Honda Research Institute EU
Honda Research Institute Europe GmbH
https://www.honda-ri.de/

Efficient Occlusive Component Analysis

Marc Henniges, Richard E. Turner, Maneesh Sahani, Julian Eggert, Jörg Lücke

2014

Preprint:

This is an accepted article published in Journal of Machine Learning Research. The final authenticated version is available online at: https://doi.org/[DOI not available]

Journal of Machine Learning Research 15 (2014) 2689-2722 Submitted 7/12; Revised 9/13; Published 8/14

Efficient Occlusive Components Analysis

Marc Henniges HENNIGES@FIAS.UNI-FRANKFURT.DE
Frankfurt Institute for Advanced Studies
Goethe-University Frankfurt, 60438 Frankfurt, Germany

Richard E. Turner RET26@CAM.AC.UK
Department of Engineering
University of Cambridge, Cambridge, CB2 1PZ, UK

Maneesh Sahani MANEESH@GATSBY.UCL.AC.UK
Gatsby Computational Neuroscience Unit
University College London, London, WC1N 3AR, UK

Julian Eggert JULIAN.EGGERT@HONDA-RI.DE
Honda Research Institute Europe GmbH
63073 Offenbach am Main, Germany

Jörg Lücke JOERG.LUECKE@UNI-OLDENBURG.DE
Cluster of Excellence Hearing4all and Faculty VI
University of Oldenburg, 26115 Oldenburg, Germany
Electrical Engineering and Computer Science
Technical University Berlin, 10587 Berlin, Germany
Frankfurt Institute for Advanced Studies (previous affiliation)
Goethe-University Frankfurt, 60438 Frankfurt, Germany

Editor: Daniel Lee

Abstract

We study unsupervised learning in a probabilistic generative model for occlusion. The model uses two types of latent variables: one indicates which objects are present in the image, and the other how they are ordered in depth. This depth order then determines how the positions and appearances of the objects present, specified in the model parameters, combine to form the image. We show that the object parameters can be learned from an unlabeled set of images in which objects occlude one another. Exact maximum-likelihood learning is intractable. Tractable approximations can be derived, however, by applying a truncated variational approach to Expectation Maximization (EM). In numerical experiments it is shown that these approximations recover the underlying set of object parameters including data noise and sparsity. Experiments on a novel version of the bars test using colored bars, and experiments on more realistic data, show that the algorithm performs well in extracting the generating components. The studied approach demonstrates that the multiple-causes generative approach can be generalized to extract occluding components, which links research on occlusion to the field of sparse coding approaches.

Keywords: generative models, occlusion, unsupervised learning, sparse coding, expectation truncation

©2014 Marc Henniges, Richard E. Turner, Maneesh Sahani, Julian Eggert and Jörg Lücke.

1. Introduction

A key problem in image analysis is to learn the shape and form of objects directly from unlabeled data. Many approaches to this unsupervised learning problem have been motivated by the observation that, although the number of objects appearing across all images is vast, the number appearing in any one image is far smaller. This property, a form of sparsity, has motivated a number of algorithms including sparse coding (SC; Olshausen and Field, 1996) and non-negative matrix factorization (NMF; Lee and Seung, 1999) with its sparse variants (e.g., Hoyer, 2004). These approaches can be framed as latent-variable models, where each possible object, or part of an object, is associated with a latent variable controlling its presence or absence in a given image. Any individual "hidden cause" is rarely active, corresponding to the small number of objects present in any one image. Despite this plausible motivation, SC or NMF make severe assumptions which coarsely approximate the physical process by which images are produced. Perhaps the most crucial assumption is that in the underlying latent variable models, objects or parts thereof, combine *linearly* to form the image. In real images the combination of individual objects depends on their relative distance from the camera or eye. If two objects occupy the same region in planar space, the nearer one occludes the other, i.e., the hidden causes non-linearly compete to determine the pixel values in the region of overlap.

In this paper we extend multiple-causes models such as SC or NMF to handle occlusion. The idea of using many hidden "cause" variables to control the presence or absence of objects is retained, but these variables are augmented by another set of latent variables which determine the relative depth of the objects, much as in the z-buffer employed by computer graphics. In turn, this enables the simplistic linear combination rule to be replaced by one in which nearby objects occlude those that are more distant. One of the consequences of moving to a richer, more complex model is that inference and learning become correspondingly harder. One of the main contributions of this paper is to show how to overcome these difficulties.

The problem of occlusion has been addressed in different contexts (Jojic and Frey, 2001; Williams and Titsias, 2004; Fukushima, 2005; Eckes et al., 2006; Lücke et al., 2009; LeRoux et al., 2011; Tajima and Watanabe, 2011). Probabilistic 'sprite' models (e.g., Jojic and Frey, 2001; Williams and Titsias, 2004) assign pixels in multiple images taken from the same scene to a fixed number of image layers. The approach is most frequently applied to automatically remove foreground and background objects. Those models are in many aspects more general than the approach discussed here. However, in contrast to our approach, they model data in which objects maintain a fixed position in depth relative to the other objects. Other approaches study occlusion in the context of neural network models (Tajima and Watanabe, 2011) or generalized versions of restricted Boltzmann machines (RBMs) which incorporate occlusion (LeRoux et al., 2011). This paper takes a new and different approach which can be regarded as a generalization of sparse coding to model occlusion.

2. A Generative Model for Occlusion

The occlusion model contains three important elements. The first is a set of variables which controls the presence or absence of objects in a particular image (this part will be analogous,

e.g., to NMF or sparse coding). The second is a variable which controls the relative depths of the objects that are present. The third is the combination rule which describes how active objects which are closer occlude more distant ones. The second and third part are the distinguishing features of the model. They describe how values of observed variables are determined by the occlusion non-linearity given a set of hidden variables. While the occlusion model will be applicable to the same data as NMF or sparse coding, and while efficient inference and learning will require sparsity, explicitly modeled occlusions will define solutions different from these models with linear combination rules. Furthermore, more general features per observed variable such as color vectors can be taken into account.

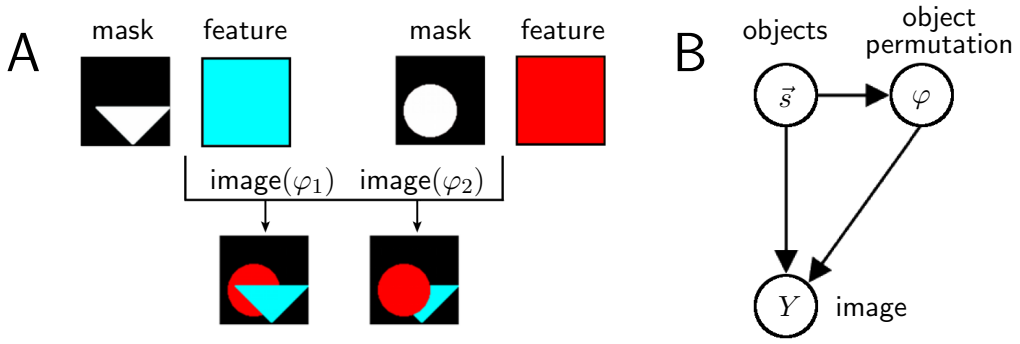


Figure 1: **A** Illustration of how object masks and features combine to generate an image. If two objects are randomly chosen ($|\vec{s}| = 2$), two different images with two different depth-orders (denoted by φ_1 and φ_2) can be generated. **B** Graphical model of the generation process with hidden variables $\vec{s}$ (object presence) and φ (depth permutation).

To model the presence or absence of objects we use H binary hidden variables $s_1, \ldots, s_H$. We assume that the presence of one object is independent of the presence of the others and, for simplicity, we also assume that each object is equally likely to be present in an image *a priori* (we refer to this probability by π). The probability for the presence and absence of objects is given by

$$p(\vec{s}\,|\,\pi) \;=\; \prod_{h=1}^{H} \text{Bernoulli}(s_h;\pi) \;=\; \prod_{h=1}^{H} \pi^{s_h}\,(1-\pi)^{1-s_h}. \tag{1}$$

Objects in a real image can be ordered by their depth and it is this ordering which determines how the objects occlude each other in regions of overlap. The depth-ordering is captured in the model by associating the active objects with a permutation. We randomly and uniformly choose a member φ of the set $\mathcal{G}(|\vec{s}|)$ which contains all permutation functions $\varphi : \{\tilde{h}_1, \ldots, \tilde{h}_{|\vec{s}|}\} \rightarrow \{1, \ldots, |\vec{s}|\}$, with $|\vec{s}| = \sum_h s_h$, where $\tilde{h}_1$ to $\tilde{h}_{|\vec{s}|}$ are the indices of the non-zero entries of $\vec{s}$. More formally, the probability of φ given $\vec{s}$ (see Figure 1B) is defined by

$$p(\varphi\,|\,\vec{s}) = \tfrac{1}{|\vec{s}|!} \quad \text{with} \quad \varphi \in \mathcal{G}(|\vec{s}|)\,. \tag{2}$$

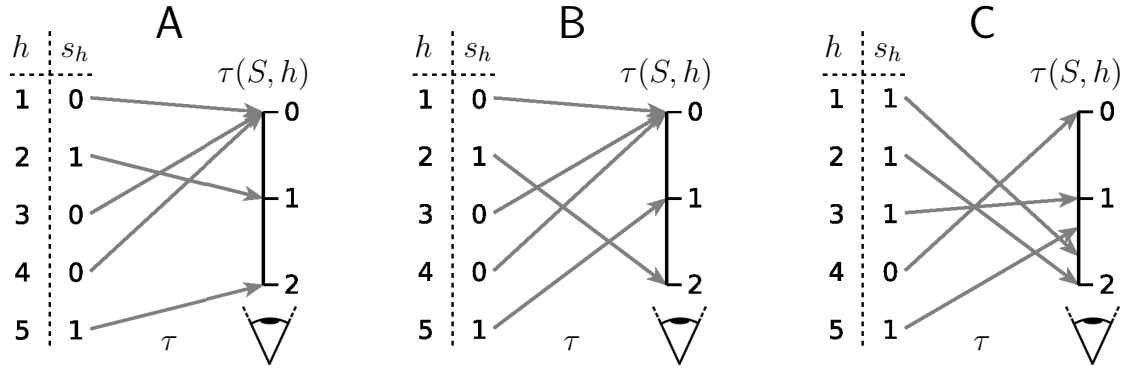


Figure 2: Visualization of the mapping $\tau(S) : \{1, \ldots, H\} \rightarrow [0, 2]$ which represents different permutations of objects in depth. The eye at the bottom illustrates the position of the observer. **A** and **B** show the two possible mappings if two causes are present. **C** shows one of the 24 mappings if four causes are present.

Note that we could have defined the prior over the order in depth (Equation 2) independently of $\vec{s}$, by choosing from $\mathcal{G}(H)$ with $p(\varphi) = \frac{1}{H!}$. But then, because the depth of absent objects ($s_h = 0$) is irrelevant, no more than $|\vec{s}|!$ distinct choices of φ would have resulted in different images.

The final stage of the generative model describes how to produce the image given a selection of active causes and an ordering in relative depth of these causes. One approach would be to choose the closest object and to set the image equal to the feature vector associated with this object. However, this would mean that every image generated from the model would comprise just one object: the closest. What is missing from this description is a notion of the extent of an object and the fact that it might only contribute to a subset of pixels in an image. For this reason, our model contains two sets of object parameters. One set of parameters, $W \in \mathbb{R}^{H \times D}$, describes whether an object contributes to a pixel and the strength of that contribution (D is the number of pixels). The vector $(W_{h1}, \ldots, W_{hD})$ is therefore described as the *mask* of object h. If an object is highly localized, this vector will contain many zero elements. The other set of parameters, $T \in \mathbb{R}^{H \times C}$, represents the features of the objects. We define one vectorial feature per object h, $\vec{T}_h \in \mathbb{R}^C$, describing, for instance, the object's RGB color ($C = 3$ in that case). Figure 1A illustrates the combination of masks and features, and Figure 1B shows the graphical model of the generation process.

Let us formalize how an image is generated given the parameters W and T and given the hidden variables $S = (\vec{s}, \varphi)$. To further abbreviate the notation, we will denote all the model's parameters by $\Theta = (W, T, \pi, \sigma)$. We define the generation of a noiseless image $\vec{\mathcal{T}}(S, \Theta)$ to be given by the following equations:

$$\vec{\mathcal{T}}_d(S,\Theta) \;=\; W_{h_o d}\,\vec{T}_{h_o} \qquad \text{where} \quad h_o \;=\; \text{argmax}_h\{\tau(S,h)\,W_{hd}\}\,, \qquad \tau(S,h) = \begin{cases} 0 & \text{if } s_h = 0 \\ \frac{3}{2} & \text{if } s_h = 1 \text{ and } |\vec{s}| = 1 \\ \frac{\varphi(h)-1}{|\vec{s}|-1} + 1 & \text{otherwise} \end{cases} \tag{3}$$

In Equation 3 the order in depth is represented by the mapping τ which intuitively can be thought of as the relative proximity of the objects. The form of this mapping has been chosen to facilitate later algebraic steps. To illustrate the combination rule of Equation 3 and the mapping τ consider Figure 1A and Figure 2. Let us assume that the mask values W_{hd} are zero or one (although we will later also allow for continuous values). As depicted in Figure 1A an object h with $s_h = 1$ occupies all image pixels with $W_{hd} = 1$ and does not occupy pixels with $W_{hd} = 0$. For all pixels with $W_{hd} = 1$ the vector $\vec{T}_h$ sets the pixels' values to a specific feature, e.g., to a specific color. The function τ maps all causes h with $s_h = 0$ to zero while all other causes are mapped to values within the interval $[1, 2]$ (see Figure 2). In this way, it assigns a proximity value $\tau(S, h) > 0$ to each present object. For a given pixel d the combination rule in Equation 3 simply states that of all objects with $W_{hd} = 1$, the most proximal is used to set the pixel property. The interval [1,2] represents a natural choice for proximity values, but any interval with boundaries greater zero would result in an equivalent generative process.

