Greedy sparse decompositions: a comparative study

Przemyslaw Dymarski, Nicolas Moreau, Gael Richard

EURASIP Journal on Advances in Signal Processing
a SpringerOpen Journal

REVIEW Open Access

Greedy sparse decompositions: a comparative study

Przemyslaw Dymarski[1*], Nicolas Moreau[2] and Gaël Richard[2]

Abstract

The purpose of this article is to present a comparative study of sparse greedy algorithms that were separately introduced in speech and audio research communities. It is particularly shown that the Matching Pursuit (MP) family of algorithms (MP, OMP, and OOMP) are equivalent to multi-stage gain-shape vector quantization algorithms previously designed for speech signals coding. These algorithms are comparatively evaluated and their merits in terms of trade-off between complexity and performances are discussed. This article is completed by the introduction of the novel methods that take their inspiration from this unified view and recent study in audio sparse decomposition.

Keywords: greedy sparse decomposition, matching pursuit, orthogonal matching pursuit, speech and audio coding

1 Introduction

Sparse signal decomposition and models are used in a large number of signal processing applications, such as, speech and audio compression, denoising, source separation, or automatic indexing. Many approaches aim at decomposing the signal on a set of constituent elements (that are termed atoms, basis or simply dictionary elements), to obtain an exact representation of the signal, or in most cases an approximative but parsimonious representation. For a given observation vector $\underline{x}$ of dimension N and a dictionary F of dimension $N \times L$, the objective of such decompositions is to find a vector $\underline{g}$ of dimension L which satisfies $F\,\underline{g} = \underline{x}$. In most cases, we have $L \gg N$ which *a priori* leads to an infinite number of solutions. In many applications, we are however interested in finding an approximate solution which would lead to a vector $\underline{g}$ with the smallest number K of non-zero components. The representation is either exact (when $\underline{g}$ is solution of $F\,\underline{g} = \underline{x}$) or approximate (when $\underline{g}$ is solution of $F\,\underline{g} \approx \underline{x}$). It is furthermore termed as sparse representation when $K \ll N$.

The sparsest representation is then obtained by finding $\underline{g} \in \mathbb{R}^L$ that minimizes $||\underline{x} - F\underline{g}||_2^2$ under the constraint $||\underline{g}||_0 \leq K$ or, using the dual formulation, by finding $\underline{g} \in \mathbb{R}^L$ that minimizes $||\underline{g}||_0$ under the constraint $||\underline{x} - F\underline{g}||_2^2 \leq \varepsilon$.

An extensive literature exists on these iterative decompositions since this problem has received a strong interest from several research communities. In the domain of audio (music) and image compression, a number of greedy algorithms are based on the founding paper of Mallat and Zhang [1], where the Matching Pursuit (MP) algorithm is presented. Indeed, this article has inspired several authors who proposed various extensions of the basic MP algorithm including: the Orthogonal Matching Pursuit (OMP) algorithm [2], the Optimized Orthogonal Matching Pursuit (OOMP) algorithm [3], or more recently the Gradient Pursuit (GP) [4], the Complementary Matching Pursuit (CMP), and the Orthogonal Complementary Matching Pursuit (OCMP) algorithms [5,6]. Concurrently, this decomposition problem is also heavily studied by statisticians, even though the problem is often formulated in a slightly different manner by replacing the L_0 norm used in the constraint by a L_1 norm (see for example, the Basis Pursuit (BP) algorithm of Chen et al. [7]). Similarly, an abundant literature exists in this domain in particular linked to the two classical algorithms Least Angle Regression (LARS) [8] and the Least Absolute Selection and Shrinkage Operator [9].

* Correspondence: dymarski@tele.pw.edu.pl
[1]Institute of Telecommunications, Warsaw University of Technology, Warsaw, Poland
Full list of author information is available at the end of the article

However, sparse decompositions also received a strong interest from the speech coding community in the eighties although a different terminology was used.

The primary aim of this article is to provide a comparative study of the greedy "MP" algorithms. The introduced formalism allows to highlight the main differences between some of the most popular algorithms. It is particularly shown in this article that the MP-based algorithms (MP, OMP, and OOMP) are equivalent to previously known multi-stage gain-shape vector quantization approaches [10]. We also provide a detailed comparison between these algorithms in terms of complexity and performance. In the light of this study, we then introduce a new family of algorithms based on the cyclic minimization concept [11] and the recent Cyclic Matching Pursuit (CyMP) [12]. It is shown that these new proposals outperform previous algorithms such as OOMP and OCMP.

This article is organized as follows. In Section 2, we introduce the main notations used in this article. In Section 3, a brief historical view of speech coding is proposed as an introduction to the presentation of classical algorithms. It is shown that the basic iterative algorithm used in speech coding is equivalent to the MP algorithm. The advantage of using an orthogonalization technique for the dictionary F is further discussed and it is shown that it is equivalent to a QR factorization of the dictionary. In Section 4, we extend the previous analysis to recent algorithms (conjugate gradient, CMP) and highlight their strong analogy with the previous algorithms. The comparative evaluation is provided in Section 5 on synthetic signals of small dimension (N = 40), typical for code excited linear predictive (CELP) coders. Section 6 is then dedicated to the presentation of the two novel algorithms called herein CyRMGS and CyOOCMP. Finally, we suggest some conclusions and perspectives in Section 7.

2 Notations

In this article, we adopt the following notations. All vectors $\underline{x}$ are column vectors where x_i is the ith component. A matrix $F \in \mathbb{R}^{N \times L}$ is composed of L column vectors such as $F = [\underline{f}^1 \cdots \underline{f}^L]$ or alternatively of NL elements denoted f_k^j, where k (resp. j) specifies the row (resp. column) index. An intermediate vector $\underline{x}$ obtained at the kth iteration of an algorithm is denoted as $\underline{x}^k$. The scalar product of the two real valued vectors is expressed by $<\underline{x}, \underline{y}>= \underline{x}^t\underline{y}$. The L_p norm is written as $||\cdot||_p$ and by convention $||\cdot||$ corresponds to the Euclidean norm (L_2). Finally, the orthogonal projection of $\underline{x}$ on $\underline{y}$ is the vector $\alpha\underline{y}$ that satisfies $<\underline{x} - \alpha\underline{y}, \underline{y} >= 0$, which brings $\alpha = <\underline{x}, \underline{y}>/||\underline{y}||^2$.

3 Overview of classical algorithms

3.1 CELP speech coding

Most modern speech codecs are based on the principle of CELP coding [13]. They exploit a simple source/filter model of speech production, where the source corresponds to the vibration of the vocal cords or/and to a noise produced at a constriction of the vocal tract, and the filter corresponds to the vocal/nasal tracts. Based on the quasi-stationary property of speech, the filter coefficients are estimated by linear prediction and regularly updated (20 ms corresponds to a typical value). Since the beginning of the seventies and the "LPC-10" codec [14], numerous approaches were proposed to effectively represent the source.

In the multi-pulse excitation model proposed in [15], the source was represented as $e(n) = \sum_{k=1}^{K} g_k\delta(n - n_k)$, where $\delta(n)$ is the Kronecker symbol. The position n_k and gain g_k of each pulse were obtained by minimizing $||\underline{x} - \underline{\hat{x}}||^2$, where $\underline{x}$ is the observation vector and $\underline{\hat{x}}$ is obtained by predictive filtering (filter $H(z)$) of the excitation signal $\underline{e}(n)$. Note that this minimization was performed iteratively, that is for one pulse at a time. This idea was further developed by other authors [16,17] and generalized by [18] using vector quantization (a field of intensive research in the late seventies [19]). The basic idea consisted in proposing a potential candidate for the excitation, i.e. one (or several) vector(s) was(were) chosen in a pre-defined dictionary with appropriate gain(s) (see Figure 1).

The dictionary of excitation signals may have a form of an identity matrix (in which nonzero elements correspond to pulse positions), it may also contain Gaussian sequences or ternary signals (in order to reduce computational cost of filtering operation). Ternary signals are also used in ACELP coders [20], but it must be stressed that the ACELP model uses only one common gain for all the pulses. Thus, it is not relevant to the sparse approximation methods, which demand a separate gain

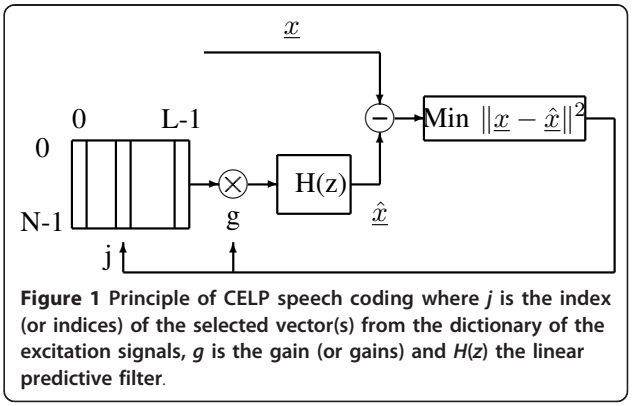

Figure 1 Principle of CELP speech coding where *j* is the index (or indices) of the selected vector(s) from the dictionary of the excitation signals, *g* is the gain (or gains) and *H*(*z*) the linear predictive filter.

for each vector selected from the dictionary. However, in any CELP coder, there is an excitation signal dictionary and a filtered dictionary, obtained by passing the excitation vectors (columns of a matrix representing the excitation signal dictionary) through the linear predictive filter $H(z)$. The filtered dictionary $F = \{\underline{f}^1, ..., \underline{f}^L\}$ is updated every 10-30 ms. The dictionary vectors and gains are chosen to minimize the norm of the error vector. The CELP coding scheme can then be seen as an operation of the multi-stage shape-gain vector quantization on a regularly updated (filtered) dictionary.

Let F be this filtered dictionary (not shown in Figure 1). It is then possible to summarize the CELP main principle as follows: given a dictionary F composed of L vectors $\underline{f}^j$, $j = 1, \cdots, L$ of dimension N and a vector $\underline{x}$ of dimension N, we aim at extracting from the dictionary a matrix A composed of K vectors amongst L and at finding a vector $\underline{g}$ of dimension K which minimizes

$$||\underline{x} - A\underline{g}||^2 = ||\underline{x} - \sum_{k=1}^{K} g_k \underline{f}^{j(k)}||^2 = ||\underline{x} - \underline{\hat{x}}||^2.$$

This is exactly the same problem as the one presented in introduction.[a] This problem, which is identical to multi-stage gain-shape vector quantization [10], is illustrated in Figure 2.

