Stagewise Weak Gradient Pursuits

Thomas Blumensath, *Member, IEEE,* Mike E. Davies, *Member, IEEE*

***Abstract*— Finding sparse solutions to underdetermined inverse problems is a fundamental challenge encountered in a wide range of signal processing applications, from signal acquisition to source separation. This paper looks at greedy algorithms that are applicable to very large problems. The main contribution is the development of a new selection strategy (called stagewise weak selection) that effectively selects several elements in each iteration. The new selection strategy is based on the realisation that many classical proofs for recovery of sparse signals can be trivially extended to the new setting. What is more, simulation studies show the computational benefits and good performance of the approach. This strategy can be used in several greedy algorithms and we argue for the use within the gradient pursuit framework in which selected coefficients are updated using a conjugate update direction. For this update, we present a fast implementation and novel convergence result.**

***Index Terms*— Sparse Representations/Approximations, Orthogonal Matching Pursuit, Weak Matching Pursuit, Gradient Pursuit, Stagewise Selection, Compressed Sensing.**

I. Introduction

Sparse signal expansions are general signal models, applicable to a wide range of signals, that approximate a signal using a linear combination of a small number of elementary waveforms selected from a large collection. These models have over the last few years found applications in a wide range of areas, from source coding [2] to de-noising [3], source separation [4] and signal acquisition [5] (i.e. compressed sensing).

A sparse signal model is specified by a matrix $\mathbf{\Phi} \in \mathbb{R}^{M\times N}$ with typically more columns than rows, that is with $M < N$. $\mathbf{\Phi}$ is often known as the dictionary or the measurement matrix, depending on the application. The column vectors ϕ_i of $\mathbf{\Phi}$ are sometimes called atoms and are here assumed to be of unit length unless stated otherwise. Given an observation $\mathbf{x} \in \mathbb{R}^M$, a sparse signal model approximates $\mathbf{x}$ using a small subset of columns from $\mathbf{\Phi}$, i.e.

$$\hat{\mathbf{x}} = \mathbf{\Phi}\hat{\mathbf{y}},$$

where $\hat{\mathbf{y}}$ is a vector with most of it's elements being zero. If we allow for a non-zero error $\mathbf{n} = \mathbf{x} - \hat{\mathbf{x}}$ we talk about a signal *approximation*, while if $\hat{\mathbf{x}} = \mathbf{x}$ we have an exact signal *representation*.

If $M < N$, then there are an infinite number of $\hat{\mathbf{y}}$ such that $\mathbf{x} = \mathbf{\Phi}\hat{\mathbf{y}}$. The problem is then to find an estimate $\hat{\mathbf{y}}$ that is sparse, such that the norm of $\mathbf{n}$ is small. Whilst there are a range of ways in which sparsity could be measured, the most typical is probably to count the number of elements of $\hat{\mathbf{y}}$ that are non-zero.

The problem of finding a vector $\hat{\mathbf{y}}$ with the smallest number of non-zero coefficients, under a constraint on $\|\mathbf{n}\|_2$ is known to be NP-hard in general [6], [7] and different sub-optimal strategies are used in practise. Commonly used strategies are typically based on convex relaxation, non-convex (often gradient based) local optimisation or greedy search strategies. Convex relaxation, one of the most popular strategies at the moment, is used in approaches such as Basis Pursuit and Basis Pursuit De-Noising [8], the Least Absolute Shrinkage and Selection Operator (LASSO) and Least Angle Regression (LARS) [9]. Recently, fast algorithms solving the LASSO or the Basis Pursuit De-Noising problem have been suggested in [10], [11], [12], [13] and [14]. Non-convex local optimisation procedures include the Focal Underdetermined System Solver FOCUSS [15] and re-weighted ℓ_1 minimisation [16] while Bayesian approaches include the Relevance Vector Machine, also known as Sparse Bayesian Learning [17] [18] or Monte Carlo based approaches such as those in [19], [20] and [21]. Another very popular approach is to use greedy algorithms, the most important of which are Matching Pursuit (MP) [22], Orthogonal Matching Pursuit (OMP) [23] and Orthogonal Least Squares (OLS) [24], also often known as ORMP, OOMP or, in the regression literature, as forward selection. Extensions to more general cost functions and kernel dictionaries are discussed in [25]. OMP typically shows greatly superior performance to MP, however, OMP is more costly in both computation time and storage requirements.

During the preparation of this article, another family of greedy algorithms has been emerging. These algorithms not only select new elements, but also include an element pruning step. These algorithms include the Subspace Pursuit algorithm [26], the Compressed Sensing Matching Pursuit (CoSaMP) [27] and the Iterative Hard Thresholding algorithm [28].

In this paper we concentrate on OMP type algorithms and their application to very large data sets. There are two main problems associated with the application of OMP to large data sets. On the one hand, the computation cost per iteration is high, both in terms of storage and computation. This problem was addressed in [29], where we have introduced a quite general framework for greedy algorithms, called collectively Gradient Pursuits. Based on this idea, we have developed two particular algorithms, with the computational complexity of MP, but with performance more akin to OMP. Compared to

The authors are with IDCOM & Joint Research Institute for Signal and Image Processing, Edinburgh University, King's Buildings, Mayfield Road, Edinburgh EH9 3JL, UK (Tel.: +44(0)131 6505659, Fax.: +44(0)131 6506554, e-mail: thomas.blumensath@ed.ac.uk, mike.davies@ed.ac.uk).

This research was supported by EPSRC grant D000246/1. MED acknowledges support of his position from the Scottish Funding Council and their support of the Joint Research Institute with the Heriot-Watt University as a component part of the Edinburgh Research Partnership. The authors would like to thank Ian Marshall and Yuehui Tao for providing access to the MRI image data.

Some of the work presented here has previously been presented in [1].

OMP, this has greatly reduced the computational and storage requirements per iteration, making the method applicable to large data sets.

However, another performance limitation of greedy methods such as MP, OMP as well as the Gradient Pursuits algorithms of [29], is that these methods select a single element per iteration. They have therefore to be run for at least as many iterations as there are non-zero elements in the solution. The main contributions of this paper is therefore the development of a novel *Stagewise Weak* selection procedure that will allow several elements to be selected in each iteration. This new selection strategy, combined with the strategies from [29], will be shown to lead to very fast and efficient algorithms to solve the sparse signal modelling problem.

A. Paper Overview

The algorithms developed in this paper fall into the category of greedy pursuit algorithms which are discussed in Section II. We here concentrate on the two main aspects of these algorithms, element selection and coefficient update and review the state of the art approaches currently in use. In Section III we look at element selection and discuss several drawbacks of current approaches. To overcome these, we devise a novel *Stagewise Weak* selection strategy in Subsection III-A. Experimental results in Subsection III-C highlight several advantages of this new approach.

We then turn to the problem of efficient coefficient updating in Section IV. We quickly review standard approaches and then discuss the Gradient Pursuit algorithms and, in particular, the (approximate) Conjugate Gradient Pursuit algorithm (CGP) (Subsection IV-A). We here propose a novel fast implementation for this method and derive a novel convergence result for this approach (Subsection IV-C).

In Section V we have then all ingredients for the *Stagewise Weak Gradient Pursuit* algorithm which combines the new Stagewise Weak selection strategy of Section III, with the Conjugate Gradient Pursuit update of Section IV. Subsections V-A and V-B present numerical results that demonstrate the advantages of this approach, whilst theoretical properties of the stagewise weak methods will be studied in Section VI.

B. Notation

The algorithms in this paper are iterative and the current iteration will be iteration n. The algorithms will keep track of a set $\Gamma^{[n]}$ of indices, that will be grown in each iteration. These indices label a subset of columns from a matrix $\mathbf{\Phi}$ and, using the index set as a subscript, the matrix $\mathbf{\Phi}_{\Gamma^{[n]}}$ will be a sub-matrix of $\mathbf{\Phi}$ containing only those columns of $\mathbf{\Phi}$ with indices in $\Gamma^{[n]}$. The same convention is used for vectors. In general, the superscript in the subscript of $\hat{\mathbf{y}}_{\Gamma^{[n]}}$ reminds us that we are in iteration n, on occasion, however, we resort to using additional superscripts (e.g. $\hat{\mathbf{y}}^{[n]}$) to label the iteration. The Gram matrix $\mathbf{G}_{\Gamma^{[n]}} = \mathbf{\Phi}_{\Gamma^{[n]}}^T \mathbf{\Phi}_{\Gamma^{[n]}}$ will also be used frequently. In general, lower case bold face characters represent vectors while upper case bold characters are used for matrices. Individual elements from a vector will be in standard type face with a subscript. For example $\mathbf{g}$ will be used to refer to a negative gradient vector with g_i denoting the i^{th} element of this vector. Inner products between vectors will often be written using angled brackets, e.g. $\langle \mathbf{x}, \mathbf{y} \rangle = \mathbf{x}^T \mathbf{y}$. We will further use the hat $\hat{}$ to distinguish an estimated quantity from the true quantity, which will be written without the hat.

II. Greedy Pursuits

The algorithms discussed and developed in this paper are all part of a general family of iterative greedy pursuit algorithms. Given a vector $\mathbf{x}$ and a matrix $\mathbf{\Phi}$, the aim of these algorithms is to identify a 'small' set Γ and a vector $\hat{\mathbf{y}}_\Gamma$ to approximate $\mathbf{x}$ using

$$\hat{\mathbf{x}} = \mathbf{\Phi}_\Gamma \hat{\mathbf{y}}_\Gamma.$$

The algorithms are initialised by setting the first residual $\mathbf{r}^{[0]} = \mathbf{x}$, setting $\hat{\mathbf{y}}^{[0]} = \mathbf{0}$ and the set $\Gamma^{[0]} = \emptyset$. Each iteration then updates these three quantities. In general, this is done as follows[1]:

1) Calculate $\mathbf{g}^{[n]} = \mathbf{\Phi}^T \mathbf{r}^{[n]}$ and select columns from $\mathbf{\Phi}$ based on the magnitude of the elements in $\mathbf{g}^{[n]}$. The indices of the selected elements are added to $\Gamma^{[n-1]}$.
2) Calculate an estimate of $\hat{\mathbf{y}}^{[n]}_{\Gamma^{[n]}}$ that reduces the cost $\|\mathbf{x} - \mathbf{\Phi}_{\Gamma^{[n]}} \hat{\mathbf{y}}_{\Gamma^{[n]}}\|_2^2$.
3) Update $\mathbf{r}^{[n+1]} = \mathbf{x} - \mathbf{\Phi} \hat{\mathbf{y}}^{[n]}$.

Different algorithms differ in steps (1) and (2). In step (1), traditional approaches select a single element in each iteration, however, it has been realised that this might be too slow in many applications and recently methods have been put forward that select several elements at a time. In step (2) an update strategy is generally used that aims at reducing the cost $\|\mathbf{x} - \mathbf{\Phi}_{\Gamma^{[n]}} \hat{\mathbf{y}}_{\Gamma^{[n]}}\|_2^2$.

III. Stagewise Weak Element Selection

This section looks at the element selection step and suggests a novel selection strategy. However, to motivate the introduction of our new approach, we first review some of the currently used approaches. Possibly one of the simplest selection strategies is the approach used in MP and OMP. Here, a *single* element is selected in each iteration[2]. This selection is based on the magnitude of the elements in $\mathbf{g}^{[n]}$ and the index of the element with the largest magnitude is added to the index set

$$\Gamma^{[n]} = \Gamma^{[n-1]} \cup \arg_i \max |g_i^{[n]}|.$$

Another very simple selection strategies is thresholding. For a given threshold λ one selects indices from the set

$$\{i : |g_i^{[n]}| \geq \lambda\},$$

where λ can either be a constant or can depend on other quantities. For example, λ can depend on $\{g_i\}$ in such a way

[1]Note that depending on the detailed implementation of the algorithm, the matrix vector product in step (3) can often be replaced by a fast recursion.

