NON-NEGATIVE SPARSE CODING

Patrik O. Hoyer
Neural Networks Research Centre
Helsinki University of Technology
P.O. Box 9800, FIN-02015 HUT, Finland
patrik.hoyer@hut.fi

Abstract. Non-negative sparse coding is a method for decomposing multivariate data into non-negative sparse components. In this paper we briefly describe the motivation behind this type of data representation and its relation to standard sparse coding and non-negative matrix factorization. We then give a simple yet efficient multiplicative algorithm for finding the optimal values of the hidden components. In addition, we show how the basis vectors can be learned from the observed data. Simulations demonstrate the effectiveness of the proposed method.

INTRODUCTION

Linear data representations are widely used in signal processing and data analysis. A traditional method of choice for signal representation is of course Fourier analysis, but also wavelet representations are increasingly being used in a variety of applications. Both of these methods have strong mathematical foundations and fast implementations, but they share the important drawback that they are not adapted to the particular data being analyzed.

Data-adaptive representations, on the other hand, are representations that are tailored to the statistics of the data. Such representations are learned directly from the observed data by optimizing some measure that quantifies the desired properties of the representation. This class of methods include principal component analysis (PCA), independent component analysis (ICA), sparse coding, and non-negative matrix factorization (NMF). Some of these methods have their roots in neural computation, but have since been shown to be widely applicable for signal analysis.

In this paper we propose to combine sparse coding and non-negative matrix factorization into *non-negative sparse coding* (NNSC). Again, the motivation comes partly from modeling neural information processing. We believe that, as with previous methods, this technique will be found useful in a more general signal processing framework.

NON-NEGATIVE SPARSE CODING

Assume that we observe data in the form of a large number of i.i.d. random vectors $\mathbf{x}_n$, where n is the sample index. Arranging these into the columns of a matrix $\mathbf{X}$, then linear decompositions describe this data as $\mathbf{X} \approx \mathbf{AS}$. The matrix $\mathbf{A}$ is called the *mixing matrix*, and contains as its columns the *basis vectors* (features) of the decomposition. The rows of $\mathbf{S}$ contain the corresponding *hidden components* that give the contribution of each basis vector in the input vectors. Although some decompositions provide an exact reconstruction of the data (i.e. $\mathbf{X} = \mathbf{AS}$) the ones that we shall consider here are approximative in nature.

In linear sparse coding [2, 8], the goal is to find a decomposition in which the hidden components are *sparse*, meaning that they have probability densities which are highly peaked at zero and have heavy tails. This basically means that any given input vector can be well represented using only a few significantly non-zero hidden coefficients. Combining the goal of small reconstruction error with that of sparseness, one can arrive at the following objective function to be minimized [2, 8]:

$$C(\mathbf{A}, \mathbf{S}) = \frac{1}{2}\|\mathbf{X} - \mathbf{AS}\|^2 + \lambda \sum_{ij} f(S_{ij}), \tag{1}$$

where the squared matrix norm is simply the summed squared value of the elements, i.e. $\|\mathbf{X}-\mathbf{AS}\|^2 = \sum_{ij}[\mathbf{X}_{ij} - (\mathbf{AS})_{ij}]^2$. The tradeoff between sparseness and accurate reconstruction is controlled by the parameter λ, whereas the form of f defines how sparseness is measured. To achieve a sparse code, the form of f must be chosen correctly: A typical choice is $f(s) = |s|$, although often similar functions that exhibit smoother behaviour at zero are chosen for numerical stability.

There is one important problem with this objective: As f typically is a strictly increasing function of the absolute value of its argument, the objective can always be decreased by simply scaling up $\mathbf{A}$ and correspondingly scaling down $\mathbf{S}$. The consequences of this is that optimization of (1) with respect to both $\mathbf{A}$ and $\mathbf{S}$ leads to the elements of $\mathbf{A}$ growing (in absolute value) without bounds whereas $\mathbf{S}$ tends to zero. More importantly, the solution found does not depend on the second term of the objective as it can always be eliminated by this scaling trick. In other words, some constraint on the scales of $\mathbf{A}$ or $\mathbf{S}$ is needed. Olshausen and Field [8] used an adaptive method to ensure that the hidden components had unit variance (effectively fixing the norm of the rows of $\mathbf{S}$), whereas Harpur [1] fixed the norms of the columns of $\mathbf{A}$.

