Forward Sequential Algorithms for Best Basis Selection*

S. F. Cotter, J. Adler†, B. D. Rao and K. Kreutz-Delgado
Electrical and Computer Engineering Department
Univ. of California, San Diego
La Jolla, California 92093-0407

Abstract

Recently, the problem of signal representation in terms of basis vectors from a large, "overcomplete", spanning dictionary has been the focus of much research. Achieving a succinct, or "sparse", representation is known as the problem of best basis representation. We consider methods which seek to solve this problem by sequentially building up a basis set for the signal. Three distinct algorithm types have appeared in the literature which we term Basic Matching Pursuit (BMP), Order Recursive Matching Pursuit (ORMP) and Modified Matching Pursuit (MMP).

The algorithms are first described and then their computation is closely examined. Modifications are made to each of the procedures which improve their computational efficiency. Each algorithm's complexity is considered in two contexts: one where the dictionary is variable (time dependent), and the other where the dictionary is fixed (time independent). Experimental results are presented which demonstrate that the ORMP method is the best procedure in terms of its ability to give the most compact signal representation, followed by MMP and then BMP which gives the poorest results. Finally, weighing the performance of each algorithm, its computational complexity and the type of dictionary available, we make recommendations as to which algorithms should be used for a given problem.

*This work was partially supported by U.S. MICRO Grant 98-125, Nokia and Qualcomm
†Soundcode Inc., Kirkland, WA 98033

1 Introduction

The problem of selecting a subset of basis elements from a large set of vectors has a long history and can be traced to the search for optimal regressions in the statistical literature [1, 2]. So called projection pursuit algorithms were first developed to solve this problem in [3]. However, it was the adaptation of this algorithm in [4] to signal decomposition which led to a great deal of interest in this problem. In [4], a greedy algorithm called *matching pursuit* was developed to choose vectors from a large dictionary (collection of waveforms) to produce a compact signal representation. The use of a redundant dictionary allows flexibility in signal representation and an appropriate choice of basis will give a compact representation.

Subsequently, there has been much research in achieving compact representations of signals by selecting subsets of elements from overcomplete bases [5, 6]. Audio signals [7, 8] and images [9, 10, 11] have been the focus of most attention and the most commonly used dictionaries are wavelet or wavelet packet dictionaries. Interestingly, subset selection problems arise in many different areas [12], such as spectral estimation, functional approximation etc. [13]-[23].

Many different algorithms have been suggested for the solution of this problem. Minimization of functionals such as the ℓ_1 norm [24, 25] or the more general $\ell_{(p\leq 1)}$ norm [17, 26] have been shown to produce sparse solutions. The most commonly used algorithms are those based on a forward sequential search [4, 18, 19, 27, 28, 29, 30, 31] where the basis vectors, which will be used to compactly represent the signal, are selected one after the other from the dictionary of available vectors. These algorithms are the focus of this paper.

The results of this paper are an extension and refinement of some of our earlier work reported in [31, 32]. In section 2, the best basis selection problem is clearly formulated and the three forward selection algorithms are described. We introduce modifications to these basic algorithms in section 3 which increase the efficiency of these algorithms over implementations described elsewhere. The full computation involved in implementing each algorithm is detailed and compared with that of the other algorithms. Various experiments are presented in section 4 to show how the algorithms perform in finding a compact basis set. In section 5, we draw some conclusions from the analysis presented in earlier sections.

2 Forward Selection Algorithms

The best basis selection problem is as follows. Let $D = \{a_l\}_{l=1}^n$ be a set/dictionary of vectors which is highly redundant, i.e. $a_l \in R^m$ and $m << n$ with $R^m = \text{Span}(D)$. For convenience, we assume that the vectors a_l have unit norm i.e. $\|a_l\| = 1, l = 1, \cdots, n$. Given a signal

vector $b \in R^m$, and a preset error tolerance, ϵ, the problem is to find the most compact representation of b to within the given tolerance using the basis vectors in the dictionary D. Therefore it involves determining the number r (the *sparsity index*) and the set of vectors $\{a_{k_i}\}_{i=1}^{r}$ that best model b. Because we are *pursuing* the goal of determining a small subset of vectors in the dictionary D that best *match* the vector b, algorithms that accomplish this goal are often referred to as *matching pursuit* algorithms.

To determine the optimal value for the sparsity of the solution we would have to search over subsets of the columns of size r where r varies from $1, \cdots, m$. This problem is NP-hard [18], and the computation quickly becomes infeasible as the dictionary size increases. Therefore, suboptimal methods of reasonable complexity, such as those described in sections 2.1-2.3, have been developed to solve this problem. In each of the algorithms to be described the basis elements are selected sequentially i.e. the basis set is built up one vector at a time. To facilitate the presentation, we develop some notation and this is summarized in Table 1.

Table 1

- b - the signal vector.
- b_p - the residual vector after the pth iteration, where $b_0 = b$.
- $I_p = \{k_1, k_2, .., k_p\}$, $I_0 = \emptyset$. This set stores the indices k_i of the p vectors selected.
- $S_p = \left[a_{k_1}, a_{k_2}, ..., a_{k_p}\right]$, $S_0 = \emptyset$. This matrix stores the selected vectors as columns.
- P_{S_p} - the orthogonal projection matrix onto the range space of S_p. Its orthogonal complement $P_{S_p}^{\perp} = \left(I - P_{S_p}\right)$, $P_{S_0} = 0$, $P_{S_0}^{\perp} = I$.
- $P_{a_l} = a_l a_l^H$ - the projection matrix onto the space spanned by a single unit norm vector a_l is denoted by P_{a_l}.

Table 1. Algorithm Notation

2.1 Basic Matching Pursuit (BMP)

This method was suggested in [4] and has the advantage of being computationally simple with provable approximation properties. Not surprisingly, similar algorithms have been developed for subset selection in other application contexts. For instance, the matching pursuit algorithm was developed independently for speech coding in the context of pulse location determination in multipulse speech coders [30, 33, 34].

In this basis selection method, in the pth iteration the vector most closely aligned with the residual b_{p-1} is chosen, where the alignment is measured as the 2-norm of the projection of the residual onto the vector, i.e.

$$\|P_{a_l} b_{p-1}\| = \|a_l a_l^H b_{p-1}\| = |a_l^H b_{p-1}|.$$

Thus the selection criterion becomes,

$$k_p = \arg\max_l \|P_{a_l} b_{p-1}\| = \arg\max_l |a_l^H b_{p-1}|, \; l = 1, \cdots, n, \; l \neq k_{p-1}. \tag{1}$$

If $k_p \notin I_{p-1}$, then the index and basis sets are updated, i.e. $I_p = I_{p-1} \cup \{k_p\}$, and $S_p = [S_{p-1}, a_{k_p}]$; otherwise $I_p = I_{p-1}$ and $S_p = S_{p-1}$. The new residual vector is then computed as

$$b_p = P^{\perp}_{a_{k_p}} b_{p-1} = b_{p-1} - (a_{k_p}^H b_{p-1}) a_{k_p}. \tag{2}$$

The procedure terminates when either $p = r$ (for specified sparsity index r) or $\|b_p\| \leq \epsilon$ (for specified ϵ). Equations (1) and (2), together with the check for termination give the Matching Pursuit (MP) algorithm (with $b_0 = b$). To distinguish this approach from the others introduced below we refer to this algorithm as the Basic Matching Pursuit Algorithm (BMP). It is evident that the algorithm is computationally simple, and it is shown in [4] that it has the desirable convergence property that the norm of the residual vector is monotonically reduced in each iteration. However, the algorithm has its drawbacks both in terms of how the residual vector is computed in (2), and in the manner in which the basis vector is selected in (1). We will elaborate on these deficiencies in section 2.3. As a consequence of these limitations, other algorithms for basis selection have been suggested which we discuss next.

