Forward - Backward Greedy Algorithms for Signal Demixing

Nikhil Rao[†] Parikshit Shah[#] Stephen Wright[⋆]
[†]Department of Electrical and Computer Engineering
[#] Wisconsin Institute for Discovery
[⋆] Department of Computer Sciences
University of Wisconsin - Madison

***Abstract*—Signal demixing arises in many applications. Common among these are the separation of sparse and low rank components in image and video processing, sparse and group sparse models in multitask learning and spikes and sinusoids in source separation problems. For specific problems of interest, many methods exist to perform recovery, but an approach that generalizes to arbitrary notions of simplicity has not been forthcoming. We propose a framework for signal demixing when the components are defined to be simple in a fairly arbitrary manner. Our method remains computationally simple and can be used in a variety of practical applications.**

I. Introduction

The problem of demixing signals from noisy versions of their sum plays a central role in many signal processing and machine learning applications. For example, in video processing applications, the signal can be expressed as a sum of a low rank (the slowly varying background) and sparse (the objects in the foreground) components [1], [2]. In multitask learning applications, modeling the parameters to be decomposable into sparse and group sparse components has been shown to yield superior performance as compared to the traditional group sparse based framework [3], [4]. Separating group sparse and low rank components has also been studied in this framework [5], [6]. The problem of demixing signals that are sparse in the canonical bases and sparse in the Fourier basis is an age old problem in signal processing. (For more applications and theory, please refer to [7]).

A common theme that unifies these problems is the recovery of signals that have a "simple" decomposition with respect to a certain basis or frame. For example, sparse components can be seen as a linear combination of a small number of scaled canonical basis vectors. Similarly, low rank matrices can be seen as a linear combination of a small number of unit rank matrices, and so on. Such concepts can be extended to include graphs with a few edges or cliques, low rank tensors, (overlapping) group sparse components and so on. These "simple" components that can be combined to yield meaningful signals are called atoms, and the notion of atomic norms [8] yield tractable convex heuristics to perform penalized recovery of these signals.

We briefly describe the concepts of atoms and atomic norms. We assume that a signal $\boldsymbol{x}$ can be expressed as a conic combination of elements $\boldsymbol{a} \in \mathcal{A}$:

$$\boldsymbol{x} = \sum_{\boldsymbol{a} \in \mathcal{A}} c_{\boldsymbol{a}} \boldsymbol{a}, \quad c_{\boldsymbol{a}} \geq 0$$

The set $\mathcal{A}$ can be uncountabley infinite, but the signal can be expressed as a sum of a finite number of *atoms* $\boldsymbol{a}$, justifying the sum notation above. The *atomic norm* [8] is defined as

$$\|\boldsymbol{x}\|_{\mathcal{A}} := \inf \left\{ \sum_{\boldsymbol{a} \in \mathcal{A}} c_a : \boldsymbol{x} = \sum_{\boldsymbol{a} \in \mathcal{A}} c_a \boldsymbol{a}, \quad c_a \geq 0 \right\}. \quad (1)$$

The notion of atoms and the atomic norm generalizes pre-existing notions of simplicity to include various structures. For example, when the atoms are the (signed) canonical basis vectors, the atomic norm reduces to the ℓ_1 norm. When the atoms are unit rank matrices, we obtain the nuclear norm. The set $\mathcal{A}$ can be fairly arbitrary in general, and our goal is to develop an algorithm that solves demixing problems involving such arbitrary formulations.

