ELSEVIER

Signal Processing 81 (2001) 1849–1864

SIGNAL PROCESSING

www.elsevier.com/locate/sigpro

Backward sequential elimination for sparse vector subset selection[☆]

S.F. Cotter, K. Kreutz-Delgado*, B.D. Rao

Electrical and Computer Engineering Department, University of California, San Diego, La Jolla, CA 92093-0407, USA

Received 27 January 2000; received in revised form 18 March 2001

Abstract

Selection of a subset of vectors from a larger dictionary of vectors arises in a wide variety of application areas. This problem is known to be NP-hard and many algorithms have been proposed for the suboptimal solution of this problem. The focus of this paper is the development of a backward sequential elimination algorithm wherein, starting from the full dictionary, elements are deleted until a subset of a desired size is obtained. In contrast to previous formulations, we start with an overcomplete dictionary of vectors which is often the problem faced in a signal representation context. Once enough vectors have been deleted to give a complete system, the algorithm is modified to allow further deletion of vectors. In addition, the derived algorithm gives access to the coefficients associated with each vector in representing the signal. This allows us to experiment with different criteria, including entropy-based and p-norm criteria, for selection of the vector to be deleted in each iteration. There is also the flexibility to combine criteria or to switch between criteria at a given stage of the algorithm. Following a series of simulations on a test-case system, we are able to conclude that the p-norm close to 1 performs best while the system considered is overcomplete. A minimum representation error criterion gives the best results once the system considered becomes undercomplete. The performance of the algorithm is also compared to that of forward selection algorithms on the test-case dictionary. © 2001 Elsevier Science B.V. All rights reserved.

Keywords: Subset selection; Sparsity; Backward elimination

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 [24,28,31,16]. This problem has also been the subject of much research interest in the recent signal processing literature. For example, much work has been done in constructing signal representation dictionaries which are collections of basic signals suitable for the decomposition of signals of interest. These dictionaries can then be used for compression of audio [22,18] and video signals [3,33,36]. These dictionaries are overcomplete in that they form a non-minimal spanning set, and so very many different representations of the same input signal are possible in terms of the dictionary elements. For the

☆This research was partially supported by the National Science Foundation Grant No. CCR-9902961.

*Corresponding author. Tel.: + 1-858-534-0688; fax: + 1-858-534-1004.

E-mail addresses: scotter@ece.ucsd.edu (S.F. Cotter), kreutz@ece.ucsd.edu (K. Kreutz-Delgado), brao@ece.ucsd.edu (B.D. Rao).

purposes of compression, we wish to use a small ("sparse") subset of vectors from the dictionary to represent the signal, and hence require algorithms which find such a representation.

Subset selection problems also arise in many different application areas [38], making a detailed analysis of basis selection algorithms an important study. For instance, these algorithms have been applied to biomagnetic inverse problems [19], bandlimited extrapolation and spectral estimation [29,4], direction of arrival estimation [20,21], functional approximation [32,9], failure diagnosis [14], and stock market analysis [37]. More recently, it has been argued that overcomplete representations and basis selection have a role in the coding of sensory information in biological systems [15,34].

The subset selection problem has been shown to be NP-hard [32] but many approaches have been proposed which give a suboptimal solution to this problem. Most commonly, a search for a sparse representation (so-called because most of the dictionary vectors are not utilized in representing the signal) is based on a *forward* search through the dictionary [32,8–10,30,2,35,12,41,1]. Elements from the overcomplete dictionary are selected one after the other until we have successfully represented the vector to within some threshold, typically set by the magnitude of the error between the signal and its representation. As the name implies, these algorithms proceed by sequentially adding elements to a set of vectors which will be used to represent the signal. Simple procedures were implemented initially [30,2] while more complex algorithms were developed in [32,35,12,41,1,8–10], yielding improved results. Other approaches have also been suggested such as those based on minimizing diversity measures including functionals whose minimization promotes sparsity like the ℓ_1 norm [6,7] or the more general $\ell_{(p\leqslant 1)}$ norm [21,39,25–27]. Yet another approach was introduced in [11,23] where the search is based on a backward elimination but for a complete or undercomplete (i.e., *non*-overcomplete) dictionary. In this case, elements are sequentially deleted from a dictionary to obtain a sparse solution. A simpler algorithm was formulated in [40] which improved computationally on the work in [11,23], but is still limited to the undercomplete case.

In this paper, we consider such backward elimination algorithms. However, in contrast to [11,23,40], where an overdetermined set of equations was considered, we start from an underdetermined system, which is usually the problem faced in signal representation. Once we arrive at a complete system, the problem then becomes equivalent to that addressed in [11,23,40].

The intuition behind our approach is detailed and recursions are developed so that intermediate solutions are available as well as the representation error. In our algorithm, the system considered remains underdetermined as a column chosen for elimination is replaced by a column of zeros. The algorithm development is divided into two parts. In the first part, we have a system of full row rank and the recursions developed here are novel. In the second part, the system is rank deficient and, while working from a different formulation of the problem, the final recursions developed in this phase are essentially identical to those of [40]. This will be expanded upon in Section 4.

In contrast to previous papers [11,23,40], we consider criteria other than the minimum representation error in choosing which elements are deleted from the dictionary. This is necessary while the system has full row rank as the representation error is zero but we also experiment with using these criteria in the rank deficient case. We demonstrate through simulation on a test-case system that the performance of the backward elimination algorithm is as good as that obtained using a forward selection algorithm. However, for highly underdetermined systems, the algorithm is computationally expensive compared to forward selection methods and, in practice, it is more likely that it would be used to prune a sparse solution obtained more cheaply using alternative algorithms.

The outline of the paper is as follows. In Section 2, we outline the problem of finding a subset to efficiently represent the input signal. We start with an underdetermined system of full row rank in Section 3 and develop an algorithm to delete elements from the dictionary. Once the system becomes rank deficient, we need to modify our algorithm so that greater sparsity can be achieved in the solution. This is developed in Section 4. The computational complexity of the backward elimination algorithm is considered in Section 5. We then investigate how these algorithms perform on a test-case system in Section 6. In particular, since we have purposely designed the algorithms to give us access to intermediate solutions and errors, we explore how these may be used as selection criteria in determining which basis vectors to delete from the dictionary. From these results, we are able to recommend an implementation which achieves a sparse signal representation. We give the conclusions of our work in Section 7.

2. Problem statement and notation

The problem can be formulated as the linear inverse problem

$$Ax = b, \tag{1}$$

where the columns of the full row rank matrix $A \in \mathscr{C}^{m\times n}$ represent the basis vectors from the dictionary, $b \in \mathscr{C}^m$ is the target vector to be sparsely represented by a subset of columns of A and $x \in \mathscr{C}^n$ is the solution vector. Since the dictionary typically has many more vectors than are required to represent b, it is overcomplete, i.e., $n \gg m$, and the system in (1) is underdetermined. A solution to this problem is said to be sparse when a significant number of components of x are zero.

A commonly used procedure to obtain a unique solution to the underdetermined system in (1) is to find the minimum 2-norm solution to this problem, i.e.,

$$x^{\dagger} = \arg\min_{x} \{\|x\| \,|\, Ax = b\}, \tag{2}$$

where $\|\cdot\|$ denotes the standard 2-norm. The solution, $x^{\dagger}$, is evaluated as $x^{\dagger} = A^{\dagger}b$, where $A^{\dagger} = A^{\mathrm{H}}(AA^{\mathrm{H}})^{-1}$ is the pseudoinverse of A. This solution lies in the space $\mathscr{R}(A^{\mathrm{H}})$ but is generally not sparse.

The starting point for the development of our algorithm is the minimum norm solution, as given in (2). Therefore, initially, $x^{\dagger}$ must be calculated which implies that the matrix $A^{\dagger}$ must be formed. Then the elements in $x^{\dagger}$ are sequentially zeroed out to give a sparse solution with the remaining non-zero elements modified if necessary. For example, in the first iteration, if the ith element is to be zeroed out, the minimum norm solution is sought for the modified system

$$A_1 x = b, \quad A_1 \triangleq A(I - e_i e_i^{\mathrm{H}}), \tag{3}$$

where e_i is the canonical unit vector with "1" in the ith position. Essentially, in forming A_1, we have discarded the ith column of A by replacing it with a column of zeros. The minimum norm solution to (3) is denoted by x_1 and is obtained by an appropriate modification of $x^{\dagger}$. By recursively implementing this procedure a sparse solution is obtained for (1). At the lth stage, we have A_l which is A with l columns deleted. If $\mathrm{rank}(A_l) = m \leqslant n$, then we still have a full row rank underdetermined system and x_l can be found which satisfies

$$A_l x_l = b. \tag{4}$$

This process can be iterated as long as $\mathrm{rank}(A_l)$ continues to be m. However, once $\mathrm{rank}(A_l) < m$ the system is rank deficient and the equality in (4) may no longer hold, i.e., in general $\|A_l x - b\| \geqslant 0 \ \forall x$, and the procedure must be modified accordingly. We consider these two cases in Sections 3 and 4 respectively.

We now introduce some notation which will be useful in describing the algorithm. The set of deleted columns at the lth iteration is stored in the set $D_l = \{k_1, k_2, \ldots, k_l\}$, where k_j is used to denote the column deleted in the jth iteration of the algorithm. For convenience, the selected unit vector in this iteration will be denoted as $\hat{e}_j \triangleq e_{k_j}$. The projection onto $\mathrm{span}(e_{k_j}^{\perp})$ is given by $P_j = (I - \hat{e}_j \hat{e}_j^{\mathrm{H}})$ with $P_0 = I$. The product of a sequence of projections is given as $P_{1,j} = \prod_{k=1}^{j} P_k = P_1 P_2 \cdots P_j$. Applying these projections to the matrix A, we introduce $A_i = AP_{1,i}$ where $A_0 = A$. The solution of $\arg\min_x \{\|x\| \,|\, A_i x = b\}$ is given by x_i and x_0 is initialized to $x_0 = x^{\dagger}$.

3. Backward elimination—full row rank case

3.1. Algorithm overview

We first consider the case where we must solve a full row rank (i.e., $\mathrm{rank}(A) = m$) underdetermined system and introduce three lemmas which will be useful in deriving an algorithm for the deletion of columns from the matrix A.