Given the latent variables and the noiseless image $\vec{\mathcal{T}}(S, \Theta)$, we take the observed variables $Y = (\vec{y}_1, \ldots, \vec{y}_D)$ to be drawn independently from a Gaussian distribution, i.e.,

$$p(Y \,|\, S, \Theta) \;=\; \prod_{d=1}^{D} p(\vec{y}_d \,|\, \vec{\mathcal{T}}_d(S, \Theta)), \qquad p(\vec{y} \,|\, \vec{t}\,) \;=\; \mathcal{N}(\vec{y}; \vec{t}, \sigma^2 \mathbb{1})\,. \tag{4}$$

Equations 1 to 4 represent a model for image generation that incorporates occlusion. We will refer to the model as the *Occlusive Components Analysis* (OCA) generative model.

3. Maximum Likelihood

One approach to learning the parameters $\Theta = (W, T, \pi, \sigma)$ of this model from data $\mathcal{Y} = \{Y^{(n)}\}_{n=1,\ldots,N}$ is to use maximum likelihood learning, that is,

$$\Theta^* \;=\; \text{argmax}_\Theta\{\mathcal{L}(\Theta)\} \quad \text{with} \quad \mathcal{L}(\Theta) \;=\; \log\left(p(Y^{(1)}, \ldots, Y^{(N)} \,|\, \Theta)\right). \tag{5}$$

However, as there is usually a large number of objects that can potentially be present in the training images, and since the likelihood involves summing over all combinations of objects and associated orderings, the computation of Equation 5 is typically intractable. More concretely, given H components the number of different sets of objects that may be present scales with 2^H. Occlusion adds additional complexity: for any subset of size γ' of the H objects that may be present there are $\gamma'!$ different depth orders. Formally, the total number of hidden states to be considered is given by

$$\text{States}_{\text{exact}}(H) \;=\; \sum_{\gamma'=0}^{H} \binom{H}{\gamma'} \gamma'!\,. \tag{6}$$

The need to consider depth-order to model occlusion means that the number of hidden states scales super-exponentially with the number of potential components H. Moreover, even if this computational tractability problem can be overcome, optimization of the likelihood is made problematic by an analytical intractability arising from the fact that the occlusion non-linearity is non-differentiable. The following section describes how to side-step both

of these intractabilities within the standard Expectation Maximization (EM) formalism for maximum likelihood learning. First, we will describe how the analytical intractability may be avoided using an approximation that softens the occlusion non-linearity, and which therefore allows parameter update equations (M-step equations) to be derived. Second, we will describe how the computational intractability can be addressed by leveraging the sparsity of visual scenes to reduce the space of solutions entertained by the posterior.

To find the maximum-likelihood parameters Θ^*, at least approximately, we use the EM formalism in the form used by Neal and Hinton (1998) and introduce the free-energy function $\mathcal{F}(\Theta, q)$ which is a function of Θ and of an unknown distribution $q(S^{(1)}, \ldots, S^{(N)})$ over the hidden variables. $\mathcal{F}(\Theta, q)$ is a lower bound of the likelihood $\mathcal{L}(\Theta)$. Approximations introduced later on can be interpreted as constraining the function q to lie within a specified class. In the model described above each image is assumed to be drawn independently and identically from an underlying distribution, $q(S^{(1)}, \ldots, S^{(N)}) = \prod_n q_n(S^{(n)}, \Theta')$, which results in the free-energy

$$\mathcal{F}(\Theta, q) \;=\; \sum_{n=1}^{N} \Big[\sum_{S} \; q_n(S\,; \Theta') \;\; \Big[\log\big(p(Y^{(n)} \,|\, S, \Theta)\big) + \log\big(p(S \,|\, \Theta)\big) \Big] \Big] + \mathbf{H}[q]\,, \quad (7)$$

where the function $\mathbf{H}[q] = -\sum_n \sum_S q_n(S\,; \Theta') \log(q_n(S\,; \Theta'))$ (the Shannon entropy) is independent of Θ. Note that $\sum_S$ in Equation 7 sums over all possible states of $S = (\vec{s}, \varphi)$, i.e., over all binary vectors and all associated permutations in depth, so that the number of terms in the sum is given by Equation 6. These large sums are the source of the computational intractability. In the EM scheme, $\mathcal{F}(\Theta, q)$ is maximized alternately with respect to the distribution q in the E-step (while the parameters Θ are kept fixed) and with respect to parameters Θ in the M-step (while q is kept fixed). Θ' refers to the model parameters of the previous iteration of the algorithm. At the end of the M-step, we thus set $\Theta' \leftarrow \Theta$. Each EM iteration increases the free-energy or leaves it unchanged. If q is unconstrained (or if any constraints imposed allow it) then the optimal setting of q in the E-step is given by the posterior distribution over the hidden states at the current parameter settings $q_n(S; \Theta') = p(S \,|\, Y^{(n)}, \Theta')$. In this case, each EM step increases the likelihood or leaves it unchanged and this process converges to a (local) maximum of the likelihood.

3.1 M-Step Equations

The M-step of EM, in which the free-energy, $\mathcal{F}$, is optimized with respect to the parameters, is usually derived by taking derivatives of $\mathcal{F}$ with respect to the parameters. Unfortunately, this standard procedure is not directly applicable because the occlusive combination rule in Equation 3 is not differentiable. However, it is possible to soften the combination rule using the differentiable approximation

$$\vec{\mathcal{T}}^{\rho}{}_d(S, \Theta) \;:=\; \frac{\sum_{h=1}^{H} (\tau(S,h)\, W_{hd})^{\rho}\, W_{hd}\, \vec{T}_h}{\sum_{h=1}^{H} (\tau(S,h)\, W_{hd})^{\rho}}, \quad (8)$$

which becomes equal to the combination rule in Equation 3 as $\rho \to \infty$. Note that for the softened combination rule with small values of ρ, the choice of the interval for the proximity values (Equation 3) can now have an effect. According to Equation 8 the hidden

states combine in the sense of a softmax operation, and different interval boundaries for the proximity values $\tau(S,h)$ change how strongly the closest cause dominates the others. Such effects can be counteracted by choosing corresponding finite values for ρ, however. For large values of ρ, differences due to different intervals become negligible again.

$\vec{\mathcal{T}}^{\rho}{}_d(S,\Theta)$ is differentiable w.r.t. the parameters W_{hd} and T_h^c (with $c \in \{1,\ldots,C\}$). For large ρ, the derivatives can be well approximated as follows:

$$\begin{array}{l} \frac{\partial}{\partial W_{id}} \vec{\mathcal{T}}^{\rho}{}_d(S,\Theta) \approx \mathcal{A}^{\rho}_{id}(S,W)\, \vec{T}_i, \\ \frac{\partial}{\partial T_i^c} \vec{\mathcal{T}}^{\rho}{}_d(S,\Theta) \approx \mathcal{A}^{\rho}_{id}(S,W)\, W_{id}\, \vec{e}_c, \end{array} \quad \text{with} \quad \begin{array}{l} \mathcal{A}^{\rho}_{id}(S,W) := \frac{(\tau(S,i)\, W_{id})^{\rho}}{\sum_{h=1}^{H} (\tau(S,h)\, W_{hd})^{\rho}}, \\ \mathcal{A}_{id}(S,W) := \lim\limits_{\rho\to\infty} \mathcal{A}^{\rho}_{id}(S,W)\,, \end{array} \tag{9}$$

where $\vec{e}_c$ is a unit vector in feature space with entry equal one at position c and zero elsewhere. The approximations on the left-hand-side above become equalities for $\rho \to \infty$. Given the approximate combination rule in Equation 9, we can compute approximations to the derivatives of $\mathcal{F}(\Theta, q)$. For large values of ρ the following holds (see Appendix B):

$$\frac{\partial}{\partial W_{id}} \mathcal{F}(\Theta,q) \approx \sum_{n=1}^{N} \left[\sum_{S} q_n(S\,;\Theta') \left(\frac{\partial}{\partial W_{id}} \vec{\mathcal{T}}^{\rho}{}_d(S,\Theta) \right)^T \vec{f}\left(\vec{y}^{\,(n)}, \vec{\mathcal{T}}^{\rho}{}_d(S,\Theta) \right) \right], \tag{10}$$

$$\frac{\partial}{\partial T_i^c} \mathcal{F}(\Theta,q) \approx \sum_{n=1}^{N} \left[\sum_{S} q_n(S\,;\Theta') \sum_{d=1}^{D} \left(\frac{\partial}{\partial T_i^c} \vec{\mathcal{T}}^{\rho}{}_d(S,\Theta) \right)^T \vec{f}\left(\vec{y}^{\,(n)}, \vec{\mathcal{T}}^{\rho}{}_d(S,\Theta) \right) \right], \tag{11}$$

$$\text{where} \quad \vec{f}(\vec{y}^{\,(n)}, \vec{t}\,) := \frac{\partial}{\partial \vec{t}} \log\left(p(\vec{y}^{\,(n)} \,|\, \vec{t}\,) \right) = -\sigma^{-2}\, (\vec{y}^{\,(n)} - \vec{t}\,).$$

Setting the derivatives in Equations 10 and 11 to zero and inserting Equations 9 yields the following necessary conditions for a maximum of the free-energy that hold in the limit $\rho \to \infty$:

$$W_{id} = \frac{\sum\limits_{n} \langle \mathcal{A}_{id}(S,W) \rangle_{q_n}\, \vec{T}_i^T\, \vec{y}_d^{\,(n)}}{\sum\limits_{n} \langle \mathcal{A}_{id}(S,W) \rangle_{q_n}\, \vec{T}_i^T\, \vec{T}_i}\,, \quad \vec{T}_i = \frac{\sum\limits_{n}\sum\limits_{d} \langle \mathcal{A}_{id}(S,W) \rangle_{q_n}\, W_{id}\, \vec{y}_d^{\,(n)}}{\sum\limits_{n}\sum\limits_{d} \langle \mathcal{A}_{id}(S,W) \rangle_{q_n}\, (W_{id})^2}\,. \tag{12}$$

Note that Equations 12 are not straightforward update rules. However, we can use them in the fixed-point sense and approximate the parameters which appear on the right-hand-side of the equations using the values from the previous iteration. For the update note that due to the multiplication of the weights and the mask, $W_{hd}\,\vec{T}_h$ in Equation 3, there is degeneracy for the object parameters: given h, the combination $\vec{\mathcal{T}}_d$ remains unchanged for the operation $\vec{T}_h \to \vec{T}_h/\varrho$ and $W_{hd} \to \varrho\, W_{hd}$ with $\varrho \neq 0$. This transformation does not leave the selection of the closest object unchanged (selection of h_o in Equation 3) because the values of W_{hd} are not binary. To remove the degeneracy and to keep the values of W_{hd} close to zero or one, we rescale after each EM iteration as follows:

$$W_{hd}^{\text{new}} = W_{hd} / \overline{W}_h, \quad \vec{T}_h^{\text{new}} = \overline{W}_h \vec{T}_h,$$

$$\text{where} \quad \overline{W}_h = \frac{1}{|\mathcal{I}|} \sum_{d\in\mathcal{I}} W_{hd} \quad \text{with} \quad \mathcal{I} = \{d \,|\, |W_{hd}| > \alpha\} \quad \text{where} \quad \alpha \in \mathbb{R}.$$

The use of $\overline{W}_h$ instead of, e.g., $W_h^{\max} = \max_d\{W_{hd}\}$ is advantageous for some data, although for many other types of data $W_h^{\max}$ works equally well. Through the influence of the scaling on the selection of closest objects, small values of W_{hd} tend to be suppressed for larger values of α and converge to zero. In general, we find the algorithm to avoid local optima more frequently if we initialize α at a small value and then slowly increase it over the EM iterations to a value near $\frac{1}{2}$. The index set $\mathcal{I}$ thus contains all entries W_{hd} at first, and only later considers exclusively entries with higher values for normalization. In this way, smaller values are suppressed only when the algorithm is already closer to an optimum than it is in the beginning of learning.

If the derivatives of the free-energy in Equation 7 w.r.t. to σ (data noise) and π (appearance frequency) are set to zero, we obtain through straightforward derivations (see Appendix B) the following two remaining update rules:

$$\sigma^{\text{new}} = \sqrt{\frac{1}{NDC}\sum_{n=1}^{N}\left\langle\sum_{d=1}^{D}\sum_{c=1}^{C}\left(y_{dc}^{(n)} - \mathcal{T}_{dc}(S,\Theta)\right)^2\right\rangle_{q_n}}\,, \tag{13}$$

$$\pi^{\text{new}} = \frac{1}{HN}\sum_{n=1}^{N}\langle|\vec{s}|\rangle_{q_n}\,. \tag{14}$$

3.2 E-Step Equations

The crucial quantities that have to be computed for update Equations 12 to 14 are expectation values w.r.t. the variational distributions $q_n(S\,;\Theta')$ in the form

$$\langle g(S,\Theta)\rangle_{q_n} \;=\; \sum_S q_n(S\,;\Theta')\,g(S,\Theta)\,, \tag{15}$$

where $g(S,\Theta)$ are functions that depend on the latent state and potentially the model parameters. The optimal choice for $q_n(S\,;\Theta')$ is the exact posterior, $q_n(S\,;\Theta') = p(S\,|\,Y^{(n)},\Theta')$, which is given by Bayes' rule, $p(S\,|\,Y^{(n)},\Theta') = \frac{p(Y^{(n)}\,|S,\Theta')\,p(S\,|\,\Theta')}{\sum_{S'}p(Y^{(n)}\,|S',\Theta')\,p(S'|\,\Theta')}$, with prior and noise distributions given by the OCA generative model in Equations 1 to 4. Unfortunately, the computation of the expectations or *sufficient statistics* in Equation 15 becomes computationally intractable in this case because of the large number of states that have to be considered (see Equation 6). To derive tractable approximations, we can, however, make use of typical properties of visual scenes: in any given scene the number of objects which are present is far smaller than the set of all objects that can potentially be present in the scene. As such, the sum over all states in Equation 15 is typically dominated by only a few terms. More specifically, components which are compatible with the observed image are the only ones to make a significant contribution to this sum, whilst components which are incompatible make only a negligible contribution. Consequently, a good approximation to the expectation values in Equation 15 can be obtained by identifying the states which carry high posterior mass and retaining only these states.

