Efficient Sparse Recovery Pursuits with Least Squares

Guy Leibovitz*, Raja Giryes†

*guyleibovitz@mail.tau.ac.il, †raja@tauex.tau.ac.il
School of Electrical Engineering, Tel-Aviv University
Tel-Aviv, Israel

Abstract

We present a new greedy strategy, with an efficient implementation technique, that enjoys similar computational complexity and stopping criteria like OMP. Moreover, its recovery performance in the noise free and the Gaussian noise cases is comparable and in many cases better than other existing sparse recovery algorithms both with respect to the theoretical and empirical reconstruction ability. Our framework has other appealing properties. Convergence is always guaranteed by design also in the case that the recovery conditions are violated. In addition, our implementation method is useful for improving the computational costs of other greedy pursuits such as orthogonal least squares (OLS).

I. Introduction

In this paper we consider the problem of finding a sparse representation $\boldsymbol{x} \in \mathbb{R}^n$ in a dictionary $\boldsymbol{D} \in R^{m\times n}$ for the vector $\boldsymbol{y} \in R^m$ such that $\boldsymbol{y} = \boldsymbol{D}\boldsymbol{x} + \boldsymbol{w}$, where $\boldsymbol{w}$ is an added Gaussian noise vector, and $m \leq n$. The sparsity k (number of non-zero elements) of $\boldsymbol{x}$ is its ℓ_0 pseudo-norm $\|\boldsymbol{x}\|_0 = k$. This problem received a lot of attention in the signal processing and statistics communities. It is prevalent in areas such as regression [22], signal and image restoration [3], and compressed sensing [10]. Finding the best k-sparse approximation of the vector $\boldsymbol{y}$ is proven to be NP-hard, and thus only approximate solutions can be obtained in polynomial time, unless the dictionary satisfies some regularity condition. A popular condition is the Restricted Isometry Property (RIP) [5] that defines a condition on subsets of columns of $\boldsymbol{D}$ as follows: A matrix $\boldsymbol{D}$ is said to satisfy the RIP of order s (an integer) with a constant $\delta_s \in (0, 1)$ if

$$(1-\delta_s)\|\boldsymbol{v}\|^2 \leq \|\boldsymbol{D}_T\boldsymbol{v}\|^2 \leq (1+\delta_s)\|\boldsymbol{v}\|^2, \quad \forall |T| \leq s, v \in \mathbb{R}^s, \tag{I.1}$$

where the matrix $\boldsymbol{D}_T$ is comprised of the columns of $\boldsymbol{D}$ indexed by the set of indices T. One of the common approaches for sparse signal estimation relies on ℓ_1-based convex optimization [7, 22]. This strategy has been proven to be very potent in finding the best sparse approximation under the RIP condition [4], but ℓ_1 methods generally require more computations than another widespread methodology, the so-called greedy pursuit approach that includes the popular Orthogonal Matching Pursuit (OMP) and Orthogonal least squares (OLS)[1]. OMP and OLS find the sparse representation by greedily selecting one atom of the dictionary $\boldsymbol{D}$ at a time. OMP picks the most correlated atom of $\boldsymbol{D}$ to the residual error between $\boldsymbol{y}$ and its temporal least-squares estimate with the currently selected columns. OLS on the other hand, improves over OMP by picking the atom that yields the smallest approximation error. However, this comes with an additional computational cost (order of k times the OMP complexity). Optimized OMP (OOMP) [20] is an acceleration of OLS, but requires a significant amount of added memory. In Section S1 of the supplementary material we propose a more efficient method for accelerating OLS based on the tools we develop in this paper. Both OLS and OMP use a one-off strategy, where an atom never leaves the selected support after it enters. One option for improving one-off methods is called back-tracing, the re-consideration of atoms in the selected support. Pursuit algorithms that use this approach include: CoSaMP and Subspace Pursuit (SP) [8, 12, 19, 21], which received a lot of attention for their high reconstruction guarantees ($\delta_{4k} \leq 0.3843$ and $\delta_{3k} \leq 0.4859$ respectively, [11, 21]) Another option is OMP with Replacement (OMPR) [14]. The guarantees for OMPR are also based on the RIP but they impose an additional constraint on the mutual coherence of $\boldsymbol{D}$ (δ_2) and on a parameter of this algorithm that requires tunning.

In this study we propose new greedy methods that resemble OMP and OLS. Both of our proposed pursuits are designed as such that there is no parameter affecting their convergence rate or accuracy, other than an input of either target sparsity (in both of them) or target residual (in one method) similar to the way OMP and OLS work. Our techniques are guaranteed to converge to the true solution under better RIP conditions compared to other greedy strategies. We validate the supremacy of our suggested programs also in simulations.

This paper is organized as follows: Section II breifly describes the OLS algorithm and presents some preliminary lemmas that are used later with the proposed algorithms in Section III. Section IV contains the theoretical performance guarantees and their proofs. Section V follows with numerical examples, and Section VI concludes the paper.

[1] The OLS algorithm entertains a plethora of designations in the literature being rediscovered under different names several times. For example , in statistics it is known as forward stepwise regression. Other names include Least Squares Orthogonal Matching Pursuit (LSOMP), forward-regression, and Order Recursive Matching Pursuit (ORMP) to name a few. See [2] for discussion and literature survey.

II. PRELIMINARIES

We use the following notation in this paper: $\boldsymbol{a}$ is a vector, $\vec{\boldsymbol{a}} = \boldsymbol{a}/\|\boldsymbol{a}\|$ is a unit vector, a is a scalar, and $\boldsymbol{A}$ is a matrix. $\boldsymbol{a}(i)$ is the i-th entry of $\boldsymbol{a}$, whereas $\boldsymbol{a}_{\setminus i}$ is the vector $\boldsymbol{a}$ without entry i, similarly $\boldsymbol{A}_{\setminus i}$ is $\boldsymbol{A}$ without column i, and the i-th column and row are deleted from $\boldsymbol{A}$ to get $\boldsymbol{A}_{\setminus i,\setminus i}$. Unless otherwise noted in the subscript $\|\cdot\|$ is the usual ℓ_2 norm. $\boldsymbol{D} \in \mathbb{R}^{m\times n}$ is the dictionary, $\boldsymbol{d}_{(i)}$ designates the i-th atom (or column) of $\boldsymbol{D}$ ($\|\boldsymbol{d}_{(i)}\|_2 = 1 \ \ \forall i$). $\boldsymbol{x} \in \mathbb{R}^n$ is the unknown sparse representation with sparsity $k = \|\boldsymbol{x}\|_0$, and $\boldsymbol{y} = \boldsymbol{D}\boldsymbol{x} + \boldsymbol{w}$ is the signal we have. Uppercase non-bold letters are sets of indices (e.g. T). We denote by $\boldsymbol{A} = \boldsymbol{D}_T$ the sub-matrix of $\boldsymbol{D}$ comprised of the columns indexed by T, and interchange the notations at convenience. The least squares estimate of $\boldsymbol{y}$ using the atoms in T is denoted as $\hat{\boldsymbol{x}}_{\boldsymbol{A}} = \hat{\boldsymbol{x}}_T = (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{y}$; the orthogonal projection onto the column space of $\boldsymbol{A}$ as $P_{\boldsymbol{A}} = P_T = \boldsymbol{A}(\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T$; and the orthogonal complement $R_{\boldsymbol{A}} = R_T = I - P_{\boldsymbol{A}}$. Finally, $\odot$ represents an element-wise multiplication.

The following are preliminary lemmas that will aid us throughout this paper in the derivation of two new greedy techniques together with their theoretical recovery guarantees. We start with two variants of the Sherman-Morrison matrix inversion lemma for a column addition and deletion [15, 16] that follow from a simple modification of the original lemma.

Lemma II.1 (Sherman-Morrison matrix inversion lemma for column addition)**.** *Let $\boldsymbol{B} = (\boldsymbol{A}^T\boldsymbol{A})^{-1}$ and $\tilde{\boldsymbol{A}} = \begin{bmatrix}\boldsymbol{A}\ \boldsymbol{a}\end{bmatrix}$. Then we may calculate $\tilde{\boldsymbol{B}} = (\tilde{\boldsymbol{A}}^T\tilde{\boldsymbol{A}})^{-1}$ as follows:*

$$\tilde{\boldsymbol{B}} = (\begin{bmatrix}\boldsymbol{A}\ \boldsymbol{a}\end{bmatrix}^T\begin{bmatrix}\boldsymbol{A}\ \boldsymbol{a}\end{bmatrix})^{-1} = \begin{bmatrix}\boldsymbol{B} & 0\\ 0 & 0\end{bmatrix} + \frac{1}{r}\begin{bmatrix}\hat{\boldsymbol{e}}\\ -1\end{bmatrix}\begin{bmatrix}\hat{\boldsymbol{e}}^T & -1\end{bmatrix}, \tag{II.1}$$

where $\hat{\boldsymbol{e}} = \boldsymbol{A}^\dagger\boldsymbol{a}$, and $r = \|R_{\boldsymbol{A}}\boldsymbol{a}\|^2$.

