

SPARSE RECONSTRUCTION FOR COMPRESSED SENSING USING STAGewise POLYTOPE FACES PURSUIT

Mark D. Plumbley*

Queen Mary University of London,
School of Elec. Eng. and Comp. Sci.,
Mile End Road, London E1 4NS, UK
mark.plumbley@elec.qmul.ac.uk

Marco Bevilacqua†

Dip. di Elettronica e Telecomunicazioni
Università degli Studi di Firenze
Via Santa Marta, 3, 50139 Firenze, Italy
marcobev@inwind.it

ABSTRACT

Compressed sensing, also known as compressive sampling, is an approach to the measurement of signals which have a sparse representation, that can reduce the number of measurements that are needed to reconstruct the signal. The signal reconstruction part requires efficient methods to perform sparse reconstruction, such as those based on linear programming. In this paper we present a method for sparse reconstruction which is an extension of our earlier Polytope Faces Pursuit algorithm, based on the polytope geometry of the dual linear program. The new algorithm adds several basis vectors at each stage, in a similar way to the recent Stagewise Orthogonal Matching Pursuit (StOMP) algorithm. We demonstrate the application of the algorithm to some standard compressed sensing problems.

Index Terms— Sparse reconstruction, Compressed Sensing, Basis Pursuit (BP), greedy algorithms, polytopes.

1. INTRODUCTION

Compressed sensing [1], also known as compressive sampling or compressive sensing [2], is an approach to signal measurement whereby a signal \mathbf{x} can be reconstructed or estimated using a small number of measurements, provided that \mathbf{x} is known to have a *sparse representation*. Specifically, we suppose that we can represent a signal of interest \mathbf{x} as

$$\mathbf{x} = \Psi \mathbf{s} \quad (1)$$

in some basis Ψ , where $\mathbf{s} = [s_1, \dots, s_N]$ has only a small number $K = \|\mathbf{s}\|_0$ non-zero entries, with $K \ll N$. Under this condition we say that \mathbf{s} is *K-sparse*. We then take $M < N$ measurements of \mathbf{x} , $y_i = \phi_i^T \mathbf{x}$, $1 \leq i \leq M$ giving the resulting system

$$\mathbf{y} = \Phi \mathbf{x} = \Phi \Psi \mathbf{s} \quad (2)$$

where ϕ_i^T is the i -th row of Φ .

In order to reconstruct \mathbf{x} given the measurement \mathbf{y} , the measurement system Φ and the basis matrix Ψ , we first reconstruct the sparse vector $\hat{\mathbf{s}}$ satisfying

$$\mathbf{y} = \mathbf{A} \mathbf{s} \quad (3)$$

where $\mathbf{A} = \Phi \Psi$. We then reconstruct the original signal using $\hat{\mathbf{x}} = \Psi \hat{\mathbf{s}}$. If the measurement system Φ and the representation basis Ψ are sufficiently *incoherent* [2], and a sufficiently large number of measurements $M < N$ are taken, then this process will have a high probability of recovering the original signal \mathbf{x} . For further details, see e.g. [2]

A critical part of this process is the recovery of the sparse vector \mathbf{s} in (3). Finding the truly sparsest vector, given by the minimum ℓ_0 solution

$$\min_{\mathbf{s}} \|\mathbf{s}\|_0 \quad \text{such that} \quad \mathbf{y} = \mathbf{A} \mathbf{s} \quad (4)$$

is NP-hard, so instead we use the minimum ℓ_1 solution

$$\min_{\mathbf{s}} \|\mathbf{s}\|_1 \quad \text{such that} \quad \mathbf{y} = \mathbf{A} \mathbf{s} \quad (5)$$

which can be solved using linear programming (LP). Equation (5) is sometimes known as the *Basis Pursuit* solution [3].

In this paper, we present a method designed to find solutions to (5), called *Stagewise Polytope Faces Pursuit*. This is a greedy method, based on the polytope geometry of the dual linear program to (5). It proceeds by adding several basis vectors at each stage, inspired by the Stagewise Orthogonal Matching Pursuit (StOMP) [4] and Stagewise Weak Orthogonal Matching Pursuit (SWOMP) [5] algorithms. In the following sections we will review existing greedy algorithms, introduce the polytope geometry ideas, review the earlier (stepwise) Polytope Faces Pursuit algorithm [6], develop the stage-wise version of the algorithm, discuss implementation and operation issues, and present results of its application to compressed sensing problems.

*This research is supported by EPSRC Leadership Fellowship EP/G007144/1 and EU FET-Open project FP7-ICT-225913 ‘‘SMALL’’.

†This work was performed while the second author was visiting Queen Mary University of London.

2. SPARSE RECONSTRUCTION

2.1. Greedy algorithms

While the Basis Pursuit problem (5) can be solved by linear programming methods such as the simplex method [3], there are faster approximate algorithms such as Matching Pursuits (MP) [7] and Orthogonal Matching Pursuits (OMP) [8] (Algorithm 1).