Lemma 1. *Let $A \in \mathscr{C}^{m\times n}$ have rank m and suppose that, in the first iteration, the ith column has been chosen for deletion. Then, the rank of $A_1 = AP_1 = A(I - e_i e_i^{\mathrm{H}})$ is m iff $(I - A^{\dagger}A)e_i \neq 0 \Leftrightarrow e_i^{\mathrm{H}} A^{\dagger} A e_i \neq 1$.*

Proof. This is based on the observation that $\mathrm{rank}(A_1) = m \Leftrightarrow \det(A_1 A_1^{\mathrm{H}}) \neq 0$, and the identity $\det(M - NN^{\mathrm{H}}) = \det(M)\det(I - N^{\mathrm{H}}M^{-1}N)$ [13],

$$\begin{aligned}\det(A_1 A_1^{\mathrm{H}}) &= \det(AP_1A^{\mathrm{H}})\\ &= \det(AA^{\mathrm{H}} - Ae_ie_i^{\mathrm{H}}A^{\mathrm{H}})\\ &= \det(AA^{\mathrm{H}})\det(1 - e_i^{\mathrm{H}}A^{\mathrm{H}}(AA^{\mathrm{H}})^{-1}Ae_i)\\ &= \det(AA^{\mathrm{H}})(1 - e_i^{\mathrm{H}}A^{\dagger}Ae_i)\end{aligned}$$

yielding the condition $(1 - e_i^{\mathrm{H}}A^{\dagger}Ae_i) \neq 0$ iff $\det(A_1A_1^{\mathrm{H}}) \neq 0$. We then rewrite this as

$$1 - e_i^{\mathrm{H}}A^{\dagger}Ae_i \neq 0 \;\Leftrightarrow\; e_i^{\mathrm{H}}(I - A^{\dagger}A)e_i \neq 0$$

and letting

$$y = (I - A^{\dagger}A)e_i,$$

$$e_i^{\mathrm{H}}y = y_i \neq 0 \Rightarrow y \neq 0 \Rightarrow (I - A^{\dagger}A)e_i \neq 0. \qquad \square$$

Lemma 1 tells us that the projection of e_i onto $\mathcal{N}(A)$ must be non-zero and so $e_i \notin \mathscr{R}(A^{\mathrm{H}})$. This agrees with our intuition since if it were true that $e_i \in \mathscr{R}(A^{\mathrm{H}})$ (e.g., imagine that e_i^{H} happens to be a row of A) then throwing away the ith column would mean a loss of row rank and the equality of (1) would, in general, be disrupted. In the next lemma, $\hat{e}_i$ is defined as described at the end of Section 2.

Lemma 2. *If $\hat{e}_{i+1} \neq \hat{e}_1, \ldots, \hat{e}_i \notin \mathscr{R}(A_i^{\mathrm{H}})$ where $A_i = AP_{1,i}$, then $\hat{e}_{i+1}$ is not in the row space of $A_l, l = 0, \ldots, i$. In particular, $\hat{e}_{i+1} \notin \mathscr{R}(A^{\mathrm{H}})$.*

Proof. The contrapositive statement is $\hat{e}_{i+1} \in \mathscr{R}(A^{\mathrm{H}}) \Rightarrow \hat{e}_{i+1} \in \mathscr{R}(A_l^{\mathrm{H}})$, $l = 1, \ldots, i$, which is proved as follows:

if $\hat{e}_{i+1} = A^{\mathrm{H}}\lambda \in \mathscr{R}(A^{\mathrm{H}})$ then $\hat{e}_{i+1}$

$$\begin{aligned}&= \prod_{k=1}^{l}(I - \hat{e}_k\hat{e}_k^{\mathrm{H}})\hat{e}_{i+1} = \prod_{k=1}^{l}(I - \hat{e}_k\hat{e}_k^{\mathrm{H}})A^{\mathrm{H}}\lambda\\ &= A_l^{\mathrm{H}}\lambda \Rightarrow \hat{e}_{i+1} \in \mathscr{R}(A_l^{\mathrm{H}}). \qquad \square\end{aligned}$$

Lemma 2 provides a link between the solutions of (1) and (4). If the chosen column does not disrupt the possibility of equality in (4) (i.e., does not result in $\mathrm{rank}(A_l) < m = \mathrm{rank}(A)$) then the minimum norm solution to (4) is a solution to (1). As we form the reduced matrices A_i, we want to zero out columns while preserving rank as long as possible and iteratively produce minimum norm solutions which still satisfy (1). By iterating on Lemma 1, we can write Lemma 3 which shows how to proceed so that these two goals are satisfied at each step.

Lemma 3. *$\forall \hat{e}_j = e_{k_j} \in \{e_{k_1}, \ldots, e_{k_l}\}$ such that $(I - A^{\dagger}A)\hat{e}_j \neq 0, j = 1, \ldots, l$, and recalling that $A_j = AP_{1,j} = AP_1 \cdots P_j$ with $\mathrm{rank}(A) = m$, we have $\mathrm{rank}(A_j) = m$.*

We will now proceed to use the above lemmas in deriving an algorithm which produces a sparse solution. By assumption the rank of the dictionary, A, is m and so it is clear that we can remove at most $v = (n - m)$ columns from the matrix A while still being able to produce a minimum norm solution which satisfies (1) with equality. Since the algorithm will iteratively remove these columns to form the reduced system, the first iteration of the algorithm is now detailed.

Algorithm.

- Choose the first column to be eliminated from the matrix A, i.e., choose $\hat{e}_1 = e_{k_1}$ such that $(I - A_0^{\dagger}A_0)\hat{e}_1 \neq 0$, where $A_0 = A$.
- Form $A_1 = A_0P_1 = A_0(I - \hat{e}_1\hat{e}_1^{\mathrm{H}})$, which replaces the chosen column by a column of zeros. Note that $\mathrm{rank}(A_1) = m$ because of the choice of $\hat{e}_1$.
- Find

$$x_1 = \arg\min_x \|x\| \quad \text{s.t. } A_1x = b.$$

The solution is found as

$$x_1 = A_1^{\dagger}b, \quad A_1^{\dagger} = A_1^{\mathrm{H}}(A_1A_1^{\mathrm{H}})^{-1}. \tag{5}$$

Note that because the solution is minimum norm, the k_1th component of x_1 will be zero, corresponding to the zeroed-out column in A_1. Thus

$(I - \hat{e}_1\hat{e}_1^{\mathrm{H}})x_1 = x_1$ and $b = A_1x_1 = A(I - \hat{e}_1\hat{e}_1^{\mathrm{H}})x_1 = Ax_1$ showing that indeed x_1 is a solution to (1), but with greater sparsity than the minimum norm solution to (1). For the general case, this will be shown rigorously.

3.2. Efficient algorithm implementation

In the initialization of the algorithm, the pseudo-inverse, $A^\dagger$, and minimum norm solution, $x^\dagger$, as given in (2), must be computed. Therefore, to efficiently implement the algorithm of Section 3.1, the pseudo-inverse needs to be updated with a low computational cost. This can be accomplished through the use of the following lemma, the proof of which is given in Appendix A.1.

Lemma 4. *With the choice of* $\hat{e}_1$ *such that* $(I - A^\dagger A)\hat{e}_1 \neq 0$ *so that* $\mathrm{rank}(A_1) = m$, $A_1^\dagger$ *can be written as* $A_1^\dagger = (I - K_1\hat{e}_1^{\mathrm{H}})A^\dagger$ *where* $A^\dagger = A^{\mathrm{H}}(AA^{\mathrm{H}})^{-1}$, $K_1 = (I - A^\dagger A)\hat{e}_1/(1 - \hat{e}_1^{\mathrm{H}}A^\dagger A\hat{e}_1)$ *and* $AK_1 = 0$.

Now, we establish a connection between the new minimum norm solution x_1 and the solution to the original system as given in (1) and this lemma is proved in Appendix A.2.

Lemma 5. *For* x_1 *given by* $x_1 = A_1^\dagger b$, *where* $A_1^\dagger$ *has been defined in Lemma* 4, *we have* $Ax_1 = b$. *Furthermore*, $P_1x_1 = (I - \hat{e}_1\hat{e}_1^{\mathrm{H}})x_1 = x_1$, *showing that* x_1 *is zero in the* k_1*th position.*

We can write the new solution $x_1 = A_1^\dagger b = A^\dagger b - K_1\hat{e}_1^{\mathrm{H}}A^\dagger b$ as

$$x_1 = x_0 + \delta x_1 \text{ where } x_0 = x^\dagger \tag{6}$$

and

$$\delta x_1 = x_1 - x_0 = -K_1\hat{e}_1^{\mathrm{H}}x_0 \in \mathcal{N}(A).$$

Since $\delta x_1 \perp x_0$,

$$\|x_1\|^2 = \|x_0\|^2 + \|\delta x_1\|^2 = \|x^\dagger\|^2 + \|\delta x_1\|^2. \tag{7}$$

This shows that $\|\delta x_1\|^2$ can be interpreted as the "cost" of zeroing out the k_1th component of $x^\dagger$ to obtain a more sparse solution. More generally at the lth iteration, we need to solve

$$x_{l+1} = \arg\min_x \|x\| \quad \text{s.t. } A_{l+1}x = b. \tag{8}$$

We have $A_{l+1}x_{l+1} = A_lP_{l+1}x_{l+1} = b$ with $\mathrm{rank}(A_l) = m$, and $\hat{e}_{l+1}$ chosen such that $(I - A_l^\dagger A_l)\hat{e}_{l+1} \neq 0$. We can therefore iterate on the solution of the problem as stated in (5) to solve (8). So, making the substitutions in Lemma 4 of $0 \to l$ and $1 \to (l+1)$, we obtain

$$\begin{aligned} x_{l+1} &= A_{l+1}^\dagger b = (I - K_{l+1}\hat{e}_{l+1}^{\mathrm{H}})x_l \\ &= (I - K_{l+1}\hat{e}_{l+1}^{\mathrm{H}})(I - K_l\hat{e}_l^{\mathrm{H}})x_{l-1} \\ &= \prod_{i=l+1}^{1} (I - K_i\hat{e}_i^{\mathrm{H}})x_0, \quad x_0 = x^\dagger. \end{aligned} \tag{9}$$

To find how this new solution vector is related to the original set of equations, we first establish the value of AK_{j+1} since the value of AK_1 was required before.

Lemma 6. $AK_{j+1} = 0$, $\forall j = 1, \ldots, l$.

