Fast atomic decomposition by the inhibition method

Pece, Arthur; Petkov, Nikolay

Published in:
EPRINTS-BOOK-TITLE

Fast atomic decomposition by the inhibition method

Arthur Pece
Department of Computer Science
University of Reading
P.O.Box 225, RG6 6AY Reading, U.K.
A.E.C.Pece@reading.ac.uk

Nikolay Petkov
Department of Computing Science
University of Groningen
P.O.Box 800, 9700 AV Groningen
The Netherlands
petkov@cs.rug.nl

Abstract

A new algorithm is introduced which is related to matching pursuit but allows updating more than one coding coefficient per iteration: the updated coefficients correspond to mutually orthogonal elements of the dictionary. Coding experiments on natural images show that the new method achieves the same trade-off as matching pursuit between number of coding coefficients and reconstruction error, but significantly faster convergence.

1. Introduction

In many image-coding schemes, an input image is represented as a linear combination of other images, which are elements of a set called the dictionary or codebook. It is often necessary to decompose the input image into as small a number of elements as possible: such a strategy (called atomic decomposition, adaptive representation, or sparse coding) has obvious advantages for storage, transmission, de-noising, and pattern recognition. When the dictionary is over-complete, finding the expansion with the smallest reconstruction error for a given number of (non-zero) coding coefficients is an NP-hard problem [6]. Several iterative algorithms have been proposed for finding a sub-optimal solution (*e.g.* [4, 10, 11, 13]). The main drawback of these algorithms is their computational cost. Since its introduction, matching pursuit [10] has been the standard of comparison for this class of algorithms. This article introduces a new algorithm for atomic decomposition, the *inhibition method*, which can be considered a generalization of matching pursuit and offers faster convergence to a given reconstruction error, while achieving the same sparseness of encoding. In the next Section, matching pursuit and the inhibition method are discussed within the same framework; experimental results are presented in Section 3 followed by the conclusions in Section 4.

2. Description of the algorithms

The image to be encoded is the vector $\mathbf{f}$ of n pixel values; the linear image size is $L = \sqrt{n}$. The dictionary D includes m (normalized) elements $\mathbf{g}_i \in D$, $\|\mathbf{g}_i\| = 1$, $i = 1 \ldots m$. These elements are vectors of n pixel values. Given that the dictionary is overcomplete, $n < m$. $\mathbf{c}$ is a vector of m coding coefficients. The linear expansion $\sum_{i=1}^{m} c_i \mathbf{g}_i$ is the reconstructed image. The difference between input image and reconstructed image $\mathbf{r} = \mathbf{f} - \sum_{i=1}^{m} c_i \mathbf{g}_i$ is the residual image. The squared L_2 norm of $\mathbf{r}$, normalized by the squared norm of $\mathbf{f}$, is the normalized squared reconstruction error $NSE = \|\mathbf{r}\|^2 / \|\mathbf{f}\|^2$, which is used as the criterion of how well the reconstructed image approximates the input image (the *PSNR* measure is linearly related to the logarithm of the *NSE*). $\mathbf{p}$ is the vector of m inner products of the residual with the m dictionary images: $p_i = \mathbf{g}_i^T \cdot \mathbf{r}$, $i = 1 \ldots m$.

When required for clarity, we write $\mathbf{c}(t)$, $\mathbf{r}(t)$, $\mathbf{p}(t)$, where t is the iteration number. The symbol $\Delta\mathbf{c}$ denotes the update of the coefficients at each iteration: $\Delta\mathbf{c}(t) = \mathbf{c}(t) - \mathbf{c}(t-1)$. Similarly, $\Delta\mathbf{r}(t) = \mathbf{r}(t) - \mathbf{r}(t-1)$ and $\Delta\mathbf{p}(t) = \mathbf{p}(t) - \mathbf{p}(t-1)$.

2.1. Basic iteration

Rather than describing matching pursuit and the inhibition method separately, it is easier to introduce a common framework for both methods and highlight the differences within this framework.

