Journal of Machine Learning Research 15 (2014) 2653-2687 Submitted 11/12; Revised 12/13; Published 8/14

A Truncated EM Approach for Spike-and-Slab Sparse Coding

Abdul-Saboor Sheikh[*] SHEIKH@TU-BERLIN.DE
Jacquelyn A. Shelton[*] SHELTON@TU-BERLIN.DE
Faculty of Electrical Engineering and Computer Science
Technical University Berlin
Marchstr. 23, 10587 Berlin, Germany

Jörg Lücke[*] JOERG.LUECKE@UNI-OLDENBURG.DE
Cluster of Excellence Hearing4all and Faculty VI
University of Oldenburg
26115 Oldenburg, Germany

Editor: Aapo Hyvärinen

Abstract

We study inference and learning based on a sparse coding model with 'spike-and-slab' prior. As in standard sparse coding, the model used assumes independent latent sources that linearly combine to generate data points. However, instead of using a standard sparse prior such as a Laplace distribution, we study the application of a more flexible 'spike-and-slab' distribution which models the absence or presence of a source's contribution independently of its strength if it contributes. We investigate two approaches to optimize the parameters of spike-and-slab sparse coding: a novel truncated EM approach and, for comparison, an approach based on standard factored variational distributions. The truncated approach can be regarded as a variational approach with truncated posteriors as variational distributions. In applications to source separation we find that both approaches improve the state-of-the-art in a number of standard benchmarks, which argues for the use of 'spike-and-slab' priors for the corresponding data domains. Furthermore, we find that the truncated EM approach improves on the standard factored approach in source separation tasks—which hints to biases introduced by assuming posterior independence in the factored variational approach. Likewise, on a standard benchmark for image denoising, we find that the truncated EM approach improves on the factored variational approach. While the performance of the factored approach saturates with increasing numbers of hidden dimensions, the performance of the truncated approach improves the state-of-the-art for higher noise levels.

Keywords: sparse coding, spike-and-slab distributions, approximate EM, variational Bayes, unsupervised learning, source separation, denoising

[*] Parts of this study were done at the Frankfurt Institute for Advanced Studies, Goethe-University Frankfurt am Main, Germany, the previous affiliation of the authors.

©2014 Abdul-Saboor Sheikh, Jacquelyn A. Shelton and Jörg Lücke.

1. Introduction

Much attention has recently been devoted to studying sparse coding models with 'spike-and-slab' distribution as a prior over the latent variables (Goodfellow et al., 2013; Mohamed et al., 2012; Lücke and Sheikh, 2012; Titsias and Lazaro-Gredilla, 2011; Carbonetto and Stephen, 2011; Knowles and Ghahramani, 2011; Yoshida and West, 2010). In general, a 'spike-and-slab' distribution is comprised of a binary (the 'spike') and a continuous (the 'slab') part. The distribution generates a random variable by multiplying together the two parts such that the resulting value is either exactly zero (due to the binary random variable being zero) or it is a value drawn from a distribution governing the continuous part. In sparse coding models, employing spike-and-slab as a prior allows for modeling the presence or absence of latents independently of their contributions in generating an observation. For example, piano keys (as latent variables) are either pressed or not (binary part), and if they are pressed, they result in sounds with different intensities (continuous part). The sounds generated by a piano are also sparse in the sense that of all keys only a relatively small number is pressed on average.

Spike-and-slab distributions can flexibly model an array of sparse distributions, making them desirable for many types of data. Algorithms based on spike-and-slab distributions have successfully been used, e.g., for deep learning and transfer learning (Goodfellow et al., 2013), regression (West, 2003; Carvalho et al., 2008; Carbonetto and Stephen, 2011; Titsias and Lazaro-Gredilla, 2011), or denoising (Zhou et al., 2009; Titsias and Lazaro-Gredilla, 2011), and often represent the state-of-the-art on given benchmarks (compare Titsias and Lazaro-Gredilla, 2011; Goodfellow et al., 2013).

The general challenge with spike-and-slab sparse coding models lies in the optimization of the model parameters. Whereas the standard Laplacian prior used for sparse coding results in uni-modal posterior distributions, the spike-and-slab prior results in multi-modal posteriors (see, e.g., Titsias and Lazaro-Gredilla, 2011; Lücke and Sheikh, 2012). Figure 1 shows typical posterior distributions for spike-and-slab sparse coding (the model will be formally defined in the next section). The figure illustrates posterior examples for the case of a two-dimensional observed and a two-dimensional hidden space. As can be observed, the posteriors have multiple modes; and the number modes increases exponentially with the dimensionality of the hidden space (Titsias and Lazaro-Gredilla, 2011; Lücke and Sheikh, 2012). The multi-modal structure of the posteriors argues against the application of the standard maximum a-posteriori (MAP) approaches (Mairal et al., 2009; Lee et al., 2007; Olshausen and Field, 1997) or Gaussian approximations of the posterior (Seeger, 2008; Ribeiro and Opper, 2011) because they rely on uni-modal posteriors. The approaches that have been proposed in the literature are, consequently, MCMC based methods (e.g., Carvalho et al., 2008; Zhou et al., 2009; Mohamed et al., 2012) and variational EM methodologies (e.g., Zhou et al., 2009; Titsias and Lazaro-Gredilla, 2011; Goodfellow et al., 2013). While MCMC approaches are more general and more accurate given sufficient computational resources, variational approaches are usually more efficient. Especially in high dimensional hidden spaces, the multi-modality of the posteriors is a particular challenge for MCMC approaches; consequently, recent applications to large hidden spaces have been based on variational EM optimization (Titsias and Lazaro-Gredilla, 2011; Goodfellow et al., 2013). The variational approaches applied to spike-and-slab models thus far (see Rattray et al.,

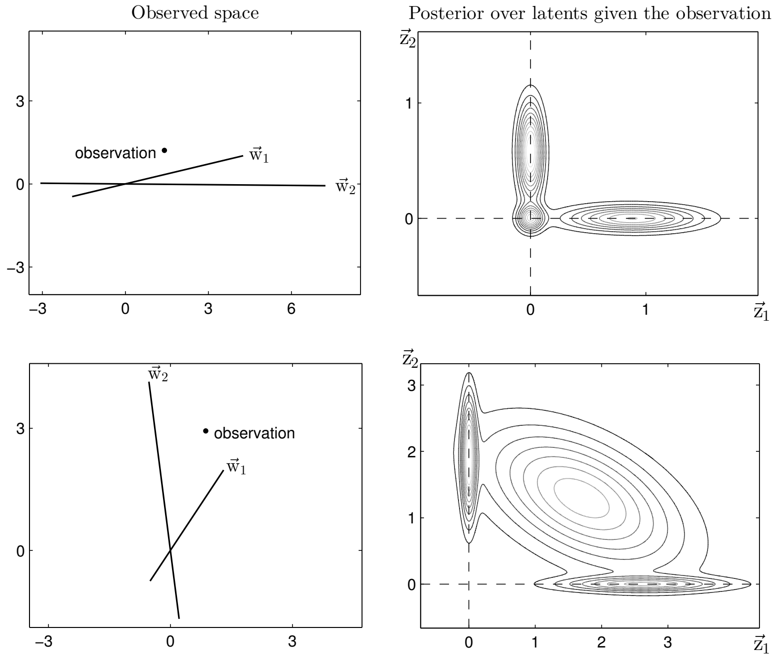

Figure 1: Left figures visualize observations generated by two different instantiations of the spike-and-slab sparse coding model (1) to (3). Solid lines are the generating bases vectors. Right figures illustrate the corresponding exact posteriors over latents computed using (13) and (15) given observations and generating model parameters. The probability mass seen just along the axes or around the origin actually lies exactly on the axis. Here we have spread the mass for visualization purposes by slightly augmenting zero diagonal entries of the posterior covariance matrix in (15).

2009; Yoshida and West, 2010; Titsias and Lazaro-Gredilla, 2011; Goodfellow et al., 2013) assume a factorization of the posteriors over the latent dimensions, that is the hidden dimensions are assumed to be independent a-posteriori. This means that any dependencies such as explaining-away effects including correlations (compare Figure 1) are ignored and not accounted for. But what consequences does such a negligence of posterior structure have? Does it result in biased parameter estimates and is it relevant for practical tasks? Biases induced by factored variational inference in latent variable models have indeed been observed before (MacKay, 2001; Ilin and Valpola, 2005; Turner and Sahani, 2011). For instance, in source separation tasks, optimization through factored inference can be biased towards finding mixing matrices that represent orthogonal sparse directions, because such solutions are most consistent with the assumed a-posteriori independence (see Ilin and Valpola, 2005, for a detailed discussion). Therefore, the posterior independence assumption in general may result in suboptimal solutions.

In this work we study an approximate EM approach for spike-and-slab sparse coding which does not assume a-posteriori independence and which can model multiple modes. The novel approach can be considered as a variational EM approach but instead of using factored distributions or Gaussians, it is based on posterior distributions truncated to regions of high probability mass (Lücke and Eggert, 2010). Such truncated EM approaches have recently been applied to different models (see e.g., Puertas et al., 2010; Shelton et al., 2011; Dai and Lücke, 2012; Bornschein et al., 2013). In contrast to the previously studied factored variational approaches (Titsias and Lazaro-Gredilla, 2011; Mohamed et al., 2012; Goodfellow et al., 2013), the truncated approach will furthermore take advantage of the fact that in the case of a Gaussian slab and Gaussian noise model, integrals over the continuous latents can be obtained in closed-form (Lücke and Sheikh, 2012). This implies that the posteriors over latent space can be computed exactly if the sums over the binary part are exhaustively evaluated over exponentially many states. This enumeration of the binary part becomes computationally intractable for high-dimensional hidden spaces. However, by applying the truncated variational approach exclusively to the binary part of the hidden space, we can still fully benefit from the analytical tractability of the continuous integrals.

In this study, we systematically compare the truncated approach to a recently suggested factored variational approach (Titsias and Lazaro-Gredilla, 2011). A direct comparison of the two variational approaches will allow for answering the questions about potential drawbacks and biases of both optimization procedures. As approaches assuming factored variational approximations have recently shown state-of-the-art performances (Titsias and Lazaro-Gredilla, 2011; Goodfellow et al., 2013), understanding their strengths and weaknesses is crucial for further advancements of sparse coding approaches and their many applications. Comparison with other approaches that are not necessarily based on the spike-and-slab model will allow for accessing the potential advantages of the spike-and-slab model itself.

In Section 2 we will introduce the used spike-and-slab sparse coding generative model, and briefly discuss the factored variational approach which has recently been applied for parameter optimization. In Section 3 we derive the closed-form EM parameter update equations for the introduced spike-and-slab model. Based on these equations, in Section 4 we derive the truncated EM algorithm for efficient learning in high dimensions. In Section 5, we numerically evaluate the algorithm and compare it to factored variational and other approaches. Finally, in Section 6 we discuss the results. The Appendix present details of the derivations and experiments.

2. Spike-and-slab Sparse Coding

The spike-and-slab sparse coding model assumes like standard sparse coding a linear superposition of basis functions, independent latents, and Gaussian observation noise. The main difference is that a spike-and-slab distribution is used as a prior. Spike-and-slab distributions have long been used for different models (e.g., Mitchell and Beauchamp, 1988, among many others) and also variants of sparse coding with spike-and-slab priors have been studied previously (compare West, 2003; Garrigues and Olshausen, 2007; Knowles and Ghahramani, 2007; Teh et al., 2007; Carvalho et al., 2008; Paisley and Carin, 2009; Zhou et al., 2009). In this work we study a generalization of the spike-and-slab sparse coding model studied by

Lücke and Sheikh (2012). The data generation process in the model assumes an independent Bernoulli prior for each component of the the binary latent vector $\vec{s} \in \{0,1\}^H$ and a multivariate Gaussian prior for the continuous latent vector $\vec{z} \in \mathbb{R}^H$:

$$
\begin{aligned}
p(\vec{s}\,|\Theta) &= \mathcal{B}(\vec{s};\vec{\pi}) = \prod_{h=1}^{H} \pi_h^{s_h}\,(1-\pi_h)^{1-s_h}, && (1)\\
p(\vec{z}\,|\Theta) &= \mathcal{N}(\vec{z};\,\vec{\mu},\Psi), && (2)
\end{aligned}
$$

where π_h defines the probability of s_h being equal to one and where $\vec{\mu}$ and Ψ parameterize the mean and covariance of $\vec{z}$, respectively. The parameters $\vec{\mu} \in \mathbb{R}^H$ and $\Psi \in \mathbb{R}^{H\times H}$ parameterizing the Gaussian slab in (2) generalize the spike-and-slab model used in (Lücke and Sheikh, 2012). A point-wise multiplication of the two latent vectors, i.e., $(\vec{s}\odot\vec{z})_h = s_h\,z_h$ generates a 'spike-and-slab' distributed variable $(\vec{s}\odot\vec{z})$, which has either continuous values or exact zero entries. Given such a latent vector, a D-dimensional observation $\vec{y} \in \mathbb{R}^D$ is generated by linearly superimposing a set of basis functions W and by adding Gaussian noise:

$$
p(\vec{y}\,|\,\vec{s},\vec{z},\Theta) = \mathcal{N}(\vec{y};\,W(\vec{s}\odot\vec{z}),\Sigma), \tag{3}
$$

where each column of the matrix $W \in \mathbb{R}^{D\times H}$ is a basis function $W = (\vec{w}_1,\ldots,\vec{w}_H)$ and where the matrix $\Sigma \in \mathbb{R}^{D\times D}$ parameterizes the observation noise. Full rank covariances Σ can flexibly parametrize noise and have been found beneficial in noisy environments (Dalen and Gales, 2008; Ranzato and Hinton, 2010; Dalen and Gales, 2011). Nevertheless the model can also be constrained to have homoscedastic noise (i.e., $\Sigma = \sigma^2 I$). We use $\Theta = (W,\Sigma,\vec{\pi},\vec{\mu},\Psi)$ to denote all the model parameters. Having a spike-and-slab prior implies that for high levels of sparsity (low values of π_h) the latents assume exact zeros with a high probability. This is an important distinction compared to the Laplace or Cauchy distributions used for standard sparse coding (Olshausen and Field, 1997).

