Application of non-linear transform coding to image processing

Jens Hocke[a] and Erhardt Barth[a] and Thomas Martinetz[a]

[a]Institute for Neuro- and Bioinformatics, University of Lübeck, Ratzeburger Allee 160, D-23538 Lübeck, Germany;

ABSTRACT

Sparse coding learns its basis non-linearly, but the basis elements are still linearly combined to form an image. Is this linear combination of basis elements a good model for natural images? We here use a non-linear synthesis rule, such that at each location in the image the point-wise maximum over all basis elements is used to synthesize the image. We present algorithms for image approximation and basis learning using this synthesis rule. With these algorithms we explore the the pixel-wise maximum over the basis elements as an alternative image model and thus contribute to the problem of finding a proper representation of natural images.

Keywords: image processing, sparse coding, maximal causes analysis, basis learning, sparseness, computational photography

1. INTRODUCTION

Novel developments in electronic imaging, for example computational photography, require better models of natural images that can help us to move from just taking an image with conventional hardware to synthesizing images based on more sophisticated devices like for example a light field camera. One classical approach is to represent images in a better basis and it has been successful for applications like image compression and denoising where images are represented in a wavelet basis.

An important extension is due to the work of Olshausen et al.[1,2] where optimal representations are learned by using the sparse coding principle. Sparse coding can help to solve under-determined problems based on the additional sparseness constraint.[3–8] However, while the learning of a basis is a non-linear procedure, the basis elements ϕ_k are still linearly combined to form an image:

$$\mathbf{x} = \sum_k a_k \phi_k, \tag{1}$$

where a_k is a scaling factor. Such a linear synthesis rule is not adequate for non-linear image features like for example occlusions that occur quite often in natural images. So one might ask if there are better models for image formation.

We here propose to use a non-linear synthesis rule which is defined such that at each location i in the image $\mathbf{x}$ the point-wise maximum over all basis elements ϕ_k is used to generate the image:

$$x(i) = \max_k a_k \phi_k(i). \tag{2}$$

Such a maximum rule has been proposed before by Lücke et.al.,[9–11] who coined it Maximal Causes Analysis (MCA).

For standard linear sparse coding there are several methods to determine the scaling factors a_k in (1) e.g. orthogonal matching pursuit (OMP)[12] or basis pursuit (BP).[13] Also for learning a basis there are several options like Sparsenet,[2] the method of optimal directions (MOD),[14] K-SVD[15] and sparse coding neural gas (SCNG).[16] We present two approaches for determining the a_k and a learning algorithm for ϕ_k in the context of MCA. Then we compare the results with linear sparse coding for both synthetic and natural images .

Further author information: (Send correspondence to Jens Hocke)
Jens Hocke: E-mail: hocke@inb.uni-luebeck.de

2. DETERMINING THE SCALING FACTORS

To approximate an image using the MCA synthesis paradigm defined by Eq. (2), we need to find the optimal scaling factors $\mathbf{a} = (a_k)$ for some given basis $[\phi_1 \phi_2 \ldots \phi_n] = \Phi \in \mathbb{R}^{m \times n}$.

We propose two approximation schemes. For both schemes we assume positive image values $x(i) \geq 0$ over the entire image. The first scheme is a greedy gradient descent based method. It starts with an empty list $\mathcal{I}$ of solution indices, indicating which basis elements are used in the solution. In every iteration a new index is added to the list. To select the new index, for every basis element $\phi_k, k \notin \mathcal{I}$ an optimal scaling factor a_k is found with a golden section search using the following error measure:

$$E(\mathbf{x}, \Phi, \mathbf{a}, \mathcal{L}) = \sum_i \left(x(i) - \max(a_{\mathcal{L}_1} \phi_{\mathcal{L}_1}(i), a_{\mathcal{L}_2} \phi_{\mathcal{L}_2}(i), \ldots, a_{\mathcal{L}_{|\mathcal{L}|}} \phi_{\mathcal{L}_{|\mathcal{L}|}}(i)) \right)^2, \tag{3}$$

where $\mathcal{L} = \mathcal{I} \cup k$ is the list of solution indices with the index k appended. Then the index $k^\star$, which minimizes the error, is added to $\mathcal{I}$. After the selection step the scaling factors $a_k, k \in \mathcal{I}$ for all the elements selected so far are updated with a line search along the gradient

$$\frac{\partial E(\mathbf{x}, \Phi, \mathbf{a}, \mathcal{I})}{\partial a_k} = -2 \sum_i \left(x(i) - \max(a_{\mathcal{I}_1} \phi_{\mathcal{I}_1}(i), a_{\mathcal{I}_2} \phi_{\mathcal{I}_2}(i), \ldots, a_{\mathcal{I}_{|\mathcal{I}|}} \phi_{\mathcal{I}_{|\mathcal{I}|}}(i)) \right) \cdot \phi_k(i). \tag{4}$$

