On Finding Nearest Neighbors in a Set of Compressible Signals

Philippe Jost and Pierre Vandergheynst
Ecole Polytechnique Fédérale de Lausanne (EPFL)
Signal Processing Institute
CH-1015 Lausanne, Switzerland
E-mail: {philippe.jost,pierre.vandergheynst}@epfl.ch

***Abstract*—Numerous applications demand that we manipulate large sets of very high-dimensional signals. A simple yet common example is the problem of finding those signals in a database that are closest to a query. In this paper, we tackle this problem by restricting our attention to a special class of signals that have a sparse approximation over a basis or a redundant dictionary. We take advantage of sparsity to approximate quickly the distance between the query and all elements of the database. In this way, we are able to prune recursively all elements that do not match the query, while providing bounds on the true distance. Validation of this technique on synthetic and real data sets confirms that it could be very well suited to process queries over large databases of compressed signals, avoiding most of the burden of decoding.**

I. Introduction

The tremendous activity in the field of sparse approximation [1], [2], [3] is strongly motivated by the potential of theses techniques for typical tasks in signal processing such as denoising, source separation or compression. Given a signal f in the space of finite d-dimensional discrete signals $\mathbb{R}^d$, the central problem of sparse approximation is the following: compute a good approximation $\tilde{f}_N$ as a linear superposition of N basic elements picked up in a collection of signals $\mathcal{D} = \{\phi_k\}$ that spans the entire signal space, referred to as a dictionary :

$$\tilde{f}_N = \sum_{k=0}^{N-1} c_k \phi_k, \quad \phi_k \in \mathcal{D}\,, \quad \|f - \tilde{f}_N\|_2 \leq \epsilon\,. \qquad (1)$$

The approximant $\tilde{f}_N$ is sparse when $N \ll d$ and is called a sparse representation if additionally $\|f - \tilde{f}_N\|_2 = 0$.

If $\mathcal{D}$ is an orthogonal basis, it is easy to solve (1) for the best approximant. However, for a generic redundant dictionary the problem becomes much more complicated and attempts at solving it have sparked an intense research stream. Schematically, it is by now known that if a representation is sufficiently sparse then it is unique and can be recovered by standard practices such a convex relaxation (Basis Pursuit) [2] or greedy algorithms like Orthogonal Matching Pursuit (OMP) [3]. If we face an approximation problem, then greedy algorithms and convex relaxation techniques can provide provably good approximants, see for example [4], [5] and the references therein.

This paper does not deal with algorithms to compute sparse approximations. In fact we will assume that we are given sparse approximations of signals and we will ignore *how* they have been computed. We will however require that our approximants possess a particular structure. Suppose first that the terms in (1) are re-ordered in decreasing order of magnitude, i.e such that $\mid c_0 \mid \geq \mid c_1 \mid \geq \cdots \geq \mid c_{N-1} \mid$. Strict sparsity requires that the number of non-zero coefficients N be small. However we can slightly relax this definition by asking that the magnitude of the coefficients drops quickly to very small values such that there are only few big coefficients. A signal that is well approximated by such an expansion over a dictionary is termed *compressible*, highlighting the idea that most of the information is contained in few coefficients [6]. Usually, and that is the case in this paper, the sorted coefficients are assumed to follow a power-law decay; the i^{th} largest is such that:

$$\mid c_i \mid \leq \mathrm{C}\ i^{-\gamma}. \qquad (2)$$

for $\gamma \geq 1$ and some positive constant C. The decay parameter γ may depend on both the signal and dictionary.

Dictionaries used to define the class of compressible signals need not be redundant. Piece-wise smooth signals for example are compressible on wavelet bases and that characteristic is at the heart of the good performances of wavelets for compression or denoising [7]. In many cases however, a simple basis cannot efficiently capture signal characteristics within a compressible representation. Dictionaries offer more flexibility and dramatically enlarge the class of compressible signals. However, when using redundant dictionaries, a decomposition is no longer unique; in addition, finding the best possible representation is a daunting task and it was long believed that it could prevent these techniques from being used in applications. Recent research shows that this bottleneck is not far from being broken. The MPTK toolkit [8] provides a very fast implementation of matching pursuit and efficient implementations of techniques for solving large scale linear or quadratic programs have recently emerged [9]. Working on the algorithmic side can also provide smarter ways to achieve sparse decompositions [10], [11]. Using redundant systems gives more freedom for the design of the dictionary, which can then be efficiently tailored, or even learned from training data, to closely match signal structures [12], [13], [14], [15], [16], [17].

With the advent of digital cameras and portable music players, modern digital signal processing has also to face the challenge of voluminous databases of signals. Clearly, signal processing algorithms must be adapted to problems where each user manipulates large collections of signals. Finding the nearest neighbor in a database is fundamental for many applications; [18] presents a good overview of this field. Generally, when the data is lying in a high dimensional space, a dimensionality reduction step is used to lower the complexity of the query. In the field of signal processing, the dimensionality reduction resulting from the sparsity of an approximation has been exploited for different tasks such as analysis, de-noising or compression. Roughly speaking, the sparser the representation, the better it is for applications. In this paper, we explore how sparsity can be used to handle huge amount of data at a lower cost. More precisely, we tackle the problem of computing in an efficient manner the correlation of a single query signal with a huge set of compressible signals. Our algorithm uses only the components c_k, ϕ_k of the signal model (1), hence can be seen as working in the *transform domain*. Since compression is key in storing large collections of signals, we thus potentially avoid the extra burden of having to decode large amounts of data for searching or browsing through the database.

In section III-A we derive the scalar product of two signals according to the parameters of their respective sparse representation. It allows to bound parts of the scalar product when dealing with *compressible* signals. In section III-B we present an algorithm to compute efficiently the projection of a signal on a set of signals. The algorithm uses the bound previously introduced. Section III-C and IV present improved versions of the simplest bound. Section V presents different experiments to illustrate the different bounds as well as the algorithm itself. We conclude in section VI on the benefits of this new approach and list the perspectives we will consider.

II. Prior art

Before moving on to the core of this paper, let us briefly describe how our contributions can be compared with existing techniques. The field of nearest-neighbor algorithms is very wide and still extremely active, we thus certainly couldn't hope to provide here a fair survey. However, we would like to highlight some key results and orientations and this will also allow us to specify constraints used in our framework.

Finding the nearest neighbor of a query in a set $\mathcal{F}$ of I d-dimensional vectors can be solved by brute force with $\mathcal{O}(dI)$ operations. Clearly, when I is big (and that is the case in most applications), this could be prohibitive. A lot of work has been devoted to trying to reduce the amount of computations needed to deal with large data sets. Most of the recent approaches have a cost scaling like $\mathcal{O}(\exp d \log I)$ provided the data base is first pre-processed to create an efficient data structure [19], [20], [21], [22]. It has to be noted that a computational cost exponential in d does not improve on the brute force technique when d is large enough, i.e $d > \log I$, highlighting the so called *curse of dimensionality*. Various algorithms have been proposed to solve this problem, with complexity that roughly scales in $\mathcal{O}(d^\beta \text{polylog}(dI))$, for some $\beta > 1$ and an appropriate data structure [18], [23], [24].