It has recently been shown (Lücke and Eggert, 2010) that this general idea (see Yuille and Kersten, 2006) can be considered as approximate variational EM. When the variational distribution q in Equation 7 is imperfectly optimized, or optimized within a constrained

space of functions, then the resulting variational EM algorithm is no longer guaranteed to converge to a local maximum of the likelihood. However, it still increases a lower bound on the likelihood, and frequently finds a good approximation to the maximum likelihood solution. The most commonly used constraint is to decompose q into a product of disjoint factors, for instance, one for each possible source object. By contrast, the approach adopted here uses a distribution truncated to a limited set $\mathcal{K}_n$ of all possible source configurations, i.e,

$$q_n(S;\Theta) = \frac{p(S \,|\, Y^{(n)},\Theta)}{\sum\limits_{S'\in\mathcal{K}_n} p(S' \,|\, Y^{(n)},\Theta)}\,\delta(S\in\mathcal{K}_n) \quad \text{with} \quad \delta(S\in\mathcal{K}_n) := \begin{cases} 1 & \text{if } S\in\mathcal{K}_n \\ 0 & \text{if } S\notin\mathcal{K}_n \end{cases}, \tag{16}$$

where $\mathcal{K}_n$ is a subset of the space of all states. If $\mathcal{K}_n$, indeed, contains most of the posterior mass given a data point $Y^{(n)}$, then $q_n(S;\Theta)$ approximates the exact posterior well. The variational approximation $q_n(S;\Theta)$ is a truncated posterior distribution that allows for the efficient estimation of the necessary expected values. The approximation is, therefore, referred to as *Expectation Truncation* (ET; Lücke and Eggert, 2010) or *truncated EM*.

In the case of the OCA generative model, we might expect good approximations if we identified a small set of candidate objects which are likely to be present in the scene, and then let $\mathcal{K}_n$ contain all of the combinations of the candidate objects. By using $q_n(S;\Theta)$ in Equation 16 as a variational distribution, the expectation values required for the M-step are of the form

$$\langle g(S,\Theta)\rangle_{q_n} = \sum_{S\in\mathcal{K}_n} \frac{p(S \,|\, Y^{(n)},\Theta)}{\sum\limits_{S'\in\mathcal{K}_n} p(S' \,|\, Y^{(n)},\Theta)}\, g(S,\Theta) = \frac{\sum\limits_{S\in\mathcal{K}_n} p(S,Y^{(n)}|\Theta')g(S,\Theta)}{\sum\limits_{S'\in\mathcal{K}_n} p(S',Y^{(n)}|\Theta')\,.} \tag{17}$$

We compute $\mathcal{K}_n$ for a given data point $Y^{(n)}$ in two stages. In the first we use a computationally inexpensive *selection* or *scoring* function (see Lücke and Eggert, 2010) to identify candidate objects. The selection function $\mathcal{S}_h\big(Y^{(n)}\big)$ seeks to assign high values to states corresponding to objects h present in the scene $Y^{(n)}$ and low values to states corresponding to objects which are not present. An ideal selection function would be monotonically related to the posterior probability of the object given the current image, but at the same time it would also be efficient to compute. The top H' states are selected as candidates and placed into an index set I_n. In the second stage we form the set $\mathcal{K}_n$ from the candidate objects. The index set I_n is used to define the set $\mathcal{K}_n$ as containing the states of all likely object combinations, i.e.,

$$\mathcal{K}_n = \{\, S \,|\, \big(\textstyle\sum_h s_h \leq \gamma \text{ and } \forall h\notin I_n : s_h = 0\big) \text{ or } \sum_j s_j \leq 1\}\,. \tag{18}$$

As an additional constraint, $\mathcal{K}_n$ does not contain combinations of more than γ objects. Furthermore, we make sure that $\mathcal{K}_n$ contains all singleton states, which proved beneficial in numerical experiments.

The selection function itself is defined as the squared distance between the observed image and the image generated by the hth component alone,

$$\mathcal{S}_h\big(Y^{(n)}\big) = -\sum_{d=1}^{D}\sum_{c=1}^{C}\big(Y^{(n)}_{cd} - \mathcal{T}_{cd}(S^{(h)};\Theta)\big)^2 \tag{19}$$

where $S^{(h)} := (\vec{s}^{(h)}, \varphi)$ with $\vec{s}^{(h)}$ being the state with only the hth object present.

Intuitively, the selection function can be thought of as a measure of the log-probability that each singleton state accounts for the current data point (also see Appendix D). Since we are only interested in the relative values of the selection function between the different components, Equation 19 contains only the exponent of the strictly monotonic exponential function without the normalization pre-factors.

3.3 Efficient EM Learning

The M-step Equations 12 to 14 together with the approximation of the expectation values in Equation 17 represent a learning algorithm for the OCA generative model. Its efficiency crucially depends on the approximation parameters H' and γ as they determine the number of latent states in $\mathcal{K}_n$ that have to be considered, that is,

$$\text{States}_{\text{ET}}(H, H', \gamma) \;=\; \sum_{\gamma'=0}^{\gamma} \binom{H'}{\gamma'} \gamma'! \;+\; (H - H')\,. \tag{20}$$

Because of the preselection of H' candidates, the combinatorics no longer scales with H. Only the number of singleton states scales linearly with H. Furthermore, the computation of the selection functions scales with H, but for the selection function specified in Equation 19 this scaling is only linear.

The potentially strongly reduced number of states in Equation 20 allows for an efficient optimization of the OCA model parameters (see Figure 8 in Appendix A for an example of such a reduction). By choosing H' and γ large enough to approximate the posteriors of the data points well, and small enough to sufficiently reduce the number of latent states that must be considered, an efficient yet accurate optimization procedure can be obtained. A crucial role for the efficiency / accuracy trade-off is played by the parameter γ which constrains the maximal number of considered components per data point (also compare Figure 8). If γ is too large, the large number of permutations that have to be considered for occlusion quickly results in computational intractabilities (scaling with $\gamma!$). If γ is too small, data points with more than γ components cannot be approximated well. Ideally we would like to learn the component parameters using as small an active set as possible (i.e., using low values of γ). However, representations of the posterior distribution which are too impoverished can result in strong biases in the parameter estimates (as is well-known, e.g., for factored variational approximations; Turner and Sahani, 2011). Since the approximation methods considered here will be at their worst for data-points that contain a large number of components, we simply discount these points and focus instead upon the data-points which are simple to learn from, thereby reducing the biases and the computational demands. This general approach was used by Lücke and Eggert (2010) who showed that such a discounting within the truncated approximate EM approach still results in approximately optimal solutions. Here we discount data-points that we believe contain more than γ components and modify the M-step equations accordingly. If we denote by $\mathcal{M}$ the subset of the N data points which are estimated to contain at most γ components, the new expressions are given

by:

$$W_{id} = \frac{\sum_{n\in\mathcal{M}} \langle \mathcal{A}_{id}(S,W)\rangle_{q_n}\, \vec{T}_i^T \vec{y}_d^{\,(n)}}{\sum_{n\in\mathcal{M}} \langle \mathcal{A}_{id}(S,W)\rangle_{q_n}\, \vec{T}_i^T \vec{T}_i} \qquad \vec{T}_i = \frac{\sum_{n\in\mathcal{M}} \sum_d \langle \mathcal{A}_{id}(S,W)\rangle_{q_n}\, W_{id}\, \vec{y}_d^{\,(n)}}{\sum_{n\in\mathcal{M}} \sum_d \langle \mathcal{A}_{id}(S,W)\rangle_{q_n}\, (W_{id})^2} \tag{21}$$

$$\sigma^{\text{new}} = \sqrt{\frac{1}{|\mathcal{M}|\, CD} \sum_{n\in\mathcal{M}} \left\langle \sum_{d=1}^{D} \sum_{c=1}^{C} \left(y_{dc}^{(n)} - \mathcal{T}_{dc}(S,\Theta) \right)^2 \right\rangle_{q_n}} \tag{22}$$

$$\pi^{\text{new}} = \frac{A(\pi)\,\pi}{B(\pi)} \frac{1}{|\mathcal{M}|} \sum_{n\in\mathcal{M}} \langle |\vec{s}| \rangle_{q_n} \;\text{ with} \tag{23}$$

$$A(\pi) = \sum_{\gamma\prime=0}^{\gamma} \binom{H}{\gamma\prime} \pi^{\gamma\prime} (1-\pi)^{H-\gamma\prime} \;\text{ and }\; B(\pi) = \sum_{\gamma\prime=0}^{\gamma} \gamma\prime \binom{H}{\gamma\prime} \pi^{\gamma\prime} (1-\pi)^{H-\gamma\prime}.$$

These modified M-step equations can be derived from a truncated free-energy (Lücke and Eggert, 2010) of the form

$$\mathcal{F}(q,\Theta) \;=\; \sum_{n\in\mathcal{M}} \sum_{S} \; q_n(S\,;\Theta') \;\log\Big(p(Y^{(n)} \,|\, S,\Theta)\, \frac{p(S\,|\,\Theta)}{\sum_{S'\in\mathcal{K}} p(S'\,|\,\Theta)} \Big), \tag{24}$$

with $q_n(S\,;\Theta')$ given in Equation 16 and with $\mathcal{K}$ being the set of all states S with at most γ non-zero components, $\mathcal{K} = \{S \,|\, |\vec{s}| \leq \gamma\}$. Details of the derivations of the update rules are given in Appendix B.

As can be observed, the update equations for W, T and σ remain essentially unchanged except for averages now running over the subset $\mathcal{M}$ instead of all data points (Equations 21 and 22). The reason is that the derivatives of the truncated free-energy w.r.t. these parameters are equal to the derivatives of the original free-energy except for reduced sums. For the derivative w.r.t. the prior parameter, the situation is different. The additional term in the denominator of the logarithm in Equation 24 results in a correction term for the update equation for π (Equation 23). Intuitively, it is clear that discounting data points with more than γ components has a direct impact on estimating the mean probability for a component to appear in a data point. The additional term in Equation 23 corrects for this.

To complete the procedure, we must determine the set $\mathcal{M}$ of all data points which are estimated to have γ active components or fewer. First note that the size of this set can be estimated given the current estimate for π. It contains an expected number of

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

data points (analogously to the π correction factor in Appendix B.2). Following Lücke and Eggert (2010), we now compute for all N data points the sums $\sum_{S\in\mathcal{K}_n} p(S, Y^{(n)}|\Theta')$, and define the set $\mathcal{M}$ to consist of the N^{cut} largest such values. The computation of $\mathcal{M}$ does not significantly increase the complexity of the algorithm, and in numerical experiments the set $\mathcal{M}$ is, indeed, found to contain almost all data points with at most γ components.

Iterating the M-step (Equations 21 to 23) and the E-step (Equations 17) results in a learning algorithm for the OCA generative model. As will be shown numerically in the next section, the algorithm allows for a very accurate estimation of model parameters based on a strongly reduced number of latent states.

4. Experiments

In order to evaluate the OCA learning algorithm, it has been applied to artificial and real-world data. Artificial data allows for an evaluation based on ground-truth information and for a comparison with other approaches. The use of real-world data enables us to test the robustness of the method.

4.1 Initialization and Annealing

For all data points, a vector $\vec{y}_d \in [0,1]^3$ represented the RGB values of a pixel. In all trials of the experiments we initialized the parameters W_{hd} and T_h^c by independently and uniformly drawing from the interval $[0,1]$. The parameters for sparseness and standard deviation were initialized as $\pi_{\text{init}} = \frac{1}{H}$ and $\sigma_{\text{init}} = 5$, respectively.

Parameter optimization in multiple-cause models is usually a non-convex problem. For the OCA model, the strongly non-linear combination rule seems to result in even more pronounced local optima in parameter space than is the case for other models such as sparse coding. To efficiently avoid convergence to local optima, we (A) applied deterministic annealing (Ueda and Nakano, 1998; Sahani, 1999) and (B) added noise to model parameters after each EM iteration. Annealing was implemented by introducing the temperature $T = \frac{1}{\beta}$. The inverse temperature β started near 0 and was gradually increased to 1 as iterations progressed. It modified the EM updates by substituting $\pi \to \pi^\beta$, $(1-\pi) \to (1-\pi)^\beta$, and $\frac{1}{\sigma^2} \to \frac{\beta}{\sigma^2}$ in all E-step equations. We also annealed the occlusion non-linearity by setting $\rho = \frac{1}{1-\beta}$; however, once β became greater than 0.95 we set $\rho = 21$ and did not increase it further. We ran 100 iterations for each trial of learning. The inverse-temperature was set to $\beta = \frac{2}{D}$ for the first 15 iterations, then linearly increased to $\beta = 1$ over the next 15 iterations, and then kept constant until termination of the algorithm.

Additive parameter noise was drawn randomly from a normal distribution with zero mean. Its standard deviation was initially set to 0.3 for the mask parameters and at 0.05 for the prior and noise parameters. The value was kept constant for the first 10 iterations and then linearly decreased to zero over the next 30 iterations. The degeneracy parameter α was initialized at 0.2 and increased to 0.6 from iteration 25 to 35. The amount of data points used for training was linearly reduced from N to N^{cut} between iteration 15 to 30. Approximation parameters were set to $\gamma = 3$ and $H' = 5$ unless stated otherwise.

4.2 Evaluation of Optimization Results

After optimizing the parameters using the derived EM approach, we obtain different sets of parameters in different runs. In the case of available ground-truth parameters, a means to identify the best run is a comparison of the learned parameters with the ground-truth. It could, for instance, be asked if all generating components (all generating objects in our case) have been recovered successfully. However, usually the ground-truth parameters

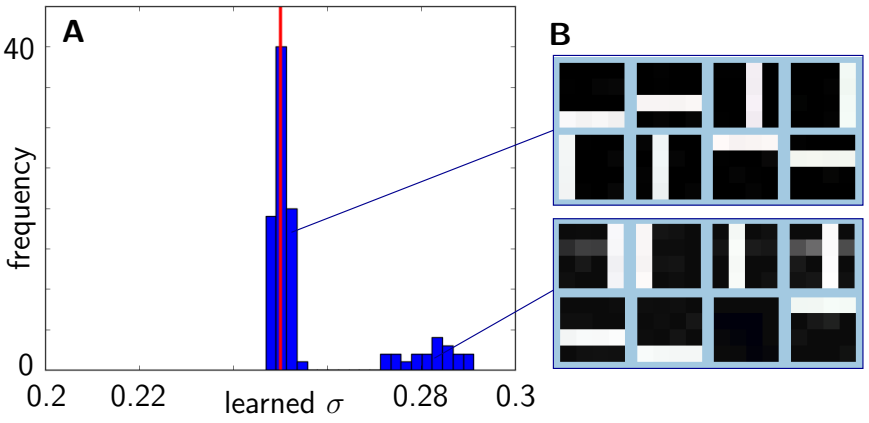


Figure 3: **A** Histogram of values for the parameter σ (Equation 13) for 100 runs of the algorithm on a standard bars test (see bars test section). The ground-truth value for σ in these runs was 0.25, indicated by the red line. As can be observed, most values lie close to this number while some values form a second mode at higher values. By thresholding the σ parameter, we can identify local optima. **B** Examples for the basis functions for the left and for the right cluster.

are not known and so another measure for the quality of a run has to be found. A good indication of the quality of the learned parameters is provided by the learned noise parameter σ (Equation 13). In fact, if we compute the derivative of the update rule for σ w.r.t. the mask and feature parameters and set these equal to zero, we obtain the same update rules for W and T as for the derivative of the free-energy. That is, maximizing the free-energy corresponds to optimizing (minimizing) the noise which the model has to assume to explain the data (see Appendix C). If this noise is small, the data are well explained by the parameters (see Figure 3 for an application to artificial data).

