These curves are flat initially because the first term in (13) (the preselection term) is small compared to the second term which is independent of H. For large H, the preselection term becomes increasingly dominant, however. The three black dashed lines describe the scaling with H for $\mathrm{ET}(\frac{H}{10},3)$, $\mathrm{ET}(\frac{H}{10},4)$, and $\mathrm{ET}(\frac{H}{10},\frac{H}{10}-2)$. The computational cost of $\mathrm{ET}(\frac{H}{10},3)$ is orders of magnitude smaller than the cost for $\mathrm{ET}(H,3)$ but finally scales with the same slope. $\mathrm{ET}(\frac{H}{10},4)$ is computational less expensive than $\mathrm{ET}(H,3)$ for $H \leq 1000$ but will finally become computationally more intensive. If the parameters H' and γ' are both scaled up with H as for $\mathrm{ET}(\frac{H}{10},\frac{H}{10}-2)$, we finally get a scaling behavior similar to that of an exact E-step. Note, however, that even in this case the computational cost remains smaller than that of $\mathrm{ET}(H,3)$ for values of H smaller that about $H = 180$.

As discussed earlier, the computational gain achievable by ET depends on the data and the generative model used. If the data contains sufficiently many data points that are well represented by few active causes, the computational gain is potentially very substantial. For instance for the data used in this paper it was sufficient to use small values for H' and γ. Compared to systems without preselection or an exact E-step, the gain amounts to several orders of magnitude (as, e.g., discussed for $H = 100$ in Section 4.2.2, or for $H = 200$ in Section 4.3.2).

Note that there are usually secondary effects that make learning slower if H increases. E.g., a large numbers of hidden variables usually requires large sets of data points to avoid overfitting. We also observed that longer cooling times are often beneficial for large values of D and/or H. However, ET scales essentially linearly with the number of data points, and the cooling schedule usually has to be increased merely by a factor of $2-8$. The secondary effects do, therefore, only play a role because ET has reduced the potentially exponential cost to a scaling which is close to linear in H.

Appendix D. Classical NMF vs. ET-NMF

The classical NMF belongs to the class of non-negative matrix factorization (NMF) methods, which makes use of non-negative data points, generative fields and sources. In this appendix, we first summarize the classical NMF and then derive the probabilistic version (ET-NMF) using Expectation Truncation.

D.1 Classical NMF

The standard NMF was proposed originally in Lee and Seung (1999) as a parameter-free method for the factorization of data into non-negative generative fields (or 'basis vectors') and source activities. Starting with generative fields $\vec{W}_h = (W_{1h}, \ldots, W_{Dh})^T$ for each source (or cause) s_h contained in the rows of a matrix $W \in \mathbb{R}^{D\times H}$, N data points $\vec{y}^{(n)}$ contained in the columns of a matrix $Y \in \mathbb{R}^{D\times N}$, and the corresponding source activity vectors $\vec{s}^{(n)}$ for each data point contained in the columns of a matrix $S \in \mathbb{R}^{H\times N}$, the NMF factorization tries to find an approximation

$$Y \approx WS \qquad \text{resp., in vector form} \qquad \vec{y}^{(n)} \approx W\vec{s}^{(n)} \tag{39}$$

with non-negative entries for Y, W and S, meaning that the generative fields $\vec{W}_h$ are combined linearly using the source activations $s_h^{(n)}$ to approximate the data vectors $\vec{y}^{(n)}$.

Different cost functions $E(W,S)$ have been proposed for this factorization; here we will consider an Euclidean cost function which we will compare in the next section to a Gaussian-based generative version of NMF used for Expectation Truncation (ET-NMF). To factorize (39), we therefore minimize

$$E(W,S) = ||Y - WS||^2 \tag{40}$$

with respect to W and S. Introducing

$$\langle f(\vec{y}, s_h)\rangle := \frac{1}{N}\sum_{n=1}^{N} f(\vec{y}^{(n)}, s_h^{(n)})$$

for a more compact notation and using the vectorial form of (40), the task is to minimize

$$E(W,S) = \left\langle ||\vec{y} - W\vec{s}||^2 \right\rangle \ , \tag{41}$$

that is, the average Euclidean reconstruction error over all data points $\vec{y}^{(n)}$.

In Lee and Seung (2001), it could be shown that the following parameter-free multiplicative update rules converge to local minima of the cost function (41),

$$s_h^{(n)} \leftarrow s_h^{(n)} \frac{(\vec{W}_h)^T \vec{y}^{(n)}}{(\vec{W}_h)^T (\sum_{h'} \vec{W}_{h'} s_{h'}^{(n)})} \ \text{ and } \ \vec{W}_h \leftarrow \vec{W}_h \odot \frac{\langle \vec{y}\, s_h\rangle}{\sum_{h'} \vec{W}_{h'} \langle s_{h'}\, s_h\rangle}, \tag{42}$$

(the multiplication $\odot$ and division in the second equation of (42) are applied component-wise on vectors). At start, the generative fields and the source activities are initialized randomly with positive values, and then the two equations (42) are applied in alternation until convergence.