This short survey brings us to our main constraint. In this paper, we target applications in user centric multimedia databases, i.e images, audio that reside on the user's computer, and in this setting we cannot afford large preprocessing time. More particularly we must be able to add and remove entries in the database at no cost. We don't extract low dimensional feature vectors from our signals. Instead we use sparse representations both for compression and description of the data. Our data structure is thus simple and forced upon us: the description of each item in terms of the coefficients and atoms' indexes in (1). As for how sparsity N depends on the dimension d, it is hard to give a precise rule. Though N is much smaller than d, we will assume that is scales linearly with d. We thus have high-dimensional vectors in our database.

III. A simple deterministic algorithm

A. Notations and warm-up

For the sake of generality, we will work from now on with a dictionary, i.e a collection of unit L^2 norm atoms $\phi_i \in \mathbb{R}^d$, $i = 1, ..., L$, with L possibly much larger than d. The dictionary is often represented as a matrix Φ whose columns are the atoms ϕ_i. Using this formalism, $G = \Phi^* \Phi$ is the Gram matrix of the dictionary and contains all possible scalar products between atoms, i.e. $G_{ij} = \langle \phi_i \mid \phi_j \rangle$.

Let us consider a set of compressible signals $\mathcal{F} = \{f^i\}_{i=i}^{I}$:

$$f^i = \sum_{j=1}^{N} c_j^i \phi_{k_j^i}. \tag{3}$$

where $\phi_{k_i} \in \mathcal{D}$. The vector $\mathbf{k^i}$ contains the indices of the atoms in the dictionary and is such that the projections c_j^i are in decreasing order of magnitude. Note that we have voluntarily discarded the N-term approximation error in (3), and we will keep on doing so from now on. We will discuss later the influence of this term.

The aim of this paper is to provide an efficient method to find, in the set of signals $\mathcal{F}$, the one that is closest to a query signal $g = \sum_{l=1}^{N_g} b_l \phi_{k_l}$ that is also compressible with N_g terms. The magnitudes of the projections are also decreasing with l. The scalar product $\langle f^i \mid g \rangle$ between a signal from the set and the new one can be written as follows:

$$\begin{aligned} \langle f^i \mid g \rangle &= \langle \sum_{j=1}^{N} c_j^i \phi_{k_j^i} \mid \sum_{l=1}^{N_g} b_l \phi_{k_l} \rangle, \\ &= \sum_{j=1}^{N} \sum_{l=1}^{N_g} c_j^i b_l G_{k_j^i, k_l}. \end{aligned} \tag{4}$$

where $G_{k_j^i, k_l} = \langle \phi_{k_j^i} \mid \phi_{k_l} \rangle$ is an entry of the Gram matrix of the dictionary $\mathcal{D}$.

The aim of the algorithm is to exploit sparsity, i.e N_g, $N \ll d$, in order to find the best matching signal. It is done by eliminating rapidly the signals whose scalar products with the query is too small. To do so, we rewrite the scalar product presented in eq. (4) as follows:

$$\langle f^i \mid g \rangle = \sum_{k=2}^{N+N_g} s_k^i, \tag{5}$$

where s^i_k represents the part of the scalar product coming from atoms participating in both decompositions such that the sum of j and l is equal to k. For the i^{th} signal of the set $\mathcal{F}$, it corresponds to:

$$s^i_k = \sum_{\substack{j,l\\ j+l=k\\ j\le N, l\le N_g}} c_{k^i_j} b_{k_l} G_{k^i_j,k_l}. \tag{6}$$

The signals of the set $\mathcal{F}$ and the query g are compressible. According to eq. (2), there exists γ and a constant C such that $| c^i_j | \le Cj^{-\gamma}$ and $| b_l | \le Cl^{-\gamma}$. Since the entries of the Gram matrix are between -1 and 1, it is possible to bound the magnitude of s^i_k as follows:

$$| s^i_k | \le \sum_{\substack{j,l\\ j+l=k\\ j\le N, l\le N_g}} C^2 j^{-\gamma} l^{-\gamma} \tag{7}$$

B. Iterative candidate rejection

Computing the scalar product of two discrete d-dimensional signals requires d multiplications and $d-1$ additions. Let us now suppose that these signals have an exact-sparse representation using respectively N and N_g terms and that the entries of the gram matrix G have been pre-computed. Computing the scalar product using eq. (4) needs $3N*N_g$ multiplications and $N*N_g-1$ additions. If $N=N_g$, there is a computational gain only if $N<\sqrt{\frac{d}{2}}$, so for very sparse signals. However, using the formalism described in the previous section, it is possible to compute iteratively the scalar product of two signals that have a sparse and compressible representation. When searching for the best matching signal in a huge set $\mathcal{F}$, it is of great interest to be able to eliminate in an early stage the signals that have no chance to match. If the scalar product is computed in an iterative way, our aim is to eliminate signals by estimating at each step an upper and a lower bound on the final scalar product, which is possible by using the bound presented by eq. (7). To do so, let us first define:

$$S^i_K = \sum_{k=2}^{K} s^i_k, \tag{8}$$

which represents the part of the scalar product $\langle f_i \mid g\rangle$ found by taking into account the atoms whose sum of indices is smaller or equal to K. Using the same formalism, it is possible to express the missing part of the scalar product:

$$R^i_K = \sum_{k=K+1}^{N+N_g} s^i_k. \tag{9}$$

If we had kept track of the approximation error in our initial model (3), we would have to add it to this residual. We simply assume that this error is sufficiently smaller than the typical values of R^i_K we will be working with. If the signals are well-compressible, this will be the case and this is indeed what our simulations suggest. Using the two preceding equations, let us express the scalar product as $\langle f^i \mid g\rangle = S^i_K + R^i_K$, $\forall K$, $2\le K\le N+N_g$. The value of S^i_K can be computed iteratively as $S^i_K = S^i_{K-1} + s^i_K$.

When looking for the signal that is most correlated with the query, one computes the absolute value of the scalar product, disregarding the sign of the projection. Thus, $\forall K$, $2\le K\le N+N_g$ the following relation holds:

$$| S^i_K | - | R^i_K | \le | \langle f_i \mid g\rangle | \le | S^i_K | + | R^i_K | . \tag{10}$$

Using eq. (7), it is possible to upper bound the residual part of the correlation $| R^i_K |$.

$$| R^i_K | \le \sum_{k=K+1}^{N+N_g} | s^i_k | \le C^2 \sum_{k=K+1}^{N+N_g} c_{k,N,N_g} (\frac{k^2}{4})^{-\gamma} = \tilde{R^i_K}, \tag{11}$$

where c_{k,N,N_g} is the number of possible products between atoms such that the sum of their indices is equal to k and knowing that we have N terms for f_i and N_g terms for the query:

$$c_{k,N,N_g} = \begin{cases} k-1 & \text{if } 2\le k\le N_g+1; \\ N_g & \text{if } N_g+1<k\le N; \\ N+N_g-k+1 & \text{if } N<k\le N+N_g; \\ 0 & \text{otherwise.} \end{cases} \tag{12}$$

Using the bound $\tilde{R^i_K}$ defined in eq. (11) it is possible to upper and lower bound the correlation at any iteration K as follows :

$$m^i_K \le | \langle f_i \mid g\rangle | \le M^i_K, \tag{13}$$

where $m^i_K =| S^i_K | - \tilde{R^i_K}$ and $M^i_K =| S^i_K | + \tilde{R^i_K}$ are respectively the lower and the upper bound. It is obvious that if $\exists K$ s.t. $M^j_K < m^i_K$ then $| \langle f_i \mid g\rangle | > | \langle f_j \mid g\rangle |$. This principle is illustrated by fig. 1 where the maximal value some candidates could eventually reach is lower than the worst case of the best matching candidate. The pseudo-code illustrating the proposed algorithm is presented by table 1.