4.3 Colored Bars Test

The component extraction capabilities of the model were tested using the colored bars test. This test is a generalization of the classical bars test (Földiák, 1990) which has become a popular benchmark task for non-linear component extraction. In the standard bars test with $H = 8$ bars the input data are 16-dimensional vectors, representing a 4×4 grid of pixels, i.e., $D = 16$. The single bars appear at the 4 vertical and 4 horizontal positions. For the colored bars test, each of the bars has a different color. Feature values were initialized such that the color values had maximal distance to each other in one brightness plane of HSV color space. Once chosen, they remained fixed for the generation of the data set. For each image a bar appeared independently with a probability $\pi = \frac{2}{H} = 0.25$ which resulted in two bars per image on average (the standard value in the literature). For the bars chosen to be active, a ranking in depth was randomly and uniformly chosen from the permutation group to generate the image. The color of each (noiseless) image pixel was determined by the least distant bar and was black, i.e., zero, if the pixel was

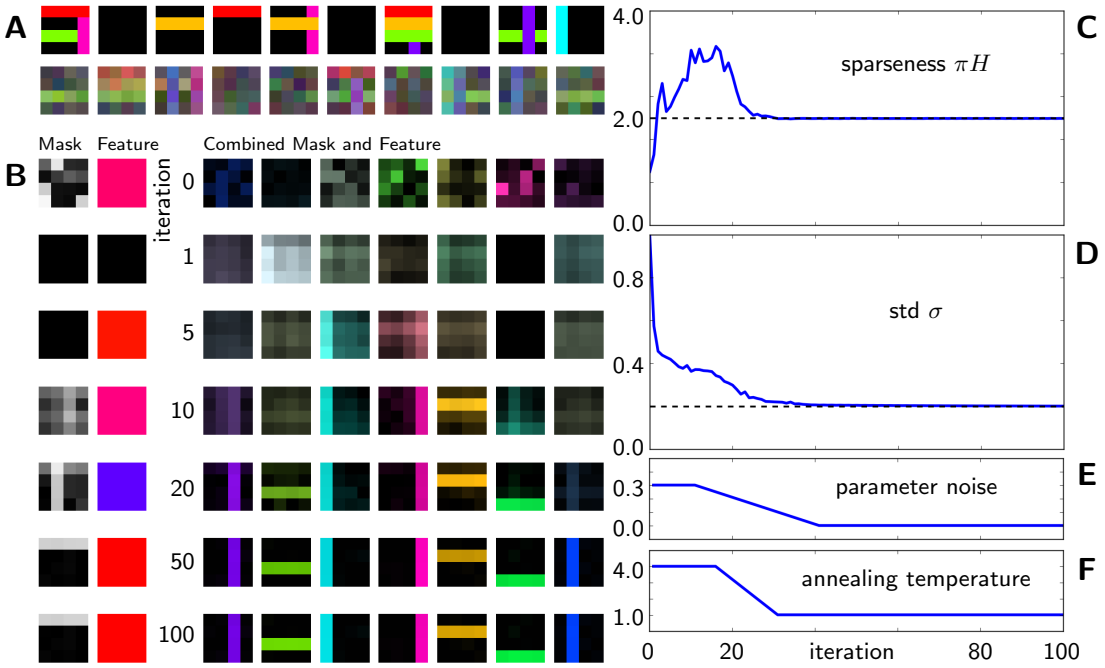


Figure 4: **A** Example of ten noiseless and noisified data points. **B** Development of the (reshaped) generative fields for the given iterations. For the first cause, mask $\vec{W}_h$ and feature $\vec{T}_h^T$ are displayed separately. The other causes are shown as product $\vec{W}_h \cdot \vec{T}_h^T$. **C** - **D** Development of data sparseness and standard deviation over the 100 iterations. **E** Magnitude of the noise which is added to the mask parameters after each iteration. **F** Annealing temperature as chosen for the algorithm.

occupied by no bar. Gaussian data noise of $\sigma = 0.25$ was added to each data point. $N = 1000$ images were generated for learning and Figure 4A shows a random selection of 10 noiseless and 10 noisy examples. The learning algorithm was applied to the colored bars test with $H = 8$ hidden units and $D = 16$ input units. The inferred approximate maximum-likelihood parameters converged to values close to the generating parameters in 97 of 100 trials. The success rate, or *reliability*, was thus 97%. Lücke et al. (2009) achieved a similar reliability of 96% with $N = 500$. Yet, here we learn, with the current version of the algorithm, more parameters, namely the data noise and the sparseness parameter. The values obtained for σ and π all lay in the interval $[0.246, 0.252]$ and $[0.241, 0.268]$, respectively. Figure 4B shows the time-course of a typical trial during learning. As can be observed, the mask values W and the feature values T converged to values close to the generating ones. More specifically, the product of each mask $\vec{W}_h$ and its color value $\vec{T}_h$ represents a true underlying bar in the right color. Reliability is affected by changes in the annealing and parameter noise schedules, i.e., by changes to those algorithmic parameters which control the mechanisms for avoiding local optima. Furthermore, we observed an effect of the approximation parameters H' and γ on the algorithm's capability to avoid local optima. Notably, smaller as well as much larger values for H' and γ lead to lower

reliabilities. In addition to increasing efficiency, Expectation Truncation, therefore, helps in avoiding local optima for this model, presumably because local optima corresponding to broad posterior distributions are avoided. A similar observation was recently reported in an application of ET to a sparse coding variant (Exarchakis et al., 2012).

4.4 Standard Bars Test

Instead of choosing the bar colors randomly as above, they can also be set to specific values. In particular, if all bar colors are white, $\vec{T} = (1, 1, 1)^T$, the classical version of the bars test is recovered. Note that the learning algorithm can be applied to this standard form without modification, even though it is impossible to recover the relative depth of the bars in this case. When the generating parameters were as above (eight bars, probability of a bar to be present $\frac{2}{8}$, $N = 1000$), all bars were successfully extracted in 80 of 100 trials (80% reliability). The estimated values of σ and π lay in the intervals $[0.247, 0.254]$ and $[0.241, 0.263]$, respectively. When learning on noiseless data, we obtained a reliability of 95%. By increasing the approximation parameters to $\gamma = 4$ and $H' = 6$, reliability changed to 91%.

For a standard setting of the parameters ($N = 500$, $H = 10$, $D = 5 \times 5$, probability of $\frac{2}{10}$ for each bar to be present) as was used in numerous previous studies (Saund, 1995; Dayan and Zemel, 1995; Hochreiter and Schmidhuber, 1999; Lücke and Sahani, 2008; Lücke and Eggert, 2010), the OCA algorithm with $\gamma = 3$ and $H' = 5$ achieved 83% for a noisy and 78% for a noiseless bars test. For $N = 1000$ data points reliability increased to 85%. For comparison, earlier generative modeling approaches such as those reported by Saund (1995) or Dayan and Zemel (1995) (both assuming a noisy-or like combination rule) achieved 27% and 69% reliability, respectively. Maximal Causes Analysis (Lücke and Sahani, 2008; Lücke and Eggert, 2010) achieved about 82% (MCA_3) reliability. And the preliminary version of the OCA algorithm (Lücke et al., 2009) achieved 86% for noiseless data. Approaches such as PCA or ICA were reported to fail in this task (Hochreiter and Schmidhuber, 1999). Furthermore, different types of objective functions and neural network approaches (Charles et al., 2002; Lücke and Malsburg, 2004; Spratling, 2006) are also successful at this task, often reporting close to 100% reliability (also see Frolov et al., 2014). The assumptions used (e.g., fixed bar appearance, noise level, parameter constraints, constraint on latent activities) are often implicit but, at the same time, can significantly facilitate learning. NMF algorithms can be successful in extracting all bars (with up to 100% reliability) but require hand-set values for sparsity constraints on hidden variables and/or parameters (see Hoyer, 2004, and for discussions Spratling, 2006, Lücke and Sahani, 2008). In general, the fewer assumptions a model makes, the more difficult it becomes to infer the parameters from a given set of data. For earlier generative models and in particular for the more general model discussed in this paper, larger data sets directly translate into higher reliabilities. A reliability of 78% for the noiseless bars test is, for the OCA algorithm discussed in this work, in this view still relatively high. Reliabilities are comparable to values for MCA and to the preliminary OCA algorithm (Lücke et al., 2009). Note, however, that the latter did use fixed values for data noise σ and bar appearance π which may explain the higher reliability.

As the recovery of optimal model parameters is the goal of the approach, we can further increase the rate of successfully recovered parameters that correspond to a representation of

all bars by considering several runs of the algorithm simultaneously. That is, given a set of N images, we can apply the algorithm M times, and use as the final result the parameters of a run with the smallest σ value. For some data sets, we even obtain two clusters of σ values (see Figure 3) where the cluster with smaller σ's represents the runs which have terminated in an optimum with parameters representing all bars. Note that clearly separable clusters are not observed for all data sets and parameter settings. In general, however, runs with small σ values tend to correspond to parameters reflecting the true underlying generative process more accurately. For the standard settings of the bars test with $D = 5 \times 5$, $N = 500$, $H = 10$, and noiseless data, the algorithm with $M = 20$ extracts all components in 50 of 50 runs. But note that each run now consists of evaluating $M = 20$ subruns. The same applies for values of M down to $M = 10$.

4.5 Inference

To briefly illustrate the algorithm's performance on an inference task, i.e., the extraction of the underlying causes and their depth order, and to show how inference can be applied to data points exceeding γ components, let us consider data points generated according to the colored bars test. Furthermore, consider the model after it has learned the parameters to represent the bars, noise level, and sparsity. Given a data point, the trained model can infer the hidden variables by applying the following procedure: We start by executing an E-step with the same values for H' and γ as used during training ($H' = 5$ and $\gamma = 3$ in this case). We then determine the maximum a-posteriori (MAP) state $\vec{s}^*$ based on the approximate posterior computed in this E-step. If this state has $|\vec{s}^*| = \gamma$ active components, we repeat the E-step with values of H' and γ increased by one each (leaving H' unchanged if $H' = H$). We terminate the procedure if the MAP state has less than γ states or if $\gamma = H' = H$. Exemplarily, Figure 5 shows three data points and the corresponding MAP states obtained with the described procedure. The data point with two components terminated after the first E-step (Figure 5A), the data points with three after γ was increased to four (Figure 5B), and the data point with four components terminated after γ was increased to five (Figure 5C,D show result for initial and final γ). For ambiguous data points, e.g., if the input contained two parallel bars, two states or more states can carry equally large probability mass due to the fact that different depth permutations do not change the image. The MAP estimate can still serve to illustrate the inference result but the approximate distribution over states represents a more accurate description in this case.

4.6 More Realistic Data

To numerically investigate the algorithm for more realistic data, it was applied to data based on pictures of objects from the COIL-100 database (Nene et al., 1996).[1] Images were scaled down to 20×20 pixels and were placed at random planar positions on a black background image of $D = 35 \times 35 = 1225$ pixels. The objects were then colored with their mean color, weighted by pixel intensity (see Figure 6A). In 100 runs, we created $N = 8000$ data points by combining $H_{\text{gen}} = 20$ of these images according to the generative model with

1. We used objects 2, 3, 4, 25, 28, 34, 47, 48, 51, 56, 58, 60, 73, 74, 77, 85, 89, 94, 97, and 112 all at 0 degree rotation.