After the vector of inner products $\mathbf{p}(0)$ is initialized, the basic iteration consists of two steps: (**I**) updating one or more coding coefficient and (**II**) re-computing the inner products between all the dictionary elements and the new residual image. The coefficients to be updated should be chosen so as to combine a large decrease of *NSE* with a small increase of the number of non-zero coefficients. One possible heuristic is to update only the one coefficient which can bring the largest decrease of *NSE*, *i.e.* the coefficient

0-7695-0750-6/00 $10.00 © 2000 IEEE

corresponding to the largest element of $\mathbf{p}$. This heuristic is the basis of matching pursuit.

To achieve faster convergence, it is desirable to update more than one coefficient per iteration. However, most of the largest inner products are likely to arise from strongly correlated dictionary elements, *i.e.* elements representing the same feature of the input image: updating all of the corresponding coefficients would produce a redundant, rather than sparse, representation. A solution to this problem is to update coefficients corresponding to dictionary elements which have larger (in absolute value) inner products with the residual than any other non-orthogonal dictionary element. This heuristic is the basis of the inhibition method. The name 'inhibition method' is inspired by the analogy with reciprocal inhibition of neurons with non-orthogonal Gabor-like receptive fields in the visual cortex (see *e.g.* [2]).

In detail, the two steps of the iteration are as follows:

I. Select one or more coefficients and update them by adding the values of the corresponding inner products:

$$\Delta c_i(t) = \begin{cases} p_i(t) & \text{if } \mathbf{g}_i \in S(t) \\ 0 & \text{otherwise} \end{cases} \quad (i = 1 \ldots m) \tag{1}$$

The definition of the subset $S \subset D$ is the essential difference between matching pursuit and the inhibition method. In matching pursuit, S is defined by:

$$S_{MP} = \{\mathbf{g}_i \in D \mid \forall j \neq i \; |p_i| > |p_j|\} \tag{2}$$

Obviously, by Eq.2, S_{MP} includes only one element.

Ideally, in the inhibition method, S should be defined by:

$$S_I = \{\mathbf{g}_i \in D \mid \forall j \neq i \; |\mathbf{g}_i^T \cdot \mathbf{g}_j| > \epsilon \Rightarrow |p_i| > |p_j|\} \tag{3}$$

where ϵ is a small constant ($\epsilon = 0.05$ in our experiments). In practice, implementing Eq.3 would require too many comparisons between elements of $\mathbf{p}$. However, comparisons can be restricted to the k largest elements of $\mathbf{p}$ (for $k << m$): by this approximation, the coefficients to be updated correspond to the elements of the set $S_{IM} = S_k \cup S_I$, where S_k is the set of the k dictionary elements having largest inner products with the residual image (in our implementation $k = 2047$). By setting $k = 1$, matching pursuit is obtained as a special case of the inhibition method. A look-up table can be used to determine whether two dictionary elements are orthogonal. By exploiting symmetries in the dictionary, *e.g.* if the elements are shifted, scaled and rotated versions of one or a few kernels, this table requires storage of $O(m)$, rather than $O(m^2)$.

It is easy to see that the inhibition method will converge whenever matching pursuit converges: the largest (in absolute value) element of the vector $\mathbf{p}$ is always selected, therefore at least one coefficient is updated at each iteration.

II. Update the vector of inner products, taking into account the change in the residual.

Step II can be broken down into two sub-steps:

IIa. Update the residual image:

$$\Delta \mathbf{r}(t+1) = -\sum_{i=1}^{m} \Delta c_i(t)\, \mathbf{g}_i \tag{4}$$

If few coefficients have been updated, the computational cost of sub-step IIa is negligible compared to the cost of sub-step IIb. For large numbers of updated coefficients, if the dictionary can be decomposed into shifted versions of a few kernels, then the update of the residual can be efficiently carried out by the application of the FFT. This is the case with our dictionary.