[2]In an exact implementation of OMP elements will only be selected once, because the orthogonal projection used in the coefficient update (see below) ensures that the residual $\mathbf{r}^{[n]}$ is orthogonal to all columns in $\mathbf{\Phi}_{\Gamma^{[n]}}$. However, if this orthogonalisation is only approximated as, for example, in the Stagewise OMP (StOMP) algorithm [30] (discussed below) or in Gradient Pursuit [29], it is advisable (both from theoretical arguments and empirical evidence) to allow the algorithm to re-select elements.

that the set of selected elements contains a specified number of elements. In the simplest uses of thresholding, one does not iterate[3]. Instead a single selection step is used followed by optimisation as in step (2).

One of the first approaches that used thresholding ideas in an iterative framework is that used in StOMP [30]. In this approach a threshold is calculated depending on the current residual $\mathbf{r}$

$$\lambda_{stomp} = t\|\mathbf{r}^{[n-1]}\|_2/\sqrt{M},$$

where M is the dimension of $\mathbf{x}$. The set of indices is then updated as

$$\Gamma^{[n]} = \Gamma^{[n-1]} \bigcup \{i : |g_i| \geq \lambda_{stomp}\}. \tag{1}$$

The selection strategy in StOMP was developed explicitly for problems in which $\mathbf{\Phi}$ has been generated from a Uniform Spherical Ensemble, i.e. the columns of $\mathbf{\Phi}$ are drawn uniformly from the unit sphere. Theoretical performance guarantees for this method when applied to more general matrices $\mathbf{\Phi}$ are therefore not available. From a practical point of view, the selection of the parameter t required in the method is critical for its performance, but there do not seem to be any intuitive guidelines available for this other than the suggestion in [30] to use a value between 2 and 3. Furthermore, a problem we encountered when using the residual to define a threshold is that the algorithm might (and in our experience sometimes does) get 'stuck' when all inner products fall below the threshold. It would then be necessary to reduce the parameter t. In many of our own experiments (see below), this approach has therefore shown mixed results.

An alternative approach is used in the Regularised OMP (ROMP) algorithm [31], [32] which groups the inner products g_i into sets J_k such that the elements in each set have a similar magnitude, i.e. they satisfy

$$|g_i| \leq \frac{1}{r}|g_j|, \text{ for all } i, j \in J_k.$$

ROMP then selects the set J_k for which $\|\mathbf{g}_{J_k}\|_2$ is largest.

For the ROMP selection strategy proposed in [31] and [32], r was assumed to be 0.5. In this case, the algorithm was shown to have uniform performance guarantees similar to those of ℓ_1 based methods. Whilst these results indicate that, asymptotically for very large N, the performance of ROMP should be better than that of OMP, the particular constants involved in the theoretical guarantees are significantly smaller than those in the equivalent statements for ℓ_1 methods. Unfortunately, practice is generally far from asymptotia and one is often interested in applying the method to problems in which the theoretic properties do not hold. Furthermore, in many practical situations, one might be more interested in average rather than worst case performance. In almost all practical applications we have studied (see below), the average performance of ROMP was notably worse than that of OMP or StOMP.

[3]Note that methods such as Subspace Pursuit [26], CoSaMP [27] and the Iterative Hard Thresholding [28] use thresholding ideas in an iterative manner. These methods do however not fall into the greedy pursuit framework discussed here as selected elements are not simply added to previously selected elements, instead, thresholding is also used to prune out already selected elements.

A. Stagewise Weak element selection

As discussed above, the selection strategies of StOMP and ROMP have several drawbacks. We therefore suggest a new selection step we call stagewise weak selection. The motivation for this is twofold. We show, using simulation studies, that the proposed stagewise weak selection is preferable in many respects to the other two methods. Furthermore, the stagewise weak selection strategy allows us to extend many theoretical results derived for MP and OMP to our new algorithms.

Instead of using the norm of the residual to define a threshold for element selection as done in StOMP, we here propose the use of a threshold based on the maximum of $|g_i|$. This idea is inspired by the *Weak Matching Pursuit* algorithm [33]. Weak Matching Pursuit is a method developed for large or infinite dimensional problems in which not all inner products can be evaluated explicitly. To accomplish this, a weakness parameter $\alpha \in (0, 1]$ is introduced into the selection criterion. Weak Matching Pursuit selects *any one* element such that

$$|g_j| \geq \alpha \max_i |g_i|.$$

Instead of selecting a *single* element satisfying the above condition, we propose to select *all* element that satisfy this condition. This selection strategy will be call *stagewise weak* selection. Using this approach, the set of indices is updated as

$$\Gamma^{[n]} = \Gamma^{[n-1]} \bigcup \{i : |g_i| \geq \alpha \max_j |g_j|\}, \tag{2}$$

that is, we select *all* elements that come within a factor of α of the largest inner product (in magnitude).

B. Stagewise Weak vs. StOMP selection

Let us briefly consider the relationship between the StOMP selection and the proposed stagewise weak selection. Both algorithms select atoms by applying thresholding to $|\mathbf{\Phi}^T\mathbf{r}^{n-1}|$. For the stagewise weak selection we have:

$$\lambda_{wss} = \alpha||\mathbf{\Phi}^T\mathbf{r}^{n-1}||_\infty.$$

Using norm inequalities we can see that:

$$\frac{\alpha\sqrt{M}}{t\sqrt{N}}\sigma_M(\mathbf{\Phi})\lambda_{stomp} \leq \lambda_{wss} \leq \frac{\alpha}{t}\sqrt{M}\sigma_1(\mathbf{\Phi})\lambda_{stomp} \tag{3}$$

where $\sigma_k(\mathbf{\Phi})$ denotes the k^{th} singular value of $\mathbf{\Phi}$. We thus see that the two thresholds are similar, however the key difference lies in the fact that the stagewise weak threshold is a function of the correlation between the atoms and the residual rather than only a function of the residual. This allows us to extend OMP recovery results to stagewise weak algorithms.

C. Experimental Evaluation of the Stagewise Weak Selection Strategy

We here study the performance of our new Stagewise Weak selection strategy. To do this, we combine the new selection step with the coefficient update of OMP. We will call this combination Stagewise Weak OMP (SWOMP). An important property of this strategy worth stressing is that by changing α, SWOMP interpolates between two well known methods for sparse approximation. A thresholding algorithm is obtained

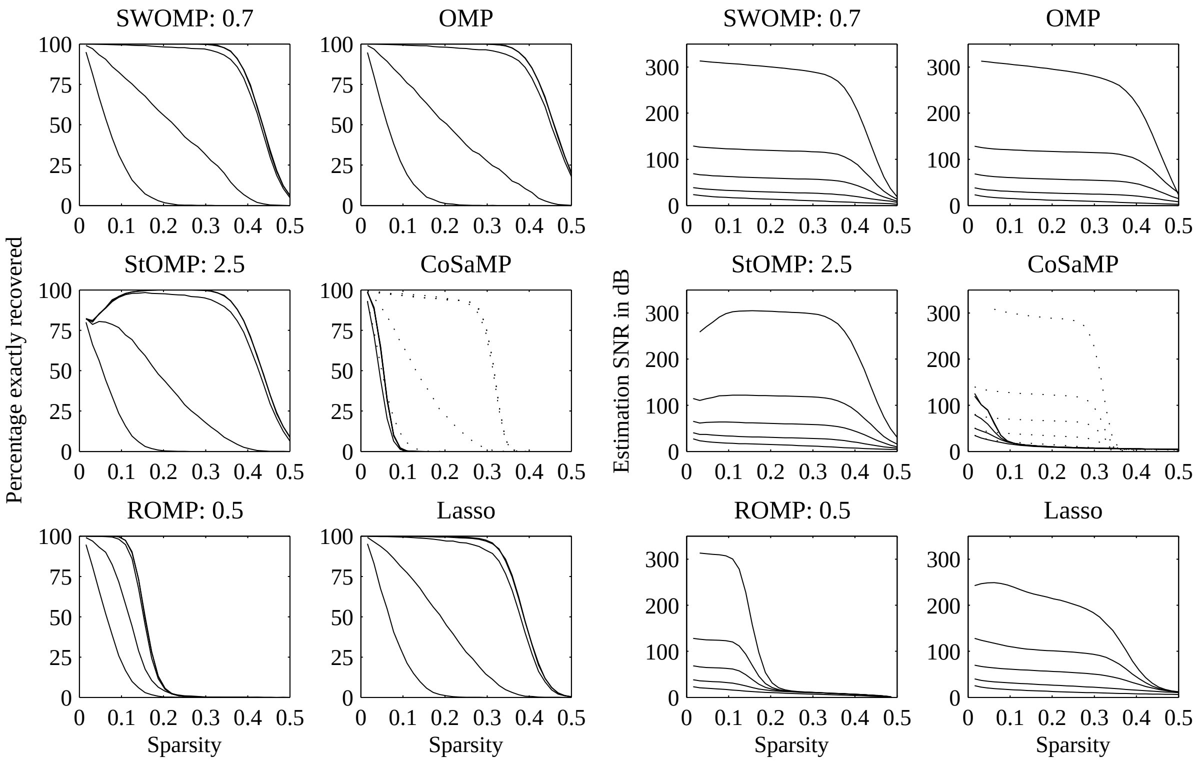


Figure 1: Normal distributed non-zero coefficients. Exact recovery performance (left) and average SNR of recovered coefficients (right) for SWOMP, StOMP, ROMP, OMP, CoSaMP (with 3 conjugate gradient steps (solid) and full projection (dotted)) and Lasso for different observation SNR values (from top right to bottom left in performance SNR = 120dB, 60dB, 30dB, 15dB (Note, the 120dB result are the same as those found with no noise, i.e. ∞dB.)). The abscissa shows the ratio between non-zero elements K and the observation dimension M. All results averaged over 10 000 realisations.

with small α, whilst for $\alpha = 1$ SWOMP becomes standard OMP.

In the first set of experiments we compare different greedy element selection strategies. In particular, we compare OMP, SWOMP ($\alpha = 0.7$), ROMP ($r = 0.5$)[4] and StOMP ($t = 2.5$). These algorithms were run until they had selected *twice* as many elements as were used to generate the observations[5]. For comparison, we also used the CoSaMP algorithm of [27] and optimised the LASSO cost function $\|\mathbf{y}\|_1$ under the constraint that $\|\mathbf{x} - \mathbf{\Phi y}\|_2 \leq \epsilon$ using the homotopy based algorithm of [9] as implemented in the SparseLab[6] toolbox (available at http://sparselab.stanford.edu/). For CoSaMP, two implementations have been suggested in [27], the first of which uses an orthogonal projection in each iteration, whilst the other method replaces the projection by three conjugate gradient steps. We here tried both approaches. The performance is significantly better when using the exact projection, however, calculating the projection is very costly and only feasible for relatively small problems.

Each problem instance was generated by drawing the columns of $\mathbf{\Phi} \in \mathbb{R}^{128\times256}$ uniformly from the unit sphere and drawing the first K elements of $\mathbf{y}$ from an i.i.d. normal distribution. The observations were generated as $\mathbf{x} = \mathbf{\Phi y} + \mathbf{n}$ where $\mathbf{n}$ was i.i.d. Gaussian noise. We varied the number of non-zero elements K and the signal to noise ratio (SNR) (∞dB, 120dB, 60dB, 30dB, and 15dB). All results were averaged over 10 000 problem realisations.