Proof.

$$\begin{aligned} AK_{j+1} &= A(I - A_j^{\mathrm{H}}(A_jA_j^{\mathrm{H}})^{-1}A_j)\hat{e}_{j+1} \\ &= A(I - P_{1,j}A^{\mathrm{H}}(A_jA_j^{\mathrm{H}})^{-1}AP_{1,j})\hat{e}_{j+1} \\ &= A\hat{e}_{j+1} - (AP_{1,j})P_{1,j}A^{\mathrm{H}}(A_jA_j^{\mathrm{H}})^{-1}A\hat{e}_{j+1} \\ &= (I - A_jA_j^{\mathrm{H}}(A_jA_j^{\mathrm{H}})^{-1})A\hat{e}_{j+1} = 0. \quad \square \end{aligned}$$

With this lemma established, the next lemma follows as a direct consequence.

Lemma 7. *For* $x_j, j = 1, \ldots, l+1$, *as generated in* (9), $Ax_j = b$ (i.e. *all* x_j *which solve* $A_jx_j = b$ *also solve* $Ax_j = b$).

From the lemmas above, it is clear that for the lth iteration, we have

$$x_{l+1} = x_l + \delta x_{l+1}, \quad \delta x_{l+1} = -K_{l+1}\hat{e}_{l+1}^{\mathrm{H}}x_l, \tag{10}$$

where both x_{l+1} and x_l solve $Ax = b$ and (as seen for $l = 0$)

$$\delta x_{l+1} = x_{l+1} - x_l \in \mathcal{N}(A). \tag{11}$$

Table 1
Full row rank system: algorithm for backward elimination

```
Initializations: $A \in \mathscr{C}^{m \times n}$, rank$(A) = m$; $A_0 = A$; $A_0^\dagger = A^\dagger = A^{\mathrm{H}}(AA^{\mathrm{H}})^{-1}$; $x_0 = x^\dagger = A^\dagger b$; $\delta x_0 = 0$; $\Delta x_0 = 0$; $D_0 = \emptyset$
loop $l \geqslant 0$ until Select = False
    • Select $= \{k_{l+1} \notin D_l, \hat{e}_{l+1}^{\mathrm{H}} A_l^\dagger A_l \hat{e}_{l+1} \neq 1\}$ and CRITERIA (see text).
    • $D_{l+1} = D_l \cup \{k_{l+1}\}$
    • $K_{l+1} = \dfrac{(I - A_l^\dagger A_l)\hat{e}_{l+1}}{1 - \hat{e}_{l+1}^{\mathrm{H}} A_l^\dagger A_l \hat{e}_{l+1}}$; $A_{l+1}^\dagger = (I - K_{l+1}\hat{e}_{l+1}^{\mathrm{H}})A_l^\dagger$
    • $\delta x_{l+1} = -K_{l+1}\hat{e}_{l+1}^{\mathrm{H}} x_l$
    • $\Delta x_{l+1} = \Delta x_l + \delta x_{l+1}$
    • $x_{l+1} = x^\dagger + \Delta x_{l+1}$
end loop
```

So by iterating (10) and (11) we obtain

$$x_{l+1} = x_0 + \Delta x_{l+1} = x^\dagger + \Delta x_{l+1} \tag{12}$$

where

$$\Delta x_{l+1} = \sum_{j=1}^{l+1} \delta x_j \in \mathcal{N}(A), \quad \Delta x_0 = 0$$

and

$$\|x_{l+1}\|^2 = \|x^\dagger\|^2 + \|\Delta x_{l+1}\|^2.$$

In deriving this algorithm, the solution vector has been calculated at every stage. Moreover, we have access to the incremental change in the solution and also the change in the solution from its initialization as the minimum 2-norm solution as given in (2). This allows us to examine many different criteria in choosing the next column to be deleted from the dictionary as we will see in Section 6. However, we first summarize the algorithm we have so far in Table 1 before moving on to consider the criteria which may be useful in finding a sparse solution.

3.3. Deletion criteria

In [11,23,40], an overdetermined system of equations was considered and the error in representing b was used as a criterion for eliminating columns. However, in our derivation, we have assumed an underdetermined system of equations. Since the representation error will be zero while the system has full row rank, this cannot be used as a criterion for deleting columns. We have derived an expression which recursively gives the solution vector in each iteration. This gives us flexibility in choosing a criterion to use for selection of the next column k_{l+1} to be deleted. Some of the possible deletion criteria which we could use are given in Table 2.

Table 2
Deletion criteria

- Random selection of k_{l+1}.
- $k_{l+1} = \arg\min_{k_{l+1} \notin D_l} \|\delta x_{l+1}\|$, i.e., the minimum perturbation, as measured in the 2-norm, of the *current* solution is used in selecting the column.
- $k_{l+1} = \arg\min_{k_{l+1} \notin D_l} \|\Delta x_l + \delta x_{l+1}\|$, i.e., we perturb the *initial* solution by the minimum amount possible as measured in the 2-norm.
- Entropy-like measures, such as the Shannon entropy-like function defined as

$$H_S(x) = -\sum_{i=1}^{n} \tilde{x}(i) \ln \tilde{x}(i) \quad \text{where } \tilde{x}(i) = \frac{|x(i)|^2}{\|x\|_2^2}. \tag{13}$$

- p norms where $p \leqslant 1$:

$$\|x\|_p = \sum_{i=1}^{n} |x(i)|^p \tag{14}$$

The p-norm and Shannon entropy-like functions given in Table 2 are known to be good concentration measures which preferentially select sparse solutions [39,25].

4. Backward elimination—rank deficient case

Since A is assumed to have full row rank, the algorithm outlined in Section 3 will break down when $(n-m)$ columns have been zeroed out from A. Effectively, these columns have been removed from A so that we no longer have a full row rank system. Therefore, the algorithm must be modified in order to remove more columns from A. This is addressed in the following subsections.

4.1. Algorithm overview

We assume that $v = (n-m)$ columns have been removed from A and that rank(A) is still m, i.e., the columns $D_v = \{k_1, k_2, \ldots, k_v\}$ have been replaced by columns of zeros in A. In order to achieve further sparsity in the solution vector, columns not in D_v must be removed. This is essentially the same problem as that considered in [11,23,40] (i.e., where the system is overdetermined). The criterion for deletion of columns in these papers was to choose the column which, when eliminated, resulted in the smallest increase in the representation error. In our approach, as was done in [40] but not [11,23], we additionally calculate the solution at each iteration. This allows us to experiment with different criteria, as given in Table 2, in choosing a column for elimination.

We continue our derivation for the rank deficient case in the same vein as was done in the previous section for the full row rank system. Therefore, we develop recursions for $A_{l+1}^{\dagger}$ and x_{l+1} for $l = v, \ldots, (n-1)$ as well as the representation error which is now no longer zero. The system of equations is still considered to be underdetermined with the columns chosen for deletion replaced by columns of zeros. Our main recursion is on $A_l^{\dagger}$ (Lemma 9) which requires the recursive computation of $(A_l^{\mathrm{H}} A_l)^{\dagger}$ in Lemma 10. A recursion was developed for the quantity $(A_l^{\mathrm{H}} A_l)^{-1}$ in [40] where the assumption is that A_l is full column rank but, since our development is different, alternate proofs are required. The recursions derived turn out to be essentially identical to those in [40]; the lemmas are given in the text with the proofs included for completeness in the appendices.

It is evident from Lemma 1 that the columns which remain must satisfy $(I - A_v^{\dagger} A_v)\hat{e}_{l+1} = 0$, which is equivalent to $\hat{e}_{l+1} \in \mathscr{R}(A_v^{\mathrm{H}})$. These columns will in general perturb the solution and so the set of equations, as in (4), can no longer hold with equality. Since error will be introduced, one possibility for selecting the column to be deleted from A is to find the column k_{l+1} which minimizes the representation error

$$\varepsilon_{l+1} \triangleq \|A_{l+1} x_{l+1} - b\|, \quad l = v, \ldots, n-1$$
$$\text{where } A_{l+1} = A_v P_{v+1,l+1}. \tag{15}$$

To search for these indices, we start with the matrix A_l which has been found at the final step of the full row rank algorithm, i.e., $l = v$.

The problem for subsequent values of l can be stated as finding

$$x_{l+1} = \arg\min_{x} \|A_{l+1} x - b\| = A_{l+1}^{\dagger} b,$$
$$l = v, \ldots, n-1, \tag{16}$$

assuming that A_l is available from the previous iteration. Recall that in this derivation, the columns are not deleted but are replaced by columns of zeros and so we always have an $m \times n$ matrix. The pseudo-inverse is therefore given as [45]

$$\begin{aligned} A_{l+1}^{\dagger} &= A_{l+1}^{\mathrm{H}} (A_{l+1} A_{l+1}^{\mathrm{H}})^{\dagger} \\ &= P_{l+1} A_l^{\mathrm{H}} (A_l P_{l+1} A_l^{\mathrm{H}})^{\dagger}, \end{aligned} \tag{17}$$

where $A_{l+1} A_{l+1}^{\mathrm{H}}$ is no longer invertible since A_{l+1} has less than m nonzero columns.

As was done through Lemma 4 of Section 3 for the full row rank system of equations, a recursion must be established for the computation of $A_{l+1}^{\dagger}$ to allow efficient implementation of this algorithm. This is detailed in the following section.

4.2. Efficient algorithm implementation

Once, more than $v = (n-m)$ columns have been removed from A, the matrices A_{l+1}, $l = v, \ldots, (n-1)$, are rank deficient, i.e., $\operatorname{rank}(A_{l+1}) < m$. The remaining columns form a linearly independent set of vectors and we establish the following lemma for each of these nonzero columns.

Lemma 8. *Let $A_{l+1} \in \mathscr{C}^{m \times n}$, $n > m$ have $p < m$ nonzero columns with the remaining columns composed entirely of zeros. In addition, assume that the p nonzero columns are linearly independent ($\operatorname{rank}(A_{l+1}) = p$). Then, for every nonzero column, a_{k_j}, $j = 1, \ldots, p$, we have $(I - A_{l+1}^{\dagger} A_{l+1})\, \hat{e}_j = 0$.*

Proof. Without loss of generality, assume that the first p columns of A_{l+1} are nonzero so that A_{l+1} can be partitioned as

$$A_{l+1} = [G\ 0],$$

with x correspondingly partitioned as $x=[x_\alpha^{\mathrm{T}}\ x_\beta^{\mathrm{T}}]^{\mathrm{T}}$. Then $(A_{l+1}A_{l+1}^{\mathrm{H}})^\dagger=(GG^{\mathrm{H}})^\dagger$ and

$$A_{l+1}^\dagger A_{l+1}=A_{l+1}^{\mathrm{H}}(A_{l+1}A_{l+1}^{\mathrm{H}})^\dagger A_{l+1}$$

$$=\begin{bmatrix} G^{\mathrm{H}}(GG^{\mathrm{H}})^\dagger G & 0\\ 0 & 0\end{bmatrix}=\begin{bmatrix} I & 0\\ 0 & 0\end{bmatrix},$$

which follows from the fact that $A_{l+1}^\dagger A_{l+1}$ is the orthogonal projection of $\mathscr{R}^n$ onto $\mathscr{R}(A_{l+1}^{\mathrm{H}})$ and, therefore, for any $x_\alpha\in\mathscr{R}^p$, $G^{\mathrm{H}}(GG^{\mathrm{H}})^\dagger Gx_\alpha=x_\alpha$ which gives the required result. □

With this condition established on the nonzero columns, we can now formulate a recursion for the computation of $A_{l+1}^\dagger$, which is proved in Appendix A.3.

