A Nearly-Linear Time Framework for Graph-Structured Sparsity

Chinmay Hegde CHINMAY@MIT.EDU
Piotr Indyk INDYK@MIT.EDU
Ludwig Schmidt* LUDWIGS@MIT.EDU
Massachusetts Institute of Technology, Cambridge, MA 02139, USA

Abstract

We introduce a framework for sparsity structures defined via graphs. Our approach is flexible and generalizes several previously studied sparsity models. Moreover, we provide efficient projection algorithms for our sparsity model that run in nearly-linear time. In the context of sparse recovery, we show that our framework achieves an information-theoretically optimal sample complexity for a wide range of parameters. We complement our theoretical analysis with experiments demonstrating that our algorithms also improve on prior work in practice.

1. Introduction

Over the past decade, sparsity has emerged as an important tool in several fields including signal processing, statistics, and machine learning. In compressive sensing, sparsity reduces the sample complexity of measuring a signal, and statistics utilizes sparsity for high-dimensional inference tasks. In many settings, sparsity is a useful ingredient because it enables us to model structure in high-dimensional data while still remaining a mathematically tractable concept. For instance, natural images are often sparse when represented in a wavelet basis, and objects in a classification task usually belong to only a small number of classes.

Due to the success of sparsity, a natural question is how we can refine the notion of sparsity in order to capture more complex structures. There are many examples where such an approach is applicable: (i) large wavelet coefficients of natural images tend to form connected *trees*, (ii) active genes can be arranged in functional *groups*, and (iii) approximate point sources in astronomical data often form *clusters*. In such cases, exploiting this additional structure can lead to improved compression ratio for images, better multi-label classification, or smaller sample complexity in compressive sensing and statistics. Hence an important question is the following: how can we model such sparsity structures, and how can we make effective use of this additional information in a computationally efficient manner?

There has been a wide range of work addressing these questions, e.g., (Yuan & Lin, 2006; Jacob et al., 2009; He & Carin, 2009; Kim & Xing, 2010; Bi & Kwok, 2011; Huang et al., 2011; Duarte & Eldar, 2011; Bach et al., 2012b; Rao et al., 2012; Negahban et al., 2012; Simon et al., 2013; El Halabi & Cevher, 2015). Usually, the proposed solutions offer a trade-off between the following conflicting goals:

Generality What range of sparsity structures does the approach apply to?

Statistical efficiency What statistical performance improvements does the use of structure enable?

Computational efficiency How fast are the resulting algorithms?

In this paper, we introduce a framework for sparsity models defined through *graphs*, and we show that it achieves a compelling trade-off between the goals outlined above. At a high level, our approach applies to data with an underlying graph structure in which the large coefficients form a small number of connected components (optionally with additional constraints on the edges). Our approach offers three main features: (i) *Generality*: the framework encompasses several previously studied sparsity models, e.g., tree sparsity and cluster sparsity. (ii) *Statistical efficiency*: our sparsity model leads to reduced sample complexity in sparse recovery and achieves the information-theoretic optimum for a wide range of parameters. (iii) *Computational efficiency*: we give a nearly-linear time algorithm for our sparsity model, significantly improving on prior work both in theory and in practice. Due to the growing size of data sets encountered in science and engineering, algorithms with (nearly-)linear running time are becoming increasingly important.

Proceedings of the 32^{nd} International Conference on Machine Learning, Lille, France, 2015. JMLR: W&CP volume 37.

*Authors ordered alphabetically.

We achieve these goals by connecting our sparsity model to the *prize collecting Steiner tree* (PCST) problem, which has been studied in combinatorial optimization and approximation algorithms. To establish this connection, we introduce a generalized version of the PCST problem and give a nearly-linear time algorithm for our variant. We believe that our sparsity model and the underlying algorithms are useful beyond sparse recovery, and we have already obtained results in this direction. To keep the presentation in this paper coherent, we focus on our results for sparse recovery and briefly mention further applications in Sec. 7.

Before we present our theoretical results in Sections 3 to 5, we give an overview in Section 2. Section 6 complements our theoretical results with an empirical evaluation on both synthetic and real data (a background-subtracted image, an angiogram, and an image of text). We defer proofs and additional details to the supplementary material.

Basic notation Let $[d]$ be the set $\{1, 2, \ldots, d\}$. We say that a vector $\beta \in \mathbb{R}^d$ is s-sparse if at most s of its coefficients are nonzero. The support of β contains the indices corresponding to nonzero entries in β, i.e., $\text{supp}(\beta) = \{i \in [d] \,|\, \beta_i \neq 0\}$. Given a subset $S \subseteq [d]$, we write β_S for the restriction of β to indices in S: we have $(\beta_S)_i = \beta_i$ for $i \in S$ and $(\beta_S)_i = 0$ otherwise. The ℓ_2-norm of β is $\|\beta\| = \sqrt{\sum_{i\in[d]} \beta_i^2}$.