In figure 1 we compare the performance in exactly identifying the support of $\mathbf{y}$ (left) and in terms of signal to noise ratio (SNR) of the estimate $\hat{\mathbf{y}}$ (right). The abscissa shows the ratio between the number of non-zero elements K used to generate the signal and the observation dimension M.

We classified the coefficients to be exactly recovered whenever the estimate was K-sparse and when the K non-zero elements of the estimate were at the same locations as in the original K-sparse vector. As most algorithms selected more than K non-zero elements, we used a post-processing step that pruned out all but the largest (in magnitude) K elements of $\hat{\mathbf{y}}$ once the algorithms had terminated. Due to the selection criterion in StOMP, this algorithm sometimes terminated before it had selected K elements. Similarly, the Lasso algorithm also on occasions selected less than K elements. In these cases,

[4]Note that, in all of our experiments, we used the same selection strategy for ROMP used in the code provided by the authors of [31] In line with the theory developed in [31], this selection strategy only considers disjoint subsets for elements selection and not all subsets. This is faster but reduces the empirical performance of the method somewhat.

[5]Note that in the noiseless case and for random $\mathbf{\Phi}$, if there is a K-sparse vector $\mathbf{y}$, such that $\mathbf{x} = \mathbf{\Phi y}$ and if $K/M < 0.5$, this vector will be unique almost surely as any other vector $\mathbf{y}$ s.t. $\mathbf{x} = \mathbf{\Phi y}$ will have more than $2K$ non-zero elements. Therefore, any $2K$-sparse vector that satisfies $\mathbf{x} = \mathbf{\Phi y}$ has to be equal to the unique K-sparse vector [31].

[6]Note also that the Lasso solution can also be found with the algorithm in [13], which in our experience is faster than the SparseLab implementation.

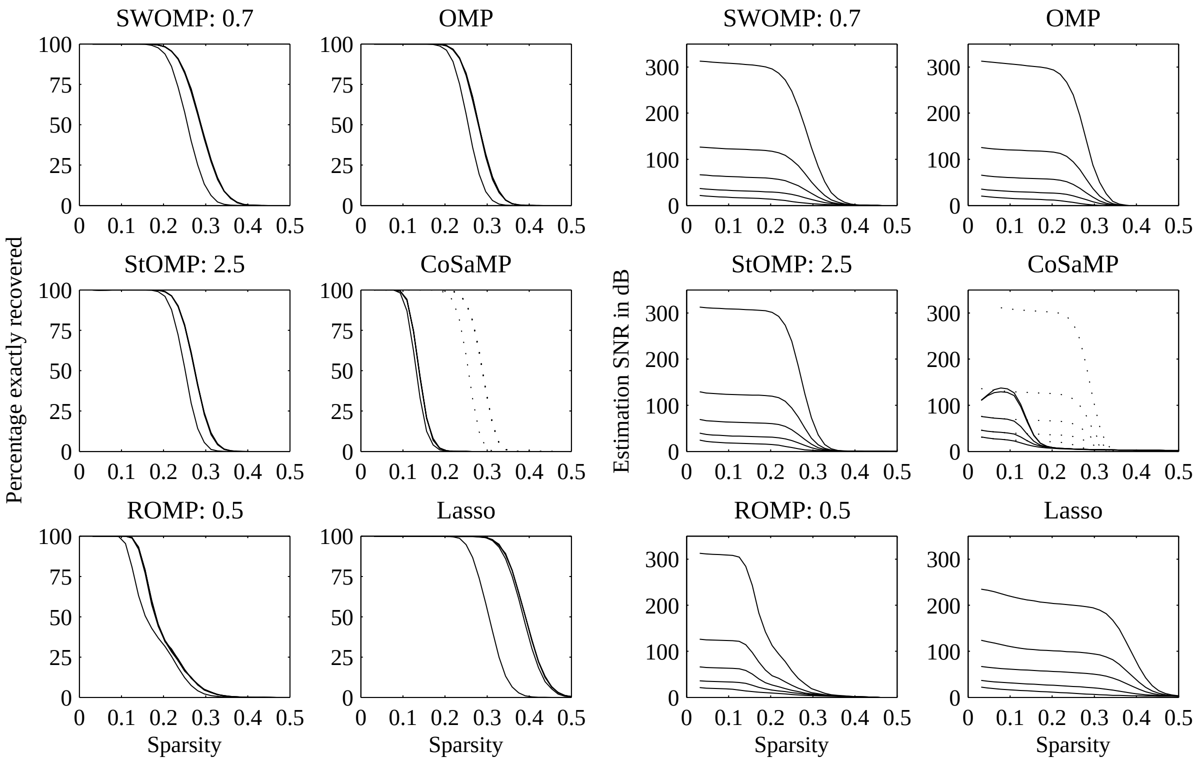


Figure 2: Bernoulli distributed non-zero coefficients. Exact recovery performance (left) and average SNR of recovered coefficients (right) for SWOMP, StOMP, ROMP, OMP, CoSaMP (with 3 conjugate gradient steps (solid) and full projection (dotted)) and Lasso for different observation SNR values (the top right curves in the left panels overlap and are the results found for a SNR of ∞dB, 120dB, 60dB and 30dB, whilst the bottom left curve is the result for 15dB). The abscissa shows the ratio between non-zero elements K and the observation dimension M. All results averaged over 10 000 realisations.

we classified the results as not exactly recovered.

It should be noted that the greedy algorithms require an estimate of the expected number of significant non-zero elements, while the Lasso requires an estimate of the noise. However, many of the greedy algorithms could alternatively be stopped depending on the error between $\mathbf{x}$ and $\mathbf{x}^{[n]}$ relative to the expected noise level. One of the reasons for here allowing the greedy algorithms to select twice as many non-zero elements as were used to generate the signal (followed by pruning to determine whether elements were exactly classified) was to show that it is often beneficial to let these algorithm select more elements than expected, because wrongly selected elements can often be pruned out later, either based on retaining a fixed number of non-zero elements as done here, or, alternatively, by only retaining element above some threshold. Using this, it can be seen that OMP outperforms the Lasso algorithm, at least for normally distributed non-zero coefficients (see however the results below for Bernoulli coefficients). More importantly, the SWOMP algorithm, with a weakness factor of 0.7, also performs better than Lasso. Whilst the theoretical results for greedy strategies are typically worse than those for ℓ_1 based approaches like Lasso, these results suggest that there might be better results possible for greedy strategies at least on average. However, it seems to help to allow the algorithm to select several incorrect elements as done here, which are later pruned out. When running the greedy algorithms until they had selected as many elements as there were non-zero elements, the performance was somewhat worse than those shown here, however, OMP was still comparable in performance to the Lasso method.

The CoSaMP implementations with exact projection (dotted lines) and with three steps of conjugate gradient optimisation (solid line) have similar theoretical performance guarantees [27], however, in the regime in which the theory does not hold[7], CoSaMP seems to break down much faster if the fast implementation is used.

We repeated the experiment using non-zero coefficients that were either -1 or 1 with equal probability (referred to as Bernoulli coefficients from now on). The results are shown in figure 2. It is known that the ℓ_1 methods are insensitive to different distributions of the non-zero coefficients, whilst greedy approaches such as OMP typically perform worse if the non-zero coefficients are all of similar magnitude. This can also be observed here. For Bernoulli coefficients, the recovery performance of all approaches is much more robust against noise, so much in fact that the curves for SNR values of ∞dB, 120dB, 60dB and 30dB basically lie on top of each other, whilst only the curve for an observation SNR of 15dB is markedly different.

[7]For the experiment shown here, it can be shown numerically, that the theoretic conditions used in the theory typically break down for signals with as few as 4 non-zero elements!

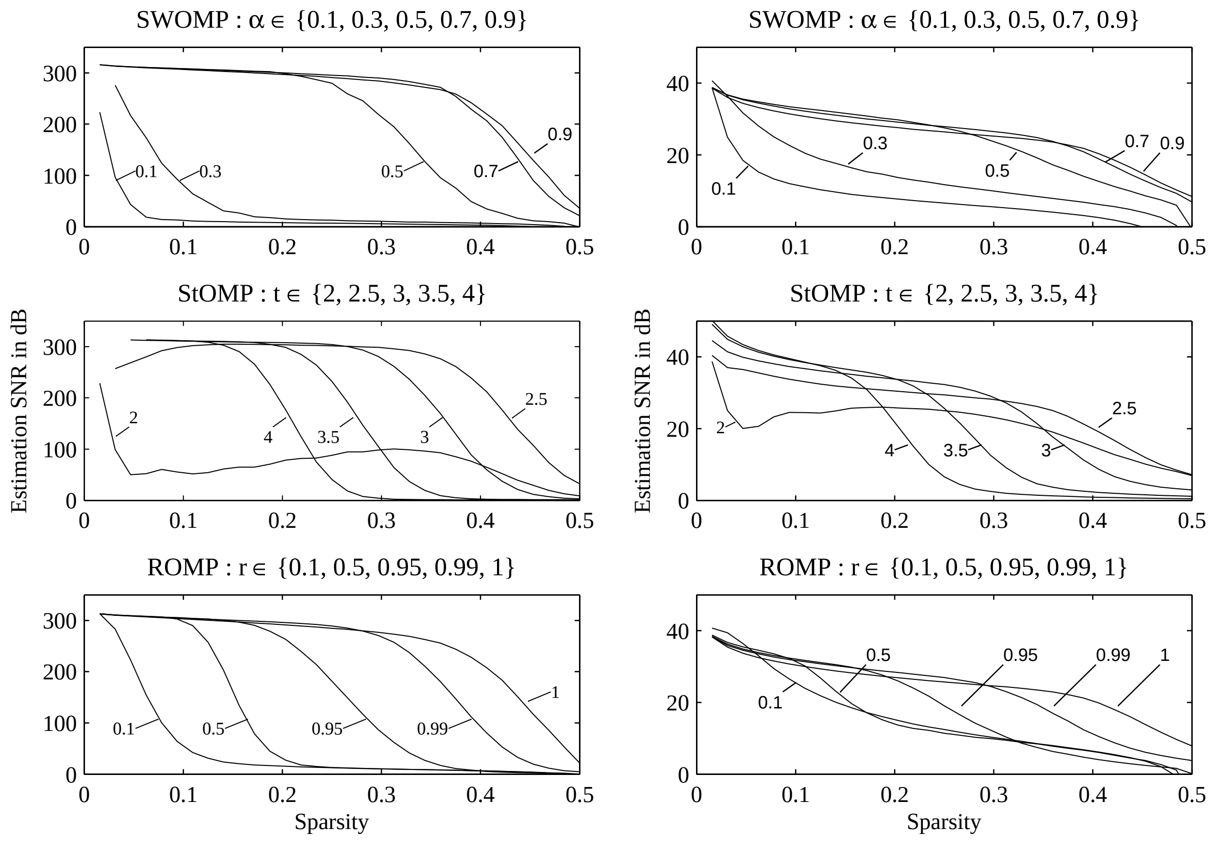


Figure 3: Normal distributed non-zero coefficients without observation noise (left) and with 30dB SNR (right). Comparison between SWOMP, StOMP, ROMP for different parameters. The parameters used are shown above each panel (for SWOMP, α = 0.1, 0.3, 0.5, 0.7 and 0,9, for StOMP, t = 2, 2.5, 3, 3.5 and 4 and for ROMP, r = 0.1, 0.5, 0.9, 0.99 and 1). The abscissa shows the average Signal to Noise ratio of the recovered signals. All results averaged over 10 000 realisations.