Algorithm 1 Orthogonal Matching Pursuits [8]

- 1: Input: \mathbf{A} , \mathbf{y}
 - 2: Set stopping conditions k_{\max} and θ_{\min}
 - 3: Initialize: $k \leftarrow 0$, $\mathcal{I}^k \leftarrow \emptyset$, $\mathbf{A}^k \leftarrow \emptyset$, $\mathbf{r}^k \leftarrow \mathbf{y}$
 - 4: **while** $k < k_{\max}$ and $\max_i \mathbf{a}_i^T \mathbf{r}^k > \theta_{\min}$ **do**
 - 5: $k \leftarrow k + 1$
 - 6: Find next vector: $i^k \leftarrow \arg \max_{i \notin \mathcal{I}^{k-1}} \{\mathbf{a}_i^T \mathbf{r}^{k-1}\}$
 - 7: Add to solution set:
 $\mathbf{A}^k \leftarrow [\mathbf{A}^{k-1}, \mathbf{a}_{i^k}]$, $\mathcal{I}^k \leftarrow \mathcal{I}^{k-1} \cup \{i^k\}$,
 $\mathbf{s}^k \leftarrow (\mathbf{A}^k)^\dagger \mathbf{y}$, $\mathbf{r}^k \leftarrow \mathbf{y} - \mathbf{A}^k \mathbf{s}^k$
 - 8: **end while**
 - 9: Output: $\mathbf{s}^* \leftarrow \mathbf{0}$ + corresponding entries from \mathbf{s}^k
-

We make two remarks about the OMP algorithm:

1. The algorithm adds basis vectors one at a time: we call this a *stepwise* algorithm;
2. The next basis to be added to the active set is determined by the basis vector \mathbf{a}_i with maximum inner product $\arg \max_i \{\mathbf{a}_i^T \mathbf{r}^{k-1}\}$ with the residual \mathbf{r}^{k-1} .

Recently, Donoho et al [4] introduced their *Stagewise Orthogonal Matching Pursuit* (StOMP), which is an extension to OMP that adds several basis vectors per *stage*, instead of one basis vectors per step (Algorithm 2). In their paper, Donoho

Algorithm 2 Stagewise Orthogonal Matching Pursuit [4]

- 1: Input: \mathbf{A} , \mathbf{y}
 - 2: Set stopping conditions l_{\max} and θ_{\min}
 - 3: Initialize: $k \leftarrow 0$, $\mathcal{I}^k \leftarrow \emptyset$, $\mathbf{A}^k \leftarrow \emptyset$, $\mathbf{r}^k \leftarrow \mathbf{y}$
 - 4: **while** $|\mathcal{I}^k| < l_{\max}$ and $\max_i \tilde{\mathbf{a}}_i^T \mathbf{r}^k > \theta_{\min}$ **do**
 - 5: $k \leftarrow k + 1$
 - 6: $\tau^k =$ a suitable threshold for step k [4]
 - 7: $\mathcal{J}^k = \{j \mid \mathbf{a}_j^T \mathbf{r}^{k-1} > \tau^k\}$
 - 8: Add to solution set:
 $\mathbf{A}^k \leftarrow [\mathbf{A}^{k-1}, \{\mathbf{a}_i \mid i \in \mathcal{J}^k\}]$, $\mathcal{I}^k \leftarrow \mathcal{I}^{k-1} \cup \mathcal{J}^k$,
 $\mathbf{s}^k \leftarrow (\mathbf{A}^k)^\dagger \mathbf{y}$, $\mathbf{r}^k \leftarrow \mathbf{y} - \mathbf{A}^k \mathbf{s}^k$
 - 9: **end while**
 - 10: Output: $\mathbf{s}^* \leftarrow \mathbf{0}$ + corresponding entries from \mathbf{s}^k
-

et al [4] choose a threshold at each step to be $\tau^k = \sigma^k t^k$ where $\sigma^k = \|\mathbf{r}^k\|_2 / \sqrt{n}$ is a formal noise parameter and t^k is a threshold parameter taking typical values in the range

$2 \leq t^k \leq 3$. The update to \mathbf{s}^k in step 8 is performed using an conjugate gradient solver.

Blumensath and Davies [5] proposed a related stagewise algorithm called *Stagewise Weak OMP* (SWOMP), which uses a different threshold $\tau^k = \beta \max_j \{\mathbf{a}_j^T \mathbf{r}^{k-1}\}$ for some $0 < \beta \leq 1$, so that step 7 in Algorithm 2 becomes

$$7: \mathcal{J}^k = \{j \mid \mathbf{a}_j^T \mathbf{r}^{k-1} > \beta \max_j \{\mathbf{a}_j^T \mathbf{r}^{k-1}\}\}.$$

For OMP this has the theoretical advantage that it has the same exact recovery criterion as Weak Orthogonal Matching Pursuit [9]. The main idea of both StOMP and SWOMP is to modify OMP to add several basis vectors at each stage, instead of only one per step. In this paper, we will apply this idea to produce a stagewise version of our previous Polytope Faces Pursuit algorithm [6].

2.2. The polytope perspective

We can get some useful insights into the sparse reconstruction problem from a geometrical perspective, by using the geometry of *polytopes* [10, 11]. To see this, we start from the traditional Basis Pursuit ℓ_1 -minimization problem (5) and express it as a linear program (LP) in its so-called *standard form* using non-negative coefficients [3]

$$\min_{\tilde{\mathbf{s}}} \mathbf{1}^T \tilde{\mathbf{s}} \quad \text{such that} \quad \mathbf{y} = \tilde{\mathbf{A}} \tilde{\mathbf{s}}, \quad \tilde{\mathbf{s}} \geq 0 \quad (6)$$

where $\mathbf{1}$ is a column vector of ones, $\tilde{\mathbf{s}} = (\tilde{s}_1 \dots, \tilde{s}_{2N})^T$ is the (doubled) nonnegative vector

$$\tilde{s}_i = \begin{cases} \max(s_i, 0) & 1 \leq i \leq N \\ \max(-s_{i-N}, 0) & N+1 \leq i \leq 2N \end{cases} \quad (7)$$

and $\tilde{\mathbf{A}} = [\mathbf{A}, -\mathbf{A}]$ is the corresponding doubled dictionary matrix which contains a positive and negative version of each basis vector of \mathbf{A} .