Sparsity models In some cases, we have more information about a vector than only "standard" s-sparsity. A natural way of encoding such additional structure is via *sparsity models* (Baraniuk et al., 2010): let $\mathbb{M}$ be a family of supports, i.e., $\mathbb{M} = \{S_1, S_2, \ldots, S_L\}$ where $S_i \subseteq [d]$. Then the corresponding sparsity model $\mathcal{M}$ is the set of vectors supported on one of the S_i:

$$\mathcal{M} \;=\; \{\beta \in \mathbb{R}^d \,|\, \text{supp}(\beta) \subseteq S \text{ for some } S \in \mathbb{M}\}\;. \quad (1)$$

2. Our contributions

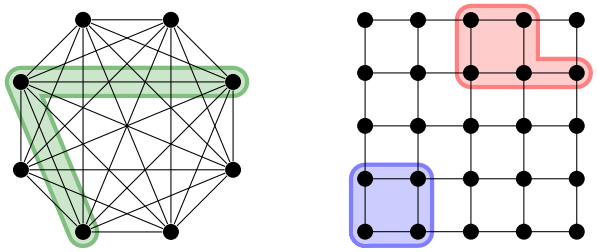
We state our main contributions in the context of sparse recovery (see Section 7 for further applications). Our goal is to estimate an unknown s-sparse vector $\beta \in \mathbb{R}^d$ from observations of the form

$$y \;=\; X\beta + e\;, \quad (2)$$

where $X \in \mathbb{R}^{n\times d}$ is the design matrix, $y \in \mathbb{R}^n$ are the observations, and $e \in \mathbb{R}^n$ is an observation noise vector. By imposing various assumptions on X and e, sparse recovery encompasses problems such as sparse linear regression and compressive sensing.

2.1. Weighted graph model (WGM)

The core of our framework for structured sparsity is a novel, **general** sparsity model which we call the *weighted graph model*. In the WGM, we use an underlying graph $G = (V, E)$ defined on the coefficients of the unknown vector β, i.e., $V = [d]$. Moreover, the graph is weighted and we denote the edge weights with $w : E \to \mathbb{N}$. We identify supports $S \subseteq [d]$ with subgraphs in G, in particular *forests* (unions of individual trees). Intuitively, the WGM captures sparsity structures with a small number of connected components in G. In order to control the sparsity patterns, the WGM offers three parameters:



(a) s-sparsity (b) Cluster sparsity

Figure 1. Two examples of the weighted graph model. (a) In a complete graph, any s-sparse support can be mapped to a single tree ($g = 1$). (b) Using a grid graph, we can model a small number of clusters in an image by setting g accordingly. For simplicity, we use unit edge weights and set $B = s - g$ in both examples.

- s, the total sparsity of S.
- g, the maximum number of connected components formed by the forest F corresponding to S.
- B, the bound on the total weight $w(F)$ of edges in the forest F corresponding to S.

More formally, let $\gamma(H)$ be the number of connected components in a graph H. Then we can define the WGM:

Definition 1. *The (G, s, g, B)-WGM is the set of supports*

$$\mathbb{M} = \{S \subseteq [d] \mid |S| = s \text{ and there is a } F \subseteq G \text{ with } V_F = S,\ \gamma(F) = g, \text{ and } w(F) \leq B\}\;. \quad (3)$$

Fig. 1 shows how two sparsity structures can be encoded with the WGM. Since our sparsity model applies to *arbitrary* graphs G, it can describe a wide range of structures. In particular, the model generalizes several previously studied sparsity models, including 1D-clusters, (wavelet) tree hierarchies, the Earth Mover Distance (EMD) model, and the unweighted graph model (see Table 1).

2.2. Recovery of vectors in the WGM

We analyze the **statistical efficiency** of our framework in the context of sparse recovery. In particular, we prove that

the sample complexity of recovering vectors in the WGM is provably smaller than the sample complexity for "standard" s-sparse vectors. To formally state this result, we first introduce a key property of graphs.

Definition 2. *Let $G = (V, E)$ be a weighted graph with edge weights $w : E \to \mathbb{N}$. Then the weight-degree $\rho(v)$ of a node v is the largest number of adjacent nodes connected by edges with the same weight, i.e.,*

$$\rho(v) \;=\; \max_{b \in \mathbb{N}} \left|\{(v', v) \in E \,|\, w(v', v) = b\}\right| . \qquad (4)$$

We define the weight-degree of G to be the maximum weight-degree of any $v \in V$.

Note that for graphs with uniform edge weights, the weight-degree of G is the same as the maximum node degree. Intuitively, the (weight) degree of a graph is an important property for quantifying the sample complexity of the WGM because the degree determines how restrictive the bound on the number of components g is. In the extreme case of a complete graph, any support can be formed with only a single connected component (see Figure 1). Using Definitions 1 and 2, we now state our sparse recovery result (see Theorem 12 in Section 5 for a more general version):

Theorem 3. *Let $\beta \in \mathbb{R}^d$ be in the (G, s, g, B)-WGM. Then*

$$n = O\left(s\left(\log \rho(G) + \log \frac{B}{s}\right) + g \log \frac{d}{g}\right) \qquad (5)$$

i.i.d. Gaussian observations suffice to estimate β. More precisely, let $e \in \mathbb{R}^n$ be an arbitrary noise vector and let $y \in \mathbb{R}^n$ be defined as in Eq. 2 where X is an i.i.d. Gaussian matrix. Then we can efficiently find an estimate $\widehat{\beta}$ such that

$$\left\|\beta - \widehat{\beta}\right\| \;\leq\; C\|e\| , \qquad (6)$$

where C is a constant independent of all variables above.