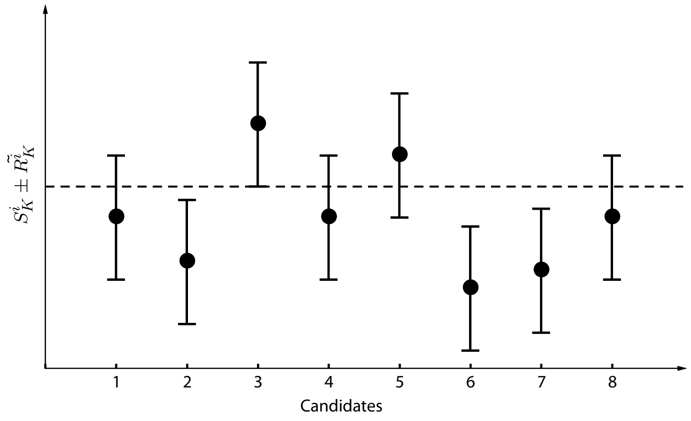

Fig. 1. Elimination of candidates 2,6 and 7.

Algorithm 1 Find best matching signal in a database of exact-sparse N-terms signals

```
**INPUTS**: A signal $g = \sum_{i=1}^{N_g} a_i \phi_i$, $g_i \in \mathcal{D}$.
A set of signals $\mathcal{F}$ having a exact-sparse representation using $N$ terms.
The Gram matrix $G$ of the dictionary used to represent the signals.
**OUTPUT**: $min_i \mid \langle f_i \mid g \rangle \mid$, the index of the signal that best matches $g$.
**INITIALIZATION**: $P = \{i\}_{i=1}^{\|\mathcal{F}\|}$ the indices of the signals in the set $\mathcal{F}$.
$K = 2$.
$S_1^i = 0$, $\forall i$.
**while** card$(P) > 1$ **do**
  Compute all $S_K^i = S_{K-1}^i + s_K^i$.
  Compute all $m_K^i$ and $M_K^i$.
  $S = \{f_i\}_{M_K^i < \max_i m_K^i}$
  $P = P \setminus S$.
  $K = K + 1$.
**end while**
```

C. Improved bounds

The absolute value of the projection is not decreasing at uniform rate. The slope is much steeper at the beginning than at the end. This behavior is closely related to the redundancy of the dictionary. Suppose now that the slope is bounded as $\mid a_i \mid \leq C i^{-\gamma_i}$ i.e. for each projection a_i, we learn a local parameter instead of using a global constant value for γ. It is now possible to bound $\mid s_k^i \mid$ as follows:

$$\mid s_k^i \mid \leq \sum_{\substack{i,j \\ i+j=k \\ i \leq N, j \leq N_g}} C^2 i^{-\gamma_i} j^{-\gamma_j}. \tag{14}$$

One reason for locally adjusting the decay parameter γ is that greedy algorithms for example tend to produce compressible approximants that have a non-monotonic decay : usually γ is much bigger for the first few coefficients and then tends to decrease slowly.

Let $\{f_n\}$ be a huge collection of signals having an exact-sparse representation using N terms. The projection of the i^{th} atom of the decomposition of f_j is denoted by a_i^j. Let also have the maximal i^{th} projection $A_i = \max_j \mid a_i^j \mid$. The best parameters C and γ used previously can be found by solving:

$$\{C, \gamma\} = \operatorname*{argmin}_{C,\gamma} \sum_{i=1}^{N} (A_i - C i^{-\gamma})^2, C i^{-\gamma} \geq A_i, \forall i. \tag{15}$$

Using the C we have learnt, we define the values γ_i for the improved bound as follows :

$$\gamma_i = -\log_i \frac{A_i}{C}. \tag{16}$$

Using the locally optimized values better matches the behavior of the absolute values of the projections. The bound is used in the algorithm to eliminate the worst matching candidates. Thus, improving the quality of this bound directly acts on the quality of the results of the proposed algorithm. It has to be noticed that the values of the bound do not depend on which signal the algorithm is working with and it can thus be computed offline.

IV. A PROBABILISTIC APPROACH

The worst case bounds presented in the previous sections are based on the hypothesis that the signs of the scalar products between atoms conspire against us and the entries of the Gram matrix of the dictionary are always equal to 1. These worst case hypotheses are far from typical cases. It is straightforward to see that if the signs are positive or negative with equal probability, then the sequences s_k^i (for sufficiently big values of k) would be zero mean. Results in the field of concentration of measure show that functions defined on a large probability space most of the time take values that do not fall too far away from the average case. We will now use these techniques to obtain much sharper bounds for R_K^i. We will have to accept that these bounds hold with high probability and not with absolute certainty.

Without loss of generality, we assume that the coefficients are always positive. The dictionary could simply be augmented to contain also all *opposite* atoms so that this property holds. At each step of the algorithm, we estimate the following amplitude:

$$\begin{aligned} \mid R_K^i \mid &= \mid \sum_{k=K+1}^{2N} s_k^i \mid & (17) \\ &= \mid \sum_{\substack{j,l \\ j+l \geq K+1 \\ j \leq N, l \leq N}} c_j^i b_l G_{k_j^i, k_l} \mid . & (18) \end{aligned}$$

First, let suppose that the *worst* case is met for the values of the Gram matrix but that the corresponding signs are random, $+1$ or -1 with probability $\frac{1}{2}$. From these considerations, we rewrite the previous equation as follows:

$$\begin{aligned} \mid R_K^i \mid &= \mid \sum_{k=K+1}^{2N} s_k^i \mid & (19) \\ &= \mid \sum_{\substack{j,l \\ j+l \geq K+1 \\ j \leq N, l \leq N}} \epsilon_{j,l} c_j^i b_l \mid & (20) \\ &= \mid \sum_n \epsilon_n a_n \mid . & (21) \end{aligned}$$

where $\sum_n \epsilon_n$ is a Rademacher sequence i.e. ϵ_i is $+1$ or -1 with equal probability.