Lemma 9. *Assume* $(I-A_l^\dagger A_l)\hat{e}_{l+1}=0$, *then*

$$A_{l+1}^\dagger=(I-K_{l+1}\hat{e}_{l+1}^{\mathrm{H}})A_l^\dagger$$

$$K_{l+1}=\frac{B_l\hat{e}_{l+1}}{\hat{e}_{l+1}^{\mathrm{H}}B_l\hat{e}_{l+1}},\quad B_l=(A_l^{\mathrm{H}}A_l)^\dagger. \tag{18}$$

In this update, the quantity $B_l=(A_l^{\mathrm{H}}A_l)^\dagger$ must be available. This was not necessary in the algorithm developed for the full row rank system. Therefore, to decrease the computation involved in the implementation, a recursion for B_l is established in the following lemma with the proof given in Appendix A.4.

Lemma 10. $B_{l+1}=(A_{l+1}^{\mathrm{H}}A_{l+1})^\dagger$ *can be computed as*

$$B_{l+1}=B_l-\frac{B_l\hat{e}_{l+1}\hat{e}_{l+1}^{\mathrm{H}}B_l^{\mathrm{H}}}{\hat{e}_{l+1}^{\mathrm{H}}B_l\hat{e}_{l+1}}$$

when $(I-A_l^\dagger A_l)\hat{e}_{l+1}=0.$ (19)

Finally, we develop a recursive expression for the representation error as given in (15). The proof of the following lemma is given in Appendix A.5.

Table 3
Rank deficient system: algorithm for backward elimination

```
Initialization: From the full row rank case, we have $x_v, \Delta_v, \delta_v, A_v^\dagger, D_v, K_v, B_v$ and we define $\tilde{D}_v=\emptyset$ to store the columns added by this algorithm. Recall that $B_l \triangleq (A_l^{\mathrm{H}}A_l)^\dagger$.
loop $l \geqslant v$ until Select = False
    • Select = $\{k_{l+1}\notin\tilde{D}_l\cup D_v, \varepsilon_{l+1}\leqslant\delta\}$ and CRITERIA
    • $\tilde{D}_{l+1}=\tilde{D}_l\cup\{k_{l+1}\}$
    • $K_{l+1}=\frac{B_l\hat{e}_{l+1}}{\hat{e}_{l+1}^{\mathrm{H}}B_l\hat{e}_{l+1}}$
    • $B_{l+1}=B_l-\frac{B_l\hat{e}_{l+1}\hat{e}_{l+1}^{\mathrm{H}}B_l^{\mathrm{H}}}{\hat{e}_{l+1}^{\mathrm{H}}B_l\hat{e}_{l+1}}$
    • $\delta x_{l+1}=-K_{l+1}\hat{e}_{l+1}^{\mathrm{H}}x_l$
    • $\Delta x_{l+1}=\Delta x_l+\delta x_{l+1}$
    • $x_{l+1}=x^\dagger+\Delta x_{l+1}$
    • $\varepsilon_{l+1}=\varepsilon_l+\delta\varepsilon_{l+1}$ where $\delta\varepsilon_{l+1}=\frac{(\hat{e}_{l+1}^{\mathrm{H}}x_l)^2}{\hat{e}_{l+1}^{\mathrm{H}}B_l\hat{e}_{l+1}}$
end loop
```

Lemma 11. *Assume* $(I-A_l^\dagger A_l)\hat{e}_{l+1}=0$, *then*

$$\varepsilon_{l+1}=\varepsilon_l+\frac{(\hat{e}_{l+1}^{\mathrm{H}}x_l)^2}{\hat{e}_{l+1}^{\mathrm{H}}B_l\hat{e}_{l+1}},\quad B_l=(A_l^{\mathrm{H}}A_l)^\dagger \tag{20}$$

which can be written as

$$\delta\varepsilon_{l+1}=\varepsilon_{l+1}-\varepsilon_l=\frac{x_l^2(k_{l+1})}{[(A_l^{\mathrm{H}}A_l)^\dagger]_{k_{l+1},k_{l+1}}}. \tag{21}$$

From (16) and (18), we have

$$\begin{aligned} x_{l+1}&=A_{l+1}^\dagger b=(I-K_{l+1}\hat{e}_{l+1}^{\mathrm{H}})A_l^\dagger b\\ &=(I-K_{l+1}\hat{e}_{l+1}^{\mathrm{H}})x_l\\ &=x_l-K_{l+1}\hat{e}_{l+1}^{\mathrm{H}}x_l \end{aligned}\tag{22}$$

or, as before,

$$x_{l+1}=x_l+\delta x_{l+1},\quad \delta x_{l+1}=-K_{l+1}\hat{e}_{l+1}^{\mathrm{H}}x_l, \tag{23}$$

where K_{l+1} is given by (18) and $(I-A_l^\dagger A_l)\hat{e}_{l+1}=0$. δx_{l+1} is of the form

$$\delta x_{l+1}=B_lM=(A_l^{\mathrm{H}}A_l)^\dagger M\in\mathscr{R}(A^{\mathrm{H}})\perp\mathscr{N}(A).$$

Since δx_{l+1} has no component in the nullspace it does not disrupt the zeroing which has been done earlier. Summarizing the development done in this section, we give the algorithm for the rank deficient case in Table 3.

4.3. Deletion criteria

Note that we were able to achieve $\Delta x_v \in \mathcal{N}(A)$, because the system A_v still had full row rank. The quantity $\Delta x_l - \Delta x_v = \sum_{j=v+1}^{l} \delta x_j \in \mathcal{R}(A^{\mathrm{H}}) \perp \mathcal{N}(A)$ is the part of the sparsity adjustment that is causing ε_l to be nonzero. There is again flexibility in the choice of selection criteria. In this case, our development was based on finding x which satisfies (15) but we can choose to use other criteria in determining which column to delete as detailed in Table 2.

It is also readily seen that it is possible to merge the algorithms given in Tables 1 and 3 based on the outcome of the test $(I - A_l^{\dagger} A_l)\hat{e}_{l+1} \neq 0$. If this holds for a given column, then the recursion detailed in Table 1 for the full row rank case is run. If it does not hold, the algorithm as given in Table 3 is run.

5. Computation

Consideration of the computation required to implement this algorithm is much the same as in [38]. We detail the computation required for the full row rank recursions since these are not considered there. The initial computation requires the calculation of $A^{\dagger}$ and $x^{\dagger}$. One method for doing this is first to form $(AA^{\mathrm{H}})^{-1}$ which requires $2nm^2 + 4m^2/3$ flops [17]. Then $A^{\dagger}$ is obtained with a further $2nm^2$ flops and $x^{\dagger}$ requires $2mn$ flops. From Table 1, it is seen that the calculation of K_{l+1} requires $A_l^{\dagger} A_l$. Initially $A^{\dagger} A$ is formed at the expense of $2mn^2$ flops and then the recursion is

$$\begin{aligned} A_{l+1}^{\dagger} A_{l+1} &= (I - K_{l+1}\hat{e}_{l+1}^{\mathrm{H}}) A_l^{\dagger} A_{l+1} \\ &= A_l^{\dagger} A_{l+1} - K_{l+1}\{\hat{e}_{l+1}^{\mathrm{H}} A_l^{\dagger} A_{l+1}\}. \end{aligned}$$

If there are i nonzero columns, then this update requires $\mathrm{O}(i^2)$ flops.

For the rank deficient case, the iteration complexity is approximately $2i^2$, as in [40]. Therefore, given that the iterations are from $i = n-1: -1 : r$, a total of $2(n^3 - r^3)/3$ flops are required. By way of comparison, the flop count for the ORMP, which is the most complex forward selection method [10] is $\mathrm{O}(m(n-r))$. Since, r is usually small, the algorithm complexities can be approximated as $\mathrm{O}(n^3)$ for backward elimination and $\mathrm{O}(nm)$ for ORMP.

6. Simulations

We now present simulation results on the algorithms that have been derived in the previous sections. In the first set of simulations, there is no noise present. Then noise is added to the solution vector and the performance of the algorithms is assessed under these noisy conditions.

6.1. Clean data

The first experiment performed was on clean data, i.e., where the solution vector b was uncorrupted by noise. The dictionary A is created as a random $m \times n$ matrix A whose entries are Gaussian random variables with mean 0 and variance 1. Each column is then normalized. A sparse solution, x_s, with a specified number of nonzero entries r is created; the indices of these r entries are randomly generated using a uniform distribution, and their amplitudes are Gaussian random variables. The vector b is computed as $b = Ax_s$ so that the solution is known and b is then normalized.

As a first test, we looked at different criteria for the choice of the column to be deleted. For the purposes of this experiment we chose the dimensions of A as 20×30 and the value of r as 4 (other dimensions and values for r were experimented with and yielded similar results). One hundred trials were performed with the matrix A, nonzero entries and amplitudes in x_s randomly created in each trial. A success is where we obtain the four vectors used in generating our solution as the four remaining vectors in our dictionary once all others have been zeroed out. Therefore, our measure of success is that of a component detection problem.