The demixing problem involves recovering a signal of the form $\boldsymbol{x} = \boldsymbol{x}^1 + \boldsymbol{x}^2$ from observations $\boldsymbol{y}$ via a sensing matrix $\boldsymbol{\Phi}$, where $\boldsymbol{x}^1$ and $\boldsymbol{x}^2$ can be expressed compactly with respect to different atomic sets $\mathcal{A}_1$ and $\mathcal{A}_2$. Recovery can then be expressed as the solution of the following convex optimization problem:

$$\begin{aligned} \underset{\boldsymbol{x}^1,\boldsymbol{x}^2}{\text{minimize}} \qquad & f(\boldsymbol{x}^1,\boldsymbol{x}^2) := \frac{1}{2}\|\boldsymbol{y}-\boldsymbol{\Phi}(\boldsymbol{x}^1+\boldsymbol{x}^2)\|^2 \\ \text{subject to} \qquad & \|\boldsymbol{x}^1\|_{\mathcal{A}_1}\le\tau_1 \text{ and } \|\boldsymbol{x}^2\|_{\mathcal{A}_2}\le \tau_2. \end{aligned} \tag{2}$$

To solve (2), we propose an algorithm based on the greedy conditional gradient (CG) method. Greedy methods are typically computationally very efficient, and scale well to large problems. However, a drawback of these schemes is that the greedy strategy might pick suboptimal atoms to represent the signal. To overcome this, [9] propose a truncation step that purges suboptimal atoms, provided the objective function does not deteriorate too much. This step is again very efficient, owing to the quadratic nature of the objective function. We call our method CoGEnT - demix, (Conditional Gradient with Enhancement and Truncation, adapted to the demixing problem.) We informally explain it below, and provide a more detailed description in the next section.

Each iteration starts by choosing an atom from $\mathcal{A}_1$ that nearly minimizes its inner product with the gradient of the objective function with respect to $\boldsymbol{x}_1$; this is the forward step with respect to $\mathcal{A}_1$. One then performs a backward step for $\mathcal{A}_1$. Next follows a similar forward step with respect to $\mathcal{A}_2$, followed by a backward step for $\mathcal{A}_2$. We then proceed to the next iteration, unless convergence is flagged. Note that the backward steps are taken only if they do not deteriorate the objective function beyond a specified threshold. The entire procedure is repeated until a termination condition is satisfied.

The rest of the paper is outlined as follows: in the next section, we give some more details of the algorithm we propose, including a pseudocode. In Section III, we test our method on a variety of demixing problems commonly encountered in signal processing. We conclude our paper in Section IV.

Algorithm 1 CoGEnT - demix

```
1:  **Inputs:** $\forall r \in [R]$, Characterization of $\mathcal{A}^r$, bounds $\tau^r$, threshold $\eta \in [0,1)$
2:  **Initialize** $t \leftarrow 0$, $\boldsymbol{a}_0^r \in \mathcal{A}^r$, $\boldsymbol{A}_t^r \leftarrow [\boldsymbol{a}_t^r]$ , $c_t^r \leftarrow \tau^r$
3:  **repeat**
4:    **for** $r \in [R]$ **do**
5:      $\nabla f_t^r := \frac{\partial f_t}{\partial \boldsymbol{x}^r}$
6:      $\boldsymbol{a}_{t+1}^r \leftarrow \arg\min_{\boldsymbol{a}\in\mathcal{A}^r} \langle \nabla f_t^r, \boldsymbol{a}\rangle$;
7:      $\tilde{\boldsymbol{A}}_{t+1}^r \leftarrow [\boldsymbol{A}_t^r \;\; \boldsymbol{a}_{t+1}^r]$
8:      $\gamma_{t+1} \leftarrow \arg\min_{\gamma\in[0,1]} f(\boldsymbol{x}_t^r + \gamma(\tau\boldsymbol{a}_{t+1}^r - \boldsymbol{x}_t^r))$;
9:      $\tilde{c}_{t+1}^r \leftarrow \arg\min_{c_{t+1}} f(\tilde{\boldsymbol{A}}_{t+1}^r c_{t+1})$ s.t. $\|c_{t+1}\|_1 \le \tau^r$, $c_{t+1} \ge 0$ with the output from Step 8 as a warm start;
10:     $\tilde{\boldsymbol{x}}_{t+1}^r = \tilde{\boldsymbol{A}}_{t+1}^r \tilde{c}_{t+1}^r$;
11:     Threshold $F_{t+1}^r := \eta f(\boldsymbol{x}_t^r) + (1-\eta) f(\tilde{\boldsymbol{x}}_{t+1}^r)$
12:     $[\boldsymbol{A}_{t+1}^r, c_{t+1}^r, x_{t+1}^r]$
          $=$ TRUNCATE$(\tilde{A}_{t+1}^r, \tilde{c}_{t+1}^r, \tau, F_{t+1}^r)$;
13:   **end for**
14:   $t \leftarrow t+1$
15: **until convergence**
```