Theorem 1: *Let* $\mathbf{a}$ *be a real vector and* ϵ *a Rademacher sequence. Then* $\forall t > 0$

$$P(\mid \sum_n \epsilon_n a_n \mid > t) \leq 2 e^{-\frac{1}{2} t^2 / \|\mathbf{a}\|_2^2}. \tag{22}$$

The proof of Theorem 1 can be found in chapter 4 of [25]. Using this theorem, it is straightforward to see that

$$P(| \sum_n \epsilon_n a_n |\leq t) \geq 1 - 2e^{-\frac{1}{2}t^2/\|\mathbf{a}\|_2^2}. \tag{23}$$

Since the magnitude of the coefficients are bounded, the entries of $\mathbf{a}$ are also bounded and this gives us a simple upper bound of its l_2-norm. It is then easy to find an upper bound $\tilde{R_K^i}$ for a given probability p by solving $1 - 2e^{-\frac{1}{2}(\tilde{R_K^i})^2/\|\mathbf{a}\|_2^2} = p$. This bound will be discussed in our experiments (Section V) for different values of p. Note that R_K^i is influenced in a unfavorable way by the l_2-norm of $\mathbf{a}$. On the other hand, this reasoning remains general enough to be valid for any dictionary $\mathcal{D}$; in particular for orthogonal bases.

The hypothesis that the vector containing the coefficients signs is a Rademacher sequence is certainly more reasonable than the worst case. However, we still have a levee to pull. So far indeed, we supposed that the entries of the Gram matrix are always 1 and that is very far from reality. We will now investigate how much we can squeeze from a probabilistic model, assuming that the entries of the Gram matrix are drawn at random and that the projections are always positive as before. The residual R_K^i can be rewritten as follows:

$$\begin{aligned} | R_K^i | &= | \sum_{k=K+1}^{2N} s_k^i | && (24) \\ &= | \sum_n G_n a_n | \,. && (25) \end{aligned}$$

where G_n are chosen as independent random variables modelling the entries of the Gram matrix of the dictionary. Let us now turn to the choice of these variables. By definition, all values of the Gram matrix are in the interval $[-1, 1]$. It is hard to deduce any general rule. For redundant dictionaries used in practice an histogram of the entries of the Gram matrix has a peak around zero signaling weakly correlated atoms, and a dirac at one representing the values on the diagonal. Without more exploitable structure, we choose the simplest random variables, i.e uniform random variables. This choice will lead to pessimistic bounds, but note that since (25) models R_K^i as a large sum of these variables we will be able to easily evaluate and control the resulting distribution.

Let $U(n)$ be a symmetric uniform random variable. Its associated probability density function is:

$$f_{U(n)}(x) = \begin{cases} \frac{1}{2n} & \text{if } -n \leq x \leq n; \\ 0 & \text{otherwise.} \end{cases} \tag{26}$$

Eq. (25) can be rewritten as a sum of not identically distributed independent uniform random variables:

$$\begin{aligned} | R_K^i | &= | \sum_n a_n U_n(1) | && (27) \\ &= | \sum_n U_n(a_n) | && (28) \end{aligned}$$

The probability distribution of uniformly distributed random variables with different symmetrical ranges is presented in [26]. For a more general approach and a survey of different results, please refer to [27]. The probability density function of a sum of uniform random variables $\sum_{n=1}^m U_n(a_n)$ is given by Theorem 1 of [27] and in [26], the cumulative density function is given as follows:

$$\begin{aligned} F_m(x) &= \sum_{\overrightarrow{\epsilon}\in\{0,1\}^m} (-1)^{\sum_{i=1}^m \epsilon_i} \Big(\frac{1}{2}(x + \sum_{i=1}^m a_i - \prod_{i=1}^m \epsilon_i a_i)\Big)^m \\ & \qquad / \quad m! \prod_{i=1}^m a_i. \end{aligned} \tag{29}$$

$F_m(x)$ is combinatorial and becomes difficult to evaluate in presence of a large number m of random variables. An asymptotic density distribution $\hat{f}_m(x)$ exists and is presented in [26]. For small values of K, the number m of uniform random variables in the sum is large and represents a typical case for using this approximation :

$$\hat{f}_m(x) \approx \left[\frac{3}{2\pi \sum_{i=1}^m a_i^2}\right]^{\frac{1}{2}} \exp\left[-\frac{3}{2}\frac{x^2}{\sum_{i=1}^m a_i^2}\right]. \tag{30}$$

Since this asymptotic probability density function is gaussian, the associated asymptotic cumulative density function is a simple error function :

$$\hat{F}_m(x) \approx \frac{1}{2}\Big(1 + \mathrm{Erf}\big(\sqrt{\frac{3}{2}}\frac{x}{\|\mathbf{a}\|_2}\big)\Big). \tag{31}$$

Given a probability parameter p, we simply have to find the bound $\tilde{R}_K^i$ such that $F_m(\tilde{R}_K^i) - F_m(-\tilde{R}_K^i) = p$.

V. Experiments

In this section, we evaluate our rejection techniques on simulated and real data. First, we consider the aspects related to the estimation of the different bounds. Second, we show their influence on the behavior of the algorithm. Finally, we present two experiments of finding the closest image in a set of compressible images.

In order to illustrate the different properties and behaviors of the algorithm, we constructed a simple synthetic dataset as follows. A set of 7200 images from the COIL-100 database [28] normalized to have the same energy have been approximated using matching pursuit with a dictionary made of anisotropic atoms [29]. This yields compressible approximants, with nicely decaying projection coefficients. We then synthesized signals by multiplying these coefficients with atoms randomly selected in a dictionary made of the union of the Haar and DCT bases. The choice to take projections values coming from real approximations is motivated by the fact that the bounds mostly rely on them. Our experiments will thus be representative of the behavior of the algorithm in real cases.

The non probabilistic upper bounds do not depend on the dictionary that is chosen but only on the projections, on the number of terms N and N_g of the sparse representations and on the amount of atoms that have already been taken into account by the algorithm. Figure 2 presents the values γ_i bounding the amplitudes of the projections. These values have been learnt from the projections and this figure illustrates well

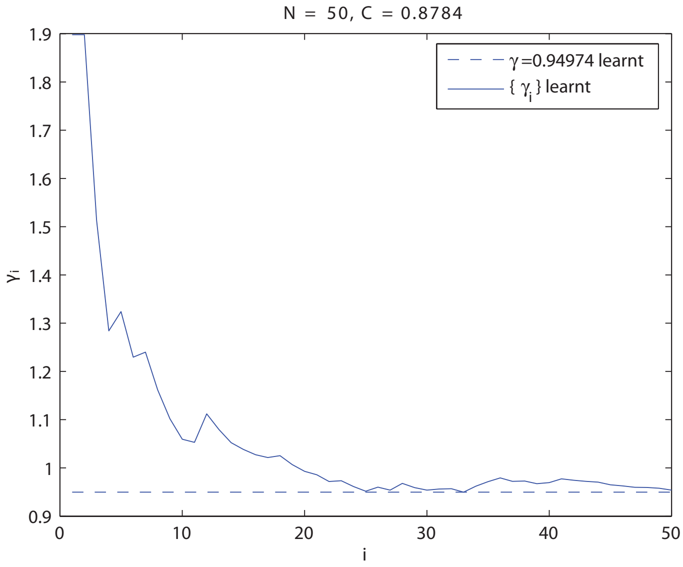

Fig. 2. Locally optimized $\{\gamma_i\}$ compared to overall learnt γ.

the fact that the amplitude of the projections is not decreasing at uniform rate. The slope is steeper at the beginning than at the end.