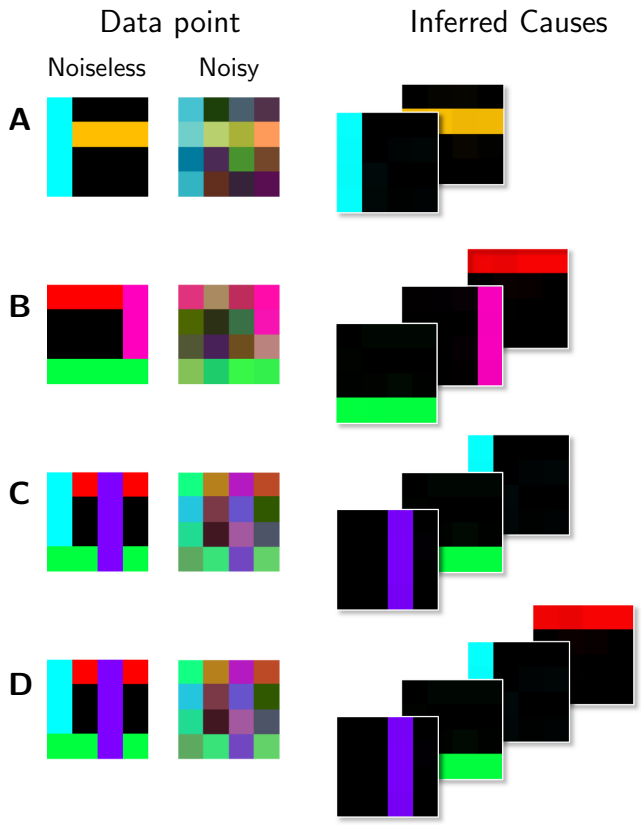


Figure 5: Examples of the inference procedure for the colored bars test. The second column shows the data points used for inference (with their noiseless versions in the first column). On the right, the causes inferred from the noisy data points are shown arranged in their inferred depth order. **A** - **D** The algorithm reliably inferred the causes for the three examples (**C** and **D** show two steps of the inference procedure). Note that we have learned the basis functions from noisy data with the same properties as those shown here.

prior parameter $\pi = \frac{2}{H_{\text{gen}}} = 0.1$, i.e., $\pi H_{\text{gen}} = 2$ and data noise $\sigma = 0.25$ (see Figure 6B). For learning, the algorithm was applied with $H = 30$ mask and feature parameters $\vec{W}_h$ and $\vec{T}_h$, i.e., 50% more than we used for data generation. Figure 6C shows the resulting mask and feature parameters for an example run (where we display each pair of feature and mask combined into one image, compare Equation 3). We obtained all 20 underlying basis functions along with 10 noisy fields in 44% of the trials. For the data noise we obtained $\sigma \in [0.251, 0.254]$ and for the sparseness parameter $\pi \in [0.062, 0.070]$, i.e., $\pi H \in [1.85, 2.11]$ for all runs. The high discrepancy in the sparseness values can be explained by the fact that we have introduced extra basis functions for learning. In the remaining 56 trials, almost all objects were extracted with usually just one and at most three objects not being

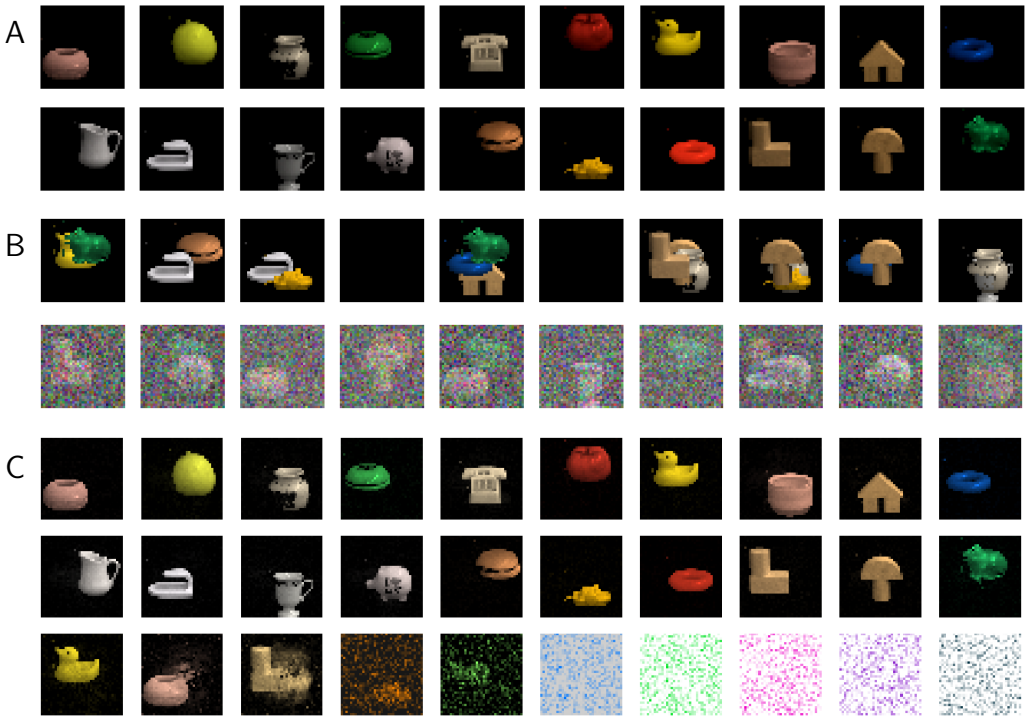


Figure 6: **A** 20 downscaled COIL images. **B** 10 noiseless and 10 noisy data points (obtained from **A** according to Equation 3). **C** 30 extracted basis functions. The first two rows display the clean extracted causes in the same order as in **A**. The third row shows the additionally learned causes which are mostly noisy fields or noisy combinations of more than one cause.

represented. Again, low values of the observation noise were found to indicate the successful extraction of all objects. By performing a series of runs and retaining the parameters of runs with the lowest learned observation noise, the reliability increased to close to 100%.

4.7 Real Data

Finally we tested the algorithm on a real world data set that includes a range of effects that are not present in synthetic data, including real-world occlusion, lighting variability due to shadows and specular variations, as well as some small translation effects. As such the data-set provides an important test of the algorithm's robustness. The data-set comprised 500 pictures of scenes consisting of up to five (toy) cars in front of a gray background. The cars could appear at different positions in depth but always in the same position in planar space (see Figure 7A). Pictures were taken from the side (as for instance a camera in a tunnel might be positioned) such that moving a car in depth had almost no effect on its vertical or horizontal position in the image (see Figure 7B). We then cut out the area of the images that contained the cars and downscaled the cut-out images to 40×165 pixels. Subsequently,

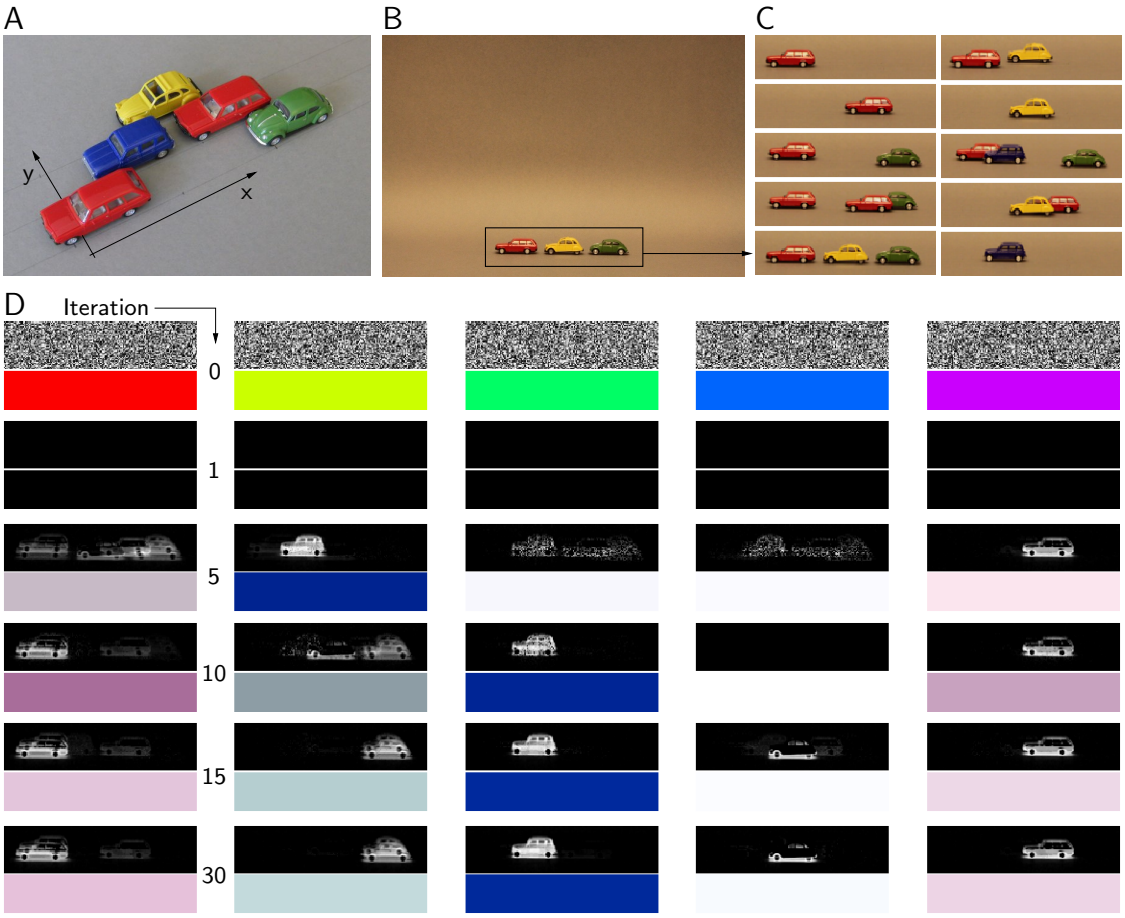


Figure 7: Numerical experiment with photographic images of cars. **A** Five cars could appear in three lanes which account for the arbitrary position in depth (y). Position in x-direction was fixed. **B** One of the 500 pictures taken of one state with three active causes. **C** 10 data points after cutting and pre-processing. **D** Generative fields (mask on top, feature below) at different iterations.

the images were normalized in luminance and the 5% lightest pixels were clamped to the same maximal value for each color channel to remove reflection effects. Figure 7C shows some example images. For learning, we then subtracted an image of an empty scene (i.e., the background) pixelwise from each input image such that pixels that did not belong to a cause became almost zero. Note that these data points can now have negative values while RGB values are usually defined to be positive. To interpret the data point as an image, one can map the values in the three color channels back to $[0, 1]$. For a homogeneous background, these images look almost the same as the original input images. We initialized the masks as random noise and the features as RGB color vectors all equally far apart from each other in color space (see Figure 7D). The inverse annealing temperature was set to $\beta = \frac{1}{15}$ and increased to 1 from iteration 5 to 25. Parameter noise was decreased between

iterations 15 and 26. Over 30 iterations we extracted the masks of all five cars along with data noise ($\sigma = 0.05$) and sparseness ($\pi = 0.17$ with $\pi_{\text{ground-truth}} = \frac{1}{H} = 0.2$). The features which had positive and negative values were mapped to $[0, 1]$ to be interpreted as color. As can be seen from the generative fields in iteration 30, not all masks were extracted cleanly. This can be explained by the fact that a different position in depth still causes a slight shift in planar space such that in some images one cause is higher or lower than in others. This smearing effect leads to a change in color because a pixel then sometimes belongs to the car and sometimes to the background which results in a color shift towards the background color (black). Another reason is that one cause does not only consist of one color but rather of a combination of the car color and background, shadow, window, and wheel color. For the yellow car which is relatively similar to the background the mask almost only represents the shadow of the car, which is the most salient part relative to the background. For the other cars, the masks correspond to representations of whole cars.

5. Discussion

We have studied learning in a multiple-cause generative model which assumes an explicit model of occlusion for component superposition. According to the OCA model assumptions, an object can appear in different depth positions for different images. This aspect reflects properties of real images and is complementary, e.g., to assumptions made by sprite models (Jojic and Frey, 2001; Williams and Titsias, 2004). A combination of sprite models and the OCA model is, therefore, a promising line of inquiry, e.g., towards systems that can learn from video data in which objects change their positions in depth. Other lines of research have also identified occlusions as an important property that has to be modeled for applications to visual data. In the context of neural network modeling, Tajima and Watanabe (2011) have recently addressed the problem (albeit with a very small number of components and in a partly supervised setting), while restricted Boltzmann machines have been augmented by LeRoux et al. (2011) to incorporate occlusions.

The directed graphical model studied here has a close connection to multiple-cause approaches such as sparse coding, NMF or ICA. All of these standard approaches use linear superpositions of elementary components to model component superposition. ICA and SC have prominently been applied to explain neural response properties, and NMF is a popular approach to learn components, e.g., for visual object recognition (e.g., Lee and Seung, 1999; Wersing and Körner, 2003; Hoyer, 2004). In the class of multiple-cause approaches our model is the first to generalize the combination rule to one that models occlusion explicitly. While non-linear combination rules have been studied before by Saund (1995); Dayan and Zemel (1995); Šingliar and Hauskrecht (2006); Frolov et al. (2014) (noisy-or), Valpola and Karhunen (2002); Honkela and Valpola (2005) (post-linear sigmoidal function) or Lücke and Sahani (2008); Puertas et al. (2010); Bornschein et al. (2013) (point-wise maximum), we go a step further and model occlusion explicitly by making the component combination dependent on an additional hidden variable for depth-ordering. As a consequence, the model requires two sets of parameters: masks and features. Masks are required because the planar space that a component occupies has to be defined. Parameterized masks are, therefore, a feature of many approaches with explicit occlusion modeling (compare Jojic and Frey, 2001; Williams and Titsias, 2004; LeRoux et al., 2011; Tajima and Watanabe, 2011). For

our model, the combination of masks and vectorial feature parameters, furthermore, allows for applications to more general sets of data than the scalar values used for SC, NMF or than in applications of sprite models (compare Jojic and Frey, 2001; Williams and Titsias, 2004). In numerical experiments we have used color images for instance. However, we can also apply our algorithm to gray-level data such as used for other algorithms. This allows for a direct quantitative comparison of the novel algorithm with state-of-the-art component extraction approaches. The reported results for the standard bars test show the competitiveness of our approach despite its larger set of parameters (compare, e.g., Spratling, 2006; Lücke and Sahani, 2008). For applications to visual data, color is the most straightforward feature to model. Possible alternatives are Gabor feature vectors which model object edges and textures, or further developments such as SIFT features (Lowe, 2004). Depending on the application, the generative model itself could also be generalized. It is, for instance, straightforward to introduce several feature vectors per cause. Although one feature (e.g., one color) per cause can represent a suitable model for many applications, it might for other applications also make sense to use multiple feature vectors per cause. In the extreme case, as many feature vectors as pixels could be used, i.e., $\vec{T}_h \rightarrow \vec{T}_{hd}$. The derivation of update rules for such features would proceed along the same lines as the derivations for single features $\vec{T}_h$. Furthermore, individual prior parameters for the frequency of object appearances could be introduced. Additional parameters could be used to model different prior probabilities for different arrangements in depth. Finally, the most interesting but also most challenging generalization direction would be the inclusion of explicit invariance models. In its current form the model uses, in common with state-of-the-art component extraction algorithms, the assumption that the component locations are fixed. Especially for images of objects, changes in planar component positions have to be addressed in general. Possible approaches that have been discussed in the literature have, for instance, been investigated by Jojic and Frey (2001) and Williams and Titsias (2004) in the context of occlusion modeling, by Eggert et al. (2004) and Wersing and Körner (2003) in the context of NMF, or by Berkes et al. (2009), Gregor and LeCun (2011) and others in the context of sparse coding. The crucial challenge of a generalization of the occlusion model studied in this work is the further increase in the dimensionality of the hidden space. By generalizing the methods as used here, such challenges could be overcome, however. On the other hand, methods such as sparse coding or NMF have proven to be useful building blocks in vision systems although they do not address translation invariance in an explicit way. As a generalization of sparse coding, the model studied here can provide a more accurate model in situations where the modeling of occlusions is important. Like for sparse coding, no prior information about the two dimensional nature of images is used in the model, i.e., learning would not suffer from a (fixed) permutation of all pixels applied to all data points. The tasks faced by the model may, therefore, appear easier for the human observer because humans make (e.g., for the COIL data) use of additional object knowledge such as of the gestalt law of proximity. This also illustrates that extensions of the model to incorporate prior knowledge about objects would further improve the approach.