In the first experiment, the following four criteria were used: k_{l+1} randomly selected (which gives a baseline for the results obtained by the other methods), Shannon entropy as was given in (13), $k_{l+1} = \arg\min_{k_{l+1} \notin D_l} \|\delta x_{l+1}\|$, $k_{l+1} = \arg\min_{k_{l+1} \notin D_l} \|\Delta x_l + \delta x_{l+1}\|$. The results obtained are given in Fig. 1. From this experiment it was seen that,

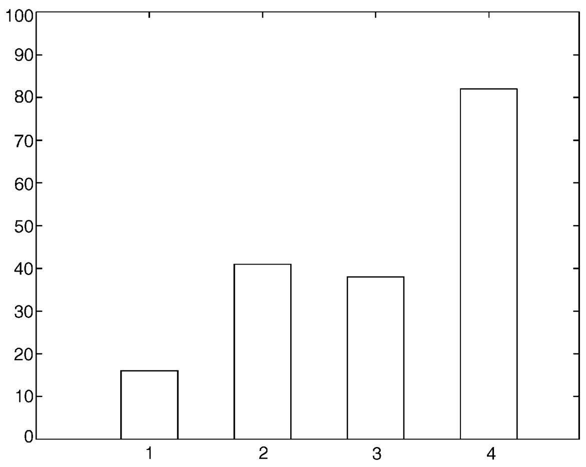

Fig. 1. Percentages of successful trials using 4 different criteria where the columns correspond to: 1. k_{l+1} randomly selected, 2. $k_{l+1} = \arg\min_{k_{l+1}\notin D_l} \|\delta x_{l+1}\|$, 3. $k_{l+1} = \arg\min_{k_{l+1}\notin D_l} \|\Delta x_l + \delta x_{l+1}\|$, 4. Shannon Entropy.

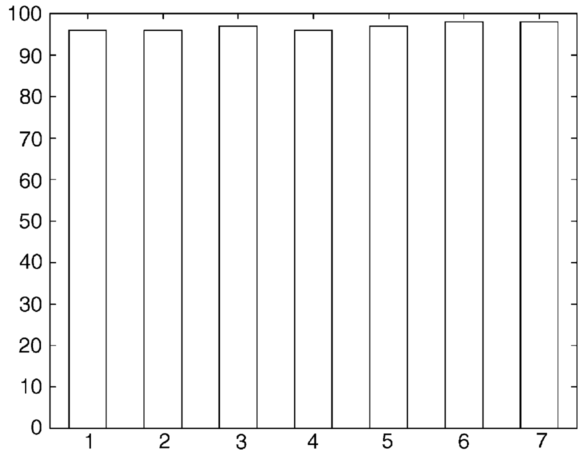

Fig. 2. Comparison of forward selection algorithms and backward elimination algorithm. 5 p-norms are used as different criteria for the backward elimination algorithm and the columns represent respectively MMP, ORMP, $p = 0.8, 0.85, 0.9, 0.95, 1.0$.

after deleting $(n - m) = 10$ columns, if we have not deleted any column from the r columns used to form the solution, then we obtain the exact solution. Therefore, it is not necessary to go to the rank deficient part of the algorithm as has been derived in Section 4. It is possible to implement this part of the algorithm but our success in finding the solution has already been determined. Purely from the algorithm as described in Section 3, we determine the solution or fail.

The Shannon entropy, a concentration measure which is defined in (13), performed much better than the other criteria. This led us to consider the class of p-norms with $p \leqslant 1$, an alternative set of concentration measures as given in (14), as a means of selecting the columns for deletion. In fact, we performed the experiments for all values of p from 0.25 to 1.2 in steps of 0.05. We only plot the five best values of p in Fig. 2. As a comparison with the first experiment, we were able to achieve 94–98% using these p-norms as opposed to the Shannon entropy where we had 82% success.

Now that we have established which criteria perform best in noise-free conditions, we look at how this backward elimination algorithm performs in comparison to two forward selection procedures termed MMP and ORMP in [1]. These results are also presented in Fig. 2 where we see that the backward elimination algorithm employing the p-norm performs slightly better than the forward pursuit algorithms. This, however, is at the expense of having a much higher operation count in the backward elimination method than in the forward pursuit algorithms as has been outlined in Section 5.

6.2. Noisy data

In the simulations presented in the previous section, it was not necessary to implement the elimination algorithm as derived for the rank deficient system of equations since the solution was obtained while the system was still of full row rank. However, in the presence of noise this is no longer the case. We still need to reduce the dictionary size further in order to find which columns were used in forming the solution vector b. The algorithm developed in Section 4 must be used to achieve this goal.

The dictionary size was again set as 20×30 and the solution was generated using $r = 4$ vectors, chosen at random, from the dictionary and linearly combined. Hundred trials are carried out, as before, but Gaussian noise of power -20 dB is added to the solution in each trial. The number of vectors sought is assumed to be known and, just as in the noise-free experiment, a success is where we obtain the four vectors used in generating our solution as the four remaining vectors in our dictionary once all others have been zeroed out. Given that there is noise in the solution, this may not necessarily be the best subset of four vectors for representation of the signal b.

In the experiment of the previous section, the p-norm was found to be the most effective search criterion. Therefore, as a starting point in these trials, this was also used as the search criterion for the rank deficient set of equations. Even though we arrived at the rank deficient part of the algorithm without having deleted one of the solution vectors in 84% of cases, using the p-norm in this part of the experiment proved disastrous. We ended up successfully finding the solution set in less than 7% of cases over a range of values of p.

Clearly, the criterion used for selection of columns to be deleted in the rank deficient case needed to be re-appraised. In this stage of the algorithm, rank$(A_{l+1}) < m$ so that error is introduced in approximating the solution vector b and this is given by (15). In [11], it is shown that selection of the column which minimizes the representation error at each stage (i.e., the column is chosen as in (15)) leads to a sparse solution. Therefore, the $p = 1$ norm was used while the system of equations considered was of full row rank and thereafter the columns were deleted so as to minimize the representation error. For comparison purposes, we exhaustively searched over all ${}^{30}C_4$ subsets to find the actual optimal subset yielding the smallest representation error. The forward selection algorithms MMP and ORMP [1] were also run on the trials. The results obtained for MMP, ORMP, backward elimination and exhaustive search are plotted in Fig. 3. In this figure, the histograms give the

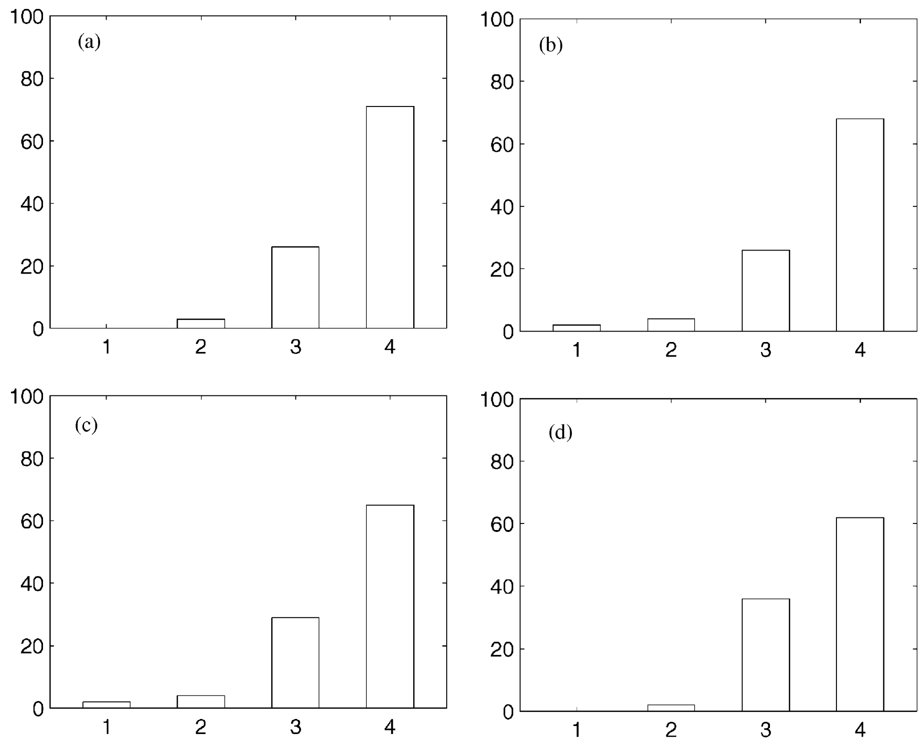

Fig. 3. Comparison of (a) Exhaustive Search, (b) ORMP, (c) MMP and (d) Backward Elimination in noise with $r = 4$. The number of vectors in the solution set which match those used in generating b was obtained for each trial and the percentages are plotted.

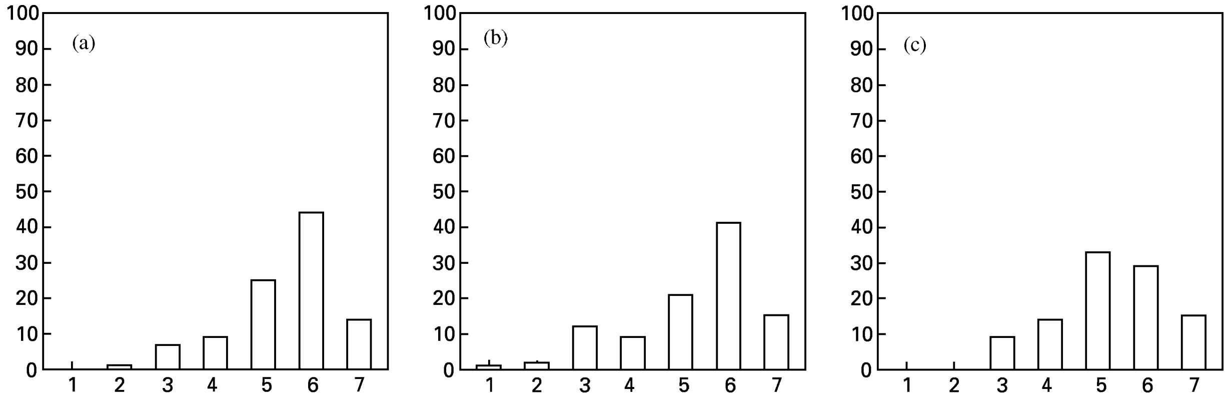

Fig. 4. Comparison of (a) ORMP, (b) MMP and (c) Backward Elimination in noise with $r = 7$. The number of vectors in the solution set which match those used in generating b was obtained for each trial and the percentages are plotted.

number of trials in which 0–4 of the vectors used in generating b are found in the solution subset. The value of r was then increased to 7 while the SNR was maintained at a level of 20 dB and the same experiment was repeated. The results from this experiment are plotted in Fig. 4 in a similar manner to Fig. 3 (the exhaustive search is no longer feasible because of the large number of subsets).

From Fig. 3, we see that with $r = 4$ the performance of all the suboptimal algorithms in detecting the dictionary components present in the noisy vector, b, is very close to that obtained by using an exhaustive search. Also, we note that the backward elimination method's performance is slightly inferior to that of the forward pursuit methods. For the case $r = 7$, the backward elimination performs as well as the other algorithms in obtaining all seven vectors. However, taking into account those trials where all seven vectors were not found, its performance is worse than that of the other methods.