2.2 Order Recursive Matching Pursuit (ORMP)

The origin of this method has its roots in many works, e.g. subset selection [2], functional approximation [18, 19], speech coding [30] etc.. We adapt these procedures to the signal representation problem. Conceptually, the pursuit of the matching pth basis vector involves solving $(n-p+1)$ order recursive least squares problems of the type $\min_y \|S_p^{(l)} y - b\|$ ([35],page 232), where we use the notation $S_p^{(l)} = [S_{p-1}, a_l]$. The vector $a_l \notin S_{p-1}$ that reduces the residual the most is selected and added to S_{p-1} to form S_p. Since order recursive least squares is the basis of this matching pursuit algorithm, we refer to it as the **O**rder **R**ecursive **M**atching **P**ursuit (ORMP) algorithm. The selected index is

$$k_p = \arg\min_l \|P^{\perp}_{S_p^{(l)}} b\|, \; l \notin I_{p-1}, \tag{3}$$

in which case $S_p = S_p^{(k_p)} = [S_{p-1}, a_{k_p}]$ and $b_p = P_{S_p}^{\perp} b$. The projection operator $P_{S_p^{(l)}}$ can be recursively updated via

$$\begin{aligned} P_{S_p^{(l)}} &= P_{S_{p-1}} + \frac{1}{\|P_{S_{p-1}}^{\perp} a_l\|^2} P_{S_{p-1}}^{\perp} a_l a_l^H P_{S_{p-1}}^{\perp} \\ &= P_{S_{p-1}} + \frac{a_l^{(p-1)} (a_l^{(p-1)})^H}{\|a_l^{(p-1)}\|^2} \end{aligned}$$

where

$$a_l^{(p)} \equiv P_{S_p}^{\perp} a_l = P_{S_p}^{\perp} a_l^{(p-1)}, \tag{4}$$

using the fact that $P_{S_p}^{\perp} = P_{S_p}^{\perp} P_{S_{p-1}}^{\perp}$ (which also shows that $b_p = P_{S_p}^{\perp} b = P_{S_p}^{\perp} b_{p-1}$). The index selection criterion (3), by incorporating (4), can therefore be simplified to

$$k_p = \arg\max_l \frac{|(a_l^{(p-1)})^H b_{p-1}|}{\|a_l^{(p-1)}\|}, \ l \notin I_{p-1}, \tag{5}$$

resulting in $I_p = I_{p-1} \cup \{k_p\}$, $S_p = S_p^{(k_p)} = [S_{p-1}, a_{k_p}]$.

The projection operator is updated as $P_{S_p} = P_{S_p^{(k_p)}} = P_{S_{p-1}} + q_p q_p^H$ where

$$q_p \equiv \frac{a_{k_p}^{(p-1)}}{\|a_{k_p}^{(p-1)}\|}, \tag{6}$$

and the orthogonalization step (4) can then be expanded as

$$a_l^{(p)} = P_{S_p}^{\perp} a_l^{(p-1)} = a_l^{(p-1)} - (q_p^H a_l^{(p-1)}) q_p. \tag{7}$$

The residual vector b_p is recursively computed as

$$b_p = P_{S_p}^{\perp} b_{p-1} = b_{p-1} - (q_p^H b_{p-1}) q_p. \tag{8}$$

The algorithm is terminated by using the same criteria as in the BMP i.e. when either $p = r$ (for specified sparsity index r) or $\|b_p\| \leq \epsilon$ (for specified ϵ). Equations (5)–(8) constitute the ORMP algorithm (with $b_0 = b$, $a_l^{(0)} = a_l$, $l = 1, \ldots, n$). Note that the residual $b_p = P_{S_p}^{\perp} b$ is the orthogonal projection of b onto the orthogonal complement of the range space of S_p, and therefore is the smallest possible error (in the 2-norm sense) when b is to be represented in the span of the columns of S_p.

Also it is to be noted that in (3), or equivalently in (5), the optimization is only over previously unselected dictionary vectors. Changing the optimization to include previously selected dictionary vectors will not change the overall outcome. The reason for this can be seen by noting that adding a basis element, a_l, which has already been used to form S_{p-1}, will not change the space spanned by the new set $S_p^{(l)}$. So $P_{S_p^{(l)}} = P_{S_{p-1}}$ and this selection will not minimize (3).

2.3 Modified Matching Pursuit (MMP)

Compared to the BMP, the ORMP algorithm differs in both the manner in which the basis vectors are chosen and in the computation of the residual. Because of the more exhaustive nature of the ORMP vector selection process, there is reason to believe that it will be more successful than BMP in finding a more compact representation. This is supported by the simulations presented in section 4. From a computational perspective, ORMP appears at first glance to be more complex. A more detailed account of the computational complexity is provided in section 3.1.3.

A closer examination of the residual computation step in BMP, as given in (2), reveals some deficiencies of the BMP method for which a fix can be readily obtained by using the ORMP residual computation approach. This results in the Modified Matching Pursuit (MMP) algorithm. Examining the residual computation step of the BMP algorithm, note that

$$b_p^{BMP} = P^{\perp}_{a_{k_p}} b_{p-1} = \Pi_{l=1}^{p} P^{\perp}_{a_{k_l}} b \neq P^{\perp}_{S_p^{BMP}} b, \; a_{k_l} \in S_p^{BMP}.$$

That is, the sequence of one-dimensional projections defining the BMP residual b_p^{BMP} is *not*, in general, equal to an orthogonal projection onto the orthogonal complement of the range space of S_p^{BMP}. Re-selection of a column is possible in BMP but avoided in ORMP through the formation of the residual $P^{\perp}_{S_p} b$. This deficiency in the BMP algorithm was also noted in [28] and an algorithm for computing $P^{\perp}_{S_p} b$, termed Orthogonal Matching Pursuit, which involves solving a set of normal equations was developed. A more efficient approach is that based on a modified Gram-Schmidt approach [29, 31], which is presented next.