Now the linear program (6) has a corresponding *dual* linear program [3]

$$\max_{\mathbf{c}} \mathbf{y}^T \mathbf{c} \quad \text{such that} \quad \tilde{\mathbf{A}}^T \mathbf{c} \leq \mathbf{1} \quad (8)$$

such that a bounded solution to (8) exists if and only if a bounded solution to (6) exists. So to find a solution \mathbf{s}^* to (6) we can instead look for a solution \mathbf{c}^* to (8), and use the Karush-Kuhn-Tucker (KKT) conditions [12] to find the equivalent solution \mathbf{s}^* to (6).

Thus we need to find the point \mathbf{c} with maximum inner product with our input signal \mathbf{y} , such that it is contained within the *polytope* (bounded N -dimensional polygon) defined by the *polar polytope*

$$P^* = \{\mathbf{c} \mid \pm \mathbf{a}_i^T \mathbf{c} \leq 1, \mathbf{a}_i \in \mathbf{A}\} \quad (9)$$

which is dual to the *primal polytope*

$$P = \text{conv}\{\pm \mathbf{a}_i, \mathbf{a}_i \in \mathbf{A}\}.$$

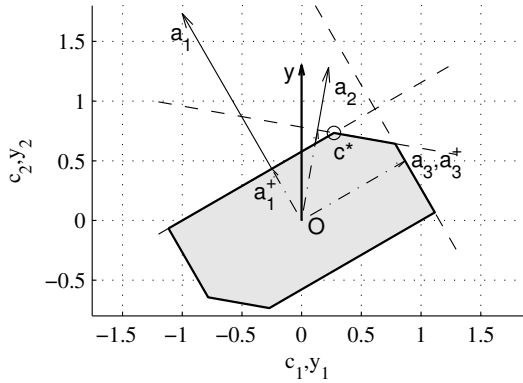


Fig. 1. Optimum point \mathbf{c}^* within polar polytope P^* (shaded).

Thus the primal polytope P is the convex hull of the basis vectors of $\pm \mathbf{a}_i$, while the polar polytope P^* is the region bounded by the half-planes $\pm \mathbf{a}_i^T \mathbf{c} \leq 1$. The optimum point \mathbf{c}^* is therefore the point in P^* , normally a vertex of P^* , which is as far in the direction of \mathbf{y} that it is possible to go without leaving P^* (Fig. 1).

This geometry leads to a projected gradient algorithm, whereby we start from $\mathbf{c} = \mathbf{0}$ and follow the gradient path initially in the direction $\dot{\mathbf{c}} = \mathbf{y}$ until we “hit” a face of P^* , then follow the projected gradient along that face until we hit the next face, and so on. Interestingly, at each step the next face to be hit corresponds to the basis vertex identified by the condition

$$\mathbf{a}^k = \arg \max_{\mathbf{a}_i \notin \tilde{\mathbf{A}}^k} \frac{\mathbf{a}_i^T \mathbf{r}^{k-1}}{1 - \mathbf{a}_i^T \mathbf{c}^{k-1}} \quad (10)$$

where $\mathbf{c}^k = (\tilde{\mathbf{A}}^k)^\dagger \mathbf{T} \mathbf{1}$ is the so-called “basis vertex” at step k , which is a point where $\mathbf{a}_i^T \mathbf{c}^k = 1$ for all currently active basis vectors $\mathbf{a}_i \in \tilde{\mathbf{A}}^k$ [6]. This is therefore slightly different to the normal admission criterion $\mathbf{a}^k = \arg \max_{\mathbf{a}_i \notin \tilde{\mathbf{A}}^k} \mathbf{a}_i^T \mathbf{r}^{k-1}$ used for OMP, although they are identical for the initial step, where $\mathbf{c}^k = \mathbf{0}$. Following the constrained gradient also requires us to *switch out* a basis vector \mathbf{a}_i whenever $\tilde{s}_i < 0$: this type of switching-out condition is not present in usual OMP-like algorithms. The full Polytope Faces Pursuit (PFP) algorithm is given in [6].

3. THE STAGewise PFP ALGORITHM

3.1. Algorithm

We now consider modifying the Polytope Faces Pursuit (PFP) algorithm by adding more than one vector to the active set per iteration. As we mentioned above, this idea has been included in the recent ‘stagewise’ algorithms of Donoho et al. [4] and Blumensath and Davies [5]. We will therefore refer to this approach as *Stagewise Polytope Faces Pursuit*.

This generalization consists only of a relatively straightforward change in implementation. Once all the adjusted

correlations are calculated, at stage k we select the $q^k \geq 1$ atoms with the largest adjusted correlations, for some fixed or adapted number q^k . We will discuss later (Section 3.3) some options for how we can choose this number q^k . In the polytope perspective, this step corresponds to taking into account the first q^k hyperplanes $\mathbf{a}_i^T \mathbf{c} = 1$ that are “pierced” as we follow the (projected) gradient $\dot{\mathbf{c}}$ towards \mathbf{y} . The Stagewise PFP