II. Algorithm

In the previous section, we informally explained our method CoGEnT - demix for the case of separating two signals that have a parsimonious representation with respect to their respective bases. Algorithm 1 explains the method in full generality, incorporating an arbitrary number $2 \le R < \infty$ of regularizers. We write $f(\boldsymbol{x}^r)$ to denote the function $f(\boldsymbol{x}^1, \cdots, \boldsymbol{x}^r)$ keeping all variables except $\boldsymbol{x}^r$ fixed.

We note below a few points with regards to Algorithm 1:

1) Algorithm 1 is the CoGEnT algorithm introduced in [9] and further developed in [10] for convex recovery of generally constrained signals. With a slight modification in the method, we extend it to the case of recovery of a mixture of signals.
2) Note that the (cyclic) alternating minimization scheme mentioned in the algorithm can be replaced by a method that randomly chooses an index $r \in [R]$ and proceeds with the forward and backward steps. In case of coordinate descent methods, such random selection of covariates has shown to sometimes outperform cyclic selection. A thorough study of cyclic versus random coordinate selection is beyond the scope of this paper.

Step 12 can be implemented in a variety of ways. The implementation we choose for most of our experiments in detailed below in Algorithm 2. Note that the backward step can be used to delete multiple atoms from the current representation, so long as the increase in the objective function value stays below a predefined threshold. This allows the resulting representation to be extremely parsimonious, even though we select atoms in a greedy fashion.

A. Effect of Enhancement and Truncation Steps

From Algorithm 1, we see that at each iteration, we perform what we call “enhancement” and “truncation” steps. A simple experiment in sparse signal recovery will highlight the effect that these steps have, compared to the standard conditional gradient method.

Algorithm 2 : TRUNCATE($\tilde{A}_{t+1}, \tilde{c}_{t+1}, \tau, F_{t+1}$)

```
1:  Input: Current basis $\tilde{\boldsymbol{A}}_{t+1}$, coefficient vector $\tilde{c}_{t+1}$, iterate $\tilde{\boldsymbol{x}}_{t+1} = \tilde{A}_{t+1}\tilde{c}_{t+1}$; bound $\tau$; threshold $F_{t+1}$;
2:  continue ← 1;
3:  while continue= 1 do
4:      $\hat{\boldsymbol{a}}_{t+1} \leftarrow \arg\min_{\boldsymbol{a} \in \tilde{\boldsymbol{A}}_{t+1}} f(\tilde{\boldsymbol{x}}_{t+1} - c_{\boldsymbol{a}}\boldsymbol{a})$
5:      $\hat{\boldsymbol{A}}_{t+1} \leftarrow \tilde{\boldsymbol{A}}_{t+1} \backslash \{\hat{\boldsymbol{a}}_{t+1}\}$;
6:      Find $\hat{c}_{t+1} \geq 0$ with $\|\hat{c}_{t+1}\|_1 \leq \tau$ such that $f(\hat{\boldsymbol{A}}_{t+1}\hat{c}_{t+1}) \leq f(\tilde{\boldsymbol{x}}_{t+1} - (\tilde{c}_{\hat{\boldsymbol{a}}_{t+1}})_{t+1}\hat{\boldsymbol{a}}_{t+1})$;
7:      if $f(\hat{A}_{t+1}\hat{c}_{t+1}) \leq F_{t+1}$ then
8:          $\tilde{\boldsymbol{A}}_{t+1} \leftarrow \hat{\boldsymbol{A}}_{t+1}$;
9:          $\tilde{\boldsymbol{x}}_{t+1} \leftarrow \hat{\boldsymbol{A}}_{t+1}\hat{c}_{t+1}$;
10:         $\tilde{c}_{t+1} \leftarrow \hat{c}_{t+1}$;
11:     else
12:         continue ← 0;
13:     end if
14: end while
15: $\boldsymbol{A}_{t+1} \leftarrow \tilde{\boldsymbol{A}}_{t+1}$; $\boldsymbol{x}_{t+1} \leftarrow \tilde{\boldsymbol{x}}_{t+1}$; $c_{t+1} \leftarrow \tilde{c}_{t+1}$;
16: Output: Possibly reduced basis $\boldsymbol{A}_{t+1}$, coefficient vector $c_{t+1} \geq 0$, and iterate $\boldsymbol{x}_{t+1}$.
```