The spike-and-slab sparse coding algorithm we derive in this work is based on the model (1) to (3). The factored variational approach (Multi–Task and Multiple Kernel Learning, MTMKL; Titsias and Lazaro-Gredilla, 2011) that we use for detailed comparison is based on a similar model. The MTMKL model is both a constrained and generalized version of the model we study. On one hand, it is more constrained by assuming the same sparsity for each latent, i.e., $\pi_h = \pi_{h'}$ (for all h,h'); and by using a diagonal covariance matrix for the observation noise, $\Sigma = \mathrm{diag}(\sigma_1^2,\ldots,\sigma_D^2)$. On the other hand, it is a generalization by drawing the basis functions W from Gaussian processes. The model (1) to (3) can then be recovered as a special case of the MTMKL model if the Gaussian processes are Dirac delta functions. For parameter optimization, the MTMKL model uses a standard factored variational optimization. In the case of spike-and-slab models, this factored approach means that the exact posterior $p(\vec{s},\vec{z}\,|\,\vec{y})$ is approximated by a variational distribution $q_n(\vec{s},\vec{z};\,\Theta)$ which assumes the combined latents to be independent a-posteriori (compare Zhou et al., 2009; Titsias and Lazaro-Gredilla, 2011; Goodfellow et al., 2013):

$$
q_n(\vec{s},\vec{z};\Theta) = \prod_{h=1}^{H} q_n^{(h)}(s_h,z_h;\Theta),
$$

where $q_n^{(h)}$ are distributions only depending on s_h and z_h and not on any of the other latents. A detailed account of the MTMKL optimization algorithm is given by Titsias and Lazaro-Gredilla (2011) and for later numerical experiments on the model, we used the source code provided along with that publication.[1] Further comparisons will include the spike-and-slab sparse coding model by Zhou et al. (2009). The generative model is similar to the spike-and-slab model in Equations (1) to (3) but uses a Beta process prior to parameterize the Bernoulli (the "spike") distribution and assumes homoscedastic observation noise. Inference in their model is based on factored variational EM or Gibbs sampling. As this model is closely related to ours, we use it as another instance for comparison in our numerical experiments in order to assess the influence of different inference method choices. This comparison allows us to explore differences of training the model with a sampling-based approach, as they yield many of the same benefits of our inference method (e.g., flexible representation of uncertainty), but where generally more computational resources are necessary.

3. Expectation Maximization for Parameter Optimization

In order to learn the model parameters Θ given a set of N independent data points $\{\vec{y}^{\,(n)}\}_{n=1,\dots,N}$ with $\vec{y}^{\,(n)} \in \mathbb{R}^D$, we maximize the data likelihood $\mathcal{L} = \prod_{n=1}^{N} p(\vec{y}^{\,(n)} \,|\, \Theta)$ by applying the Expectation Maximization (EM) algorithm. Instead of directly maximizing the likelihood, the EM algorithm (in the form studied by Neal and Hinton, 1998) maximizes the free-energy, a lower bound of the log-likelihood given by:

$$\mathcal{F}(\Theta^{\mathrm{old}}, \Theta) = \sum_{n=1}^{N} \Big\langle \log p(\vec{y}^{\,(n)}, \vec{s}, \vec{z} \,|\, \Theta) \Big\rangle_n + H(\Theta^{\mathrm{old}}), \tag{4}$$

where $\langle \,\cdot\, \rangle_n$ denotes the expectation under the posterior over the latents $\vec{s}$ and $\vec{z}$ given $\vec{y}^{\,(n)}$

$$\big\langle f(\vec{s}, \vec{z}) \big\rangle_n = \sum_{\vec{s}} \int_{\vec{z}} p(\vec{s}, \vec{z} \,|\, \vec{y}^{\,(n)}, \Theta^{\mathrm{old}})\, f(\vec{s}, \vec{z})\, \mathrm{d}\vec{z} \tag{5}$$

and $H(\Theta^{\mathrm{old}}) = -\sum_{\vec{s}} \int_{\vec{z}} p(\vec{s}, \vec{z} \,|\, \vec{y}^{\,(n)}, \Theta^{\mathrm{old}}) \log(p(\vec{s}, \vec{z} \,|\, \vec{y}^{\,(n)}, \Theta^{\mathrm{old}}))\, \mathrm{d}\vec{z}$ is the Shannon entropy, which only depends on parameter values held fixed during the optimization of $\mathcal{F}$ w.r.t. Θ in the M-step. Here $\sum_{\vec{s}}$ is a summation over all possible binary vectors $\vec{s}$.

The EM algorithm iteratively optimizes the free-energy by alternating between two steps. First, in the E-step given the current parameters Θ^{old}, the relevant expectation values under the posterior $p(\vec{s}, \vec{z} \,|\, \vec{y}^{\,(n)}, \Theta^{\mathrm{old}})$ are computed. Next, the M-step uses these posterior expectations and maximizes the free-energy $\mathcal{F}(\Theta^{\mathrm{old}}, \Theta)$ w.r.t. Θ. Iteratively applying E- and M-steps locally maximizes the data likelihood. In the following section we will first derive the M-step equations which themselves will require expectation values over the posteriors (5). The required expressions and approximations for these expectations (the E-step) will be derived afterwards.

3.1 M-step Parameter Updates

The M-step parameter updates of the model are canonically obtained by setting the derivatives of the free-energy (4) w.r.t. the second argument to zero. Details of the derivations

[1] We downloaded the code from `http://www.well.ox.ac.uk/~mtitsias/code/varSparseCode.tar.gz`.

are given in Appendix A and the resulting update equations are as follows:

$$W = \frac{\sum_{n=1}^{N} \vec{y}^{(n)} \big\langle \vec{s} \odot \vec{z} \big\rangle_n^{\mathrm{T}}}{\sum_{n=1}^{N} \big\langle (\vec{s} \odot \vec{z})(\vec{s} \odot \vec{z})^{\mathrm{T}} \big\rangle_n}, \tag{6}$$

$$\vec{\pi} = \frac{1}{N} \sum_{n=1}^{N} \big\langle \vec{s} \big\rangle_n, \tag{7}$$

$$\vec{\mu} = \frac{\sum_{n=1}^{N} \big\langle \vec{s} \odot \vec{z} \big\rangle_n}{\sum_{n=1}^{N} \big\langle \vec{s} \big\rangle_n}, \tag{8}$$

$$\Psi = \sum_{n=1}^{N} \Big[\big\langle (\vec{s} \odot \vec{z})(\vec{s} \odot \vec{z})^{\mathrm{T}} \big\rangle_n - \big\langle \vec{s}\,\vec{s}^{\mathrm{T}} \big\rangle_n \odot \vec{\mu}\vec{\mu}^{\mathrm{T}} \Big] \odot \Big(\sum_{n=1}^{N} \Big[\big\langle \vec{s}\,\vec{s}^{\mathrm{T}} \big\rangle_n \Big] \Big)^{-1}, \tag{9}$$

$$\text{and } \Sigma = \frac{1}{N} \sum_{n=1}^{N} \Big[\vec{y}^{(n)} (\vec{y}^{(n)})^{\mathrm{T}} - W \big[\big\langle (\vec{s} \odot \vec{z}) \big\rangle_n \big\langle (\vec{s} \odot \vec{z}) \big\rangle_n^{\mathrm{T}} \big] W^{\mathrm{T}} \Big]. \tag{10}$$

3.2 E-step Expectation Values

The M-step equations (6) to (10) require expectation values w.r.t. the posterior distribution be computed over the whole latent space, which requires either analytical solutions or approximations of integrals/sums over the latent space. For the derivation of closed-form E-step equations it is useful to know that the discrete latent variable $\vec{s}$ can be combined with the basis function matrix W so that we can rewrite (3) as

$$p(\vec{y} \,|\, \vec{s}, \vec{z}, \Theta) = \mathcal{N}(\vec{y};\, \tilde{W}_{\vec{s}}\, \vec{z}, \Sigma),$$

where we have defined $(\tilde{W}_{\vec{s}})_{dh} = W_{dh} s_h$ such that $W(\vec{s} \odot \vec{z}) = \tilde{W}_{\vec{s}}\, \vec{z}$.

Here the data likelihood $p(\vec{y} \,|\, \Theta)$ can be derived in closed-form after marginalizing the joint $p(\vec{y}, \vec{s}, \vec{z} \,|\, \Theta)$ over $\vec{z}$:

$$\begin{aligned} p(\vec{y}, \vec{s} | \, \Theta) &= \mathcal{B}(\vec{s}; \vec{\pi}) \int \mathcal{N}(\vec{y}; \tilde{W}_{\vec{s}}\, \vec{z}, \Sigma)\, \mathcal{N}(\vec{z}; \vec{\mu}, \Psi)\, \mathrm{d}\vec{z} \\ &= \mathcal{B}(\vec{s}; \vec{\pi})\ \mathcal{N}(\vec{y}; \tilde{W}_{\vec{s}}\, \vec{\mu}, C_{\vec{s}}), \end{aligned} \tag{11}$$

where $C_{\vec{s}} = \Sigma + \tilde{W}_{\vec{s}} \Psi \tilde{W}_{\vec{s}}^T$. The second step follows from standard identities for Gaussian random variables (e.g., Bishop, 2006). We can then sum the resulting expression over $\vec{s}$ to obtain

$$p(\vec{y} \,|\, \Theta) = \sum_{\vec{s}} \mathcal{B}(\vec{s}; \vec{\pi})\ \mathcal{N}(\vec{y}; \tilde{W}_{\vec{s}}\, \vec{\mu}, C_{\vec{s}}). \tag{12}$$

Thus, the marginal distribution takes the form of a Gaussian mixture model with 2^H mixture components indexed by $\vec{s}$. However, unlike in a standard Gaussian mixture model, the mixing proportions and the parameters of the mixture components are not independent but coupled together. Therefore, the following steps will lead to closed-form EM updates that are notably not a consequence of closed-form EM for classical Gaussian mixtures. In contrast, Gaussian mixture models assume independent mixing proportions and independent component parameters. By using Equations (11) and (12) the posterior over the binary latents $p(\vec{s}\,|\,\vec{y},\Theta)$ is given by:

$$p(\vec{s}\,|\,\vec{y},\Theta) = \frac{p(\vec{s},\vec{y}\,|\,\Theta)}{p(\vec{y}\,|\,\Theta)} = \frac{\mathcal{B}(\vec{s};\vec{\pi})\;\mathcal{N}(\vec{y};\tilde{W}_{\vec{s}}\,\vec{\mu},C_{\vec{s}})}{\sum_{\vec{s}'}\mathcal{B}(\vec{s}';\vec{\pi})\;\mathcal{N}(\vec{y};\,\tilde{W}_{\vec{s}'}\,\vec{\mu},C_{\vec{s}'})}. \tag{13}$$

We can now consider the factorization of the posterior $p(\vec{s},\vec{z}\,|\,\vec{y},\Theta)$ into the posterior over the binary part $p(\vec{s}\,|\,\vec{y},\Theta)$ and the posterior over the continuous part given the binary state $p(\vec{z}\,|\,\vec{s},\vec{y},\Theta)$:

$$p(\vec{s},\vec{z}\,|\,\vec{y},\Theta) = p(\vec{s}\,|\,\vec{y},\Theta)\,p(\vec{z}\,|\,\vec{s},\vec{y},\Theta). \tag{14}$$

Like the first factor in (14), the second factor is also analytically tractable and given by:

$$\begin{aligned}p(\vec{z}\,|\,\vec{s},\vec{y},\Theta) &= \frac{p(\vec{s}\,|\,\Theta)\,p(\vec{z}\,|\,\Theta)\,p(\vec{y}\,|\,\vec{z},\vec{s},\Theta)}{p(\vec{s}\,|\,\Theta)\int p(\vec{y}\,|\,\vec{z},\vec{s},\Theta)\,p(\vec{z}\,|\,\Theta)\mathrm{d}\vec{z}}\\ &\propto \mathcal{N}(\vec{z};\vec{\mu},\Psi)\,\mathcal{N}(\vec{y};\tilde{W}_{\vec{s}}\,\vec{z},\Sigma)\\ &= \mathcal{N}(\vec{z};\vec{\kappa}_{\vec{s}},\Lambda_{\vec{s}}),\end{aligned}$$

where the last step again follows from standard Gaussian identities with definitions

$$\begin{aligned}\Lambda_{\vec{s}} &= (\tilde{W}_{\vec{s}}^{\mathrm{T}}\,\Sigma^{-1}\,\tilde{W}_{\vec{s}}\,+\,\Psi_{\vec{s}}^{-1})^{-1},\\ \vec{\kappa}_{\vec{s}}^{(n)} &= (\vec{s}\odot\vec{\mu}) + \Lambda_{\vec{s}}\,\tilde{W}_{\vec{s}}^{\mathrm{T}}\,\Sigma^{-1}\,(\vec{y}^{\,(n)} - \tilde{W}_{\vec{s}}\,\vec{\mu})\end{aligned} \tag{15}$$

and with $\Psi_{\vec{s}} = \Psi\big(\mathrm{diag}(\vec{s})\big)$. The full posterior distribution can thus be written as

$$p(\vec{s},\vec{z}\,|\,\vec{y}^{\,(n)},\Theta) \;\;=\;\; \frac{\mathcal{B}(\vec{s};\vec{\pi})\,\mathcal{N}(\vec{y}^{\,(n)};\tilde{W}_{\vec{s}}\,\vec{\mu},C_{\vec{s}})\,\mathcal{N}(\vec{z};\,\vec{\kappa}_{\vec{s}}^{(n)},\Lambda_{\vec{s}})}{\sum_{\vec{s}'}\mathcal{B}(\vec{s}';\vec{\pi})\,\mathcal{N}(\vec{y}^{\,(n)};\tilde{W}_{\vec{s}'}\,\vec{\mu},C_{\vec{s}'})}. \tag{16}$$

Equation (16) represents the crucial result for the computation of the E-step below because, first, it shows that the posterior does not involve analytically intractable integrals and, second, for fixed $\vec{s}$ and $\vec{y}^{\,(n)}$ the dependency on $\vec{z}$ follows a Gaussian distribution. This special form allows for the derivation of analytical expressions for the expectation values as required for the M-step updates. Because of the Gaussian form, the integrations over the continuous part are straight-forward and the expectation values required for the M-step are given as follows:

$$\big\langle\vec{s}\big\rangle_n \;=\; \sum_{\vec{s}} q_n(\vec{s};\Theta)\,\vec{s}, \tag{17}$$

$$\big\langle\vec{s}\,\vec{s}^{\mathrm{T}}\big\rangle_n \;=\; \sum_{\vec{s}} q_n(\vec{s};\Theta)\,\vec{s}\,\vec{s}^{\mathrm{T}}, \tag{18}$$

$$\big\langle\vec{s}\odot\vec{z}\big\rangle_n \;=\; \sum_{\vec{s}} q_n(\vec{s};\Theta)\,\vec{\kappa}_{\vec{s}}^{(n)}, \tag{19}$$

$$\text{and}\;\; \big\langle(\vec{s}\odot\vec{z})(\vec{s}\odot\vec{z})^{\mathrm{T}}\big\rangle_n \;=\; \sum_{\vec{s}} q_n(\vec{s};\Theta)\,\big(\Lambda_{\vec{s}} + \vec{\kappa}_{\vec{s}}^{(n)}(\vec{\kappa}_{\vec{s}}^{(n)})^{\mathrm{T}}\big). \tag{20}$$

In all of the expressions above, the left-hand-sides are expectation values over the full latent space w.r.t. the posterior $p(\vec{s},\vec{z}\,|\,\vec{y}^{\,(n)},\Theta)$, whereas the right-hand-sides now take the form of expectation values only over the binary part w.r.t. the posterior $p(\vec{s}\,|\,\vec{y}^{(n)},\Theta)$ in Equation (13). The derivations of E-step equations (17) to (20) are a generalization of the derivations by Lücke and Sheikh (2012). While Gaussian identities and marginalization have been used to obtain analytical results for mixture-of-Gaussians priors before (e.g. Moulines et al., 1997; Attias, 1999; Olshausen and Millman, 2000; Garrigues and Olshausen, 2007), the above equations are the first closed-form solutions for the spike-and-slab model (first appearing in Lücke and Sheikh, 2012). The observation that the Gaussian slab and Gaussian noise model allows for analytically tractable integrals has, in parallel work, also been pointed out by Mohamed et al. (2012).