For the line search also a golden section search was used. This update of the selected scaling factors can be repeated until it converges (we found a fixed value of five iterations to work fine). The selection and the update of all scaling factors are repeated until the desired sparsity s, i.e. $\mathbf{a}$ has s non-zero components, has been reached. Since this approach selects solution elements in a greedy manner and optimizes the scaling factors using a gradient descent we call the method greedy gradient maximal causes approximation (GGMCA).

We developed a second method to determine the scaling factors based on correlation. It is easy to see that for a basis containing only one basis element $\phi \in \mathbb{R}^{m \times 1}$ with unit length the optimal scaling factor a is given by its correlation, i.e. by the scalar product

$$a = \langle \mathbf{x}, \phi \rangle. \tag{5}$$

For more than one basis element, we start with an empty list of basis-element indices $\mathcal{I}$. Due to the non-linear combination of the basis elements, it is not possible to add basis elements based on their correlation to the residual, like it is done in orthogonal matching pursuit (OMP). We introduce the pattern vector $\mathbf{p}$, defined as follows:

$$p(i) = \begin{cases} 1, & \text{if } x(i) - \hat{x}(i) > 0 \\ 0, & \text{else,} \end{cases} \tag{6}$$

where $\hat{x}$ is the current approximation. Initially $\mathbf{p}$ is set to one everywhere. In each iteration one scaling factor is added to the list $\mathcal{I}$. The scaling factor are obtained as :

$$a_k = \left\langle \mathrm{diag}(\mathbf{p})\mathbf{x}, \frac{\mathrm{diag}(\mathbf{p})\phi_k}{||\mathrm{diag}(\mathbf{p})\phi_k||_2} \right\rangle. \tag{7}$$

where only basis elements $\phi_k, k \notin \mathcal{I}$ are considered. To minimize the error in the region $\mathbf{p}$ the index with the highest correlation $k^\star = \arg\max_k a_k$ is added to $\mathcal{I}$. Note that this selection step is locally optimal for the region $\mathbf{p}$ due to the projection, but may not be optimal globally for the whole image.

After a new index and, therefore, a new basis element has been added to the solution, the scaling coefficients a_k need to be updated. We introduce the variable m_{ik} to keep track of the basis element that contributes to the maximum at a particular pixel:

$$m_{ik} = \begin{cases} 1, & \text{if } k = \arg\max_{k'} a_{k'} \phi_{k'}(i) \\ 0, & \text{else.} \end{cases} \tag{8}$$

In the beginning all $m_{ik} = 0$. When a new element with the index $k^\star$ is added to the solution, the $m_{ik^\star}$ are set to $p(i)$. Using the m_{ik}, we can project the basis elements only at those pixels where they contribute to the solution:

$$a_k = \left\langle \mathrm{diag}(m_{ik})\mathbf{x}, \frac{\mathrm{diag}(m_{ik})\phi_k}{||\mathrm{diag}(m_{ik})\phi_k||_2} \right\rangle. \tag{9}$$

By using only the brightest pixels, that have maximum intensity values, in the current synthesis step, the scaling factor is optimized for these pixels only. The other pixels do not have an effect on the solution. Note that the above update rule depends on the m_{ik} and therefore the way the scaling factors a_k are obtained will of course change depending on which pixels are the brightest. Therefore, this step has to be repeated a few times until a good solution is found. The update of the a_k is followed by an update of the $m_{ik}, \exists i : k = \arg\max_{k'} a_{k'} \phi_{k'}(i)$. Only a subset of the m_{ik} are updated, because if after every update of a_k all m_{ik} were updated, then those basis elements, which do not contribute to the solution in the current step will never have the chance to contribute again. This is because all corresponding m_{ik} will be set to zero. Therefore, only those m_{ik} are updated for which the corresponding ϕ_k has contributed to the solution at some pixel.