Modifying the BMP procedure, in the pth iteration the index k_p is selected by finding the vector best aligned with the residual obtained by projecting b onto the orthogonal complement of the range space of S_{p-1} i.e.

$$\begin{aligned} k_p &= \arg\max_l |a_l^H P^{\perp}_{S_{p-1}} b| \\ &= \arg\max_l |a_l^H b_{p-1}|, \; l \notin I_{p-1}, \end{aligned} \tag{9}$$

where $b_{p-1} = P^{\perp}_{S_{p-1}} b$. Then $I_p = I_{p-1} \cup \{k_p\}$, $S_p = S_p^{(k_p)} = [S_{p-1}, a_{k_p}]$. As in ORMP, note that limiting the range of the search to $l \notin I_{p-1}$ yields the same result as searching over the entire dictionary. In contrast to the ORMP basis selection step (5), in (9) there is no need to compute $a_l^{(p-1)} = P^{\perp}_{S_{p-1}} a_l$, for every $l \notin I_{p-1}$. However, the quantity $b_{p-1} = P^{\perp}_{S_{p-1}} b$ must be calculated. Using the insights from the ORMP algorithm, we find that this can be efficiently done by using a Modified Gram-Schmidt type of procedure as follows. With the

initialization, $\hat{a}_{k_p}^{(0)} = a_{k_p}, q_0 = 0$, we have $P_{S_p} = P_{[S_{p-1}, a_{k_p}]} = P_{S_{p-1}} + q_p q_p^H$ where

$$\begin{aligned} \hat{a}_{k_p}^{(\ell)} &= \hat{a}_{k_p}^{(\ell-1)} - (q_{\ell-1}^H \hat{a}_{k_p}^{(\ell-1)}) q_{\ell-1}, \ \ell = 1, .., p \\ q_p &= \frac{\hat{a}_{k_p}^{(p)}}{\|\hat{a}_{k_p}^{(p)}\|}. \end{aligned} \tag{10}$$

The residual is now formed as

$$b_p = P_{S_p}^{\perp} b_{p-1} = b_{p-1} - (q_p^H b_{p-1}) q_p. \tag{11}$$

In common with the other algorithms, the sequence of iterations is terminated when either $p = r$ (for specified sparsity index r) or $\|b_p\| \leq \epsilon$ (for specified ϵ).

Equations (9)–(11) define the Modified Matching Pursuit (MMP) algorithm. It is noted that MMP avoids the burdensome step (7) required by ORMP, of having to project *all* the vectors remaining in the dictionary at each iteration onto $S_p^{\perp}$. This has been replaced by the need to project only the single optimal vector in (10). Also note that in (9) the optimization is only over previously unselected dictionary vectors, thereby avoiding the re-selection problem of the BMP. Comparison of (1)–(2) with (9)–(11) shows that MMP retains much of the computational simplicity of BMP; the two algorithms essentially differ only in the addition of the projection step (10). Therefore the MMP algorithm is potentially intermediate in cost between the BMP and the ORMP. The MMP should exhibit the benefits of working with the optimal $P_{S_p}^{\perp} b$–residual thereby avoiding the vector re-selection problem.

3 Computation Analysis

In the previous section, the basic algorithms for selecting a subset of the basis elements have been presented. We now turn our attention to the computation involved in each algorithm and describe modifications which result in more efficient implementations of the algorithms. Motivated by applications, we consider the computational complexity of the algorithms in two contexts: one where the dictionary D is variable (time dependent), and the other where the dictionary D is fixed (time independent). For instance, in multipulse speech coding the dictionary varies from frame to frame [12, 30, 33, 34] giving rise to the variable dictionary scenario, while a fixed dictionary can be used in time–frequency representations of a signal [4, 25]. The choice of a fixed or variable dictionary is important because it involves a trade–off between memory usage and computation. For instance, with the use of a fixed dictionary, certain computations can be viewed as overhead and can lead to a lower complexity implementation at the expense of increased memory requirements. Available resources may therefore dictate which approach is taken.

3.1 Algorithm Computation

3.1.1 BMP

In the BMP algorithm, it can be noted that only $a_l^H b_p$ is required to choose the next basis element. The intermediate step as given in (2) can be replaced by the following recursion which updates the inner product $a_l^H b_p$ instead [4],

$$a_l^H b_p = a_l^H b_{p-1} - \frac{(a_l^H a_{k_p})(a_{k_p}^H b_{p-1})}{\|a_{k_p}\|^2}, l = 1, \cdots, n. \tag{12}$$

In the initialization of BMP, the norm of each vector, $\|a_l\|^2$, is computed. From this, the ratio $\frac{1}{\|a_l\|^2}$ is formed and stored. The normalizing factor $\frac{1}{\|a_{k_p}\|^2}$ is required in this equation because the basis elements are no longer assumed to be of unit norm. However, because of the initial computation, this quantity is available and so no divisions are required in the iteration. We also include computation of $a_l^H b_0$ as an initial step.

From (12), we note that it is not necessary to either explicitly form b_p or compute the new inner products $a_l^H b_p$ in each iteration. This results in a large saving in computation if we have the inner product $a_l^H a_{k_p}$ available to us (see section 3.4). If these inner products are not available then we need to compute $a_l^H a_{k_p}$ which is computationally equivalent to forming $a_l^H b_p$. Recursion (12) requires a further n additions while forming b_p, as in (2), requires $2m$ multiplies. Computationally, the two methods are similar but from a storage perspective, it is preferable just to store b_p (m locations) instead of $a_l^H b_p$ (n locations). The computational results in section 3.3, where the dictionary is time-varying, are based on forming b_p explicitly and then forming $a_l^H b_p$. With a fixed dictionary, as considered in section 3.4, recursion (12) is used to give an efficient implementation.

The check for termination may require the calculation of $\|b_p\|$ if an ϵ is specified but this can be easily obtained by noting that

$$\|b_p\|^2 = \|b_{p-1}\|^2 - \frac{|a_{k_p}^H b_{p-1}|^2}{\|a_{k_p}\|^2}, \tag{13}$$

and that both the numerator and denominator of the final term in this expression are available.

3.1.2 MMP

In the MMP, it is clear that we can use a similar modification to that used for the BMP (12). Instead of updating b_{p-1}, the inner product $a_l^H b_{p-1}$ is updated and this is done via the following recursion

$$a_l^H b_p = a_l^H b_{p-1} - (a_l^H q_p)(q_p^H b_{p-1}). \tag{14}$$

In this recursion, we must form $a_l^H q_p$ for each column, $a_l,\ l \notin I_p$ and compute $q_p^H b_{p-1}$. In the fixed dictionary case, for the BMP, by using (12) and precomputing inner products $a_l^H a_{k_p}$, we were able to save on computation. However, in (14) precomputation of $a_l^H q_p$ is not possible since q_p is not available. Therefore, based on the same arguments as in 3.1.1, in both the case of a fixed and time-varying dictionary, it is cheaper computationally and storage-wise to implement the algorithm as given in (9)–(11).

3.1.3 ORMP

The description of the ORMP which has been presented in section 2.2 above was based on [18]. Work presented in [19, 27] attempted to reduce the complexity of the ORMP algorithm. For the overdetermined case i.e. $m > n$, it was shown that the computational complexity was reduced. However, as the authors stated, their implementations were not computationally better than [18] for the underdetermined case, i.e. $m < n$, which is the case of interest here. We now formulate three modifications to the basic ORMP algorithm which substantially reduce its complexity where the dictionary D is underdetermined.