Iteratively computing the E-step equations (17) to (20) using the current parameters Θ and the M-step equations (6) to (10), represents a closed-form and exact EM algorithm which increases the data likelihood of the model to (possibly local) maxima.

4. Truncated EM

While being exact, the execution of the above EM algorithm results in considerable computational costs for larger-scale problems. Without approximations, the computational resources required scale exponentially with the number of hidden dimensions H. This can be seen by considering the expected values w.r.t. the posterior $p(\vec{s}\,|\,\vec{y},\Theta)$ above, which each require a summation over all binary vectors $\vec{s}\in\{0,1\}^H$. For tasks involving low dimensional hidden spaces, the exact algorithm is still applicable. For higher dimensional problems approximations are required, however. Still, we can make use of the closed-form EM solutions by applying an approximation solely to the binary part. Instead of sampling-based or factored approximations to the posterior $p(\vec{s},\vec{z}\,|\,\vec{y},\Theta)$, we use a truncated approximation to the posterior $p(\vec{s}\,|\,\vec{y}^{(n)},\Theta)$ in Equation (13). The truncated approximation is defined to be proportional to the true posteriors on subspaces of the latent space with high probability mass (compare *Expectation Truncation*, Lücke and Eggert, 2010). More concretely, a posterior distribution $p(\vec{s}\,|\,\vec{y}^{(n)},\Theta)$ is approximated by a distribution $q_n(\vec{s};\Theta)$ that only has support on a subset $\mathcal{K}_n\subseteq\{0,1\}^H$ of the state space:

$$q_n(\vec{s};\Theta)=\frac{p(\vec{s},\vec{y}^{(n)}\,|\,\Theta)}{\sum\limits_{\vec{s}\,'\in\mathcal{K}_n}p(\vec{s}\,',\vec{y}^{(n)}\,|\,\Theta)}\,\delta(\vec{s}\in\mathcal{K}_n), \tag{21}$$

where $\delta(\vec{s}\in\mathcal{K}_n)$ is an indicator function, i.e., $\delta(\vec{s}\in\mathcal{K}_n)=1$ if $\vec{s}\in\mathcal{K}_n$ and zero otherwise.

The basic assumption behind the approximation in (21) is that the posterior over the entire hidden space is concentrated in small volumes, which is represented by the reduced support of subset $\mathcal{K}_n$. When using a spike-and-slab sparse coding model to gain a generative understanding of the data, sparsity in the posterior distribution usually emerges naturally. We can see an illustration of this in Figure 2 (generation details in Section 5.4). Figure 2A shows (for three typical data points) how much posterior mass is carried by each of the $H=10$ latent dimensions. Figure 2B shows (for the same data points) histograms of the posterior mass marginalized across the whole range of hyperplanes spanned by the 10–dimensional latent space. Figure 2A indicates that only a subset of the H latents is significantly relevant for encoding the posterior, while Figure 2B allows us to observe that the posterior mass is primarily contained within low-dimensional hyperplanes of the H–

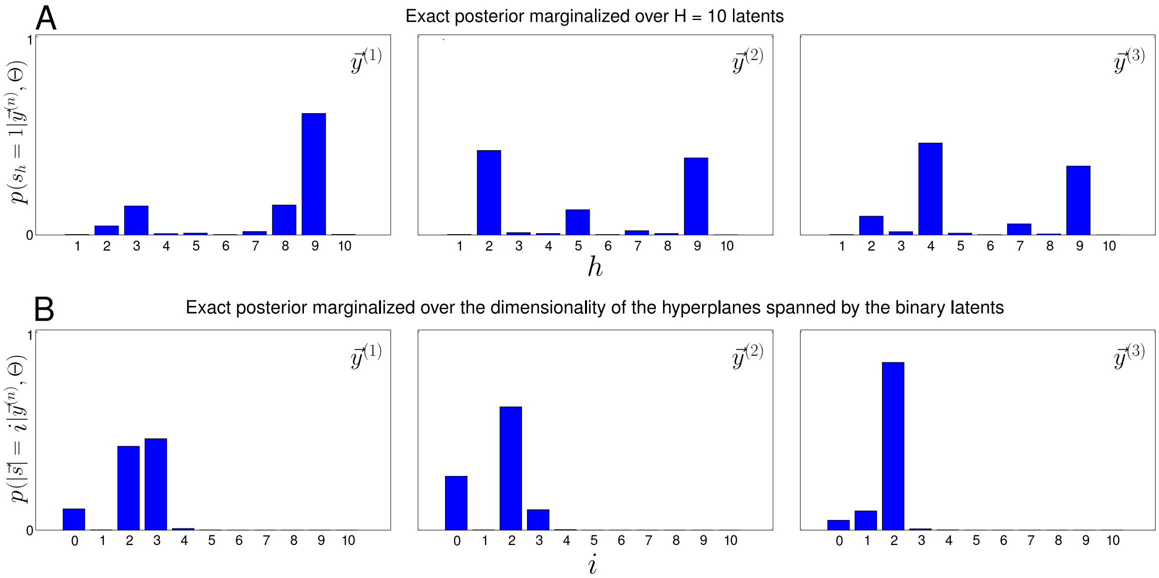

Figure 2: Visualization of the exact posterior probabilities of the spike-and-slab model with $H = 10$ latents, computed for three given data points $\vec{y}^{(n)}$. The model was trained on natural data (see Section 5.4 for more details). **A** Histograms of the posterior mass over the H latents: $p(s_h = 1\,|\,\vec{y}^{(n)}, \Theta) = \sum_{\vec{s}\,\mathrm{with}\, s_h=1} p(\vec{s}\,|\,\vec{y}^{(n)}, \Theta) / \sum_{\vec{s}} p(\vec{s}\,|\,\vec{y}^{(n)}, \Theta)$. Low values for most h imply that these latents can be neglected (i.e., clamped to zero) for a posterior approximation. **B** Histograms of the posterior mass over the hyperplanes of increasing dimensionality i: $p(|\vec{s}| = i\,|\,\vec{y}^{(n)}, \Theta) = \sum_{\vec{s}\,\mathrm{where}\,|\vec{s}|=i} p(\vec{s}\,|\,\vec{y}^{(n)}, \Theta) / \sum_{i'=0}^{H} \sum_{\vec{s}\,\mathrm{where}\,|\vec{s}|=i'} p(\vec{s}\,|\,\vec{y}^{(n)}, \Theta)$. In case of all the three examples presented here, subspaces with $i > 4$ can be neglected as another approximation step for posterior estimation.

dimensional hidden space. In other words, given a data point we find that most of the posterior mass is concentrated in low-dimensional subspaces spanned by $H' \ll H$ of the latent dimensions. The sparse nature of the posterior as illustrated by Figure 2 allows us to apply approximation (21), where we define the subsets $\mathcal{K}_n$ based on index sets $I_n \subseteq \{1, \ldots, H\}$, which contain the indices of H' most relevant sparse latents (compare Figure 2A) for the corresponding data points $\vec{y}^{(n)}$:

$$\mathcal{K}_n = \{\vec{s} \mid \textstyle\sum_h s_h \leq \gamma \text{ and } \forall h \notin I_n : s_h = 0\} \cup \mathcal{U}, \tag{22}$$

where the indices comprising I_n have the highest value of a selection (or scoring) function $\mathcal{S}_h(\vec{y}^{(n)}, \Theta)$ (which we define later). The set $\mathcal{U}$ is defined as $\mathcal{U} = \{\vec{s} \mid \sum_h s_h = 1\}$ and ensures that $\mathcal{K}_n$ contains all singleton states (compare Lücke and Eggert, 2010). Otherwise, $\mathcal{K}_n$ only contains vectors with at most γ non-zero entries and with non-zero entries only permitted for $h \in I_n$. The parameter $\gamma \leqslant H'$ sets the maximal dimensionality of the considered hyper-planes (compare Figure 2B). It was empirically shown by Lücke and Eggert (2010) that for appropriately defined subspaces $\mathcal{K}_n$, the KL-divergence between the true posteriors and their truncated approximations converges to values close to zero.

If we now use the concrete expressions of the sparse spike-and-slab model (Equations (1) to (3)) for the variational distribution in Equation (21), the truncated approximation is given by:

$$p(\vec{s}\,|\,\vec{y}^{(n)},\Theta) \;\approx\; q_n(\vec{s};\Theta) \;\;=\;\; \frac{\mathcal{N}(\vec{y}^{(n)};\tilde{W}_{\vec{s}}\,\vec{\mu},C_{\vec{s}})\,\mathcal{B}(\vec{s};\vec{\pi})}{\sum_{\vec{s}\,'\in\mathcal{K}_n}\mathcal{N}(\vec{y}^{(n)};\tilde{W}_{\vec{s}\,'}\vec{\mu},C_{\vec{s}\,'})\,\mathcal{B}(\vec{s}\,';\vec{\pi})}\,\delta(\vec{s}\in\mathcal{K}_n). \tag{23}$$

The approximation can now be used to compute the expectation values which are required for the M-step equations. If we use the variational distributions in Equation (23) for $q_n(\vec{s};\Theta)$ on the right-hand-sides of Equations (17) to (20), we obtain:

$$\sum_{\vec{s}} q_n(\vec{s};\Theta)\,f(\vec{s}) \;\;=\;\; \frac{\sum_{\vec{s}\in\mathcal{K}_n}\mathcal{N}(\vec{y}^{(n)};\tilde{W}_{\vec{s}}\,\vec{\mu},C_{\vec{s}})\,\mathcal{B}(\vec{s};\vec{\pi})\,f(\vec{s})}{\sum_{\vec{s}\,'\in\mathcal{K}_n}\mathcal{N}(\vec{y}^{(n)};\tilde{W}_{\vec{s}\,'}\vec{\mu},C_{\vec{s}\,'})\,\mathcal{B}(\vec{s}\,';\vec{\pi})}, \tag{24}$$

where $f(\vec{s})$ denotes any of the (possibly parameter dependent) functions of (17) to (20). Instead of having to compute sums over the entire binary state space with 2^H states, only sums over subsets $\mathcal{K}_n$ have to be computed. Since for many applications the posterior mass is finally concentrated in small volumes of the state space, the approximation quality can stay high even for relatively small sets $\mathcal{K}_n$.

Note that the definition of $q_n(\vec{s};\Theta)$ in Equation (21) neither assumes uni-modality like MAP approximations (Mairal et al., 2009; Lee et al., 2007; Olshausen and Field, 1997) or Gaussian approximations of the posterior (Ribeiro and Opper, 2011; Seeger, 2008), nor does it assume a-posteriori independence of the latents as factored approximations (Jordan et al., 1999; Goodfellow et al., 2013; Titsias and Lazaro-Gredilla, 2011). The approximation scheme we have introduced here exploits the inherent property of the sparse spike-and-slab model to have posterior probabilities concentrated in low-dimensional subspaces. The quality of our approximated posterior $q_n(\vec{s};\Theta)$ primarily depends on an appropriate selection of the relevant subspaces $\mathcal{K}_n$ (see Section 4.2 below).

The truncated approximation is similar to factored variational approximations or MAP approximations in the sense that it can be formulated as an approximate distribution $q_n(\vec{s};\Theta)$ within the free-energy formulation by Neal and Hinton (1998). Within this formulation, $q_n(\vec{s};\Theta)$ is often referred to as *variational* approximation, and we therefore refer to our approximation as *truncated variational EM*. Like factored variational approaches, we here aim to minimize the KL-divergence between the true posterior and the approximation in Equation (21). However, we do not use variational parameters and a gradient based optimization of such parameters for the minimization. Our approach is therefore not a variational approach in the sense of classical variational calculus.

4.1 Computational Complexity

The truncated E-step defined by (17) to (20) with (24) scales with the approximation parameters γ and H' which can be defined independently of the latent dimensionality H. The complexity scales as $\mathcal{O}\big(N\sum_{\gamma'=0}^{\gamma}\binom{H'}{\gamma'}(D+\gamma')^3\big)$, where the D^3 term can be dropped from the cubic expansion if the observed noise Σ is considered to be diagonal or homoscedastic.

Also the truncated approximation yields sparse matrices in Equations (21) and (23) which results in more efficient and tractable matrix operations.

Although the total number of data points N above defines a theoretical upper bound, in practice we can further benefit from the preselection step of the truncated approach to achieve significantly improved runtime performances. Clustering the data points using the index sets I_n saves us from redundantly performing various computationally expensive operations involved in Equations (15) and (23), that given a state $\vec{s} \in \mathcal{K}_n$ are independent of individual data points sharing the same subspace $\mathcal{K}_n$. Furthermore, such a batch processing strategy is also readily parallelizable as the truncated E-step can be performed independently for individual data clusters (see Appendix C for details). Using the batch execution mode we have observed an average runtime speedup of up to an order of magnitude.

4.2 Selection Function

To compose appropriate subspaces $\mathcal{K}_n$ a selection function $\mathcal{S}_h(\vec{y}^{\,(n)}, \Theta)$ is defined, which prior to each E-step allows us to select the relevant H' hidden dimensions (i.e., the elements of the index sets I_n) for a given observation $\vec{y}^{\,(n)}$. A selection function is essentially a heuristic-based scoring mechanism, that ranks all the latents based on their potential for being among the generating causes of a given observation. Selection functions can be based on upper bounds for probabilities $p(s_h = 1 \,|\, \vec{y}^{\,(n)}, \Theta)$ (compare Lücke and Eggert, 2010; Puertas et al., 2010) or deterministic functions such as the scalar product between a basis vector and a data point (derived from noiseless limits applied to observed space; compare Lücke and Eggert, 2010; Bornschein et al., 2013).