Figure 3 exhibits the values of the different bounds presented in this paper. The simple bound is far from reality especially for small values of K whilst the bound using the learnt values γ_i (shown in figure 2) is almost half. This fact is illustrated by the upper plot. The two other plots present probabilistic approaches. In the middle, the signs come from a Rademacher sequence whilst on the lower part, the entries of the Gram matrix are modelled as uniform random variables. Varying the value of the parameter p leads to different bounds. For both probabilistic models, when $p = 1$ the obtained curve would be the bound obtained using the learnt values γ_i on the upper part of the figure. Using small values for p leads to sharper bounds at the cost of increasing the probability that the algorithm commits errors whilst still guaranteeing that $P(\text{error}) \leq 1 - p$. Notice that, even when p is large (i.e $p = 0.9$), the associated bound is an order of magnitude smaller than in the deterministic case.

The computational gain of the algorithm greatly depends on its ability to eliminate at early stages non suitable candidates. Figure 4 illustrates the relation between the number of potential candidates and the different bounds. Most of the energy of the signals is caught by the first terms of the decompositions. Signals whose first atoms have low correlation with the first atoms of the query signal are not likely to be the best ones and should rapidly be eliminated by the algorithm. This favorable case happens if the bound is tight enough. The upper part of figure 4 presents the evolution of the cardinality of the set of potential candidates during the execution of the algorithm for the simple bound and the one based on the learnt values γ_i. For these bounds, the elimination of non suitable candidates happens late during the execution of the algorithm. Clearly, the probabilistic bounds are much tighter and this has a direct influence on the behavior of the algorithm. Indeed, one can observe on the other plots of figure 4 that the cardinality of the set of potential candidates is decreasing very quickly in earlier stages. The choice of the value of p has a major influence on the behavior of the algorithm and on its ability to eliminate non suitable candidates given a query signal.

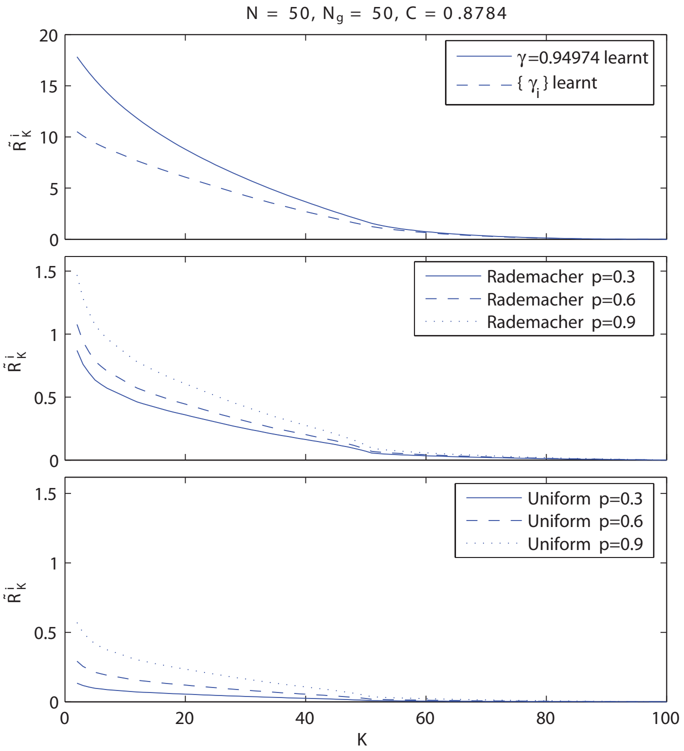

Fig. 3. The simplest bound is greatly improved by using a better upper bound for the energy (up). The signs modelled as a Rademacher sequence (middle). The entries of the Gram matrix modelled by a uniform random variable (down).

At step K of the algorithm, the computational complexity depends on the number signals in the set of potential candidates. The evolution of the cardinality of this set is closely linked to the chosen bound. By extension, the overall complexity of the algorithm also depends on the chosen bound. When using a probabilistic approach, it is possible that the algorithm commits mistakes and eliminates the best matching signal. Note though that our models are *pessimistic* and the true probability of error is smaller than the corresponding $1-p$. Figure 5 (middle-top) shows the relationship between the probability parameter p of the used probability model and the computed probability of error. The computations were done for p going from 0.01 to 0.99 by steps of 0.01. According to the model, the probability of error should be lower than $1 - p$ which is the case for all experiments. It has to be noticed that the *real* probability of error is much lower than the theoretic bound of the model. For example, the algorithm committed no error for values of p bigger than 0.43. However, we can not derive any general rule as these results depends on the characteristics of the dictionary.

During its execution, the algorithm considers successively pairs of atoms. For each of them, it has to multiply the corresponding entry of the gram matrix by both projection parameters and add it to the current estimation of the scalar product. We considered that each of these individual operation

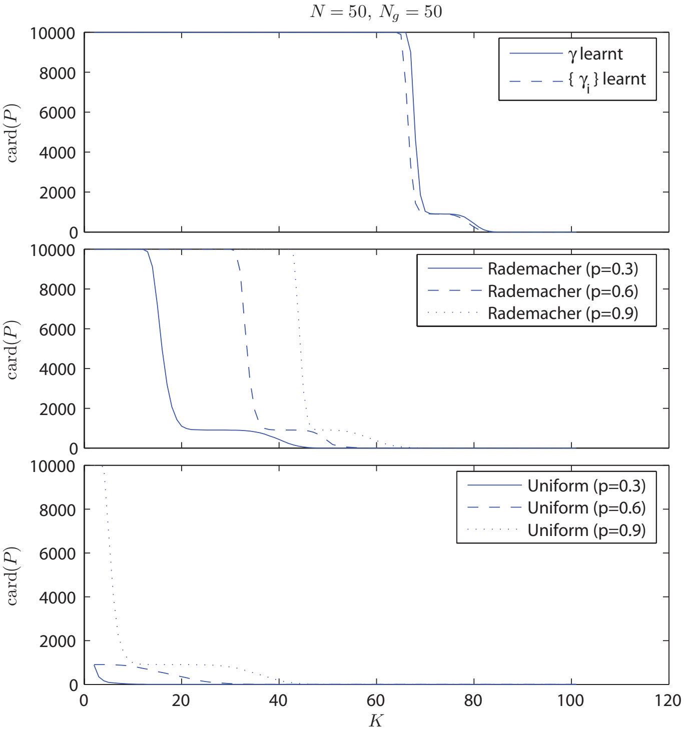

Fig. 4. A database of 3000 signals having 2000 samples is used. The three parts of the figure present the cardinality card(P) of the set of potential candidates for the different bounds defined in the paper averaged over 20000 queries.

has unit weight. Thus, for each pair, the algorithm makes 3 operations. We did not take into account the computations needed to find the candidates to eliminate. Figure 5 (top) presents the number of operations needed for different values of p. The direct computation of the scalar product would require 2000 multiplications and 1999 addition for each signal in the database. Thus, the overall number of operations is approximatively 40×10^6.