To investigate robustness, we have applied the developed algorithm to real images, and observed that it is robust enough to work on non-artificial data. We do not regard this work as providing a directly applicable algorithm, however. The main goal of this study was rather to show that the challenges of a multiple-cause model with explicit occlusions can

be overcome. Replacing the standard linear superposition of sparse coding by an occlusion superposition resulted in a number of challenges that all had to be addressed:

1) The occlusion model required parameterized masks.
2) The learning equations are not closed-form.
3) Occlusion leads to a much larger combinatorial explosion of possible configurations.
4) Posterior probabilities are not unimodal.
5) Local optima in parameter space are more pronounced.

By generalizing the treatment of non-linear superpositions developed for maximal causes analysis (see Lücke and Sahani, 2008), parameter update equations were derived for all parameters of the occlusion model: for masks and features as well as data noise and sparsity (addressing points 1 and 2). The combinatoric challenge of the model's large latent space (point 3) was addressed using Expectation Truncation (ET; Lücke and Eggert, 2010) which, furthermore, does not make any assumptions about unimodal posteriors (point 4). Combined with deterministic annealing, the algorithm efficiently avoided local optima (point 5). Compared to the earlier version of the OCA learning algorithm (Lücke et al., 2009), Expectation Truncation provides a further increase of efficiency by selecting relevant latent causes using selection functions. In this way, the complexity was reduced from scaling polynomially with H (usually with H to the power of 3 or 4) to a linear scaling with H. The combinatorics of states instead only affects the much smaller space of the H' candidates selected for each data point (compare Figure 8). Given H' and γ the combinatorics is known exactly (Equation 20) and this is the main factor that determines the scalability of the algorithm. How large the values of H' and γ have to be depends on the data. A large number of objects per image will require higher values and consequently a large number of states. If the average number of objects per data point remains constant, H' and γ can be kept constant for increasing H, and the number of states that have to be evaluated will scale only linearly with H. Secondary effects may lead to the algorithm scaling super-linearly, however. Larger values of H mean a higher number of parameters which may in turn require larger data sets to prevent overfitting. Such effects can be considered as much less severe than combinatorial effects that increase the state space. Because of the generally favorable scaling with H, we could handle numerical experiments with larger numbers of latents than previously considered. For the COIL data set, the algorithm was run with $H = 30$ hidden variables and $D = 35 \times 35$ observed variables. For the colored bars test the algorithm was run with up to $H = 80$ hidden and $D = 40 \times 40$ observed variables (but extraction of all bars becomes increasingly challenging). In practice and depending on the data, learning times may differ. For some data longer learning may be required for the parameters to converge or in order to efficiently avoid local optima with slower annealing. A precise theoretical quantification of these data-dependent effects is, like for most learning algorithms, difficult. In all our empirical evaluations we found that the mechanisms in place to avoid local optima are important. We applied deterministic annealing and parameter noise for the algorithm to converge to approximately optimal global solutions, i.e., to solutions corresponding to parameters that all represented true data components (in cases when these components were known). Without annealing or parameter noise the algorithm converged to approximate global optima only in very few cases, and local optima were usually reached after a small number of steps. Both annealing and parameter noise

had an influence on the typical convergence points. With only parameter noise (i.e., without annealing), the algorithm usually converged within few EM iterations to local optima with a relatively large number of fields representing more than one component. With only annealing (i.e., without parameter noise), the algorithm often converged to local optima in which most components were represented correctly but where few generative fields represented two components. The combination of annealing and parameter noise resulted in a frequent representation of all causes (see experiments). As stated earlier, we also observed a positive effect of the approximation scheme in avoiding local optima presumably because shallow local optima corresponding to solutions with dense states are not considered by the truncated approximation.

A further improvement that followed from the application of Expectation Truncation is the availability of learning rules for data noise and sparsity. While data noise could have been inferred within the preliminary study of the occlusion model (Lücke et al., 2009), inference of sparsity requires a correction term that compensates for considering a reduced space of latent configurations, and Expectation Truncation provides a systematic way to derive such a correction (compare Appendix B.2). Data noise and sparsity parameters are, notably, not a consequence of modeling occlusion. They are potential parameters also of standard sparse coding approaches or NMF objective functions. Nonetheless, most sparse coding approaches only optimize the generative fields because of limitations induced by the usual maximum a-posteriori based learning (but see Berkes et al., 2008; Henniges et al., 2010, for exceptions). Likewise, NMF approaches focus on learning of generative fields / basis functions. Sparsity is at most indirectly inferred by standard SC or NMF through cross-validation.

To summarize, our study shows that the challenges of occlusion modeling with explicit depth orders can be overcome, and that all model parameters can be efficiently inferred. The approach complements established approaches of occlusion modeling in the literature by generalizing standard approaches such as sparse coding or NMF to incorporate one of the most salient properties of visual data.

Acknowledgments

MH, JE and JL acknowledge funding by the German Research Foundation (DFG) in the project LU 1196/4-2, by the Honda Research Institute Europe GmbH (HRI Europe), by the DFG Cluster of Excellence EXC 1077/1 (Hearing4all), and by the German Federal Ministry of Education and Research (BMBF) in the project 01GQ0840 (BFNT Frankfurt). RET acknowledges funding by EPSRC Postdoctoral Fellowship Grant EP/G050821/1, and MS acknowledges funding by The Gatsby Charitable Foundation.

Appendix A. Illustration of Hidden State Combinatorics

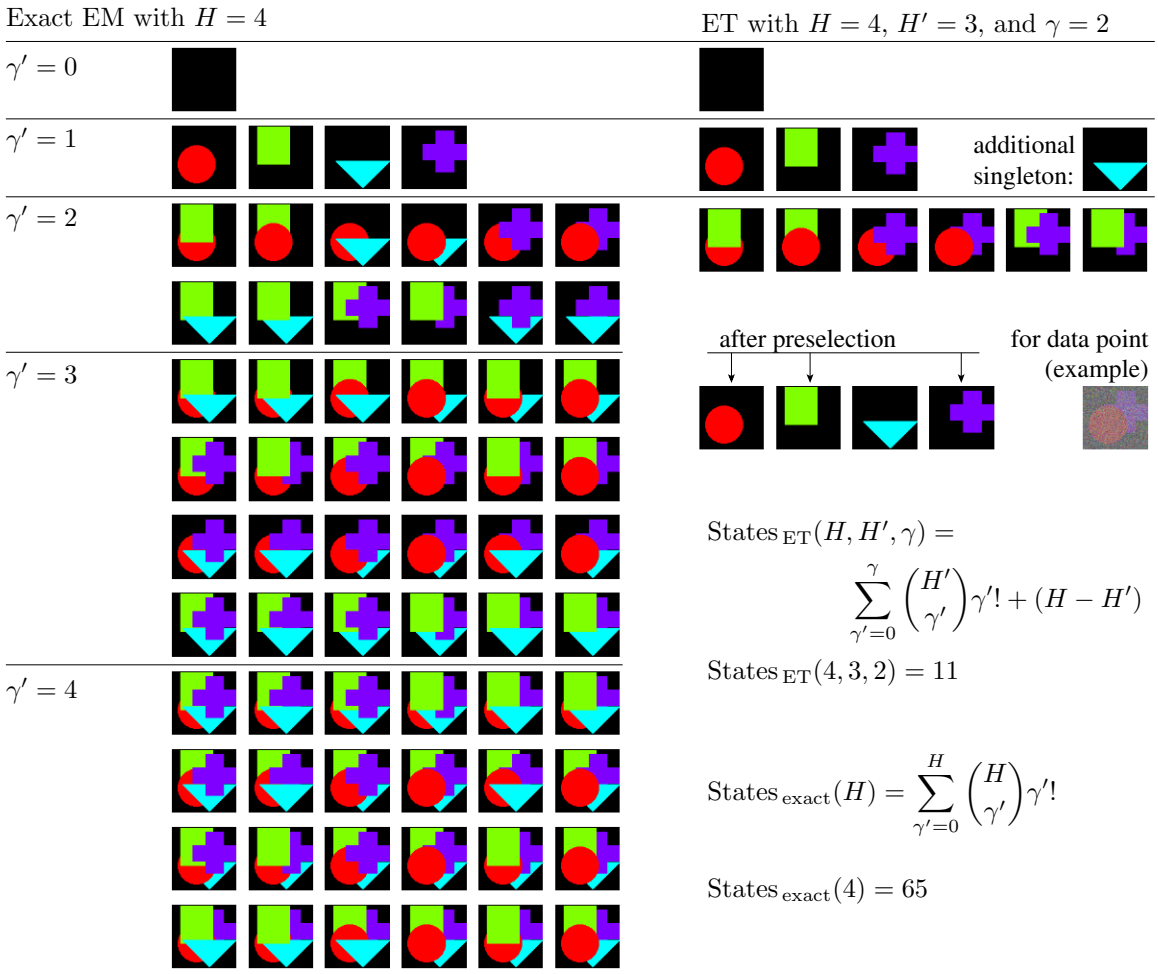


Figure 8: This figure shows all possible combinations for hidden states with given basis functions for exact EM as well as for a reduced number of combinations. The left-hand-side corresponds to exact EM and thus displays all possible combinations of $H = 4$ different causes separated for each value of $\gamma' = |\vec{s}|$. On the right-hand-side, we see all combinations of the three most relevant objects in which at most two objects appear simultaneously. The reduced number of combinations corresponds to the states evaluated by the used approximation (Equation 16) with parameters $H' = 3$ and $\gamma = 2$ in Equation 18. Given a noisy data point, the first, second, and fourth component are preselected for this example. Note that we additionally consider all singleton states. The formulas for the number of considered states are given on the bottom right.

Appendix B. Derivation of Update Rules

Our goal is to optimize the free-energy, i.e.,

$$\mathcal{F}(\Theta, q) = \sum_{n=1}^{N} \Big[\sum_{S} \; q_n(S\,; \Theta') \;\; \Big[\log\big(p(Y^{(n)} \,|\, S, \Theta)\big) + \log\big(p(S \,|\, \Theta)\big) \Big] \Big] \; + \; \mathbf{H}[q]\,,$$

where

$$p(Y^{(n)} \,|\, S, \Theta) = \prod_{d=1}^{D} p(\vec{y}_d^{\,(n)} | \vec{\mathcal{T}}_d(S, \Theta)) \text{ with } p(\vec{y}\,|\vec{t}\,) = \mathcal{N}(\vec{y}; \vec{t}, \sigma^2 \mathbb{1})\,.$$

More explicitly,

$$\begin{aligned} p(Y^{(n)} \,|\, S, \Theta) &= \prod_{d=1}^{D} \prod_{c=1}^{C} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2} \left(y_{cd}^{(n)} - \mathcal{T}_{cd}(S, \Theta) \right)^2 \right) \\ &= \left(2\pi\sigma^2\right)^{-\frac{CD}{2}} \prod_{d=1}^{D} \prod_{c=1}^{C} \exp\left(-\frac{1}{2\sigma^2} \left(y_{cd}^{(n)} - \mathcal{T}_{cd}(S, \Theta) \right)^2 \right) \end{aligned}$$

and for the logarithm

$$\log\Big(p(Y^{(n)} \,|\, S, \Theta)\Big) = -\frac{CD}{2} \log\left(2\pi\sigma^2\right) - \sum_{d=1}^{D} \sum_{c=1}^{C} \frac{1}{2\sigma^2} \left(y_{cd}^{(n)} - \mathcal{T}_{cd}(S, \Theta) \right)^2 .$$

The prior term, we defined to be

$$p(S \,|\, \Theta) = \pi^{|\vec{s}\,|} (1-\pi)^{(H - |\vec{s}\,|)} \frac{1}{|\vec{s}\,|!},$$

such that

$$\log\left(p(S \,|\, \Theta)\right) = |\vec{s}\,| \log(\pi) + (H - |\vec{s}\,|) \log(1-\pi) - \sum_{\gamma=1}^{|\vec{s}\,|} \log(\gamma)\,.$$

The free-energy thus takes the form

$$\begin{aligned} \mathcal{F}(\Theta, q) = \sum_{n=1}^{N} \Big[\sum_{S} \; q_n(S\,; \Theta') \Big[&- \frac{CD}{2} \log\left(2\pi\sigma^2\right) - \frac{1}{2\sigma^2} \sum_{d=1}^{D} \sum_{c=1}^{C} \left(y_{cd}^{(n)} - \mathcal{T}_{cd}(S, \Theta) \right)^2 \\ &+ |\vec{s}\,| \log(\pi) + (H - |\vec{s}\,|) \log(1-\pi) - \sum_{\gamma=1}^{|\vec{s}\,|} \log(\gamma) \Big] \Big] \\ &+ \mathbf{H}[q]\,. \end{aligned}$$

B.1 Optimization of the Data Noise

Let us start be deriving the M-step equation for σ as follows:

$$\begin{aligned}
&\frac{\partial}{\partial\sigma}\mathcal{F}(\Theta,q)\\
&=\sum_{n=1}^{N}\Bigg[\sum_{S}\ q_n(S\,;\Theta')\Bigg[-\frac{CD}{2}\frac{\partial}{\partial\sigma}\log\left(2\pi\sigma^2\right)-\left(\frac{\partial}{\partial\sigma}\frac{1}{2\sigma^2}\right)\sum_{d=1}^{D}\sum_{c=1}^{C}\left(y_{cd}^{(n)}-\mathcal{T}_{cd}(S,\Theta)\right)^2\Bigg]\Bigg]\\
&=\sum_{n=1}^{N}\Bigg[\sum_{S}\ q_n(S\,;\Theta')\Bigg[-\frac{CD}{2}\frac{4\pi\sigma}{2\pi\sigma^2}-\left(-2\,(2\sigma)^{-3}\,2\right)\sum_{d=1}^{D}\sum_{c=1}^{C}\left(y_{cd}^{(n)}-\mathcal{T}_{cd}(S,\Theta)\right)^2\Bigg]\Bigg]\\
&=\sum_{n=1}^{N}\Bigg[\sum_{S}\ q_n(S\,;\Theta')\Bigg[-\frac{CD}{\sigma}+\frac{1}{2\sigma^3}\sum_{d=1}^{D}\sum_{c=1}^{C}\left(y_{cd}^{(n)}-\mathcal{T}_{cd}(S,\Theta)\right)^2\Bigg]\Bigg]\stackrel{!}{=}0
\end{aligned}$$

$$\begin{aligned}
\Rightarrow&\frac{1}{2\sigma^3}\sum_{n=1}^{N}\Bigg[\sum_{S}\ q_n(S\,;\Theta')\sum_{d=1}^{D}\sum_{c=1}^{C}\left(y_{cd}^{(n)}-\mathcal{T}_{cd}(S,\Theta)\right)^2\Bigg]=\frac{NCD}{\sigma}\\
\Rightarrow&\sigma^2=\frac{1}{NCD}\sum_{n=1}^{N}\Bigg[\sum_{S}\ q_n(S\,;\Theta')\sum_{d=1}^{D}\sum_{c=1}^{C}\left(y_{cd}^{(n)}-\mathcal{T}_{cd}(S,\Theta)\right)^2\Bigg]
\end{aligned}$$