The update rules (42) conserve non-negativity and, interestingly, provide a good compromise between speed and ease of implementation. This can be seen by considering directly the gradient descent update equations derived for $\vec{W}_h$ from the minimization of Equation 41,

$$\vec{W}_h \leftarrow \vec{W}_h + \vec{\alpha} \odot \left(\langle \vec{y} s_h \rangle - \sum_{h'} \vec{W}_{h'} \, \langle s_{h'} \, s_h \rangle \right) , \tag{43}$$

with a stepsize vector $\vec{\alpha}$, and modifying an argument from Lee and Seung (2001), diagonally rescaling the variables using a stepsize

$$\vec{\alpha} = \frac{\vec{W}_h}{\sum_{h'} \vec{W}_{h'} \, \langle s_{h'} \, s_h \rangle} \; . \tag{44}$$

Inserting (44) into (43) yields the second, multiplicative update rule from (42), so that in conjunction with the convergence proof of the multiplicative NMF equations from Lee and Seung (2001) we can interprete NMF as a gradient-descent optimization of (41) with $\vec{\alpha}$ from (44) being a good choice of the gradient descent stepsize for the update of the generative fields $\vec{W}_h$.

D.2 ET-NMF

For the generative version of NMF, the choice of the source activities occurs according to the Bernoulli prior (22) from Section 4. As for the classical NMF, we combine the generative fields $\vec{W}_h$ linearly with the source activities s_h using $\overline{W}(\vec{s}, W) := W\vec{s}$, with the difference that the activities are now binary and constitute the hidden variables of the system (remark that now we write s_h instead of $s_h^{(n)}$). In addition, we use a Gaussian noise model such that the probability of a data vector $\vec{y}$ given $\vec{s}$ is defined by $p(\vec{y} \,|\, \vec{s}, \Theta)$ given by Equation 24 from Section 4.1.

At this point we apply the Expectation Maximization formalism introduced in Section 2. From the data likelihood of the generative model resp. the free energy (2) we then get for the W-relevant terms ('$+\ldots$' denotes terms that do not depend on W) an expression similar to Equation 41,

$$E(W) = \left\langle ||\vec{y} - W\vec{s}||^2 \right\rangle_{\mathrm{ET}} + \ldots \, , \tag{45}$$

so that again the average Euclidean reconstruction error should be minimized. The difference is that now the averaging $\langle \ldots \rangle$ runs not only over all data points, but also over all possible source combinations $\vec{s}$ used to generate each data point. Here we wrote $\langle \ldots \rangle_{\mathrm{ET}}$, to express that in addition, we use the Expectation Truncation formalism for the computation of averages. That is, the averaging runs over the subset of data vectors $n \in \mathcal{M}$ and the set of source vectors $\vec{s} \in \mathcal{K}_n$ gained by the source preselection and sparseness assumptions:

$$\langle f(\vec{y}, s_h) \rangle_{\mathrm{ET}} := \sum_{n \in \mathcal{M}} \sum_{\vec{s} \in \mathcal{K}_n} p(\vec{s} | \vec{y}^{(n)}, \Theta) \, f(\vec{y}^{(n)}, s_h) = \sum_{n \in \mathcal{M}} \left\langle f(\vec{y}^{(n)}, s_h) \right\rangle_{q^{(n)}} \; . \tag{46}$$

From here on, all following steps can be considered in analogy to the classical NMF, by replacing $\langle \ldots \rangle \to \langle \ldots \rangle_{\mathrm{ET}}$. The task of ET-NMF is therefore to minimize (45) with respect to W, using the preselected causes and all possible source activation vectors $\vec{s}$ generated by those causes.

It can be easily seen now that the gradient-descent minimization of W and a diagonal rescaling of the stepsize according to

$$\vec{\alpha} = \frac{\vec{W}_h}{\sum_{h'} \vec{W}_{h'} \left\langle s_{h'} s_h \right\rangle_{\mathrm{ET}}}$$

can also be done in analogy to the classical NMF, leading to the multiplicative update equation for the generative fields of ET-NMF,

$$\vec{W}_h \leftarrow \vec{W}_h \odot \frac{\left\langle \vec{y}\, s_h \right\rangle_{\mathrm{ET}}}{\sum_{h'} \vec{W}_{h'} \left\langle s_{h'}\, s_h \right\rangle_{\mathrm{ET}}} \,. \tag{47}$$

The multiplicative, parameter-free equation 47 therefore leads to a near-to-optimal decrease of the ET-NMF model resp. the energy function (45) for positive data points, positive generative fields and Bernoulli-prior-distributed source activations. Since the convergence proof of the original NMF as presented by Lee and Seung (2001) considers convergence of W and S separately, it is straightforward to apply it to the ET-NMF by considering the selected source activations $\vec{s} \in \mathcal{K}_n$ in (46) as given, and applying the classical NMF to an expanded data set which incorporates the given activation vectors together with the data vectors and the probabilities as occurrence frequencies. For a fixed set of $\vec{s}$'s, (47) can then be applied until convergence.