For the sparse coding model under consideration we define a selection function as follows:

$$\mathcal{S}_h(\vec{y}^{\,(n)}, \Theta) = \mathcal{N}(\vec{y}^{\,(n)}; \tilde{W}_{\vec{s}_h}\vec{\mu}, C_{\vec{s}_h}) \propto p(\vec{y}^{\,(n)} \,|\, \vec{s} = \vec{s}_h, \Theta), \tag{25}$$

where $\vec{s}_h$ represents a singleton state in which only the entry h is non-zero. The selection function (25) is basically the data likelihood given a singleton state $\vec{s}_h$. The function does not take into account the probability of the state itself (i.e., $p(\vec{s}_h \,|\, \Theta)$), as this may introduce a bias against less active latent dimensions. Similar to previously used selection functions (compare e.g., Lücke and Eggert, 2010; Puertas et al., 2010), in order to maintain a linear scaling behavior w.r.t. the number of latents, the selection function introduced here avoids computationally demanding higher-order combinatorics of the latents by only assessing one-to-one correspondences between individual latents and an observed data point. In the next section we empirically evaluate the efficacy of our selection function by means of numerical experiments that are based on the KL-divergence between the exact and the approximated posteriors computed from the subspaces $\mathcal{K}_n$.

Equations (21) to (23) replace the computation of the expectation values w.r.t. the exact posterior, and represent the approximate EM algorithm used in the experiments section. The algorithm will be applied without any further mechanisms such as annealing as we found it to be very robust in the form derived above. Furthermore, no data preprocessing such as mean subtraction or variance normalization will be used in any of the experiments. To distinguish the algorithm from others in comparative experiments, we will refer to it as *Gaussian Sparse Coding* (GSC) algorithm in order to emphasize the special Gaussian case of the spike-and-slab model used.

5. Numerical Experiments

We investigate the performance of the GSC algorithm on artificial data as well as various realistic source separation and denoising benchmarks. For all experiments the algorithm was implemented to run in parallel on multiple CPUs with no dependency on their arrangement as physically collocated arrays with shared memory or distributed among multiple compute nodes (see Bornschein et al., 2010, for more details). We further extended the basic technique to make our implementation more efficient and suitable for parallelization by applying the batch execution (the observation discussed in Section 4.1 on Computational Complexity and Appendix C). In all the experiments, the GSC model parameters were randomly initialized.[2] The choice of GSC truncation parameters H' and γ is in general straight-forward: the larger they are the closer the match to exact EM but the higher are also the computational costs. The truncation parameters are therefore capped by the available computational resources. However, empirically we observed that often much smaller values were sufficient than those that are maximally affordable.[3] Note that factored variational approaches do not usually offer such a trade-off between the exactness and computational demand of their inference schemes by means of a simple parameter adjustment.

5.1 Reliability of the Selection Function

To assess the reliability of the selection function we perform a number of experiments on small scale artificial data generated by the model, such that we can compute both the exact (13) and truncated (23) posteriors. To control for the quality of the truncated posterior approximation—and thus the selection function—we compute the ratio between posterior mass within the truncated space $\mathcal{K}_n$ and the overall posterior mass (compare Lücke and Eggert, 2010):

$$Q^{(n)} = \frac{\sum_{\vec{s}\in\mathcal{K}_n} \int_{\vec{z}} p(\vec{s},\vec{z}\,|\,\vec{y}^{(n)},\Theta)\,\mathrm{d}\vec{z}}{\sum_{\vec{s}\,'} \int_{\vec{z}\,'} p(\vec{s}\,',\vec{z}\,'\,|\,\vec{y}^{(n)},\Theta)\,\mathrm{d}\vec{z}\,'} = \frac{\sum_{\vec{s}\in\mathcal{K}_n} \mathcal{B}(\vec{s};\vec{\pi})\,\mathcal{N}(\vec{y}^{(n)};\tilde{W}_{\vec{s}}\,\vec{\mu},C_{\vec{s}})}{\sum_{\vec{s}\,'} \mathcal{B}(\vec{s}\,';\vec{\pi})\,\mathcal{N}(\vec{y}^{(n)};\tilde{W}_{\vec{s}\,'}\vec{\mu},C_{\vec{s}\,'})}, \tag{26}$$

where the integrals over the latent $\vec{z}$ in (26) are again given in closed-form. The metric $Q^{(n)}$ ranges from zero to one and is directly related to the KL-divergence between the approximation q_n in Equation (23) and the true posterior:

$$D_{KL}(q_n(\vec{s},\vec{z};\,\Theta),p(\vec{s},\vec{z}\,|\,\vec{y}^{(n)},\Theta)) = -\log(Q^{(n)})\,.$$

If $Q^{(n)}$ is close to one, the KL-divergence is close to zero.

Data for the control experiments were generated by linearly superimposing basis functions that take the form of horizontal and vertical bars (see e.g., Földiák, 1990; Hoyer, 2002) on a $D = D_2 \times D_2$ pixel grid, where $D_2 = H/2$. This gives us D_2 possible horizontal as well as vertical locations for bars of length D_2, which together form our generating bases W^{gen}. Each bar is then randomly assigned either a positive or negative value with

[2] We randomly and uniformly initialized the π_h between 0.05 and 0.95. $\vec{\mu}$ was initialized with normally distributed random values and the diagonal of Ψ was initialized with strictly positive uniformly distributed random values. We set Σ to the covariance across the data points, and the elements of W were independently drawn from a normal distribution with zero mean and unit variance.

[3] Compare Appendix B for trade-off between complexity and accuracy of the truncated EM approach.

magnitude 10. We set the sparsity such that there are on average two active bars per data point, i.e., $\pi_h^{\text{gen}} = 2/H$ for all $h \in H$. We assume homoscedastic[4] observed noise $\Sigma^{\text{gen}} = \sigma^2 I_D$, where $\sigma^2 = 2.0$. The mean of the generating slab is i.i.d. drawn from a Gaussian: $\vec{\mu}^{\text{gen}} \sim \mathcal{N}(0, 5)$, and the covariance of the slab is $\Psi^{\text{gen}} = I_H$. We generate $N = 1000$ data points. We run experiments with different sets of values for the truncation parameters $(H', \gamma) \in \{(4, 4), (5, 4), (5, 3)\}$ for each $H \in \{10, 12\}$. Each run consists of 50 EM iterations and after each run we compute the Q-value over all the data points. For all the experiments we find the average Q-values to be above 0.99, which shows that the state subspaces (22) constructed from the H' latents chosen through the selection function (25) contain almost the entire posterior probability mass in this case. The small fraction of remaining posterior mass lies in other discrete subspaces and its principle form is known to not contain any heavy tails (see Equation (16)). The contribution of the truncated posterior mass to parameter updates can therefore be considered negligible.

5.2 Consistency

Prior to delving into a comparative analysis of GSC with other methods, we assess the consistency of the approach by applying both its exact and truncated variational inference schemes on the task of recovering sparse latent directions w.r.t. increasing numbers of training data. For this experiment we work with synthetic data generated by the GSC model itself. Moreover, we also apply the truncated variational inference on standard sparse coding data generated with a standard Laplace prior(Olshausen and Field, 1996; Lee et al., 2007). Taking into account the computational demands of the exact inference, we set both the hidden as well as observed dimensions (H and D respectively) to 10. For the experiment we exponentially increase N from 1000 to 512000. For each trial in the experiment we generate a new ground-truth mixing matrix $W^{\text{gen}} \in \mathbb{R}^{D \times H}$ by randomly generating a set of H orthogonal bases and perturbing them with a Gaussian noise with zero mean and a variance of 2.0. We set the sparsity parameters π_h to $1/H$, while the observed noise is assumed to be homoscedastic with $\sigma = 1.0$. When generating data with a spike-and-slab prior, the slab is considered to have its mean at zero with an identity covariance matrix, i.e., $\mu_h = 0.0$ for all $h \in H$ and $\Psi^{\text{gen}} = I_H$, respectively. In each trial after performing 100 EM iterations and inferring the whole set of GSC parameters Θ, we quantify the quality of the inference in terms of how well the inferred bases W align with the corresponding ground truth bases W^{gen}. As a measure of discrepancy between the generating and the recovered bases we use the Amari index (Amari et al., 1995):

$$A(W) \;=\; \frac{1}{2H(H-1)} \sum_{h,h'=1}^{H} \Big(\frac{|O_{hh'}|}{\max_{h''} |O_{hh''}|} + \frac{|O_{hh'}|}{\max_{h''} |O_{h''h'}|} \Big) - \frac{1}{H-1}, \qquad (27)$$

where $O_{hh'} = \big(W^{-1} W^{\text{gen}} \big)_{hh'}$. The Amari index is either positive or zero. It is zero only when the basis vectors of W and W^{gen} represent the same set of orientations, which in our case implies a precise recovery of the (ground truth) sparse directions.

[4] To infer homoscedastic noise we set in the M-step the updated noise matrix Σ to $\sigma^2 I_D$ where $\sigma^2 = \mathrm{Tr}\big(\Sigma\big)/D$. This is equivalent to parameter update for σ^2 if the model originally assumes homoscedastic noise.

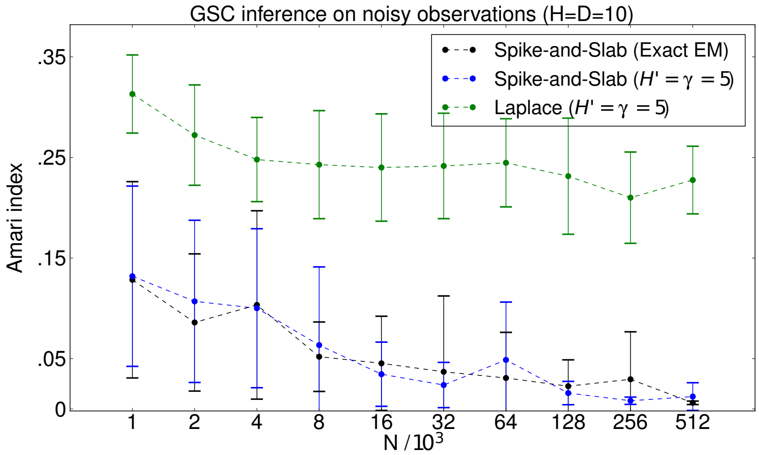

Figure 3: Numerical experiment investigating the consistency of the exact as well as the truncated variational GSC algorithm for increasing numbers of data points. The curves show results for the recovery of sparse directions for different numbers of data points. Data points were generated by both the spike-and-slab generative model (black and blue) and a standard sparse coding model with Laplace prior (green). The curves show the mean Amari index and standard deviations computed based on 15 repetitions of the learning algorithm.

Figure 3 summarizes the results of the experiment. Each error bar in the plot extends one standard deviation on both sides of its corresponding mean Amari index, which is computed from 15 repetitions. The black curve shows the results of the exact GSC inference on spike-and-slab generated data, while the blue and green curves illustrate the results of the truncated variational inference ($H' = \gamma = 5$) on data generated by spike-and-slab and Laplace priors respectively. For data generated with the spike-and-slab prior, we observe a gradually more accurate recovery of the sparse directions, as the mean Amari indices gradually converge towards the minimum value of zero for increasing numbers of training data. The minimum Amari index values that we obtain for the black and blue curves for $N \in \{128K, 256K, 512K\}$ are all below 6×10^{-3}. For the standard sparse coding data, we also see an improvement in performance with more data; however, higher mean values of the Amari index in this case can presumably be attributed to the model mismatch.

5.3 Recovery of Sparse Directions on Synthetic Data

In our first comparison with other methods, we measure the performances of GSC (using the truncated variational approximation) and MTMKL (which uses a factored variational approximation) approaches on the sparse latent direction recovery task given synthetic data generated by standard sparse coding models. In one set of experiments we generate data using sparse coding with Cauchy prior (Olshausen and Field, 1996), and in another set of experiments we use the standard Laplace distribution as a prior (Olshausen and Field, 1996; Lee et al., 2007). For each trial in the experiments a new mixing matrix

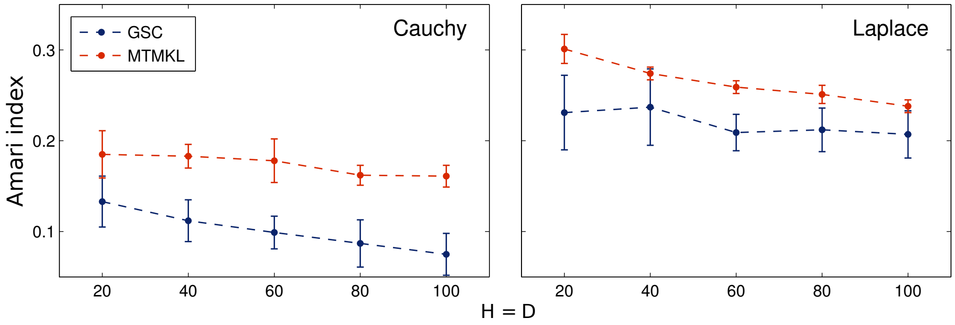

Figure 4: Performance of GSC (with $H' = \gamma = \frac{H}{10}$) vs. MTMKL on data generated by standard sparse coding models both with Cauchy and Laplace priors. Performance compared on the Amari index (27).

W^{gen} was generated without any constraints on the sparse directions (i.e., matrices were non-orthogonal in general). In both sets of experiments we simultaneously vary both the observed and latent dimensions D and H between 20 and 100, and repeat 15 trials per given dimensionality. For each trial we randomly generated a new data set of $N = 5000$ noisy observations with $\Sigma^{\text{gen}} = I_D$. Per trial, we perform 50 iterations of both algorithms. The GSC truncation parameters H' and γ were set to $\frac{H}{10}$. We assess the performances of the algorithms w.r.t. the Amari index (27).

The results for GSC and MTMKL in Figure 4 show that both approaches do relatively well in recovering the sparse directions, which shows that they are robust against the model mismatch imposed by generating from models with other priors. Furthermore, we observe that the GSC approach consistently recovers the sparse directions more accurately.