Algorithm 3 Stagewise Polytope Faces Pursuit

- 1: Input: $\tilde{\mathbf{A}} = [\tilde{\mathbf{a}}_i], \mathbf{y}$ {If required, set $\tilde{\mathbf{A}} \leftarrow [\mathbf{A}, -\mathbf{A}]$ }
 - 2: Set maximum number of bases to select per stage q_{\max}
 - 3: Set stopping conditions l_{\max} and θ_{\min}
 - 4: Initialize: $k \leftarrow 0, \mathcal{I}^k \leftarrow \emptyset, \tilde{\mathbf{A}}^k \leftarrow \emptyset, \mathbf{c}^k \leftarrow \mathbf{0}, \tilde{\mathbf{s}}^k \leftarrow \emptyset, \hat{\mathbf{y}}^k \leftarrow \mathbf{0}, \mathbf{r}^k \leftarrow \mathbf{x}$
 - 5: **while** $|\mathcal{I}^k| < l_{\max}$ and $\max_i \tilde{\mathbf{a}}_i^T \mathbf{r}^{k-1} > \theta_{\min}$ **do** {Find more faces}
 - 6: $k \leftarrow k + 1$
 - 7: $q^k =$ a suitable number of vectors to add
 - 8: Find first q^k faces encountered:
 $\mathcal{J}^k \leftarrow \arg \max_{i \notin \mathcal{I}^{k-1}} \{(\tilde{\mathbf{a}}_i^T \mathbf{r}^{k-1}) / (1 - \tilde{\mathbf{a}}_i^T \mathbf{c}^{k-1}) \mid \tilde{\mathbf{a}}_i^T \mathbf{r}^{k-1} > 0\}$
 - 9: Add constraints:
 $\tilde{\mathbf{A}}^k \leftarrow [\tilde{\mathbf{A}}^{k-1}, \{\mathbf{a}_i \mid i \in \mathcal{J}^k\}], \mathcal{I}^k \leftarrow \mathcal{I}^{k-1} \cup \mathcal{J}^k$
 - 10: $\tilde{\mathbf{s}}^k \leftarrow (\tilde{\mathbf{A}}^k)^\dagger \mathbf{x}$
 - 11: **while** $\tilde{\mathbf{s}}^k \not\geq \mathbf{0}$ **do** {Release retarding constraints}
 - 12: Select some $j \in \mathcal{I}^k$ such that $\tilde{s}_j^k < 0$; remove column \mathbf{a}_j from $\tilde{\mathbf{A}}^k$
 - 13: Update: $\mathcal{I}^k \leftarrow \mathcal{I}^k \setminus \{j\}, \tilde{\mathbf{s}}^k \leftarrow (\tilde{\mathbf{A}}^k)^\dagger \mathbf{x}$
 - 14: **end while**
 - 15: $\mathbf{c}^k \leftarrow (\tilde{\mathbf{A}}^k)^\dagger \mathbf{T} \mathbf{1}, \hat{\mathbf{y}}^k \leftarrow \tilde{\mathbf{A}}^k \tilde{\mathbf{s}}^k, \mathbf{r}^k \leftarrow \mathbf{y} - \hat{\mathbf{y}}^k$
 - 16: **end while**
 - 17: Output: $\mathbf{c}^* = \mathbf{c}^k,$
 $\tilde{\mathbf{s}}^* \leftarrow \mathbf{0} +$ corresponding entries from $\tilde{\mathbf{s}}^k$
 {If required, get $s_i^* \leftarrow (\tilde{s}_i^* - \tilde{s}_{i+n}^*), 1 \leq i \leq n$ }
-

algorithm is summarized in Algorithm 3. In this algorithm, the notation $\arg \max_i^q \{f(\cdot)\}$ means the set of values of i corresponding to the q maximal values of $f(\cdot)$. If we use $q^k = 1$ face per step we obtain the original PFP algorithm [6].

As well as the matrix \mathbf{A} and the observation vector \mathbf{y} , the algorithm uses as stopping criteria l_{\max} , the maximum number of basis vectors to select (maximal K -sparsity of the resulting vector \mathbf{s}^*), and θ_{\min} , the minimum residual correlation.

3.2. Calculating the pseudoinverse

One of the computationally expensive parts of this algorithm, and of other OMP-like algorithms, is the calculation of the Moore-Penrose pseudo-inverse $(\tilde{\mathbf{A}}^k)^\dagger$. A standard method, as used for example in the Matlab *pinv* function, is to use Singular Value Decomposition (SVD). Here, the matrix is decomposed as $\tilde{\mathbf{A}}^k = \mathbf{U} \Sigma \mathbf{V}^T$, where \mathbf{U} and \mathbf{V} are unitary ma-

trices and Σ is a diagonal matrix. The pseudoinverse is then $(\tilde{\mathbf{A}}^k)^\dagger = \mathbf{V}\Sigma^\dagger\mathbf{U}^T$.

However, since OMP-like algorithms update the matrix \mathbf{A} one column at a time, it can be more efficient to compute the pseudoinverse by using the Cholesky factorization. If \mathbf{A} is has full column rank, then its pseudoinverse is given by $\mathbf{A}^\dagger = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T = (\mathbf{R}^T\mathbf{R})^{-1}\mathbf{A}^T$, where \mathbf{R} is an upper triangular matrix, forming the Cholesky factorization of the symmetric matrix $\mathbf{A}^T\mathbf{A}$. The Cholesky factor \mathbf{R} can be updated as a single column is added to the matrix \mathbf{A} .

Now our Stagewise algorithm allows $q > 1$ basis vectors to be added to $\tilde{\mathbf{A}}^k$ at each stage. To update the Cholesky factor \mathbf{R} as q columns are added to $\tilde{\mathbf{A}}^k$, we could perform q 1-column updates. As an alternative, we have also implemented a direct q -column Cholesky update, which is conceptually similar to the 1-column update, but requires an additional direct Cholesky factorization on a $q \times q$ matrix corresponding to the orthogonalized components of the q new vectors. All of these updates require that the matrix $\tilde{\mathbf{A}}^k$ retains full rank at each stage.

3.3. Basis selection strategies

In each stage of the Stagewise PFP algorithm we have to determine q^k , the number of basis vectors to add at each iteration. One simple approach is the *fixed* q method, where we select a fixed number $q^k = q$ of basis vectors with the largest adjusted correlations at each stage. However, alternative strategies for the algorithm selection step are possible.