A straight forward consequence of Lemma II.1 is the following update for column removal:

Lemma II.2 (Sherman-Morrison matrix inversion lemma for column removal)**.** *Let $\tilde{\boldsymbol{B}} = (\tilde{\boldsymbol{A}}^T\tilde{\boldsymbol{A}})^{-1}$ with $\tilde{\boldsymbol{A}} = \begin{bmatrix}\boldsymbol{A}\ \boldsymbol{a}\end{bmatrix}$. Then we may calculate $\boldsymbol{B} = (\boldsymbol{A}^T\boldsymbol{A})^{-1}$ as follows:*

$$\hat{\boldsymbol{e}} = -\tilde{\boldsymbol{B}}_{\setminus i}(:,i), \qquad r = \tilde{\boldsymbol{B}}(i,i)^{-1}, \qquad \boldsymbol{B} = \tilde{\boldsymbol{B}}_{\setminus i,\setminus i} - r\hat{\boldsymbol{e}}\hat{\boldsymbol{e}}^T. \tag{II.2}$$

Note that in order to use Lemma II.1 for column insertion at a general location in $\boldsymbol{A}$, simply insert $\boldsymbol{a}$ at the last index and permute $\tilde{\boldsymbol{A}}$ and $\tilde{\boldsymbol{B}}$ afterwards.

Lemma II.3 (The change in error)**.** *Let the mean squared error of estimating $\boldsymbol{y}$ using $\boldsymbol{A}$ be $\|\boldsymbol{y} - \hat{\boldsymbol{y}}\|^2 = \|\boldsymbol{y} - P_{\boldsymbol{A}}\boldsymbol{y}\|^2 = \|R_{\boldsymbol{A}}\boldsymbol{y}\|^2$. Then, the change in $\|R_{\tilde{\boldsymbol{A}}}\boldsymbol{y}\|^2 = \|R_{\begin{bmatrix}\boldsymbol{A}\ \boldsymbol{a}\end{bmatrix}}\boldsymbol{y}\|^2$ after a vector addition, in terms of $\|R_{\boldsymbol{A}}\boldsymbol{y}\|^2$ can be written as:*

$$\left\langle \overrightarrow{R_{\boldsymbol{A}}\boldsymbol{a}}, \boldsymbol{y}\right\rangle^2 = (\boldsymbol{y}^T\overrightarrow{R_{\boldsymbol{A}}\boldsymbol{a}})^2 = \|R_{\boldsymbol{A}}\boldsymbol{y}\|^2 - \|R_{\tilde{\boldsymbol{A}}}\boldsymbol{y}\|^2 = \|P_{\tilde{\boldsymbol{A}}}\boldsymbol{y}\|^2 - \|P_{\boldsymbol{A}}\boldsymbol{y}\|^2. \tag{II.3}$$

Proof. By utilizing lemma II.1, we can write

$$\|R_{\tilde{\boldsymbol{A}}}\boldsymbol{y}\|^2 = \|\boldsymbol{y} - P_{\tilde{\boldsymbol{A}}}\boldsymbol{y}\|^2 \overset{(a)}{=} \|\boldsymbol{y}\|^2 - \boldsymbol{y}^T\boldsymbol{A}\boldsymbol{B}\boldsymbol{A}^T\boldsymbol{y} - \frac{1}{r}\left(\boldsymbol{y}^T\begin{bmatrix}\boldsymbol{A}\ \boldsymbol{a}\end{bmatrix}\begin{bmatrix}\hat{\boldsymbol{e}}\\ -1\end{bmatrix}\right)^2, \tag{II.4}$$

where (a) follows from (II.1) and some simple algebraic steps. Since $\|R_{\boldsymbol{A}}\boldsymbol{y}\|^2 = \|\boldsymbol{y}\|^2 - \|P_{\boldsymbol{A}}\boldsymbol{y}\|^2 = \|\boldsymbol{y}\|^2 - \boldsymbol{y}^T\boldsymbol{A}\boldsymbol{B}\boldsymbol{A}^T\boldsymbol{y}$, we have,

$$\|R_{\tilde{\boldsymbol{A}}}\boldsymbol{y}\|^2 = \|R_{\boldsymbol{A}}\boldsymbol{y}\|^2 - \frac{1}{r}\left(\boldsymbol{y}^T(\boldsymbol{A}\hat{\boldsymbol{e}} - \boldsymbol{a})\right)^2 = \|R_{\boldsymbol{A}}\boldsymbol{y}\|^2 - (\boldsymbol{y}^T\overrightarrow{R_{\boldsymbol{A}}\boldsymbol{a}})^2. \tag{II.5}$$

Reordering (II.5) together with the fact that $\|P_{\boldsymbol{A}}\boldsymbol{y}\|^2 = \|\boldsymbol{y}\|^2 - \|R_{\boldsymbol{A}}\boldsymbol{y}\|^2$ leads to (II.3). □

Lemma II.4 (The value of $\hat{\boldsymbol{x}}_{\boldsymbol{A}}(i)$)**.** *Let $\hat{\boldsymbol{x}}_{\boldsymbol{A}} = (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{y}$, the least squares estimate of $\boldsymbol{x}$. It is in effect the representation of $\boldsymbol{y}$ in a bi-orthogonal basis for the space spanned by the columns of $\boldsymbol{A}$, i.e., its i-th entry is*

$$\hat{\boldsymbol{x}}_{\boldsymbol{A}}(i) = \frac{1}{\|R_{\boldsymbol{A}_{\setminus i}}\boldsymbol{d}_{(i)}\|^2}\left\langle R_{\boldsymbol{A}_{\setminus i}}\boldsymbol{d}_{(i)}, \boldsymbol{y}\right\rangle. \tag{II.6}$$

Proof. Without loss of generality, we prove this formula for the last entry of $\hat{\boldsymbol{x}}_{\boldsymbol{A}}$ (i.e. $i = k$). Let $\tilde{\boldsymbol{A}} = \begin{bmatrix}\boldsymbol{A}\ \boldsymbol{a}\end{bmatrix}$, with k columns, consider the k-th entry of the expression $\hat{\boldsymbol{x}}_{\tilde{\boldsymbol{A}}} = (\tilde{\boldsymbol{A}}^T\tilde{\boldsymbol{A}})^{-1}\tilde{\boldsymbol{A}}^T\boldsymbol{y} = \tilde{\boldsymbol{B}}\tilde{\boldsymbol{A}}^T\boldsymbol{y}$,. Using Lemma II.1 together with the notation therein we may rewrite $\hat{\boldsymbol{x}}_{\tilde{\boldsymbol{A}}}$ as (recall $\boldsymbol{A} = \tilde{\boldsymbol{A}}_{\setminus k}$):

$$\hat{\boldsymbol{x}}_{\tilde{\boldsymbol{A}}}(k) = \frac{1}{\|R_{\boldsymbol{A}}\boldsymbol{a}\|^2}(\boldsymbol{a}^T - \boldsymbol{a}^T\boldsymbol{A}\boldsymbol{B}\boldsymbol{A}^T)\boldsymbol{y},$$

which equals (II.6) for $i = k$. By permuting the entries of $\hat{\boldsymbol{x}}$ and $\tilde{\boldsymbol{A}}$, the same can be proved for all $i \in [1, k]$. □

To conclude this section, the following lemma is an interesting consequence of the results above.

Lemma II.5. *Let $\hat{\boldsymbol{x}} = \tilde{\boldsymbol{B}}\tilde{\boldsymbol{A}}^T\boldsymbol{y}$, with $\tilde{\boldsymbol{B}} = (\tilde{\boldsymbol{A}}^T\tilde{\boldsymbol{A}})^{-1}$, then the least contributing column i of $\tilde{\boldsymbol{A}}$ for the least squares estimate of $\boldsymbol{x}$ from $\boldsymbol{y}$ is the one whose index corresponds to*

$$\arg\min_i \hat{\boldsymbol{x}}(i)^2/\tilde{\boldsymbol{B}}(i,i). \tag{II.7}$$

Proof. From Lemma (II.6) we have that

$$\hat{\boldsymbol{x}}_{\boldsymbol{A}}(i)^2 = \frac{1}{\|R_{\boldsymbol{A}_{\setminus i}}\boldsymbol{a}\|^4}\langle\overrightarrow{R_{\boldsymbol{A}}\boldsymbol{a}}, \boldsymbol{y}\rangle^2.$$

Combining this with the expression for r in (II.2) that specifies the value of the diagonal of $\tilde{\boldsymbol{B}}$, we get that $\langle\overrightarrow{R_{\boldsymbol{A}}\boldsymbol{a}}, \boldsymbol{y}\rangle = \hat{\boldsymbol{x}}(i)^2/\tilde{\boldsymbol{B}}(i,i)$. From Lemma (II.3) we have $\|R_{\boldsymbol{A}}\boldsymbol{y}\|^2 - \|R_{\tilde{\boldsymbol{A}}}\boldsymbol{y}\|^2 = \langle\overrightarrow{R_{\boldsymbol{A}}\boldsymbol{a}}, \boldsymbol{y}\rangle$ which validates the claim. □

A. Orthogonal Least Squares (OLS)

Lemma II.3 implies that given a set of columns $\boldsymbol{A}$ used for the estimation of a signal, adding to it a column that satisfies $\boldsymbol{d} = \arg\max_{\boldsymbol{d}\in\boldsymbol{D}}(\boldsymbol{y}^T\overrightarrow{R_{\boldsymbol{A}}\boldsymbol{d}})^2$ yields the smallest residual among all atoms in the dictionary. This notion is the basis for OLS. Notice that in OMP (and other methods such as SP, CoSaMP and OMPR) the selection criterion is based on $\arg\max_{\boldsymbol{d}\in\boldsymbol{D}}(\boldsymbol{y}^T R_{\boldsymbol{A}}\boldsymbol{d})^2$, lacking the normalization by $\|R_{\boldsymbol{A}}\boldsymbol{d}\|^2$. OLS enlarges its selected support iteratively, where at each step a new atom that satisfies Lemma II.3 is selected. We relegate to the supplementary material the OLS pseudo-code and an acceleration we propose for it.