5.4 Source Separation

On synthetic data we have seen that spike-and-slab sparse coding can effectively recover sparse directions such as those generated by standard sparse coding models. As many signals such as acoustic speech data are sparse, and as different sources mix linearly, the assumptions of sparse coding match such data well. Source separation is consequently a natural application domain of sparse coding approaches, and well suited for benchmarking novel spike-and-slab as well as other sparse coding algorithms. To systematically study the a-posteriori independence assumption in factored variational approaches, we monitor the recovery of sparse directions of GSC and MTMKL for an increasing degree of the mixing matrix's non-orthogonality. Figure 5 shows the performance of both the methods based on three different source separation benchmarks obtained from (ICALAB; Cichocki et al., 2007). The error bars show two standard deviations estimated based on 15 trials per experiment. The x-axis in the figure represents the degree of orthogonality of the ground truth mixing bases W^{gen}. Starting from strictly orthogonal at the left, the bases were made increasingly non-orthogonal by randomly generating orthogonal bases and adding Gaussian distributed noise to them with $\sigma \in \{4, 10, 20\}$, respectively. For Figure 5 no observation

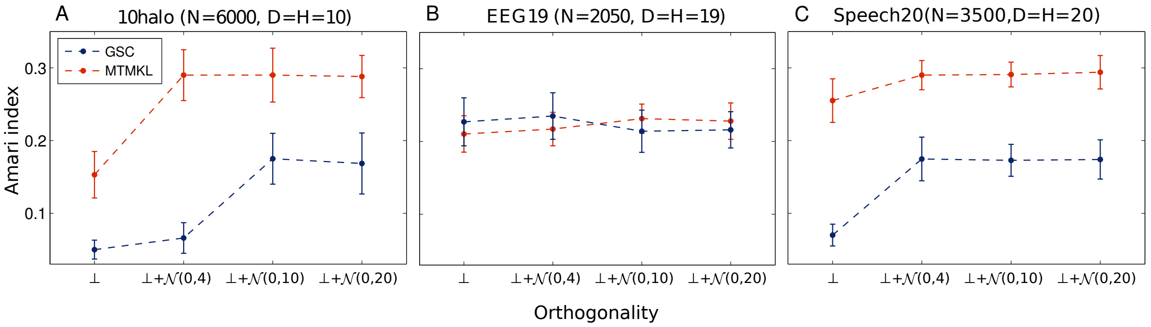

Figure 5: Performance of GSC vs. MTMKL on source separation benchmarks with varying degrees of orthogonality of the mixing bases. The orthogonality on the x-axis varies from being orthogonal $\perp$ to increasingly non-orthogonal mixing as randomly generated orthogonal bases are perturbed by adding Gaussian noise $\mathcal{N}(0, \sigma)$ to them. No observation noise was assumed for these experiments. Performances are compared on the Amari index (27).

noise was added to the mixed sources. For both the algorithms we performed 100 iterations per run.[5] The GSC truncation parameters H' and γ were set to 10 for all the following experiments, therefore for *10halo* the GSC inference was exact. As can be observed, both approaches recover the sparse directions well. While performance on the EEG19 data set is the same, GSC consistently performs better than MTMKL on *10halo* and *Speech20*. If observation noise is added, the difference can become still more pronounced for some data sets. Figure 6 shows the performance in the case of *Speech*20 (with added Gaussian noise with $\sigma = 2.0$), for instance. Along the x-axis orthogonality decreases, again. While the performance of MTMKL decreases with decreasing orthogonality, performance of GSC increases in this case. For other data sets increased observation noise may not have such effects, however (see Appendix, Figure 12 for two examples).

Next we look at MAP based sparse coding algorithms for the source separation task. Publicly available methods which we compare with are (SPAMS; Mairal et al., 2009) and the efficient sparse coding algorithms (ESCA; Lee et al., 2007). These methods are based on linear regression with lasso regularization, where sparsity is induced by introducing a parameter-regulated penalty term in the objective function,[6] which penalizes the L_1-norm of regressors (or latent variables). In a probabilistic context this is equivalent to assuming a Laplace prior on the regressors. In this experiment we test the performance on another set of ICALAB (Cichocki et al., 2007) benchmarks used previously (Suzuki and Sugiyama, 2011; Lücke and Sheikh, 2012). Following Suzuki and Sugiyama (2011) we use $N = 200$ and $N = 500$ data points from each benchmark and generate observed data by mixing the

[5] For the MTMKL algorithm we observed convergence after 100 iterations while the GSC algorithm continued to improve with more iterations. However, allowing the same number of iterations to both the algorithms, the reported results are obtained with 100 iterations.

[6] For both the algorithms compared here, optimal values for sparsity controlling regularization parameters were chosen through cross-validation.

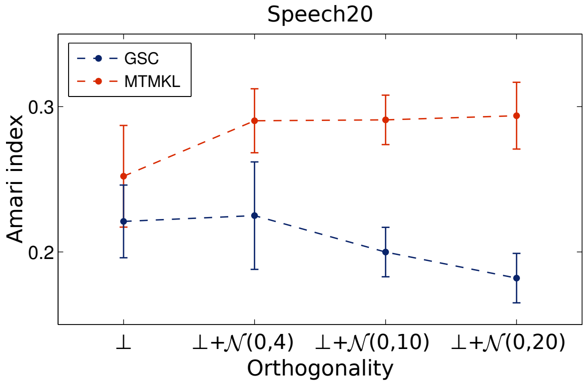

Figure 6: Performance of GSC vs. MTMKL in terms of the Amari index (27) on the Speech20 benchmark with varying degrees of orthogonality of the mixing bases and Gaussian noise (with $\sigma = 2$) added to observed data. The orthogonality on the x-axis varies from being orthogonal $\perp$ to increasingly non-orthogonal mixing as randomly generated orthogonal bases are perturbed by adding Gaussian noise $\mathcal{N}(0, \sigma)$ to them.

benchmark sources with randomly generated orthogonal bases and adding no noise to the observed data. For each experiment we performed 50 trials with a new randomly generated orthogonal data mixing matrix W^{gen} and new parameter initialization in each trial. The GSC inference was exact for these experiments with better results obtained with observed noise constrained to be homoscedastic. We performed up to 350 iterations of the GSC algorithm (with more iterations continuing to improve the performance) while for the other algorithms we observed convergence between 100 and 300 iterations.

Table 1 lists the performances of the algorithms. As can be observed, the spike-and-slab based models perform better than the standard sparse coding models for all except of one experiment (Sergio7, 200 data points) where SPAMS performs comparably well (or slightly better). Among the spike-and-slab models, GSC performs best for all settings with 500 data points, while MTMKL is better in two cases for 200 data points.[7] Further improvements on some settings in Table 1 can be obtained by algorithms constrained to assume orthogonal bases (Suzuki and Sugiyama, 2011; Lücke and Sheikh, 2012). However, for *10halo* and *speech4* GSC and MTMKL are better without such an explicit constraint.

Figure 2 was generated in a similar fashion on the *10halo* data set. There we computed the exact posterior (13) over $H = 10$ latent dimensions, thus the approximation parameters were $\gamma = H' = H$ (exact E-step). After performing 50 EM iterations and learning all the model parameters, we then visualized marginalized posteriors for a given data point along each column of the figure. The top row of the figure allows us get an idea of how

[7] In Table 1 the results do not necessarily improve with an increased number of data points. However, the data points considered here are not independent samples. Following Suzuki and Sugiyama (2011) we always took consecutive 200 or 500 data points (after an offset) from each of the benchmarks. Therefore, due to time-dependencies in the signals, the underlying data point statistics change with the number of data points.

data sets			Amari index - mean (std.)			
name	H = D	N	GSC	MTMKL	SPAMS	ESCA
10halo	10	200	0.27(.04)	**0.21(.05)**	0.28(0)	0.31(.02)
		500	**0.17(.03)**	0.20(.03)	0.29(0)	0.29(.02)
Sergio7	7	200	**0.19(.05)**	**0.19(.03)**	**0.18(0)**	0.27(.04)
		500	**0.13(.04)**	0.23(.04)	0.19(0)	0.18(.04)
Speech4	4	200	**0.13(.04)**	**0.14(.03)**	0.18(0)	0.23(.02)
		500	**0.10(.04)**	0.14(.08)	0.16(0)	0.17(0)
c5signals	5	200	0.29(.08)	**0.24(.08)**	0.39(0)	0.47(.05)
		500	**0.31(.06)**	**0.32(.03)**	0.42(0)	0.48(.05)

Table 1: Performance of GSC, MTMKL and other publicly available sparse coding algorithms on benchmarks for source separation. Performances are compared based on the Amari index (27). Bold values highlight the best performing algorithm(s).

concentrated and sparse a data point is in terms of the latents contributing to its posterior mass. The bottom row of the figure on the other hand allows us to observe the sparsity in the posterior w.r.t. the dimensionality of the hyperplanes spanned by the latents, with a posterior mass accumulation in low-dimensional hyperplanes.

5.5 Computational Complexity vs. Performance

In terms of computational complexity, GSC and MTMKL algorithms are significantly different, so we also looked at the trade-off between their computational costs versus performance. Subfigures A and B in Figure 7 show performance against compute time for both algorithms. The error bars for the Speech20 plot were generated from 15 trials per experiment. For MTMKL we obtained the plot by increasing the number of iterations from 50 to 100 and 1000, while for the GSC plot we performed 100 iterations with $H' = \gamma \in [2, 3, 5, 7, 10]$. For the image denoising task (described next), the MTMKL plot was generated from a run with $H = 64$ latents and the number of iterations going up to 12K. The GSC plot was generated from $H = 400$ latents with H' and γ being 10 and 5 respectively. The last point on the GSC (blue) curve corresponds to the $120th$ EM iteration. As can be observed for both tasks, the performance of MTMKL saturates from certain runtime values onwards. GSC on the other hand continues to show improved performance with increasing computational resources.

For the denoising task we also compared the performance of both the algorithms against an increasing number of latents H. While the computational cost of the MTMKL algorithm increases linearly w.r.t. H, the runtime cost of the truncated variational GSC remains virtually unaffected by it, since it scales w.r.t. the parameters H' and γ (see Section 4.1). In this experiment we performed 65 iterations of the GSC algorithm for $H \in \{64, 256\}$ and up to 120 iterations for $H = 400$. For MTMKL we performed up to 120 iterations for each given H. Figure 7C summarizes the results of this experiment. In the figure we can see a constant performance increase for GSC, while for MTMKL we actually observe a slight

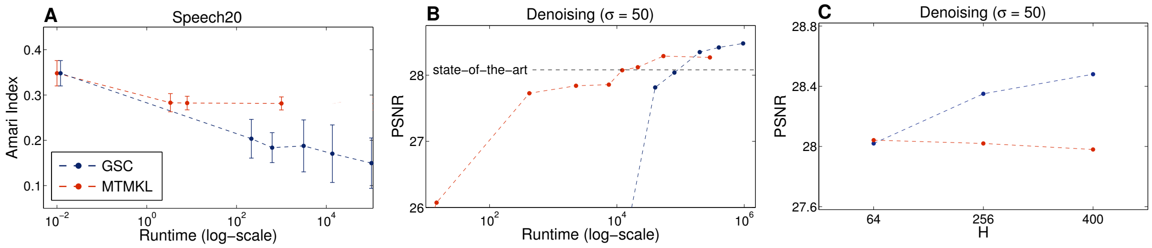

Figure 7: **A,B:** Runtime vs. performance comparison of GSC (blue) and MTMKL (red) on source separation and denoising tasks. Source separation is compared on the Amari index (the lower the better) while the denoising is compared on the peak signal-to-noise (PSNR) ratio (the higher the better). **C**: Performance of GSC (blue) and MTMKL (red) on the denoising task against an increasing number of latents.

decrease in performance. This is in conformity with what Titsias and Lazaro-Gredilla (2011) report in their work that for the denoising task they observed no performance improvements for larger number of latents.

5.6 Image Denoising

Finally, we investigate performance of the GSC algorithm on the standard "house" benchmark for denoising which has been used for the evaluation of similar approaches (e.g., Li and Liu, 2009; Zhou et al., 2009) including the MTMKL spike-and-slab approach. The MTMKL approach currently represents the state-of-the-art on this benchmark (see Titsias and Lazaro-Gredilla, 2011). We also compare with the approach by Zhou et al. (2009) as a representative sampling-based optimization scheme. For the task a noisy input image is generated by adding Gaussian noise (with zero mean and standard deviation determining the noise level) to the 256×256 image (see Figure 8). Following the previous studies, we generated $62,001$ overlapping (shifted by 1 pixel) 8×8 patches from the noisy image. We then applied 65 iterations of the GSC algorithm for $H \in \{64, 256\}$ for different noise levels $\sigma \in \{15, 25, 50\}$. The truncation parameters H' and γ for each run are listed in Table 2. We assumed homoscedastic observed noise with a priori unknown variance in all these experiments (as the MTMKL model).

A comprehensive comparison of the denoising results of the various algorithms is shown in Table 2, where performance is measured in terms of the peak signal-to-noise (PSNR) ratio. We found that for the low noise level ($\sigma = 15$) GSC is competitive with other approaches but with MTMKL performing slightly better. For the higher noise levels of $\sigma = 25$ and $\sigma = 50$, GSC outperforms all the other approaches including the MTMKL approach that represented the state-of-the-art. In Figure 8 we show our result for noise level $\sigma = 25$. The figure contains both the noisy and the GSC denoised image along with the inferred sparsity vector $\vec{\pi}$ and all bases with appearance probabilities significantly larger than zero (sorted from high such probabilities to low ones). We also applied GSC with higher numbers of latent dimensions: Although for low noise levels of $\sigma = 15$ and $\sigma = 25$ we did not measure significant improvements, we observed a further increase for $\sigma = 50$. For instance, with $H = 400$, $H' = 10$ and $\gamma = 8$, we obtained for $\sigma = 50$ a PSNR of 28.48dB.

As for source separation described in Section 5.5, we also compared performance vs. computational demand of both algorithms for the task of image denoising. As illustrated in A and B of Figure 7, MTMKL performs better when computational resources are relatively limited. However, when increasingly more computational resources are made available, MTMKL does not improve much further on its performance while GSC performance continuously increases and eventually outperforms MTMKL on this task.

Noise	PSNR (dB)						
	Noisy img	MTMKL$^{exp.}$	K-SVD$^{mis.}$	*K-SVD^{match}	Beta pr.	GSC (H=64)	GSC (H=256)
σ=15	24.59	**34.29**	30.67	34.22	34.19	32.68 (H'=10,γ=8)	33.78 (H'=18,γ=3)
σ=25	20.22	31.88	31.52	32.08	31.89	31.10 (H'=10,γ=8)	**32.01** (H'=18,γ=3)
σ=50	14.59	28.08	19.60	27.07	27.85	28.02 (H'=10,γ=8)	**28.35** (H'=10,γ=8)

Table 2: Comparison of the GSC algorithm with other methods applied to the "house" benchmark. The compared methods are: MTMKL (Titsias and Lazaro-Gredilla, 2011), K-SVD (Li and Liu, 2009), and Beta process (Zhou et al., 2009). Bold values highlight the best performing algorithm(s). *High values for K-SVD matched are not made bold-faced as the method assumes the noise variance to be known a-priori (see Li and Liu, 2009).