With either of the above scale constraints the objective (1) is well-behaved and its minimization can produce useful decompositions of many types of data. For example, it was shown in [8] that applying this method to image data yielded features closely resembling simple-cell receptive fields in the mammalian primary visual cortex. The learned decomposition is also similar to wavelet decompositions, implying that it could be useful in applications where wavelets have been successfully applied.

In standard sparse coding, described above, the data is described as a combination of elementary features involving both additive and subtractive interactions. The fact that features can 'cancel each other out' using subtraction is contrary to the intuitive notion of combining parts to form a whole [6]. Thus, Lee and Seung [6, 7] have recently forcefully argued for non-negative representations [9]. Other arguments for non-negative representations come from biological modeling [3, 4, 6], where such constraints are related to the non-negativity of neural firing rates. These non-negative representations assume that the input data $\mathbf{X}$, the basis $\mathbf{A}$, and the hidden components $\mathbf{S}$ are all non-negative.

Non-negative matrix factorization[1] (NMF) can be performed by the minimization of the following objective function [7, 9]:

$$C(\mathbf{A}, \mathbf{S}) = \frac{1}{2}\|\mathbf{X} - \mathbf{A}\mathbf{S}\|^2 \tag{2}$$

with the non-negativity constraints $\forall ij: \; A_{ij} \geq 0, \; S_{ij} \geq 0$. This objective requires no constraints on the scales of $\mathbf{A}$ or $\mathbf{S}$.

In [6], the authors showed how non-negative matrix factorization applied to face images yielded features that corresponded to intuitive notions of face parts: lips, nose, eyes, etc. This was contrasted with the holistic representations learned by PCA and vector quantization.

We suggest that both the non-negativity constraints and the sparseness goal are important for learning parts-based representations. Thus, we propose to combine these two methods into non-negative sparse coding:

Definition 1 *Non-negative sparse coding (NNSC) of a non-negative data matrix* $\mathbf{X}$ *(i.e.* $\forall ij: \; X_{ij} \geq 0$*) is given by the minimization of*

$$C(\mathbf{A}, \mathbf{S}) = \frac{1}{2}\|\mathbf{X} - \mathbf{A}\mathbf{S}\|^2 + \lambda \sum_{ij} S_{ij} \tag{3}$$

subject to the constraints $\forall ij: \; A_{ij} \geq 0, \; S_{ij} \geq 0$ *and* $\forall i: \|\mathbf{a}_i\| = 1$*, where* $\mathbf{a}_i$ *denotes the i:th column of* $\mathbf{A}$*. It is also assumed that the constant* $\lambda \geq 0$*.*

Notice that we have here chosen to measure sparseness by a linear activation penalty (i.e. $f(s) = s$). This particular choice is primarily motivated by the fact that this makes the objective function quadratic in $\mathbf{S}$. This is useful in the development and convergence proof of an efficient algorithm for optimizing the hidden components $\mathbf{S}$.

ESTIMATING THE HIDDEN COMPONENTS

We will first consider optimizing $\mathbf{S}$, for a given basis $\mathbf{A}$. As the objective (3) is quadratic with respect to $\mathbf{S}$, and the set of allowed $\mathbf{S}$ (i.e. the set where $S_{ij} \geq$

[1]Note that error measures other than the summed squared error were also considered in [6, 7].