In a second set of experiments we evaluated the influence of the parameters t, r and α on the performance of the fast selection strategies. The experimental setup was here the same as in the first experiment above (using no noise and a noise level of 30dB SNR). The results (average SNR), are shown in figures 3 for the noiseless experiment (left) and for the noisy setting with 30dB SNR (right). The parameters are shown above each of the panels as well as next to their corresponding curves. For ROMP and SWOMP, a decrease in the parameter leads to a decrease in performance whilst this is not true for StOMP. Here, a parameter of 2.5 works better than larger or smaller values in general. However, very sparse signals are often not recovered with this parameter as the algorithm often stopped before it had selected K elements.

Whilst StOMP and SWOMP can both perform well, the influence of the parameter t in StOMP is more complicated than the smooth decay in performance observed for SWOMP. ROMP, even though it has certain nice theoretic properties when $r = 0.5$, does show significantly worse average performance. When increasing r and α to one, both ROMP and SWOMP are effectively OMP, it is therefore not surprising that they both have comparable performance in this limit. However, the number of iterations both methods used when, for example $\alpha = 0.5$ and $r = 0.99$ (notice the performance of these methods is similar in this case) were quite different. In this case for a sparseness $K/M = 0.2$, ROMP used on average nearly four times as many iterations as SWOMP.

The above results demonstrate several benefits of the *Stagewise Weak* selection strategy. On the one hand, we can often choose a quite small value for α without significantly sacrificing performance. For example, with $\alpha = 0.7$ the SWOMP algorithm performs similar to the OMP algorithm, but uses fewer iterations. Comparing SWOMP, ROMP and StOMP, we see that the decrease in performance of SWOMP with decreasing α is much more controlled than for the other two methods, making the choice of the parameter somewhat simpler. Finally, comparing SWOMP and ROMP, we see that, if we adjust the parameters such that both methods show similar performance, SWOMP uses significantly fewer iterations.

IV. GRADIENT PURSUIT

Our new *Stagewise Weak* selection strategy can significantly reduce the number of iterations required. We therefore now turn to the problem of reducing the computational cost of each iteration. Focussing on the coefficient update step, we first discuss current approaches and review the gradient pursuit update in somewhat more detail. For this update, we derive two new results, firstly, a new recursion allows this method to be implemented more efficiently and secondly, a new convergence theorem is stated.

Both OMP and ROMP update the coefficients $\hat{\mathbf{y}}^{[n]}$ by searching for the minimiser of $\|\mathbf{x} - \mathbf{\Phi}_{\Gamma^{[n]}}\hat{\mathbf{y}}_{\Gamma^{[n]}}\|_2^2$. This is done using an orthogonal projection, that is, by calculateing $\hat{\mathbf{y}}^{[n]}_{\Gamma^{[n]}} = \mathbf{\Phi}^{\dagger}_{\Gamma^{[n]}}\mathbf{x}$ where $\mathbf{\Phi}^{\dagger}_{\Gamma^{[n]}}$ is the pseudo inverse[8] of $\mathbf{\Phi}_{\Gamma^{[n]}}$.

However, for applications in which $\mathbf{\Phi}$ is large, two problems arise. Firstly, storage of $\mathbf{\Phi}$ can be problematic. Secondly, direct matrix vector products involving $\mathbf{\Phi}$ or its adjoint are costly. Therefore, in many applications, $\mathbf{\Phi}$ is designed with additional structure such that $\mathbf{\Phi}$ does not have to be stored explicitly and that matrix vector products involving $\mathbf{\Phi}$ and its adjoint can be evaluated more efficiently. For example, if the fast Fourier transform is used, the computation time can often be reduced to be $O(N\log M)$ instead of the $O(MN)$ for unstructured matrices. In this regime, calculating $\hat{\mathbf{y}}^{[n]}$ as done in OMP and ROMP often becomes infeasible.

One approach to utilise this structure and to overcome computational requirements associated with the exact calculation of the orthogonal projection is that used in StOMP [30], where instead of calculating $\hat{\mathbf{y}}^{[n]}$ as in OMP or ROMP, $\hat{\mathbf{y}}^{[n]}$ is approximated using *a few* iterations of a conjugate gradient solver.

Instead of using the approach used in StOMP, which starts a new conjugate gradient solver in each iteration, we have previously argued to approximate the orthogonal projection of OMP using (a single) directional optimisation step, which can be done much more efficiently [29]. This led to the Gradient Pursuit family of algorithms which uses directional optimisation to update $\hat{\mathbf{y}}^{[n-1]}$ in each iteration. In particular

$$\hat{\mathbf{y}}^{[n]} = \hat{\mathbf{y}}^{[n-1]} + \beta\mathbf{d}^{[n]},$$

where, as shown in [34, pp. 521], the optimum step size is

$$\beta = \frac{\langle \mathbf{r}^n, \mathbf{\Phi}_{\Gamma^{[n]}}\mathbf{d}^n\rangle}{\|\mathbf{\Phi}_{\Gamma^{[n]}}\mathbf{d}^n\|_2^2}. \tag{4}$$

Whilst the updates in MP and OMP also fall into this framework, different directions $\mathbf{d}^{[n]}$ might be beneficial. In [29], the gradient and an (approximate)[9] conjugate gradient method were suggested. The update directions for the approximate conjugate gradient method is calculated using

$$\mathbf{d}^{[n]}_{\Gamma^{[n]}} = \mathbf{g}^{[n]}_{\Gamma^{[n]}} + \upsilon\mathbf{d}^{[n-1]}_{\Gamma^{[n]}}, \tag{5}$$

where $\upsilon = -\big\langle(\mathbf{\Phi}_{\Gamma^n}\mathbf{d}^{n-1}_{\Gamma^n}), (\mathbf{\Phi}_{\Gamma^n}\mathbf{g}^{n}_{\Gamma^n})\big\rangle/\|\mathbf{\Phi}_{\Gamma^n}\mathbf{d}^{n-1}_{\Gamma^n}\|_2^2$ ensures that that $\langle\mathbf{\Phi}_{\Gamma^{[n]}}\mathbf{d}^{[n]}_{\Gamma^{[n]}}, \mathbf{\Phi}_{\Gamma^{[n]}}\mathbf{d}^{[n-1]}_{\Gamma^{[n]}}\rangle = 0$, that is, consecutive update directions are *conjugate*. It is important to note that this strategy uses a single update direction after each element selection step. This update direction is chosen to be conjugate to the update step in the previous iteration. This is different from using a full conjugate gradient solver after each new element selection. See [29] for a more detailed discussion.

In the next subsection we derive an additional recursion that allows the approximate conjugate gradient to be calculated with the same computational complexity as the gradient. What is more, an approximate conjugate gradient step is guaranteed to reduce the squared error more than a gradient step. We therefore here endorse the approximate conjugate gradient approach, which for simplicity will be called Conjugate Gradient Pursuit (CGP) throughout this paper.

A. Implementation

We here propose a new and efficient way to calculate the conjugate gradient update, which is based on a novel recursion to calculate the update direction and uses auxiliary vectors

$$\mathbf{v}^{[n]} = \mathbf{\Phi}_{\Gamma^{[n]}}\mathbf{d}^{[n]}_{\Gamma^{[n]}} \tag{6}$$

$$\mathbf{w}^{[n]} = \mathbf{\Phi}_{\Gamma^{[n]}}\mathbf{g}^{[n]}_{\Gamma^{[n]}}. \tag{7}$$

Because $\mathbf{d}^{[n]}_{\Gamma^{[n]}} = \mathbf{g}^{[n]}_{\Gamma^{[n]}} + \upsilon^{[n]}\mathbf{d}^{[n-1]}_{\Gamma^{[n]}}$, using $\mathbf{w}^{[n]}$, $\mathbf{v}^{[n]}$ can be calculated recursively $\mathbf{v}^{[n]} = \mathbf{w}^{[n]} + \upsilon^{[n]}\mathbf{v}^{[n-1]}$. Furthermore, using $\eta^{[n]} = \|\mathbf{v}^{[n]}\|_2^2$, calculation of $\upsilon^{[n]}$ and $\beta^{[n]}$ can now be done efficiently $\upsilon^{[n]} = -\langle\mathbf{v}^{[n-1]}, \mathbf{w}^{[n]}\rangle/\eta^{[n-1]}$ and $\beta^{[n]} = \langle\mathbf{r}^{[n-1]}, \mathbf{v}^{[n]}\rangle/\eta^{[n]}$.

In summary, the algorithm is

```
• Input: $\mathbf{x}, \mathbf{\Phi}$ and stopping criterion
• Initialise: $\hat{\mathbf{y}}^{[0]} = \mathbf{0}, \Gamma^{[0]} = \emptyset, \mathbf{r}^{[0]} = \mathbf{x}, n = 1$
• iterate until stopping criterion is met:
    1) $\mathbf{g}^{[n]} = \mathbf{\Phi}^T\mathbf{r}^{[n-1]}$
    2) Select a set of new elements $\mathcal{I}$.
    3) $\Gamma^{[n]} = \Gamma^{[n-1]}\bigcup\mathcal{I}$
    4) if $n = 1$
        ◇ $\mathbf{d}^{[n]}_{\Gamma^{[n]}} = \mathbf{g}^{[n]}_{\Gamma^{[n]}}$
        ◇ $\mathbf{v}^{[n]} = \mathbf{\Phi}_{\Gamma^{[n]}}\mathbf{d}^{[n]}_{\Gamma^{[n]}}$
       else,
        ◇ $\mathbf{w}^{[n]} = \mathbf{\Phi}_{\Gamma^{[n]}}\mathbf{g}^{[n]}_{\Gamma^{[n]}}$
        ◇ $\upsilon^{[n]} = -\langle\mathbf{v}^{[n-1]}, \mathbf{w}^{[n]}\rangle/\eta^{[n-1]}$
        ◇ $\mathbf{d}^{[n]}_{\Gamma^{[n]}} = \mathbf{g}^{[n]}_{\Gamma^{[n]}} + \upsilon^{[n]}\mathbf{d}^{[n-1]}_{\Gamma^{[n]}}$
        ◇ $\mathbf{v}^{[n]} = \mathbf{w}^{[n]} + \upsilon^{[n]}\mathbf{v}^{[n-1]}$
    5) $\eta^{[n]} = \|\mathbf{v}^{[n]}\|_2^2$
    6) $\beta^{[n]} = \langle\mathbf{r}^{[n-1]}, \mathbf{v}^{[n]}\rangle/\eta^{[n]}$
    7) $\hat{\mathbf{y}}^{[n]}_{\Gamma^{[n]}} = \hat{\mathbf{y}}^{[n-1]}_{\Gamma^{[n]}} + \beta^{[n]}\mathbf{d}^{[n]}_{\Gamma^{[n]}}$
    8) $\mathbf{r}^{[n]} = \mathbf{r}^{[n-1]} - \beta\mathbf{v}^{[n]}$
    9) $n \mapsto n + 1$
• Output $\mathbf{r}^{[n-1]}, \Gamma^{[n-1]}$ and $\hat{\mathbf{y}}^{[n-1]}$
```

Different stopping criteria can be used. For example, the algorithm can be stopped whenever a desired number of non-zero elements has been found or whenever the norm of the error $\mathbf{r}^{[n]}$ decreases below some threshold. Note also that we have here deliberately left the element selection step ambiguous. In [29] we used the same strategy as in MP and OMP, however, the Stagewise Weak selection strategy of the previous section can be used instead.

[8]In an efficient implementation, the pseudo inverse is in general not calculated explicitly in each iteration. Instead, fast implementations of OMP either keep track of a QR factorisation of $\mathbf{\Phi}_{\Gamma^{[n]}}$, which is updated efficiently in each iteration or, alternatively, keep track of a Cholesky factorisation of the Gram matrix $\mathbf{G}^{[n]}_{\Gamma^{[n]}} = \mathbf{\Phi}^T_{\Gamma^{[n]}}\mathbf{\Phi}_{\Gamma^{[n]}}$ which is also updated from iteration to iteration. More details on these methods can be found in, for example, [29].