The optimization of the coefficients a_k can be repeated till it converges but we used only five iterations in all examples. Then $\mathbf{p}$ is updated and we go back to the selection of the next basis element until the desired number of basis elements has been reached. The second approach is much faster than the first one due to the use of projections. In the following we will refer to the second approach as the iterative correlation maximal components approximation (ICMCA).

3. LEARNING ALGORITHM

Next we have to learn an optimal basis for the above image models. Suppose we have some non-optimal basis and found the scaling factors a_k by one of the approximation methods described above. Then the representation error for the signal $\mathbf{x}$ is:

$$E(\mathbf{x}, \Phi) = \sum_{i=1} \left(x(i) - \sum_{k=1} m_{ik} a_k \phi_k(i) \right)^2 . \tag{10}$$

This error function shall now be minimized based on its gradient for every basis element $\phi_{k^\star}(i^\star)$ at every pixel $i^\star$

$$\frac{\partial E(\mathbf{x}, \Phi)}{\partial \phi_{k^\star}(i^\star)} = \frac{\partial}{\partial \phi_{k^\star}(i^\star)} \left(x(i^\star) - \sum_{k=1} m_{i^\star k} a_k \phi_k(i^\star) \right)^2 = 2 \left(x(i^\star) - \sum_{k=1} m_{i^\star k} a_k \phi_k(i^\star) \right) \left(-m_{i^\star k^\star} a_{k^\star} \right) . \tag{11}$$

Then the gradient descent update rule, using some ϵ_1 as learning rate, is

$$\phi_k(i)^{(t+1)} = \phi_k(i)^{(t)} - \epsilon_1 \frac{\partial E(\mathbf{x}, \Phi^{(t)})}{\partial \phi_k^{(t)}} = \phi_k(i)^{(t)} + \epsilon_1 m_{ik} a_k \left(x(i) - \sum_{k'=1} m_{ik'} a_{k'} \phi_{k'}(i) \right) ; \forall i, \forall k. \tag{12}$$

Note that the factor 2 has been absorbed into the learning rate. We can see that only those basis elements are updated, which are the maximum elements at a particular pixel i, because for every other basis element the m_{ik} are equal to zero. We call this hard MCA learning because only one ϕ_k is updated at a particular pixel. This hard learning rule leads to the problem that two basis elements containing only parts of one true underlying basis element will most likely not converge to the single underlying basis element. Both elements will then be activated together and, since they represent the data correctly, there will be no further learning. Learning is therefore possible only in situations where, due to the sparseness constraint, one of the two basis elements is not selected.

To overcome this limitation we introduce a softer learning rule such that all elements are updated (not only the above winners). However, the elements that are not maximal at pixel i (and were not updated before) are updated at a different learning rate ϵ_2 and this leads to the following update rule

$$\begin{aligned} \phi_k(i)^{(t+1)} &= \phi_k(i)^{(t)} + \epsilon_1 m_{ik} a_k \left(x(i) - \sum_{k'=1} m_{ik'} a_{k'} \phi_{k'}(i) \right) && (13) \\ &\quad + \epsilon_2 \left(1 - m_{ik}\right) a_k \left(x(i) - \sum_{k'=1} m_{ik'} a_{k'} \phi_{k'}(i) \right) && (14) \\ &= \phi_k(i)^{(t)} + \left(\epsilon_1 m_{ik} + \epsilon_2 \left(1 - m_{ik}\right) \right) a_k \left(x(i) - \sum_{k'=1} m_{ik'} a_{k'} \phi_{k'}(i) \right) ; \forall i, \forall k. && (15) \end{aligned}$$

Note that since $m_{ik} = 1$ for the winners and zero else, the winners are updated with a learning rate of ϵ_1 and all other elements with ϵ_2.

We chose dynamic learning rates that decrease over time according to

$$\epsilon_1^{(t)} = \epsilon_1^{(0)} \left(\frac{\epsilon_1^{(\text{final})}}{\epsilon_1^{(0)}} \right)^{\frac{t}{t_{\max}}}, \epsilon_2^{(t)} = \epsilon_2^{(0)} \left(\frac{\epsilon_2^{(\text{final})}}{\epsilon_2^{(0)}} \right)^{\frac{t}{t_{\max}}}. \tag{16}$$

The final learning rate $\epsilon_2^{(\text{final})}$ is choosen to be much lower than $\epsilon_1^{(\text{final})}$, so that at the end the learning is hard (good values are $\epsilon_1^{(0)} = 0.9, \epsilon_1^{(\text{final})} = 10^{-3}, \epsilon_2^{(0)} = 0.9, \epsilon_2^{(\text{final})} = 10^{-6}$). We call this learning approach soft MCA learning.