0) is convex, we are guaranteed that no suboptimal local minima exist. The global minimum can be found using, for example, quadratic programming or gradient descent. Gradient descent is quite simple to implement, but convergence can be slow. On the other hand, quadratic programming is much more complicated to implement. To address these concerns, we have developed a multiplicative algorithm based on the one introduced in [7] that is extremely simple to implement and nonetheless seems to be quite efficient. This is given by iterating the following update rule:

Theorem 1 *The objective (3) is nonincreasing under the update rule:*

$$\mathbf{S}^{t+1} = \mathbf{S}^{t} \;.\!*\; (\mathbf{A}^T\mathbf{X}) \;./\; (\mathbf{A}^T\mathbf{A}\mathbf{S}^t + \lambda) \tag{4}$$

where $.*$ *and* $./$ *denote elementwise multiplication and division (respectively), and the addition of the scalar* λ *is done to every element of the matrix* $\mathbf{A}^T\mathbf{A}\mathbf{S}^t$*.*

This is proven in the Appendix. As each element of $\mathbf{S}$ is updated by simply multiplying with some non-negative factor, it is guaranteed that the elements of $\mathbf{S}$ stay non-negative under this update rule. As long as the initial values of $\mathbf{S}$ are all chosen strictly positive, iteration of this update rule is in practice guaranteed to reach the global minimum to any required precision.

LEARNING THE BASIS

In this section we consider optimizing the objective (3) with respect to both the basis $\mathbf{A}$ and the hidden components $\mathbf{S}$, under the stated constraints. First, we consider the optimization of $\mathbf{A}$ only, holding $\mathbf{S}$ fixed.

Minimizing (3) with respect to $\mathbf{A}$ *under the non-negativity constraint only* could be done exactly as in [7], with a simple multiplicative update rule. However, the constraint of unit-norm columns of $\mathbf{A}$ complicates things. We have not found any similarly efficient update rule that would be guaranteed to decrease the objective while obeying the required constraint. Thus, we here resort to projected gradient descent. Each step is composed of three parts:

1. $\mathbf{A}' = \mathbf{A}^t - \mu(\mathbf{A}^t\mathbf{S} - \mathbf{X})\mathbf{S}^T$
2. Any negative values in $\mathbf{A}'$ are set to zero
3. Rescale each column of $\mathbf{A}'$ to unit norm, and then set $\mathbf{A}^{t+1} = \mathbf{A}'$.

This combined step consists of a gradient descent step (Step 1) followed by projection onto the closest point satisfying both the non-negativity and the unit-norm constraints (Steps 2 and 3). This projected gradient step is guaranteed to decrease the objective if the stepsize $\mu > 0$ is small enough and we are not already at a local minimum. (In this case there is no guarantee of reaching the *global* minimum, due to the non-convex constraints.)

In the previous section, we gave an update step for $\mathbf{S}$, holding $\mathbf{A}$ fixed. Above, we showed how to update $\mathbf{A}$, holding $\mathbf{S}$ fixed. To optimize the objective with respect to both, we can of course take turns updating $\mathbf{A}$ and $\mathbf{S}$. This yields the following algorithm:

```
Algorithm for NNSC

  1. Initialize $\mathbf{A}^0$ and $\mathbf{S}^0$ to random strictly positive matrices of the
     appropriate dimensions, and rescale each column of $\mathbf{A}^0$ to unit
     norm. Set $t = 0$.

  2. Iterate until convergence:

     (a) $\mathbf{A}' = \mathbf{A}^t - \mu(\mathbf{A}^t\mathbf{S}^t - \mathbf{X})(\mathbf{S}^t)^T$
     (b) Any negative values in $\mathbf{A}'$ are set to zero
     (c) Rescale each column of $\mathbf{A}'$ to unit norm, and then set
         $\mathbf{A}^{t+1} = \mathbf{A}'$.
     (d) $\mathbf{S}^{t+1} = \mathbf{S}^t .* ((\mathbf{A}^{t+1})^T\mathbf{X}) ./ ((\mathbf{A}^{t+1})^T(\mathbf{A}^{t+1})\mathbf{S}^t + \lambda)$
     (e) Increment $t$.
```