The differences and similarities between the classical NMF and ET-NMF can then be shortly stated as follows. The classical NMF uses data points in fixed association with their corresponding continuous activation vectors and minimizes the Euclidean cost function (41) for both the source activations *and* the generative fields, the latter according to the multiplicative update rule (42). ET-NMF explores a subset of allowed binary source activations for each data point and minimizes in its M-step the Euclidean cost function (45) for the generative fields only, according to the multiplicative update rule (47). The E-step of ET-NMF is based on the truncated expectation values (6) to calculate the averaged quantities in the W update equations according to

$$\left\langle \vec{y}\, s_h \right\rangle_{\mathrm{ET}} = \sum_{n\in\mathcal{M}} \vec{y}^{(n)} \left\langle s_h \right\rangle_{q^{(n)}} \qquad \text{and} \qquad \left\langle s_{h'}\, s_h \right\rangle_{\mathrm{ET}} = \sum_{n\in\mathcal{M}} \left\langle s_{h'}\, s_h \right\rangle_{q^{(n)}} \tag{48}$$

so that the sufficient statistics of the ET-NMF model that have to be computed for the M-step will be given by the first and second order moments $\left\langle s_h \right\rangle_{q^{(n)}}$ and $\left\langle s_{h'}\, s_h \right\rangle_{q^{(n)}}$ of the approximate posterior. The full Expectation Truncation formalism then comprises the calculation of the expectation values $\left\langle s_h \right\rangle_{q^{(n)}}$ and $\left\langle s_{h'}\, s_h \right\rangle_{q^{(n)}}$ (needed for (48)), and afterwards the adjustment of the generative fields according to (47).

Appendix E. Details of Measurements

For the measurements in Section 4, we here give details about the procedure we used to determine if a cause is represented by the parameters of a hidden variable.

Let $\vec{y}_{h'}^{\mathrm{cause}}$ denote a data point showing cause h' without noise (e.g., one noiseless bar). Let further $\vec{s}_h$ denote a hidden state with just one non-zero entry at position h. To determine if cause h' is represented, we compute the approximate posterior probabilities $\tilde{p}(\vec{s}_h \,|\, \vec{y}_{h'}^{\mathrm{cause}}, \Theta)$ for each vector $\vec{s}_h$ using the approximation provided by ET. The hidden unit h with the highest posterior value $\tilde{p}(\vec{s}_h \,|\, \vec{y}_{h'}^{\mathrm{cause}}, \Theta)$ is taken as the unit representing cause h'. We only take all causes to be represented if the mapping from the causes to the representing latents is injective. Additionally, we demand that the mean average error (MAE) between the generating cause parameters $\vec{y}^{\text{cause}}_{h'}$ and the model parameters $\vec{W}_h$ of the representing unit is smaller than 1.0 for each cause. For the data used in this paper, the threshold discounted any generative fields $\vec{W}_h$ with significant traces of more than one cause.

References

P. Berkes, R. E. Turner, and M. Sahani. A structured model of video reproduces primary visual cortical organisation. *PLoS Computational Biology*, 5(9):e1000495, 2009.

C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

N. J. Butko and J. Triesch. Learning sensory representations with intrinsic plasticity. *Neurocomputing*, 70(7-9):1130–1138, 2007.

D. Charles, C. Fyfe, D. MacDonald, and J. Koetsier. Unsupervised neural networks for the identification of minimum overcomplete basis in visual data. *Neurocomputing*, 47(1-4):119–143, 2002.

P. Comon. Independent component analysis, a new concept? *Signal Processing*, 36(3):287–314, 1994.

P. Dayan and R. S. Zemel. Competition and multiple cause models. *Neural Computation*, 7:565 – 579, 1995.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm (with discussion). *Journal of the Royal Statistical Society B*, 39:1–38, 1977.

P. Földiák. Forming sparse representations by local anti-Hebbian learning. *Biological Cybernetics*, 64:165 – 170, 1990.

M. Fromer and A. Globerson. An LP View of the M-best MAP problem. In *Advances in Neural Information Processing Systems 22*, pages 567–575, 2009.