[9]Approximate, because the method only guarantees conjugacy of the current update direction to the previous update direction, but not to all previous update directions [29].

B. Computation Cost per Iteration

An important property of the algorithm as outlined in Subsection IV-A is that it only requires the storage of vectors and scalars. The only exception is the required storage of the mapping $\mathbf{\Phi}$. The storage requirements are therefore low.

The computational complexity is also low. The bottleneck is the application of $\mathbf{\Phi}$ and its adjoint, which are only applied twice in each iteration. Due to the new recursion, the computational requirements, which we summarise in table I, are better than those reported in [29].

C. Convergence

Although the gradient based directional updates do not fully minimise the residual, it can be shown that under certain circumstances a single optimization step actually does a pretty good job. In [29] we have shown that a Gradient Pursuit algorithm with update $\mathbf{d}^{[n]}_{\Gamma^{[n]}} = \mathbf{g}^{[n]}_{\Gamma^{[n]}}$ converges linearly. In particular we had

Theorem 1: There exists a constant $\omega < 1$, which only depends on $\mathbf{\Phi}$, such that the residual calculated with the gradient based Gradient Pursuit algorithm decays as

$$\|\mathbf{r}^{[n]}\|_2^2 \le \omega \|\mathbf{r}^{[n-1]}\|_2^2.$$

The constant ω can be expressed in terms of the following quantities of interest in the theoretical study of sparse signal recovery. The restricted isometry constant δ_K [35] is a symmetric bound on the singular values of any sub-matrix of $\mathbf{\Phi}$ with K (or less) elements and is defined as the smallest quantity such that

$$(1 - \delta_K(\mathbf{\Phi})) \le \frac{\|\mathbf{\Phi}\mathbf{y}\|_2^2}{\|\mathbf{y}\|_2^2} \le (1 + \delta_K(\mathbf{\Phi}))$$

holds for all $\mathbf{y}$ with no more than K non-zero elements.

Using δ_K to bound the denominator on the right site of Equation (29) in [29], it can be shown that

$$\omega \le \left(1 - \frac{\rho}{1+\delta_K}\right), \tag{8}$$

where $\rho > 0$ is such that $\|\mathbf{\Phi}^T\mathbf{x}\|_\infty^2 > \rho\|\mathbf{x}\|_2^2$, for all $\mathbf{x}$ [33, pp. 422].

An argument for the use of gradient based optimisation can also be given based on the restricted isometry. If the dictionary has a small restricted isometry constant δ_K, then every subdictionary is very nearly orthogonal. Because eigenvalues of sub-matrices are nested between those of the full matrix [36, Theorem 7.3.9], for $n \le K$, the condition number, κ, of the sub-dictionary's Gram matrix, $\mathbf{G}^{[n]}_\Gamma$ is bounded by

$$\kappa(\mathbf{G}^{[n]}_\Gamma) \le \left(\frac{1+\delta_K}{1-\delta_K}\right)^2.$$

This can be used to explain the good performance of the gradient based updates. Using

$$f(\mathbf{y}_{\Gamma^n}) = \|\mathbf{x} - \mathbf{\Phi}_{\Gamma^n}\mathbf{y}_{\Gamma^n}\|_2^2,$$

a worst case analysis of the gradient line search gives [37]:

$$\begin{aligned}\frac{f(\hat{\mathbf{y}}^{[n]}_{\Gamma^n}) - f(\mathbf{y}^*_{\Gamma^n})}{f(\hat{\mathbf{y}}^{[n-1]}_{\Gamma^n}) - f(\mathbf{y}^*_{\Gamma^n})} &\le \left(\frac{\kappa-1}{\kappa+1}\right)^2 \\ &\le \left(\frac{2\delta_K}{(1-\delta_K)^2}\right)^2\end{aligned} \tag{9}$$

where $\mathbf{y}^*_{\Gamma^n}$ denotes the least squares solution of $f(\mathbf{y}_{\Gamma^n})$. Hence for small δ_K the convergence, even of a single gradient iteration, is good.

The convergence of the CGP algorithm discussed above was not derived in [29]. The following theorem shows that the reduction in $f(\mathbf{y}_{\Gamma^n})$ is at least as good when using update (5) than when using update $\mathbf{d}^{[n]}_{\Gamma^{[n]}} = \mathbf{g}^{[n]}_{\Gamma^{[n]}}$.

Theorem 2: Use ω and β as defined in Subsection IV-A. In the Gradient Pursuit framework, using the update direction (5) reduces the ℓ_2 norm of the residual $\mathbf{r}^{[n]} = \mathbf{x} - \mathbf{\Phi}\hat{\mathbf{y}}^{[n]} = \mathbf{x} - \mathbf{\Phi}(\hat{\mathbf{y}}^{[n-1]} + \beta\mathbf{d}^{[n]})$ at least as much as using the update $\mathbf{d}^{[n]}_{\Gamma^{[n]}} = \mathbf{g}^{[n]}_{\Gamma^{[n]}}$. Therefore, the convergence for the algorithm using the direction defined in (5) is at least as good as that in theorem 1.
The proof is basically that of [38], but care has to be taken to take account of the fact that the Gram matrix changes from iteration to iteration.

V. STAGEWISE WEAK CONJUGATE GRADIENT PURSUIT: FASTER AND GREEDIER

In this section we combine the developments of the previous two sections and join the stagewise weak selection with the conjugate gradient update. Whilst it might seem intuitive that SWOMP will be faster than OMP as it uses fewer iterations, this is not necessarily true. For example, in the fast implementations of OMP based on QR or Cholesky factorisation, the QR or Cholesky factorisations have to be updated for *each* of the newly selected elements. Overall, there will be as many of these updates as there are elements to be selected. As the updates dominate the computation cost, the computational advantage of using stagewise selection strategies with OMP are therefore small.

Instead, StOMP [30] used a small number of conjugate gradient steps in each iteration to approximate the required orthogonalisation. We promote the use of the CGP algorithm to do the required approximate orthogonalisation. The selection step in the CGP algorithm is replaced by the stagewise weak selection step in (2) such that the overall number of iterations is potentially reduced significantly, while the computational complexity of each iterations remains the same as that of the standard CGP method. This approach will be referred to as a Stagewise Weak Conjugate Gradient algorithm (SWCGP). The weak selection strategy has now been incorporated into the implementation of CGP in the sparsify matlab toolbox to be found on the first authors web-page. The algorithm is accessible through the call to function greed_nomp (nomp for Nearly Orthogonal Matching Pursuit).

A. SWCGP vs OMP and MP

The next step is to evaluate the influence of replacing the exact orthogonalisation with the conjugate gradient update in

I: Computation and storage cost of CGP in iteration $n \geq 2$.

computation	storage
2 applications of $\mathbf{\Phi}$ (once using the adjoint)	$\mathbf{\Phi}$
3 inner products (M-vectors)	1 N-vector
2 M-vector additions (with scalar multiplication)	4 M-vectors
2 K-vector additions (with scalar multiplication)	3 K-vectors

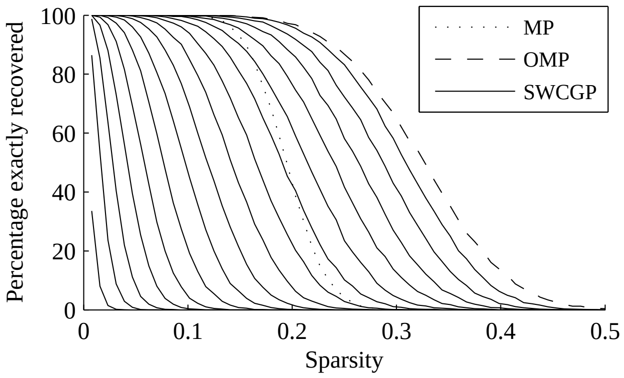


Figure 4: Comparison between Matching Pursuit (dotted), Orthogonal Matching Pursuit (dashed) and Stagewise Conjugate Gradient Pursuit (solid) in terms of exactly recovering the original coefficients. The ordinate shows the fraction of runs in which the algorithms exactly recovered the index set Γ used to generate the data while the abscissa shows the ratio of the size of Γ to the dimension of $\mathbf{x}$. Results averaged over 10 000 runs. The solid lines correspond to (from left to right): $\alpha = 0.25$ to 1.0 in steps of 0.05.

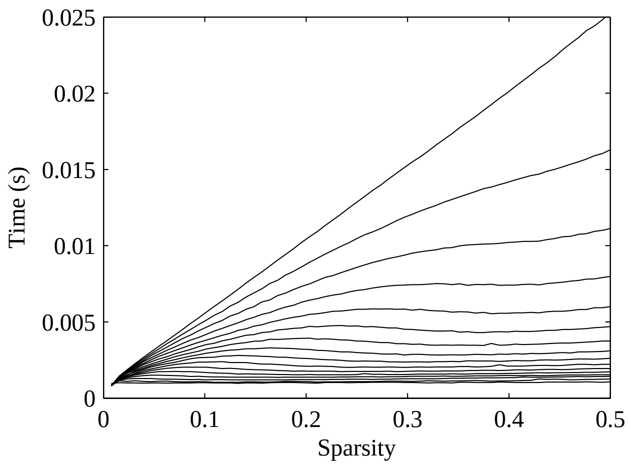


Figure 5: Comparison of the computation time for SWCGP with the different values of alpha as in figure 4. The curves correspond to (going from top to bottom): $\alpha = 1$ to 0.25 in steps of 0.05.

(5) and to evaluate the influence of varying α. We therefore repeated the experiment of Section III (using the noiseless setting) using the SWCGP algorithm. We here only run the method until it had selected K elements. Figure 4 studies the influence of the weakness parameter α. For $\alpha = 1$, the method is equivalent (up to ties) to CGP. For comparison, also shown are the results obtained with OMP and MP.

It is clear that weakening the selection criterion reduces (in a controlled manner) the recovery performance. The advantage of this is a reduction in computational cost. This is shown in figure 5. Here the curves correspond to (from top to bottom) α decreasing from 1 in steps of 0.05. The top curve indicates that the computational cost for CGP (SWCGP with $\alpha = 1.0$) grows linearly with the number of non-zero coefficients. In contrast for $\alpha < 1.0$ the computational cost grows much more slowly. It should be noted here that these figures do not fully capture the performance of SWCGP since the dictionaries used do not have a fast implementation. However they do provide a fair relative comparison between different values of α.

B. Medical Imaging example

This section demonstrates the applicability of the proposed SWCGP algorithm to large sparse inverse problems. We here study a Compressed sensing [5] problem. Compressed sensing is a recent development based on sparse signal modelling ideas. One particularly promising application domain of this technique is Magnetic Resonance Imaging (MRI) [39] and we take our next example from this area using the Shepp-Logan phantom.

Acquiring MRI images is equivalent to taking one dimensional slices from the 2-dimension Fourier domain of the image. For rapid MR imaging it is desirable to take only a subset of these slices. For example one could take a reduced number of radial lines of the Fourier domain data. In order to reconstruct the original image, we utilize the fact that the image has a sparse representation in the Haar wavelet transform.[10] For this particular image of size 256×256, it was observed that the original image is well approximated (over 300 dB peak signal to noise ratio) using only 4000 of the wavelet coefficients.

In [29] the performance of Gradient Pursuit, OMP and various L_1 methods were reported for this problem. Here we examine the speed and performance of SWCGP for α between 0.5 and 1.0, to reconstruct the image from 15% of the Fourier data. The results are presented in table II. In each case the algorithm was stopped once at least 4000 atoms were selected.[11] Notice that for this data it is possible to obtain an approximate speed up of 80 times using the stagewise algorithm instead of the stepwise version. Even using a relatively conservative value for α, of 0.9, gave an 8 times reduction is computation time.