III. Proposed algorithms

In this section we introduce two new algorithms that utilize a similar approach to OLS, by selecting atoms for inclusion/exclusion based on $\langle\overrightarrow{R_T\boldsymbol{d}_{(j)}}, y\rangle$ as the metric.

A. Orthogonal least squares with replacement (OLSR)

The first method we introduce is called OLS with replacement since it starts by producing a support estimation of size $k+1$, similarly to OLS, followed by subsequent attempts to improves the residual by replacing atoms in the support according to Lemma II.3, until convergence occurs. At each step OLSR extracts an atom from the selected support (of size $k+1$), then, if an atom exists that lowers the residual compared to the beginning of the iteration it is inserted to the support. The algorithm stops when there are no such atoms in the dictionary, ending up with a support of size k.

B. Iterative orthogonal least squares with replacement (IOLSR)

The second improvement to OLS (in the same computational cost) introduced in this paper is the IOLSR algorithm. It resembles Efroymson's algorithm ([18], section 3.3) for stepwise regression. At each step a new atom enters the support according to the regular OLS selection rule (II.3), and a test is performed to see if taking out one of the other atoms in the support will lower the residual compared to the beginning of the iteration (i.e. if the least-contributing column (II.7) is different than the one just added). If yes, a column is removed from the selected support, otherwise, the selection set length is enlarged by 1.

C. IOLSR and OLSR properties

Both methods require a single $\mathcal{O}(\mathrm{Mul})$ operation at each loop iteration, similar to OMP, where $\mathcal{O}(\mathrm{Mul})$ is the complexity of multiplication by $\boldsymbol{D}$. In general $\mathcal{O}(\mathrm{Mul}) = \mathcal{O}(mn)$ but for specific dictionaries it might be lower, e.g., for DCT it is $\mathcal{O}(n\log n)$. To facilitate this, we introduce two length n vectors with an efficient update scheme that requires a single dictionary times vector multiplication. To formulate this update scheme recall Lemma II.3. We calculate $\langle\overrightarrow{R_T\boldsymbol{d}_{(j)}}, \boldsymbol{y}\rangle^2$ for each of the atoms in the dictionary by dividing the square of the elements of $\boldsymbol{c}(j) = \langle R_T\boldsymbol{d}_{(j)}, \boldsymbol{y}\rangle$ by the elements of $\boldsymbol{\rho}(j) = \|R_T\boldsymbol{d}_{(j)}\|^2$. Denote $\boldsymbol{v} = R_T\boldsymbol{d}_{(i)}$, $\tilde{\boldsymbol{\rho}} = \boldsymbol{D}^T\boldsymbol{v}$, where i is an index of the atom that was currently added to T, and $T' = T\cup i$. Then upon the addition of $\boldsymbol{d}_{(i)}$ to the support, $\boldsymbol{c}$ and $\boldsymbol{\rho}$ are updated as follows:

$$\begin{aligned}\boldsymbol{c}_n(j) &= \left\langle R_{T'}\boldsymbol{d}_{(i)}, \boldsymbol{y}\right\rangle = \left\langle(R_T - \vec{\boldsymbol{v}}\vec{\boldsymbol{v}}^T)\boldsymbol{d}_{(i)}, \boldsymbol{y}\right\rangle = \left\langle R_T\boldsymbol{d}_{(i)}, y\right\rangle - \left\langle\vec{\boldsymbol{v}}, \boldsymbol{d}_{(i)}\right\rangle\langle\vec{\boldsymbol{v}}, \boldsymbol{y}\rangle \\ &= \boldsymbol{c}_{n-1}(j) - \tilde{\boldsymbol{\rho}}(j)\boldsymbol{c}_{n-1}(i)/\|\boldsymbol{v}\|^2 \end{aligned}\tag{III.1}$$

$$\begin{aligned}\boldsymbol{\rho}_n(j) &= \|R_{T'}\boldsymbol{d}_{(j)}\|^2 = \left\langle(R_T - \vec{\boldsymbol{v}}\vec{\boldsymbol{v}}^T)\boldsymbol{d}_{(j)}, \boldsymbol{d}_{(j)}\right\rangle = \left\langle R_T\boldsymbol{d}_{(j)}, \boldsymbol{d}_{(j)}\right\rangle - \left\langle\vec{\boldsymbol{v}}, \boldsymbol{d}_{(j)}\right\rangle^2 \\ &= \boldsymbol{\rho}_{n-1}(j) - \tilde{\boldsymbol{\rho}}(j)^2/\|\boldsymbol{v}\|^2. \end{aligned}\tag{III.2}$$

Removing a column works in a similar way (with addition instead of subtraction in (III.1) and (III.2) as the residual is enlarged as the subspace of $\boldsymbol{D}_T$ is reduced). The following two functions perform the above procedure efficiently based on (III.1) and (III.2) and lemmas II.4 and II.5 using the updated value of $\boldsymbol{B}$ and $\hat{\boldsymbol{x}}$ from the previous iteration (denote $\tau = |T|$, we always insert to the last index):

TABLE I
THE OLSR AND IOLSR ALGORITHMS

OLS with replacement (OLSR)	Iterative OLS with replacement (IOLSR)
Input: dictionary $\boldsymbol{D}$, measurement $\boldsymbol{y}$, cardinality k (or target residual ϵ_t -IOLSR only) Output: $\hat{\boldsymbol{x}}$ with k elements (or ϵ_t residual -IOLSR only), and T its support	

$\{T, \boldsymbol{c}, \boldsymbol{\rho}, \epsilon_0\} \leftarrow \mathrm{OLS}(\boldsymbol{D}, \boldsymbol{y}, k+1)$
if $\epsilon_0 \approx 0$, **return** T,$\hat{\boldsymbol{x}} = \boldsymbol{D}_T^{\dagger}\boldsymbol{y}$, and **exit**
$\boldsymbol{c}^0 \leftarrow \boldsymbol{D}^T\boldsymbol{y}$
$\boldsymbol{B} = (\boldsymbol{D}_T^T\boldsymbol{D}_T)^{-1}$
$\hat{\boldsymbol{x}} \leftarrow \boldsymbol{B}\boldsymbol{c}_T^0$
loop
$j \leftarrow \arg\min_{j\in T}\left\{\hat{\boldsymbol{x}}(j)^2/\boldsymbol{B}(j,j)\right\}$
$\{\boldsymbol{c}, \boldsymbol{\rho}\} \leftarrow \mathrm{updRem}(\boldsymbol{D}, \boldsymbol{B}, T, \hat{\boldsymbol{x}}, \boldsymbol{c}, \boldsymbol{\rho}, j)$
remove $\boldsymbol{d}_{(j)}$ from $\boldsymbol{B}$ using (II.2)
$T \leftarrow T \setminus \{j\}$
$i \leftarrow \arg\max_{i\notin T}\left\{\boldsymbol{c}(i)^2/\boldsymbol{\rho}(i)\right\}$
exit loop if $\boldsymbol{c}(i)^2/\boldsymbol{\rho}(i) \geq \hat{\boldsymbol{x}}(j)^2/\boldsymbol{B}(j,j)$
update $\boldsymbol{B}$ with $\boldsymbol{d}_{(i)}$ using (II.1)
$T \leftarrow T \cup \{i\}$
$\hat{\boldsymbol{x}} \leftarrow \boldsymbol{B}\boldsymbol{c}_T^0$
$\{\boldsymbol{c}, \boldsymbol{\rho}\} \leftarrow \mathrm{updAdd}(\boldsymbol{D}, \boldsymbol{B}, T, \hat{\boldsymbol{x}}, \boldsymbol{c}, \boldsymbol{\rho})$
end loop
return T, $\hat{\boldsymbol{x}}$
	$T \leftarrow \{\}$, $\epsilon_0 \leftarrow \|\boldsymbol{y}\|^2$, $\boldsymbol{\rho} \leftarrow \boldsymbol{1}_{N\times 1}$
	$\boldsymbol{c}^0 \leftarrow \boldsymbol{D}^T\boldsymbol{y}$, $\boldsymbol{c} \leftarrow \boldsymbol{c}^0$
	while $(|T| < k+1)$ or $(\epsilon_0 < \epsilon_t$ and $j = |T|)$
	$i \leftarrow \arg\max_{i\notin T}\left\{\boldsymbol{c}(i)^2/\boldsymbol{\rho}(i)\right\}$
	update $\boldsymbol{B}$ with $\boldsymbol{d}_{(i)}$ according to (II.1)
	$T \leftarrow T \cup \{i\}$
	$\hat{\boldsymbol{x}} \leftarrow \boldsymbol{B}\boldsymbol{c}_T^0$
	$\epsilon_0 \leftarrow \epsilon_0 - \hat{\boldsymbol{x}}(|T|)^2/\boldsymbol{B}(|T|,|T|)$
	$\{\boldsymbol{c}, \boldsymbol{\rho}\} \leftarrow \mathrm{updAdd}(\boldsymbol{D}, \boldsymbol{B}, T, \hat{\boldsymbol{x}}, \boldsymbol{c}, \boldsymbol{\rho})$
	$j \leftarrow \arg\min_j\left\{\hat{\boldsymbol{x}}(j)^2/\boldsymbol{B}(j,j)\right\}$
	if $j \neq |T|$
	$\{\boldsymbol{c}, \boldsymbol{\rho}\} \leftarrow \mathrm{updRem}(\boldsymbol{D}, \boldsymbol{B}, T, \hat{\boldsymbol{x}}, \boldsymbol{c}, \boldsymbol{\rho}, j)$
	$\epsilon_0 \leftarrow \epsilon_0 + \hat{\boldsymbol{x}}(j)^2/\boldsymbol{B}(j,j)$
	remove $\boldsymbol{d}_{(j)}$ from $\boldsymbol{B}$ using (II.2)
	$T \leftarrow T \setminus \{j\}$
	end if
	end while
	if $(|T| > k)$ or $(\epsilon_0 + \hat{\boldsymbol{x}}(j)^2/\boldsymbol{B}(j,j) < \epsilon_t)$
	perform: $T \leftarrow T \setminus \{j\}$
	return T, $\hat{\boldsymbol{x}} = \boldsymbol{D}_T^{\dagger}\boldsymbol{y}$