EXPERIMENTS

To demonstrate how sparseness can be essential for learning a parts-based non-negative representation, we performed a simple simulation where the generating features were known. The interested reader can find the code to perform these experiments (as well as the experiments reported in [3]) on the web at:

`http://www.cis.hut.fi/phoyer/code/`

In our simulations, the data vectors were 3×3 -pixel images with non-negative pixel values. We manually constructed 10 original features: the six possible horizontal and vertical bars, and the four possible horizontal and vertical double bars. Each feature was normalized to unit norm, and entered as a column in the matrix $\mathbf{A}_{\text{orig}}$. The features are shown in the leftmost panel of Figure 1. We then generated random sparse non-negative data $\mathbf{S}_{\text{orig}}$, and obtained the data vectors as $\mathbf{X} = \mathbf{A}_{\text{orig}}\mathbf{S}_{\text{orig}}$. A random sample of 12 such data vectors are also shown in Figure 1.

We ran NNSC and NMF on this data $\mathbf{X}$. With 10 hidden components (rows of $\mathbf{S}$), NNSC can correctly identify all the features in the dataset. This result is shown in Figure 1 under **NNSC**. However, NMF cannot find all the features with any hidden dimensionality. With 6 components, NMF finds all the single bar features. With a dimensionality of 10, not even all of the single bars are correctly estimated. These results are illustrated in the two

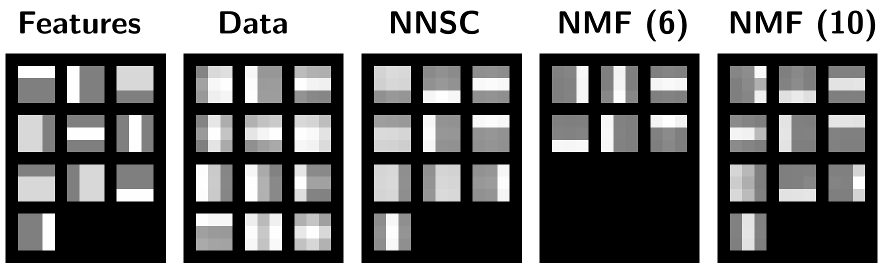

Figure 1: Experiments on bars data. Features: **The** 10 **original features that were used to construct the dataset.** Data: **A random sample of 12 data vectors. These constitute superpositions of the original features.** NNSC: **Features learned by NNSC, with dimensionality of the hidden representation equal to 10, starting from random initial values.** NMF (6): **Features learned by NMF, with dimensionality 6.** NMF (10): **Features learned by NMF, with dimensionality 10. See main text for discussion.**

rightmost panels of Figure 1.

It is not difficult to understand why NMF cannot learn all the features. The data $\mathbf{X}$ can be perfectly described as an additive combination of the six single bars (because all double bars can be described as two single bars). Thus, NMF essentially achieves the optimum (zero reconstruction error) already with 6 features, and there is no way in which an *overcomplete* representation could improve that. However, when sparseness is considered as in NNSC, it is clear that it is useful to have double bar features because these allow a sparser description of such data patterns.

In addition to these simulations, we have performed experiments with natural image data, reported elsewhere [3, 4]. These confirm our belief that sparseness is important when learning non-negative representations from data.

RELATION TO OTHER WORK

In addition to the tight connection to linear sparse coding [2, 8] and non-negative matrix factorization [6, 7, 9], this method is intimately related to independent component analysis [5]. In fact, when the fixed-norm constraint is placed on the rows of $\mathbf{S}$ instead of the columns of $\mathbf{A}$, the objective (3) could be directly interpreted as the negative joint log-posterior of the basis vectors and components, given the data $\mathbf{X}$, in the noisy ICA model [4]. This connection is valid when the independent components are assumed to have exponential distributions, and of course the basis vectors are assumed to be non-negative as well.