Note that in the noiseless case ($e = 0$), we are guaranteed to recover β exactly. Moreover, our estimate $\widehat{\beta}$ is in a slightly enlarged WGM for any amount of noise, see Section 5. Our bound (5) can be instantiated to recover previous sample complexity results, e.g., the $n = O(s \log \frac{d}{s})$ bound for "standard" sparse recovery, which is tight (Do Ba et al., 2010).[1] For the image grid graph example in Figure 1, Equation (5) becomes $n = O(s + g \log \frac{d}{g})$, which matches the information-theoretic optimum $n = O(s)$ as long as the number of clusters is not too large, i.e., $g = O(s/\log d)$.[2]

[1]To be precise, encoding s-sparsity with a complete graph as in Figure 1 gives a bound of $n = O(s \log d)$. To match the $\log \frac{d}{s}$ term, we can encode s-sparsity as $g = s$ clusters of size one in a fully disconnected graph with no edges.

[2]Optimality directly follows from a simple dimensionality argument: even if the s-sparse support of the vector β is known, recovering the unknown coefficients requires solving a linear system with s unknowns uniquely. For this, we need at least s linear equations, i.e., s observations.

2.3. Efficient projection into the WGM

The algorithmic core of our sparsity framework is a **computationally efficient** procedure for projecting arbitrary vectors into the WGM. More precisely, the model-projection problem is the following: given a vector $b \in \mathbb{R}^d$ and a WGM $\mathcal{M}$, find the best approximation to b in $\mathcal{M}$, i.e.,

$$P_{\mathcal{M}}(b) \;=\; \arg\min_{b' \in \mathcal{M}} \|b - b'\| . \qquad (7)$$

If such a model-projection algorithm is available, one can instantiate the framework of (Baraniuk et al., 2010) in order to get an algorithm for sparse recovery with the respective sparsity model.[3] However, solving Problem (7) exactly is NP-hard for the WGM due to a reduction from the classical Steiner tree problem (Karp, 1972). To circumvent this hardness result, we use the *approximation-tolerant* framework of (Hegde et al., 2014a). Instead of solving (7) exactly, the framework requires *two* algorithms with the following complementary approximation guarantees.

Tail approximation: Find an $S \in \mathbb{M}$ such that

$$\|b - b_S\| \leq c_T \cdot \min_{S' \in \mathbb{M}} \|b - b_{S'}\| . \qquad (8)$$

Head approximation: Find an $S \in \mathbb{M}$ such that

$$\|b_S\| \geq c_H \cdot \max_{S' \in \mathbb{M}} \|b_{S'}\| . \qquad (9)$$

Here, $c_T > 1$ and $c_H < 1$ are arbitrary, fixed constants. Note that a head approximation guarantee does not imply a tail guarantee (and vice versa). In fact, stable recovery is not possible with only one type of approximate projection guarantee (Hegde et al., 2014a). We provide two algorithms for solving (8) and (9) (one per guarantee) which both run in *nearly-linear time*.

Our model-projection algorithms are based on a connection to the prize-collecting Steiner tree problem (PCST), which is a generalization of the classical Steiner tree problem. Instead of finding the cheapest way to connect *all* terminal nodes in a given weighted graph, we can instead omit some terminals from the solution and pay a specific price for each omitted node. The goal is to find a subtree with the optimal trade-off between the cost paid for edges used to connect a subset of the nodes and the price of the remaining, unconnected nodes (see Section 3 for a formal definition).

We make the following three main algorithmic contributions. Due to the wide applicability of the PCST problem, we believe that these algorithms can be of independent interest (see Section 7).

[3]Note that the framework does not supply general projection algorithms. Instead, the model-projection algorithms have to be designed from scratch for each model.

- We introduce a variant of the PCST problem in which the goal is to find a set of g trees instead of a single tree. We call this variant the prize-collecting Steiner forest (PCSF) problem and adapt the algorithm of (Goemans & Williamson, 1995) for this variant.
- We reduce the projection problems (8) and (9) to a small set of adaptively constructed PCSF instances.
- We give a nearly-linear time algorithm for the PCSF problem and hence also the model projection problem.