$\{\boldsymbol{c}, \boldsymbol{\rho}\} = \mathrm{updAdd}(\boldsymbol{D}, \boldsymbol{B}, T, \hat{\boldsymbol{x}}, \boldsymbol{c}, \boldsymbol{\rho})$
$\boldsymbol{v} \leftarrow \boldsymbol{D}_T\boldsymbol{B}(:, \tau)$
$\tilde{\boldsymbol{\rho}} \leftarrow \boldsymbol{D}^T\boldsymbol{v}$
$\boldsymbol{c} \leftarrow \boldsymbol{c} - (\hat{\boldsymbol{x}}(k)/\boldsymbol{B}(\tau,\tau))\tilde{\boldsymbol{\rho}}$
$\boldsymbol{\rho} \leftarrow \boldsymbol{\rho} - (1/\boldsymbol{B}(\tau,\tau))\tilde{\boldsymbol{\rho}} \odot \tilde{\boldsymbol{\rho}}$

$\{\boldsymbol{c}, \boldsymbol{\rho}\} = \mathrm{updRem}(\boldsymbol{D}, \boldsymbol{B}, T, \hat{\boldsymbol{x}}, \boldsymbol{c}, \boldsymbol{\rho}, j)$
$\boldsymbol{v} \leftarrow \boldsymbol{D}_T\boldsymbol{B}(:, j)$
$\tilde{\boldsymbol{\rho}} \leftarrow \boldsymbol{D}^T\boldsymbol{v}$
$\boldsymbol{c} \leftarrow \boldsymbol{c} + (\hat{\boldsymbol{x}}(j)/\boldsymbol{B}(j,j))\tilde{\boldsymbol{\rho}}$
$\boldsymbol{\rho} \leftarrow \boldsymbol{\rho} + (1/\boldsymbol{B}(j,j))\tilde{\boldsymbol{\rho}} \odot \tilde{\boldsymbol{\rho}}$

IOLSR and OLSR share the same performance guarantees in theorems IV.6 and IV.4, and we provide convergence speed analysis for OLSR in Theorem IV.5. Note that in simulations IOLSR demonstrates linear dependency on k with a similar slope as OMP and OLSR. Another difference worth noting between IOLSR and OLSR, as well as other methods that enable backtracking such as subspace pursuit [8] or CoSaMP [19] is that IOLSR is able to run with a target residual norm (designated ϵ_t) similarly to OMP and OLS, rather than target sparsity. Target residual as a stopping condition is commonly used in many applications. Usually, IOLSR supplies a better approximation than OLSR, this is because of the fact that the IOLSR subspace is sequentially optimized (i.e. the back tracing is performed on subspaces of increasing sizes rather than on a subspace of constant size), in comparison to OLSR and other methods that begin with a given support. Both algorithms are described in table I, and MATLAB code is provided in the supplementaries.

IV. THEORETICAL PERFORMANCE GUARANTEES

The OLSR and IOLSR performance guarantees will be presented here using a series of lemmas and theorems. To this end the following notation will be used. L is the true support s.t. $\boldsymbol{x}(j) \neq 0$, $\forall j \in L$, and 0 otherwise. T is a set of k indices that represent the chosen support of an algorithm at a specific iteration. $\tilde{L} \triangleq L \setminus T$, $\tilde{T} \triangleq T \setminus L$. $\tilde{\boldsymbol{x}}(i) = \boldsymbol{x}(i)$ $\forall i \in \tilde{L}$, and 0 otherwise, $\tilde{\boldsymbol{y}} = \boldsymbol{D}\tilde{\boldsymbol{x}}$. κ the number of atoms in the true support that are not yet identified ($\kappa = |\tilde{L}| = \|\tilde{\boldsymbol{x}}\|_0$).

In the proofs contained in this section we rely on the following lemma from [17].

Lemma IV.1 (Lemma 2.1 in [17])**.** *Given $\boldsymbol{x}_1, \boldsymbol{x}_2$, such that $\|\boldsymbol{x}_1\|_0 + \|\boldsymbol{x}_2\|_0 \leq 2k$, $\boldsymbol{x}_1 \perp \boldsymbol{x}_2$, and a dictionary $\boldsymbol{D}$ with a RIP constant $\delta < 1$ of order $2k$, then:*

$$|\cos\angle(\boldsymbol{D}\boldsymbol{x}_1, \boldsymbol{D}\boldsymbol{x}_2)| \leq \delta. \tag{IV.1}$$

By using Lemma IV.1 and by noting that the support of $\tilde{\boldsymbol{y}}$ and T are disjoint by definition, we have,

$$\|P_A\tilde{\boldsymbol{y}}\|^2 = \langle P_A\tilde{\boldsymbol{y}}, \tilde{\boldsymbol{y}}\rangle = \|P_A\tilde{\boldsymbol{y}}\|\|\tilde{\boldsymbol{y}}\|\cos\angle(P_A\tilde{\boldsymbol{y}}, \tilde{\boldsymbol{y}}) \leq \|P_A\tilde{\boldsymbol{y}}\|\|\tilde{\boldsymbol{y}}\|\delta.$$

Hence

$$\|P_A\tilde{\boldsymbol{y}}\|^2 \leq \delta^2\|\tilde{\boldsymbol{y}}\|^2. \tag{IV.2}$$

Noting that $\|P_A\tilde{\boldsymbol{y}}\|^2 + \|R_A\tilde{\boldsymbol{y}}\|^2 = \|\tilde{\boldsymbol{y}}\|^2$ we get

$$\|R_A\tilde{\boldsymbol{y}}\|^2 \geq (1-\delta^2)\|\tilde{\boldsymbol{y}}\|^2. \tag{IV.3}$$

Lemma IV.2. *The square of the maximum cosine of the angle between $R_A\boldsymbol{d}_{(i)}$, where $\boldsymbol{d}_{(i)}$ is an atom in $\tilde{L}$, and $R_A\tilde{\boldsymbol{y}}$ obeys*

$$\max_{i\in\tilde{L}} \left(\cos\angle(R_A\boldsymbol{d}_{(i)}, R_A\tilde{\boldsymbol{y}})\right)^2 \geq \frac{1}{\kappa}c_\delta = \frac{1}{\kappa}(1-\delta^2)(1-\delta). \tag{IV.4}$$