Typical values for the different parameters greatly vary depending on the application. For example, in speech coding [20] (and especially for low bit rate) a highly redundant dictionary ($L \gg N$) is used and coupled with high sparsity (K very small).[b] In music signals coding, it is common to consider much larger dictionaries and to select a much larger number of dictionary elements (or atoms). For example, in the scheme proposed in [21], based on an union of MDCTs, the observed vector $\underline{x}$ represents several seconds of the music signal sampled at 44.1 kHz and typical values could be $N > 10^5$, $L > 10^6$, and $K \approx 10^3$.

3.2 Standard iterative algorithm

If the indices $j(1) \cdots j(K)$ are known (e.g., the matrix A), then the solution is easily obtained following a least square minimization strategy [22]. Let $\underline{\hat{x}}$ be the best approximate of $\underline{x}$, e.g. the orthogonal projection of $\underline{x}$ on the subspace spanned by the column vectors of A verifying:

$$< \underline{x} - A\underline{g}, \ \underline{f}^{j(k)} >= 0 \text{ for } k = 1 \cdots K$$

The solution is then given by

$$\underline{g} = (A^t A)^{-1} A^t \underline{x} \tag{1}$$

when A is composed of K linearly independent vectors which guarantees the invertibility of the Gram matrix A^tA.

The main problem is then to obtain the best set of indices $j(1) \cdots j(K)$, or in other words to find the set of indices that minimizes $||\underline{x} - \underline{\hat{x}}||^2$ or that maximizes

$$||\underline{\hat{x}}||^2 = \underline{\hat{x}}^t \underline{\hat{x}} = \underline{g}^t A^t A \underline{g} = \underline{x}^t A (A^t A)^{-1} A^t \underline{x} \tag{2}$$

since we have $||\underline{x} - \underline{\hat{x}}||^2 = ||\underline{x}||^2 - ||\underline{\hat{x}}||^2$ if $\underline{g}$ is chosen according to Equation 1.

This best set of indices can be obtained by an exhaustive search in the dictionary F (e.g., the optimal solution exists) but in practice the complexity burdens impose to follow a greedy strategy.

The main principle is then to select one vector (dictionary element or atom) at a time, iteratively. This leads to the so-called Standard Iterative algorithm [16,23]. At the kth iteration, the contribution of the k - 1 vectors (atoms) previously selected is subtracted from $\underline{x}$

$$\underline{e}^k = \underline{x} - \sum_{i=1}^{k-1} g_i \underline{f}^{j(i)},$$

and a new index $j(k)$ and a new gain g_k verifying

$$j(k) = \arg\max_j \frac{< \underline{f}^j, \underline{e}^k >^2}{< \underline{f}^j, \underline{f}^j >} \text{ and } g_k \frac{< \underline{f}^{j(k)}, \underline{e}^k >}{< \underline{f}^{j(k)}, \underline{f}^{j(k)} >}$$

are determined.

Let

$\alpha^j = < \underline{f}^j, \underline{f}^j > = ||\underline{f}^j||^2$ be the vector (atom) energy,
$\beta_1^j = < \underline{f}^j, \underline{x} >$ be the crosscorrelation between $\underline{f}^j$ and $\underline{x}$ then $\beta_k^j = < \underline{f}^j, \ \underline{e}^k >$ the crosscorrelation between $\underline{f}^j$ and the error (or residual) $\underline{e}^k$ at step k,
$r_k^j = < \underline{f}^j, \underline{f}^{j(k)} >$ the updated crosscorrelation.

By noticing that

$$\beta_{k+1}^j = < \underline{f}^j, \ \underline{e}^k - g_k \underline{f}^{j(k)} >= \beta_k - g_k r_k^j$$

one obtains the Standard Iterative algorithm, but called herein as the MP (cf. Appendix). Indeed, although

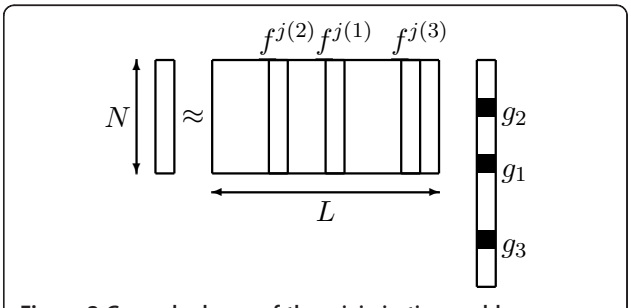

Figure 2 General scheme of the minimization problem.

it is not mentioned in [1], this standard iterative scheme is strictly equivalent to the MP algorithm.

To reduce the sub-optimality of this algorithm, two common methodologies can be followed. The first approach is to recompute all gains at the end of the minimization procedure (this method will constitute the reference MP method chosen for the comparative evaluation section). A second approach consists in recomputing the gains at each step by applying Equation 1 knowing $j(1) \cdots j(k)$, i.e., matrix A. Initially proposed in [16] for multi-pulse excitation, it is equivalent to an orthogonal projection of $\underline{x}$ on the subspace spanned by $\underline{f}^{j(1)} \cdots \underline{f}^{j(k)}$, and therefore, equivalent to the OMP later proposed in [2].

3.3 Locally optimal algorithms

3.3.1 Principle

A third direction to reduce the sub-optimality of the standard algorithm aims at directly finding the subspace which minimizes the error norm. At step k, the subspace of dimension k - 1 previously determined and spanned by $\underline{f}^{j\ (1)} \cdots \underline{f}^{j\ (k-1)}$ is extended by the vector $\underline{f}^{j\ (k)}$, which maximizes the projection norm of $\underline{x}$ on all possible subspaces of dimension k spanned by $\underline{f}^{j(1)} \cdots \underline{f}^{j\ (k-1)}\ \underline{f}^{j}$. As illustrated in Figure 3, the solution obtained by this algorithm may be better than the other solution obtained by the previous OMP algorithm.

This algorithm produces a set of locally optimal indices, since at each step, the best vector is added to the existing subspace (but obviously, it is not globally optimal due to its greedy process). An efficient mean to implement this algorithm consists in orthogonalizing the dictionary F at each step k relatively to the k - 1 chosen vectors.

This idea was already suggested in [17], and then later developed in [24,25] for multi-pulse excitation, and formalized in a more general framework in [26,23]. This framework is recalled below and it is shown as to how it encompasses the later proposed OOMP algorithm [3].

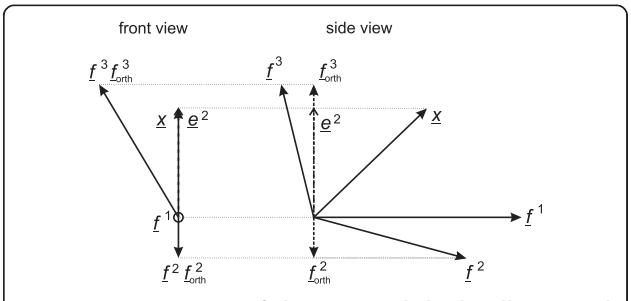

Figure 3 Comparison of the OMP and the locally optimal algorithm: let $\underline{x}$, $\underline{f}^1$, $\underline{f}^2$ lie on the same plane, but $\underline{f}^3$ stem out of this plane. At the first step both algorithms choose $\underline{f}^1$ (min angle with $\underline{x}$) and calculate the error vector $\underline{e}^2$. At the second step the OMP algorithm chooses $\underline{f}^3$ because $\measuredangle(\underline{e}^2, \underline{f}^3) < \measuredangle(\underline{e}^2, \underline{f}^2)$. The locally optimal algorithm makes the optimal choice $\underline{f}^2$ since $\underline{e}^2$ and $\underline{f}^2_{orth}$ are collinear.

3.3.2 Gram-Schmidt decomposition and QR factorization

Orthogonalizing a vector $\underline{f}^j$ with respect to vector $\underline{q}$ (supposed herein of unit norm) consists in subtracting from $\underline{f}^j$ its contribution in the direction of $\underline{q}$. This can be written:

$$\underline{f}^j_{\text{orth}} = \underline{f}^j - < \underline{f}^j,\ \underline{q} > \underline{q} = \underline{f}^j - \underline{q}\underline{q}^t\underline{f}^j = (I - \underline{q}\underline{q}^t)\underline{f}^j.$$

More precisely, if k - 1 successive orthogonalizations are performed relatively to the k - 1 vectors $\underline{q}^1 \cdots \underline{q}^{k-1}$ which form an orthonormal basis, one obtains for step k:

$$\begin{aligned}\underline{f}^j_{\text{orth(k)}} &= \underline{f}^j_{\text{orth(k-1)}} - < \underline{f}^j_{\text{orth(k-1)}},\ \underline{q}^{k-1} > \underline{q}^{k-1} \\ &= [I - \underline{q}^{k-1}(\underline{q}^{k-1})^t]\underline{f}^j_{\text{orth(k-1)}}\end{aligned}$$

Then, maximizing the projection norm of $\underline{x}$ on the subspace spanned by $\underline{f}^{j(1)}_1 \underline{f}^{j(2)}_{\text{orth}(2)} \cdots \underline{f}^{j(k-1)}_{\text{orth(k-1)}} \underline{f}^{j}_{\text{orth(k)}}$ is done by choosing the vector maximizing $(\beta^j_k)^2 \Big/ \alpha^j_k$ with

$$\alpha^j_k = < \underline{f}^j_{\text{orth(k)}},\ \underline{f}^j_{\text{orth(k)}} >$$

and

$$\beta^j_k = < \underline{f}^j_{\text{orth(k)}},\ \underline{x} - \underline{\hat{x}}^{k-1} > = < \underline{f}^j_{\text{orth(k)}},\ \underline{x} >$$

In fact, this algorithm, presented as a Gram-Schmidt decomposition with a partial QR factorization of the matrix $\underline{f}$, is equivalent to the OOMP algorithm [3]. This is referred herein as the OOMP algorithm (see Appendix).