Other researchers have also recently considered the constraint of non-negativity in the context of ICA. In particular, Plumbley [11] has considered estimation of the noiseless ICA model (with equal dimensionality of components and observations) in the case of non-negative components. On the other hand, Parra et al. [10] considered estimation of the ICA model where the basis (but not the components) was constrained to be non-negative. The main novelty of the present work is the application of the non-negativity constraints in the sparse coding framework, and the simple yet efficient algorithm developed to estimate the components.

CONCLUSIONS

In this paper, we have defined non-negative sparse coding as a combination of sparse coding with the constraints of non-negative matrix factorization. Although this is essentially a special case of the general sparse coding framework, we believe that the proposed constraints can be important for learning parts-based representations from non-negative data. In addition, the constraints allow a very simple yet efficient algorithm for estimating the hidden components.

APPENDIX

To prove Theorem 1, first note that the objective (3) is separable in the columns of **S** so that each column can be optimized without considering the others. We may thus consider the problem for the case of a single column, denoted **s**. The corresponding column of **X** is denoted **x**, giving the objective

$$F(\mathbf{s}) = \frac{1}{2}\|\mathbf{x} - \mathbf{A}\mathbf{s}\|^2 + \lambda \sum_i s_i. \tag{5}$$

The proof will follow closely the proof given in [7] for the case $\lambda = 0$. (Note that in [7], the notation $v = \mathbf{x}$, $W = \mathbf{A}$ and $h = \mathbf{s}$ was used.) We define an auxiliary function $G(\mathbf{s}, \mathbf{s}^t)$ with the properties that $G(\mathbf{s}, \mathbf{s}) = F(\mathbf{s})$ and $G(\mathbf{s}, \mathbf{s}^t) \geq F(\mathbf{s})$. We will then show that the multiplicative update rule corresponds to setting, at each iteration, the new state vector to the values that minimize the auxiliary function:

$$\mathbf{s}^{t+1} = \arg\min_{\mathbf{s}} G(\mathbf{s}, \mathbf{s}^t). \tag{6}$$

This is guaranteed not to increase the objective function F, as

$$F(\mathbf{s}^{t+1}) \leq G(\mathbf{s}^{t+1}, \mathbf{s}^t) \leq G(\mathbf{s}^t, \mathbf{s}^t) = F(\mathbf{s}^t). \tag{7}$$

Following [7], we define the function G as

$$G(\mathbf{s}, \mathbf{s}^t) = F(\mathbf{s}^t) + (\mathbf{s} - \mathbf{s}^t)^T \nabla F(\mathbf{s}^t) + \frac{1}{2}(\mathbf{s} - \mathbf{s}^t)^T \mathbf{K}(\mathbf{s}^t)(\mathbf{s} - \mathbf{s}^t) \tag{8}$$

where the diagonal matrix $\mathbf{K}(\mathbf{s}^t)$ is defined as

$$K_{ab}(\mathbf{s}^t) = \delta_{ab} \frac{(\mathbf{A}^T \mathbf{A}\mathbf{s}^t)_a + \lambda}{\mathbf{s}_a^t}. \tag{9}$$

It is important to note that the elements of our choice for $\mathbf{K}$ are always greather than or equal to those of the $\mathbf{K}$ used in [7], which is the case where $\lambda = 0$. It is obvious that $G(\mathbf{s}, \mathbf{s}) = F(\mathbf{s})$. Writing out

$$F(\mathbf{s}) = F(\mathbf{s}^t) + (\mathbf{s} - \mathbf{s}^t)^T \nabla F(\mathbf{s}^t) + \frac{1}{2}(\mathbf{s} - \mathbf{s}^t)^T (\mathbf{A}^T \mathbf{A})(\mathbf{s} - \mathbf{s}^t), \tag{10}$$

we see that the second property, $G(\mathbf{s}, \mathbf{s}') \geq F(\mathbf{s})$, is satisfied if

$$0 \leq (\mathbf{s} - \mathbf{s}^t)^T [\mathbf{K}(\mathbf{s}^t) - \mathbf{A}^T \mathbf{A}](\mathbf{s} - \mathbf{s}^t). \tag{11}$$

Lee and Seung proved this positive semidefiniteness for the case of $\lambda = 0$ [7]. In our case, with $\lambda > 0$, the matrix whose positive semidefiniteness is to be proved is the same except that a strictly non-negative diagonal matrix has been added (see the above comment on the choice of $\mathbf{K}$). As a non-negative diagonal matrix is positive semidefinite, and the sum of two positive semidefinite matrices is also positive semidefinite, the $\lambda = 0$ proof in [7] also holds when $\lambda > 0$.