6. Discussion

The last years have seen a surge in the application of sparse coding algorithms to different research domains, along with developments of new sparse coding approaches with increased capabilities. There are currently different lines of research followed for developing new algorithms: one direction is based on the standard sparse coding algorithm (Olshausen and Field, 1996) with Laplace prior and parameter optimization using maximum a-posteriori (MAP) estimates of the latent posteriors for efficient training. This original approach has since been made more efficient and precise. Many sparse coding algorithms based on the MAP estimation are continuously being developed and are successfully applied in a variety of settings (e.g., Lee et al., 2007; Mairal et al., 2009). Another line of research aims at a fully Bayesian description of sparse coding and emphasizes greater flexibility by using different (possibly non-Gaussian) noise models and estimations of the number of hidden dimensions. The great challenge of these general models is the procedure of parameter estimation. For instance, the model by Mohamed et al. (2012) uses Bayesian methodology involving conjugate priors and hyper-parameters in combination with Laplace approximation and different sampling schemes.

A line of research falling in between conventional and fully Bayesian approaches is represented by the truncated variational approach studied here and by other very recent develop-

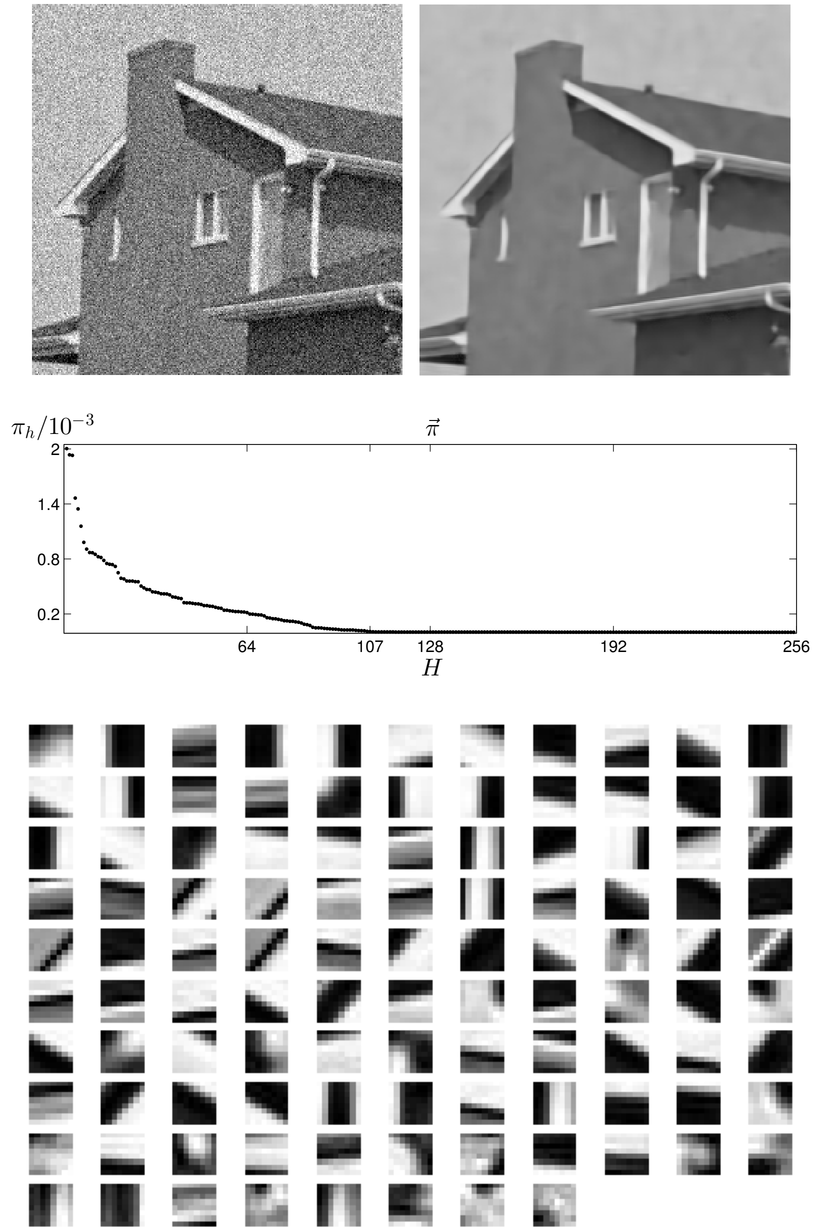

Figure 8: Top left: Noisy "house" image with $\sigma = 25$. Top right: GSC denoised image. Middle: Inferred sparsity values π_h in descending order indicate that finally around 107 of in total 256 latent dimensions significantly contribute to model the data. Bottom: Basis functions (ordered from left to right, top to bottom) corresponding to the first 107 latent dimensions sorted w.r.t. the decreasing sparsity values π_h .

ments (Titsias and Lazaro-Gredilla, 2011; Goodfellow et al., 2013). While these approaches are all based on spike-and-slab generalizations of sparse coding (like fully Bayesian approaches), they maintain deterministic approximation procedures for parameter optimization. Variational approximations allow for applications to large hidden spaces which pose a challenge for sampling approaches especially in cases of multi-modal posteriors. Using the novel and existing approaches in different experiments of this study, we have confirmed the advantages of spike-and-slab priors for sparse coding, and the scalability of variational approximations for such models. The newly developed truncated variational algorithm scales almost linearly with the number of hidden dimensions for fixed truncation parameters (see for instance the scaling behavior in supplemental Figure 10 for H going up to 1024). The MTMKL algorithm by Titsias and Lazaro-Gredilla (2011) has been applied on the same scale. Using a similar approach also based on factored distributions Goodfellow et al. (2013) report results for up to a couple of thousands latent dimensions (albeit on small input dimensions and having a more constrained generative model). Sampling based algorithms for non-parametric and fully Bayesian approaches are more general but have not been applied to such large scales.

A main focus of this work and reasoning behind the algorithm's development is due to the long-known biases introduced by factored variational approximations (MacKay, 2001; Ilin and Valpola, 2005; Turner and Sahani, 2011). Our systematic comparison of the GSC algorithm to the method by Titsias and Lazaro-Gredilla (2011) confirms the earlier observation (Ilin and Valpola, 2005) that factored variational approaches are biased towards orthogonal bases. If we compare the performance of both algorithms on the recovery of non-orthogonal sparse directions, the performance of the factored variational approach is consistently lower than the performance of the truncated variational algorithm (Figure 4). The same applies for experiments for unmixing real signals in which we increased the non-orthogonality (Figure 5A,C; suppl. Figure 12); although for some data performance is very similar (Figure 5B). Also if sources are mixed orthogonally, we usually observe better performance of the truncated variational approach (Table 1), which is presumably due to the more general underlying prior (i.e., a fully parameterized Gaussian slab). Overall, GSC is the best performing algorithm on source separation tasks involving non-orthogonal sparse directions (compare Suzuki and Sugiyama, 2011, for algorithms constrained to orthogonal bases). For some data sets with few data points, we observed an equal or better performance of the MTMKL approach, which can be explained by their Bayesian treatment of the model parameters (see Table 1, performance with 200 data points). Notably, both approaches are consistently better on source separation benchmarks than the standard sparse coding approaches SPAMS (Mairal et al., 2009) and ESCA (Lee et al., 2007) (see Table 1). This may be taken as evidence for the better suitability of a spike-and-slab prior for such types of data.

For source separation our approach (like conventional sparse coding or ICA) seeks to infer sparse directions by capturing the sparse, latent structures from the spatial domain of the input signals. However, when dealing with data that also carry a temporal structure (e.g., speech or EEG recordings), other approaches which explicitly model temporal regularities such as Hidden Markov Models (HMMs) may as well be a more natural and (depending on the task) a more suitable choice. Such methodologies can in principle be combined with the sparse coding approaches studied and compared here to form more comprehensive models for spatio-temporal data, which can yield improved performance on blind source separation tasks (compare e.g., Gael et al., 2008; Mysore et al., 2010).

In the last experiment of this study, we finally compared the performance of factored and truncated variational approximations on a standard image denoising task (see Table 2). The high PSNR values observed for both approaches again in general speak for the strengths of spike-and-slab sparse coding. The MTMKL model represented the state-of-the-art on this benchmark, so far. Differences of MTMKL to previous approaches are small, but this is due to the nature of such long-standing benchmarks (compare, e.g., the MNIST data set). For the same denoising task with standard noise levels of $\sigma = 25$ and $\sigma = 50$ we found the GSC model to further improve the state-of-the-art (compare Table 2 with data by Li and Liu, 2009, Zhou et al., 2009, Titsias and Lazaro-Gredilla, 2011). While we observed a continuous increase of performance with the number of hidden dimensions used for GSC, the MTMKL algorithm (Titsias and Lazaro-Gredilla, 2011) is reported to reach saturation at $H = 64$ latent dimensions. As the learned sparse directions become less and less orthogonal the more over-complete the setting gets, this saturation may again be due to the bias introduced by the factored approach. GSC with $H = 256$ improves the state-of-the-art with 32.01dB for $\sigma = 25$ and with 28.35dB for $\sigma = 50$ (with even higher PSNR for $H = 400$). As we assume an independent Bernoulli prior per latent dimension, GSC can also prune out latent dimensions by inferring very low values of π_h for the bases that make negligible contribution in the inference procedure. This can be observed in Figure 8, where for the application of GSC to the denoising task with $\sigma = 25$, we found only about 107 of the 256 basis functions to have significant probabilities to contribute to the task. This means that GSC with about 100 basis functions can be expected to achieve almost the same performance as GSC with 256 basis functions. However, in practice we observed that the average performance increases with more basis functions because local optima can more efficiently be avoided. This observation is not limited to the particular approach studied here; also for other approaches to sparse learning, efficient avoidance of local optima has been reported if the number of assumed hidden dimensions was increased (e.g. Spratling, 2006; Lücke and Sahani, 2008). In comparison to MTMKL, GSC can make use of significantly more basis functions. It uses about 100 functions while MTMKL performance saturates at about 64 as mentioned previously. On the other hand, we found MTMKL to perform better on the low noise level setting (see $\sigma = 15$ in Table 2) or when relatively limited computational resources are available (see Figure 7).

In conclusion, we have studied a novel learning algorithm for sparse coding with spike-and-slab prior and compared it with a number of sparse coding approaches including other spike-and-slab based methods. The results we obtained show that the truncated EM approach is a competitive method. It shows that posterior dependencies and multi-modality can be captured by a scalable deterministic approximation. Furthermore, the direct comparison with a factored variational approach in source separation experiments confirms earlier observations that assumptions of a-posteriori independence introduces biases, and that avoiding such biases, e.g. by a truncated approach, improves the state-of-the-art on source separation benchmarks as well as on standard denoising tasks. However, we also find that under certain constraints and settings, factored variational learning for spike-and-slab sparse coding may perform as well or better. In general, our results argue in favor of spike-and-slab sparse coding models and recent efforts for developing improved algorithms for inference and learning in such models.

Acknowledgements

We acknowledge funding by the German Research Foundation (DFG), grant LU 1196/4-2, and by the German Ministry of Research and Education (BMBF), grant 01GQ0840 (BFNT Frankfurt). Furthermore, we acknowledge support by the Frankfurt Center for Scientific Computing (CSC Frankfurt).

Appendix A. Derivation of M-step Equations

Our goal is to optimize the free-energy w.r.t. Θ:

$$\begin{aligned}
\mathcal{F}(\Theta^{\mathrm{old}},\Theta) &= \sum_{n=1}^{N} \Big\langle \log p(\vec{y}^{\,(n)}, \vec{s}, \vec{z} \,|\, \Theta) \Big\rangle_n + H(\Theta^{\mathrm{old}}) \\
&= \sum_{n=1}^{N} \sum_{\vec{s}} \int_{\vec{z}} \; p(\vec{s}, \vec{z} \,|\, \vec{y}^{\,(n)}, \Theta^{\mathrm{old}}) \\
&\quad \Big[\log \big(p(\vec{y}^{\,(n)} \,|\, \vec{s}, \vec{z}, \Theta) \big) + \log \big(p(\vec{z} \,|\vec{s},\, \Theta) \big) + \log \big(p(\vec{s} \,|\, \Theta) \big) \Big] \,\mathrm{d}\vec{z} \,+\, H(\Theta^{\mathrm{old}})\,,
\end{aligned}$$

where

$$\begin{aligned}
\log \big(p(\vec{y}^{\,(n)} \,|\, \vec{s}, \vec{z}, \Theta) \big) &= -\frac{1}{2} \big(\log(2\pi^{D}) + \log|\Sigma| \big) \\
&\quad -\frac{1}{2} \Big(\vec{y}^{\,(n)} \,-\, W(\vec{s} \odot \vec{z}) \Big)^{\mathrm{T}} \,\Sigma^{-1}\, \Big(\vec{y}^{\,(n)} \,-\, W(\vec{s} \odot \vec{z}) \Big)\,, \\
\log \big(p(\vec{z} \,|\vec{s},\, \Theta) \big) &= -\frac{1}{2} \Big(\log(2\pi^{|\vec{s}|}) + \log|\Psi \odot \vec{s}\,\vec{s}^{\mathrm{T}}| \Big) \\
&\quad -\frac{1}{2} \big(\, (\vec{z} \,-\, \vec{\mu}) \odot \vec{s} \,\big)^{\mathrm{T}} \big(\Psi \odot \vec{s}\,\vec{s}^{\mathrm{T}} \big)^{-1} \big(\, (\vec{z} \,-\, \vec{\mu}) \odot \vec{s} \,\big) \\
\text{and} \qquad \log \big(p(\vec{s} \,|\Theta) \big) &= \sum_{h=1}^{H} \log \big(\pi_h^{s_h} \,(1-\pi_h)^{1-s_h} \big)\,.
\end{aligned}$$

The free-energy thus takes the form:

$$\begin{aligned}
\mathcal{F}(\Theta^{\mathrm{old}},\Theta) &= \sum_{n=1}^{N} \sum_{\vec{s}} \int_{\vec{z}} \; q_n(\vec{s}, \vec{z}; \Theta^{\mathrm{old}}) \\
&\quad \Big[-\tfrac{1}{2} \big(\log(2\pi^{D}) + \log|\Sigma| \big) - \tfrac{1}{2} \big(\vec{y}^{\,(n)} \,-\, W(\vec{s} \odot \vec{z}) \big)^{\mathrm{T}} \,\Sigma^{-1}\, \big(\vec{y}^{\,(n)} \,-\, W(\vec{s} \odot \vec{z}) \big) \\
&\quad -\tfrac{1}{2} \big(\log(2\pi^{|\vec{s}|}) + \log|\Psi \odot \vec{s}\,\vec{s}^{\mathrm{T}}| \big) \\
&\quad -\frac{1}{2} \big(\, (\vec{z} \,-\, \vec{\mu}) \odot \vec{s} \,\big)^{\mathrm{T}} \big(\Psi \odot \vec{s}\,\vec{s}^{\mathrm{T}} \big)^{-1} \big(\, (\vec{z} \,-\, \vec{\mu}) \odot \vec{s} \,\big) \\
&\quad + \textstyle\sum_{h=1}^{H} \log \big(\pi_h^{s_h} \,(1-\pi_h)^{1-s_h} \big) \Big] \,\mathrm{d}\vec{z} \,+\, H(\Theta^{\mathrm{old}})\,,
\end{aligned}$$

where $q_n(\vec{s},\vec{z};\Theta^{\mathrm{old}})$ denotes the posterior $p(\vec{s},\vec{z}\,|\,\vec{y}^{\,(n)},\Theta^{\mathrm{old}})$. Now we can derive the M-step equations (6) to (10) by canonically setting the derivatives of the free-energy above w.r.t. each parameter in Θ to zero.