4. EXPERIMENTS ON SYNTHETIC DATA

We started by testing the algorithms for determining the scaling factors on synthetic data. This has the advantage that we know the underlying basis. Some samples can be seen in Figure 1. These samples where generated from a basis consisting of 8 horizontal and 8 vertical bars on 5×5 pixel patches. Every basis element is chosen with probability 0.125, so there should be on average 2 active basis elements per image sample. The basis elements are then combined according to the non-linear maximum synthesis rule (2), with $a_k = 1$ for the active and $a_k = 0$ for the inactive basis elements. We compared GGMCA, ICMCA and, as a reference, linear sparse coding

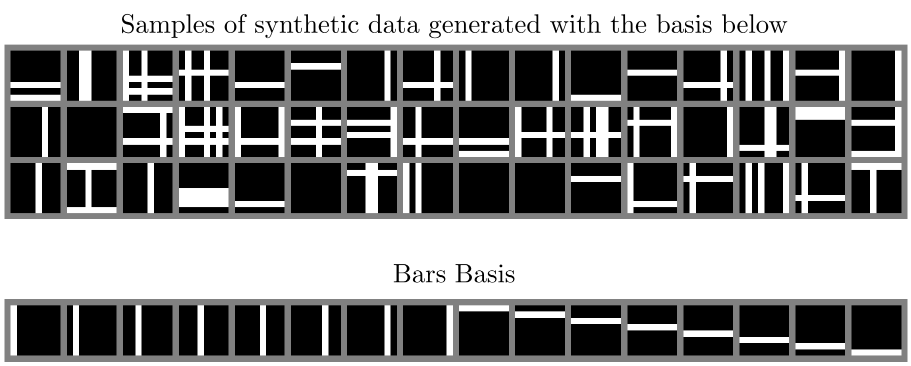

Figure 1. Example data and the basis underlying the data for the standard bars test.

OMP in terms of how well the a_k can be determined. To measure the approximation quality, the mean squared error (MSE) between the reconstructions and the original samples was used. For GGMCA, ICMCA and OMP, the MSE was averaged over 100 random samples. For all algorithms we used the parameters suggested in their description. Both methods we presented here perform significantly better (GGMCA: MSE=0.00019; ICMCA: MSE=0.00023) than OMP (MSE=0.0013). This result was predictable since GGMCA and ICMCA are based on the underlying model of the data, whereas OMP assumes linear combinations of the basis elements.

Next we compared hard MCA and soft MCA learning. Both need an algorithm to determine the scaling factors a_k. The hard MCA learning was tested with GGMCA. The soft MCA learning we tested with both, GGMCA and ICMCA. For comparison we also tested a linear model based algorithm. We chose Sparse Coding Neural Gas (SCNG) as a state of the art standard for sparse coding basis learning.

Learning a basis from the data as described above is known as the bars test and was introduced by Földiák.[17] We used $s = 2$ as sparsity level, since that is the expected sparsity of the samples. All our learning methods and also the SCNG were able to recover the basis from 10.000 samples generated from the bars basis. To make the problem more challenging, we changed the bars in the basis to have random instead of equal values. Also the basis elements where scaled with random scaling factors a_k when generating the samples (see Figure 2).

In Figure 3 we see the resulting basis elements obtained by using our algorithms and SCNG. Again the bases were learned from 10.000 samples. Clearly our soft learning algorithms learn the basis well, while the hard MCA