First we recall that $a_l^{(p-1)} = P_{S_{p-1}}^{\perp} a_l$, so

$$\begin{aligned}(a_l^{(p-1)})^H b_{p-1} &= (P_{S_{p-1}}^{\perp} a_l)^H b_{p-1} \\ &= a_l^H (P_{S_{p-1}}^{\perp} b_{p-1}) \\ &= a_l^H b_{p-1}. \end{aligned} \tag{15}$$

This means that the projection of each of the columns a_l implied by (5), which is the main computational bottleneck in ORMP, is not required! The selection step can be rewritten as

$$k_p = \arg\max_l \frac{|a_l^H b_{p-1}|}{\|a_l^{(p-1)}\|}. \tag{16}$$

A recursion with the same form as (14) can be used to compute the numerator. The norm $\|a_l^{(p)}\|$ in the denominator of this equation must still be computed for each value of l. However, these norms can be formed recursively, producing a further reduction in computation, using

$$\|a_l^{(p)}\|^2 = \|a_l^{(p-1)}\|^2 - \frac{(|a_l^H a_{k_p}^{(p-1)}|)^2}{\|a_{k_p}^{(p-1)}\|^2}. \tag{17}$$

This recursion constitutes the second modification to the algorithm. We found that by carrying out (14) and then (17), computation is reduced over explicitly forming the residual b_p, computing $a_l^H b_p$ and then using (17). The reason for this is that the expression $\frac{(a_l^H a_{k_p}^{(p-1)})}{\|a_{k_p}^{(p-1)}\|^2}$ arises in both (14) and (17) and the saving in computation is $O(mn)$ multiplies per iteration.

Thirdly, since $a_l^{(p-1)}$ is no longer in the numerator, the orthogonalization of *all* of the unchosen columns is no longer required. Only the chosen set of columns, which is a much smaller set, must be orthogonalized. This may be done using the same formulation as was developed for the MMP algorithm (10), and clearly this represents a huge saving in computation over having to orthogonalize all the vectors at each step.

The difference in complexity between this reduced complexity ORMP algorithm and the MMP reduces essentially to the calculation of the norms $\|a_l^{(p)}\|^2$. From (17), it is seen that $\frac{1}{\|a_l^{(p)}\|^2}$ is required to correctly select the next basis element and this means that divisions are necessary. These divisions are noteworthy as they do not arise in either the BMP or MMP!

3.1.4 ORMP via Cholesky Decomposition

The ORMP algorithm described in section 2.2 uses a QR Decomposition of the basis set to solve the sequence of least squares problems which arise. Another approach to these least squares problems is to use the Cholesky Decomposition to solve the associated Normal Equations [30]. As we will compare its complexity to that of the algorithms already described, we give an outline of this algorithm.

Recall that at the pth step, $(p-1)$ columns have been chosen and the matrix $S_p^{(i)} = [S_{p-1}, a_i] = [a_{k_1}, a_{k_2}, \cdots, a_{k_{p-1}}, a_i]$ is formed for each unchosen basis element a_i. The index k_p is then chosen as

$$k_p = \arg\min_i \|b - S_p^{(i)} x_p\|.$$

This requires solving many least squares problems for which the corresponding normal equations are

$$S_p^{(i)^T} S_p^{(i)} x_p = S_p^{(i)^T} b.$$

The Cholesky Decomposition is used to solve this system of equations by forming $S_p^{(i)^T} S_p^{(i)} = L(i)L^T(i)$, where $L(i)$ is a lower triangular matrix. The computational efficiency is achieved by noting that in the previous iteration the Cholesky Decomposition for $S_{p-1}^T S_{p-1}$ has already been determined. Hence, only the final row in the matrix $L(i)$ has to be calculated. In [30], it is shown that only the computation of the last two elements in the final row, $l_{p,p-1}^{(i)}$ and $l_{p,p}^{(i)}$ is required and how the elements of the matrix $L(i)$ are used to efficiently select the optimal column, a_{k_p}.

3.2 Computing the Solution

Now that the basis elements to be used have been selected, in order to find the compact representation, it remains to find the coefficients associated with each of these elements. The

computation involved in doing this varies with the algorithm chosen and this is described in this section.

As has been stated in section 2.3, the residual at each step of the BMP algorithm does not represent the smallest residual obtainable, in general, when the signal is represented by the subset of basis vectors chosen. A final projection using a conjugate gradient descent algorithm [4], may be carried out to form this residual involving an extra computational load of $O((n+1)mr)$ multiplications. With this additional computation the BMP complexity becomes comparable to that of the MMP. However, because we may have reselected vectors, more iterations than are necessary may have been performed. It is also possible to carry out a projection after each iteration but if this is implemented then BMP is a more complex algorithm than MMP.

In MMP, to solve for the approximate solution vector, x_r, the QR Decomposition of the chosen vector set $S_r = [a_{k_1}, a_{k_2}, \cdots, a_{k_r}]$ is used. Therefore, $S_r = QR$ and the equation to be solved is $S_r x_r = b^{(0)} - b^{(r)}$ where $b^{(0)}$ is the searched for vector and $b^{(r)}$ is the remainder after the rth step. Similarly, in the case of both ORMP algorithms, the solution must be obtained using a backsolve.

3.3 Algorithm Complexity with D Variable

Computation Comparison with Variable Dictionary				
	Computation			
Algorithm	Step 0	Step p	Step r	Solution
BMP	$2mn$ mults + n divs	$\{(n-1)(m+2)+m\}$ mults	$2(n-1)$ mults	*(see section 3.2)
MMP	$2mn$ mults $+n$ divs	$\{(n-p)(m+2)+(2m+1)p+m+1\}$ mults + 1 div	$\{2(n-r+1)+(2m+1)(r-1)+m\}$ mults	$r(r-1)/2$ mults + r divs
ORMP I	$2mn$ mults	$\{(n-p)(m+5)+(2m+1)p-m+1\}$ mults $+\{n-p+1\}$ divs	$\{2(n-r+1)+(2m+1)(r-1)+m\}$ mults $+\{n-r+1\}$ divs	$r(r-1)/2$ mults $+r$ divs
ORMP II	$n(2m+1)$ mults $+(n+1)$ divs + 1 sqrt	$\{(n-p+1)(m+2+p)\}$ mults $+(n-p+3)$ divs + 1 sqrt	$\{(n-r+1)(m+r+2)\}$ mults $+(n-r+3)$ divs + 1 sqrt	$r(r-1)/2$ mults $+r$ divs

Table 1: The computation required for each of the basis selection methods is compared. Here ORMP I refers to the reduced complexity ORMP algorithm and ORMP II to the Cholesky ORMP algorithm. r is the sparsity required and the basis set consists of n vectors, each of dimension m.

The total complexity of each algorithm based on our discussion in section 3.1 and includ-

ing the computations required to find the solution as detailed in section 3.2, is summarized in Table 2.

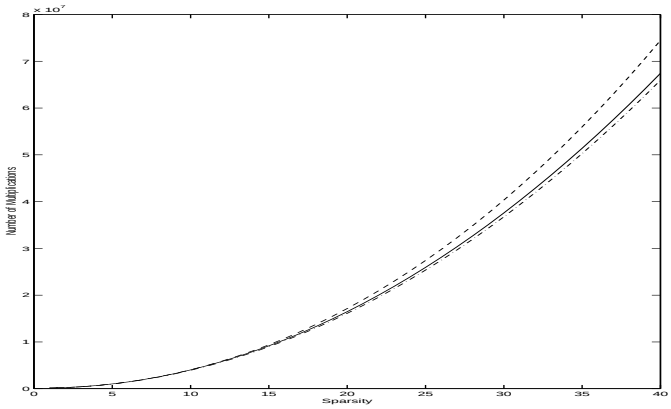

Figure 1: Comparison of Multiplications Required for Reduced ORMP(solid), Cholesky ORMP($--$),MMP($-.$)

Clearly, depending on the dimensions of the dictionary D and the sparsity required r, the amount of computation required by each algorithm will vary. For basis selection problems $m \ll n$ so we set the dimensions of the basis set as $m = 80, n = 1000$ to illustrate the complexity of each algorithm. Figure 1 shows a comparison of the number of multiplications involved in finding a sparse solution. This shows that the reduced complexity ORMP algorithm requires fewer multiplications than the Cholesky based ORMP. The MMP offers a further reduction in the number of multiplications required. From Table 2, a comparison of the computation in each step shows that while both ORMP algorithms require approximately $(n - p + 1)$ divisions at each step, the MMP requires just 1. Since n is often large, this saving in divides, which are cumbersome to implement in DSP hardware, is the big advantage in using the MMP.