G. E. Hinton, P. Dayan, B. J. Frey, and R. M. Neal. The ‘wake-sleep’ algorithm for unsupervised neural networks. *Science*, 268:1158 – 1161, 1995.

S. Hochreiter and J. Schmidhuber. Feature extraction through LOCOCODE. *Neural Computation*, 11:679 – 714, 1999.

P. O. Hoyer. Non-negative sparse coding. In *Neural Networks for Signal Processing XII: Proceedings of the IEEE Workshop on Neural Networks for Signal Processing*, pages 557 – 565. IEEE, 2002.

A. Hyvärinen, J. Hurri, and P. O. Hoyer. *Natural Image Statistics*. Springer, Heidelberg London New York, 2009.

T. Jaakkola. Tutorial on variational approximation methods. In M. Opper and D. Saad, editors, *Advanced mean field methods: theory and practice.* MIT Press, 2000.

M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37(2):183–233, 1999.

E. Körner, M. O. Gewaltig, U. Körner, A. Richter, and T. Rodemann. A model of computation in neocortical architecture. *Neural Networks*, 12:989 – 1005, 1999.

U. Köster and A. Hyvärinen. A two-layer ICA-like model estimated by score matching. In *Proc. International Conference on Artificial Neural Networks*, LNCS 4669, pages 798–807. Springer, 2007.

T. W. Lam and H.-F. Ting. Selecting the k largest elements with parity tests. *Discrete Appl. Math.*, 101(1-3):187–196, 2000.

D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–91, 1999.

D. D. Lee and H. S. Seung. Algorithm for non-negative matrix factorization. In *Advances in Neural Information Processing Systems*, volume 13, 2001.

T. S. Lee and D. Mumford. Hierarchical Bayesian inference in the visual cortex. *J Opt Soc Am A Opt Image Sci Vis*, 20(7):1434–1448, 2003.

J. Lücke. Hierarchical self-organization of minicolumnar receptive fields. *Neural Networks*, 17/8–9: 1377 – 1389, 2004.

J. Lücke and M. Sahani. Generalized softmax networks for non-linear component extraction. In *Proc. International Conference on Artificial Neural Networks*, LNCS 4668, pages 657 – 667. Springer, 2007.

J. Lücke and M. Sahani. Maximal causes for non-linear component extraction. *Journal of Machine Learning Research*, 9:1227 – 1267, 2008.

J. Lücke and C. von der Malsburg. Rapid processing and unsupervised learning in a model of the cortical macrocolumn. *Neural Computation*, 16:501 – 533, 2004.

J. Lücke, R. Turner, M. Sahani, and M. Henniges. Occlusive components analysis. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 1069–1077, 2009.

D. J. C. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003. Available from http://www.inference.phy.cam.ac.uk/mackay/itila/.

T. P. Minka. Expectation propagation for approximate Bayesian inference. In *UAI '01: Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence*, pages 362–369, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc. ISBN 1-55860-800-1.

R. Neal and G. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In M. I. Jordan, editor, *Learning in Graphical Models*. Kluwer, 1998.

B. A. Olshausen. *Probabilistic Models of the Brain: Perception and Neural Function*, chapter Sparse Codes and Spikes, pages 257 – 272. MIT Press, 2002.

B. A. Olshausen and D. J. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381:607 – 609, 1996.

M. Riesenhuber and T. Poggio. Hierarchical models of object recognition in cortex. *Nature Neuroscience*, 211(11):1019 – 1025, 1999.

M. Sahani. Latent variable models for neural data analysis, 1999. PhD Thesis, Caltech.

E. Saund. A multiple cause mixture model for unsupervised learning. *Neural Computation*, 7:51 – 71, 1995.

M. W. Spratling. Learning image components for object recognition. *Journal of Machine Learning Research*, 7:793 – 815, 2006.

Y. W. Teh, M. Welling, S. Osindero, and G. E. Hinton. Energy-based models for sparse overcomplete representations. *The Journal of Machine Learning Research*, 4(7), 2003.

N. Ueda and R. Nakano. Deterministic annealing EM algorithm. *Neural Networks*, 11(2):271–282, 1998.

R. van Rullen and S. J. Thorpe. Rate coding versus temporal order coding: What the retinal ganglion cells tell the visual cortex. *Neural Computation*, 13(6):1255–1283, 2001.

G. Westphal and R. P. Würtz. Combining feature- and correspondence-based methods for visual object recognition. *Neural Computation*, 21(7):1952–1989, 2009.

A. Yuille and D. Kersten. Vision as Bayesian inference: analysis by synthesis? *Trends in Cognitive Sciences*, 10(7):301–308, 2006.