We consider a sparse signal $\boldsymbol{x}$ of length $p = 20000$, with 5% of randomly set to nonzero values. Setting $n = 5000$, we construct the $n \times p$ matrix $\boldsymbol{\Phi}$ to have i.i.d. Gaussian entries, and corrupt the measurements with Gaussian noise (AWGN) of standard deviation $\sigma = 0.01$. We set $\tau = \|\boldsymbol{x}^\star\|_1$, where $\boldsymbol{x}^\star$ is the chosen optimal signal. Fig 1 plots the objective function value (on a logarithmic scale) as a function of time and the number of iterations taken. Clearly, we can see that the method we used achieves much faster convergence. In fact, the enhancement step is key in achieving faster convergence rates, and is costly related to the Chebyshev Weak Greedy Algorithm considered in [11].

We note below a few special cases that are typically of interest to the signal processing community, and show that the greedy selection step 6 can be solved very efficiently

- When $\|\boldsymbol{x}\|_{\mathcal{A}} = \|\boldsymbol{x}\|_1$, we merely need to pick the coordinate with maximum absolute value.
- When $\|\boldsymbol{x}\|_{\mathcal{A}} = \|\boldsymbol{x}\|_*$, step 6 involves computing the leading singular vectors of the gradient matrix. Note that this can be performed efficiently using power iterations, and is an order of magnitude cheaper than computing the full SVD of the matrix, as will be the case when proximal point methods are used.
- For group sparse signals $\|\boldsymbol{x}\|_{\mathcal{A}} = \sum_{G \in \mathcal{G}} \|\boldsymbol{x}_G\|_2$, we only need to pick the group with maximal norm. The same scheme will work even when the groups overlap arbitrarily, thus obviating the need to explicitly replicate variables for typical first order methods.

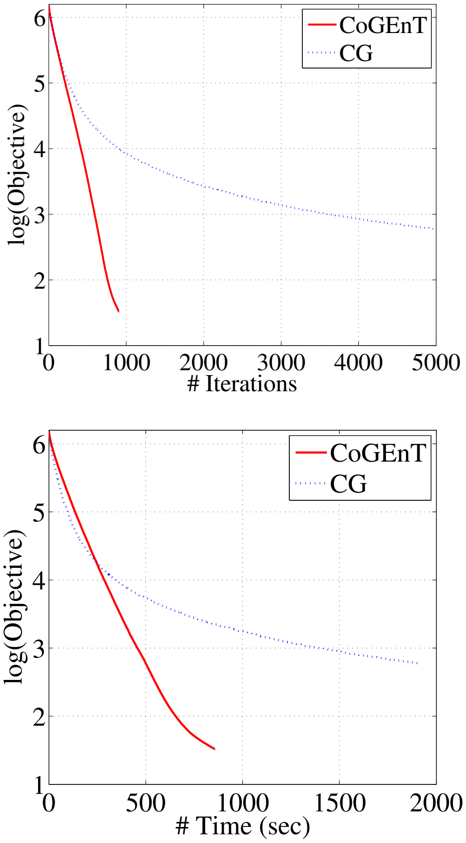
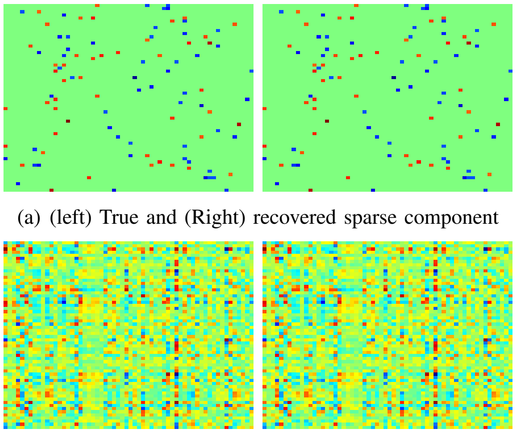