As was stated previously, the part of the algorithm which handles the full row rank system will in many cases delete one of the columns required for the solution. Therefore, as a final indication of how well the rank deficient algorithm works, we discount these cases (16 trials) in the experiment as described for Fig. 3 and plot our results over the remaining trials. The results of this experiment are shown in Fig. 5 and indicate a success rate of 68% in these trials.

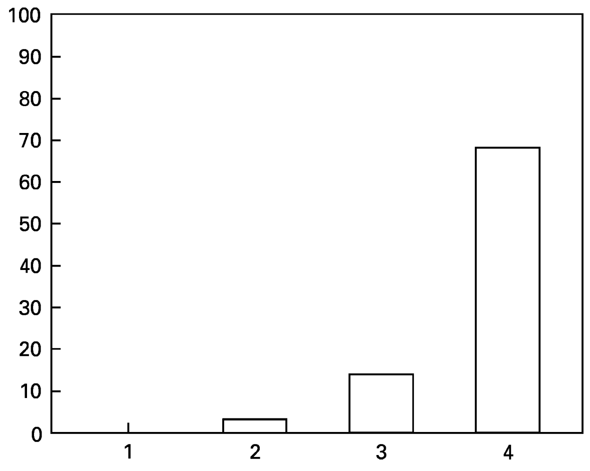

Fig. 5. The 16 trials in Fig. 3 which failed due to the full row rank part of the algorithm are removed. For the remaining trials, the percentages in which 0–4 of the vectors used in the generating set are matched in the solution set are plotted.

7. Conclusion

We started from an underdetermined system of equations and set out to find a solution which is sparse using a backward elimination algorithm. In developing an algorithm to achieve this, we found the new solution in each iteration. This allowed us to examine many different (including entropy-based) criteria for the deletion of vectors. Different criteria could potentially be combined and it is possible to switch between criteria at a given stage of the algorithm. In our simulations, from our exploration of different criteria in the implementation of the backward elimination algorithm, we can conclude that the set of p-norms close to 1 performs best in choosing columns to remove from the full row rank system. However, once the system becomes rank deficient, this criterion no longer leads to sparsity, and a minimum representation error criterion gives the best results.

The computational complexity of the algorithm was outlined and its complexity is, in general, far greater than that of a forward selection algorithm. We demonstrated that the algorithm can perform as well as a forward selection algorithm in a component detection problem on a test-case system. However, we envision that the backward elimination algorithm would, most likely, be used in practice to prune a sparse solution obtained more cheaply using an alternative algorithm.

Appendix A

A.1. Proof of Lemma 4

$$A_1^\dagger = A_1^{\mathrm{H}}(A_1A_1^{\mathrm{H}})^{-1},$$
$$A_1^\dagger = (I-\hat{e}_1\hat{e}_1^{\mathrm{H}})A^{\mathrm{H}}\{A(I-\hat{e}_1\hat{e}_1^{\mathrm{H}})A^{\mathrm{H}}\}^{-1}.$$

Now

$$\begin{aligned}\{A(I-\hat{e}_1\hat{e}_1^{\mathrm{H}})A^{\mathrm{H}}\}^{-1} &= \{AA^{\mathrm{H}} - A\hat{e}_1\hat{e}_1^{\mathrm{H}}A^{\mathrm{H}}\}^{-1}\\ &= \{\Pi - vv^{\mathrm{H}}\}^{-1} \quad \text{where } \Pi \triangleq AA^{\mathrm{H}},\ v \triangleq A\hat{e}_1\\ &= \Pi^{-1} + \frac{\Pi^{-1}vv^{\mathrm{H}}\Pi^{-1}}{1-v^{\mathrm{H}}\Pi^{-1}v}\\ &= (AA^{\mathrm{H}})^{-1} + \frac{(AA^{\mathrm{H}})^{-1}A\hat{e}_1\hat{e}_1^{\mathrm{H}}A^{\mathrm{H}}(AA^{\mathrm{H}})^{-1}}{1-\hat{e}_1^{\mathrm{H}}A^{\mathrm{H}}(AA^{\mathrm{H}})^{-1}A\hat{e}_1}.\end{aligned}$$

Therefore

$$\begin{aligned}A_1^\dagger &= (I-\hat{e}_1\hat{e}_1^{\mathrm{H}})\left\{I + \frac{A^\dagger A\hat{e}_1\hat{e}_1^{\mathrm{H}}}{1-\hat{e}_1^{\mathrm{H}}A^\dagger A\hat{e}_1}\right\}A^\dagger\\ &= \frac{(1-\hat{e}_1^{\mathrm{H}}A^\dagger A\hat{e}_1)(I-\hat{e}_1\hat{e}_1^{\mathrm{H}}) + A^\dagger A\hat{e}_1\hat{e}_1^{\mathrm{H}} - (\hat{e}_1^{\mathrm{H}}A^\dagger A\hat{e}_1)\hat{e}_1\hat{e}_1^{\mathrm{H}}}{1-\hat{e}_1^{\mathrm{H}}A^\dagger A\hat{e}_1}\\ &= \left\{I - \frac{(I-A^\dagger A)\hat{e}_1\hat{e}_1^{\mathrm{H}}}{1-\hat{e}_1^{\mathrm{H}}A^\dagger A\hat{e}_1}\right\}A^\dagger\\ &= (I-K_1\hat{e}_1^{\mathrm{H}})A^\dagger.\end{aligned}$$

A.2. Proof of Lemma 5

We note that $K_1 \propto (I-A^\dagger A)$ and therefore $AK_1 = 0$ [5]. Thus $Ax_1 = AA_1^\dagger b = A(I-K_1\hat{e}_1^{\mathrm{H}})A^\dagger b = AA^\dagger b = b$. Also,

$$\begin{aligned}P_1x_1 &= P_1A_1^\dagger b = P_1A_1^{\mathrm{H}}(A_1A_1^{\mathrm{H}})^{-1}b\\ &= P_1P_1A^{\mathrm{H}}(A_1A_1^{\mathrm{H}})^{-1}b = P_1A^{\mathrm{H}}(A_1A_1^{\mathrm{H}})^{-1}b\\ &= A_1^{\mathrm{H}}(A_1A_1^{\mathrm{H}})^{-1}b = A_1^\dagger b = x_1.\end{aligned}$$

A.3. Proof of Lemma 9

To compute (17), we need to make use of the following lemma from [5] which generalizes the Sherman–Morrison–Woodbury matrix inversion lemma.

Lemma (Campbell and Meyer [5]). *Consider the matrix $\Pi + cd^{\mathrm{H}}$ where $c \in \mathscr{R}(\Pi)$, $d \in \mathscr{R}(\Pi^{\mathrm{H}})$, and $1 + d^{\mathrm{H}}\Pi^\dagger c = 0$. Then*

$$(\Pi + cd^{\mathrm{H}})^\dagger = \Pi^\dagger - kk^\dagger\Pi^\dagger - \Pi^\dagger h^\dagger h + (k^\dagger\Pi^\dagger h^\dagger)kh, \tag{A.1}$$

where $k = \Pi^\dagger c$ and $h = d^{\mathrm{H}}\Pi^\dagger$.

Using the above lemma, we now proceed to the proof of Lemma 9 taking $\Pi = A_lA_l^{\mathrm{H}}$,

$$\begin{aligned}c &= A_l\hat{e}_{l+1} \in \mathscr{R}(A_l) = \mathscr{R}(A_lA_l^{\mathrm{H}}) = \mathscr{R}(\Pi),\\ d &= -c = -A_l\hat{e}_{l+1} \in \mathscr{R}(A_l)\\ &= \mathscr{R}(A_lA_l^{\mathrm{H}}) = \mathscr{R}(\Pi) = \mathscr{R}(\Pi^{\mathrm{H}}),\end{aligned}$$

$$\begin{aligned}1 + d^{\mathrm{H}}\Pi^\dagger c = 1 - c^{\mathrm{H}}\Pi^\dagger c &= 1 - \hat{e}_{l+1}^{\mathrm{H}}A_l^{\mathrm{H}}(A_lA_l^{\mathrm{H}})A_l\hat{e}_{l+1}\\ &= 1 - \hat{e}_{l+1}^{\mathrm{H}}A_l^\dagger A_l\hat{e}_{l+1} = 0\\ &\quad \text{since } (I - A_l^\dagger A_l)\hat{e}_{l+1} = 0.\end{aligned}$$

Then

$$k = \Pi^\dagger c = \Pi^\dagger A_l\hat{e}_{l+1} = A_l^{\dagger^{\mathrm{H}}}\hat{e}_{l+1}$$

and

$$h = d^{\mathrm{H}}\Pi^\dagger = -c^{\mathrm{H}}\Pi^\dagger = -k^{\mathrm{H}} = \hat{e}_{l+1}^{\mathrm{H}}A_l^\dagger.$$

Noting that

$$k^\dagger = \frac{k^{\mathrm{H}}}{\|k\|^2},$$

we have

$$\begin{aligned}(A_lP_{l+1}A_l^{\mathrm{H}})^\dagger &= (A_lA_l^{\mathrm{H}} - A_l\hat{e}_{l+1}\hat{e}_{l+1}^{\mathrm{H}}A_l^{\mathrm{H}})^\dagger\\ &= (\Pi + cd^{\mathrm{H}})^\dagger = \Pi^\dagger - kk^\dagger\Pi^\dagger\\ &\quad - \Pi^\dagger kk^\dagger + (k^\dagger\Pi^\dagger k^{\dagger^{\mathrm{H}}})kk^{\mathrm{H}}\\ &= \Pi^\dagger - \frac{kk^{\mathrm{H}}}{\|k\|^2}\Pi^\dagger - \frac{\Pi^\dagger kk^{\mathrm{H}}}{\|k\|^2}\\ &\quad + \frac{kk^{\mathrm{H}}(k^{\mathrm{H}}\Pi^\dagger k)}{\|k\|^4} \equiv G.\end{aligned}$$