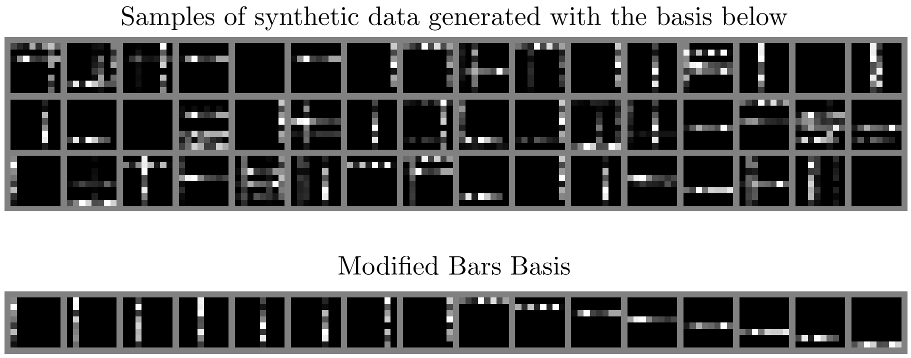

Figure 2. Example data and the basis underlying the data for the modified bars test.

learning and SCNG fail to learn all basis elements (there are more than one bars in a single basis element and in case of hard MCA learning there is one basis element without a bar). This does not always happen but often these methods end up in some local minima, whereas the soft learning methods find the solution reliably. Note, however, that SCNG has learned inverted versions of some basis elements which are as good for representing data as non-inverted ones. However, to make visual comparison easier we show the absolute values in Figure 3.

Using the synthetic data we were able to show that both algorithms for determining the scaling factors and the soft learning algorithm work in the non-linear model they were designed for. Also they are superior to the sparse coding algorithm designed for the linear model.

5. EXPERIMENTS ON NATURAL IMAGES

We have seen that our algorithms work better on the non-linear data than a standard linear sparse coding algorithm. So if the maximal components model is a better model for natural images than the standard linear model, then it should be possible to represent images with a higher quality at the same sparsity level of the coefficient vectors. Therefore, the next step was to test our approach on natural images.

First, we learned bases from 200.000 natural image patches. We used 8×8 patches from the van Hateren natural image database.[18] As sparsity level we set $s = 16$. The learned overcomplete bases with 128 basis

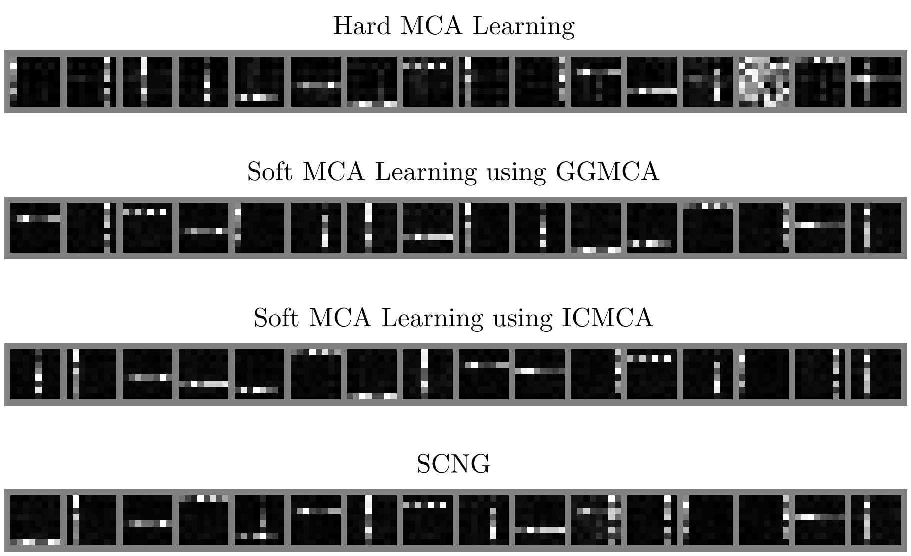

Figure 3. The results of basis learning for the modified bars test.