IIb. Re-compute the inner products of the residual image $\mathbf{r}$ with all dictionary elements:

$$\Delta p_i(t+1) = \mathbf{g}_i^T \cdot \Delta \mathbf{r}(t+1) \qquad (i = 1 \ldots m) \tag{5}$$

Again, this convolution can be carried out efficiently by the application of the FFT.

Alternatively, step II can be carried out by updating the inner products directly, using the relation:

$$\Delta p_i(t+1) = -\sum_{j=1}^{m} \Delta c_j \cdot (\mathbf{g}_i^T \cdot \mathbf{g}_j) \qquad (i = 1 \ldots m) \tag{6}$$

If Eq.6 is implemented, then the computational cost of step II becomes proportional to the number of updated coefficients. In this case, updating a number ν of coefficients by the inhibition method in one iteration would require as much computation as updating ν coefficients by matching pursuit in ν iterations. However, this technique is competitive only if most dictionary elements are mutually orthogonal (which is not the case in a highly oversampled dictionary), the inner products between elements can be quickly computed analytically, and Fourier techniques cannot be used to speed up convolutions and deconvolutions (*i.e.* Eq.4 and Eq.5). None of these conditions applies in our case.

2.2. Dictionary

The dictionary used in the coding experiments is composed of two-dimensional real-valued Gabor functions [5] of unit L_2 norm and zero mean (so that the DC component of the input image was encoded separately). The Gabor functions are of the form:

$$g(\mathbf{x}) = \frac{1}{Z} \exp\left(-\frac{1}{2}\mathbf{x}^T \cdot \mathbf{\Sigma}^{-2} \cdot \mathbf{x}\right) \cos\left(\frac{2\pi x}{\lambda} + \varphi\right) \tag{7}$$

$$\mathbf{x} = \begin{pmatrix} x \\ y \end{pmatrix} \qquad \mathbf{\Sigma} = \begin{pmatrix} \sigma_x & 0 \\ 0 & \sigma_y \end{pmatrix}$$

$$x = x_0 + u\cos\Theta - v\sin\Theta$$

$$y = y_0 + u\sin\Theta + v\cos\Theta$$

where Z is a normalizing constant; u and v are image coordinates; x_0, y_0, σ_x and σ_y are parameters of a two-dimensional Gaussian window; λ and φ are parameters of a sinusoidal grating and Θ is the angle between the u axis of the image and the x axis of the Gabor function. In our dictionary, two of the parameters are fixed by the relations: $\sigma_y/\sigma_x = \sqrt{2}$ and $\lambda/\sigma_x = \sqrt{2}\pi$. The remaining scale parameter ranges over 11 different scales [$\lambda = 2 \cdot a^s$ pixels, $a = \sqrt{2}$, $0 \leq s \leq \log_a(L/4)$]. The parameters x_0 and y_0 assume all integer values between 1 and L, *i.e.* the Gabor functions are centered on all image pixels (no downsampling). The other parameters range over 8 different orientations [$\Theta = (q/8)\pi$ radians, $0 \leq q < 8$] and 2 different phases [$\varphi \in \{0, \pi/2\}$], except at the smallest spatial scale ($\lambda = 2$ pixels), for which only Gabor functions in cosine phase ($\varphi = 0$) and with horizontal or vertical orientations [$\Theta \in \{0, \pi/2\}$] are used, to avoid aliasing. The dictionary is thus overcomplete by a factor of $10 \cdot 8 \cdot 2 + 2 = 162$. For images of linear size $L = 128$ pixels, this sampling scheme results in approximately 2.65 million dictionary elements.

Gabor functions with odd symmetry ($\varphi = \pi/2$) have zero mean. The DC component of Gabor functions with even symmetry ($\varphi = 0$) was set equal to zero in the frequency domain.