Fig. 1: Comparison between CoGEnT and standard conditional gradient (CG).

III. Experiments and Results

In our first example, we consider the standard recovery of sparse + low rank matrices. We consider a matrix of size 50×50, which is a sum of a random rank 4 matrix and a sparse matrix with 100 entries. The sets $\mathcal{A}_1$ and $\mathcal{A}_2$ are defined in the usual way for these types of matrices. Figure 2 shows that CoGEnT recovers the components.

We also consider recovery of a mixture of signals that are sparse in the canonical and DCT bases. We generated random signals with sparsity level 10 in each of the bases, and applied our method to perform recovery. Fig 3 shows that our method indeed recovers the components accurately



(a) (left) True and (Right) recovered sparse component

(b) (left) True and (Right) recovered low rank component

Fig. 2: Recovery of a sparse + low rank matrix. The left column shows true components, and the right column shows recovered components. Error in each recovered component is at most 10^{-7}.

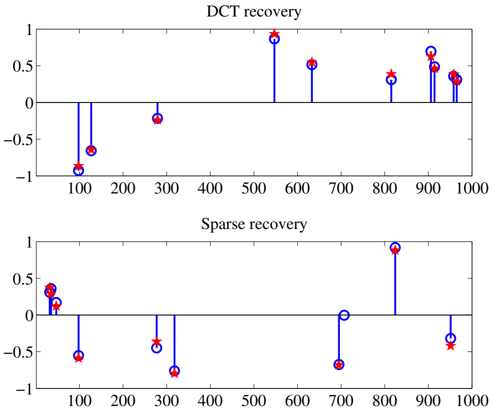


Fig. 3: Recovery of a signal that is sparse in the DCT and canonical basis. The MSE for the top figure is 2.3×10^{-5}, and that for the lower figure is 3.3×10^{-5}. The blue bars represent the true components and the red stars represent the recovered coefficients

A. *Novel Application: Graph Deconvolution*

Finally we apply the demixing method we present here to the problem of deconvolution of graphs. More formally, consider two simple, undirected weighted graphs $\mathcal{G}_1 = (V, W_1)$ and $\mathcal{G}_2 = (V, W_2)$ where V represents a (common) vertex set and W_1, W_2 are the weighted adjacency matrices, with superposition $W = W_1 + W_2$. Problems of this form are of interest in *covariance estimation*: W_1 and W_2 may correspond to covariance matrices of random vectors X_1 and X_2, and from samples of $X = X_1 + X_2$, one may wish to recover the covariances W_1 and W_2.

As an example, we consider a graph of $|V| = 50$ nodes in which $\mathcal{G}_1$ and $\mathcal{G}_2$ are each restricted to a specific family of graphs $\mathfrak{T}_1$ and $\mathfrak{T}_2$, respectively, with the following properties.