2.4. Improvements for existing sparsity models

Our results are directly applicable to several previously studied sparsity models that can be encoded with the WGM. Table 1 summarizes these results. In spite of its generality, our approach at least matches the sample complexity of prior work in all cases and actually offers an improvement for the EMD model. Moreover, our running time is always within a polylogarithmic factor of the best algorithm, even in the case of models with specialized solvers such as tree sparsity. For the EMD and cluster models, our algorithm is significantly faster than prior work and improves the time complexity by a polynomial factor. To complement these theoretical results, our experiments in Section 6 show that our algorithm is more than one order of magnitude faster than previous algorithms with provable guarantees and offers a better sample complexity in many cases.

2.5. Comparison to related work

In addition to the "point-solutions" for individual sparsity models outlined above, there has been a wide range of work on general frameworks for utilizing structure in sparse recovery. The approach most similar to ours is (Baraniuk et al., 2010), which gives a framework underlying many of the algorithms in Table 1. However, the framework has one important drawback: it does not come with a full recovery algorithm. Instead, the authors only give a recovery scheme that assumes the existence of a model-projection algorithm satisfying (7). Such an algorithm must be constructed from scratch for each model, and the techniques that have been used for various models so far are quite different. Our contribution can be seen as complementing the framework of (Baraniuk et al., 2010) with a nearly-linear time projection algorithm that is applicable to a wide range of sparsity structures. This answers a question raised by the authors of (Huang et al., 2011), who also give a framework for structured sparsity with a universal and complete recovery algorithm. Their framework is applicable to a wide range of sparsity models, but the corresponding algorithm is significantly slower than ours, both in theory ("Graph clusters" in Table 1) and in practice (see Section 6). Moreover, our recovery algorithm shows more robust performance across different shapes of graph clusters.

Both of the approaches mentioned above use iterative greedy algorithms for sparse recovery. There is also a large body of work on combining M-estimators with convex regularizers that induce structured sparsity, e.g., see the surveys (Bach et al., 2012a) and (Wainwright, 2014). The work closest to ours is (Jacob et al., 2009), which uses an overlapping group Lasso to enforce graph-structured sparsity (graph Lasso). In contrast to their approach, our algorithm gives more fine-grained control over the number of clusters in the graph. Moreover, our algorithm has better computational complexity, and to the best of our knowledge there are no formal results relating the graph structure to the sample complexity of the graph Lasso. Empririically, our algorithm recovers an unknown vector with graph structure faster and from fewer observations than the graph Lasso (see Section A in the supplementary material).

3. The prize-collecting Steiner forest problem

We now establish our connection between prize-collecting Steiner tree (PCST) problems and the weighted graph model. First, we formally define the PCST problem: Let $G = (V, E)$ be an undirected, weighted graph with edge costs $c : E \to \mathbb{R}_0^+$ and node prizes $\pi : V \to \mathbb{R}_0^+$. For a subset of edges $E' \subseteq E$, we write $c(E') = \sum_{e \in E'} c(e)$ and adopt the same convention for node subsets. Moreover, for a node subset $V' \subseteq V$, let $\overline{V'}$ be the complement $\overline{V'} = V \setminus V'$. Then the goal of the PCST problem is to find a subtree $T = (V', E')$ such that $c(E') + \pi(\overline{V'})$ is minimized. We sometimes write $c(T)$ and $\pi(\overline{T})$ if the node and edge sets are clear from context.

Similar to the classical Steiner tree problem, PCST is NP-hard. Most algorithms with provable approximation guarantees build on the seminal work of (Goemans & Williamson, 1995) (GW), who gave an efficient primal-dual algorithm with the following guarantee:

$$c(T) + 2\pi(\overline{T}) \;\leq\; 2 \min_{T' \text{ is a tree}} c(T') + \pi(\overline{T'}) \,. \qquad (10)$$

Note that the PCST problem already captures three important aspects of the WGM: (i) there is an underlying graph G, (ii) edges are weighted, and (iii) nodes have prizes. If we set the prizes to correspond to vector coefficients, i.e., $\pi(i) = b_i^2$, the term $\pi(\overline{T})$ in the PCST objective function becomes $\pi(\overline{T}) = \|b - b_T\|^2$, which matches the objective in the model-projection problems (8) and (9). However, there are two important differences. First, the objective in the PCST problem is to find a *single* tree T, while the WGM can contain supports defined by *multiple* connected components (if $g > 1$). Moreover, the PCST problem optimizes the trade-off $c(T) + \pi(\overline{T})$, but we are interested in minimizing $\|b - b_T\|$ subject to hard constraints on the support cardinality $|T|$ and the support cost $c(T)$ (the parameters s and B, respectively). In this section, we ad-

Table 1. Results of our sparsity framework applied to several sparsity models. In order to simplify the running time bounds, we assume that all coefficients are polynomially bounded in d, and that $s \leq d^{1/2-\mu}$ for some $\mu > 0$. For the graph cluster model, we consider the case of graphs with constant degree. The exponent τ depends on the degree of the graph and is always greater than 1. The parameters w and h are specific to the EMD model, see (Hegde et al., 2014a) for details. We always have $w \cdot h = d$ and $s \geq w$. Our sparsity framework improves on the sample complexity and running time of both the EMD and graph cluster models (bold entries).