One alternative we have considered is *weak stagewise selection* approach, derived by the weak-OMP (WOMP) algorithm of Davies and Blumensath [13]. For our version, we let

$$\theta(\mathbf{a}_i^k) = \frac{\mathbf{a}_i^T \mathbf{r}^{k-1}}{1 - \mathbf{a}_i^T \mathbf{c}^{k-1}} \quad (11)$$

be the adjusted correlation for the basis vector \mathbf{a}_i at step k . Then, our weak stagewise selection step consists of selecting all basis vectors \mathbf{a}_i such that:

$$|\theta(\mathbf{a}_i^k)| \geq \beta \max_i |\theta(\mathbf{a}_i^k)| \quad (12)$$

where $0 < \beta \leq 1$ is a threshold control parameter. This selection criterion (12) selects basis vectors by applying thresholding to the adjusted correlations relative to the maximum. If $\beta = 1$ we will only select the basis vector with the largest adjusted correlation (10), and therefore (provided the basis vectors are in general position) we will have the stepwise one-atom algorithm. We normally impose an upper limit q_{\max} , which is the maximum number of bases selects at each stage.

Note that although this criterion has a resemblance to the weak OMP criterion considered by Tropp [9], since we use adjusted correlations (11) in our polytope algorithm rather than the direct correlations $\mathbf{a}_i^T \mathbf{r}^{k-1}$ used in standard MP/OMP, the theoretical ‘‘Weak OMP’’ results of Tropp [9] do not necessarily hold for this ‘‘Weak PFP’’ criterion.

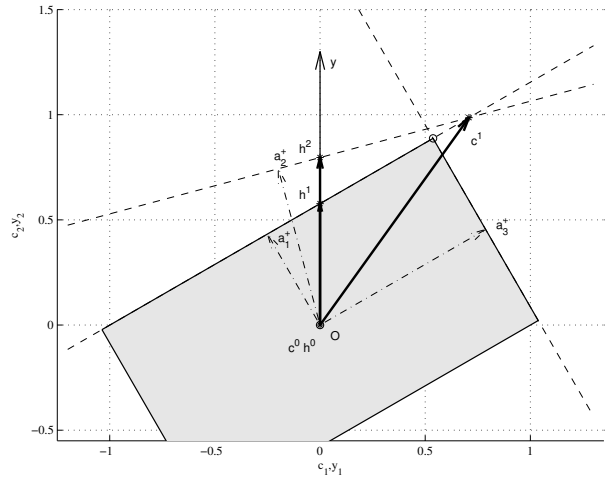


Fig. 2. Degenerate case in the Stagewise PFP algorithm.

We could also adopt a noise-based criteria similar to that used by Dohono et al. [4] in their StOMP algorithm, but that would need adapting to our adjusted correlation criterion and we do not consider that further in the present paper.

3.4. The degenerate case

One possible drawback of the Stagewise PFP algorithm is that the selection of $q > 1$ basis vectors at once no longer guarantees that the resulting \mathbf{c} will stay inside the polar polytope P^* . If this happens, we call this a *degenerate case*.

An example degenerate case is shown in Fig. 2. In the figure, we have a 2-D polar polytope P^* with six signed basis vectors $\pm\mathbf{a}_i$, $i = 1, 2, 3$. Let us apply the Stagewise PFP algorithm to this arrangement with $q = 2$ faces selected per stage. As we can see from the figure, the first two faces encountered are those corresponding to the basis vectors \mathbf{a}_1 and \mathbf{a}_2 . However, the selection of these bases leads us to a degenerate case, since the resulting vertex $\mathbf{c}^1 = (\tilde{\mathbf{A}}^1)^\dagger \mathbf{1}$, where $\tilde{\mathbf{A}}^1 = [\mathbf{a}_1, \mathbf{a}_2]$, turns out to be outside the polytope constraint set. Here the constraints $\mathbf{c}^T \mathbf{a}_3 \leq 1$ is violated (the point \mathbf{c}^1 is to the right of the dotted line $\mathbf{c}^T \mathbf{a}_3 = 1$), so we have $\mathbf{c}^1 \notin P^*$.

If the degenerate case occurs, we can no longer ensure that the reconstructed vector $\hat{\mathbf{s}}$ is strictly optimal. However, experimental results so far suggest that this problem does not appear to cause significantly ‘‘bad’’ estimates $\hat{\mathbf{s}}$ of the sparse vector to occur. We are currently investigating techniques to either avoid this degenerate case, or to ‘‘pull back’’ into the constraint surface if the degenerate case is detected, but in the meantime our results to date indicate that the algorithm as presented produces good results in practice, even though ℓ_1 -optimality is no longer guaranteed.

4. EXPERIMENTS

To confirm the suitability of the Stagewise PFP algorithm as an algorithm for sparse reconstruction in Compressed Sensing problems, we have applied it to the following (real domain) problems from the Sparco toolbox [14]:

- Problem 5 (gcoospike): Cosine with spikes measured with Gaussian ensemble;
- Problem 6 (p3poly): Piecewise cubic polynomial;
- Problem 11 (gausspike): Gaussian spikes, real domain.

We will first give results for Problem 5 (gcoospike), with implementation and selection options discussed in sections 3.2 and 3.3. We will give additional results for the other problems.

4.1. Sparco Problem 5: Fixed- q selection

In Sparco Problem 5 the signal \mathbf{x} is a 1024×1 column vector which is represented by a cosine-like function with some random spikes added. The signal is measured by a Gaussian ensemble, taking $M = 300$ random measurements \mathbf{y} , while the basis matrix Ψ is a 1024×2048 rectangular matrix consisting of two separate dictionaries (DCT and Dirac bases). The resulting matrix \mathbf{A} is therefore a 300×2048 matrix.