It remains to be shown that the update rule in (4) selects the minimum of G. This minimum is easily found by taking the gradient and equating it to zero:

$$\nabla_{\mathbf{s}} G(\mathbf{s}, \mathbf{s}^t) = \mathbf{A}^T(\mathbf{A}\mathbf{s}^t - \mathbf{x}) + \lambda \mathbf{c} + \mathbf{K}(\mathbf{s}^t)(\mathbf{s} - \mathbf{s}^t) = 0, \tag{12}$$

where $\mathbf{c}$ is a vector with all ones. Solving for $\mathbf{s}$, this gives

$$\begin{aligned}
\mathbf{s} &= \mathbf{s}^t - \mathbf{K}^{-1}(\mathbf{s}^t)(\mathbf{A}^T\mathbf{A}\mathbf{s}^t - \mathbf{A}^T\mathbf{x} + \lambda\mathbf{c}) && (13)\\
&= \mathbf{s}^t - (\mathbf{s}^t./(\mathbf{A}^T\mathbf{A}\mathbf{s}^t + \lambda\mathbf{c})).\!* (\mathbf{A}^T\mathbf{A}\mathbf{s}^t - \mathbf{A}^T\mathbf{x} + \lambda\mathbf{c}) && (14)\\
&= \mathbf{s}^t.\!* (\mathbf{A}^T\mathbf{x})./(\mathbf{A}^T\mathbf{A}\mathbf{s}^t + \lambda\mathbf{c})) && (15)
\end{aligned}$$

which is the desired update rule (4). This completes the proof.

REFERENCES

[1] G. F. Harpur, **Low Entropy Coding with Unsupervised Neural Networks**, Ph.D. thesis, University of Cambridge, 1997.

[2] G. F. Harpur and R. W. Prager, "Development of low entropy coding in a recurrent network," **Network: Computation in Neural Systems**, vol. 7, pp. 277–284, 1996.

[3] P. O. Hoyer, "Modeling receptive fields with non-negative sparse coding," Submitted. Available online, see `http://www.cis.hut.fi/phoyer/papers/`.

[4] P. O. Hoyer and A. Hyvärinen, "A multi-layer sparse coding network learns contour coding from natural images," **Vision Research**, In press. Available online, see `http://www.cis.hut.fi/phoyer/papers/`.

[5] A. Hyvärinen, J. Karhunen and E. Oja, **Independent Component Analysis**, Wiley Interscience, 2001.

[6] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," **Nature**, vol. 401, no. 6755, pp. 788–791, 1999.

[7] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in **Advances in Neural Information Processing 13 (Proc. NIPS*2000)**, MIT Press, 2001.

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

[9] P. Paatero and U. Tapper, "Positive Matrix Factorization: A Non-negative Factor Model with Optimal Utilization of Error Estimates of Data Values," **Environmetrics**, vol. 5, pp. 111–126, 1994.

[10] L. Parra, C. Spence, P. Sajda, A. Ziehe and K.-R. Müller, "Unmixing Hyperspectral Data," in **Advances in Neural Information Processing 12 (Proc. NIPS*99)**, MIT Press, pp. 942–948, 2000.

[11] M. Plumbley, "Conditions for non-negative independent component analysis," **IEEE Signal Processing Letters**, Submitted.

ACKNOWLEDGEMENTS

I wish to acknowledge Aapo Hyvärinen for useful discussions and helpful comments on an earlier version of the manuscript.