The BMP computation is not included since without a final projection step its complexity is not of the same order as these algorithms. With the final projection, it has the same complexity as the MMP.

3.4 Algorithm Complexity with D Fixed

The BMP algorithm [4] was proposed in the context of certain wavelet bases where the inner products $a_l^H a_{k_p}$ as given in (12) require little or no computation. This can be viewed as equivalent to the case where a fixed dictionary D is used to decompose many different signals. The once-off formation of the inner products is attractive for such applications and in the light of this we re-consider the computational requirements of each of the three algorithms.

The columns, a_l, $l = 1, \cdots, n$, are normalized and the inner products $(a_i^H a_j), i = 1, \cdots, n; j = i, \cdots n$ are formed and stored. This substantially reduces the cost of performing the BMP, as explained in section 3.1.1, since just a single multiplication is now required to carry out recursion (12) for each value of $l, l \neq k_p$. In the Cholesky ORMP algorithm, initial computation of these inner products also leads to a low complexity iteration.

In the MMP and ORMP algorithms, as noted in sections 3.1.2, 3.1.3, a Gram-Schmidt orthogonalization has to be carried out as given in (10). Therefore, computing and storing the inner products $(a_i^H a_j), i = 1, \cdots, n;\ j = i, \cdots n$, does not make sense in implementing these algorithms. In the BMP, MMP and ORMP algorithms, the initial computation (denoted Step 0 in the table) consists of the formation of the inner products $a_l^H b_0$.

Computation Comparison with Fixed Dictionary				
	Computation			
Algorithm	Step 0	Step p	Step r	Solution
BMP	mn mults	$(n-1)$ mults	0 mult	*(see section 3.2)
MMP	mn mults	$\{(n-p)m + (2m+1)p\}$ mults + 1 div	$\{(2m+1)(r-1)+m\}$ mults	$r(r-1)/2$ mults + r divs
ORMP I	mn mults	$\{(n-p)(m+5)+(2m+1)p-m+1\}$ mults $+\{n-p+1\}$ divs	$\{2(n-r+1)+(2m+1)(r-1)+m\}$ mults $+\{n-r+1\}$ divs	$r(r-1)/2$ mults $+r$ divs
ORMP II	$n(m+1)$ mults $+(n+1)$ divs + 1 sqrt	$\{(n-p+1)(p+2)\}$ mults $+(n-p+3)$ divs + 1 sqrt	$\{(n-r+1)(r+2)\}$ mults $+(n-r+3)$ divs + 1 sqrt	$r(r-1)/2$ mults $+r$ divs

Table 2: The computation required for each of the basis selection methods is compared where the basis D is assumed fixed. Here ORMP I refers to the reduced complexity ORMP algorithm and ORMP II to the Cholesky ORMP algorithm. r is the sparsity required and the basis set consists of n vectors, each of dimension m.

The computation involved in each algorithm for a fixed dictionary D is summarized in Table 3 where again the Cholesky ORMP complexity is included for completeness. This table has the same format as Table 2 and a comparison of the tables shows that the computation of each algorithm has been reduced. It is evident that the BMP is by far the least computationally intensive of the algorithms, followed by the Cholesky ORMP implementation, then by the MMP and finally the reduced-complexity ORMP. The BMP and Cholesky ORMP have gained a computational advantage over the other algorithms at the cost of storing inner products. The computation of the MMP and ORMP is essentially unchanged, but the storage requirements for these algorithms have not been increased.

4 Performance in Determining Compact Representations

We now present a series of experiments which illustrate the performance of the BMP, MMP and ORMP algorithms.

4.1 Experiment 1.

In this experiment, the dictionary is created as a random $m \times n$ matrix A whose entries are Gaussian random variables with mean zero and variance 1. A sparse solution, x_s, with a specified number of nonzero entries r is then created; the indices of these r entries are random, and their amplitudes are random. The vector b is then computed as $b = Ax_s$ and the error tolerance, ϵ, is set to 10^{-6}. The experiment is repeated 100 times and a histogram is plotted. We define

$$\text{redundancy index } = \frac{\text{number of columns in solution}}{\text{number of columns used to generate } b\ (r)}. \tag{18}$$

An algorithm with a redundancy index histogram concentrated around 1 indicates a good procedure.

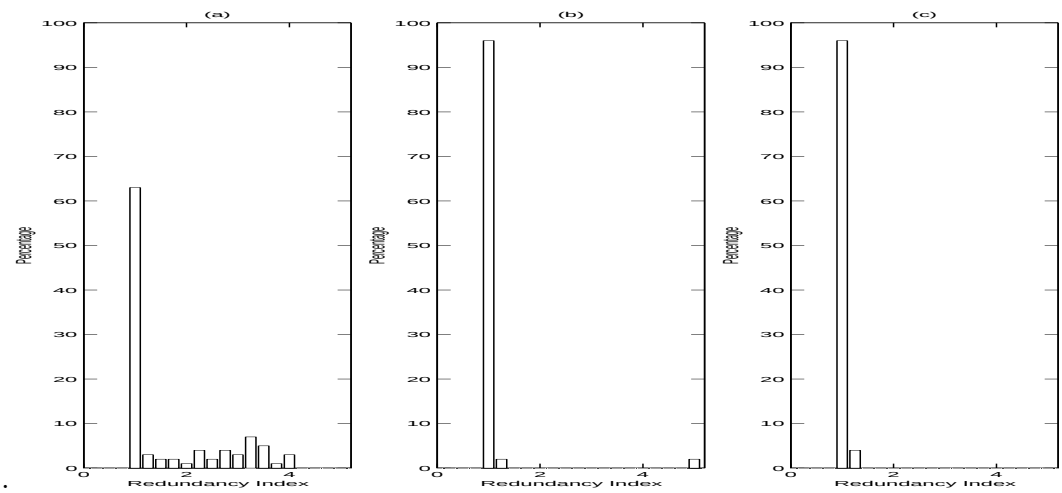

Figure 2: Results of Experiment 1 using (a) BMP, (b) MMP and (c) ORMP

The dimensions of the matrix A were set as 20×30 (other dimensions yielded similar results) for this experiment, and the BMP, MMP and ORMP were run with sparsity r set to 4. The results are shown in figures 2(a)–(c). From this experiment, we can conclude that the MMP algorithm gives a significant improvement over BMP and is comparable in performance to the ORMP algorithm. The results suggest that ORMP performs slightly better than MMP in finding a sparse solution set.