Model	Reference	Best previous sample complexity	Our sample complexity	Best previous running time	Our running time
1D-cluster	(Cevher et al., 2009b)	$O(s + g \log \frac{d}{g})$	$O(s + g \log \frac{d}{g})$	$O(d \log^2 d)$	$O(d \log^4 d)$
Trees	(Hegde et al., 2014b)	$O(s)$	$O(s)$	$O(d \log^2 d)$	$O(d \log^4 d)$
EMD	(Hegde et al., 2014a)	$O(s \log \frac{B \log \frac{s}{w}}{s})$	$\boldsymbol{O(s \log \frac{B}{s})}$	$O(sh^2 B \log d)$	$\boldsymbol{O(wh^2 \log^4 d)}$
Graph clusters	(Huang et al., 2011)	$O(s + g \log d)$	$\boldsymbol{O(s + g \log \frac{d}{g})}$	$O(d^\tau)$	$\boldsymbol{O(d \log^4 d)}$

dress the first of these two issues; Section 4 then completes the connection between PCST and the WGM. We begin by defining the following variant of the PCST problem.

Definition 4 (The prize-collecting Steiner forest problem)**.** *Let $g \in \mathbb{N}$ be the target number of connected components. Then the goal of the prize-collecting Steiner forest (PCSF) problem is to find a subgraph $F = (V', E')$ with $\gamma(F) = g$ that minimizes $c(E') + \pi(\overline{V'})$.*

As defined in Section 2.1, $\gamma(F)$ is the number of connected components in the (sub-)graph F. To simplify notation in the rest of the paper, we say that a forest F is a g-forest if $\gamma(F) = g$. There is always an optimal solution for the PCSF problem which consists of g trees because removing edges cannot increase the objective value. This allows us to employ the PCSF problem for finding supports in the WGM that consist of several connected components. In order to give a computationally efficient algorithm for the PCSF variant, we utilize prior work for PCST: (i) To show correctness of our algorithm, we prove that the GW scheme for PCST can be adapted to our PCSF variant. (ii) To achieve a good time complexity, we show how to simulate the GW scheme in nearly-linear time.

3.1. The Goemans-Williamson (GW) scheme for PCSF

A useful view of the GW scheme is the "moat-growing" interpretation of (Jünger & Pulleyblank, 1995), which describes the algorithm as an iterative clustering method that constructs "moats" around every cluster. These moats are essentially the dual variables in the linear program of the GW scheme. Initially, every node forms its own active cluster with a moat of size 0. The moats around each active cluster then grow at a uniform rate until one of the following two events occurs:

Cluster deactivation When the sum of moats in a cluster reaches the sum of node prizes in that cluster, the cluster is deactivated.

Cluster merge When the sum of moats that are intersected by an edge e reaches the cost of e, the clusters at the two endpoints of e are merged and e is added to the current solution.

The moat-growing stage of the algorithm terminates when only a single active cluster remains. After that, the resulting set of edges is pruned in order to achieve a provable approximation ratio. We generalize the proof of (Feofiloff et al., 2010) and show that it is possible to extract more than one tree from the moat-growing phase as long as the trees come from different clusters. Our modification of GW terminates the moat-growing phase when exactly g active clusters remain, and we then apply the GW pruning algorithm to each resulting tree separately. This gives the following result.

Theorem 5. *There is an algorithm for the PCSF problem that returns a g-forest F such that*

$$c(F) + 2\pi(\overline{F}) \;\leq \min_{F' \subseteq G,\, \gamma(F') \leq g} 2c(F') + 2\pi(\overline{F'}) \,. \quad (11)$$

For $g = 1$, the theorem recovers the guarantee in (10). We defer the proof to Sec. D.1 of the supplementary material.

3.2. A fast algorithm for Goemans-Williamson

While the modified GW scheme produces good approximate solutions, it is not yet sufficient for a nearly-linear time algorithm: we still need an efficient way of simulating the moat-growing phase. There are two main difficulties: (i) The remaining "slack" amounts on edges can shrink at different rates depending on how many of the edge endpoints are in active clusters. (ii) A single cluster event (merge or deactivation) can change this rate for up to $\Theta(|V|)$ edges. In order to maintain edge events efficiently, we use the dynamic edge splitting approach introduced by (Cole et al., 2001). This technique essentially ensures that every edge always has at most one active endpoint, and hence its slack either shrinks at rate 1 or not

at all. However, edge splitting introduces additional edge events that do not directly lead to a cluster merge. While it is relatively straightforward to show that every such intermediate edge event halves the remaining amount of slack on an edge, we still need an overall bound on the number of intermediate edge events necessary to achieve a given precision. For this, we prove the following new result about the GW moat growing scheme.