When computing nearest neighbors in high dimensional spaces, one often seeks for a $(1+\epsilon)$-approximate nearest neighbor $f_i \in \mathcal{F}$ of g such that $\forall f_k \in \mathcal{F}$, $\text{dist}(f_i, g) \leq (1+\epsilon)\text{dist}(f_k, g)$. Where dist() is some distance function or some norm. Our algorithm computes scalar products between signals, we have thus used the simplest distance:

$$\text{dist}(f_i, g) = \left(1 - \left(\frac{|\langle f_i, g\rangle|}{\|g_i\|_2\|g\|_2}\right)^2\right)^{1/2}. \tag{32}$$

Since all signals have approximatively unit norm, $\text{dist}(f_i, g) \approx \sqrt{1 - |\langle f_i, g\rangle|^2}$. Note that one could use more complicated distances, for example kernel distances, provided they are still based on evaluating scalar products. The traditional kernels used in classification are based on scalar products between vectors. Some of them are listed below, but the interested reader can get more insight in any textbook, for example [30] :

$$\begin{aligned}
\text{Polynomial kernel:} \quad & k(f_i, g) = && \langle f_i, g\rangle^\alpha \\
\text{Radial basis function:} \quad & k(f_i, g) = && \exp - \frac{\|f_i - g\|^2}{2\sigma^2} \\
\text{Sigmoid:} \quad & k(f_i, g) = && \tanh\left(\kappa\langle f_i, g\rangle + c\right)
\end{aligned}$$

Choosing the best distance, though, is application dependent and out of the scope of the present paper. We will thus perform all our experiments using the simple euclidean distance. Note as well that some classification schemes, most notably support vector machines [30], rely heavily on selecting particular data points based on their scalar products with reference vectors. These techniques could potentially benefit from fast rejection algorithms.

For each experiment, we computed the value ϵ, called slackness. The two lower parts of figure 5 presents the mean and the maximal values of the slackness. The mean slackness is very near 0; this is due to the fact that when the algorithm correctly identifies the best signal, the corresponding ϵ is 0. Generally, one is more interested in the maximal possible value for ϵ as it bounds the worst case. For small values of p, the maximal value of the slackness is quiet high whilst it decreases rapidly when p increases.

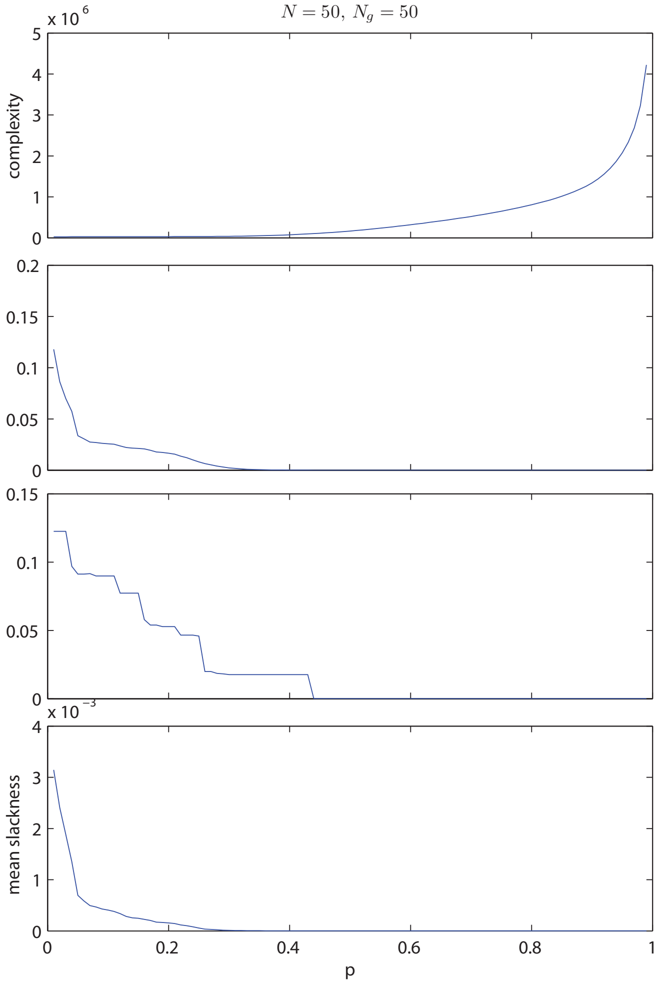

Fig. 5. Comparing the value of the parameter p with the real probability error (middle up) and its influence on the complexity (up). The maximal slackness (middle down) and the mean slackness (down) are also presented. The algorithm uses a global probabilistic bound; the signals have 2000 samples, the database contains 10000 signals. The results are the mean of 20000 experiments.

Since no conditions were imposed on the dictionary apart from the fact that the signals should be compressible, it is straightforward to see that the bound using the Rademacher sum is valid whatever dictionary is used. In the present experiment, a set of 7200 images from the COIL-100 database [28] where approximated with 200 terms of a wavelet decomposition (i.e the dictionary is an orthogonal basis). The images are of size 128×128 and the filter used for the wavelet transform is a Daubechies of length 20. The database of signals is made of 2500 randomly chosen images and the other ones where used to test the algorithm. Figure 6 shows that the cardinality of the set of potential signals decays quickly with the number of iterations. Different parameters p have been used to obtain the bound, but this didn't change significantly the behaviour of the algorithm. Moreover, it has to be noticed that the algorithm always found the best signal in the database.

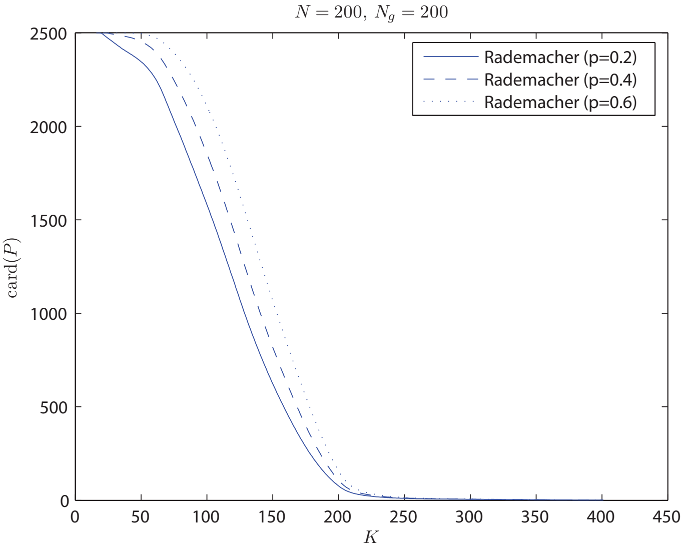

Fig. 6. Cardinality of the set of potential candidates during the execution of the algorithm for images approximated with wavelets.