The QR factorization can be shown as follows. If r^j_k is the component of $\underline{f}^j$ on the unit norm vector $\underline{q}^k$, one obtains:

$$\underline{f}^j_{\text{orth(k+1)}} = \underline{f}^j_{\text{orth(k+1)}} - r^j_k\underline{q}^k = \underline{f}^j - \sum_{i=1}^{k} r^j_i\underline{q}^i$$

$$\underline{f}^j = r^j_1\underline{q}^1 + \cdots + r^j_k\underline{q}^k + \underline{f}^j_{\text{orth(k+1)}}$$

$$r^j_k = < \underline{f}^j, \underline{q}^k > = < \underline{f}^j_{\text{orth(k)}} + \sum_{i=1}^{k-1} r^j_i\underline{q}^i, \underline{q}^k >$$

$$r^j_k = < \underline{f}^j_{\text{orth(k)}}, \underline{q}^k >$$

For the sake of clarity and without loss of generality, let us suppose that the kth selected vector corresponds to the kth column of matrix F (note that this can always be obtained by column wise permutation), then, the following relation exists between the original (F) and the

orthogonalized ($F_{\text{orth(k+1)}}$) dictionaries

$$F = [\underline{q}^1 \cdots \underline{q}^k \underline{f}^{k+1}_{\text{orth(k+1)}} \cdots \underline{f}^{L}_{\text{orth(k+1)}}] \times \begin{bmatrix} r_1^1 r_1^2 \cdots \cdots \cdots r_1^L \\ 0 r_2^2 r_2^3 \cdots \cdots r_2^L \\ \vdots \ddots \ddots \ddots \ddots \ddots \vdots \\ \vdots \ddots \ddots r_k^k \cdots r_k^L \\ 0 \cdots \cdots 0 I_{L-k} \end{bmatrix}.$$

where the orthogonalized dictionary $F_{\text{orth(k+1)}}$ is given by

$$F_{\text{orth(k+1)}} = [\underline{0} \cdots \underline{0} \underline{f}^{k+1}_{\text{orth(k+1)}} \cdots \underline{f}^{L}_{\text{orth(k+1)}}]$$

due to the orthogonalization step of vector $\underline{f}^{j(k)}_{\text{orth(k)}}$ by $\underline{q}^k$.

This readily corresponds to the Gram-Schmidt decomposition of the first k columns of the matrix F extended by the remaining L - k vectors (referred as the modified Gram-Schmidt (MGS) algorithm by [22]).

3.3.3 Recursive MGS algorithm

A significant reduction of complexity is possible by noticing that it is not necessary to explicitly compute the orthogonalized dictionary. Indeed, thanks to orthogonality properties, it is sufficient to update the energies α_k^j and cross-correlations β_k^j as follows:

$$\begin{aligned} \alpha_k^j &= ||\underline{f}^j_{\text{orth(k)}}||^2 \\ &= ||\underline{f}^j_{\text{orth(k - 1)}}||^2 - 2r_{k-1}^j < \underline{f}^j_{\text{orth(k - 1)}}, \underline{q}^{k-1} > \\ &\quad + (r_{k-1}^j)^2 ||\underline{q}^{k-1}||^2 \\ &= \alpha_{k-1}^j - (r_{k-1}^j)^2 \end{aligned}$$

$$\beta_k^j = < \underline{f}^j_{\text{orth(k)}}, \underline{x} > = < \underline{f}^j_{\text{orth(k - 1)}}, \underline{x} > - r_{k-1}^j < \underline{q}^{k-1}, \underline{x} >$$

$$\beta_k^j = \beta_{k-1}^j - r_{k-1}^j \frac{\beta_{k-1}^{j(k-1)}}{\sqrt{\alpha_{k-1}^{j(k-1)}}}.$$

A recursive update of the energies and crosscorrelations is possible as soon as the crosscorrelation r_k^j is known at each step. The crosscorrelations can also be obtained recursively with

$$\begin{aligned} r_k^j &= \frac{[< \underline{f}^j, \underline{f}^{j(k)} > - \sum_{i=1}^{k-1} r_i^{j(k)} < \underline{f}^j, \underline{q}^i >]}{\sqrt{\alpha_k^{j(k)}}} \\ &= \frac{[< \underline{f}^j, \underline{f}^{j(k)} > - \sum_{i=1}^{k-1} r_i^{j(k)} r_i^j]}{\sqrt{\alpha_k^{j(k)}}} \end{aligned}$$

The gains $\bar{g}_1 \cdots \bar{g}_K$ can be directly obtained. Indeed, it can be seen that the scalar $< \underline{q}^{k-1}, \underline{x} > = \beta_{k-1}^{j(k-1)} / \sqrt{\alpha_{k-1}^{j(k-1)}}$ corresponds to the component of $\underline{x}$ (or gain) on the $(k - 1)^{th}$ vector of the current orthonormal basis, that is, the gain $\bar{g}_{k-1}$. The gains which correspond to the non-orthogonalized vectors can simply be obtained as:

$$\begin{aligned} [\underline{q}^1 \cdots \underline{q}^K] \begin{bmatrix} \bar{g}_1 \\ \vdots \\ \bar{g}_K \end{bmatrix} &= [\underline{f}^{j(1)} \cdots \underline{f}^{j(K)}] \begin{bmatrix} g_1 \\ \vdots \\ g_K \end{bmatrix} \\ &= [\underline{q}^1 \cdots \underline{q}^K] R \begin{bmatrix} g_1 \\ \vdots \\ g_K \end{bmatrix} \end{aligned}$$

with

$$R = \begin{bmatrix} r_1^{j(1)} & r_1^{j(2)} & \cdots & r_1^{j(K)} \\ 0 & r_2^{j(2)} & \cdots & r_2^{j(K)} \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & r_K^{j(K)} \end{bmatrix}$$

which is an already computed matrix since it corresponds to a subset of the matrix R of size $K \times L$ obtained by QR factorization of matrix F. This algorithm will be further referenced herein as RMGS and was originally published in [23].

4 Other recent algorithms

4.1 GP algorithm

This algorithm is presented in detail in [4]. Therefore, the aim of this section is to provide an alternate view and to show that the GP algorithm is similar to the standard iterative algorithm for the search of index $j(k)$ at step k, and then corresponds to a direct application of the conjugate gradient method [22] to obtain the gain g_k and error $\underline{e}^k$. To that aim, we will first recall some basic properties of the conjugate gradient algorithm. We will highlight how the GP algorithm is based on the conjugate gradient method and finally show that this algorithm is exactly equivalent to the OMP algorithm.[c]

4.1.1 Conjugate gradient

The conjugate gradient is a classical method for solving problems that are expressed by $A\underline{g} = \underline{x}$, where A is a $N \times N$ symmetric, positive-definite square matrix. It is an iterative method that provides the solution $\underline{g}^* = A^{-1}\underline{x}$ in N iterations by searching the vector $\underline{g}$ which minimizes

$$\Phi(\underline{g}) = \frac{1}{2}\underline{g}^t A \underline{g} - \underline{x}^t \underline{g}. \tag{3}$$

Let $\underline{e}^{k-1} = \underline{x} - A\underline{g}^{k-1}$ be the error at step k and note that $\underline{e}^{k-1}$ is in the opposite direction of the gradient $\Phi(\underline{g})$ in

$\underline{g}^{k-1}$. The basic gradient method consists in finding at each step the positive constant c_k which minimizes $\Phi(\underline{g}^{k-1} + c_k\underline{e}^{k-1})$. In order to obtain the optimal solution in N iterations, the Conjugate Gradient algorithm consists of minimizing $\Phi(\underline{g})$, using all successive directions $\underline{q}^1 \cdots \underline{q}^N$. The search for the directions $\underline{q}^k$ is based on the A-conjugate principle.[d]

It is shown in [22] that the best direction $\underline{q}^k$ at step k is the closest one to the gradient $\underline{e}^{k-1}$ that verifies the conjugate constraint (that is, $\underline{e}^{k-1}$ from which its contribution on $\underline{q}^{k-1}$ using the scalar product $<u, Av>$ is subtracted):

$$\underline{q}^k = \underline{e}^{k-1} - \frac{<\underline{e}^{k-1}, A\underline{q}^{k-1}>}{<\underline{q}^{k-1}, A\underline{q}^{k-1}>}\underline{q}^{k-1}. \quad (4)$$

The results can be extended to any $N \times L$ matrix A, noting that the two systems $A\underline{g} = \underline{x}$ and $A^tA\underline{g} = A^t\underline{x}$ have the same solution in $\underline{g}$. However, for the sake of clarity, we will distinguish in the following the error $\underline{e}^k = \underline{x} - A\underline{g}^k$ and the error $\tilde{\underline{e}}^k = A^t\underline{x} - A^tA\underline{g}^k$.

4.1.2 Conjugate gradient for parsimonious representations

Let us recall that the main problem tackled in this article consists in finding a vector $\underline{g}$ with K non-zero components that minimizes $||\underline{x} - F\underline{g}||^2$ knowing $\underline{x}$ and F. The vector $\underline{g}$ that minimizes the following cost function

$$\frac{1}{2}||\underline{x} - F\underline{g}||^2 = \frac{1}{2}||\underline{x}||^2 - (F^t\underline{x})^t\underline{g} + \frac{1}{2}\underline{g}F^tF\underline{g}$$

verifies $F^t\underline{x} = F^tF\underline{g}$. The solution can then be obtained, thanks to the conjugate gradient algorithm (see Equation 3). Below, we further describe the essential steps of the algorithm presented in [4].

Let $A^k = [\underline{f}^{j(1)} \cdots \underline{f}^{j(k)}]$ be the dictionary at step k. For $k = 1$, once the index $j(1)$ is selected (e.g. A^1 is fixed), we look for the scalar

$$g^1 = \arg\min_g \frac{1}{2}||\underline{x} - A^1g||^2 = \arg\min_g \Phi(g)$$

where

$$\Phi(g) = -((A^1)^t\underline{x})^tg + \frac{1}{2}g(A^1)^tA^1g$$

The gradient writes

$$\nabla\Phi(g) = -[(A^1)^t\underline{x} - (A^1)^tA^1g] = -\tilde{\underline{e}}^0(g)$$

The first direction is then chosen as $\underline{q}^1 = \tilde{\underline{e}}^0(0)$.