4.2 Experiment 2.

In this test case, the dictionary D is more structured and is chosen to be the rows of the Discrete Fourier Transform (DFT) matrix. The number of columns indicates the resolution in the frequency domain and the number of rows the length of the time series data. This matrix provides the opportunity to evaluate an algorithm when the columns are correlated and there is structure in the dictionary vectors. b is generated as before by selecting a few columns of D.

The dimensions chosen were $m = 32$ and $n = 128$ and ϵ was set to 10^{-6}. When an initial experiment was run with two widely spaced columns (6 and 28) chosen to form b each algorithm found the correct solution. However, due to re-selection of columns, BMP took 6 iterations to complete while MMP and ORMP completed in 2 iterations.

The experiment was re-run but this time the columns were selected close together (5 and 9). Figures 3(a)-(d) show the minimum 2-norm solution and the solution obtained using BMP, MMP and ORMP. The magnitudes of the non-zero coefficients of each of the 128 basis vectors which can be selected is plotted. All of the algorithms have their largest coefficients as 4 and 10 but this magnitude plot skews how the algorithms performed. The actual values of the largest coefficients, rather than the absolute values, are given in Table 4, along with the number of basis elements selected by each algorithm. In the case of BMP, MMP and ORMP, the number of basis elements selected equals the number of iterations performed by each algorithm. This experiment shows that all three forward sequential search algorithms can produce incorrect results under certain conditions. The drawback can be traced to the sequential nature of the basis selection process. Such situations indicate the possible utility of nonsequential methods such as those which have been suggested in [17, 25, 26].

Comparison of Algorithm Performance in DFT Experiment			
Algorithm	Component 1	Component 2	Elements Selected
BMP	Column 10: 0.794+0.732j	Column 4: 0.615-0.587j	29
MMP	Column 10: 0.664+0.664j	Column 4: 0.664-0.664j	32
ORMP	Column 10: 0.823+0.829j	Column 4: 0.792-0.787j	18
Min. 2-norm	Column 10: 0.199+0.183j	Column 4: 0.199-0.183j	128

Table 3: The largest components and the number of components in the minimum 2-norm solution and the solutions obtained using BMP, MMP and ORMP are compared.

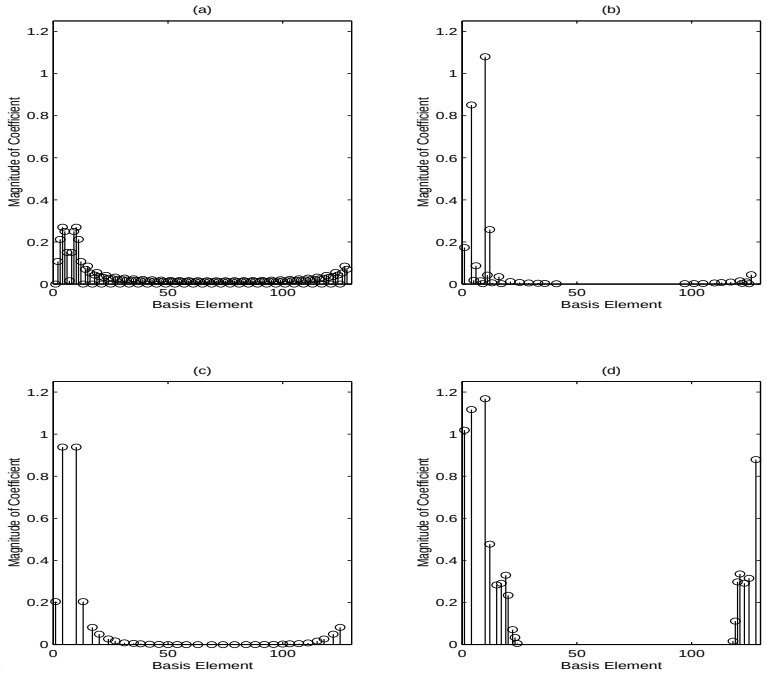

Figure 3: Plot of absolute value of non–zero coefficients obtained from Experiment 2 where b is formed from columns 5 and 9 - (a) minimum 2-norm solution, (b) BMP, (c) MMP, (d) ORMP.

4.3 Experiment 3.

Two data sets from [25] are used to compare the algorithms. The first signal used is the Gong waveform. This signal is zero up to the time $t = t_0$ and for $t > t_0$ is a decaying sinusoid; it is depicted in figure 4(a). The basis set used is a cosine packet dictionary based on a bell of width 16 samples. The basis elements in this dictionary are well localized in time and frequency and are computed with a quadratic filter-bank algorithm [4, 36]. Four elements from this dictionary are shown in the first column of figure 5. 256 samples of the

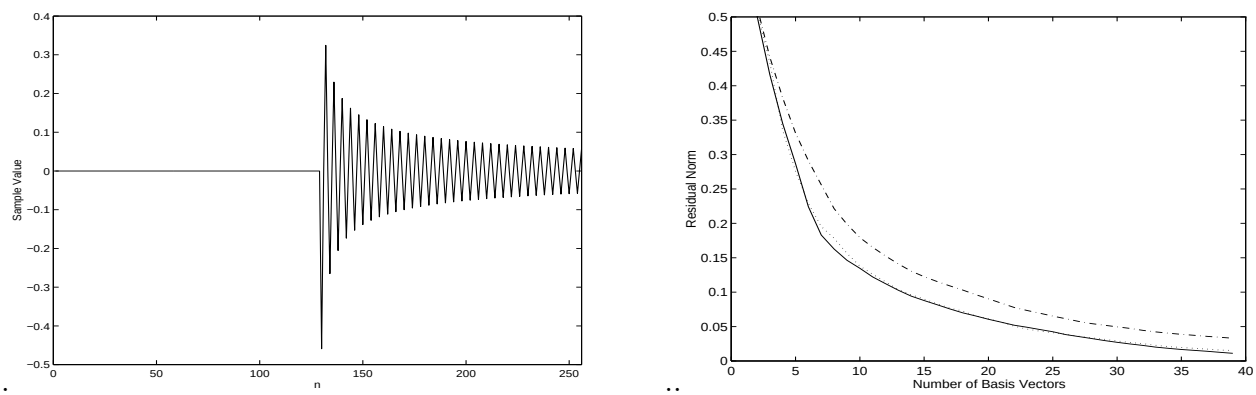

Figure 4: (a) Gong Waveform and (b) Plot of the Residual Norm vs Number of Basis Elements Selected: ORMP(solid), MMP($\cdots$), BMP($-.$)

waveform are used and 2304 vectors are used in the overcomplete basis. In figure 4(b), the fall off in the error ϵ is plotted as the number of basis vectors selected increases.