For our next set of experiments, we turn to a redundant dictionary. Let the set $\mathcal{F}$ be a collection of images having a sparse decomposition using a dictionary $\mathcal{D}$. We used images from the ORL face database [31].The dictionary is built using generating functions that are scaled, rotated and translated [29]. The generating function is made of a Gaussian in one direction and its second derivative in the other direction. It has a good ability to capture edges and is well located in space and frequency.

In our experiments, the atoms have translation parameters that take any positive integer value smaller than the size of the image. The rotation parameter varies by increments of $\frac{\pi}{18}$. The scaling parameters are uniformly distributed on a logarithmic scale from one up to an eighth of the size of the image, with a resolution of one third of octave. The scaling along the second derivative part is always smaller. The dictionary also contains Gaussian atoms. Their translation parameters can take the same values as for the anisotropic atoms, their scaling is isotropic and varies from $\frac{1}{32}$ to $\frac{1}{4}$ of the size of the image on a logarithmic scale with a resolution of one third of octave. Due to isotropy, rotation is obviously useless for this kind of atoms.

An atom is uniquely defined by the set of parameters defining the generating function, the translation, the scaling and the rotation. Using images of size 128×128, the dictionary is made of $40'271'872$ atoms, which makes it difficult to store the Gram matrix. However, due to their particular analytical form, it is possible to compute at low cost any entry of the gram matrix knowing the parameters of the atoms without having to create them.

Figure 7 presents the last steps of the execution of the algorithm when one or more candidates are eliminated. The candidates are shown using the number of atoms at disposal at the current step. The cardinality of the set of face images is 300; at step $K = 17$, corresponding to the second row of the figure, 9 candidates are remaining. The images representing the same person are likely to have a strong correlation with the reference image and are massively present in the remaining candidates.

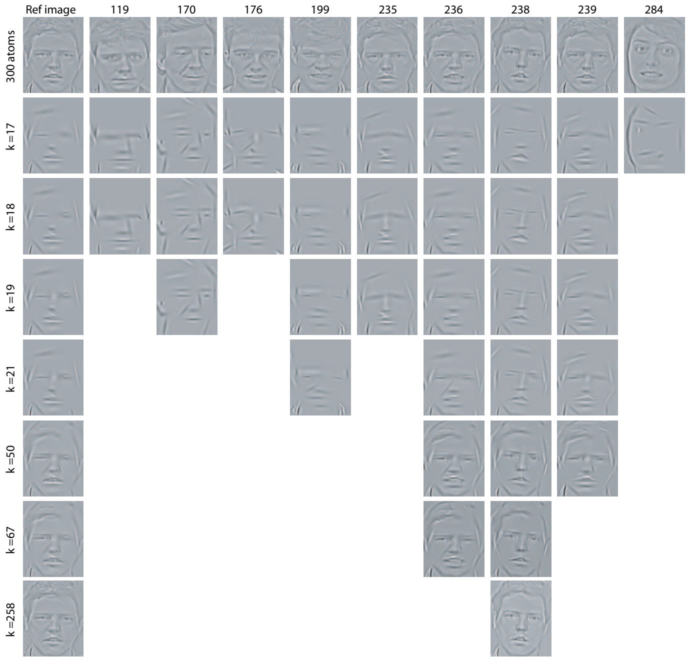

Fig. 7. The column most left presents the reference images whilst the other contain the candidates. The first row contains all fully recreated images whilst in the other rows, the images are reconstructed according to the number of atoms the algorithm takes into account.

The same experiment has been done with wavelet representations of images from the COIL-100 database [28]. A simple pretreatment consisting in normalizing the energy of the images has been done. The database contained 1500 images chosen randomly and all the images of size 128x128 were approximated using 1000 terms. Figure 9 presents the evolution of the cardinality of the set of potential candidates during the first 85 steps. The following steps are presented by figure 8. The first row presents the query image and the images present in P after 85 steps reconstructed using 1000 wavelets. In the next rows, the images are reconstructed using only the wavelets that have been taken into account by the algorithm at this step. The algorithm is efficient in eliminating signals that are not from the *good* class. The four last rows contain the same object and as they are very similar, the algorithm need many steps to identify the best one. However, as the cardinality of the set of potential candidates is very low, the complexity is low too.

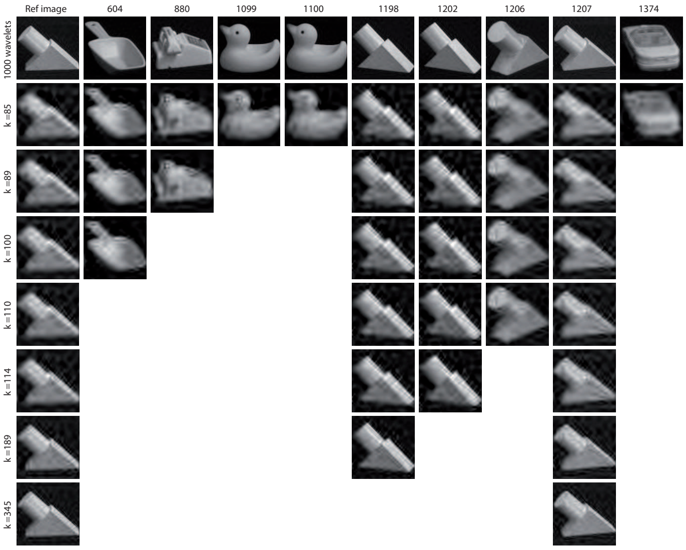

Fig. 8. The column most left presents the reference images whilst the other contain the candidates. The first row contains all fully recreated images whilst in the other rows, the images are reconstructed according to the number of wavelets the algorithm takes into account.

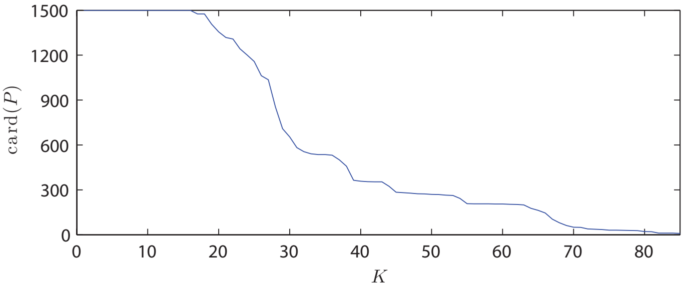

Fig. 9. Evolution of the cardinality of the set of potential candidates before reaching the state shown in Figure 8.