Proof. We prove by contradiction, inspired by the proof of Theorem 2.2 in [17]. Consider $\|R_A\tilde{\boldsymbol{y}}\|_2$:

$$\|R_A\tilde{\boldsymbol{y}}\|_2 = \frac{|\langle R_A\tilde{\boldsymbol{y}}, R_A\tilde{\boldsymbol{y}}\rangle|}{\|R_A\tilde{\boldsymbol{y}}\|_2} = \frac{\left\langle\sum_{i\in\tilde{L}} \boldsymbol{x}(i)R_A\boldsymbol{d}_{(i)}, R_A\tilde{\boldsymbol{y}}\right\rangle}{\|R_A\tilde{\boldsymbol{y}}\|_2} \leq \frac{\sum_{i\in\tilde{L}}|\boldsymbol{x}(i)\langle R_A\boldsymbol{d}_{(i)}, R_A\tilde{\boldsymbol{y}}\rangle|}{\|R_A\tilde{\boldsymbol{y}}\|_2}, \tag{IV.5}$$

which leads to:

$$\|R_A\tilde{\boldsymbol{y}}\|_2 \overset{(a)}{\leq} \sum_{i\in\tilde{L}}|\boldsymbol{x}(i)\cos\alpha_i| \overset{(b)}{<} \sqrt{\frac{1}{\kappa}c_\delta}\|\tilde{\boldsymbol{x}}\|_1 \overset{(c)}{\leq} \sqrt{c_\delta}\|\tilde{\boldsymbol{x}}\|_2, \tag{IV.6}$$

where we define $\alpha_i = \angle(R_A\boldsymbol{d}_{(i)}, R_A\tilde{\boldsymbol{y}})$, and use the fact that $\|R_A\boldsymbol{d}_{(i)}\|^2 \leq \|\boldsymbol{d}_{(i)}\|^2 = 1$ in transition (a). For step (b) we assume that for some positive constant c_δ, $|\cos\alpha_i| < \sqrt{c_\delta/\kappa}$, for all $i \in \tilde{L}$. For step (c) we use the relationship $\|\tilde{\boldsymbol{x}}\|_1 \leq \sqrt{\kappa}\|\tilde{\boldsymbol{x}}\|_2$.

On the other hand, By (IV.3) and the RIP we have:

$$\|R_A\tilde{\boldsymbol{y}}\| \geq \sqrt{1-\delta^2}\|\tilde{\boldsymbol{y}}\|_2 \geq \sqrt{1-\delta^2}\sqrt{1-\delta}\|\tilde{\boldsymbol{x}}\|_2. \tag{IV.7}$$

Combining (IV.7) and (IV.6) we get that in order to produce a contradiction we need to set $\sqrt{1-\delta^2}\sqrt{1-\delta} = \sqrt{c_\delta}$. □

The following Lemma gives an upper bound for the contribution to the residual of at least one of the erroneously selected atoms in the chosen support. We will later bound from below the contribution to the residual of a member of $\tilde{L}$.

Lemma IV.3. *For a given support T with κ atoms that are not in the true support of $\boldsymbol{x}$, there exists an atom $\boldsymbol{d}_{(j)}$, $j \in \tilde{T}$ such that:*

$$\frac{1}{\kappa}\frac{\delta^2}{1-\delta}\|\tilde{\boldsymbol{y}}\|^2 \geq \left\langle\overrightarrow{R_T\boldsymbol{d}_{(j)}}, \boldsymbol{y}\right\rangle^2. \tag{IV.8}$$

Proof. We start with a bound on the norm of $P_T\tilde{\boldsymbol{y}}$ from below as follows:

$$\begin{aligned}\|P_T\tilde{\boldsymbol{y}}\|^2 &\overset{(a)}{\geq} (1-\delta)\|\hat{\tilde{\boldsymbol{x}}}\|^2 \overset{(b)}{=} (1-\delta)\sum_{j\in T}\frac{1}{\|R_{T\setminus j}\boldsymbol{d}_{(j)}\|^4}\left\langle R_{T\setminus j}\boldsymbol{d}_{(j)}, \tilde{\boldsymbol{y}}\right\rangle^2 \\ &\overset{(c)}{\geq} (1-\delta)\sum_{j\in\tilde{T}}\frac{1}{\|R_{T\setminus j}\boldsymbol{d}_{(j)}\|^2}\left\langle R_{T\setminus j}\boldsymbol{d}_{(j)}, \boldsymbol{y}\right\rangle^2 = (1-\delta)\sum_{j\in\tilde{T}}\left\langle\overrightarrow{R_{T\setminus j}\boldsymbol{d}_{(j)}}, \boldsymbol{y}\right\rangle^2,\end{aligned}$$

where (a) follows from the RIP, (b) follows from (II.6), and (c) is because we sum on fewer atoms $|\tilde{T}| < |T|$, reduce the power in the denominator from 4 to 2, and replace $\tilde{\boldsymbol{y}}$ with $\boldsymbol{y}$ since $R_{T\setminus j} \perp \boldsymbol{d}_{(j)}$ $\forall j \in T \setminus \tilde{T}$. The assertion in Lemma IV.3 now follows by noting that the smallest value of $\langle\overrightarrow{R_{T\setminus j}\boldsymbol{d}_{(j)}}, \boldsymbol{y}\rangle^2$ for $j \in \tilde{T}$ is no bigger than the average on κ atoms in $\tilde{T}$, and by noting that $\delta^2\|\tilde{\boldsymbol{y}}\|^2 \geq \|P_A\tilde{\boldsymbol{y}}\|^2$ from (IV.2). □

Theorem IV.4 (RIP bound)**.** *Given $\boldsymbol{y} = \boldsymbol{D}\boldsymbol{x}$, where $\|\boldsymbol{x}\|_0 = k$ and $\boldsymbol{D}$ satisfies the RIP of order $2k$ with a constant $\delta_{2k} \leq 0.445$ the OLSR and IOLSR algorithms yield perfect support reconstruction, $T = L$.*

Proof of Theorem IV.4 . Assume that the algorithms converged to a support T, with $\tilde{T} \neq \emptyset$. Using Lemma IV.2 we get that there exists $i \in \tilde{L} \neq \emptyset$ such that:

$$\left\langle\overrightarrow{R_T\boldsymbol{d}_{(i)}}, \boldsymbol{y}\right\rangle^2 \overset{(a)}{=} \left\langle\overrightarrow{R_T\boldsymbol{d}_{(i)}}, R_T\tilde{\boldsymbol{y}}\right\rangle^2 = \|R_T\tilde{\boldsymbol{y}}\|^2|\cos\alpha_i|^2 \geq \|R_T\tilde{\boldsymbol{y}}\|^2\frac{1}{\kappa}c_\delta, \tag{IV.9}$$

where in transition (a) we used $R_T\boldsymbol{y} = R_T\tilde{\boldsymbol{y}}$ since $(\boldsymbol{y}-\tilde{\boldsymbol{y}}) \in \text{span}\{\boldsymbol{D}_T\}$.

The OLSR algorithm stops when there are no atoms that can replace the one that has been extracted (the condition for replacement is that the new residual is lower than the previous one). From Lemma IV.3 we have that the last atom extracted satisfies (IV.8). We combine this with (IV.9) and (IV.3) that yield $\|R_T\tilde{\boldsymbol{y}}\|^2\frac{1}{\kappa}c_\delta \geq \|\tilde{\boldsymbol{y}}\|^2\frac{1}{\kappa}(1-\delta^2)c_\delta$. Therefore we have that in order to contradict the assumption that the stopping criterion was reached with $\boldsymbol{d}_{(i)}$ out of the selected support the following needs to hold:

$$\|\tilde{\boldsymbol{y}}\|^2\frac{1}{\kappa}(1-\delta^2)c_\delta \geq \frac{1}{\kappa}\frac{\delta^2}{1-\delta}\|\tilde{\boldsymbol{y}}\|^2.$$

Substituting c_δ from (IV.4) and reorganizing the terms we get the condition in Theorem IV.4.

IOLSR differs from OLSR in that the new candidate is first inserted into the support, and then an atom for elimination is selected (in OLSR it is the other way around, first an atom is removed from the support, and then a new one that improves the residual is inserted if it exists). The resulting bound is identical. The way to show it is similar except for a few changes. To this end denote by T the selected set at the beginning of the current iteration, set $\tilde{\boldsymbol{y}}' = \tilde{\boldsymbol{y}} - \boldsymbol{d}_{(i)}\boldsymbol{x}(i)$ and note that:

$$\|R_T\tilde{\boldsymbol{y}}\|^2 \geq \|R_T\tilde{\boldsymbol{y}}\|^2 - \left\langle\overrightarrow{R_T\boldsymbol{d}_{(i)}}, \tilde{\boldsymbol{y}}\right\rangle^2 \overset{(a)}{=} \|R_{T\cup i}\tilde{\boldsymbol{y}}'\|^2 \overset{(b)}{\geq} (1-\delta^2)\|\tilde{\boldsymbol{y}}'\|^2, \tag{IV.10}$$

where (II.3) and (IV.3) were used in transitions (a) and (b) respectively. Now, combine (IV.9) with (IV.10) to get the required bound with $\tilde{\boldsymbol{y}}'$ instead of $\tilde{\boldsymbol{y}}$. □

Theorem IV.5. *Let $\gamma = c_\delta - \delta^2/c_\delta$, with c_δ defined by* (IV.4)*. The number of iteration needed for the OLSR algorithm to converge to a solution with residual lower than ϵ_t is bounded from above by*

$$b = k\left(1 + \frac{1}{\gamma}\ln\left(\frac{\|\boldsymbol{y}\|^2}{\epsilon_t e^{c_\delta}}\right)\right). \tag{IV.11}$$

Note that this is a worst case analysis. In practice, only several iterations are needed for convergence in addition to the OLS number of iterations that we use to get the initial k-sparse solution. To get a feeling of the bound, assume $\delta = 1/3$ and $\epsilon_t = 10^{-3}\|\boldsymbol{y}\|^2$, this yields:

$$\gamma \sim 0.44, \quad b \sim 16k,$$

meaning that less than $16k$ iterations will be needed for convergence, regardless of the value of k.