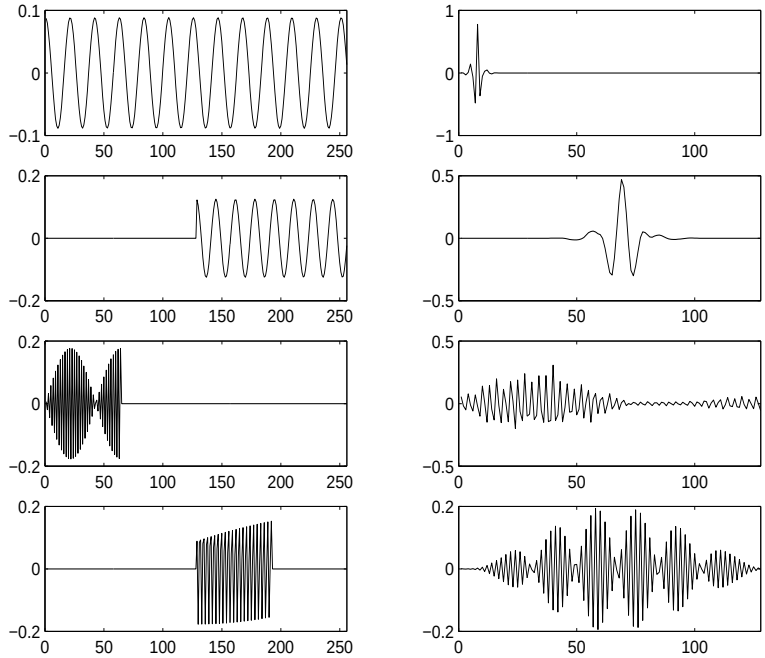

Figure 5: The four basis elements on the left are from a cosine packet dictionary (vector length 256). The four elements on the right are from a dictionary generated using filters for Symmlets with 8 vanishing moments (vector length 128).

The analyzing dictionary in the second case is generated by filters for a class of wavelets called Symmlets. Daubechies [37] (also see [38](Chapter 6)) introduced wavelets where the higher order moments of the wavelets are set to zero. This results in wavelet functions with a high degree of smoothness. Symmlets are the least asymmetric, compactly supported, wavelets for a given number of zero moments which, in our case, was set to 8. In the second column of figure 5, we give 4 sample vectors from this basis. The input signal consists of a linear combination of elements from this dictionary: a Dirac, a sinusoid and 4 mutually orthogonal wavelet packet atoms. It is shown in figure 6(a); this is referred to as "Carbon" in [25]. The number of samples of the input was chosen to be 128 and the number of basis elements used was 1024 so that once again the basis is overcomplete. In figure 6(b), the residual error ϵ is plotted as basis elements are added to the set used to represent this signal.

The three algorithms were run on these two real-world examples. The performance of the algorithms which emerged using the more artificial data in Experiment 1 is re-emphasized in this experiment. Clearly, the MMP and ORMP algorithms offer a performance advantage over the BMP algorithm. In figure 4(b), the ORMP is better than the MMP but its performance advantage is very small; in figure 6(b), the ORMP algorithm performs significantly

better than BMP and MMP where we select more than 6 basis vectors. As outlined in section 3.3, the price paid for this advantage is an increase in complexity.

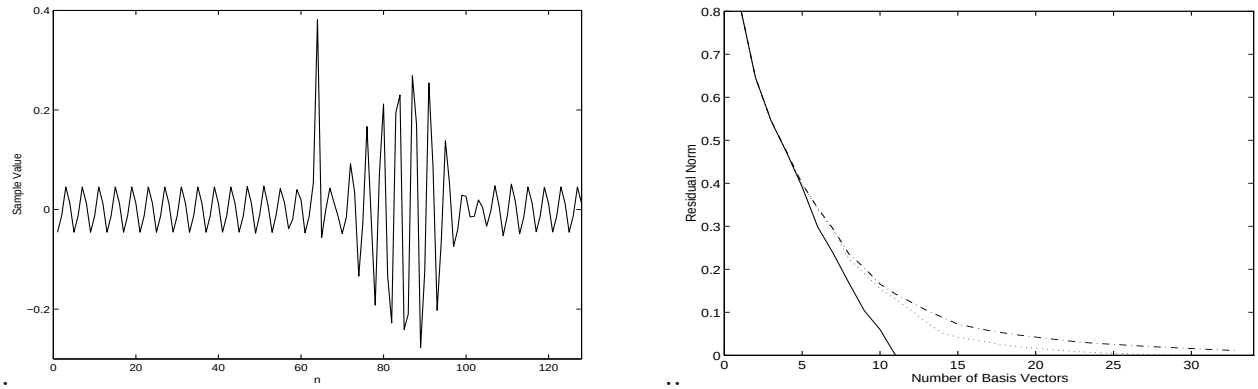

Figure 6: (a) Carbon Waveform and (b) Plot of the Residual Norm vs Number of Basis Elements Selected: ORMP(solid), MMP($\cdots$), BMP($-.$)

5 Conclusion

In this paper, the complexity and performance of forward sequential algorithms in best basis selection problems have been analyzed. The BMP, MMP and ORMP procedures were first presented and, in the algorithm descriptions, we drew attention to the re-selection problem which occurs in the BMP and how this is avoided in the selection procedures of the MMP and ORMP algorithms.

The computation involved in each algorithm was considered and more efficient implementations were detailed. In particular, the complexity of the ORMP for underdetermined basis sets has been substantially reduced in comparison to implementations presented elsewhere. The basis set may either be fixed or variable, depending on the application, and the impact of this on the complexity of the algorithms was examined.

Three experiments were presented to show how the algorithms perform. The results definitively show that the performance of the BMP lags behind that of the MMP and ORMP, while the ORMP performs better than MMP.

If the best possible subset is required, in the case where the basis is variable, then we must recommend the ORMP in its reduced complexity form as introduced here. However, in a situation where lower complexity is desirable, in particular where the number of divides should be as small as possible, the MMP is to be recommended at the cost of a very slight degradation in performance. If the basis set is fixed, the BMP is a very simple procedure to implement; the Cholesky based ORMP algorithm has the lowest complexity of the other algorithms in this case and offers much better performance.

References

[1] HOCKING, R.R. and LESLIE, R.N.: 'Selection of the Best Subset in Regression Analysis', *Technometrics*, 1967, **9**, pp. 531-540

[2] LaMOTTE, L.R. and HOCKING, R.R.: 'Computational Efficiency in the Selection of Regression Variables', *Technometrics*, 1970, **12**, pp. 83-93

[3] FRIEDMAN, J.H. and STUETZLE, W.: 'Projection Pursuit Regression', *J. Amer. Statist. Assoc.*, 1981, **76**, pp. 817-823

[4] MALLAT, S.G. and ZHANG, Z.: 'Matching Pursuits with Time-Frequency Dictionaries', *IEEE Trans. on Signal Processing*, 1993, **41**, (12), pp. 3397-415

[5] WICKERHAUSER, M.V.: 'Adapted Wavelet Analysis from Theory to Software', (A.K. Peters, Wellesley, MA, 1994)

[6] DONOHO, D.: 'On Minimum Entropy Segmentation' in CHUI, C.K., MONTEFUSCO, L. and PUCCIO, L. (Eds.): 'Wavelets: Theory, Algorithms and Applications' (Academic Press, Inc., 1994), pp. 233-69.