For $k = 2$, knowing A^2, we look for the bi-dimensional vector $\underline{g}$

$$\underline{g}^2 = \arg\min_{\underline{g}} \Phi(\underline{g}) = \arg\min_{\underline{g}}[-((A^2)^t\underline{x})^t\underline{g} + \frac{1}{2}\underline{g}^t(A^2)^tA^2\underline{g}]$$

The gradient now writes

$$\nabla\Phi(\underline{g}) = -[(A^2)^t\underline{x} - (A^2)^tA^2\underline{g}] = -\tilde{\underline{e}}^1(\underline{g})$$

As described in the previous section, we now choose the direction $\underline{q}^2$ which is the closest one to the gradient $\tilde{\underline{e}}^1(g^1)$, which satisfies the conjugation constraint (e.g., $\tilde{\underline{e}}^1$ from which its contribution on $\underline{q}^1$ using the scalar product $< u, (A^2)^tA^2v >$ is subtracted):

$$\underline{q}^2 = \tilde{\underline{e}}^1\frac{<\tilde{\underline{e}}^1, (A^2)^tA^2\underline{q}^1>}{<\underline{q}^1, (A^2)^tA^2\underline{q}^1>}\underline{q}^1.$$

At step k, Equation 4 does not hold directly since in this case the vector $\underline{g}$ is of increasing dimension which does not directly guarantee the orthogonality of the vectors $\underline{q}^1 \cdots \underline{q}^k$. We then must write:

$$\underline{q}^k = \tilde{\underline{e}}^{k-1} - \sum_{i=1}^{k-1}\frac{<\tilde{\underline{e}}^{k-1}(A^k)^tA^k\underline{q}^i>}{<\underline{q}^i, (A^k)^tA^k\underline{q}^i>}\underline{q}^i. \quad (5)$$

This is referenced as GP in this article. At first, it is the standard iterative algorithm (described in Section 3.2), and then it is a conjugate gradient algorithm presented in the previous section, where the matrix A was replaced by the A^k and where the vector $\underline{q}^k$ was modified according to Equation 5. Therefore, this algorithm is equivalent to the OMP algorithm.

4.2 CMP algorithms

The CMP algorithm and its orthogonalized version (OCMP) [5,6] are rather straightforward variants of the standard algorithms. They exploit the following property: if the vector $\underline{g}$ (again of dimension L in this section) is the minimal norm solution of the underdetermined system $F\underline{g} = \underline{x}$, then it is also a solution of the equation system

$$F^t(FF^t)^{-1}F\underline{g} = F^t(FF^t)^{-1}\underline{x}$$

if in F there are N linearly independent vectors. Then, a new family of algorithms can be obtained by simply applying one of the previous algorithms to this new system of equations $\Phi\underline{g} = \underline{\gamma}$ with $\Phi = F^t(FF^t)^{-1}F$ and $\underline{\gamma} = F^t(FF^t)^{-1}\underline{x}$. All these algorithms necessitate the computation of $\alpha^j = <\underline{\varphi}^j, \underline{\varphi}^j>$, $\beta^j = <\underline{\varphi}^j, \underline{\gamma}>$ and $r_k^j = <\underline{\phi}^j, \underline{\phi}^{j(k)}>$. It is easily shown that if

$$C = [\underline{c}^1 \cdots \underline{c}^L] = (FF^t)^{-1}F$$

then, one obtains $\alpha^j = <\underline{c}^j, \underline{f}^j>$, $\beta^j = <\underline{c}^j, \underline{x}^j>$ and $r_k^j = <\underline{c}^j, \underline{f}^{j(k)}>$.

The CMP algorithm shares the same update equations (and therefore same complexity) as the standard

iterative algorithm except for the initial calculation of the matrix C which requires the inversion of a symmetric matrix of size $N \times N$. Thus, in this article the simulation results for the OOCMP will be obtained with the RMGS algorithm with the modified formulas for α^j, β^j, and r_k^j as shown above. The OCMP algorithm, requiring the computation of the $L \times L$ matrix $\Phi = F^t (FF^t)^{-1} F$ is not retained for the comparative evaluation since it is of greater computational load and lower signal-to-noise (SNR) than OOCMP.

4.3 Methods based on the minimization of the L_1 norm

It must be underlined that an exhaustive comparison of L_1 norm minimization methods is beyond the scope of this article and the BP algorithm is selected here as a representative example.

Because of the NP complexity of the problem,

$$\min||\underline{x} - F\underline{g}||_2^2, \ ||\underline{g}||_0 = K$$

it is often preferred to minimize the L_1 norm instead of the L_0 norm. Generally, the algorithms used to solve the modified problem are not greedy and special measures should be taken to obtain a gain vector having exactly K nonzero components (i.e., $||\underline{g}||_0 = K$). Some algorithms, however, allow to control the degree of sparsity of the final solution–namely the LARS algorithms [8]. In these methods, the codebook vectors $\underline{f}^{j(k)}$ are consecutively appended to the base. In the kth iteration, the vector $\underline{f}^{j(k)}$ having the minimum angle with the current error $\underline{e}^{k-1}$ is selected. The algorithm may be stopped if K different vectors are in the base. This greedy formulation does not lead to the optimal solution and better results may be obtained using, e.g., linear programming techniques. However, it is not straightforward in such approaches to control the degree of sparsity $||\underline{g}||_0$. For example, the solution of the problem [9,27]

$$\min\{\lambda||\underline{g}||_1 + ||\underline{x} - F\underline{g}||_2^2\} \quad (6)$$

will exhibit a different degree of sparsity depending on the value of the parameter λ. In practice, it is then necessary to run several simulations with different parameter values to find a solution with exactly K non-zero components. This further increases the computational cost of the already complex L_1 norm approaches. The L_1 norm minimization may be iteratively re-weighted to obtain better results. Despite the increase of complexity, this approach is very promising [28].

5 Comparative evaluation

5.1 Simulations

We propose in this section a comparative evaluation of all greedy algorithms listed in Table 1.

Table 1 Tested algorithms and corresponding acronyms

Standard iterative algorithm ≡ matching pursuit	MP
OMP or GP	OMP
Locally optimal algorithms (MGS, RMGS or OOMP)	RMGS
Complementary matching pursuit	CMP
Optimized orthogonal CMP	OOCMP
Least angle regression	LARS

For the sake of coherence, other algorithms based on L1 minimization (such as the solution of the problem (6)) are not included in this comparative evaluation, since they are not strictly greedy (in terms of constantly growing L_0). They will be compared with the other non-greedy algorithms (see Section 6).

We recall that the three algorithms, MGS, RMGS, and OOMP are equivalent except on computation load. We therefore only use for the performance evaluation the least complex algorithm RMGS. Similarly, for the OMP and GP, we will only use the least complex OMP algorithm. For MP, the three previously described variants (standard, with orthogonal projection and optimized with iterative dictionary orthogonalization) are evaluated. For CMP, only two variants are tested, i.e., the standard one and the OOCMP (RMGS-based implementation). The LARS algorithm is implemented in its simplest, stepwise form [8]. Gains are recalculated after the computation of the indices of the codebook vectors.

To highlight specific trends and to obtain reproducible results, the evaluation is conducted on synthetic data. Synthetic signals are widely used for comparison and testing of sparse approximation algorithms. Dictionaries usually consist of Gaussian vectors [6,29,30], and in some cases with a constraint of uniform distribution on the unit sphere [4]. This more or less uniform distribution of the vectors on the unit sphere is not necessarily adequate in particular for speech and audio signals where strong correlations exist. Therefore, we have also tested the sparse approximation algorithms on correlated data to simulate conditions which are characteristic to speech and audio applications.

The dictionary F is then composed of $L = 128$ vectors of dimension $N = 40$. The experiments will consider two types of dictionaries: a dictionary with uncorrelated elements (realization of a white noise process) and a dictionary with correlated elements [realizations of a second order AutoRegressive (AR) random process]. These correlated elements are obtained; thanks to the filter $H(z)$:

$$H(z) = \frac{1}{1 - 2\rho\cos(\varphi)z^{-1} + \rho^2 z^{-2}}$$

with $\rho = 0.9$ and $\phi = \pi/4$.

The observation vector $\underline{x}$ is also a realization of one of the two processes mentioned above. For all algorithms, the gains are systematically recomputed at the end of the iterative process (e.g., when all indices are obtained). The results are provided as SNR ratio for different values of K. For each value of K and for each algorithm, M = 1000 random draws of F and $\underline{x}$ are performed. The SNR is computed by

$$SNR = \frac{\sum_{i=1}^{M} ||\underline{x}(i)||^2}{\sum_{i=1}^{M} ||\underline{x}(i) - \hat{\underline{x}}(i)||^2}.$$

As in [4], the different algorithms are also evaluated on their capability to retrieve the exact elements that were used to generate the signal ("exact recovery performance").

Finally, overall complexity figures are given for all algorithms.

5.2 Results

5.2.1 Signal-to-noise ratio

The results in terms of SNR (in dB) are given in Figure 4 both for the case of a dictionary of uncorrelated (left) and correlated elements (right). Note that in both cases, the observation vector $\underline{x}$ is also a realization of the corresponding random process, but it is not a linear combination of the dictionary vectors.

Figure 5 illustrates the performances of the different algorithms in the case where the observation vector $\underline{x}$ is also a realization of the selected random process but this time it is a linear combination of P = 10 dictionary vectors. Note that at each try, the indices of these P vectors and the coefficients of the linear combination are randomly chosen.

5.2.2 Exact recovery performance

Finally, Figure 6 gives the success rate as a function of K, that is, the relative number of times that all the correct vectors involved in the linear combination are retrieved (which will be called exact recovery).

It can be noticed that the success rate never reaches 1. This is not surprising since in some cases the coefficients of the linear combination may be very small (due to the random draw of these coefficients in these experiments) which makes the detection very challenging.

5.2.3 Complexity

The aim of the section is to provide overall complexity figures for the raw algorithms studied in this article, that is, without including the complexity reduction techniques based on structured dictionaries.

These figures, given in Table 2 are obtained by only counting the multiplication/additions operations linked to the scalar product computation and by only retaining the dominant terms[e] (more detailed complexity figures are provided for some algorithms in Appendix).

The results are also displayed in Figure 7 for all algorithms and different values of K. In this figure, the complexity figures of OOMP (or MGS) and GP are also provided and it can be seen, as expected, that their complexity is much higher than RMGS and OMP, while they share exactly the same SNR performances.