Hence $A_{l+1}^\dagger = A_{l+1}^{\mathrm{H}}(A_{l+1}A_{l+1}^{\mathrm{H}})^\dagger = P_{l+1}A_l^{\mathrm{H}}G = (I - \hat{e}_{l+1}\hat{e}_{l+1}^{\mathrm{H}})A_l^{\mathrm{H}}G$. The second and fourth terms in G contain kk^{H} and therefore we consider $(I - \hat{e}_{l+1}\hat{e}_{l+1}^{\mathrm{H}})A_l^{\mathrm{H}}kk^{\mathrm{H}}$. As a first step, note that $k = A_l^{\dagger^{\mathrm{H}}}\hat{e}_{l+1}$, $kk^{\mathrm{H}} = A_l^{\dagger^{\mathrm{H}}}\hat{e}_{l+1}\hat{e}_{l+1}^{\mathrm{H}}A_l^{\mathrm{H}}$, and that $(I - A_l^\dagger A_l)\hat{e}_{l+1} = 0 \Rightarrow \hat{e}_{l+1} \in \mathscr{R}(A_l^{\mathrm{H}}) \Rightarrow \hat{e}_{l+1} = A_l^{\mathrm{H}}\lambda \Rightarrow (I - A_l^{\mathrm{H}}A_l^{\dagger^{\mathrm{H}}})\hat{e}_{l+1} = 0$ which shows that $A_l^{\mathrm{H}}A_l^{\dagger^{\mathrm{H}}}\hat{e}_{l+1} = \hat{e}_{l+1}$, so that

$$\begin{aligned}&\hat{e}_{l+1}^{\mathrm{H}}A_l^{\mathrm{H}}k = \hat{e}_{l+1}^{\mathrm{H}}A_l^{\mathrm{H}}A_l^{\dagger^{\mathrm{H}}}\hat{e}_{l+1} = \hat{e}_{l+1}^{\mathrm{H}}\hat{e}_{l+1} = 1\\ &\quad\Rightarrow (I - \hat{e}_{l+1}\hat{e}_{l+1}^{\mathrm{H}})A_l^{\mathrm{H}}kk^{\mathrm{H}} = A_l^{\mathrm{H}}kk^{\mathrm{H}}\\ &\qquad - \hat{e}_{l+1}(\hat{e}_{l+1}^{\mathrm{H}}A_l^{\mathrm{H}}k)k^{\mathrm{H}}\\ &\quad = A_l^{\mathrm{H}}A_l^{\dagger^{\mathrm{H}}}\hat{e}_{l+1}\hat{e}_{l+1}^{\mathrm{H}}A_l^\dagger - \hat{e}_{l+1}k^{\mathrm{H}}\\ &\quad = A_l^{\mathrm{H}}A_l^{\dagger^{\mathrm{H}}}\hat{e}_{l+1}\hat{e}_{l+1}^{\mathrm{H}}A_l^\dagger\\ &\qquad - \hat{e}_{l+1}\hat{e}_{l+1}^{\mathrm{H}}A_l^\dagger\\ &\quad = (A_l^{\mathrm{H}}A_l^{\dagger^{\mathrm{H}}} - I)\hat{e}_{l+1}\hat{e}_{l+1}^{\mathrm{H}}A_l^{\mathrm{H}} = 0.\end{aligned}$$

Therefore,

$$\begin{aligned}A_{l+1}^\dagger &= (I - \hat{e}_{l+1}\hat{e}_{l+1}^{\mathrm{H}})A_l^{\mathrm{H}}G\\ &= (I - \hat{e}_{l+1}\hat{e}_{l+1}^{\mathrm{H}})A_l^{\mathrm{H}}\left\{\Pi^\dagger - \Pi^\dagger\frac{kk^{\mathrm{H}}}{\|k\|^2}\right\}\\ &= (I - \hat{e}_{l+1}\hat{e}_{l+1}^{\mathrm{H}})\{A_l^{\mathrm{H}}\Pi^\dagger\\ &\quad - \left.\frac{A_l^{\mathrm{H}}\Pi^\dagger A_l^{\dagger^{\mathrm{H}}}\hat{e}_{l+1}\hat{e}_{l+1}^{\mathrm{H}}A^\dagger}{\hat{e}_{l+1}^{\mathrm{H}}(A_l^{\mathrm{H}}A_l)^\dagger\hat{e}_{l+1}}\right\}\\ &= (I - \hat{e}_{l+1}\hat{e}_{l+1}^{\mathrm{H}})\left\{I - \frac{A_l^\dagger A_l^{\dagger^{\mathrm{H}}}\hat{e}_{l+1}\hat{e}_{l+1}^{\mathrm{H}}}{\hat{e}_{l+1}^{\mathrm{H}}(A_l^{\mathrm{H}}A_l)^\dagger\hat{e}_{l+1}}\right\}A_l^\dagger;\\ &\qquad A_l^\dagger = A_l^{\mathrm{H}}\Pi^\dagger\\ &= (I - \hat{e}_{l+1}\hat{e}_{l+1}^{\mathrm{H}})\left\{I - \frac{(A_l^{\mathrm{H}}A_l)^\dagger\hat{e}_{l+1}\hat{e}_{l+1}^{\mathrm{H}}}{\hat{e}_{l+1}^{\mathrm{H}}(A_l^{\mathrm{H}}A_l)^\dagger\hat{e}_{l+1}}\right\}A_l^\dagger\\ &= \left\{I - \hat{e}_{l+1}\hat{e}_{l+1}^{\mathrm{H}} - \frac{(A_l^{\mathrm{H}}A_l)^\dagger\hat{e}_{l+1}\hat{e}_{l+1}^{\mathrm{H}}}{\hat{e}_{l+1}^{\mathrm{H}}(A_l^{\mathrm{H}}A_l)^\dagger\hat{e}_{l+1}}\right.\\ &\quad \left. + \frac{\hat{e}_{l+1}(\hat{e}_{l+1}^{\mathrm{H}}(A_l^{\mathrm{H}}A_l)^\dagger\hat{e}_{l+1})\hat{e}_{l+1}^{\mathrm{H}}}{\hat{e}_{l+1}^{\mathrm{H}}(A_l^{\mathrm{H}}A_l)^\dagger\hat{e}_{l+1}}\right\}A_l^\dagger\\ &= \left\{I - \frac{(A_l^{\mathrm{H}}A_l)^\dagger\hat{e}_{l+1}\hat{e}_{l+1}^{\mathrm{H}}}{\hat{e}_{l+1}^{\mathrm{H}}(A_l^{\mathrm{H}}A_l)^\dagger\hat{e}_{l+1}}\right\}A_l^\dagger\\ &= (I - K_{l+1}\hat{e}_{l+1}^{\mathrm{H}})A_l^\dagger;\end{aligned}$$

where K_{l+1}

$$= \frac{B_l\hat{e}_{l+1}}{\hat{e}_{l+1}B_l\hat{e}_{l+1}},\ B_l = (A_l^{\mathrm{H}}A_l)^\dagger.$$

A.4. Proof of Lemma 10

$$\begin{aligned}B_{l+1} &= (A_{l+1}^{\mathrm{H}}A_{l+1})^\dagger = A_{l+1}^\dagger A_{l+1}^{\dagger^{\mathrm{H}}}\\ &= (I - K_{l+1}\hat{e}_{l+1}^{\mathrm{H}})A_l^\dagger A_l^{\dagger^{\mathrm{H}}}(I - \hat{e}_{l+1}K_{l+1}^{\mathrm{H}})\\ &= \left\{I - \frac{(A_l^{\mathrm{H}}A_l)^\dagger\hat{e}_{l+1}\hat{e}_{l+1}^{\mathrm{H}}}{\hat{e}_{l+1}^{\mathrm{H}}B_l\hat{e}_{l+1}}\right\}(A_l^{\mathrm{H}}A_l)^\dagger\\ &\quad\times\left\{I - \frac{\hat{e}_{l+1}\hat{e}_{l+1}^{\mathrm{H}}(A_l^{\mathrm{H}}A_l)^\dagger}{\hat{e}_{l+1}^{\mathrm{H}}B_l\hat{e}_{l+1}}\right\}\\ &= B_l - \frac{B_l\hat{e}_{l+1}\hat{e}_{l+1}^{\mathrm{H}}B_l}{\hat{e}_{l+1}^{\mathrm{H}}B_l\hat{e}_{l+1}} - \frac{B_l\hat{e}_{l+1}\hat{e}_{l+1}^{\mathrm{H}}B_l}{\hat{e}_{l+1}^{\mathrm{H}}B_l\hat{e}_{l+1}}\\ &\quad + \frac{B_l\hat{e}_{l+1}(\hat{e}_{l+1}^{\mathrm{H}}B_l\hat{e}_{l+1})\hat{e}_{l+1}^{\mathrm{H}}B_l}{\hat{e}_{l+1}^{\mathrm{H}}B_l\hat{e}_{l+1}}\\ &= B_l - \frac{B_l\hat{e}_{l+1}\hat{e}_{l+1}^{\mathrm{H}}B_l}{\hat{e}_{l+1}^{\mathrm{H}}B_l\hat{e}_{l+1}}.\end{aligned}$$

A.5. Proof of Lemma 11

$$\begin{aligned}\varepsilon_{l+1} &= \|A_{l+1}x_{l+1} - b\|^2\\ \Xi_{l+1} &= A_{l+1}x_{l+1} - b = A_lP_{l+1}A_{l+1}^\dagger b - b\\ &= \{A_lP_{l+1}A_{l+1}^\dagger - I\}b\\ &= \{A_lP_{l+1}(I - K_{l+1}\hat{e}_{l+1}^{\mathrm{H}})A_l^\dagger - I\}b\end{aligned}$$

Note,

$$\begin{aligned}&P_{l+1}(I - K_{l+1}\hat{e}_{l+1}^{\mathrm{H}})\\ &\quad = (I - \hat{e}_{l+1}\hat{e}_{l+1}^{\mathrm{H}})\left(I - \frac{B_l\hat{e}_{l+1}\hat{e}_{l+1}^{\mathrm{H}}}{\hat{e}_{l+1}^{\mathrm{H}}B_l\hat{e}_{l+1}}\right)\end{aligned}$$

$$= I - \hat{e}_{l+1}\hat{e}_{l+1}^{\mathrm{H}} - \frac{B_l\hat{e}_{l+1}\hat{e}_{l+1}^{\mathrm{H}}}{\hat{e}_{l+1}^{\mathrm{H}}B_l\hat{e}_{l+1}} + \frac{e_{l+1}(\hat{e}_{l+1}^{\mathrm{H}}B_l\hat{e}_{l+1})e_{l+1}^{\mathrm{H}}}{\hat{e}_{l+1}^{\mathrm{H}}B_l\hat{e}_{l+1}}$$

$$= I - \frac{B_l\hat{e}_{l+1}\hat{e}_{l+1}^{\mathrm{H}}}{\hat{e}_{l+1}^{\mathrm{H}}B_l\hat{e}_{l+1}} = (I - K_{l+1}\hat{e}_{l+1}^{\mathrm{H}}).$$