Proof. Consider the OLS algorithm, and recall (II.3) in Lemma II.3,

$$\left\langle \overrightarrow{R_{T_n}\boldsymbol{a}}, \boldsymbol{y} \right\rangle^2 = (\boldsymbol{y}^T \overrightarrow{R_{T_n}\boldsymbol{a}})^2 = \|R_{T_n}\boldsymbol{y}\|^2 - \|R_{T_{n+1}}\boldsymbol{y}\|^2 = \|P_{T_{n+1}}\boldsymbol{y}\|^2 - \|P_{T_n}\boldsymbol{y}\|^2,$$

where the subscripts n and $n+1$ represent the set T in subsequent loop iterations. By using Lemma IV.2 we have $\left\langle \overrightarrow{R_{T_{n+1}}\boldsymbol{a}}, \boldsymbol{y} \right\rangle^2 \geq \|R_{T_n}\boldsymbol{y}\|^2 \frac{c_\delta}{\kappa}$, yielding the following relationship between the residuals in subsequent loop iterations of OLS:

$$\|R_{T_{n+1}}\boldsymbol{y}\|^2 \leq \left(1 - \frac{c_\delta}{\kappa}\right)\|R_{T_n}\boldsymbol{y}\|^2.$$

We can replace $1/\kappa$ with $1/k$ to get a more restrictive bound $(1 - c_\delta/k)^k \|\boldsymbol{y}\|^2 \geq \|R_{T_{n=k}}\boldsymbol{y}\|^2$. For large enough k this converges to: $e^{-c_\delta}\|\boldsymbol{y}\|^2 \geq \|R_{T_{n=k}}\boldsymbol{y}\|^2$. Turning now to the replacement part of the OLSR algorithm, assume that tk $(t > 0)$ additional iterations were performed in this stage. From Lemma IV.3 we have that at each step we take out of the support an atom with at most $\frac{\delta^2}{\kappa(1-\delta)}\|\tilde{\boldsymbol{y}}\|^2$ contribution to the residual, which can be further bounded by using (IV.3), yielding $\frac{\delta^2}{\kappa c_\delta}\|R_{T_n}\tilde{\boldsymbol{y}}\|^2$. Hence we get that at iteration tk

$$\|R_{T_{n=tk+k}}\boldsymbol{y}\|^2 \leq \left(1 - \frac{\gamma}{k}\right)^{tk}\|R_{T_{n=k}}\boldsymbol{y}\|^2 \leq \left(1 - \frac{\gamma}{k}\right)^{tk} e^{-c_\delta}\|\boldsymbol{y}\|^2,$$

with $\gamma = c_\delta - \delta^2/c_\delta$ (note that if $\delta < 0.445$, $\gamma > 0$). Thus for the estimation error to be less than ϵ_t we need $t \geq \gamma^{-1}\ln(\|\boldsymbol{y}\|^2/\epsilon_t e^{c_\delta})$, with $(1+t)k$ as an upper bound on the number of iterations. □

The remainder of this section deals with the case that the measurements are corrupted by an additive white Gaussian noise (AWGN). To this end denote the noise variance by σ^2, and recall that $\boldsymbol{y}_0 = \boldsymbol{D}\boldsymbol{x}$, $\boldsymbol{y} = \boldsymbol{y}_0 + \boldsymbol{w}$, $\boldsymbol{w} \sim \mathcal{N}(0, \sigma^2\boldsymbol{I})$.

Theorem IV.6. *Given measurements corrupted by AWGN with variance σ^2, and some parameter $a \geq 0$, a perfect support reconstruction is achieved by OLSR and IOLSR with probability exceeding $1 - (\sqrt{\pi(1+a)\log n}n^a)^{-1}$ if*

$$\frac{\|\boldsymbol{y}_0\|^2}{\sigma^2 k} \geq \frac{(1+\sqrt{1-\delta})^2(2(1+a)\log n)}{((1-\delta)(1-\delta^2)-\delta)^2}.$$

The proof of Theorem IV.6 is relegated to the supplementary material.

The following is a consequence of Theorem IV.6:

Corollary IV.6.1. *The root mean squared error of the estimation of $\boldsymbol{x}$ by the OLSR and IOLSR algorithms is bounded, with probability exceeding $1 - \left(\sqrt{\pi(1+a)\log n}n^a\right)^{-1}$, by*

$$\|x - \hat{x}\| \leq \sqrt{k}\sigma\left(\frac{1}{\sqrt{1-\delta^2}} + \frac{(1+\delta)(1+\sqrt{1-\delta})}{\sqrt{1-\delta}((1-\delta)(1-\delta^2)-\delta)}\sqrt{2(1+a)\log n}\right). \tag{IV.12}$$

The proof of corollary IV.6.1 is relegated to the supplementary material. Its implication is that the IOLSR and OLSR error is proportional, up to a constant times $\log n$, to the error $(\sqrt{k}\sigma)$ of an oracle estimator that knows the true support. These bounds are similar to the ones achieved for other methods including SP and CoSaMP [6, 12].

V. Simulations

In this section we demonstrate the performance of our algorithms numerically using MATLAB routines that are available in the supplementary material.

The results appear in Figure 1. There are three plots: (a) A phase transition diagram, following the methodology of [9]. We fix m and variate n and k according to two auxiliary variables. We use a threshold of $\|\boldsymbol{x} - \hat{\boldsymbol{x}}\|^2/\|\boldsymbol{x}\|^2 \leq 10^{-4}$ and plot the resulting curve. $\boldsymbol{D}$ is normalized Gaussian matrix, and $\boldsymbol{x}$ is a Gaussian signal. Results are averaged over 50 realizations. For BP

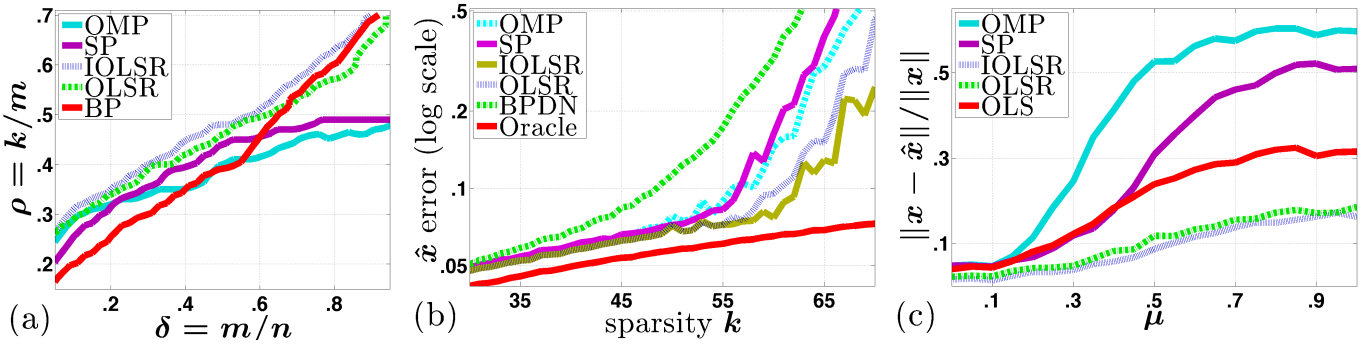

Fig. 1. (a) Phase transition diagram for $m = 400$, areas below the lines of each method represent an error threshold of $\|\hat{\boldsymbol{x}} - \boldsymbol{x}\|^2/\|\boldsymbol{x}\|^2 \leq 10^{-4}$. (b) Error ($\|\hat{\boldsymbol{x}} - \boldsymbol{x}\|$) vs. sparsity in presence of noise $\sigma = 0.1\|\boldsymbol{y}_0\|/\sqrt{m}$, $n = 600$, $m = 200$ compared to an oracle that knows the true support. (c) Error as a function of a coherency damaging parameter μ; $n = 300$, $m = 100$, $k = 30$.