Theorem 6. *Let all edge costs $c(e)$ and node prizes $\pi(v)$ be even integers. Then all finished moats produced by the GW scheme have integer sizes.*

In a nutshell, this theorem shows that one additional bit of precision is enough to track all events in the moat-growing phase accurately. We prove the theorem via induction over the events in the GW scheme, see Section D.2.2 for details. On the theoretical side, this result allows us to bound the overall running time of our algorithm for PCSF. Combined with suitable data structures, we can show the following:

Theorem 7. *Let α be the number of bits used to specify a single edge cost or node prize. Then there is an algorithm achieving the PCSF guarantee of Theorem 5 in time $O(\alpha \cdot |E| \log|V|)$.*

On the practical side, we complement Theorem 6 with a new adaptive edge splitting scheme that leads to a small number of intermediate edge events. Our experiments show that our scheme results in less than 3 events per edge on average (see Section D.3 in the supplementary material).

4. Sparse approximation with the WGM

In order to utilize the WGM in sparse recovery, we employ the framework of (Hegde et al., 2014a). As outlined in Section 2.3, the framework requires us to construct *two* approximation algorithms satisfying the head- and tail-approximation guarantees (8) and (9). We now give two such model-projection algorithms, building on our tools for PCSF developed in the previous section.

4.1. Tail-approximation algorithm

We can connect the PCSF objective to the WGM quantities by setting $\pi(i) = b_i^2$ and $c(e) = w(e) + 1$, which gives:

$$c(F) = w(F) + (|F| - g) \quad \text{and} \quad \pi(\overline{F}) = \|b - b_F\|^2 .$$

After multiplying the edge costs with a trade-off parameter λ, the PCSF objective $\lambda \cdot c(F) + \pi(\overline{F})$ essentially becomes a *Lagrangian relaxation* of the model-constrained optimization problem (8). We build our tail-approximation algorithm on top of this connection, starting with an algorithm for the "tail"-variant of the PCSF problem. By performing a binary search over the parameter λ (see Algorithm 1), we get a bicriterion guarantee for the final forest.

Algorithm 1 PCSF-TAIL

Input: G, c, π, g, cost-budget C, parameters ν and δ.
We write $c_\lambda(e) = \lambda \cdot c(e)$.
$\pi_{\min} \leftarrow \min_{\pi(i)>0} \pi(i)$, $\quad \lambda_0 \leftarrow \frac{\pi_{\min}}{2C}$
$F \leftarrow$ PCSF-GW$(G, c_{\lambda_0}, \pi, g)$
if $c(F) \le 2C$ and $\pi(\overline{F}) = 0$ **then return** F
$\lambda_r \leftarrow 0$, $\quad \lambda_l \leftarrow 3\pi(G)$, $\quad \varepsilon \leftarrow \frac{\pi_{\min}\delta}{C}$
while $\lambda_l - \lambda_r > \varepsilon$ **do**
  $\lambda_m \leftarrow (\lambda_l + \lambda_r)/2$
  $F \leftarrow$ PCSF-GW$(G, c_{\lambda_m}, \pi, g)$
  if $c(F) \ge C$ and $c(F) \le \nu C$ **then return** F
  if $c(F) > \gamma C$ **then** $\lambda_r \leftarrow \lambda_m$ **else** $\lambda_l \leftarrow \lambda_m$
end while
return $F \leftarrow$ PCSF-GW$(G, c_{\lambda_l}, \pi, g)$

Theorem 8. *Let $\nu > 2$ and $\delta > 0$. Then* PCSF-TAIL *returns a g-forest $F \subseteq G$ such that $c(F) \le \nu \cdot C$ and*

$$\pi(\overline{F}) \le \left(1 + \frac{2}{\nu - 2} + \delta\right) \min_{\gamma(F')=g, c(F') \le C} \pi(\overline{F'}) . \quad (12)$$

Theorem 8 does not give $c(F) \le C$ exactly, but the cost of the resulting forest is still guaranteed to be within a constant factor of C. The framework of (Hegde et al., 2014a) also applies to projections into such slightly larger models. As we will see in Section 5, this increase by a constant factor does not affect the sample complexity.

For the trade-off between support size and support weight, we also make use of approximation. By scalarizing the two constraints carefully, i.e., setting $c(e) = w(e) + \frac{B}{s}$, we get the following result. The proofs of Theorems 8 and 9 can be found in the supplementary material, Section C.1.

Theorem 9. *Let $\mathcal{M}$ be a (G, s, g, B)-WGM, let $b \in \mathbb{R}^d$, and let $\nu > 2$. Then there is an algorithm that returns a support $S \subseteq [d]$ in the $(G, 2\nu \cdot s + g, g, 2\nu \cdot B)$-WGM satisfying* (8) *with $c_T = \sqrt{1 + 3/(\nu - 2)}$. Moreover, the algorithm runs in time $O(|E| \log^3 d)$.*

4.2. Head-approximation algorithm

For our head-approximation algorithm, we also use the PCSF objective as a Lagrangian relaxation of the model-constraint problem (9). This time, we multiply the node prizes instead of the edge costs with a parameter λ. We perform a binary search similar to Alg. 1, but the final step of the algorithm requires an extra subroutine. At the end of the binary search, we are guaranteed to have a forest with good "density" $\frac{\pi(F)}{c(F)}$, but the good forest could correspond to either the lower bound λ_l or the upper bound λ_r. In the latter case, we have no bound on the cost of the corresponding forest F_r. However, it is always possible to extract a high-density sub-forest with bounded cost from F_r:

Algorithm 2 GRAPH-COSAMP

Input: y, X, G, s, g, B, number of iterations t.
$\widehat{\beta}_0 \leftarrow 0$
for $i \leftarrow 1, \ldots, t$ **do**
  $b \leftarrow X^T(y - X\widehat{\beta}_{i-1})$
  $S' \leftarrow \text{supp}(\widehat{\beta}_{i-1}) \cup \text{HEADAPPROX'}(b, G, s, g, B)$
  $z_{S'} \leftarrow X^{\dagger}_{S'} y, \quad z_{S'^C} \leftarrow 0$
  $S \leftarrow \text{TAILAPPROX}(z, G, s, g, B)$
  $\widehat{\beta}_i \leftarrow z_S$
end for
return $\widehat{\beta} \leftarrow \widehat{\beta}_i$

Lemma 10. *Let T be a tree and $C' \leq c(T)$. Then there is a subtree $T' \subseteq T$ such that $c(T') \leq C'$ and $\pi(T') \geq \frac{C'}{6} \cdot \frac{\pi(T)}{c(T)}$. Moreover, we can find such a subtree T' in linear time.*

The algorithm first converts the tree into a list of nodes corresponding to a tour through T. Then we can extract a good sub-tree by either returning a single, high-prize node or sliding a variable-size window across the list of nodes. See Section C.2 in the supplementary material for details. Combining these components, we get a head-approximation algorithm with the following properties.

Theorem 11. *Let $\mathcal{M}$ be a (G, s, g, B)-WGM and let $b \in \mathbb{R}^d$. Then there is an algorithm that returns a support $S \subseteq [d]$ in the $(G, 2s+g, g, 2B)$-WGM satisfying* (9) *with $c_H = \sqrt{1/14}$. The algorithm runs in time $O(|E| \log^3 d)$.*

5. Application in sparse recovery

We now instantiate the framework of (Hegde et al., 2014a) to give a sparse recovery algorithm using the WGM. The resulting algorithm (see Alg. 2) is a variant of CoSaMP (Needell & Tropp, 2009) and uses the head- and tail-approximation algorithms instead of the hard thresholding operators.[4] In order to state the corresponding recovery guarantee in full generality, we briefly review the definition of the (model-) restricted isometry property (RIP) (Candès & Tao, 2005; Baraniuk et al., 2010). We say that a matrix X satisfies the $(\mathcal{M}, \delta)$-model-RIP if for all $\beta \in \mathcal{M}$:

$$(1-\delta) \cdot \|\beta\|^2 \leq \|X\beta\|^2 \leq (1+\delta) \cdot \|\beta\|^2 . \qquad (13)$$

Theorem 12. *Let $\beta \in \mathbb{R}^d$ be in the (G, s, g, B)-WGM $\mathcal{M}$ and let $X \in \mathbb{R}^{n \times d}$ be a matrix satisfying the model-RIP for a $(G, c_1 s, g, c_2 B)$-WGM and a fixed constant δ, where c_1 and c_2 are also fixed constants. Moreover, let $e \in \mathbb{R}^n$ be an arbitrary noise vector and let $y \in \mathbb{R}^n$ be defined as in* (2)*. Then* GRAPH-COSAMP *returns a $\widehat{\beta}$ in the $(G, 5s, g, 5B)$-WGM such that $\left\|\beta - \widehat{\beta}\right\| \leq c_3 \|e\|$, where c_3 is a fixed constant. Moreover,* GRAPH-COSAMP *runs in time*

$$O\left((T_X + |E| \log^3 d) \log \frac{\|\beta\|}{\|e\|} \right) ,$$

where T_X is the time complexity of a matrix-vector multiplication with X.

In order to establish sample complexity bounds for concrete matrix ensembles (e.g., random Gaussian matrices as in Theorem 3), we use a result of (Baraniuk et al., 2010) that relates the sample complexity of sub-Gaussian matrix ensembles to the size of the model, i.e., the quantity $|\mathbb{M}|$. More precisely, $n = O(s + \log|\mathbb{M}|)$ rows / observations suffice for such matrices to satisfy the model-RIP for $\mathcal{M}$ and a fixed constant δ. For the WGM, we use a counting argument to bound $|\mathbb{M}|$ (see Section B in the supplementary material). Together with Theorem 12, the following theorem establishes Theorem 3 from Section 2.2.