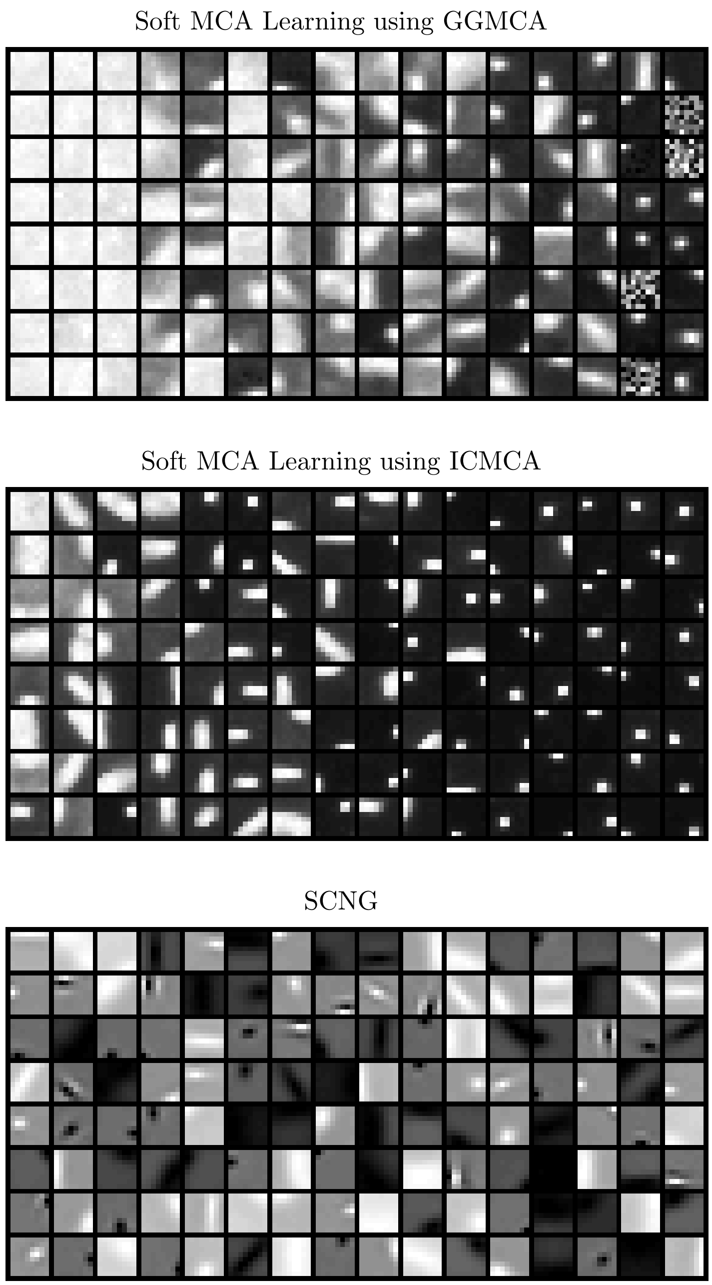

Figure 4. Bases learned from natural image patches with three different algorithms.

elements are shown in Figure 4. If the basis is learned using SCNG one can see the known oriented structures. Also both variants of the soft MCA learning yield oriented structures, but they differ significantly. In the basis obtained using soft MCA learning with GGMCA, there are some noisy elements, so probably the 200.000 image patches were not sufficient for convergence.

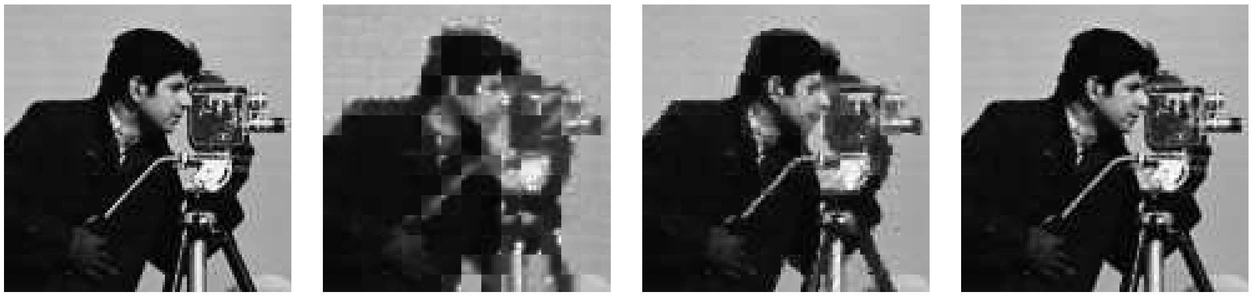

Figure 5. A section of the cameraman image and reconstructions of it.

To test how well the natural image bases that were learned above can represent images, we tested them on the cameraman image. The 256×256 pixel cameraman image was split up into 8×8 pixel tiles. So there are 32×32 tiles in total (the tiles are non-overlaping). These tiles where coded using the learned bases and algorithms for determining the scaling factors a_k. For both learning and synthesis the same algorithms were used to determine the scaling factors but for the basis learned with SCNG where we used OMP to determine the scaling factors. Thereafter the image was reconstructed from the coded patches according to the underlying model of the basis used for coding. The results of the reconstruction are shown in Figure 5. The quality was measured using the peak signal to noise ratio (PSNR). The best results are obtained with the SCNG representation (PSNR=32.18). Our MCA models yield visibly worse results (soft MCA learning using GGMCA with PSNR = 23.20, Soft MCA Learning using ICMCA with PSNR=26.22).