A.1 Optimization of the Data Noise

Let us start with the derivation of the M-step equation for Σ:

$$
\begin{aligned}
&\frac{\partial}{\partial\Sigma}\mathcal{F}(\Theta^{\mathrm{old}},\Theta)\\
&\quad=\sum_{n=1}^{N}\sum_{\vec{s}}\int_{\vec{z}} q_n(\vec{s},\vec{z};\Theta^{\mathrm{old}})\\
&\qquad\qquad\Big[-\tfrac{1}{2}\tfrac{\partial}{\partial\Sigma}\left(\log|\Sigma|\right)-\tfrac{1}{2}\tfrac{\partial}{\partial\Sigma}\left(\vec{y}^{\,(n)}\;-\;W(\vec{s}\odot\vec{z})\right)^{\mathrm{T}}\,\Sigma^{-1}\,\left(\vec{y}^{\,(n)}\;-\;W(\vec{s}\odot\vec{z})\right)\Big]\mathrm{d}\vec{z}\\
&\quad=\sum_{n=1}^{N}\sum_{\vec{s}}\int_{\vec{z}} q_n(\vec{s},\vec{z};\Theta^{\mathrm{old}})\\
&\qquad\qquad\Big[-\tfrac{1}{2}\Sigma^{-1}+\tfrac{1}{2}\Sigma^{-2}\left(\vec{y}^{\,(n)}\;-\;W(\vec{s}\odot\vec{z})\right)\,\left(\vec{y}^{\,(n)}\;-\;W(\vec{s}\odot\vec{z})\right)^{\mathrm{T}}\;\Big]\mathrm{d}\vec{z}\overset{!}{=}0
\end{aligned}
$$

$$
\begin{aligned}
\Rightarrow\Sigma&=\frac{1}{N}\sum_{n=1}^{N}\sum_{\vec{s}}\int_{\vec{z}} q_n(\vec{s},\vec{z};\Theta^{\mathrm{old}})\Big[\,\left(\vec{y}^{\,(n)}\;-\;W(\vec{s}\odot\vec{z})\right)\left(\vec{y}^{\,(n)}\;-\;W(\vec{s}\odot\vec{z})\right)^{\mathrm{T}}\;\Big]\mathrm{d}\vec{z}\\
&=\frac{1}{N}\sum_{n=1}^{N}\Big[\Big(\vec{y}^{\,(n)}-W\big\langle(\vec{s}\odot\vec{z})\big\rangle_n\Big)\Big(\vec{y}^{\,(n)}-W\big\langle(\vec{s}\odot\vec{z})\big\rangle_n\Big)^{\mathrm{T}}\\
&\qquad\qquad\qquad\qquad+W\big[\big\langle(\vec{s}\odot\vec{z})(\vec{s}\odot\vec{z})^{\mathrm{T}}\big\rangle_n-\big\langle(\vec{s}\odot\vec{z})\big\rangle_n\big\langle(\vec{s}\odot\vec{z})\big\rangle_n^{\mathrm{T}}\big]W^{\mathrm{T}}\Big]\\
&=\frac{1}{N}\sum_{n=1}^{N}\Big[\vec{y}^{\,(n)}(\vec{y}^{\,(n)})^{\mathrm{T}}-W\big[\big\langle(\vec{s}\odot\vec{z})\big\rangle_n\big\langle(\vec{s}\odot\vec{z})\big\rangle_n^{\mathrm{T}}\big]W^{\mathrm{T}}\Big],
\end{aligned}
$$

where $\langle\,\cdot\,\rangle_n$ denotes the expectation value in Equation (5).

A.2 Optimization of the Bases

We will now derive the M-step update for the basis functions W:

$$
\begin{aligned}
&\frac{\partial}{\partial W}\mathcal{F}(\Theta^{\mathrm{old}},\Theta)\\
&\quad=\sum_{n=1}^{N}\sum_{\vec{s}}\int_{\vec{z}} q_n(\vec{s},\vec{z};\Theta^{\mathrm{old}})\Big[-\tfrac{1}{2}\tfrac{\partial}{\partial W}\left(\vec{y}^{\,(n)}\;-\;W(\vec{s}\odot\vec{z})\right)^{\mathrm{T}}\,\Sigma^{-1}\,\left(\vec{y}^{\,(n)}\;-\;W(\vec{s}\odot\vec{z})\right)\Big]\mathrm{d}\vec{z}\\
&\quad=\sum_{n=1}^{N}\sum_{\vec{s}}\int_{\vec{z}} q_n(\vec{s},\vec{z};\Theta^{\mathrm{old}})\Bigg[-\tfrac{1}{\Sigma}\bigg(\vec{y}^{\,(n)}(\vec{s}\odot\vec{z})^{\mathrm{T}}-W(\vec{s}\odot\vec{z})(\vec{s}\odot\vec{z})^{\mathrm{T}}\bigg)\Bigg]\mathrm{d}\vec{z}\overset{!}{=}0
\end{aligned}
$$

$$\Rightarrow W = \frac{\sum_{n=1}^{N} \vec{y}^{(n)} \big\langle \vec{s} \odot \vec{z} \big\rangle_n^{\mathrm{T}}}{\sum_{n=1}^{N} \big\langle (\vec{s} \odot \vec{z})(\vec{s} \odot \vec{z})^{\mathrm{T}} \big\rangle_n}.$$

A.3 Optimization of the Sparsity Parameter

Here we take the derivative of the free-energy w.r.t. $\vec{\pi}$:

$$\begin{aligned}
\frac{\partial}{\partial \vec{\pi}} \mathcal{F}(\Theta^{\mathrm{old}}, \Theta) &= \sum_{n=1}^{N} \sum_{\vec{s}} \int_{\vec{z}} q_n(\vec{s}, \vec{z}; \Theta^{\mathrm{old}}) \Bigg[\tfrac{\partial}{\partial \vec{\pi}} \Big(\vec{s} \log \vec{\pi} + (1 - \vec{s})\, log(1 - \vec{\pi}) \Big) \Bigg] \mathrm{d}\vec{z} \\
&= \sum_{n=1}^{N} \sum_{\vec{s}} q_n(\vec{s}; \Theta^{\mathrm{old}}) \Big[\tfrac{\vec{s}}{\vec{\pi}} - \tfrac{(1-\vec{s})}{(1-\vec{\pi})} \Big] \overset{!}{=} 0
\end{aligned}$$

$$\Rightarrow \vec{\pi} = \frac{1}{N} \sum_{n=1}^{N} \big\langle \vec{s} \big\rangle_n.$$

A.4 Optimization of the Latent Mean

Now we derive the M-step update for the mean $\vec{\mu}$ of the Gaussian slab:

$$\begin{aligned}
\frac{\partial}{\partial \vec{\mu}} \mathcal{F}(\Theta^{\mathrm{old}}, \Theta) &= \sum_{n=1}^{N} \sum_{\vec{s}} \int_{\vec{z}} q_n(\vec{s}, \vec{z}; \Theta^{\mathrm{old}}) \\
&\qquad \Big[-\tfrac{1}{2} \tfrac{\partial}{\partial \vec{\mu}} \big((\vec{z} - \vec{\mu}) \odot \vec{s} \,\big)^{\mathrm{T}} \big(\Psi \odot \vec{s}\,\vec{s}^{\mathrm{T}} \big)^{-1} \big((\vec{z} - \vec{\mu}) \odot \vec{s} \,\big) \Big] \mathrm{d}\vec{z} \\
&= \sum_{n=1}^{N} \sum_{\vec{s}} \int_{\vec{z}} q_n(\vec{s}, \vec{z}; \Theta^{\mathrm{old}}) \Bigg[\big(\Psi \odot \vec{s}\,\vec{s}^{\mathrm{T}} \big)^{-1} \Big((\vec{z} - \vec{\mu}) \odot \vec{s} \,\Big) \Bigg] \mathrm{d}\vec{z} \overset{!}{=} 0
\end{aligned}$$

$$\Rightarrow \vec{\mu} = \frac{\sum_{n=1}^{N} \big\langle \vec{s} \odot \vec{z} \big\rangle_n}{\sum_{n=1}^{N} \big\langle \vec{s} \big\rangle_n}.$$

A.5 Optimization of the Latent Covariance

Lastly we derive the M-step update for the latent covariance Ψ:

$$\begin{aligned}
&\frac{\partial}{\partial \Psi} \mathcal{F}(\Theta^{\mathrm{old}}, \Theta) \\
&\quad = \sum_{n=1}^{N} \sum_{\vec{s}} \int_{\vec{z}} q_n(\vec{s}, \vec{z}; \Theta^{\mathrm{old}}) \\
&\qquad \Big[-\tfrac{1}{2} \tfrac{\partial}{\partial \Psi} \big(\log |\Psi \odot \vec{s}\,\vec{s}^{\mathrm{T}}| \big) - \tfrac{1}{2} \tfrac{\partial}{\partial \Psi} \big((\vec{z} - \vec{\mu}) \odot \vec{s} \,\big)^{\mathrm{T}} \big(\Psi \odot \vec{s}\,\vec{s}^{\mathrm{T}} \big)^{-1} \big((\vec{z} - \vec{\mu}) \odot \vec{s} \,\big) \Big] \mathrm{d}\vec{z}
\end{aligned}$$

$$
\begin{aligned}
&= \sum_{n=1}^{N} \sum_{\vec{s}} \int_{\vec{z}} q_n(\vec{s}, \vec{z}; \Theta^{\mathrm{old}}) \\
&\qquad \Big[-\tfrac{1}{2} \big(\Psi \odot \vec{s}\,\vec{s}^{\mathrm{T}}\big)^{-1} + \tfrac{1}{2} \big(\Psi \odot \vec{s}\,\vec{s}^{\mathrm{T}}\big)^{-2} \big((\vec{z} - \vec{\mu}) \odot \vec{s}\, \big) \big((\vec{z} - \vec{\mu}) \odot \vec{s}\, \big)^{\mathrm{T}} \Big] \mathrm{d}\vec{z} \overset{!}{=} 0
\end{aligned}
$$

$$
\begin{aligned}
\Rightarrow \Psi &= \sum_{n=1}^{N} \Big[\big\langle\, (\vec{z} - \vec{\mu})\, (\vec{z} - \vec{\mu})^{\mathrm{T}} \odot \vec{s}\,\vec{s}^{\mathrm{T}} \big\rangle_n \Big] \odot \Big(\sum_{n=1}^{N} \Big[\big\langle \vec{s}\,\vec{s}^{\mathrm{T}} \big\rangle_n \Big] \Big)^{-1} \\
&= \sum_{n=1}^{N} \Big[\big\langle (\vec{s} \odot \vec{z})(\vec{s} \odot \vec{z})^{\mathrm{T}} \big\rangle_n - \big\langle \vec{s}\,\vec{s}^{\mathrm{T}} \big\rangle_n \odot \vec{\mu}\vec{\mu}^{\mathrm{T}} \Big] \odot \Big(\sum_{n=1}^{N} \Big[\big\langle \vec{s}\,\vec{s}^{\mathrm{T}} \big\rangle_n \Big] \Big)^{-1}.
\end{aligned}
$$

Appendix B. Performance vs. Complexity Trade-Off

If the approximation parameters H' and γ are held constant, the computational cost of the algorithm scales with the computational cost of the selection function. If the latter cost scales linearly with H (as is the case here), then so does the overall computational complexity (compare complexity considerations by Lücke and Eggert, 2010)). This is consistent with numerical experiments in which we measured the increase in computational demand (see Figure 10). In experiments with H increasing from 16 to 1024, we observed a, finally, close to linear increase of computational costs. However, a larger H implies a larger number of parameters, and thus may require more data points to prevent over-fitting. Although a larger data set increases computational demand, our truncated approximation algorithm allows us to take advantage of parallel computing architecture in order to more efficiently deal with large data sets (see Appendix C for details). Therefore in practice, we can weaken the extent of an increase in computational cost due to a higher demand for data. Furthermore, we examined the benefit of using GSC (in terms of average speedup over EM iterations) versus the cost regarding algorithmic performance. We compared approximation parameters in the range of $H' = \gamma = [1, 10]$ and again observed the performance of the algorithm on the task of source separation (with randomly generated orthogonal ground truth mixing bases and no observed noise). Figure 9 shows that a high accuracy can still be achieved for relatively small values of H' γ which, at the same time, results in strongly reduced computational demands.

Appendix C. Dynamic Data Repartitioning for Batch/Parallel Processing

As described in Section 4, the truncated variational approach deterministically selects the most likely H' causes of a given observation $\vec{y}$ for efficiently approximating the posterior distribution over a truncated latent space. In practice one can also use the selected latent causes for applying clustering to the observed data, which allows for an efficient and parallelizable batch-mode implementation of the E-step of the truncated variational EM algorithm.

In the batch processing mode, prior to each E-step the observed data can be partitioned by clustering together the data points w.r.t. their selected latent causes. The resulting clusters can then be processed individually (e.g., on multiple compute cores) to perform

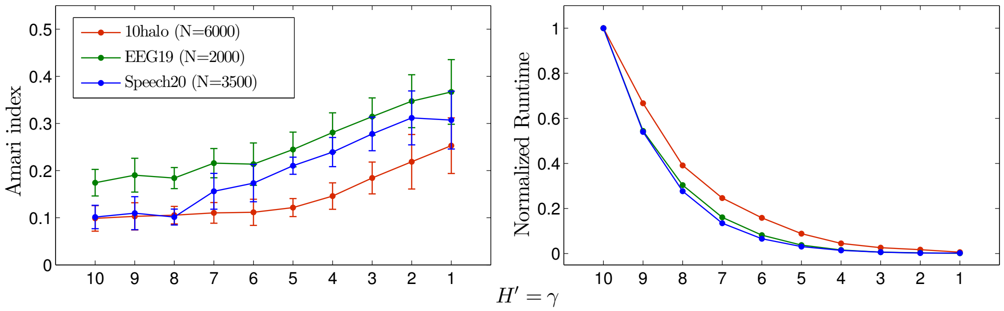

Figure 9: Performance of the GSC on 10halo, EEG19 and Speech20 benchmarks for decreasing truncation parameters H' and γ. The right plot shows how the computational demand of the truncated variational algorithm decreases with decreasing values of the truncation parameters. The runtime plots are normalized by the runtime value obtained for $H' = \gamma = 10$ for each of the benchmarks.