3. Coding results

The inhibition method was tested on a set of 19 natural images. Fig.1 shows the results obtained with the Lena image: the *NSE* is plotted as a function of both number of iterations and number of coefficients; the two curves are undistinguishable for matching pursuit. As can be seen, the two algorithms offered the same trade-off between *NSE* and number of coding coefficients: the data points (crosses) for the inhibition method lie on the data line for matching pursuit. On the other hand, the number of iterations required to reach a given *NSE* was smaller for the inhibition method than for matching pursuit, except for the first two iterations, in which the inhibition method only updated one coefficient per iteration. As the main low-frequency, large-scale components of the image were removed, it became possible for the inhibition method to detect more and more mutually orthogonal image components simultaneously, so that the number of updated coefficients increased with the iteration number.

Over the entire set of 19 images, the number of iterations needed for the inhibition method to reach $NSE = 0.01$ was 42.1 ± 4.4 (average $\pm$ standard deviation) and the number of coefficients needed to reach $NSE = 0.01$ was 3370 ± 1179

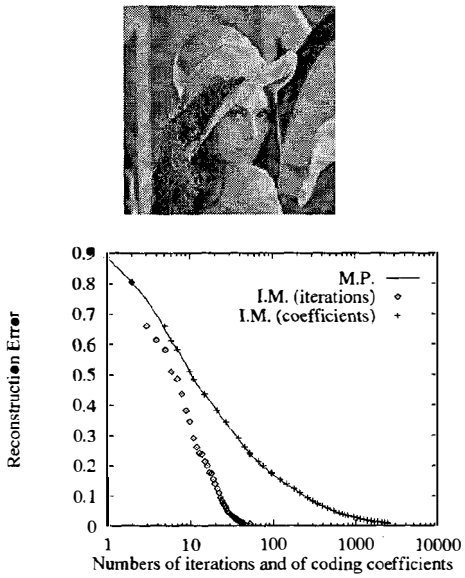
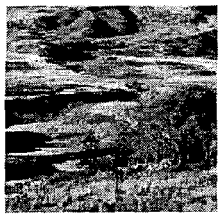


Figure 1. Performance of the two algorithms on the Lena image.

(average $\pm$ standard deviation); in other words, the number of iterations was approximately constant, irrespective of the number of coefficients required to encode the image. The average number of coefficients updated per iteration was 77.0 . Assuming that one iteration of the inhibition method has almost twice the computational cost of one iteration of matching pursuit (because the deconvolution step, Eq.4, has negligible cost in matching pursuit), we conclude that the inhibition method converges about 38 times faster than matching pursuit. This average hides the fact that the inhibition method is not much faster than matching pursuit in the first few iterations, but can be over 100 times faster in the last few iterations.

These estimates are based on the assumption that matching pursuit would encode all the images with about the same number of coefficients as used by the inhibition method. In addition to the Lena image (Fig.1), this assumption was tested on the two images most likely to prove it wrong: the image which the inhibition method encoded with the largest number of coding coefficients; and the image (shown in Fig.2) which was encoded at the fastest rate by the inhibition method: the more coefficients are updated per iteration, the higher the likelihood that redundant coefficients are updated. In both cases, the plots of *NSE vs.* number of coefficients were overlapping, as was the case for Lena.



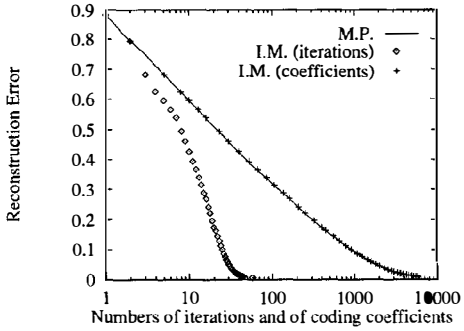


Figure 2. Performance of the two algorithms applied to a landscape.

4. Conclusions

Several auxiliary techniques have been developed to speed up the matching pursuit iteration [1, 12, 3], but most of these techniques work by speeding up the computation of inner products between the dictionary elements and the residual image (step II), *i.e.* reducing the computational cost of one iteration; therefore, they could be combined with the inhibition method to achieve the same result. Only hierarchical parallel matching pursuit [7] is similar to the inhibition method in allowing the updating of several coefficients per iteration, although the method used to select these coefficients is different.