5.3 Discussion

As exemplified in the results provided above, the tested algorithms exhibit significant differences in terms of complexity and performances. However, they are sometimes based on different trade-off between these two characteristics. The MP algorithm is clearly the less complex algorithm but it does not always lead to the poorest performances. At the cost of slight increasing complexity due to the gain update at each step, the OMP algorithm shows a clear gain in terms of performance. The three algorithms (OOMP, MGS, and RMGS) allow to reach higher performances (compared to OMP) in nearly all cases, but these algorithms are

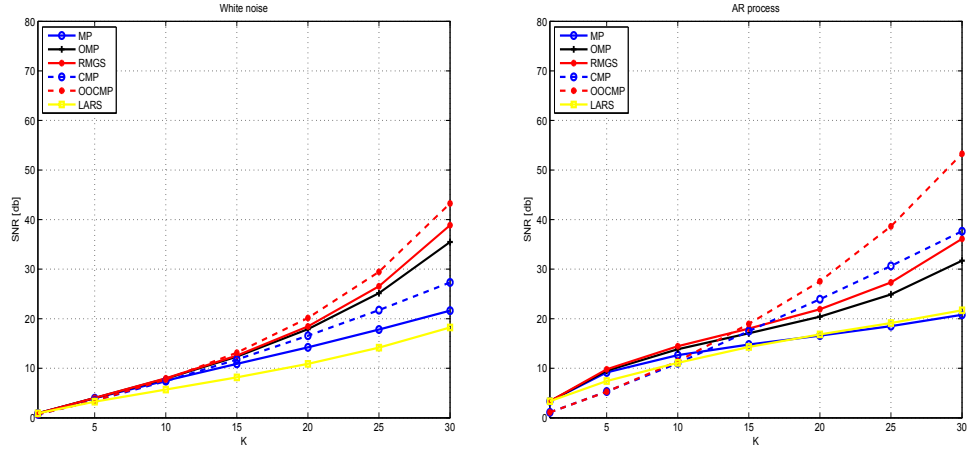

Figure 4 SNR (in dB) for different values of K for uncorrelated signals (left) and correlated signals (right).

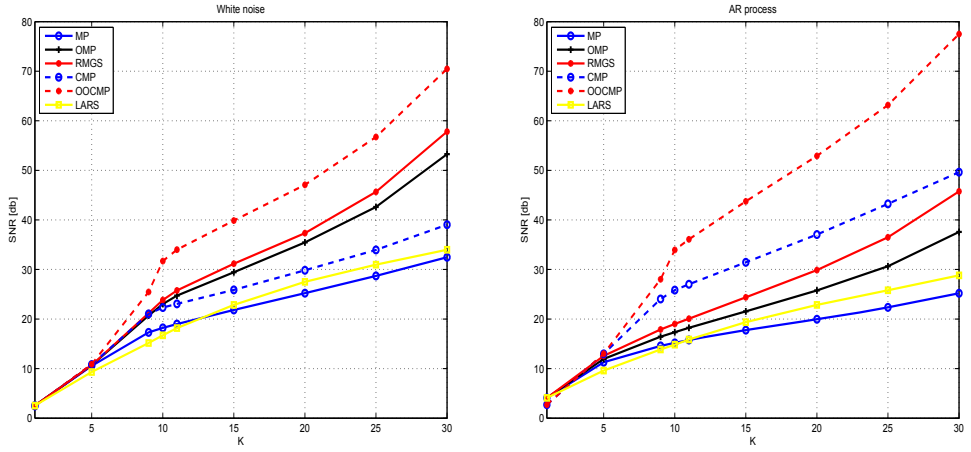

Figure 5 SNR (in dB) for different values of K when the observation signal $\underline{x}$ is a linear combination of $P = 10$ dictionary vectors in the uncorrelated case (left) and correlated case (right).

not at all equivalent in terms of complexity. Indeed, due to the fact that the updated dictionary does not need to be explicitly computed in RMGS, this method has nearly the same complexity as the standard iterative (or MP) algorithm including for high values of K.

The complementary algorithms are clearly more complex. It can be noticed that the CMP algorithm has a complexity curve (see Figure 7) that is shifted upwards compared with the MP's curve, leading to a dramatic (relative) increase for small values of K. This is due to the fact that in this algorithm an initial processing is needed (it is necessary to determine the matrix C - see Section 4.2). However, for all applications where numerous observations are processed from a single dictionary, this initial processing is only needed once which makes this approach quite attractive. Indeed, these algorithms obtain significantly improved results in terms of SNR and in particular OOCMP outperforms RMGS in all but one case. In fact, as depicted in Figure 4, RMGS still obtained better results when the signals were correlated and also in the case where $K << N$ which are desired properties in many applications.

The algorithms CMP and OOCMP are particularly effective when the observation vector $\underline{x}$ is a linear combination of dictionary elements, and especially, when the dictionary elements are correlated. These algorithms can, almost surely, find the exact combination of vectors (contrary to the other algorithms). This can be explained by the fact that the crosscorrelation properties of the normalized dictionary vectors (angles between vectors) are not the same for F and Φ. This is illustrated in Figure 8, where the histograms of the cosines of the angles between the dictionary elements are provided for different values of the parameter ρ of the AR(2) random process. Indeed, the angle between the elements of the dictionary Φ are all close to $\pi/2$, or in other words they are, for a vast majority,

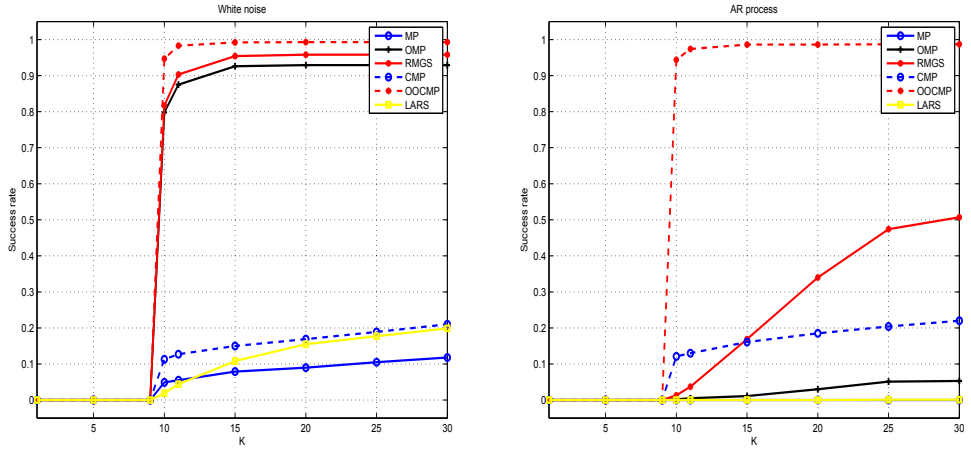

Figure 6 Success rate for different values of K for uncorrelated signals (left) and correlated signals (right).

Table 2 Overall complexity in number of multiplications/ additions per algorithm (approximated)

MP	$(K + 1)NL + K^2N$
OMP	$(K + 1)NL + K^2(3N/2 + K^2/12)$
RMGS	$(K + 1)NL + K^2L/2$
CMP	$(K + 1)NL + K^2N + N^2(2L + N/3)$
OCMP	$NL(2N + L) + K(KL + L^2 + KN)$
OOCMP	$4KNL + N^3/3 + 2N^2L$
LARS	variable, depending on the number of steps
OOMP	$4KNL$
GP	$(K + 1)NL + K^2(10N + K^2)/4$

nearly orthogonal whatever the value of ρ be. This property is even stronger when the F matrix is obtained with realizations of white noise ($\rho = 0$).

This is a particularly interesting property. In fact, when the vector $\underline{x}$ is a linear combination of P vectors of the dictionary F, then the vector $\underline{y}$ is a linear combination of P vectors of the dictionary Φ, and the quasi-orthogonality of the vectors of Φ allows to favor the choice of good vectors (the others being orthogonal to $\underline{y}$). In CMP, OCMP, and OOCMP, the first selected vectors are not necessarily minimizing the norm $||F\underline{g} - \underline{x}||$, which explains why these methods are poorly performing for a low number K of vectors. Note that the operation $\Phi = C^tF$ can be interpreted as a preconditioning of matrix F [31], as also observed in [6].

Finally, it can be observed that the GP algorithm exhibits a higher complexity than OMP in its standard version but can reach lower complexity by some approximations (see [4]).

It should also be noted, that the simple, stepwise implementation of the LARS algorithm yields comparable SNR values to the MP algorithm, at a rather high computational load. It then seems particularly important

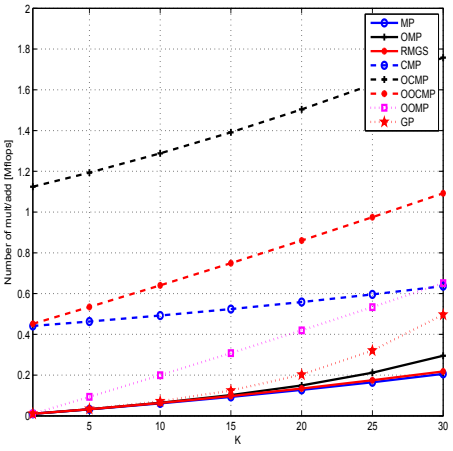

Figure 7 Complexity figures (number of multiplications/ additions in Mflops for different values of *K*).

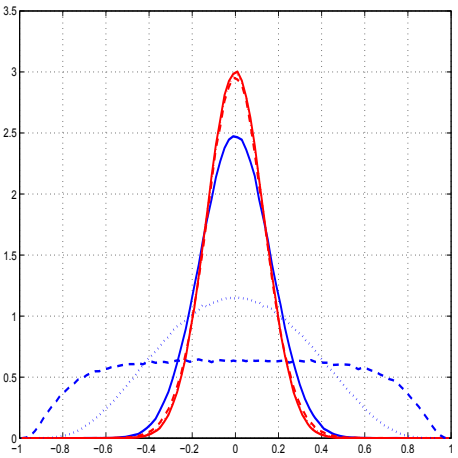

Figure 8 Histogram of the cosines of the angles between dictionary vectors for *F* (in blue) and *Φ* (in red) for $\rho = 0$ (straight line), 0.9 (dotted), 0.99 (intermittent line).

to use more elaborated approaches based on the L_1 minimization. In the next section, we will evaluate in particular a method based on the study of [32].