For ET, all we need to change is the amount of data points we consider. We thus obtain for the update of the data noise that

$$\sigma^{\text{new}}=\sqrt{\frac{1}{|\mathcal{M}|\,CD}\sum_{n\in\mathcal{M}}\left\langle\sum_{d=1}^{D}\sum_{c=1}^{C}\left(y_{dc}^{(n)}-\mathcal{T}_{dc}(S,\Theta)\right)^2\right\rangle_{q_n}}\,.$$

B.2 Optimization of the Prior Parameter

Now we will derive the M-Step equation for the update of the parameter π as follows:

$$\begin{aligned}
\frac{\partial}{\partial\pi}\mathcal{F}(\Theta,q)&=\sum_{n=1}^{N}\Bigg[\sum_{S}\ q_n(S\,;\Theta')\left[\frac{\partial}{\partial\pi}|\vec{s}|\log(\pi)+\frac{\partial}{\partial\pi}(H-|\vec{s}|)\log(1-\pi)\right]\Bigg]\\
&=\sum_{n=1}^{N}\Bigg[\sum_{S}\ q_n(S\,;\Theta')\left[\frac{|\vec{s}|}{\pi}-\frac{H-|\vec{s}|}{1-\pi}\right]\Bigg]\\
&=\sum_{n=1}^{N}\sum_{S}\ q_n(S\,;\Theta')\frac{|\vec{s}|-H\pi}{\pi(1-\pi)}\stackrel{!}{=}0
\end{aligned}$$

$$\begin{aligned}
\Rightarrow\quad&\sum_{n=1}^{N}\sum_{S}\ q_n(S\,;\Theta')|\vec{s}|=H\pi N\\
\Rightarrow\quad&\pi=\frac{1}{NH}\sum_{n=1}^{N}\sum_{S}\ q_n(S\,;\Theta')|\vec{s}|
\end{aligned}$$

With ET, we have to introduce a normalization factor, A, which changes $p(S|\pi)$ to

$$p_{\mathrm{ET}}(S|\pi) = \begin{cases} \frac{1}{A}\, p(S|\pi), & |\vec{s}| \leq \gamma \\ 0, & |\vec{s}| > \gamma \end{cases} .$$

With $\sum_S p_{\mathrm{ET}}(S|\pi) = \sum_{S \in \mathcal{K}_n} \frac{1}{A}\, p(S|\pi) \approx \sum_{S;|\vec{s}|<\gamma} \frac{1}{A}\, p(S|\pi) \stackrel{!}{=} 1$, we find that

$$\begin{aligned} A = \sum_{S,\,|\vec{s}|\leq\gamma} p(S|\,\pi) &= \sum_{S,\,|\vec{s}|\leq\gamma} \pi^{|\vec{s}|}(1-\pi)^{(H-|\vec{s}|)} \frac{1}{|\vec{s}|!} \\ &= \big[1 \times \pi^0(1-\pi)^H + H \times \pi^1(1-\pi)^{(H-1)} \\ &\quad + H(H-1) \times \pi^2(1-\pi)^{(H-2)} \frac{1}{2!} + \ldots\big] \\ &= \sum_{\gamma'=0}^{\gamma} \binom{H}{\gamma'} \pi^{\gamma'} (1-\pi)^{H-\gamma'} . \end{aligned}$$

As we are going to need it below, we define

$$B(\pi) := \sum_{\gamma'=0}^{\gamma} \gamma' \binom{H}{\gamma'} \pi^{\gamma'} (1-\pi)^{H-\gamma'}$$

and also calculate the derivative of A w.r.t. π as follows:

$$\begin{aligned} \frac{\partial}{\partial\pi} A(\pi) &= \sum_{\gamma'=0}^{\gamma} \binom{H}{\gamma'} \left[\frac{\gamma'}{\pi}\, \pi^{\gamma'}(1-\pi)^{(H-\gamma')} - \pi^{\gamma'}(1-\pi)^{(H-\gamma')}\, \frac{H-\gamma'}{1-\pi} \right] \\ &= \sum_{\gamma'=0}^{\gamma} \binom{H}{\gamma'} \left[\pi^{\gamma'}(1-\pi)^{(H-\gamma')} \left(\frac{\gamma'}{\pi} - \frac{H}{1-\pi} + \frac{\gamma'}{1-\pi} \right) \right] \\ &= \frac{1}{\pi} \sum_{\gamma'=0}^{\gamma} \gamma' \binom{H}{\gamma'} \pi^{\gamma'}(1-\pi)^{(H-\gamma')} \\ &\quad - \frac{H}{1-\pi} \sum_{\gamma'=0}^{\gamma} \binom{H}{\gamma'} \pi^{\gamma'}(1-\pi)^{(H-\gamma')} \\ &\quad + \frac{1}{1-\pi} \sum_{\gamma'=0}^{\gamma} \gamma' \binom{H}{\gamma'} \pi^{\gamma'}(1-\pi)^{(H-\gamma')} \\ &= \frac{1}{\pi} B(\pi) - \frac{H}{1-\pi} A(\pi) + \frac{1}{1-\pi} B(\pi) \\ &= \frac{B(\pi)}{\pi(1-\pi)} - \frac{HA(\pi)}{1-\pi} \end{aligned}$$

As we now take the derivative of the ET prior, we find that

$$\begin{aligned}
\frac{\partial}{\partial\pi}\log\left(p_{\mathrm{ET}}(S|\pi)\right) &= \frac{\partial}{\partial\pi}\log\left[\frac{1}{A(\pi)}\,\pi^{|\vec{s}|}(1-\pi)^{(H-|\vec{s}|)}\frac{1}{|\vec{s}|!}\right] \\
&= \frac{|\vec{s}|}{\pi} - \frac{H-|\vec{s}|}{1-\pi} - \frac{\partial}{\partial\pi}\log A(\pi) \\
&= \frac{|\vec{s}|}{\pi} - \frac{H-|\vec{s}|}{1-\pi} - \frac{1}{A(\pi)}\frac{\partial}{\partial\pi}A(\pi) \\
&= \frac{|\vec{s}|}{\pi} - \frac{H-|\vec{s}|}{1-\pi} - \frac{B(\pi)}{A(\pi)\pi(1-\pi)} + \frac{H}{1-\pi} \\
&= \frac{|\vec{s}|}{\pi} - \frac{H}{1-\pi} + \frac{|\vec{s}|}{1-\pi} - \frac{B(\pi)}{A(\pi)\pi(1-\pi)} + \frac{H}{1-\pi} \\
&= \frac{|\vec{s}|}{\pi(1-\pi)} - \frac{B(\pi)}{A(\pi)\pi(1-\pi)}\,.
\end{aligned}$$

We now have to set the free-energy with this expression equal to zero:

$$\begin{aligned}
\sum_{n\in\mathcal{M}}\sum_{S\in\mathcal{K}_n} q^{(n)}(S,\Theta')\left[\frac{|\vec{s}|}{\pi(1-\pi)} - \frac{B(\pi)}{A(\pi)\pi(1-\pi)}\right] &\stackrel{!}{=} 0 \\
\Rightarrow \sum_{n\in\mathcal{M}}\sum_{S\in\mathcal{K}_n} q^{(n)}(S,\Theta')|\vec{s}| &= \frac{B(\pi)|\mathcal{M}|}{A(\pi)} \\
\Leftrightarrow \frac{A(\pi)}{B(\pi)|\mathcal{M}|}\sum_{n\in\mathcal{M}}\sum_{S\in\mathcal{K}_n} q^{(n)}(S,\Theta')|\vec{s}| &= 1
\end{aligned}$$

In a fixed-point sense, this expression can be multiplied with π on both sides, one representing the updated π_{new} and one the old π from the iteration before:

$$\pi_{\mathrm{new}} = \frac{A(\pi)\,\pi}{B(\pi)}\,\frac{1}{|\mathcal{M}|}\sum_{n\in\mathcal{M}}\langle|\vec{s}|\rangle_{q_n}$$

B.3 Optimization of the Basis Functions

For the M-step equations of the mask and feature parameters, we observe that

$$\begin{aligned}
\frac{\partial}{\partial W_{id}}\mathcal{F}(\Theta,q) =& \frac{1}{2\sigma^2}\sum_{n=1}^{N}\sum_{S}\; q_n(S\,;\Theta')\frac{\partial}{\partial W_{id}}\sum_{d=1}^{D}\sum_{c=1}^{C}\left(y_{cd}^{(n)} - \mathcal{T}_{cd}(S,\Theta)\right)^2 \\
=& -\frac{1}{\sigma^2}\sum_{n=1}^{N}\sum_{S}\; q_n(S\,;\Theta')\sum_{c=1}^{C}\left(y_{cd}^{(n)} - \mathcal{T}_{cd}(S,\Theta)\right)\frac{\partial}{\partial W_{id}}\mathcal{T}_{cd}(S,\Theta)
\end{aligned}$$

and

$$\begin{aligned}
\frac{\partial}{\partial T_{ic}}\mathcal{F}(\Theta,q) =& \frac{1}{2\sigma^2}\sum_{n=1}^{N}\sum_{S}\; q_n(S\,;\Theta')\frac{\partial}{\partial T_{ic}}\sum_{d=1}^{D}\sum_{c=1}^{C}\left(y_{cd}^{(n)} - \mathcal{T}_{cd}(S,\Theta)\right)^2 \\
=& -\frac{1}{\sigma^2}\sum_{n=1}^{N}\sum_{S}\; q_n(S\,;\Theta')\sum_{d=1}^{D}\left(y_{cd}^{(n)} - \mathcal{T}_{cd}(S,\Theta)\right)\frac{\partial}{\partial T_{ic}}\mathcal{T}_{cd}(S,\Theta)\,.
\end{aligned}$$

We thus have to calculate the derivative of the combination rule. Since the original non-linear combination rule is not differentiable, we calculate the derivative of the approximated function and find that

$$\begin{aligned}
\frac{\partial}{\partial W_{id}}\mathcal{T}^{\rho}_{cd}(S,\Theta) =& \frac{\partial}{\partial W_{id}}\frac{\sum_{h=1}^{H}(\tau(S,h)\,W_{hd})^{\rho}\,W_{hd}\,T_{hc}}{\sum_{h=1}^{H}(\tau(S,h)\,W_{hd})^{\rho}}\left[=\frac{u'v}{v^2}-\frac{uv'}{v^2}\right]\\
=& \frac{(\tau(S,i)\,W_{id})^{\rho}T_{ic}(\rho+1)\times\sum_{h=1}^{H}(\tau(S,h)\,W_{hd})^{\rho}}{\left(\sum_{h=1}^{H}(\tau(S,h)\,W_{hd})^{\rho}\right)^2}\\
&-\frac{\sum_{h=1}^{H}(\tau(S,h)\,W_{hd})^{\rho}\,W_{hd}\,T_{hc}\times\rho\,(\tau(S,i)\,W_{id})^{\rho-1}\,\tau(S,i)}{\left(\sum_{h=1}^{H}(\tau(S,h)\,W_{hd})^{\rho}\right)^2}\\
=& \frac{(\tau(S,i)\,W_{id})^{\rho}T_{ic}(\rho+1)}{\sum_{h=1}^{H}(\tau(S,h)\,W_{hd})^{\rho}}-\mathcal{T}^{\rho}_{cd}(S,\Theta)\times\rho\,(\tau(S,i)\,W_{id})^{\rho-1}\,\tau(S,i)\\
=& \ldots .
\end{aligned}$$

As this derivation does not result in an analytically tractable solution, we introduce another approximation: The prefactor $(\tau(S,h)\,W_{hd})^{\rho}$ together with the normalizing denominator $\sum_{h=1}^{H}(\tau(S,h)\,W_{hd})^{\rho}$ simulates a differentiable step-function, i.e., its derivative will be zero almost everywhere, except for close to the point where the actual 'step' is where it is infinitely large. We will thus treat this entity as a constant prefactor. We obtain that

$$\begin{aligned}
\frac{\partial}{\partial W_{id}}\mathcal{T}^{\rho}_{cd}(S,\Theta) =& \frac{\sum_{h=1}^{H}(\tau(S,h)\,W_{hd})^{\rho}\,\frac{\partial}{\partial W_{id}}W_{hd}\,T_{hc}}{\sum_{h=1}^{H}(\tau(S,h)\,W_{hd})^{\rho}}\\
=& \frac{(\tau(S,i)\,W_{id})^{\rho}\,T_{ic}}{\sum_{h=1}^{H}(\tau(S,h)\,W_{hd})^{\rho}}\\
=& \mathcal{A}^{\rho}_{id}(S,W)\,T_{ic}\,,
\end{aligned}$$

where we defined for convenience that

$$\mathcal{A}^{\rho}_{id}(S,W) := \frac{(\tau(S,i)\,W_{id})^{\rho}}{\sum_{h=1}^{H}(\tau(S,h)\,W_{hd})^{\rho}} \text{ with } \mathcal{A}_{id}(S,W) := \lim_{\rho\to\infty}\mathcal{A}^{\rho}_{id}(S,W)\,.$$

For the feature parameter, we find that

$$\frac{\partial}{\partial T_{ic}}\mathcal{T}^{\rho}_{cd}(S,\Theta) = \mathcal{A}^{\rho}_{id}(S,W)\,W_{id}\,.$$