The elements of the vector $\mathbf{p}$ can be interpreted as measures of similarity between the residual image and the elements of the dictionary. Recently, it has been shown that a more sparse encoding can be achieved with a different similarity measure [8]. This different measure can be combined with the inhibition method as well as with matching pursuit, since it consists of a modification of Eq.5.

This paper has presented results on images, because this is an application which puts a heavy computational load on matching pursuit; however, the inhibition method can be applied to one-dimensional signals as well. Atomic decomposition has proved useful in a number of pattern-recognition applications (*e.g.* [9, 14, 15, 16]). The inhibition method, by its faster convergence, allows the practical application of atomic decomposition to new areas.

Acknowledgements: The authors thank Dr. J. Roerdink for helpful comments on an earlier version of this article. Most of the images were obtained from the PICS image archive (http://pics.psych.stir.ac.uk).

References

[1] F. Bergeaud and S. Mallat. Matching pursuit: adaptive representations of images and sounds. *Comp. Appl. Math.*, 15:97–109, 1996.

[2] M. Carandini and D. Heeger. Summation and division by neurons in primate visual cortex. *Science*, 264:1333–1336, 1994.

[3] T.-H. Chao, B. Lau, and W. Miceli. Optical implementation of a matching pursuit for image representation. *Optical Eng.*, 33(7):2303–2309, 1994.

[4] S. Chen and D. Donoho. Examples of basis pursuit. *Proc. SPIE*, 2569(2):564–574, 1995.

[5] J. Daugman. Uncertainty relations for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters. *J. Opt. Soc. Am. A*, 2:1160–1169, 1985.

[6] G. Davis, S. Mallat, and M. Avellaneda. Greedy adaptive approximation. *Journal of Constructive Approximation*, 13:57–98, 1997.

[7] H. Feichtinger, A. Türk, and T. Strohmer. Hierarchical parallel matching pursuit. *Proc. SPIE*, 2302:222–232, 1994.

[8] S. Jaggi, W. Karl, S. Mallat, and A. Willsky. High resolution pursuit for feature extraction. *Applied and Computational Harmonic Analysis*, 1998.

[9] S. Jaggi, W. Karl, S. Mallat, and A. Willsky. Silhouette recognition using high-resolution pursuit. *Pattern Recognition*, 32(5):753–771, 1999.

[10] S. Mallat and Z. Zhang. Matching pursuit with time-frequency dictionaries. *IEEE Trans. Signal Proc.*, 41(12):3397–3415, 1993.

[11] M. Nafie, M. Ali, and A. Tewfik. Optimal subset selection for adaptive signal representation. *Proc. IEEE Int. Conf. Acoust. Speech Signal Proc.*, pages 2511–2514, 1996.

[12] R. Neff and A. Zakhor. Very low bit-rate video coding based on matching pursuits. *IEEE Trans. Circuits and Systems for Video Technology*, 7(1):158–171, 1997.

[13] Y. Pati, R. Rezaiifar, and P. Krishnaprasad. Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition. *Proc. 27th Asilomar Conference on Signals, Systems and Computers*, pages 40–44, 1993.

[14] Y. M. Ro, R. Neff, and A. Zakhor. Matching pursuited data acquisition in MRI. *Proceedings on SPIE, Medical Imaging*, 3202(1):530–540, 1997.

[15] A. Shmilovici and O. Maimon. Application of adaptive matching pursuit to adaptive control of nonlinear dynamic systems. *IEE Proceedings - Control Theory and Application*, 145(6):575–582, November 1998.

[16] H. Yoshida. Matching pursuit with optimally weighted wavelet packets for extraction of microcalcifications in mammograms. *Applied Signal Processing*, 5(3):127–141, 1999.