- $\mathfrak{T}_1$ is the class of all tree-structured graphs on 50 nodes. Note that the only information we exploit here is the fact that $\mathcal{G}_1$ is tree structured. Neither the edges of the tree nor the edge weights are known.
- $\mathfrak{T}_2$ is the class of two-dimensional 5×10 grid graphs on 50 nodes. The nodes of the graph are known up to a cyclic permutation. Once again, neither the edges of the graph nor the corresponding weights are known. The only information available is that one of the 50 cyclic permutations of the nodes yields the desired grid-structured graph.

For set $\mathfrak{T}_1$, we define the atomic set $\mathcal{A}_1$ to be the set of all matrices with Frobenius norm 1, whose nonzero structure is the adjacency matrix of a tree. For the set $\mathfrak{T}_2$ we define the atomic set $\mathcal{A}_2$ as follows. Let $\mathcal{P} \subseteq \mathbb{R}^{n \times n}$ denote the set of all permutation matrices corresponding to the cyclic permutations (that is, permutations in the cyclic group of order n). Let $\mathcal{G}(p, q)$ (with $pq = n$) denote the set of all weighted adjacency matrices (of unit Frobenius norm) of $p \times q$ grid graphs with a fixed canonical labeling of the nodes. The atomic set $\mathcal{A}_2$ is the set of weighted adjacency matrices for cyclic permutations of all these adjacency matrices.

Given these definitions, and assuming that we observe the full matrices, we state this deconvolution problem as:

$$\begin{array}{ll} \underset{X_1, X_2}{\text{minimize}} & \frac{1}{2}\|W - X_1 - X_2\|^2 \\ \text{subject to} & \|X_1\|_{\mathcal{A}_1} \leq \tau_1 \text{ and } \|X_2\|_{\mathcal{A}_2} \leq \tau_2. \end{array}$$

From the algorithm presented in Algorithm 1, we see that an efficient implementation of the greedy step 6 is possible when the dual norm is efficiently computable. Indeed, for any norm $\| \cdot \|_{\mathcal{A}}$, the dual is given by

$$\| \cdot \|_{\mathcal{A}}^* = \sup_{\|\boldsymbol{x}\|_{\mathcal{A}} \leq 1} \langle \cdot, \boldsymbol{x} \rangle$$

and we see that the greedy step amounts to finding the argument that achieves the supremum in the above definition, with $(\cdot)$ being the negative gradient of the loss function at the present iteration.

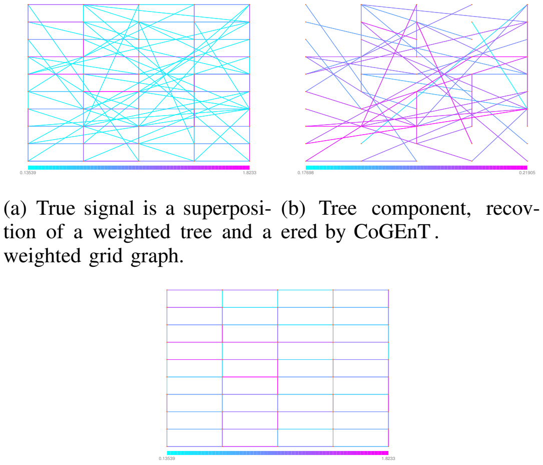
For the setting of recovering graph adjacency matrices, the variational descriptions of the dual atomic norms are given by:

$$\begin{aligned}\|Y\|_{\mathcal{A}_i}^* &= \max_{\|Z\|_{\mathcal{A}_i} \leq 1} [\text{trace}\,(ZY)] \\ &= \max_{A \in \mathcal{A}_i} [\text{trace}\,(AY)]\end{aligned}$$

For $\mathcal{A}_1$, the dual norm essentially amounts to computation of a maximum weight spanning tree, while for A_2, the dual norm can be computed in a straightforward way by sweeping through the n possible permutations of the grid graph to solve:

$$\|Y\|_{\mathcal{A}_2}^* = \max_{P \in \mathcal{P}, \|\mathcal{G}(p,q)\|_F \leq 1} \text{trace} \left(P^{'} \mathcal{G}(p,q) P Y \right).$$

Our problem instances are generated as follows. We created a random tree by generating a random (symmetric) matrix with entries distributed as $\mathcal{U}[0,1]$, and extracting its maximum weight spanning tree. The grid component was also chosen with similarly chosen random weights. The resulting graphs were then superposed and then randomly permuted. Results are shown in Fig. 4. CoGEnT achieves exact recovery; that is, the edges as well as the edge weights of the constituent graphs are correctly recovered.