We first consider the *fixed- q* selection strategy, in which the number of bases to select per stage is fixed at $q^k = q$. We evaluated the performance of the Stagewise PFP algorithm varying the implementation strategy for the pseudoinverse stage, according to the three criteria mentioned in section 3.2. For all the tests made, the stopping conditions were set to $\theta_{\min} = 10^{-2}$ and $l_{\max} = 100$. The behaviour of the algorithm in terms of running time (averaged over 5 runs) and recovery accuracy (MSE of the recovered signal $\hat{\mathbf{x}}$) is shown in Fig. 3, which shows the performance of the n -column Cholesky update (Fig. 3(a)), the single Cholesky update (Fig. 3(b)) and Matlab *pinv* function (Fig. 3(c)).

As we can see from the Figure, when we use a Cholesky update (either n -update or single update) for the pseudoinverse implementation step, the stagewise algorithm can lead to almost a factor of 3 speed-up when compared to the original stepwise ($q = 1$) version. The best performance on this problem is for q in the range $13 \leq q \leq 17$ basis vectors per stage, while recovery errors are still kept very small.

For the direct pseudoinverse implementation via the Matlab *pinv* function, we get a speed benefit for only very relatively small values of q_{\max} (up to 3): above this the behaviour of the algorithm appears to become somewhat irregular.

Comparing the three different pseudoinverse implementations, we find that using the Cholesky factorization, either the n -update or single update form, is considerably faster than the pseudoinverse direct method, with a speed-up factor between 4 and 10 for the same number of bases per stage (Fig. 4).

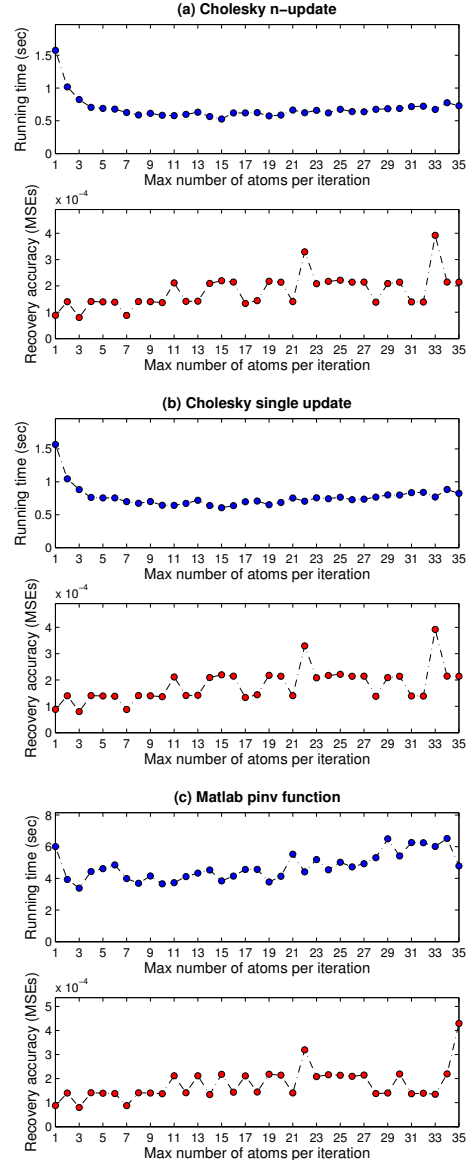


Fig. 3. Sparco problem 5: Running time and recovery accuracy against number of atoms added per iteration, as the pseudoinverse step is implemented by using (a) a n -column Cholesky update, (b) a single Cholesky update or (c) the Matlab *pinv* function.

Of the Cholesky factorization methods, the n -column update is slightly faster than the single update, although only by a relatively small margin.

4.2. Sparco Problem 5: Weak stagewise selection

In section 3.3 we mentioned two different selection strategies: fixed- q selection, as explored for Sparco Problem 5 in the results above, and weak stagewise selection. Fig. 5 shows the performance of Stagewise PFP on Sparco Problem 5 using

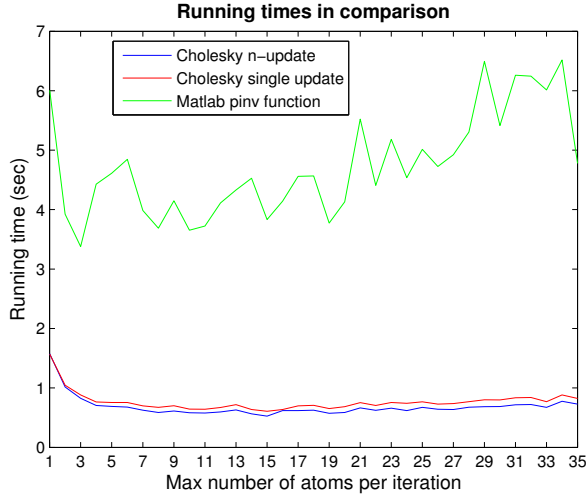


Fig. 4. Sparco problem 5: Running times in comparison as the pseudoinverse step implementation varies.

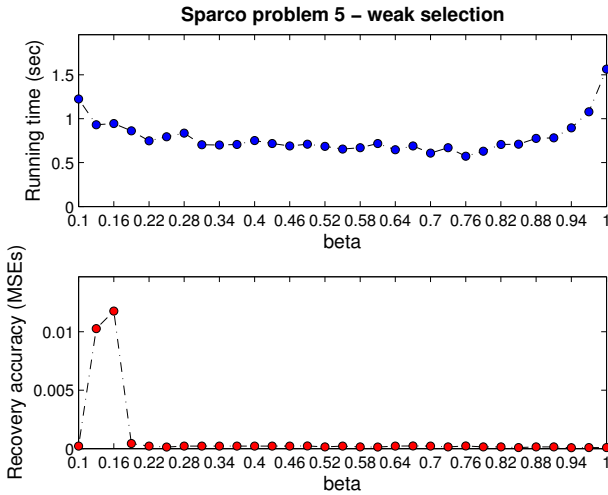


Fig. 5. Sparco problem 5: Running time and recovery accuracy using the weak stagewise selection criterion, as the threshold control parameter β varies.