For large ρ, we find for a well-behaved function $f(t)$ that

$$\mathcal{A}^{\rho}_{id}(S,W)f(\mathcal{T}^{\rho}_{cd}(S,\Theta)) \approx \mathcal{A}^{\rho}_{id}(S,W)f(W_{id}T_{ic})\,.$$

When we insert this, together with the derivations above, into the free-energy, we observe that

$$\frac{\partial}{\partial W_{id}}\mathcal{F}(\Theta,q) = \frac{1}{\sigma^2}\sum_{n=1}^{N}\left[\sum_{S}\;q_n(S\,;\Theta')\sum_{c=1}^{C}\left(y^{(n)}_{cd}-W_{id}T_{ic}\right)\mathcal{A}^{\rho}_{id}(S,W)T_{ic}\right]\stackrel{!}{=}0$$

and

$$\frac{\partial}{\partial T_{ic}}\mathcal{F}(\Theta,q)=\frac{1}{\sigma^2}\sum_{n=1}^{N}\Big[\sum_{S}\;q_n(S\,;\Theta')\sum_{d=1}^{D}\Big(y_{cd}^{(n)}-W_{id}T_{ic}\Big)\,\mathcal{A}_{id}^{\rho}(S,W)W_{id}\Big]\stackrel{!}{=}0\,.$$

Then it follows that

$$\sum_{n=1}^{N}\sum_{S}\;q_n(S\,;\Theta')\mathcal{A}_{id}^{\rho}(S,W)\vec{T}_i^{T}\vec{y}_d^{\,(n)}=\sum_{n=1}^{N}\sum_{S}\;q_n(S\,;\Theta')\mathcal{A}_{id}^{\rho}(S,W)W_{id}\vec{T}_i^{T}\vec{T}_i$$

and

$$\sum_{n=1}^{N}\sum_{S}\;q_n(S\,;\Theta')\sum_{d=1}^{D}\mathcal{A}_{id}^{\rho}(S,W)y_{cd}^{(n)}W_{id}=\sum_{n=1}^{N}\sum_{S}\;q_n(S\,;\Theta')\sum_{d=1}^{D}\mathcal{A}_{id}^{\rho}(S,W)T_{ic}(W_{id})^2\,.$$

After a transformation, we find that

$$\sum_{n=1}^{N}\big\langle\mathcal{A}_{id}^{\rho}(S,W)\big\rangle_{q_n}\vec{T}_i^{T}\vec{y}_d^{\,(n)}=W_{id}\sum_{n=1}^{N}\big\langle\mathcal{A}_{id}^{\rho}(S,W)\big\rangle_{q_n}\vec{T}_i^{T}\vec{T}_i$$

and

$$\sum_{n=1}^{N}\sum_{d=1}^{D}\big\langle\mathcal{A}_{id}^{\rho}(S,W)\big\rangle_{q_n}y_{cd}^{(n)}W_{id}=T_{ic}\sum_{n=1}^{N}\sum_{d=1}^{D}\big\langle\mathcal{A}_{id}^{\rho}(S,W)\big\rangle_{q_n}(W_{id})^2\,.$$

Solving for the feature and mask parameters, we then obtain the necessary conditions for a maximum of the free-energy that need to hold in the limit $\rho\rightarrow\infty$. They are given as follows:

$$W_{id}=\frac{\displaystyle\sum_{n=1}^{N}\big\langle\mathcal{A}_{id}^{\rho}(S,W)\big\rangle_{q_n}\vec{T}_i^{T}\vec{y}_d^{\,(n)}}{\displaystyle\sum_{n=1}^{N}\big\langle\mathcal{A}_{id}^{\rho}(S,W)\big\rangle_{q_n}\vec{T}_i^{T}\vec{T}_i}$$

and

$$T_{ic}=\frac{\displaystyle\sum_{n=1}^{N}\sum_{d=1}^{D}\big\langle\mathcal{A}_{id}^{\rho}(S,W)\big\rangle_{q_n}W_{id}\,y_{cd}^{(n)}}{\displaystyle\sum_{n=1}^{N}\sum_{d=1}^{D}\big\langle\mathcal{A}_{id}^{\rho}(S,W)\big\rangle_{q_n}(W_{id})^2}\,.$$

For ET, we need to restrict the sums over the data points to only those summands corresponding to data points which can be expected to be explained by less or equal γ causes, i.e., data points which are in the set $\mathcal{M}$. The resulting update equations are given as follows:

$$W_{id}^{\text{new}}=\frac{\displaystyle\sum_{n\in\mathcal{M}}\langle\mathcal{A}_{id}(S,W)\rangle_{q_n}\vec{T}_i^{T}\vec{y}_d^{\,(n)}}{\displaystyle\sum_{n\in\mathcal{M}}\langle\mathcal{A}_{id}(S,W)\rangle_{q_n}\vec{T}_i^{T}\vec{T}_i}\,,\qquad\vec{T}_i^{\,\text{new}}=\frac{\displaystyle\sum_{n\in\mathcal{M}}\sum_{d}\langle\mathcal{A}_{id}(S,W)\rangle_{q_n}W_{id}\,\vec{y}_d^{\,(n)}}{\displaystyle\sum_{n\in\mathcal{M}}\sum_{d}\langle\mathcal{A}_{id}(S,W)\rangle_{q_n}(W_{id})^2}\,.$$

Appendix C. Influence of the Basis Functions on the Data Noise

Notably, as we alter the mask and feature values during learning, these new values have an effect on the value for σ. More specifically, we have seen that optimization runs resulting in low values for sigma closely correspond to a representation of the true underlying causes while runs with comparably high sigma values do not. For data such as provided by the bars test the final sigma values for different runs may even form corresponding clusters (Figure 3). The interplay between sigma values and object parameters will be investigated here in more detail: As we compare the derivative of the free-energy w.r.t. the masks and features

$$\frac{\partial}{\partial W_{id}}\mathcal{F}(\Theta, q) = -\frac{1}{\sigma^2}\sum_{n=1}^{N}\sum_{S} q_n(S\,;\Theta')\sum_{d=1}^{D}\sum_{c=1}^{C}\left(y_{cd}^{(n)} - \mathcal{T}_{cd}(S,\Theta)\right)\frac{\partial}{\partial W_{id}}\mathcal{T}_{cd}(S,\Theta)$$

and

$$\frac{\partial}{\partial T_{ic}}\mathcal{F}(\Theta, q) = -\frac{1}{\sigma^2}\sum_{n=1}^{N}\sum_{S} q_n(S\,;\Theta')\sum_{d=1}^{D}\sum_{c=1}^{C}\left(y_{cd}^{(n)} - \mathcal{T}_{cd}(S,\Theta)\right)\frac{\partial}{\partial T_{ic}}\mathcal{T}_{cd}(S,\Theta)$$

with the derivative of the obtained update rule for the data noise squared, again w.r.t. both basis function parameters

$$\frac{\partial}{\partial W_{id}}\sigma^2 = \frac{2}{NCD}\sum_{n=1}^{N}\sum_{S} q_n(S\,;\Theta')\sum_{d=1}^{D}\sum_{c=1}^{C}\left(y_{cd}^{(n)} - \mathcal{T}_{cd}(S,\Theta)\right)\frac{\partial}{\partial W_{id}}\mathcal{T}_{cd}(S,\Theta)$$

$$\frac{\partial}{\partial T_{ic}}\sigma^2 = \frac{2}{NCD}\sum_{n=1}^{N}\sum_{S} q_n(S\,;\Theta')\sum_{d=1}^{D}\sum_{c=1}^{C}\left(y_{cd}^{(n)} - \mathcal{T}_{cd}(S,\Theta)\right)\frac{\partial}{\partial T_{ic}}\mathcal{T}_{cd}(S,\Theta)$$

we find that these are virtually identical, except for a pre-factor which will disappear when we set the derivatives equal to zero. The optimal values for the mask and feature vectors in terms of the free-energy thus also optimize the data noise.

Appendix D. Selection Function

A straightforward selection function is given by the posterior for only one active cause which we calculate through Bayes' rule as, i.e.,

$$p(S_h|Y^{(n)},\Theta) = \frac{p(Y^{(n)}\,|S_h,\Theta)p(S_h|\Theta)}{p(Y^{(n)}\,|\Theta)}$$

where $S^{(h)} := (\vec{s}^{\,(h)},\varphi)$ with $\vec{s}^{\,(h)}$ being the state with only the hth object present.

Since we are only interested in comparing the numbers per data point, a normalization w.r.t. $p(Y^{(n)}\,|\Theta)$ is not required and does not have to be calculated for the selection function. Since the prior $p(S_h|\Theta)$ is identical for all S_h, we can omit that entity as well. We are, therefore, left with only the noise (or likelihood) term, i.e.,

$$p(Y^{(n)}\,|\,S_h,\Theta) = \left(2\pi\sigma^2\right)^{-\frac{CD}{2}}\prod_{d=1}^{D}\prod_{c=1}^{C}\exp\left(-\frac{1}{2\sigma^2}\left(y_{cd}^{(n)} - \mathcal{T}_{cd}(S_h,\Theta)\right)^2\right).$$

Since the logarithm is a strictly monotonic function, we instead can consider

$$\log\Big(p(Y^{(n)} \mid S_h, \Theta)\Big) = -\frac{CD}{2}\log\left(2\pi\sigma^2\right) - \frac{1}{2\sigma^2}\sum_{d=1}^{D}\sum_{c=1}^{C}\Big(y_{cd}^{(n)} - \mathcal{T}_{cd}(S_h, \Theta)\Big)^2 .$$

As the first term is constant for all causes, as is the prefactor $-\frac{1}{2\sigma^2}$, we omit these and are then left with

$$\mathcal{S}_h\big(Y^{(n)}\big) = -\sum_{d=1}^{D}\sum_{c=1}^{C}\big(y_{cd}^{(n)} - \mathcal{T}_{cd}(S_h; \Theta)\big)^2,$$

which is the function used to select the most likely hidden units given $Y^{(n)}$ (compare Equation 19).

References

P. Berkes, R. Turner, and M. Sahani. On sparsity and overcompleteness in image models. In *Advances in Neural Information Processing Systems*, volume 20, pages 89–96, 2008.

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.

J. Bornschein, M. Henniges, and J. Lücke. Are V1 receptive fields shaped by low-level visual occlusions? A comparative study. *PLOS Computational Biology*, 9(6):e1003062, 2013.

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

C. Eckes, J. Triesch, and C. von der Malsburg. Analysis of cluttered scenes using an elastic matching approach for stereo images. *Neural Computation*, 18(6):1441–1471, 2006.

J. Eggert, H. Wersing, and E. Körner. Transformation-invariant representation and NMF. In *2004 IEEE International Joint Conference on Neural Networks*, pages 2535–39, 2004.

G. Exarchakis, M. Henniges, J. Eggert, and J. Lücke. Ternary sparse coding. In *Proceedings LVA/ICA*, pages 204–212, 2012.

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

A. A. Frolov, D. Husek, and P. Y. Polyakov. Two expectation-maximization algorithms for Boolean factor analysis. *Neurocomputing*, 130:83–97, 2014.

K. Fukushima. Restoring partly occluded patterns: a neural network model. *Neural Networks*, 18(1):33–43, 2005.

K. Gregor and Y. LeCun. Efficient learning of sparse invariant representations. *CoRR*, abs/1105.5307, 2011.

M. Henniges, G. Puertas, J. Bornschein, J. Eggert, and J. Lücke. Binary sparse coding. In *Proceedings LVA/ICA*, LNCS 6365, pages 450–57. Springer, 2010.

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

A. Honkela and H. Valpola. Unsupervised variational Bayesian learning of nonlinear models. In *Advances in Neural Information Processing Systems*, volume 17, pages 593–600, 2005.

P. O. Hoyer. Non-negative matrix factorization with sparseness constraints. *Journal of Machine Learning Research*, 5:1457–69, 2004.

N. Jojic and B. Frey. Learning flexible sprites in video layers. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 199–206, 2001.

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

N. LeRoux, N. Heess, J. Shotton, and J. Winn. Learning a generative model of images by factoring appearance and shape. *Neural Computation*, 23:593–650, 2011.

D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

J. Lücke and J. Eggert. Expectation truncation and the benefits of preselection in training generative models. *Journal of Machine Learning Research*, 11:2855–900, 2010.

J. Lücke and C. Malsburg. Rapid processing and unsupervised learning in a model of the cortical macrocolumn. *Neural Computation*, 16:501–33, 2004.

J. Lücke and M. Sahani. Maximal causes for non-linear component extraction. *Journal of Machine Learning Research*, 9:1227–67, 2008.

J. Lücke, R. Turner, M. Sahani, and M. Henniges. Occlusive components analysis. In *Advances in Neural Information Processing Systems*, volume 22, pages 1069–77, 2009.

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.

S. Nene, S. Nayar, and H. Murase. Columbia object image library (COIL 100). 1996.

B. Olshausen and D. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381:607–9, 1996.

G. Puertas, J. Bornschein, and J. Lücke. The maximal causes of natural scenes are edge filters. In *Advances in Neural Information Processing Systems*, volume 23, pages 1939–47. 2010.

M. Sahani. *Latent Variable Models for Neural Data Analysis*. PhD thesis, Caltech, 1999.

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

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

S. Tajima and M. Watanabe. Acquisition of nonlinear forward optics in generative models: Two-stage "downside-up" learning for occluded vision. *Neural Networks*, 24(2):148–58, 2011.

R. Turner and M. Sahani. Two problems with variational expectation maximisation for time-series models. In D. Barber, T. Cemgil, and S. Chiappa, editors, *Bayesian time series models*, chapter 5, pages 109–130. Cambridge University Press, 2011.

N. Ueda and R. Nakano. Deterministic annealing EM algorithm. *Neural Networks*, 11(2): 271–82, 1998.

H. Valpola and J. Karhunen. An unsupervised ensemble learning method for nonlinear dynamic state-space models. *Neural Computation*, 14(11):2647–2692, 2002.

T. Šingliar and M. Hauskrecht. Noisy-or component analysis and its application to link analysis. *Journal of Machine Learning Research*, 7:2189–2213, 2006.

H. Wersing and E. Körner. Learning optimized features for hierarchical models of invariant object recognition. *Neural Computation*, 15(7):1559–88, 2003.

C. K. I. Williams and M. K. Titsias. Greedy learning of multiple objects in images using robust statistics and factorial learning. *Neural Computation*, 16:1039–62, 2004.

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