6. DISCUSSION

We presented algorithms for dealing with a non-linear image model called MCA. The algorithms we presented work well on synthetic data that fits the MCA model, but on natural images the results are still worse than with standard linear sparse coding. So either the MCA model is not as good as the linear model for natural images, or our approximation algorithms do not find the best solutions. This needs to be clarified in future work.

Even though we think that the model which underlies natural images is non-linear, the MCA model might not capture the right non-linearities. For example, the MCA model assumes that the brightest components in a scene are always in front of darker ones, which is not true in general. However, the MCA model might be better suited for coding gradients of images.

In conclusion, we have shown on synthetic data that optimal basis elements can be learned even in case of non-linear image models but it remains an open question how well this particular model describes natural images.

REFERENCES

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

[2] Olshausen, B. A. and Field, D. J., "Sparse coding with an overcomplete basis set: A strategy employed by v1?," *Vision Res.* **37**(23), 3311–3325 (1997).

[3] Donoho, D. L., "Compressed sensing.," *IEEE Transactions on Information Theory* **52**(4), 1289–1306 (2006).

[4] Candès, E. J. and Tao, T., "Decoding by linear programming.," *IEEE Transactions on Information Theory* **51**(12), 4203–4215 (2005).

[5] Elad, M. and Aharon, M., "Image denoising via learned dictionaries and sparse representation.," in [*CVPR (1)*], 895–900, IEEE Computer Society (2006).

[6] Lustig, M., Donoho, D., and Pauly, J., "Sparse mri: The application of compressed sensing for rapid mr imaging," *Magnetic Resonance in Medicine* **58**, 1182–1195 (2007).

[7] Marcia, R. F. and Willett, R. M., "Compressive coded aperture superresolution image reconstruction," *ICASSP 2008* , 833–836 (2008).

[8] Labusch, K., Barth, E., and Martinetz, T., "Soft-competitive Learning of Sparse Codes and its Application to Image Reconstruction," *Neurocomputing* **74**, 1418–1428 (April 2011).

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

[10] Lücke, J., "Receptive field self-organization in a model of the fine-structure in V1 cortical columns," *Neural Computation* **21**(10), 2805–2845 (2009).

[11] Puertas, J., Bornschein, J., and Lücke, J., "The maximal causes of natural scenes are edge filters," in [*Advances in Neural Information Processing Systems 23*], Lafferty, J., Williams, C. K. I., Zemel, R., Shawe-Taylor, J., and Culotta, A., eds., 1939–1947 (2010).

[12] Pati, Y., Rezaiifar, R., and Krishnaprasad, P., "Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition," in [*Signals, Systems and Computers, 1993. 1993 Conference Record of The Twenty-Seventh Asilomar Conference on*], 40–44, IEEE (1993).

[13] Chen, S., Donoho, D., and Saunders, M., "Atomic decomposition by basis pursuit," *SIAM Journal on Scientific Computing* **43**(1), 129–159 (2001).

[14] Engan, K., Aase, S., and Husøy, J., "Multi-frame compression: Theory and design," *Signal Processing* **80**(10), 2121–2140 (2000).

[15] Aharon, M., Elad, M., and Bruckstein, A., "The k-svd: An algorithm for designing of overcomplete dictionaries for sparse representation," *IEEE Trans. On Signal Processing* **54**(11), 4311–4322 (2006).

[16] Labusch, K., Barth, E., and Martinetz, T., "Sparse coding neural gas: Learning of overcomplete data representations," *Neurocomputing* **72**, 1547–1555 (2009).

[17] Földiák, P., "Forming sparse representations by local anti-hebbian learning," *Biological cybernetics* **64**(2), 165–170 (1990).

[18] van Hateren, J. and van der Schaaf, A., "Independent component filters of natural images compared with simple cells in primary visual cortex," *Proceedings of the Royal Society of London. Series B: Biological Sciences* **265**(1394), 359 (1998).