6 Toward improved performances

6.1 Improving the decomposition

Most of the algorithms described in the previous sections are based upon K steps iterative or greedy process, in which, at step k, a new vector is appended to a subspace defined at step k - 1. In this way, a K-dimensional subspace is progressively created.

Such greedy algorithms may be far from optimality and this explains the interest for better algorithms (i.e., algorithms that would lead to a better subspace), even if they are at the cost of increased computational complexity. For example, in the ITU G.729 speech coder, four vectors are selected in the four nested loops [20]. It is not a full-search algorithm (there are 2^{17} combinations of four vectors in this coder), because the innermost loop is skipped in most cases. It is, however, much more complex than the algorithms described in the previous sections. The Backward OOMP algorithm introduced by Andrle et al. [33] is a less complex solution than the nested loop approach. The main idea of this algorithm is to find a $K' > K$ dimensional subspace (by using the OOMP algorithm) and to iteratively reduce the dimension of the subspace until the targeted dimension K is reached. The criterion used for the dimension reduction is the norm of the orthogonal projection of the vector $\underline{x}$ on the subspace of reduced dimension.

In some applications, the temporary increase of the subspace dimension is not convenient or even not possible (e.g., ACELP [20]). In such cases, optimization of the subspace of dimension K may be performed using the

cyclic minimization concept [11]. Cyclic minimizers are frequently employed to solve the following problem:

$$\min_{\theta_1,\ldots,\theta_K} V(\theta_1, \ldots, \theta_K)$$

where V is a function to be minimized and $\theta_1,\ldots, \theta_K$ are scalars or vectors. As presented in [11], cyclic minimization consists in performing, for $i = 1,\ldots, K$, the minimization with respect to one variable:

$$\bar{\theta}_i = \arg\min_{\theta_i} V(\theta_1, \ldots, \theta_i, \ldots, \theta_K)$$

and substituting the new value $\bar{\theta}_i$ for the previous one: $\bar{\theta}_i \rightarrow \theta_i$. The process can be iterated as many times as desired.

In [12], the cyclic minimization is employed to find the signal model consisting of complex sinusoids. In the augmentation step, a new sinusoid is added (according to the MP approach in frequency domain), and then in the optimization step the parameters of the previously found sinusoids are consecutively revised. This approach, termed as CyMP by the authors, has been extended to the time-frequency dictionaries (consisting of Gabor atoms and Dirac spikes) and to OMP algorithms [34].

Our idea is to combine the cyclic minimization approach with the locally optimal greedy algorithms like RMGS and OOCMP to improve the subspace generated by these algorithms.

Recently some other non-greedy algorithms have been proposed, which also tend to improve the subspace, namely the COSAMP [35] and the Subspace Pursuit (SP) [29]. These algorithms enable, in the same iteration, to reject some of the basis vectors and to introduce new candidate vectors. Greedy algorithms also exist, namely the Stagewise Orthogonal Matching Pursuit (StOMP) [36] and the Regularized Orthogonal Matching Pursuit (ROMP) [30], in which, a series of vectors is selected in the same iteration. It has been shown that the non-greedy SP outperforms the greedy ROMP [29]. This motivates our choice only to include the non-greedy COSAMP and SP algorithms in our study.

The COSAMP algorithm starts with the iteration index $k = 0$, the codebook F, the error vector $\underline{e} = \underline{x}$, and the L-dimensional gain vector $\underline{g}^k = \underline{0}$. Number of non-zero gains in the output gain vector should be equal to K. Each iteration consists of the following steps:

- $k = k + 1$,
- Crosscorrelation computation: $\underline{\beta} = F^t\underline{e}$.
- Search for the $2K$ indices of the largest crosscorrelations:

$\Omega = \text{supp}_{2K}(\underline{\beta})$.

- Merging of the new and previous indices: $T = \Omega \cup \text{supp}_K(\underline{g}^{k-1})$.
- Selection of the codebook vectors corresponding to the indices $T : A = F_T$.
- Calculation of the corresponding gains (least squares): $\underline{g}_T = (A^tA)^{-1}A^t\underline{x}$(the remaining gains are set to zero).
- Pruning $\underline{g}_T$ to obtain K nonzero gains of maximum absolute values: $\underline{g}^k$.
- Update of the error vector: $\underline{e} = \underline{x} - F\underline{g}^k$.
- Stop if $||\underline{e}||_2 < \varepsilon_1$ or $||\underline{g}^k - \underline{g}^{k-1}|| < \varepsilon_2$ or $k = k_{max}$.

Note that, in COSAMP, $2K$ new indices are merged with K old ones, while the SP algorithm merges K old and K new indices. This constitutes the main difference between the two algorithms. For the sake of fair comparison, the stopping condition has been modified and unified for both algorithms.

6.2 Combining algorithms

We propose in this section a new family of algorithms which, like the CyMP, consist of an augmentation phase and an optimization phase. In our approach, the augmentation phase is performed using one of the greedy algorithms described in previous sections, yielding the initial K-dimensional subspace. The cyclic optimization phase consists in substituting new vectors for the previously chosen ones, without modification of the subspace dimension K. The K vectors spanning the subspace are consecutively tested by removing them from the subspace. Each time a K - 1 -dimensional subspace is created. A substitution takes place, if one of the L - K codebook vectors, appended to this K - 1 -dimensional subspace, forms a better K-dimensional subspace than the previous one. The criterion is, naturally, the approximation error, i.e., $||\underline{x} - \underline{\hat{x}}||$. In this way a "wandering subspace" is created: a K-dimensional subspace evolves in the N-dimensional space, trying to approach the vector $\underline{x}$ being modeled. Generic scheme of the proposed algorithms may be described as follows:

1. **The augmentation phase:** Creation of a K-dimensional initial subspace, using one of the locally optimal greedy algorithms.
2. **The cyclic optimization phase:**

(a) **Outer loop:** testing of codebook vectors $\underline{f}^{j(i)}$, $i = 1,\ldots, K$, spanning the K-dimensional subspace. In the i-th iteration vector $\underline{f}^{j(i)}$ is temporarily removed from the subspace.

(b) **Inner loop:** testing the codebook vectors $\underline{f}^l$, $l = 1,\ldots, L$ - except for vectors belonging to the subspace. Substitute $\underline{f}^l$ for $\underline{f}^{j(i)}$ if the obtained new K-dimensional subspace yields better approximation of the modeled vector $\underline{x}$. If there are no substitutions in the inner loop, put the vector $\underline{f}^{j(i)}$ back to the set of spanning vectors.

3. Stop if there are no substitutions in the outer loop (i.e., in the whole cycle).

In the augmentation phase, any greedy algorithm may be used, but, due to the local convergence of the cyclic optimization algorithm, a good initial subspace yields a better final result and reduces the computational cost. Therefore, the OOMP (RMGS algorithm) and OOCMP were considered and the proposed algorithms will be referred below as CyOOMP or CyRMGS and CyOOCMP. In the cyclic optimization phase, the implementation of the operations in both loops is always based on the RMGS (OOMP) algorithm (no matter which algorithm has been used in the augmentation phase). In the outer loop the K - 1 steps of the RMGS algorithm are performed, using already known vector indices. In the inner loop, the Kth step of the RMGS algorithm is made, yielding the index of the best vector belonging to the orthogonalized codebook. Thus, in the inner loop, it may be either one substitution (if the vector $\underline{f}^{j}$ calculated using the RMGS algorithm is better than the vector $\underline{f}^{j(i)}$ temporarily removed from the subspace) or no substitution.

If the initial subspace is good (e.g., created by the OOCMP algorithm), then, in most cases, there are no substitutions at all (the outer loop operations are performed only once). If the initial subspace is poor (e.g., randomly chosen), the outer loop operations are performed many times and the algorithm becomes computationally complex. Moreover, this algorithm stops in some suboptimal subspace (it is not equivalent to the full search algorithm), and it is therefore, important to start from a good initial subspace. The final subspace is, in any case, not worse than the initial one and the algorithm may be stopped at any time.

In [34], the cyclic optimization is performed at each stage of the greedy algorithm (i.e., the augmentation steps and cyclic optimization steps are interlaced). This yields a more complex algorithm, but which possesses a higher probability of finding a better subspace.

The proposed algorithms are compared with the other non-greedy procedures: COSAMP, SP, and L_1 minimization. The last algorithm is based on minimization of (6), using the BP procedure available in [32]. Ten trials are performed with different values of the parameter λ. These values are logarithmically distributed within a range depending on the demanded degree of sparsity K. At the end of each trial, pruning is performed, to select K codebook vectors having the maximum gains. The gains are recomputed according to the least squares criterion.

6.3 Results

The performance results are shown in Figure 9 in terms of SNR (in dB) for different values of K, when the dictionary elements are realizations of the white noise process (left) or AR(2) random process (right).

It can be observed that since RMGS and OOCMP are already quite efficient for uncorrelated signal, the gain in performance for CyRMGS and CyOOCMP are only significant for correlated signals. We then discuss below only the results obtained for the correlated case. Figure 10 (left) provides the SNRs in the case where the vector $\underline{x}$ is a linear combination of P = 10 dictionary vectors and the success rate to retrieve the exact vectors (right).

The SNR are clearly improved for both the algorithms compared with their initial core algorithm in all tested cases. A typical gain of 5 dB is obtained for CyRMGS (compared to RMGS). This cyclic substitution technique also significantly improves the initially poor results of OOCMP for small values of K. One can also notice that a typical gain of 10 dB is observed for the simulations, where $\underline{x}$ is a linear combination of P = 10 dictionary

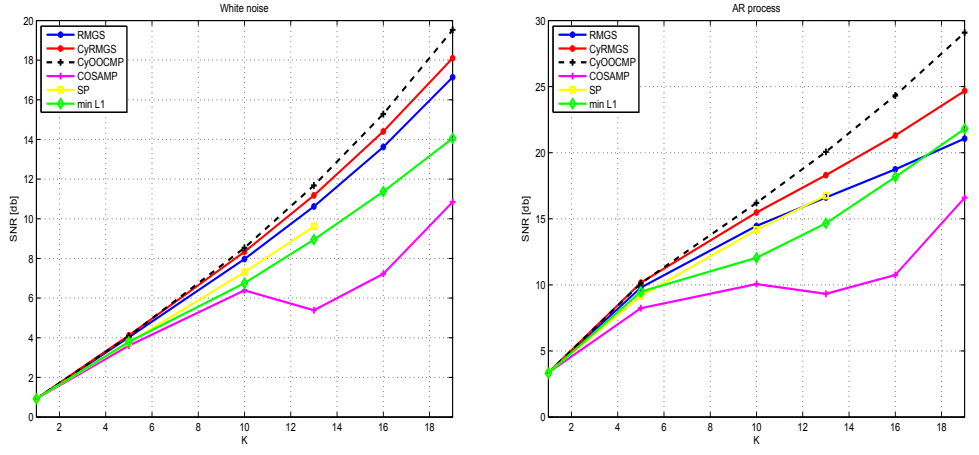

Figure 9 SNR (in dB) for different values of K. These simulations are based on uncorrelated signals (left) and on correlated signals (right).