These improvements suggest that SWCGP should be a good

[10]It is important to note that we here use a Haar wavelet basis as our sparse representation and not a total variation based constraint as used for example in [5].

[11]These simulations were performed using Matlab running on a 2GHz Pentium PC.

II: Influence of α on: number of iterations; approximate computation time and; PSNR performance (dB).

α	0.5	0.6	0.7	0.8	0.9	1.0
No. of iterations	51	81	214	293	474	4087
computation time (sec.)	19.4	33.3	82	114	182	1562
PSNR (dB)	59	79	311	311	309	301

candidate for tackling very large-scale problems such as those encountered in dynamic MRI imaging.

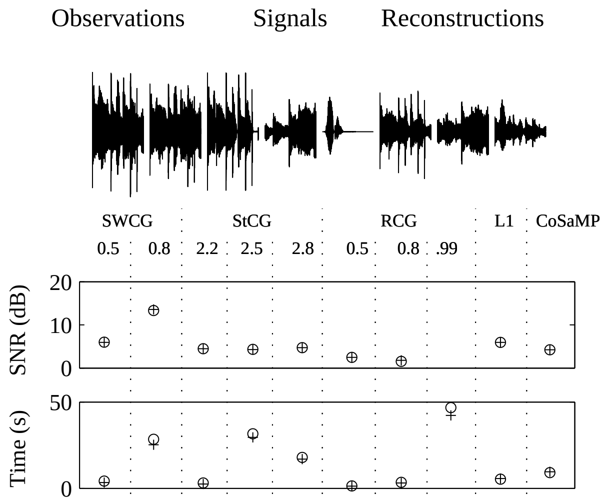
To test the algorithm on an even larger and more realistic problem, we used a subsampled version of a fully sampled MRI image sequence of a beating mouse heart to simulate rapid imaging. The sequence consisted of 8 consecutive 256×256 images of the heart. We here used a 3-dimensional Haar basis as the sparse representation. As in the previous example, measurements were taken using radial lines in the spatial Fourier domain for each image. To add a degree of randomness the orientation of the lines was selected uniformly at random for each image. We used SWCGP with $\alpha = 0.7$ (stopped after $20,000$ atoms were selected). The overall PSNR of the reconstruction was 31.3dB. Furthermore the reconstruction took 50 minutes (44 iterations), which is a speed up of approximately 450 times (based on iteration count) compared with the stepwise algorithm!

C. Evaluation on different signal processing problems

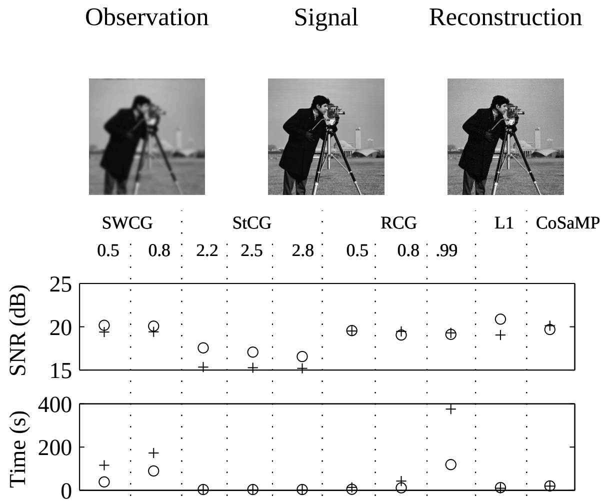
In order to compare and showcase the performance of our proposed stagewise weak method on a set of different problems often addressed with sparse approximation techniques, we have chosen three different problems from the SPARCO matlab toolbox (available at http://www.cs.ubc.ca/labs/scl/sparco/).

Problem 402 is a source separation problem. Three audio sources are mixed using an instantaneous mixing system to give two observations. To invert the underdetermined mixing system and separate the sources, the original audio is assumed to be sparse in a localised discrete cosine transform basis. The problem size is M=29,166, N=86,016, K=14,583. Problem 701 is an image de-blurring example. The image is assumed to be sparse in the wavelet domain. The problem size is M=65,536, N=65,536, K=9,000. Problem 703 is a missing data problem in which scratches are to be removed from a fingerprint image, which is assumed to be sparse in the 2D curvelet domain.

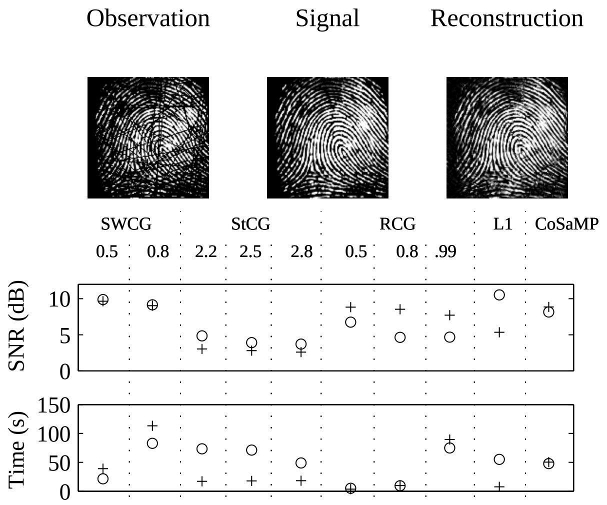
For each problem, the observation (possibly mapped back using a linear projection into the signal space), the signal and the signal estimate calculated with the Stagewise Weak Conjugate Gradient (SW) algorithm with $\alpha = 0.5$ are shown at the top of each panel in figure 6. Below this we show the ratio between the true signal and the error in its estimate (SNR in dB) above the computation time required by the different methods in seconds (all simulations were run in matlab on a Macintosh 2.5Ghz quad G5 computer). We here compare the three different selection strategies Stagewise Weak (SW) (with $\alpha \in \{0.5, 0.8\}$), Stagewise (St) (with $t \in \{2.2, 2.5, 2.8\}$) and regularised (R) (with $r \in \{0.5, 0.8, 0.99\}$). In addition, we also calculated the solution of the optimisation problem $\min_{\mathbf{x}} \|\mathbf{y} - \mathbf{\Phi x}\|_2 + \lambda\|\mathbf{x}\|_1$ (L1) (using the TwIST al-



(a) Sparco problem 402



(b) Sparco problem 701



(c) Sparco problem 703

Figure 6: Observed signal, original signal and reconstruction using SWCGP ($\alpha = 0.5$) above SNR in dB of estimate calculated with different approaches and computation time. With (+) and without (o) normalisation of columns.

gorithm [12] (http://www.lx.it.pt/ bioucas/TwIST/TwIST.htm)) and used CoSaMP. As we are here interested in methods that can be applied to very large problems, we combined the three greedy selection strategies with the fast conjugate gradient putsuit update and used the fast implementation of CoSaMP using three conjugate gradient iterations.

For all algorithms we selected the stopping criterion as well as the regularisation parameter required in L1 optimisation by trial and error until the observed SNR was optimal. Whilst this is not possible in practice, it allows a more or less fair comparison of the methods. All greedy algorithms used the approximate conjugate gradient update step and differed only in the element selection step. The difference in the computation time observed with these methods is therefore purely due to the different number of iterations used.

The matrices $\mathbf{\Phi}$ available in the SPARCO toolbox have columns of different norm. As the algorithms compared here will favour columns of $\mathbf{\Phi}$ that do have a larger ℓ_2 norm, it is in general desirable to design the measurement system with equal norm columns. Otherwise, it is often possible to pre-calculate the norm of the columns of $\mathbf{\Phi}$. However, if this is also not feasible, a possibly sub-optimal approach would ignore the difference in norm. To study the influence of this normalisation, the results shown in figure 6 were therefore calculated with (+) and without (o) normalisation.

Comparing the SNR results for the different greedy strategies, it is evident that SWCGP performs consistently better than the other greedy approaches. L1 optimisation on the other hand can be seen to often rival SWCGP in terms of SNR as well as computation time. The fast version of CoSaMP used here did often not perform as well as the SWCGP approach. SWCGP therefore seems to offer a competitive alternative to ℓ_1 based approaches as well as to CoSaMP and is applicable in a diverse range of settings to solve a large range of signal processing challenges.

VI. Recovery analysis

Several surprising results have been derived over the years that give guarantees on the quality of the solutions to sparse inverse problems calculated with different algorithms. For example, the papers [40], [41], [42], [43], [44], [45], [5] have shown that under certain conditions, solving a convex ℓ_1 problem will simultaneously solve the problem of finding a vector with the minimal number of non-zero entries. Similar guarantees also hold for the methods developed in [26], [27] and [28]. Comparable results have also been derived for OMP [46] and [47].

One important motivation for the development of our new selection strategy was that it allows similar theoretical statements to be derived for stagewise weak algorithms. For example, nearly all results presented in, [46], [47] and [48] also apply to our method, possibly with minor modifications that take account of the stagewise weak selection. For example, we have the following bound on the approximation error, which we state in terms of the restricted isometry constant.

Theorem 3: (SWCGP recovery of general signals) For any observation $\mathbf{x} = \mathbf{\Phi y} + \mathbf{n}$, run the SWCGP algorithm until it has selected K non-zero elements. If

$$\delta_{K+1} < \frac{\alpha}{\sqrt{K}+\alpha}, \tag{10}$$

then the estimate $\hat{\mathbf{x}}^{[n]} = \mathbf{\Phi}\hat{\mathbf{y}}^{[n]}$ satisfies

$$\|\mathbf{x} - \hat{\mathbf{x}}^{[n]}\|_2 \le c\|\mathbf{x} - \hat{\mathbf{x}}^\star\|_2, \tag{11}$$

where

$$c = \sqrt{1 + \frac{1}{\left(\alpha\sqrt{(1-\delta_K)/K} - \delta_{K+1}/\sqrt{1-\delta_K}\right)^2}} \tag{12}$$

and where $\hat{\mathbf{x}}^\star = \mathbf{\Phi}_{\Gamma^\star}\hat{\mathbf{y}}_{\Gamma^\star}$ with $\hat{\mathbf{y}}_{\Gamma^\star} = \mathbf{\Phi}^\dagger_{\Gamma^\star}\mathbf{x}$ and $\Gamma^\star$ is the index set of the largest K elements in $\mathbf{y}$.
The proof is similar to that in [46], but with the difference that we derived the result in terms of the restricted isometry constant δ_{K+1}. The changes to the proof in [46] required for this setting are given in appendix I.

Whilst the above theorem bounds the approximation error $\|\mathbf{x} - \hat{\mathbf{x}}^{[n]}\|_2$, a simple argument (see appendix II) can be used to also bound the estimation error $\|\hat{\mathbf{y}}^{[n]} - \mathbf{y}\|_2$.

Theorem 4: For any $\mathbf{y}$, let $\mathbf{x} = \mathbf{\Phi y} + \mathbf{n}$ and stop the algorithm before it selects more than K non-zero elements. Let the last iteration be iteration $n^\star$ and let $\hat{\mathbf{y}}^{[n^\star]}$ be the estimation of $\mathbf{y}$ calculated at this iteration. If

$$\delta_{K+1} < \frac{\alpha}{\sqrt{K}+\alpha}, \tag{13}$$

then there exist a constant $\hat{c}$ (depending on α and δ_{2K}), such that

$$\|\hat{\mathbf{y}}^{[n^\star]} - \mathbf{y}\|_2 \le \hat{c}\epsilon, \tag{14}$$

where

$$\epsilon = \|(\mathbf{y} - \mathbf{y}_K)\|_2 + \frac{\|(\mathbf{y} - \mathbf{y}_K)\|_1}{\sqrt{K}} + \|\mathbf{n}\|_2 \tag{15}$$

and where $\mathbf{y}_K$ is the best K-term approximation to $\mathbf{y}$. For example, if $\delta_{2K} < \frac{\alpha}{\sqrt{K}+\alpha} \le 0.5$, then $\|\hat{\mathbf{y}}^{[n^\star]} - \mathbf{y}\|_2 \le (3 + 6.5c)\epsilon$, where c is as in Theorem 3.