[7] we used the CVX package [13] to solve $\min \|\boldsymbol{x}\|_1$ s.t $\boldsymbol{y} = \boldsymbol{D}\boldsymbol{x}$, and then to improve its recovery we extract the maximal k values from the result and compute a new approximation for the sparse representation by least squares. (b) An experiment with noise, where we fix the size of the dictionary, corrupt the measurements by AWGN with $\sigma = 0.1 \cdot \|\boldsymbol{y}_0\|/\sqrt{m}$, and plot the reconstruction error vs. the sparsity. Results are averaged over 1000 instances of $\boldsymbol{D}$ and $\boldsymbol{x}$ drawn from a Gaussian distribution. For BPDN [7] we used the TFOCS package [1] to solve $\min_{\boldsymbol{x}} \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{D}\boldsymbol{x}\|_2^2 + \lambda\|\boldsymbol{x}\|_1$ with $\lambda = \sqrt{2\log n}\sigma$ following [7]. (c) This plot demonstrates resilience to correlation in the dictionary. For a fixed n, m, k we generate $\boldsymbol{D}$ from a Gaussian distribution. We then increase the correlation in the dictionary by performing $\boldsymbol{d}_{(i)} = \boldsymbol{d}_{(i)} + \mu\boldsymbol{d}_{(i+1)}$ for each atom, repeating the process for 5 times to increase the effect. Results are averaged over 1000 realizations.

VI. Concluding Remarks and Future Work

We have presented here two new pursuits for sparse signal recovery, along with theoretical analysis of their properties. We maintain that IOLSR is a good alternative to OMP and OLS in applications, where the target of the sparse approximation is the magnitude of the residual (stemming perhaps from knowledge of noise variance), and where optimizing various method parameters may be hard. Both IOLSR and OLSR are simple to implement and demonstrate little overhead in regards to OMP. We posit that this kind of fast-converging approach can lead to interesting developments in applications such as dictionary learning where optimization is often carried one atom at a time, and in applications suffering from highly coherent dictionaries, as the correlation of an atom to the selected support is built into the algorithms (in the form of the vector $\boldsymbol{\rho}$) and does not require additional computations.

REFERENCES

[1] S. Becker, E. Candes, and M. Grant. TFOCS: Matlab software, version 1.4. http://cvxr.com/tfocs, October 2014.

[2] T. Blumensath and M.E. Davies. On the difference between orthogonal matching pursuit and orthogonal least squares. 2007.

[3] A.M. Bruckstein, D.L. Donoho, and M. Elad. From sparse solutions of systems of equations to sparse modeling of signals and images. *SIAM review*, 51(1):34–81, 2009.

[4] T. Cai and A. Zhang. Sharp RIP bound for sparse signal and low-rank matrix recovery. *Applied and Computational Harmonic Analysis*, 35(1):74–93, 2013.

[5] E. Candes and T. Tao. Decoding by linear programming. *IEEE Transactions on Information Theory*, 51(12):4203–4215, 2005.

[6] E. Candes and T. Tao. The dantzig selector: Statistical estimation when p is much larger than n. *The Annals of Statistics*, 35(6):2313–2351, 2007.

[7] S.S. Chen, D.L. Donoho, and M.A. Saunders. Atomic decomposition by basis pursuit. *SIAM review*, 43(1):129–159, 2001.

[8] W. Dai and O. Milenkovic. Subspace pursuit for compressive sensing signal reconstruction. *Information Theory, IEEE Transactions on*, 55(5):2230–2249, 2009.

[9] D. Donoho and J. Tanner. Observed universality of phase transitions in high-dimensional geometry, with implications for modern data analysis and signal processing. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 367(1906):4273–4293, 2009.

[10] Y.C. Eldar and G. Kutyniok. *Compressed sensing: theory and applications*. Cambridge University Press, 2012.

[11] S. Foucart. Sparse recovery algorithms: sufficient conditions in terms of restricted isometry constants. In *Approximation Theory XIII: San Antonio 2010*, pages 65–77. Springer, 2012.

[12] R. Giryes and M. Elad. RIP-based near-oracle performance guarantees for SP, CoSaMP, and IHT. *IEEE Transactions on Signal Processing*, 60(3):1465–1468, March 2012.

[13] M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming, version 2.1. http://cvxr.com/cvx, March 2014.

[14] P. Jain, A. Tewari, and Inderjit S.D. Orthogonal matching pursuit with replacement. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 1215–1223. Curran Associates, Inc., 2011.

[15] P.A. Janakiraman and S. Renganathan. Recursive computation of pseudo-inverse of matrices. *Automatica*, 18(5):631–633, 1982.

[16] M. Khan. Updating inverse of a matrix when a column is added. Technical report, UBC, 2008.

[17] C. Ling-Hua and W. Jwo-Yuh. An improved rip-based performance guarantee for sparse signal recovery via orthogonal matching pursuit. *IEEE Transactions on Information Theory*, 60(9):5702–5715, Sept 2014.

[18] A. Miller. *Subset selection in regression*. CRC Press, 2002.

[19] D. Needell and J.A. Tropp. CoSaMP: Iterative signal recovery from incomplete and inaccurate samples. *Applied and Computational Harmonic Analysis*, 26(3):301–321, 2009.

[20] L. Rebollo-Neira and D. Lowe. Optimized orthogonal matching pursuit approach. *IEEE signal processing Letters*, 9(4):137–140, 2002.

[21] C.-B. Song, S.-T. Xia, and X.-J. Liu. Improved analysis for subspace pursuit algorithm in terms of restricted isometry constant. *IEEE Signal Processing Letters*, 21(11):1365–1369, 2014.

[22] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

Efficient Sparse Recovery Pursuits with Least Squares
Supplemental Material

S1. ALGORITHMS

A. Orthogonal least squares

Cited in table S1 for completeness, The computationally demanding step in OLS is the calculation of $\|R_T \boldsymbol{d}_{(i)}\|$ for each atom in $\boldsymbol{D}$ at every loop iteration, which costs $\mathcal{O}(mnk)$. In section S1-B we propose a more computationally efficient way to calculate OLS using two auxiliary length-n vectors.

B. Fast orthogonal least squares

Designated Fast OLS, this method requires only two auxiliary length n vectors to be stored in memory. One for storing the residual norm of each atom in the current iteration and the other to store the magnitude of the correlation between the measurement residual and each atom. Utilizing the auxiliary vectors, the runtime is reduced to that of regular OMP. As was mentioned earlier, it resembles OOMP [20] in its mechanics, but it requires substantially less memory - $\mathcal{O}(2n)$ in FOLS vs. $\mathcal{O}(mn)$ in OOMP. The FOLS algorithm is depicted alongside OLS in table S1. The computational cost of the steps in the loop is $\mathcal{O}(2n + mn + mk + k^2)$. Note that there is a single step that requires $\mathcal{O}(mn)$ flops, similar to OMP. The way FOLS operates is by keeping two vectors:

$$\boldsymbol{c}(j) = \left\langle R_T \boldsymbol{d}_{(j)}, y \right\rangle, \qquad \boldsymbol{\rho}(j) = \|R_T \boldsymbol{d}_{(j)}\|^2$$

that are updated using (denote $\boldsymbol{v} = R_T \boldsymbol{d}_{(i)}$, $\tilde{\boldsymbol{\rho}} = \boldsymbol{D}^T \boldsymbol{v}$, $T' = T \cup i$):

$$\begin{aligned}
\boldsymbol{c}(j) &= \left\langle R_{T'} \boldsymbol{d}_{(i)}, \boldsymbol{y} \right\rangle = \left\langle (R_T - \vec{\boldsymbol{v}}\vec{\boldsymbol{v}}^T)\boldsymbol{d}_{(i)}, \boldsymbol{y} \right\rangle = \left\langle R_T \boldsymbol{d}_{(i)}, y \right\rangle - \left\langle \vec{\boldsymbol{v}}, \boldsymbol{d}_{(i)} \right\rangle \langle \vec{\boldsymbol{v}}, \boldsymbol{y} \rangle \\
&= \boldsymbol{c}(j) - \frac{\tilde{\boldsymbol{\rho}}(j)\boldsymbol{c}(i)}{\|\boldsymbol{v}\|^2} \qquad \text{(S1.1)} \\
\boldsymbol{\rho}(j) &= \|R_{T'} \boldsymbol{d}_{(j)}\|^2 = \left\langle (R_T - \vec{\boldsymbol{v}}\vec{\boldsymbol{v}}^T)\boldsymbol{d}_{(j)}, \boldsymbol{d}_{(j)} \right\rangle = \left\langle R_T \boldsymbol{d}_{(j)}, \boldsymbol{d}_{(j)} \right\rangle - \left\langle \vec{\boldsymbol{v}}, \boldsymbol{d}_{(j)} \right\rangle^2 \\
&= \boldsymbol{\rho}(j) - \frac{\tilde{\boldsymbol{\rho}}(j)^2}{\|\boldsymbol{v}\|^2}. \qquad \text{(S1.2)}
\end{aligned}$$

S2. PROOFS

Proof of Theorem IV.6. We describe the proof for $\kappa = k$, since this is the most restrictive case, it can be easily generalized and checked for the case $\kappa < k$. Hence, $\tilde{\boldsymbol{y}} = \boldsymbol{y}$. Denote for use in this proof:

$$\boldsymbol{w}_T = \frac{P_T \boldsymbol{w}}{\|\boldsymbol{y}_0\|}, \quad w_i = \frac{\left\langle \overrightarrow{R_T \boldsymbol{d}_{(i)}}, \boldsymbol{w} \right\rangle}{\|\boldsymbol{y}_0\|}. \qquad \text{(S2.1)}$$