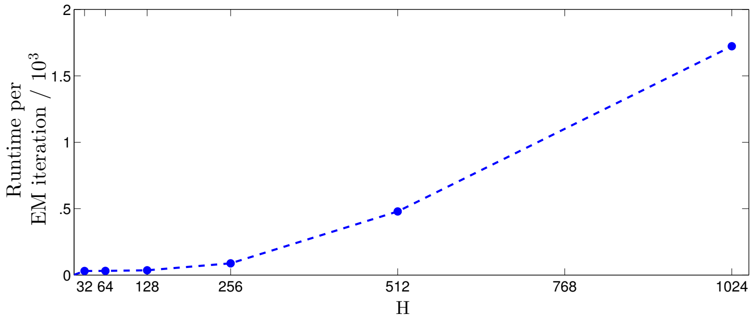

Figure 10: Time scaling behavior of GSC for increasing latent dimensions H and fixed truncation parameters H' and γ.

the E-step (Equations (21) to (23)) for all data points in a given cluster. This approach not only pursues a natural partitioning of data, but in a parallel execution environment, it can prove to be more efficient than uniformly distributing data (as in Bornschein et al., 2010) among multiple processing units. By maximizing the similarity (in latent space) of individual data points assigned to each of the processing units, we can overall minimize the number of redundant computations involved in Equations (15) and (23), that are tied to specific states of the latents. This can be observed by considering Equation (21), which is as follows:

$$p(\vec{s}, \vec{z} \,|\, \vec{y}^{\,(n)}, \Theta) \;\approx\; \frac{\mathcal{N}(\vec{y}^{\,(n)}; \vec{\mu}_{\vec{s}}, C_{\vec{s}})\, \mathcal{B}(\vec{s}; \vec{\pi})\, \mathcal{N}(\vec{z};\, \vec{\kappa}_{\vec{s}}^{(n)}, \Lambda_{\vec{s}})}{\sum_{\vec{s}\,' \in \mathcal{K}_n} \mathcal{N}(\vec{y}^{\,(n)}; \vec{\mu}_{\vec{s}\,'}, C_{\vec{s}\,'})\, \mathcal{B}(\vec{s}\,'; \vec{\pi})}\, \delta(\vec{s} \in \mathcal{K}_n). \tag{28}$$

Here the parameters $\vec{\mu}_{\vec{s}}, C_{\vec{s}}$ and $\Lambda_{\vec{s}}$ entirely depend on a particular latent state $\vec{s}$. Also, $\vec{\kappa}_{\vec{s}}^{(n)}$ takes prefactors that can be precomputed given the $\vec{s}$. It turns out that to compute (28) our clustering-based, dynamic data repartitioning and redistribution strategy is more efficient than the uniform data distribution approach of Bornschein et al. (2010). This is illustrated in Figure 11, which shows empirical E-step speedup over the latter approach taken as a baseline. The error bars were generated by performing 15 trials per given data size N. For all the trials, model scale (i.e., data dimensionality) and truncation approximation parameters were kept constant.[8] Each trial was run in parallel on 24 computing nodes. The red plot in the figure also shows the speedup as a result of an intermediate approach. There we initially uniformly distributed the data samples which were then only locally clustered by each processing unit at every E-step. The blue plot on the other hand shows the speedup as a result of globally clustering and redistributing the data prior to every E-step. All the reported results here also take into account the cost of data clustering and repartitioning.

In a parallel setup, we perform the data clustering process by having each processing unit cluster its own data locally and then merging the resulting clusters globally. In order to avoid uneven data distribution, we also bound the maximum size of a cluster. Currently we pick (per iteration) top α percentile of occurring cluster sizes as the threshold.[9] Any cluster larger than α is evenly broken into smaller clusters of maximum size α. Moreover, to minimize communication overhead among computational units, we actually only cluster and redistribute the data indices. This entails that the actual data must reside in a shared memory structure which is efficiently and dynamically accessible by all the computational units. Alternatively, all the units require their own copy of the whole data set.

Here we have introduced and illustrated the gains of dynamic data repartitioning technique in the context of a specific sparse coding model, which in fact involves computationally expensive, state-dependent operations for computing posterior distributions. The technique however is inherently generic and can be straight-forwardly employed for other types of multi-causal models.

[8] The observed and the latent dimensions of the GSC model were 25 and 20 respectively. The truncation approximation parameters H' and γ (maximum number of active causes in a given latent state) were 8 and 5 respectively.

[9] The α for the reported experiments was 5.

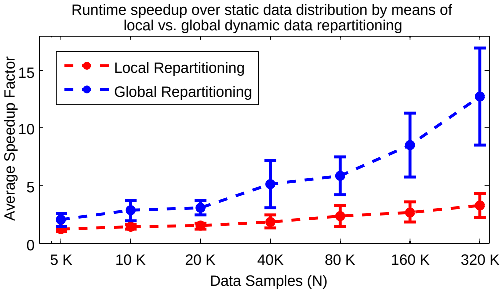

Figure 11: Runtime speedup of the truncated variational E-step (Equations (21) to (23)) with the static data distribution strategy taken as a baseline. The red plot shows the speedup when initially uniformly distributed data samples were only clustered locally by each processing unit, while the blue plot shows the speedup as a result of globally clustering and redistributing the data. The runtimes include the time taken by clustering and repartitioning modules.

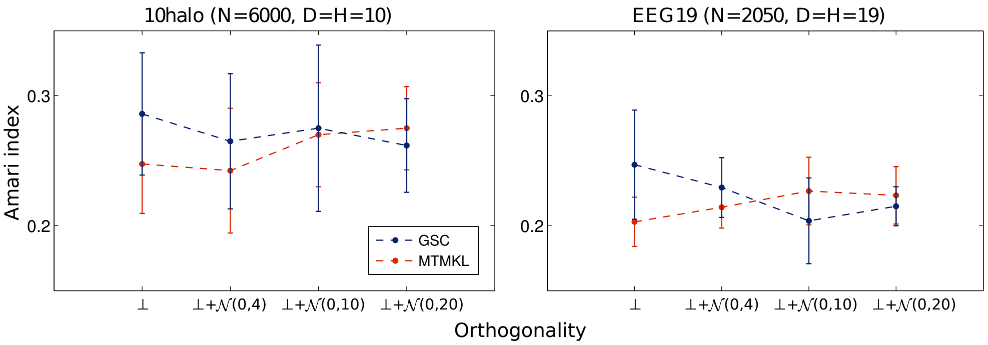

Figure 12: Source separation with observation noise. Performance of GSC vs. MTMKL on 10halo and EEG19 benchmarks with varying degrees of orthogonality of the mixing bases and Gaussian noise added to observations. Performance of GSC vs. MTMKL on the Speech20 benchmark with varying degrees of orthogonality of the mixing bases with Gaussian noise added to observed data. The orthogonality on the x-axis varies from being orthogonal $\perp$ to increasingly non-orthogonal mixing as randomly generated orthogonal bases are perturbed by adding Gaussian noise $\mathcal{N}(0, \sigma)$ to them. Performance is compared on the Amari index (27).

References

S. Amari, A. Cichocki, and H. H. Yang. A new learning algorithm for blind signal separation. In *Advances in Neural Information Processing Systems*, pages 757–763, 1995.

H. Attias. Independent factor analysis. *Neural Computation*, 11:803–851, 1999.

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

J. Bornschein, Z. Dai, and J. Lücke. Approximate EM learning on large computer clusters. In *NIPS Workshop on Big Learning*, 2010.

J. Bornschein, M. Henniges, and J. Lücke. Are v1 simple cells optimized for visual occlusions? A comparative study. *PLoS Computational Biology*, 9(6):e1003062, 2013.

P. Carbonetto and M. Stephen. Scalable variational inference for Bayesian variable selection in regression, and its accuracy in genetic association studies. *Bayesian Analysis Journal*, 2011.

C. M. Carvalho, J. Chang, J. E. Lucas, J. R. Nevins, Q. Wang, and M. West. High-dimensional sparse factor modeling: Applications in gene expression genomics. *Journal of the American Statistical Association*, 103(484):1438–1456, 2008.

A. Cichocki, S. Amari, K. Siwek, T. Tanaka, A.H. Phan, and R. Zdunek. ICALAB-MATLAB Toolbox Version 3, 2007.

Z. Dai and J. Lücke. Autonomous cleaning of corrupted scanned documents — a generative modeling approach. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3338–3345, 2012.

R. C. Dalen and M. J. F. Gales. Covariance modelling for noise-robust speech recognition. In *Annual Conference of the International Speech Communication Association*, pages 2000–2003, 2008.

R. C. Dalen and M. J. F. Gales. Extended VTS for noise-robust speech recognition. *IEEE Transactions on Audio, Speech and Language Processing*, 19(4), 2011.

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

J. V. Gael, Y. W. Teh, and Z. Ghahramani. The infinite factorial hidden Markov model. In *Advances in Neural Information Processing Systems*, pages 1697–1704, 2008.

P. Garrigues and B. A. Olshausen. Learning horizontal connections in a sparse coding model of natural images. In *Advances in Neural Information Processing Systems*, 2007.

I. Goodfellow, A. Courville, and Y. Bengio. Scaling up spike-and-slab models for unsupervised feature learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1902–1914, 2013.

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

A. Ilin and H. Valpola. On the effect of the form of the posterior approximation in variational learning of ICA models. *Neural Processing Letters*, 22(2):183–204, 2005.

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

D. Knowles and Z. Ghahramani. Infinite sparse factor analysis and infinite independent components analysis. In *Proceedings of International Conference on Independent Component Analysis and Signal Separation*, pages 381–388, 2007.

D. Knowles and Z. Ghahramani. Nonparametric Bayesian sparse factor models with application to gene expression modeling. *The Annals of Applied Statistics*, 5(2B):1534–1552, 2011.

H. Lee, A. Battle, R. Raina, and A. Ng. Efficient sparse coding algorithms. In *Advances in Neural Information Processing Systems*, pages 801–08, 2007.

H. Li and F. Liu. Image denoising via sparse and redundant representations over learned dictionaries in wavelet domain. In *Proceedings of International Conference on Image and Graphics*, pages 754–758, 2009.

J. Lücke and J. Eggert. Expectation truncation and the benefits of preselection in training generative models. *Journal of Machine Learning Research*, 11:2855–2900, 2010.

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

J. Lücke and A. S. Sheikh. Closed-form EM for sparse coding and its application to source separation. In *Proceedings of International Conference on Latent Variable Analysis and Signal Separation*, pages 213–221, 2012.

D. J. C. MacKay. Local minima, symmetry-breaking, and model pruning in variational free energy minimization. Online publication: www.inference.phy.cam.ac.uk/mackay/minima.ps.gz, 2001.

J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online dictionary learning for sparse coding. In *Proceedings of International Conference on Machine Learning*, page 87, 2009.

T. J. Mitchell and J. J. Beauchamp. Bayesian variable selection in linear regression. *Journal of the American Statistical Association*, 83(404):1023–1032, 1988.

S. Mohamed, K. Heller, and Z. Ghahramani. Evaluating Bayesian and L1 approaches for sparse unsupervised learning. In *Proceedings of International Conference on Machine Learning*, 2012.

E. Moulines, J.-F. Cardoso, and E. Gassiat. Maximum likelihood for blind separation and deconvolution of noisy signals using mixture models. In *Proceedings of International Conference on Acoustics, Speech and Signal Processing*, volume 5, pages 3617–3620, 1997.

G. J. Mysore, P. Smaragdis, and B. Raj. Non-negative hidden Markov modeling of audio with application to source separation. In *Proceedings of International Conference on Latent Variable Analysis and Signal Separation*, pages 140–148, 2010.

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

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

B. Olshausen and D. Field. Sparse coding with an overcomplete basis set: A strategy employed by V1? *Vision Research*, 37(23):3311–3325, 1997.

B. Olshausen and K. Millman. Learning sparse codes with a mixture-of-Gaussians prior. In *Advances in Neural Information Processing Systems*, volume 12, pages 841–847, 2000.

J. Paisley and L. Carin. Nonparametric factor analysis with beta process priors. In *Proceedings of International Conference on Machine Learning*, pages 777–784, 2009.

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.'A. Ranzato and G. Hinton. Modeling pixel means and covariances using factorized third-order boltzmann machines. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2551–2558, 2010.

M. Rattray, O. Stegle, K. Sharp, and J. Winn. Inference algorithms and learning theory for Bayesian sparse factor analysis. *Journal of Physics: Conference Series*, 197:012002 (10pp), 2009.

F. Ribeiro and M. Opper. Expectation propagation with factorizing distributions: A gaussian approximation and performance results for simple models. *Neural Computation*, 23: 1047–1069, 2011.

M. Seeger. Bayesian inference and optimal design for the sparse linear model. *Journal of Machine Learning Research*, 9:759–813, 2008.

J. Shelton, J. Bornschein, A.-S. Sheikh, P. Berkes, and J. Lücke. Select and sample — a model of efficient neural inference and learning. In *Advances in Neural Information Processing Systems*, pages 2618–2626, 2011.

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

T. Suzuki and M. Sugiyama. Least-squares independent component analysis. *Neural Computation*, 23(1):284–301, 2011.

Y. W. Teh, D. Görür, and Z. Ghahramani. Stick-breaking construction for the Indian buffet process. *Journal of Machine Learning Research*, 2:556–563, 2007.

M. Titsias and M. Lazaro-Gredilla. Spike and slab variational inference for multi-task and multiple kernel learning. In *Advances in Neural Information Processing Systems*, pages 2339–2347, 2011.

R. E. Turner and M. Sahani. Two problems with variational expectation maximisation for time-series models. In *Bayesian Time Series Models*. Cambridge University Press, 2011.

M. West. Bayesian factor regression models in the "large p, small n" paradigm. *Bayesian Statistics*, pages 723–732, 2003.

R. Yoshida and M. West. Bayesian learning in sparse graphical factor models via variational mean-field annealing. *Journal of the American Statistical Association*, 99:1771–1798, 2010.

M. Zhou, H. Chen, J. Paisley, L. Ren, G. Sapiro, and L. Carin. Non-parametric Bayesian dictionary learning for sparse image representations. In *Advances in Neural Information Processing Systems*, pages 2295–2303, 2009.