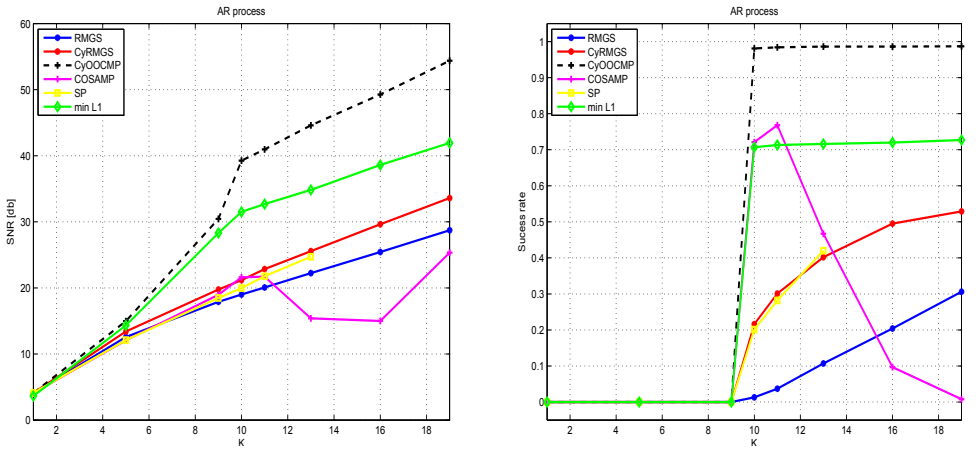

Figure 10 SNR (in dB) for different values of K when the observation signal $\underline{x}$ is a linear combination of $P = 10$ dictionary vectors for correlated signals (left) and the exact recovery performances (right).

vectors for correlated signals (see Figure 10 (left)). Finally, the exact recovery performances are also improved as compared with for both the core algorithms (RMGS and OOCMP).

$L1$ minimization (BP algorithm) performs nearly as good as the Cyclic OOMP, but is more complex in practice due to the necessity of running several trials with different values for the parameter λ.

SP outperforms COSAMP, but both methods yield lower SNR as compared with the cyclic implementations. Moreover, COSAMP and SP do not guarantee monotonic decrease of the error. Indeed, in practice, they often reach a local minimum and yield the same result in consecutive iterations, which stops the procedure. In some other situations they may exhibit oscillatory behaviors, repeating the same sequence of solutions. In that case, the iterative procedure is only stopped after $k_{\max}$ iterations which, for typical value of $k_{\max} = 100$, considerably increases the average computational load. Detection of the oscillations should diminish the computational complexity of these two algorithms.

Nevertheless, the main drawback of these new algorithms is undoubtedly the significant increase in complexity. One may indeed observe that the complexity figures displayed in Figure 11 are of order one in magnitude and higher than those displayed in Figure 7.

7 Conclusion

The common ground of all the methods discussed in this article is the iterative procedure to greedily compute a basis of vectors $\underline{g}^1 \cdots \underline{g}^K$ which are

- simply $\underline{\ell}^{(1)} \cdots \underline{\ell}^{(K)}$ in MP, OMP, CMP, and LARS algorithms,
- orthogonal in OOMP, MGS, and RMGS (explicit computation for the first two algorithms and only implicit for RMGS),
- A-conjugate in GP algorithm.

It was shown in particular in this article that some methods often referred as different techniques in the literature are equivalent. The merit of the different methods was studied in terms of complexity and performances and it is clear that some approaches realize a better trade-off between these two facets. As an example, the RMGS provides substantial gain in performance to the standard MP algorithm with only a very minor complexity increase. Its main interest is indeed the use of a dictionary that is iteratively orthogonalized, but without explicitly building that dictionary. On the

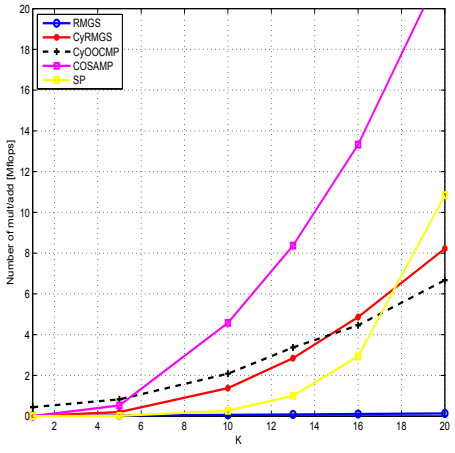

Figure 11 Complexity figures (number of multiplications/additions) for different values of K.

other end, for application where complexity is not a major issue, CMP-based algorithms represent an excellent choice, and especially the newly introduced CyOOCMP.

The cyclic algorithms are compared with the other non-greedy procedures, i.e., COSAMP, SP, and L_1 minimization. The proposed cyclic complementary OOMP successfully competes with these algorithms in solving the sparse and non-sparse problems of small dimension (encountered, e.g., in CELP speech coders).

Although it is not discussed in this article, it is interesting to note that the efficiency of an algorithm may be dependent on how the dictionary F is built. As noted, in the introduction, the dictionary may have an analytic expression (e.g., when F is an union of several transforms at different scales). But F can also be built by machine learning approaches (such as K-means [10], K-SVD [37], or other clustering strategy [38]).

Finally, a recent and different paradigm was introduced, the compressive sampling [39]. Based on solid grounds, it clearly opens the path for different approaches that should permit better performances with possibly smaller dictionary sizes.

Appendix

The algorithmic description of the main algorithms discussed in the article along with the more precise complexity figures is presented in this section. Note that all implementations are directly available on line at http://www. telecom-paristech.fr/~grichard/EURASIP_Moreau2011/.

Algorithm 1 Standard Iterative algorithm (MP)

for j = 1 to L **do**
  $\alpha^j = <\underline{f}^j, \underline{f}^j>$
  $\beta_1^j = <\underline{f}^j, \underline{x}>$
end for
for k = 1 to K **do**
  $j(k) = \arg\max_j (\beta_k^j/\sqrt{\alpha^j})^2$
  $g_k = \beta_k^{j(k)}/\alpha^{j(k)}$
  for j = 1 to L (if $k < K$) **do**
    $r_k^j = <\underline{f}^j, \underline{f}^{j(k)}>$
    $\beta_{k+1}^j = \beta_k^j - g_k r_k^j$
  end for
end for
Option : recompute all gains
$A = [\underline{f}^{j(1)} \cdots \underline{f}^{j(K)}]$
$\underline{g} = (A^t A)^{-1} A^t \underline{x}$
Complexity: $(K + 1)NL + \alpha(K)$, where $\alpha(K) \approx K^3/3$ is the cost of final gains calculation

Algorithm 2 Optimized Orthogonalized MP (OOMP)

for j = 1 to L **do**
  $\alpha_1^j = <\underline{f}^j, \underline{f}^j>$
  $\beta_1^j = <\underline{f}^j, \underline{x}>$
  $\underline{f}_{\mathrm{orth}(1)}^j = \underline{f}^j$
end for
for k = 1 to K **do**
  $j(k) = \arg\max_j (\beta_k^j/\sqrt{\alpha_k^j})^2$
  $\underline{q}^k = -\underline{f}_{\mathrm{orth(k)}}^{j(k)}/\sqrt{\alpha_k^{j(k)}}$
  for j = 1 to L (if $k < K$) **do**
    $\underline{f}_{\mathrm{orth(k+1)}}^j = [I - \underline{q}^k (\underline{q}^k)^t]\underline{f}_{\mathrm{orth(k)}}^j$
    $\alpha_{k+1}^j = <\underline{f}_{\mathrm{orth(k+1)}}^j, \underline{f}_{\mathrm{orth(k+1)}}^j>$
    $\beta_{k+1}^j = <\underline{f}_{\mathrm{orth(k+1)}}^j, \underline{x}>$
  end for
end for
$A = [\underline{f}^{j(1)} \cdots \underline{f}^{j(K)}]$
$\underline{g} = (A^t A)^{-1} A^t x$
Complexity: $(K + 1)NL + 3(K - 1)NL + \alpha(K)$

Algorithm 3 Recursive modified Gram-Schmidt

for j = 1 to L **do**
  $\alpha_1^j = <\underline{f}^j, \underline{f}^j>$
  $\beta_1^j = <\underline{f}^j, \underline{x}>$
end for
for k = 1 to K **do**
  $j(k) = \arg\max_j (\beta_k^j/\sqrt{\alpha_k^j})^2$
  $\bar{g}_k = \beta_k^{j(k)}/\sqrt{\alpha_k^{j(k)}}$
  for j = 1 to L (if $k < K$) **do**
    $r_k^j = [<\underline{f}^j, \underline{f}^{j(k)}> - \sum_{i=1}^{k-1} r_i^{j(k)} r_i^j]/\sqrt{\alpha_k^{j(k)}}$
    $\alpha_{k+1}^j = \alpha_k^j - (r_k^j)^2$
    $\beta_{k+1}^j = \beta_k^j - \bar{g}_k r_k^j$
  end for
end for $g_K = \bar{g}_K/\sqrt{\alpha_K^{j(K)}}$
for k = K - 1 to 1 **do**
  $g_k = (\bar{g}_k - \sum_{i=k+1}^{K} r_k^{j(i)} g_i)/\sqrt{\alpha_k^{j(k)}}$
end for
Complexity: $(K + 1)NL + (K - 1)L(1 + K/2)$