More generally, if $\delta_{2K} < 1$, then, whenever the algorithm has calculated a K-sparse estimate $\hat{\mathbf{y}}^{[n^\star]}$ with

$$\frac{\|\mathbf{\Phi}\hat{\mathbf{y}}^{[n^\star]}\|_2}{\|\mathbf{x}\|_2} = \Omega, \tag{16}$$

then we are guaranteed that

$$\|\hat{\mathbf{y}}^{[n^\star]} - \mathbf{y}\|_2 \le \frac{\Omega}{\sqrt{1-\delta_{2K}}}\|\mathbf{x}\|_2 + c_2\epsilon, \tag{17}$$

where $c_2 = 1 + \frac{\sqrt{1+\delta_K}}{\sqrt{1-\delta_{2K}}}$.

It is important to note that a dependence on the error ϵ is in-fact optimal up to the constant [28]. The above bound is therefore similar to that derived for other methods such as ℓ_1 based optimisation [49], CoSaMP [27], Subspace Pursuit [26] and Iterative Hard thresholding [28]. However, for the above bound to hold, we require that $K^{0.5}\delta_{2K}$ is small, which implies that we required $M = O(K^2\log(N/K))$, which is typical for OMP type algorithms.

A better result can however be achieved if we take random measurements and only require the algorithm to recover a single signal $\mathbf{y}$ [47]. For example, OMP was shown in [47] to be able to recover the correct support of a K-sparse signal $\mathbf{y}$ with high probability when M is of the order of $K\ln(N)$. The proof in [47] can again be easily adapted to the weak setting discussed here and the following result can be derived.

Theorem 5: (SWCGP with random measurements) Suppose that $\mathbf{y}$ is an arbitrary K-sparse signal in $\mathbb{R}^N$ and draw a random $M\times N$ matrix $\mathbf{\Phi}$ with i.i.d. Gaussian or Bernoulli entries, normalised so that the expected Euclidean column norm is one. Given the data $\mathbf{x}=\mathbf{\Phi}\mathbf{y}$ and choosing $M\geq c\alpha^{-2}K\log(N/\sqrt{\delta})$. If the SWCGP algorithm has selected K atoms in at most K iterations, then it has found the support of the signal $\mathbf{y}$ with probability at least $1-\delta$.

VII. Discussion and conclusion

Underdetermined inverse problems with a sparsity constraint on the solution are found in many areas of modern signal processing. In this paper, we have introduced a novel greedy strategy that in each iteration selects several new elements. The coefficients are then updated using a directional optimisation step in which the update direction is conjugate to the previous update direction. This procedure addressed two issues arising when OMP is applied to large scale problems. On the one hand, the use of directional optimisation reduces the computational cost of each iteration. On the other hand, picking several elements in each iteration reduces the overall number of iterations required.

The use of the conjugate update direction was introduced in an earlier paper [29] and we have here derived a novel recursion that allows a more efficient implementation of this method. We have also given a new convergence guarantee. The main focus was however on the new selection strategy. We have discussed the prior art in this respect and highlighted several disadvantages of current approaches. To overcome these, we presented the new Stagewise Weak selection strategy. This selection strategy has several desirable properties. These are summarised in table III, where we also list advantages and disadvantages of previously suggested approaches.

Using this strategy in OMP does not necessarily offer computational advantages, which were achieved only when combining the stagewise weak selection and the conjugate gradient update. In this paper we presented a range of numerical experiments. Synthetic data highlighted several properties of the method and its good performance. In particular, the weakness parameter allowed a smooth trade-off between the sparsity K/M and computational complexity of the recovery problem. The choice of alpha is therefore a trade-off between performance and algorithm speed and ultimately depends on the particular application. For example, we have demonstrated the applicability of the algorithm to very large data-sets using a dynamic MRI inversion problem, where significant speed advantages were achieved with only minor sacrifices in terms of performance.

Another important advantage of the novel selection step is that it allows a simple extension of many of the theoretical results derived for other OMP type algorithms to our new setting. This was demonstrated here with the help of three theorems that give performance guarantees for our SWCGP algorithm.

Appendix I
Proof of theorem 3

Let $\Gamma^\star$ and $\mathbf{y}^\star_{\Gamma^\star}=\mathbf{\Phi}^\dagger_{\Gamma^\star}\mathbf{x}$ be as in the theorem. In iteration n, assume the algorithm has recovered $\Gamma^{[n]}\subset\Gamma^\star$. Let the residual be $\mathbf{r}^{[n]}=\mathbf{x}-\hat{\mathbf{x}}^{[n]}$ and let $\hat{\mathbf{x}}^\star=\mathbf{\Phi}_{\Gamma^\star}\mathbf{y}^\star_{\Gamma^\star}$. Let the sets $G=\Gamma^\star$ and $B=\{i: i\notin G\}$ be the good and bad sets. We then have from [46]

$$\frac{\|\mathbf{\Phi}_B^T\mathbf{r}^{[n]}\|_\infty}{\|\mathbf{\Phi}_G^T\mathbf{r}^{[n]}\|_\infty}\leq\frac{\|\mathbf{\Phi}_B^T(\mathbf{x}-\hat{\mathbf{x}}^\star)\|_\infty}{\|\mathbf{\Phi}_G^T(\hat{\mathbf{x}}^\star-\hat{\mathbf{x}}^{[n]})\|_\infty}+\frac{\|\mathbf{\Phi}_B^T(\hat{\mathbf{x}}^\star-\hat{\mathbf{x}}^{[n]})\|_\infty}{\|\mathbf{\Phi}_G^T(\hat{\mathbf{x}}^\star-\hat{\mathbf{x}}^{[n]})\|_\infty}\tag{18}$$

where

$$\begin{aligned}\frac{\|\mathbf{\Phi}_B^T(\hat{\mathbf{x}}^\star-\hat{\mathbf{x}}^{[n]})\|_\infty}{\|\mathbf{\Phi}_G^T(\hat{\mathbf{x}}^\star-\hat{\mathbf{x}}^{[n]})\|_\infty}&\leq\max_{i\in B}\|\mathbf{\Phi}_G^\dagger\phi_i\|_1\\&\leq\sqrt{K}\max_{i\in B}\|\mathbf{\Phi}_G^\dagger\phi_i\|_2\\&\leq\sqrt{K}\max_{i\in B}\|(\mathbf{\Phi}_G^T\mathbf{\Phi}_G)^{-1}\|_2\|\mathbf{\Phi}_G^T\phi_i\|_2\\&\leq\frac{\sqrt{K}}{1-\delta_K}\delta_{K+1},\end{aligned}\tag{19}$$

where the last inequality comes from standard properties of the RIP constant (see [27] Proposition 3.1 and 3.2).

The other term can be bounded by (see [46])

$$\frac{\|\mathbf{\Phi}_B^T(\mathbf{x}-\hat{\mathbf{x}}^\star)\|_\infty}{\|\mathbf{\Phi}_G^T(\hat{\mathbf{x}}^\star-\hat{\mathbf{x}}^{[n]})\|_\infty}\leq\frac{\sqrt{K}\|\mathbf{x}-\hat{\mathbf{x}}^\star\|_2}{\sqrt{1-\delta_K}\|\hat{\mathbf{x}}^\star-\hat{\mathbf{x}}^{[n]}\|_2}.$$

In iteration n, the algorithm therefore selects elements from set G if

$$\frac{\sqrt{K}\|\mathbf{x}-\hat{\mathbf{x}}^\star\|_2}{\sqrt{1-\delta_K}\|\hat{\mathbf{x}}^\star-\hat{\mathbf{x}}^{[n]}\|_2}+\frac{\sqrt{K}\delta_{K+1}}{1-\delta_K}<\alpha,\tag{20}$$

which (as both terms on the left need to be positive) is only possible if $\delta_{K+1}<\frac{\alpha}{\sqrt{K}+\alpha}$. Rewriting this and noting that due to the optimality of $\hat{\mathbf{x}}^\star$, the error $(\mathbf{x}-\hat{\mathbf{x}}^\star)$ is orthogonal to all elements in $\mathbf{\Phi}_G$ so that $\|\mathbf{x}-\hat{\mathbf{x}}^{[n]}\|_2^2=\|\hat{\mathbf{x}}^\star-\hat{\mathbf{x}}^{[n]}\|_2^2+\|\mathbf{x}-\hat{\mathbf{x}}^\star\|_2^2$, we get

$$\sqrt{1+\frac{1}{\left(\alpha\sqrt{(1-\delta_K)/K}-\delta_{K+1}/\sqrt{1-\delta_K}\right)^2}}\|\mathbf{x}-\hat{\mathbf{x}}^\star\|_2<\|\mathbf{x}-\hat{\mathbf{x}}^{[n]}\|_2.\tag{21}$$

The same argument as used in the proof of Corollary 4.3 in [46] then proofs the theorem.

Appendix II
Proof of Theorem 4

For any $\mathbf{y}$, let $\mathbf{x}=\mathbf{\Phi}\mathbf{y}+\mathbf{n}$. Let $\mathbf{y}_K$ be the best K-term approximation to $\mathbf{y}$ and note that $\mathbf{x}=\mathbf{\Phi}\mathbf{y}_K+\mathbf{\Phi}(\mathbf{y}-\mathbf{y}_K)+\mathbf{n}$. Let $\tilde{\mathbf{n}}=\mathbf{\Phi}(\mathbf{y}-\mathbf{y}_K)+\mathbf{n}$. Note that by Lemma 6.1 in [27], we have

$$\|\mathbf{\Phi}(\mathbf{y}-\mathbf{y}_K)+\mathbf{n}\|_2\leq\sqrt{1+\delta_K}\left(\|(\mathbf{y}-\mathbf{y}_K)\|_2+\frac{\|(\mathbf{y}-\mathbf{y}_K)\|_1}{\sqrt{K}}\right)+\|\mathbf{n}\|_2.\tag{22}$$

III: Comparision of selection strategies

Selection strategy	Advantages	Disadvantages
MP	Fast to calculate Allows theoretical analysis for general $\mathbf{\Phi}$	Only selects a single element per iteration
ROMP	Good theoretical guarantees Selects several elements per iteration	Poor performance in practice Somewhat more costly to calculate Parameter r difficult to select
StOMP	Fast to calculate Selects several elements per iteration	Theory based on Gaussian matrices only Can (and will) get stuck Difficult to select parameter
Stagewise Weak	Fast to calculate Selects several elements per iteration Shows good performance in practise Allows theoretical analysis for general $\mathbf{\Phi}$ Allows smooth tradeoff between speed and performance	Theoretical bounds are somewhat weaker than those for ROMP with $r = 0.5$ or StOMP for USE matrices.