the weak selection strategy and the n -update Cholesky implementation: we can compare with the results for fixed- q selection in Fig. 3. We also show the comparison in tabular form in Table 1 (fixed- q selection) and 2 (weak selection), which demonstrate that the results for these two criteria are largely comparable, with fastest results around $\beta = 0.7$. We notice in particular that the cases $q = 1$ and $\beta = 1$ give similar results, with both corresponding to the original (stepwise) PFP algorithm, differing only due to minor implementation issues. Very small values of the threshold β , e.g. $\beta \lesssim 0.2$ appear to lead to increased recovery errors on this problem, perhaps since this can lead to a large number of basis functions to be

q	Time (sec)	MSE / 10^{-5}
1	1.57	8.8
3	0.83	8.0
5	0.71	13.9
7	0.64	8.7
9	0.63	14.0
11	0.58	21.1
13	0.64	14.2
15	0.53	21.9
17	0.62	13.3
19	0.58	21.7

Table 1. Sparco problem 5: Running times and MSEs against selection parameter q (fixed- q selection strategy).

β	Time (sec)	MSE / 10^{-5}
0.46	0.68	20.8
0.52	0.68	14.4
0.58	0.67	13.9
0.64	0.64	21.1
0.7	0.60	21.2
0.76	0.57	22.0
0.82	0.70	14.0
0.88	0.77	14.0
0.94	0.89	7.9
1	1.56	8.8

Table 2. Sparco problem 5: Running times and MSEs against selection parameter β (weak stagewise selection strategy).

admitted at each stage.

We can verify that the weak stagewise selection strategy can lead to very different numbers of basis vectors selected at each stage. Fig. 6 illustrates this, showing the number of basis vectors selected at each stage when we use the weak stagewise selection with $\beta = 0.7$ on Problem 5. We see, for example, that at the first three stages only one basis vector is selected per stage, but by step 5 a very large number of basis vectors (36) are selected. Observations confirm that many of these will be switched out by the second part of the algorithm, but this indicates that there may need to be some refinement to this weak selection method to tailor it for the Stagewise PFP algorithm.

4.3. Sparco Problems 6 and 11

To confirm our results on other Compressed Sensing problems, we also tested the algorithm on Sparco Problem 6 and Sparco Problem 11. In both cases the pseudoinverse step is implemented via the n -column Cholesky update, and a fixed- q selection strategy is used.

In Problem 6 the signal \mathbf{x} is a piecewise cubic polynomial measured by a Gaussian ensemble, and the representation ma-

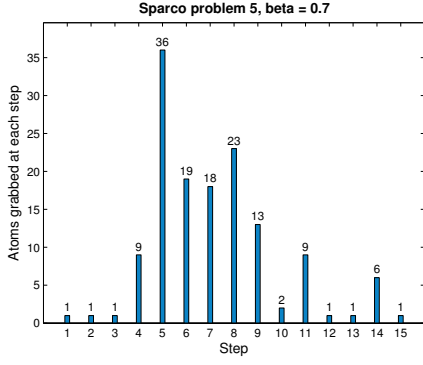


Fig. 6. Sparco problem 5: Number of bases selected per stage, using the weak selection strategy with $\beta = 0.7$.

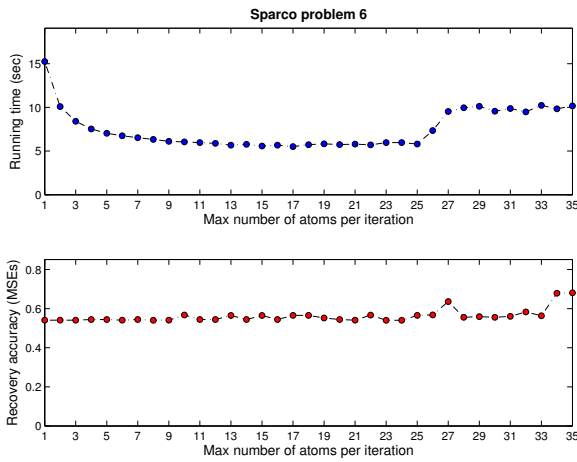


Fig. 7. Sparco Problem 6: Running time and recovery accuracy against number of atoms added per iteration (fixed- q criterion), pseudoinverse step implemented using a n -column Cholesky update.

trix consists of a wavelet bases. The dimension of the matrix is \mathbf{A} is 600×2048 . Results are shown in Fig. 7.

In Problem 11 we have spikes whose amplitude follow a Gaussian distribution. The measurement process is performed by a Gaussian ensemble again, whereas this time sparsity is exactly achieved with an Identity matrix as the basis matrix. In this case the dimension of the problem is 256×1024 . Results are shown in Fig. 8.

Performance on both of these problems follows similar trends to that for Problem 5. In both cases the stagewise algorithm gives faster results than the stepwise ($q = 1$) algorithm, and the accuracy recovery does not appear to be significantly affected by selection of multiple basis vectors.