Consider the proof of Lemma IV.3. By using (S2.1) and noting that

$$\|P_T \boldsymbol{y}\| = \|P_T \boldsymbol{y}_0 + P_T \boldsymbol{w}\| \leq \|P_T \boldsymbol{y}_0\| + \|P_T \boldsymbol{w}\| \leq (\delta + \|\boldsymbol{w}_T\|)\|\boldsymbol{y}_0\|, \qquad \text{(S2.2)}$$

Lemma IV.3 takes the form

$$\frac{\delta + \|\boldsymbol{w}_T\|}{\sqrt{k(1-\delta)}}\|\boldsymbol{y}_0\| \geq \min_{j \in T} \left| \left\langle \overrightarrow{R_T \boldsymbol{d}_{(j)}}, \boldsymbol{y} \right\rangle \right|. \qquad \text{(S2.3)}$$

From Lemma IV.2 we have

$$\left| \left\langle \overrightarrow{R_T \boldsymbol{d}_{(i)}}, R_T \boldsymbol{y}_0 + \boldsymbol{w} \right\rangle \right| \geq \|R_T \boldsymbol{y}_0\| |\cos \alpha_i| - \|\boldsymbol{y}_0\| |w_i| \geq \|\boldsymbol{y}_0\| \left(\sqrt{\frac{c_\delta (1-\delta^2)}{k}} - |w_i| \right), \qquad \text{(S2.4)}$$

for some $i \in L$ (the true support). So, to contradict the assumption that the algorithms converged with $\boldsymbol{d}_{(i)}$ out of the estimated support, combine (S2.3) and (S2.4) and replace the value for c_δ

$$(1-\delta)(1-\delta^2) - \delta \geq \|\boldsymbol{w}_T\| + \sqrt{k(1-\delta)}|w_i|. \qquad \text{(S2.5)}$$

Note that w_i and $\boldsymbol{w}_T$ reside in orthogonal spaces to each other and that $\boldsymbol{w}_T$ is with dimension k whereas w_i is of dimension 1. Thus we can use the following confidence interval developed in [6]:

$$Pr\left(\sup_{\boldsymbol{a} \in \boldsymbol{D}} |\langle \boldsymbol{a}, \boldsymbol{w} \rangle| > \sigma \sqrt{2(1+a)\log n} \right) \leq \left(\sqrt{\pi (1+a) \log n} n^a \right)^{-1}. \qquad \text{(S2.6)}$$

We can provide a bound for the right hand side of (S2.5) with high probability:

$$\|\boldsymbol{w}_T\| + \sqrt{k(1-\delta)}|w_i| \leq \frac{\sqrt{k}(1 + \sqrt{1-\delta})\sigma\sqrt{2(1+a)\log N}}{\|\boldsymbol{y}_0\|}. \qquad \text{(S2.7)}$$

TABLE S1
THE OLS AND FOLS ALGORITHMS

Orthogonal least squares (OLS)	Fast orthogonal least squares (FOLS)
Input: dictionary $\boldsymbol{D}$, measurement $\boldsymbol{y}$, either target cardinality k or target residual norm ϵ_t Output: $\hat{\boldsymbol{x}}$ with k elements or ϵ_t residual, and T its support	
$T \leftarrow \{\}$, $\epsilon_0 \leftarrow \|\boldsymbol{y}\|^2$, $\boldsymbol{r} \leftarrow \boldsymbol{y}$ **while** $\|T\| < k$ or $\epsilon_0 < \epsilon_t$ $i \leftarrow \arg\max_i \left\{ \langle \boldsymbol{r}, \boldsymbol{d}_{(i)} \rangle^2 / \|R_T \boldsymbol{d}_{(i)}\|^2 \right\}$ $T \leftarrow T \cup \{i\}$ $\epsilon_0 \leftarrow \epsilon_0 - \langle \boldsymbol{r}, \boldsymbol{d}_{(i)} \rangle^2 / \|R_T \boldsymbol{d}_{(i)}\|^2$ $\boldsymbol{r} \leftarrow R_T \boldsymbol{y}$ **end while** **return** T, $\hat{\boldsymbol{x}} = \boldsymbol{D}_T^\dagger \boldsymbol{y}$	$T \leftarrow \{\}$, $\epsilon_0 \leftarrow \|\boldsymbol{y}\|^2$, $\boldsymbol{\rho} \leftarrow \mathbf{1}_{n\times 1}$, $\boldsymbol{c} \leftarrow \boldsymbol{D}^T \boldsymbol{y}$ **while** $\|T\| < k$ or $\epsilon_0 < \epsilon_t$ $i \leftarrow \arg\max_{i \notin T} \boldsymbol{c}(i)^2 / \boldsymbol{\rho}(i)$ $T \leftarrow T \cup \{i\}$ $\epsilon_0 \leftarrow \epsilon_0 - \boldsymbol{c}(i)^2 / \boldsymbol{\rho}(i)$ $\boldsymbol{v} \leftarrow R_T \boldsymbol{d}_{(i)}$ $\tilde{\boldsymbol{\rho}} \leftarrow \boldsymbol{D}^T \boldsymbol{v}$ $\boldsymbol{c} \leftarrow \boldsymbol{c} - \frac{\boldsymbol{c}(i)}{\|\boldsymbol{v}\|^2} \tilde{\boldsymbol{\rho}}$ $\boldsymbol{\rho} \leftarrow \boldsymbol{\rho} - \frac{1}{\|\boldsymbol{v}\|^2} \tilde{\boldsymbol{\rho}} \odot \tilde{\boldsymbol{\rho}}$ **end while** **return** T, $\hat{\boldsymbol{x}} = \boldsymbol{D}_T^\dagger \boldsymbol{y}$

Combine (S2.5) and (S2.7) to conclude the proof. □

Proof of Corollary IV.6.1. The oracle square root of the MSE satisfies:

$$\sqrt{\frac{k}{1-\delta^2}}\sigma \geq \|\boldsymbol{x} - \hat{\boldsymbol{x}}_{oracle}\| \geq \sqrt{k}\sigma \tag{S2.8}$$

This can be seen directly by noting that the oracle MSE satisfies $\|\boldsymbol{x} - \hat{\boldsymbol{x}}_{oracle}\|^2 = \text{trace}\{(D_L^T D_L)^{-1}\}\sigma^2$, and combining the results from (IV.3), and Lemma II.1. The worst case MSE on the other hand (assuming none of the vectors belonging to the true support were selected) satisfies (assume $\boldsymbol{A} = \boldsymbol{D}_T$ is the selected support, and that $\boldsymbol{A}\hat{\boldsymbol{x}}_{worst\ case} = \hat{\boldsymbol{y}}_{worst\ case}$):

$$\begin{aligned}
\|\boldsymbol{x} - \hat{\boldsymbol{x}}_{worst\ case}\| &= \|\boldsymbol{x} - (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T(\boldsymbol{y}_0 + \boldsymbol{w})\| \\
&\leq \|(\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{y}_0\| + \|\boldsymbol{x}\| + \|(\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{w}\| \\
&\leq \|\hat{\boldsymbol{x}}_{worst\ case}\| + \|\boldsymbol{x}\| + \sqrt{\frac{k}{1-\delta^2}}\sigma \\
&\leq \frac{1}{\sqrt{1-\delta}}(\|\boldsymbol{y}_0\| + \|\hat{\boldsymbol{y}}_0\|) + \sqrt{\frac{k}{1-\delta^2}}\sigma \\
&\leq \frac{1+\delta}{\sqrt{1-\delta}}\|\boldsymbol{y}_0\| + \sqrt{\frac{k}{1-\delta^2}}\sigma
\end{aligned} \tag{S2.9}$$

Combining the condition from Theorem IV.6 and the expression from (S2.8) we get that (S2.9) can be written as:

$$\|\boldsymbol{x} - \hat{\boldsymbol{x}}\| \leq \sqrt{\frac{k}{1-\delta^2}}\sigma + \frac{1+\delta}{\sqrt{1-\delta}}\|\boldsymbol{y}_0\| \cdot \mathbf{1}_{\left\{\sigma \geq \frac{((1-\delta)(1-\delta^2)-\delta)\|\boldsymbol{y}_0\|}{\sqrt{k}(1+\sqrt{1-\delta})\sqrt{2(1+a)\log N}}\right\}} \tag{S2.10}$$

where $\mathbf{1}_{\{\cdot\}}$ is the indicator function. The result is linear in σ, except for a step in a certain threshold. To get (IV.12), approximate (S2.10) as a sum of two slopes, the first for the oracle MSE, and the second to approximate the step at $\sigma = \frac{((1-\delta)(1-\delta^2)-\delta)\|\boldsymbol{y}_0\|}{\sqrt{k}(1+\sqrt{1-\delta})\sqrt{2(1+a)\log N}}$ □