We bound the error in iteration n using the triangle inequality

$$\|\hat{\mathbf{y}}^{[n]} - \mathbf{y}\|_2 \leq \|\hat{\mathbf{y}}^{[n]} - \mathbf{y}_K\|_2 + \|\mathbf{y}_K - \mathbf{y}\|_2. \tag{23}$$

To bound the first term on the right we note that $\hat{\mathbf{y}}^{[n]} - \mathbf{y}_K$ has at most $2K$ non-zero elements so that by definition of δ_{2K}:

$$\begin{aligned}\|\hat{\mathbf{y}}^{[n]} - \mathbf{y}_K\|_2 &\leq \frac{1}{\sqrt{1-\delta_{2K}}}\|\mathbf{\Phi}(\hat{\mathbf{y}}^{[n]} - \mathbf{y}_K)\|_2 \\ &\leq \frac{1}{\sqrt{1-\delta_{2K}}}\|\mathbf{\Phi}(\hat{\mathbf{y}}^{[n]} - \mathbf{y}_K) - \tilde{\mathbf{n}}\|_2 \\ &\quad + \frac{1}{\sqrt{1-\delta_{2K}}}\|\tilde{\mathbf{n}}\|_2 \\ &\leq \frac{1}{\sqrt{1-\delta_{2K}}}\|\mathbf{x} - \hat{\mathbf{x}}^{[n]}\|_2 + \frac{1}{\sqrt{1-\delta_{2K}}}\|\tilde{\mathbf{n}}\|_2, \\ &\leq \frac{c}{\sqrt{1-\delta_{2K}}}\|\mathbf{x} - \hat{\mathbf{x}}^{\star}\|_2 + \frac{1}{\sqrt{1-\delta_{2K}}}\|\tilde{\mathbf{n}}\|_2, \end{aligned} \tag{24}$$

where we use the triangle inequality and the third line uses $\mathbf{x} = \mathbf{\Phi}\mathbf{y}_K + \tilde{\mathbf{n}}$ and the fourth line is theorem 3 (where c is defined).

To bound $\|\mathbf{x} - \hat{\mathbf{x}}^{\star}\|_2$ we have

$$\begin{aligned}\|\mathbf{x} - \hat{\mathbf{x}}^{\star}\|_2 &= \|\mathbf{\Phi}\mathbf{y} + \mathbf{n} - \mathbf{\Phi}_{\Gamma^\star}\mathbf{\Phi}^{\dagger}_{\Gamma^\star}\mathbf{x}\|_2 \\ &\leq \|\mathbf{\Phi}_{\Gamma^\star}(\mathbf{y}_K - \mathbf{\Phi}^{\dagger}_{\Gamma^\star}\mathbf{x})\|_2 + \|\tilde{\mathbf{n}}\|_2 \\ &\leq \sqrt{1+\delta_K}\|(\mathbf{y}_K - \mathbf{\Phi}^{\dagger}_{\Gamma^\star}\mathbf{x})\|_2 + \|\tilde{\mathbf{n}}\|_2.\end{aligned}$$

Finally we bound $\|(\mathbf{y}_K - \mathbf{\Phi}^{\dagger}_{\Gamma^\star}\mathbf{x})\|_2$ using [28]

$$\|\mathbf{y} - \mathbf{\Phi}^{\dagger}_{\Gamma^\star}\mathbf{x}\|_2 \leq \left(1 + \frac{\sqrt{1+\delta_K}}{\sqrt{1-\delta_K}}\right)\|\tilde{\mathbf{n}}\|_2 \tag{25}$$

to get the constant in the theorem as

$$\hat{c} = \frac{2 - \delta_K + c\left(1 + \sqrt{1+\delta_K} + \frac{(1+\delta_K)}{\sqrt{1-\delta_K}}\right)}{\sqrt{1-\delta_{2K}}}, \tag{26}$$

where c is again as in theorem 3.

For the second part of the theorem we use

$$\|\mathbf{\Phi}\hat{\mathbf{y}}^{[n]} - \mathbf{x}\|_2 = \Omega\|\mathbf{x}\|_2.$$

in the third inequality in equality (24) to get the result.

REFERENCES

[1] M. E. Davies and T. Blumensath, "Faster & greedier: algorithms for sparse reconstruction of large datasets," in *Proc. of the third IEEE International Symposium on Communications, Control, and Signal Processing*, (St Julians, Malta), March 2008.

[2] V. K. Goyal, M. Vetterli, and N. T. Thao, "Quantized overcomplete expansions in R^N: Analysis, synthesis and algorithms," *IEEE Transactions on Information Theory*, vol. 44, pp. 16–31, Jan. 1998.

[3] C. Fevotte and S. Godsill, "Sparse linear regression in unions of bases via Bayesian variable selection," *IEEE Signal Processing Letters*, vol. 13, no. 7, pp. 441–444, 2006.

[4] M. Davies and N. Mitianoudis, "A simple mixture model for sparse overcomplete ICA," *IEE Proc.-Vision, Image and Signal Processing*, vol. 151, pp. 35–43, August 2004.

[5] E. Candès, J. Romberg, and T. Tao, "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information.," *IEEE Transactions on information theory*, vol. 52, pp. 489–509, Feb 2006.

[6] G. Davis, *Adaptive Nonlinear Approximations.* PhD thesis, New York University, 1994.

[7] B. K. Natarajan, "Sparse approximate solutions to linear systems," *SIAM Journal on Computing*, vol. 24, pp. 227–234, Apr 1995.

[8] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM Journal of Scientific Computing*, vol. 20, no. 1, pp. 33–61, 1998.

[9] B. Efron, T. Hastie, I. Johnston, and R. Tibshirani, "Least angle regression," *The Annals of Statistics*, vol. 32, no. 2, pp. 407–499, 2004.

[10] S. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky, "A method for large-scale l_1-regularised least squares problems with applications in signal processing and statistics.," *submitted*, 2007.

[11] M. A. T. Figueiredo, R. D. Nowak, and S. J. Wright, "Gradient projection for sparse reconstruction: application to compressed sensing and other inverse problems," *submitted manuscript*, 2007.

[12] J. Bioucas-Dias and M. Figueiredo, "A new TwIST: two-step iterative shrinkage/thresholding algorithms for image restoration," *IEEE Trans. on Image Processing*, vol. 16, no. 12, pp. 2992 – 3004, 2007.

[13] E. van den Berg and M. P. Friedlander, "Probing the Pareto frontier for basis pursuit solutions," *SIAM J. on Scientific Computing*, vol. 31, no. 2, pp. 890–912, 2008.

[14] R. Griesse and D. A. Lorenz, "A semismooth Newton method for tikhonov functionals with sparsity constraints," *Inverse Problems*, vol. 24, no. 3, 2008.

[15] J. F. Murray and K. Kreutz-Delgado, "An improved FOCUSS-based learning algorithm for solving sparse linear inverse problems," in *Conf. Record of the Thirty-Fifth Asilomar Conf. on Signals, Systems and Computers*, pp. 347–351, 2001.

[16] E. Candès, M. Wakin, and S. Boyd, "Enhancing sparsity by reweighted l1 minimization," tech. rep., California Institute of Technology, 2007.

[17] M. E. Tipping, "Sparse Bayesian learning and the relevance vector machine," *Journal of Machine Learning Research*, vol. 1, pp. 211–244, 2001.

[18] D. Wipf and B. D. Rao, "Sparse Bayesian learning for basis selection," *IEEE Transactions on Signal Processing*, vol. 52, no. 8, pp. 2153–2164, 2004.

[19] R. E. McCulloch and E. I. George, "Approaches for Bayesian variable selection," *Statistica Sinica*, vol. 7, no. 2, pp. 339–374, 1997.

[20] R. E. McCulloch and E. I. George, "Variable selection via Gibbs sampling," *Journal of the American Statistical Association*, pp. 881–889., September 1993.

[21] J. Geweke, "Variable selection and model comparison in regression," in *Bayesian Statistics 5.* (J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith, eds.), Oxford University Press, 1996.

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

[23] Y. C. Pati, R. Rezaifar, and P. S. Krishnaprasad, "Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition," in 27^{th} *Asilomar Conf. on Signals, Systems and Comput.*, Nov. 1993.

[24] S. Chen, S. A. Billings, and W. Luo, "Orthogonal least squares methods and their application to non-linear system identification.," *International Journal of Control*, vol. 50, no. 5, pp. 1873–1896, 1989.

[25] P. Vincent and Y. Bengio, "Kernel matching pursuit," *Machine Learning*, vol. 48, pp. 169–191, 2002.

[26] W. Dai and O. Milenkovic, "Subspace pursuit for compressed sensing: Closing the gap between performance and complexity," *submitted*, 2008.

[27] D. Needell and J. Tropp, "COSAMP: Iterative signal recovery from incomplete and inaccurate samples.," *to appear in Applied Computational Harmonic Analysis*, 2008.

[28] T. Blumensath and M. Davies, "Iterative hard thresholding for compressed sensing," *to appear in Applied and Computational Harmonic Analysis*, 2009.

[29] T. Blumensath and M. Davies, "Gradient pursuits," *IEEE Transactions on Signal Processing*, vol. 56, pp. 2370–2382, June 2008.

[30] D. Donoho, Y. Tsaig, I. Drori, and J. Starck, "Sparse solutions of underdetermined linear equations by stagewise orthogonal matching pursuit," 2006.

[31] D. Needell and R. Vershynin, "Uniform uncertainty principle and signal recovery via regularized orthogonal matching pursuit," *Foundations of Computational Mathematics*, 2009.

[32] D. Needell and R. Vershynin, "Signal recovery from incomplete and inacurate measurements via regularized orthogonal matching pursuit," *submitted*, 2008.

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

[34] G. H. Golub and F. Van Loan, *Matrix Computations*. Johns Hopkins University Press, 3rd ed., 1996.

[35] E. J. Candès and T. Tao, "Decoding by linear programming," *IEEE Transactions on Information Theory*, vol. 51, pp. 4203–4215, 2004.

[36] R. A. Horn and C. R. Johnson, *Matrix Analysis*. Cambridge University Press, 1985.

[37] J. R. Shewchuk, "An introduction to the conjugate gradient method without the agonizing pain," tech. rep., School of Computer Science, Carnegie Mellon University, 1994.

[38] H. Crowder and P. Wolfe, "Linear convergence of the conjugate gradient method," *Numerical Computation*, vol. 16, no. 4, pp. 431–433, 1972.

[39] M. Lustig, D. L. Donoho, and J. M. Pauly, "Sparse MRI: The application of compressed sensing for rapid MR imaging," *submitted*, 2007.

[40] R. Gribonval and M. Nielsen, "Sparse decompositions in "incoherent" dictionaries," in *Proc. IEEE Intl. Conf. on Image Proc. (ICIP'03)*, (Barcelona, Spain), September 2003.

[41] J.-J. Fuchs, "On sparse representations in arbitary redundant bases," *IEEE Transactions on Information Theory*, vol. 50, no. 6, pp. 1341–1344, 2004.

[42] J.-J. Fuchs, "Recovery of exact sparse representations in the presence of bounded noise," tech. rep., Institut de Recherche en Informatique et Systemes Aléatoires, 2004.

[43] D. L. Donoho and M. Elad, "Optimally-sparse representation in general (non-orthogonal) dictionaries via l1 minimization," *Proc. Nat. Aca. Sci.*, vol. 100, pp. 2197–2202, 2003.

[44] D. L. Donoho, M. Elad, and V. Temlyakov., "Stable recovery of sparse overcomplete representations in the presence of noise," *IEEE Transactions On Information Theory*, Februrary 2004.

[45] D. Donoho, "Compressed sensing," *IEEE Trans. on Information Theory*, vol. 52, no. 4, pp. 1289–1306, 2006.

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

[47] J. A. Tropp and A. C. Gilbert, "Signal recovery from partial information via orthogonal matching pursuit," *IEEE Transactions on Information Theory*, vol. 53, no. 12, pp. 4655–4666, 2006.

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

[49] E. Candès, "The restricted isometry property and its implications for compressed sensing," tech. rep., California Institute of Technology, 2008.