Algorithm 4 Gradient pursuit

for j = 1 to L **do**
  $\alpha^j = <\underline{f}^j, \underline{f}^j>$
  $\beta_1^j = <\underline{f}^j, \underline{x}>$

end for
$\underline{e}^0 = \underline{x}$
$\underline{g}^0 = \underline{0}$
for $k = 1$ to K **do**
  $j(k) = \arg\max_j (\beta_k^j / \sqrt{\alpha^j})^2$
  $A_k = [\underline{f}^{j(1)} \cdots \underline{f}^{j(k)}]$
  $B^k = (A^k)^t A^k$
  $\tilde{\underline{e}} = (A^k)^t \underline{e}^{k-1}$
  if $k = 1$ **then**
    $\underline{q}^k = \tilde{\underline{e}}$
  else
    $a = < \tilde{\underline{e}},\ B^k \underline{q}^{k-1} > / < \underline{q}^{k-1},\ B^k \underline{q}^{k-1} >$
    $\underline{q}^k = \tilde{\underline{e}} - \sum_{i=1}^{k-1} a \underline{q}^{k-1}$
  end if
  $c_k = < \underline{q}^k, \tilde{\underline{e}} > / < \underline{q}^k,\ B^k \underline{q}^k >$
  $\underline{g}^k = \underline{g}^{k-1} + c_k \underline{q}^k$
  $\underline{e}^k = \underline{x} - A^k \underline{g}^k$
  for $j = 1$ to L (if $k < K$) **do**
    $\beta_{k+1}^j = < \underline{f}^j,\ \underline{e}^k >$
  end for
end for
Complexity: $(K+1)NL + \sum_{k=1}^{K} [3Nk + 2k^2 + k^3] + \alpha(K)$

Endnotes

[a]Note though that the vector $\underline{g}$ is now of dimension K instead of L, the indices $j(1) \cdots j(K)$ point to dictionary vectors (columns of F) corresponding to non-zero gains. [b]$K = 2$ or 3, $L = 512$ or 1024, $N = 40$ for a sampling rate of 8kHz are typical values found in speech coding schemes. [c]Several alternatives of this algorithm are also proposed in [4], and in particular the "approximate conjugate gradient pursuit" (ACGP) which exhibits a significantly lower complexity. However, in this article all figures and discussion will only consider the primary GP algorithm. [d]Two vectors u and v are A-conjugate, if they are orthogonal with respect to the scalar product $u^t A v$. [e]The overall complexity figures were obtained by considering the following approximation for small i values: $\sum_{k=1}^{K} k^i \approx K^{i+1}/i$ and by only keeping dominant terms considering that $K \ll N$. Practical simulations showed that the approximation error with these approximative figures was less than 10% compared to the exact figures

Abbreviations

BP: basis pursuit; CELP: code excited linear predictive; CMP: complementary matching pursuit; CyMP: cyclic matching pursuit; GP: gradient pursuit; LARS: least angle regression; MP: matching pursuit; OCMP: orthogonal complementary matching pursuit; OMP: orthogonal matching pursuit; OOMP: optimized orthogonal matching pursuit.

Acknowledgements

The authors would like to warmly thank Prof. Laurent Daudet for his detailed and constructive comments on an earlier version of this manuscript.

Author details

[1]Institute of Telecommunications, Warsaw University of Technology, Warsaw, Poland [2]Institut Telecom, Telecom ParisTech, CNRS-LTCI, Paris, France

Competing interests

The authors declare that they have no competing interests.

Received: 16 February 2011 Accepted: 3 August 2011
Published: 3 August 2011

References

1. S Mallat, Z Zhang, Matching pursuits with time-frequency dictionaries. IEEE Trans. Signal Process. **41**(12), 3397–3415 (1993). doi:10.1109/78.258082
2. Y Pati, R Rezaifar, P Krishnaprasad, Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition, in *Proceedings of the 27th Asilomar Conference on Signals, Systems and Computers* (1993)
3. L Rebollo-Neira, D Lowe, Optimized orthogonal matching pursuit approach. IEEE Signal Process. Lett. **9**, 137–140 (2002). doi:10.1109/LSP.2002.1001652
4. T Blumensath, M Davies, Gradient pursuits. IEEE Trans. Signal Process. **56**(6), 2370–2382 (2008)
5. G Rath, C Guillemot, Sparse approximation with an orthogonal complementary matching pursuit algorithm, in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, 3325–3328 (2009)
6. G Rath, A Sahoo, A comparative study of some greedy pursuit algorithms for sparse approximation, in *Eusipco* (2009)
7. S Chen, D Donoho, M Saunders, Atomic decomposition by basis pursuit. SIAM Rev. **43**(1), 129–159 (2001). doi:10.1137/S003614450037906X
8. B Efron, T Hastie, I Johnstone, R Tibshirani, Least angle regression. Ann. Stat. **32**(2), 407–499 (2004). doi:10.1214/009053604000000067
9. R Tibshirani, Regression shrinkage and selection via the lasso. J. R. Stat. Soc. **58**(1), 267–288 (1996)
10. A Gersho, R Gray, Vector Quantization and Signal Compression. Kluwer Academic Publishers (1992)
11. P Stoica, Y Selen, Cyclic minimizers, majorization techniques, and the expectation-maximization algorithm: a refresher. IEEE Signal Process. Mag. **21**(1), 112–114 (2004). doi:10.1109/MSP.2004.1267055
12. M Christensen, S Jensen, The cyclic matching pursuit and its application to audio modeling and coding, in *Rec. Asilomar Conf. Signals Systems and Computers* (2007)
13. N Moreau, Tools for Signal Compression. ISTE Wiley (2011)
14. T Tremain, The government standard linear predictive coding algorithm: LPC-10. Speech Technology Magazine. **1**, 40–49 (1982)
15. B Atal, J Remde, A new model of LPC excitation for producing natural-sounding speech at low bit rates, in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, 614–617 (1982)
16. M Berouti, H Garten, P Kabal, P Mermelstein, Efficient computation and encoding of the multi-pulse excitation for LPC, in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, 10.1.1–10.1.4 (1984)
17. P Kroon, E Deprettere, Experimental evaluation of different approaches to the multi-pulse coder, in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, 10.4.1–10.4.4 (1984)
18. M Schroeder, B Atal, Code-excited linear prediction (CELP): high-quality speech at very low bit rates, in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, 937–940 (1985)
19. Y Linde, A Buzo, R Gray, An algorithm for vector quantizer design. IEEE Trans. Commun. **COM-28**, 84–95 (1980)
20. R Salami, C Laflamme, J Adoul, A Kataoka, S Hayashi, C Lamblin, D Massaloux, S Proust, P Kroon, Y Shoham, Design and description of CS-ACELP: a toll quality 8 kb/s speech coder. IEEE Trans. Speech Audio Process. **6**(2), 116–130 (1998). doi:10.1109/89.661471
21. E Ravelli, G Richard, L Daudet, Union of MDCT bases for audio coding. IEEE Trans. Speech, Audio Lang. Process. **16**(8), 1361–1372 (2008)
22. G Golub, C Van Loan, Matrix Computations (Johns Hopkins University Press, 1983) (second edition 1989, third edition 1996)
23. P Dymarski, N Moreau, A Vigier, Optimal and sub-optimal algorithms for selecting the excitation in linear predictive coders, in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing* 485–488 (1990)

24. S Singhal, Reducing computation in optimal amplitude multipulse coders, in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, 2363–2366 (1986)
25. S Singhal, B Atal, Amplitude optimization and pitch prediction in multipulse coders. IEEE Trans. Acoust. Speech Signal Process. **37**(3), 317–327 (1989). doi:10.1109/29.21700
26. N Moreau, P Dymarski, Mixed excitation CELP coder, in *Proceedings of the European Conference on Speech Communication and Technology*, 322–325 (1989)
27. D Giacobello, MG Christensen, MN Murthi, SH Jensen, M Moonen, Retrieving sparse patterns using a compressed sensing framework: applications to speech coding based on sparse linear prediction. IEEE Signal Process. Lett. **17**(1), 103–106 (2010)
28. D Giacobello, MG Christensen, MN Murthi, SH Jensen, M Moonen, Enhancing sparsity in linear prediction of speech by iteratively reweighted 1-norm minimization, in *Proceedings of the Internatinoal Conference on Acoustics, Speech, and Signal Processing*, 4650–4653 (2010)
29. W Dai, O Milenkovic, Subspace pursuit for compressive sensing signal reconstruction. IEEE Trans. Inf. Theory, **55**(5), 2230–2249 (2009)
30. D Needell, R Vershynin, Signal recovery from incomplete and inaccurate measurements via regularized orthogonal matching pursuit. IEEE J. Sel. Topics Signal Process. **4**(2), 310–316 (2010)
31. K Schnass, P Vandergheynst, Dictionary preconditioning for greedy algorithms. IEEE Trans. Signal Process. **56**(5), 1994–2002 (2008)
32. G Peyre, Toolbox Sparsity–Sparsity-Based Signal Processing Related Functions. Matlab Central. (2007)
33. M Andrle, L Rebollo-Neira, E Sagianos, Backward-optimized orthogonal matching pursuit approach. IEEE Signal Process. Lett. **11**(9), 705–708 (2004). doi:10.1109/LSP.2004.833503
34. B Sturm, M Christensen, Cyclic matching pursuit with multiscale time-frequency dictionaries, in *Rec. Asilomar Conference on Signals Systems and Computers* (2010)
35. D Needell, JA Tropp, Cosamp: iterative signal recovery from incomplete and inaccurate samples. Appl. Comput. Harmonic Anal. **26**(3), 301–321 (2008)
36. D Donoho, Y Tsaig, I Drori, J Starck, Sparse solution of underdetermined linear equations by stagewise orthogonal matching pursuit, vol. 2. Technical Report, Department of Statistics, Stanford University (2006)
37. M Aharon, M Elad, A Bruckstein, K-SVD: an algorithm for designing overcomplete dictionaries for sparse representation. IEEE Trans. Signal Process. **54**(11), 4311–4322 (2006)
38. P Leveau, E Vincent, G Richard, L Daudet, Instrument-specific harmonic atoms for mid-level music representation. IEEE Trans. Speech Audio Lang. Process. **16**(1), 116–128 (2008)
39. E Candès, M Wakin, An introduction to compressive sampling. IEEE Signal Process. Mag. **56**(6), 21–30 (2008)

doi:10.1186/1687-6180-2011-34
Cite this article as: Dymarski *et al.*: **Greedy sparse decompositions: a comparative study.** *EURASIP Journal on Advances in Signal Processing* 2011 **2011**:34.