So,

$$\Xi_{l+1} = \{A_l(I - K_{l+1}\hat{e}_{l+1}^{\mathrm{H}})A_l^{\dagger} - I\}b$$
$$= (A_lA_l^{\dagger} - I)b - A_lK_{l+1}\hat{e}_{l+1}^{\mathrm{H}}A_l^{\dagger}b.$$

The two terms in this expression are orthogonal to one another since $A_l^{\mathrm{H}}(I - A_lA_l^{\dagger}) = A_l^{\mathrm{H}}P_{\mathcal{N}(A^{\mathrm{H}})} = 0$. Therefore,

$$\varepsilon_{l+1} = \|A_lx_l^{\dagger} - b\|^2 + \|A_lK_{l+1}\hat{e}_{l+1}^{\mathrm{H}}A_l^{\dagger}b\|^2$$
$$= \varepsilon_l + \delta\varepsilon_{l+1},$$

$$\delta\varepsilon_{l+1} = (b^{\mathrm{H}}A_l^{\dagger^{\mathrm{H}}}\hat{e}_{l+1})(K_{l+1}^{\mathrm{H}}A_l^{\mathrm{H}}A_lK_{l+1})(\hat{e}_{l+1}^{\mathrm{H}}A_l^{\dagger}b),$$

$$K_{l+1}^{\mathrm{H}}A_l^{\mathrm{H}}A_lK_{l+1} = \frac{\hat{e}_{l+1}^{\mathrm{H}}B_l(A_l^{\mathrm{H}}A_l)B_l\hat{e}_{l+1}}{(\hat{e}_{l+1}^{\mathrm{H}}B_l\hat{e}_{l+1})^2}$$
$$= \frac{\hat{e}_{l+1}^{\mathrm{H}}B_l\hat{e}_{l+1}}{(\hat{e}_{l+1}^{\mathrm{H}}B_l\hat{e}_{l+1})^2} = \frac{1}{\hat{e}_{l+1}^{\mathrm{H}}B_l\hat{e}_{l+1}}.$$

Therefore,

$$\delta\varepsilon_{l+1} = \frac{(\hat{e}_{l+1}^{\mathrm{H}}A_l^{\dagger}b)^2}{\hat{e}_{l+1}^{\mathrm{H}}B_l\hat{e}_{l+1}}.$$

References

[1] J. Adler, B.D. Rao, K. Kreutz-Delgado, Comparison of basis selection methods, 30th Asilomar Conference on Signals, Systems and Computers, Pacific Grove, CA, USA, (1), November 1996, pp. 252–257.

[2] B.S. Atal, J.R. Remde, 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, pp. 614–617.

[3] F. Bergeaud, S. Mallat, Matching pursuit of images, Proceedings of International Conference on Image Processing, Los Alamitos, CA, USA, October 1995, pp. 53–66.

[4] S.D. Cabrera, T.W. Parks, Extrapolation and spectral estimation with iterative weighted norm modification, IEEE Trans. Signal Process. 39 (4) (April 1991) 842–851.

[5] S.L. Campbell, C.D. Meyer, Generalized Inverses of Linear Transformations, Dover, New York, 1991.

[6] S. Chen, D. Donoho, Basis pursuit, 28th Asilomar Conference on Signals, Systems and Computers, Pacific Grove, CA, USA, (1), November 1994, pp. 41–44.

[7] S.S. Chen, D.L. Doonoho, M.A. Saunders, Atomic decomposition by basis pursuit, SIAM J. Sci. Comput. 20 (1) (1998) 33–61.

[8] S. Chen, J. Wigger, Fast orthogonal least squares algorithm for efficient subset model selection, IEEE Trans. Acoust. Speech Signal Process. 43 (7) (July 1995) 1713–1715.

[9] E.S. Cheng, S. Chen, B. Mulgrew, Efficient computational schemes for the orthogonal least squares learning algorithm, IEEE Trans. Signal Process. 43 (1) (January 1995) 373–376.

[10] S.F. Cotter, M.N. Murthi, B.D. Rao, Fast basis selection methods, 31st Asilomar Conference on Signals, Systems and Computers, Pacific Grove, CA, USA, (2), November 1997, pp. 1474–1478.

[11] C. Couvreur, Y. Bresler, On the optimality of the backward greedy algorithm for the subset selection problem, SIAM J. Matrix Anal. Appl. 21 (3) (1999) 797–808.

[12] G. Davis, S. Mallat, Z. Zhang, Adaptive time-frequency decompositions, Opt. Eng. 33 (7) (July 1994) 2183–2191.

[13] P.J. Dhrymes, Mathematics for Econometrics, 2nd Edition, Springer, New York, 1984.

[14] P. Duhamel, J.C. Rault, Automatic test generation techniques for analog circuits and systems: a review, IEEE Trans. Circuits Systems 26 (7) (July 1979) 411–440.

[15] D.J. Field, What is the goal of sensory coding, Neural Comput. 6 (4) (April 1994) 559–601.

[16] J.H. Friedman, W. Stuetzle, Projection pursuit regression, J. Amer. Statist. Assoc. 76 (1981) 817–823.

[17] G.H. Golub, C.F. Van Loan, Matrix Computations, John Hopkins University Press, Baltimore, MD, 1989.

[18] M. Goodwin, M. Vetterli, Atomic decompositions of audio signals, Proceedings of Workshop on Applications of Signal Processing to Audio and Acoustics, October 1997, New York, USA, pp. 4–7.

[19] I.F. Gorodnitsky, J.S. George, B.D. Rao, Neuromagnetic source imaging with FOCUSS: a recursive weighted minimum norm algorithm, J. Electroencephalography Clin. Neurophysio. 95 (4) (April 1995) 231–251.

[20] I.F. Gorodnitsky, B.D. Rao, A recursive weighted minimum-norm algorithm: analysis and applications, Proceedings of the International Conference on Acoustics, Speech and Signal Processing, ICASSP '93, Minneapolis, MN, USA, (3), April 1993, pp. 456–459.

[21] I.F. Gorodnitsky, B.D. Rao, Sparse signal reconstructions from limited data using FOCUSS: a re-weighted minimum norm algorithm, IEEE Trans. Signal Process. 45 (3) (March 1997) 600–616.

[22] R. Gribonval, E. Bacry, S. Mallat, P.R. Depalle, X. Rodet, 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, pp. 125–128.

[23] G. Harikumar, C. Couvreur, Y. Bresler, Fast optimal and suboptimal algorithms for sparse solutions to linear inverse problems, Proceedings of the International Conference on Acoustics, Speech and Signal Processing, ICASSP'98, Seattle, WA, USA, (III), May 1998, pp. 1877–1880.

[24] R.R. Hocking, R.N. Leslie, Selection of the best subset in regression analysis, Technometrics 9 (November 1967) 531–540.

[25] K. Kreutz-Delgado, B.D. Rao, A general approach to sparse basis selection: majorization, concavity, and affine scaling, Center For Information Engineering Report No. UCSD-CIE-97-7-1, ECE Department, UC San Diego. URL: http://raman.ucsd.edu, July, 1997.

[26] K. Kreutz-Delgado, B.D. Rao, Sparse basis selection, ICA, and majorization: towards a unified perspective, Proceedings of the International Conference on Acoustics, Speech and Signal Processing, ICASSP'99, Phoenix, AZ, USA, Vol. 2, pp. 1081–1084.

[27] K. Kreutz-Delgado, et al., Convex/Schur–Convex (CSC) log-priors and sparse coding, Proceedings of the 6th Joint Symposium on Neural Computation, Pasadena, CA, USA, May 1999.

[28] L.R. LaMotte, R.R. Hocking, Computational efficiency in the selection of regression variables, Technometrics 12 (February 1970) 83–93.

[29] H. Lee, D.P. Sullivan, T.H. Huang, Improvement of discrete band-limited signal extrapolation by iterative subspace modification, Proceedings of the International Conference on Acoustics, Speech and Signal Processing, ICASSP'87, Dallas, TX, USA, (3), April 1987, pp. 1569–1572.

[30] S.G. Mallat, Z. Zhang, Matching pursuits with time-frequency dictionaries, IEEE Trans. Signal Process. 41 (12) (December 1993) 3397–3415.

[31] A.J. Miller, Subset Selection in Regression, Chapman & Hall, London, 1990.

[32] B.K. Natarajan, Sparse approximate solutions to linear systems, SIAM J. Comput. 24 (2) (February 1995) 227–234.

[33] R. Neff, A. Zakhor, Very low bit-rate video coding based on matching pursuits, IEEE Trans. Circuits Systems Video Technol. 7 (1) (January 1997) 158–171.

[34] B.A. Olshausen, D.J. Field, Emergence of simple-cell receptive field properties by learning a sparse code for natural images, Nature 381 (6583) (1996) 607–609.

[35] Y.C. Pati, R. Rezaiifar, P.S. Krishnaprasad, Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition, 27th Asilomar Conference on Signal, Systems, and Computers, Pacific Grove, CA, USA, (1), November 1993, pp. 40–44.

[36] H.R. Rabiee, R.L. Kashyap, S.R. Safavian, Adaptive multiresolution image coding with matching and basis pursuits, Proceedings of International Conference on Image Processing, Lausanne, Switzerland, (1), September 1996, pp. 273–276.

[37] J.B. Ramsey, Z. Zhang, The application of waveform dictionaries to stock market index data, in: J. Kadtke, A. Kravtsov (Eds.), Predictablility of Complex Dynamical Systems, Springer, Berlin, 1996.

[38] B.D. Rao, Signal processing with the sparseness constraint, Proceedings of the International Conference on Acoustics, Speech and Signal Processing, ICASSP'98, Seattle, WA, USA, (III), May 1998, pp. 1861–1864.

[39] B.D. Rao, K. Kreutz-Delgado, An affine scaling methodology for best basis selection, IEEE Trans. Signal Process. 47 (1) (January 1999) 187–200.

[40] S. Reeves, An efficient implementation of the backward greedy algorithm for sparse signal reconstruction, IEEE Signal Process. Lett. 6 (10) (October 1999) 266–268.

[41] S. Singhal, B.S. Atal, Amplitude optimization and pitch prediction in multipulse coders, IEEE Trans. Acoust. Speech Signal Process. 37 (3) (March 1989) 317–327.