However, for Problem 11, we note that the running time increases approximately above about $q = 11$, and can become significantly slower than the original stepwise ($q = 1$) algo-

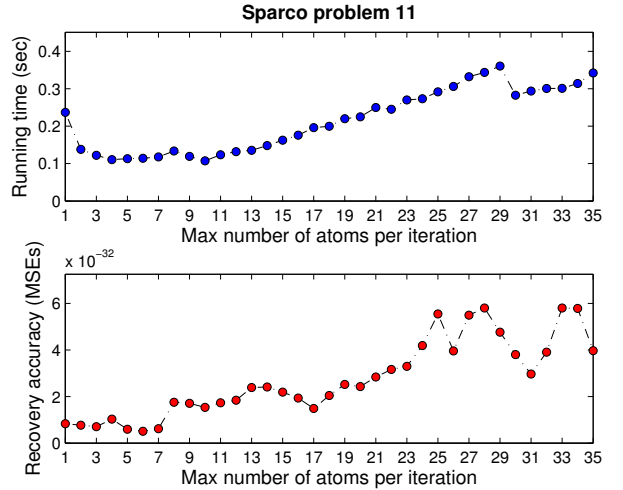


Fig. 8. Sparco Problem 11: Running time and recovery accuracy against number of atoms added per iteration (fixed- q criterion), pseudoinverse step implemented using a n -column Cholesky update.

riith. Preliminary investigations suggest that, although the recovery error is still quite small, the algorithm may spend considerable time switching out several of the “extra” basis vectors that have been added. This confirms the need to choose an appropriate basis selection strategy.

5. CONCLUSIONS

We have introduced a new stagewise greedy algorithm to find an approximation of the underlying sparse representation \mathbf{s} of a real signal \mathbf{x} , such that it satisfies the typical Compressed Sensing equation $\mathbf{y} = \mathbf{A}\mathbf{s}$, where \mathbf{y} is the undersampled observation vector. We call this method *Stagewise Polytope Faces Pursuit*.

This algorithm is the generalization of our previous Polytope Faces Pursuit algorithm, in that it adds several basis vectors per iteration instead of selecting only one basis vector at a time. Both algorithms rely on the the dual linear program $\max_{\mathbf{c}} \{\mathbf{y}^T \mathbf{c} \mid \mathbf{A}^T \mathbf{c} \leq \mathbf{1}\}$, which is the dual of the classic Basis Pursuit linear program. The feasible region of the dual LP is defined by the polar polytope $P^* = \{\mathbf{c} \mid \tilde{\mathbf{A}}^T \mathbf{c} \leq \mathbf{1}\}$, where $\tilde{\mathbf{A}} = [\mathbf{A}, -\mathbf{A}]$ is the doubled basis matrix. These algorithms adopt a path-following approach through the relative interior of the faces of the polar polytope, updating the position of the current vertex \mathbf{c} .

We have tested two different stagewise selection strategies: a fixed- q strategy, in which we just select the top q bases at each iteration, and a weak selection strategy, based on the Weak OMP approach.

We have applied our new stagewise algorithm to example Compressed Sensing problems from the Sparco problem suite

[14]. Our results indicate that the stagewise generalization can speed up the operation of our algorithm by up to a factor of 3, while obtaining similar recovery errors, demonstrating that the Stagewise PFP algorithm is suitable for solving compressed sensing problems.

In future work, we plan to compare the Stagewise PFP algorithm to a range of other sparse reconstruction algorithms. We also plan to investigate the effect of the degenerate case, when the point c leaves the constraint set, and to design possible strategies to overcome this if necessary. We also intend to investigate alternative selection strategies, and explore further possible speed improvements by replacing the Cholesky-based calculation of the pseudoinverse with a conjugate gradient (CG) approach.

6. REFERENCES

- [1] D.L. Donoho, "Compressed sensing," *IEEE Transactions on Information Theory*, vol. 52, no. 4, pp. 1289–1306, April 2006.
- [2] R. Baraniuk, "Compressive sensing," *IEEE Signal Processing Magazine*, vol. 24, no. 4, pp. 118–121, July 2007.
- [3] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM Journal on Scientific Computing*, vol. 20, no. 1, pp. 33–61, 1998.
- [4] D. L. Donoho, Y. Tsaig, I. Drori, and J.-L. Starck, "Sparse solution of underdetermined linear equations by stagewise orthogonal matching pursuit," Tech. Rep. 2006-2, Department of Statistics, Stanford University, 2006.
- [5] T. Blumensath and M. E. Davies, "Stagewise weak gradient pursuits. Part I: Fundamentals and numerical studies," Submitted, 17 Sept. 2008.
- [6] M. D. Plumbley, "Recovery of sparse representations by polytope faces pursuit," in *Proceedings of the 6th International Conference on Independent Component Analysis and Blind Source Separation (ICA 2006)*, Charleston, SC, USA, 5–8 March 2006, pp. 206–213, LNCS 3889.
- [7] S. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Transactions on Signal Processing*, vol. 41, no. 12, pp. 3397–3415, 1993.
- [8] Y. C. Pati, R. Rezaifar, and P. S. Krishnaprasad, "Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition," in *Conference Record of The Twenty-Seventh Asilomar Conference on Signals, Systems and Computers, Pacific Grove, CA*, 1-3 Nov. 1993, pp. 40–44.
- [9] J. A. Tropp, "Greed is good: Algorithmic results for sparse approximation," *IEEE Transactions on Information Theory*, vol. 50, no. 10, pp. 2231–2242, Oct. 2004.
- [10] D. L. Donoho, "Neighborly polytopes and sparse solutions of underdetermined linear equations," Tech. Rep., Statistics Department, Stanford University, December 2004.
- [11] M. D. Plumbley, "On polar polytopes and the recovery of sparse representations," *IEEE Transactions on Information Theory*, vol. 53, no. 9, pp. 3188–3195, Sept. 2007.
- [12] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004.
- [13] M. E. Davies and T. Blumensath, "Faster & greedier: Algorithms for sparse reconstruction of large datasets," in *Proc. 3rd International Symposium on Communications, Control and Signal Processing ISCCSP 2008*, 2008, pp. 774–779.
- [14] E. van den Berg, M. P. Friedlander, G. Hennenfent, F. Herrmann, R. Saab, and Ö. Yılmaz, "Sparco: A testing framework for sparse reconstruction," Tech. Rep. TR-2007-20, Dept. Computer Science, University of British Columbia, Vancouver, October 2007.