(a) True signal is a superposition of a weighted tree and a weighted grid graph.

(b) Tree component, recovered by CoGEnT .

(c) Grid graph component, recovered by CoGEnT .

Fig. 4: Recovering constituent graph components from a superposition of weighted graphs. Edge weights are color-coded, with darker colors representing higher weights. CoGEnT correctly deconvolves the graph into its constituent components. (Best seen in color)

IV. Conclusions and Open Questions

In this paper, we extended the CoGEnT optimization framework to solve problems in signal demixing. The method proceeds by solving one iteration of the minimization problem with respect to a single variable, keeping all other variables fixed. The method allows one to separate signals that are simple in a very general sense of the term, and only relies on an efficient method to compute the dual of the (convex) constraint on the simplicity of the signal. Experiments on a variety of problems show that CoGEnT - demix indeed recovers constituent signals from corrupted measurements to within reasonable accuracy.

Recent results have hinted that upon appropriate initialization of the variables, alternating minimization schemes converge to the optimal solution even though the problem is convex in each variable alone. It will be interesting to investigate such convergence criteria for general atomic norm based signal demixing as considered in this paper. Furthermore, the ability to handle arbitrary notions of simplicity in the solution opens up a wide variety of problems that can be considered and solved for.

References

[1] H. Ji, S. Huang, Z. Shen, and Y. Xu, “Robust video restoration by joint sparse and low rank matrix approximation,” *SIAM Journal on Imaging Sciences*, vol. 4, no. 4, pp. 1122–1142, 2011.

[2] A. E. Waters, A. C. Sankaranarayanan, and R. G. Baraniuk, “Sparcs: Recovering low-rank and sparse matrices from compressive measurements.” in *NIPS*, 2011, pp. 1089–1097.

[3] A. Jalali, P. D. Ravikumar, S. Sanghavi, and C. Ruan, “A dirty model for multi-task learning,” *Advances in Neural Information Processing Systems*, vol. 23, pp. 964–972, 2010.

[4] A. Jalali, P. Ravikumar, and S. Sanghavi, “A dirty model for multiple sparse regression.”

[5] J. Chen, J. Zhou, and J. Ye, “Integrating low-rank and group-sparse structures for robust multi-task learning,” in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2011, pp. 42–50.

[6] X. Cui, J. Huang, S. Zhang, and D. N. Metaxas, “Background subtraction using low rank and group sparsity constraints,” in *Computer Vision–ECCV 2012*. Springer, 2012, pp. 612–625.

[7] M. B. McCoy, V. Cevher, Q. T. Dinh, A. Asaei, and L. Baldassarre, “Convexity in source separation: Models, geometry, and algorithms,” preprint arXiv:1311.0258, 2013.

[8] V. Chandrasekaran, B. Recht, P. A. Parrilo, and A. S. Willsky, “The convex geometry of linear inverse problems,” *Foundations of Computational Mathematics*, vol. 12, no. 6, pp. 805–849, 2012.

[9] N. Rao, P. Shah, and S. Wright, “Conditional gradient with enhancement and truncation for atomic-norm regularization,” *NIPS workshop on Greedy Algorithms*, 2013.

[10] ——, “Forward-backward greedy algorithms for atomic norm regularization,” *arXiv preprint arXiv:1404.5692*, 2014.

[11] V. Temlyakov, *Greedy approximation*. Cambridge University Press, 2011.