VI. Conclusion

The sparse structure of compressible signals offers a rather straightforward way to reduce the dimensionality of complex signals. In this paper, we have exploited this structure to recursively localize those elements of a large set of signals that are closest to a fixed query. Our technique requires fundamental inputs. First, the coefficients of the expansion of each signal in the database must be stored and easily accessible. Note this is not a particularly stringent requirement since it is very likely that one would store compressed signals, using precisely the sparsity of their representation over a given basis or dictionary. Our technique is then able to work on the compressed signals, in the sense that one doesn't have to reconstruct them for processing. Second, the gram matrix of the dictionary used to express signals must be stored or computed, too. If this not a problem when the dictionary is an orthogonal basis, it could be a severe limitation in the case of a general redundant dictionary since the gram matrix is a priori large and without particular structure. However, the gram matrix entries of many dictionaries used in practice can be computed in a fast way.

We showed that is possible to maintain deterministic or probabilistic bounds on the true distance between the query and the tested signals. Clearly though, probabilistic bounds are much more favorable than our worst case deterministic bounds. Indeed, we presented clear experimental evidence showing the ability of the algorithm to eliminate non suitable candidates at early stages.

In view of the recent results in compressed sensing, one may wonder wether it would be possible to avoid computing sparse approximations over a fixed dictionary since most of the information of compressible signals can be captured by random projections [32]. Exploring the possibility of working solely with random projections will be one of our future research directions.

References

[1] S. Mallat and Z. Zhang, "Matching pursuit with time-frequency dictionaries," *IEEE Transactions on Signal Processing*, vol. 41, no. 12, pp. 3397–3415, Dec 1993.

[2] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM Journal on Scientific Computing*, vol. 20, no. 1, pp. 33–61, 1999.

[3] J. A. Tropp, "Greed is good: algorithmic results for sparse approximation," *Information Theory, IEEE Transactions on*, vol. 50, no. 10, pp. 2231–2242, Oct. 2004.

[4] R. Gribonval and P. Vandergheynst, "On the exponential convergence of matching pursuits in quasi-incoherent dictionaries," *IEEE Transactions in Inf. Theory*, vol. 52, no. 1, pp. 255–261, January 2006.

[5] D. L. Donoho, M. Elad, and V. Temlyakov, "Stable recovery of sparse overcomplete representations in the presence of noise," *IEEE Transactions on Information Theory*, vol. 52, no. 1, January 2006, Working draft.

[6] D. L. Donoho, M. Vetterli, R. A. DeVore, and I. Daubechies, "Data compression and harmonic analysis," *IEEE Transactions on Information Theory*, vol. 44, pp. 391–432, August 1998.

[7] S. Mallat, *A Wavelet Tour of Signal Processing*, Academic Press, 1999.

[8] R. Gribonval and S. Krstulovic, *The Matching Pursuit ToolKit*, http://gforge.inria.fr/projects/mptk/.

[9] $\ell - 1$ *magic*, http://www.acm.caltech.edu/l1magic/.

[10] P. Jost, P. Vandergheynst, and P. Frossard, "Tree-based pursuit: Algorithm and properties," *IEEE Transactions on Signal Processing*, vol. 54, no. 12, pp. 4685–4697, 2006.

[11] M. D. Plumbley, "Recovery of sparse representations by polytope faces pursuit," in *Proceedings of the 6th International Conference on Independent Component Analysis and Blind Source Separation (ICA 2006)*, Charleston, SC, USA, March 2006, LNCS 3889, pp. 206–213, Springer Verlag, Berlin.

[12] A. J. Bell and T. J. Sejnowski, "The "independent components" of natural scenes are edge filters," *Vision Research*, vol. 37, no. 23, pp. 3327–3338, 1997.

[13] M. S. Lewicki and B. A. Olshausen, "Probabilistic framework for the adaptation and comparison of image codes," *JOSA A*, vol. 16, no. 7, pp. 1587–1601, 1999.

[14] M. S. Lewicki and T. J. Sejnowski, "Learning overcomplete representations," *Neural Computation*, vol. 12, pp. 337–365, 2000.

[15] K. Kreutz-Delgado, J. F. Murray, B. D. Rao, K. Engan, T. W. Lee, and T. J. Sejnowski, "Dictionary learning algorithms for sparse representation," *Neural Computation*, vol. 15, pp. 349–396, 2003.

[16] S. A. Abdallah and M. D. Plumbley, "If edges are the independent components of natural images, what are the independent components of natural sounds?," in *Proceedings of the International Conference on Independent Component Analysis and Blind Signal Separation*, december 2001, pp. 534–539.

[17] P. Jost, S. Lesage, P. Vandergheynst, and R. Gribonval, "Motif: An efficient algorithm for learning translation invariant dictioaries," in *IEEE ICASSP '06*, 2006.

[18] E. Kushilevitz, R. Ostrovsky, and Y. Rabani, "Efficient search for approximate nearest neighbor in high dimensional spaces," *SIAM J. Comput*, vol. 30, pp. 457–474, 2000.

[19] D. Dobkin and R. Lipton, "Multidimensional search problems," *SIAM J. Computing*, vol. 5, pp. 181–186, 1976.

[20] K. Clarkson, "A randomized algorithm for closest-point queries," *SIAM J. Computing*, vol. 17, pp. 830–847, 1988.

[21] K. Clarkson, "An algorithm for approximate closest-point queries," in *Proc. 10th ACM Symp. on Computational Geometry*, 1994.

[22] S. Arya, D. Mount, N. Netanyahu, R. Silverman, and A. Wu, "An optitmal algorithm for approximate nearest neighbor searching in fixed dimensions," in *Proc. 5th ACM-SIAM SODA*, 1994.

[23] J. Kleinberg, "Two algorithms for nearest-neighbor search in high dimensions," in *Proc. 29th STOC*, 1997, pp. 599–608.

[24] Piotr Indyk and Rajeev Motwani, "Approximate nearest neighbors: towards removing the curse of dimensionality," in *Proc. of 30th STOC*, 1998, pp. 604–613.

[25] M. Ledoux and M. Talagrand, *Probability in Banach spaces : isoperimetry and processes*, Springer-Verlag, 1991.

[26] S. K. Mitra, "On the probability distribution of the sum of uniformly distributed random variables," *SIAM J. Appl. Math.*, vol. 20, no. 2, pp. 195, 1971.

[27] D. M. Bradley and R. C. C. Gupta, "On the distribution of the sum of n non-identically distributed uniform random variables," *Annals of the Institute of Statistical Mathematics*, vol. 54, no. 3, pp. 689–700, 2002.

[28] S. A. Nene, S. K. Nayar, and H. Murase, "Columbia object image library (coil-100)," Tech. Rep., CUCS-006-96, February 1996.

[29] R. Figueras i Ventura, P. Vandergheynst, and P. Frossard, "Low rate and flexible image coding with redundant representations," *IEEE Transactions on Image Processing*, vol. 15, no. 3, pp. 726 – 739, 2006.

[30] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*, Cambridge University Press, 2000.

[31] F. Samaria and A. Harter, "Parameterisation of a stochastic model for human face identification," in *2nd IEEE Workshop on Applications of Computer Vision*, Sarasota (Florida), December 1994.

[32] E.J. Candès, J. Romberg, and T. Tao, "Stable signal recovery from incomplete and inaccurate measurements," *Comm. Pure Appl. Math.*, vol. 59, pp. 1207–1223, 2005.