[7] GRIBONVAL, R., BACRY, E., MALLAT, S., DEPALLE, P.R., and RODET, X: 'Analysis of sound signals with high resolution matching pursuit', Proceedings of Third International Symposium on Time-Frequency and Time-Scale Analysis, TFTS-96, June 1996, Paris, France, pp. 125-128

[8] GOODWIN, M. and VETTERLI, M.: 'Atomic decompositions of audio signals', Proceedings of Workshop on Applications of Signal Processing to Audio and Acoustics, Oct. 1997, New York, USA, pp. 4-7

[9] BERGEAUD, F. and MALLAT, S.: 'Matching pursuit of images', Proceedings of International Conference on Image Processing, Oct. 1995, Los Alamitos, CA, USA, (1), pp. 53-6

[10] NEFF, R. and ZAKHOR, A.: 'Very low bit-rate video coding based on matching pursuits', *IEEE Transactions on Circuits and Systems for Video Technology*, 1997, **7**, (1), pp. 158-71

[11] RABIEE, H.R., KASHYAP, R.L. and SAFAVIAN, S.R.: 'Adaptive multiresolution image coding with matching and basis pursuits', Proceedings of International Conference on Image Processing, Sept. 1996, Lausanne, Switzerland, (1), pp. 273-6

[12] RAO, B.D.: 'Signal Processing with the Sparseness Constraint', Proceedings of the International Conference on Acoustics, Speech and Signal Processing, ICASSP '98, May 1998, Seattle, WA, USA, (III), pp. 1861-1864

[13] GORODNITSKY, I.F., GEORGE, J.S. and RAO, B.D.: 'Neuromagnetic Source Imaging with FOCUSS: a recursive weighted minimum norm algorithm', *Journal of Electroencephalography and Clinical Neurophysiology*, 1995, **95**, (4), pp. 231-251

[14] LEE, H., SULLIVAN, D.P. and HUANG, T.H.: 'Improvement of discrete band-limited signal extrapolation by iterative subspace modification', Proceedings of the International Conference on Acoustics, Speech and Signal Processing, ICASSP '87, Apr. 1987, Dallas, TX, USA, (3), pp. 1569-72

[15] CABRERA, S.D. and PARKS, T.W.: 'Extrapolation and spectral estimation with iterative weighted norm modification', *IEEE Trans. on Signal Processing*, 1991, **39**, (4), pp. 842-51

[16] GORODNITSKY, I.F. and RAO, B.D.: 'A Recursive Weighted Minimum-Norm Algorithm: Analysis and Applications', Proceedings of the International Conference on Acoustics, Speech and Signal Processing, ICASSP '93, Apr. 1993, Minneapolis, MN, USA, (3), pp. 456-9

[17] GORODNITSKY, I.F. and RAO, B.D.: 'Sparse Signal Reconstructions from Limited Data using FOCUSS: A Re-weighted Minimum Norm Algorithm', *IEEE Trans. on Signal Processing*, 1997, **45**, (3), pp. 600-616

[18] NATARAJAN, B.K.: 'Sparse Approximate Solutions to Linear Systems', *SIAM Journal on Computing*, 1995, **24**, (2), pp. 227-234

[19] CHENG, E.S., CHEN, S. and MULGREW, B.: 'Efficient Computational Schemes for the Orthogonal Least Squares Learning Algorithm', *IEEE Trans. on Signal Processing*, 1995, **43**, (1), pp. 373-376

[20] DUHAMEL, P. and RAULT, J.C.: 'Automatic Test Generation Techniques for Analog Circuits and Systems: A Review', *IEEE Trans. on Circuits and Systems*, 1979, **26**, pp. 411-440

[21] RAMSEY, J.B. and ZHANG, Z.: 'The Application of Waveform Dictionaries to Stock Market Index Data' in KADTKE, J. and KRAVTSOV, A. (Eds.): 'Predictablility of Complex Dynamical Systems' (Springer-Verlag, Berlin, 1996)

[22] FIELD, D.J.: 'What is the Goal of Sensory Coding', *Neural Computation*, 1994, **6**, (4), pp. 559-601

[23] OLSHAUSEN, B.A. and FIELD, D.J.: 'Emergence of simple-cell receptive field properties by learning a sparse code for natural images', *Nature*, 1996, **381**, (6583), pp. 607-609

[24] CHEN, S. and DONOHO, D.: 'Basis Pursuit', 28th Asilomar Conference on Signals, Systems and Computers, Nov. 1994, Pacific Grove, CA, USA, (1), pp. 41-44

[25] CHEN, S., DONOHO, D. and SAUNDERS, M.A.: 'Atomic Decomposition by Basis Pursuit', *Technical Report 479, Department of Statistics, Stanford*, May 1995. (available from http://www-stat.stanford.edu/∼donoho/Reports/1995/30401.pdf)

[26] RAO, B.D. and KREUTZ-DELGADO, K.: 'An Affine Scaling Methodology for Best Basis Selection', *IEEE Trans. on Signal Processing*, 1999, **47**, (1), pp. 187-200.

[27] CHEN, S. and WIGGER, J.: 'Fast Orthogonal Least Squares Algorithm for Efficient Subset Model Selection', *IEEE Trans. on ASSP*, 1995, **43**, (7), pp. 1713-15.

[28] PATI, Y.C., REZAIIFAR, R. and KRISHNAPRASAD, P.S.: 'Orthogonal Matching Pursuit: Recursive Function Approximation with Applications to Wavelet Decomposition', 27th Asilomar Conference on Signal, Systems, and Computers, Nov. 1993, Pacific Grove, CA, USA, (1), pp. 40-44

[29] DAVIS, G., MALLAT, S. and ZHANG, Z.: 'Adaptive time-frequency decompositions', *Optical Engineering*, 1994, **33**, (7), pp. 2183-91

[30] SINGHAL, S. and ATAL, B.S.: 'Amplitude Optimization and Pitch Prediction in Multipulse Coders', *IEEE Trans. on ASSP*, 1989, **37**, (3), pp. 317-327

[31] ADLER, J., RAO, B.D. and KREUTZ-DELGADO, K.: 'Comparison of Basis Selection Methods', 30th Asilomar Conference on Signals, Systems and Computers, Nov. 1996, Pacific Grove, CA, USA, (1), pp. 252-7.

[32] COTTER, S.F., MURTHI, M.N. and RAO, B.D.: 'Fast basis selection methods', 31st Asilomar Conference on Signals, Systems and Computers, Nov. 1997, Pacific Grove, CA, USA, (2), pp. 1474-78

[33] ATAL, B.S. and REMDE, J.R.: 'A new model of LPC excitation for producing natural-sounding speech at low bit rates', Proceedings of the International Conference on

Acoustics, Speech and Signal Processing, ICASSP '82, May 1982, Paris, France, (1), pp. 614-17

[34] OZAWA, K., ONO, S. and ARASEKI, T.: 'A study on pulse search algorithms for multipulse excited speech coder realization', *IEEE Journal on Selected Areas in Communications*, 1986, **4**, (1), pp. 133-41

[35] KAY, S.M.: 'Fundamentals of Statistical Signal Processing', (Prentice Hall, Englewood Cliffs, NJ, USA, 1993)

[36] RIOUL, O. and VETTERLI, M.: 'Wavelets and signal processing', *IEEE Signal Processing Magazine*, 1991, **8**, (4), pp. 14-38

[37] DAUBECHIES, I.: 'Ten Lectures on Wavelets', (SIAM, Philadelphia, PA, USA, 1992)

[38] BURRUS, G.S., GOPINATH, R.A. and GUO, H.: 'Introduction to Wavelets and Wavelet Transforms', (Prentice Hall, Upper Saddle River, NJ, USA, 1998)