Theorem 13. *Let $\mathbb{M}$ be the set of supports in the (G, s, g, B)-WGM. Then*

$$\log|\mathbb{M}| \;=\; O\left(s\left(\log \rho(G) + \log \frac{B}{s} \right) + g \log \frac{d}{g} \right) .$$

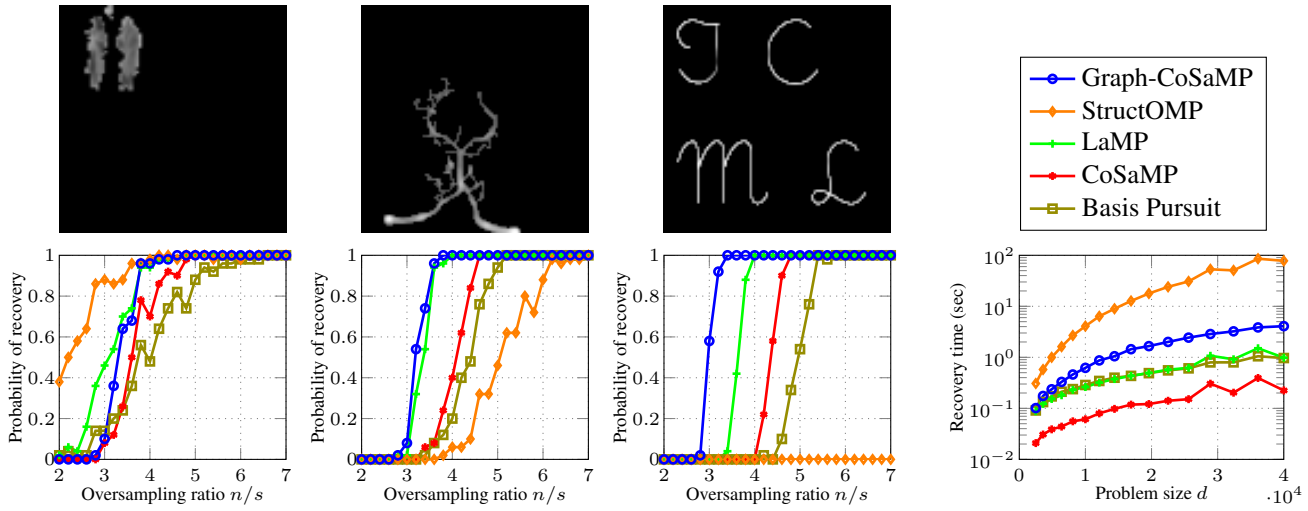
Next, we turn our attention to the running time of GRAPH-COSAMP. Since our model-projection algorithms run in nearly-linear time, the matrix-vector products involving X can become the bottleneck in the overall time complexity:[5] for a dense Gaussian matrix, we have $T_X = \Omega(sd)$, which would dominate the overall running time. If we can control the design matrix (as is often the case in compressive sensing), we can use the construction of (Hegde et al., 2014b) to get a sample-optimal matrix with nearly-linear T_X in the regime of $s \leq d^{1/2-\mu}, \mu > 0$. Such a matrix then gives an algorithm with nearly-linear running time. Note that the bound on s is only a theoretical restriction in this construction: as our experiments show, a partial Fourier matrix empirically performs well for significantly larger values of s.

6. Experiments

We focus on the performance of our algorithm Graph-CoSaMP for the task of recovering 2D data with clustered sparsity. Multiple methods have been proposed for this problem, and our theoretical analysis shows that our algorithm should improve upon the state of the art (see Table 1). We compare our results to StructOMP (Huang et al., 2011) and the heuristic Lattice Matching Pursuit (LaMP) (Cevher et al., 2009a). The implementations were supplied by the

[4] Strictly speaking, HEADAPPROX' is a "boosted" version of the head-approximation algorithm developed here. See (Hegde et al., 2014a) for details.

[5] It is not necessary to compute a full pseudo-inverse $X^{\dagger}$. See (Needell & Tropp, 2009) for details.



(a) Background-subtracted image (b) Angiogram (c) Text (d) Running times

Figure 2. Sparse recovery experiments. The images in the top row are the original images β. In the regime where the algorithms recover with high probability, the estimates $\widehat{\beta}$ are essentially identical to the original images. Our algorithm Graph-CoSaMP achieves consistently good recovery performance and offers the best sample complexity for images (b) and (c). Moreover, our algorithm is about 20 times faster than StructOMP, the other method with provable guarantees for the image cluster model.

authors and we used the default parameter settings. Moreover, we ran two common recovery algorithms for "standard" s-sparsity: Basis Pursuit (Candès et al., 2006) and CoSaMP (Needell & Tropp, 2009).

We follow a standard evaluation procedure for sparse recovery / compressive sensing: we record n observations $y = X\beta$ of the (vectorized) image $\beta \in \mathbb{R}^d$ using a subsampled Fourier matrix X. We assume that all algorithms possess prior knowledge of the sparsity s and the number of connected-components g in the true support of the image β. We declare a trial successful if the squared ℓ_2-norm of the recovery error is at most 5% of the squared ℓ_2-norm of the original vector β. The probability of successful recovery is then estimated by averaging over 50 trials. We perform several experiments with varying oversampling ratios n/s and three different images. See Section A in the supplementary material for a description of the dataset, experiments with noise, and a comparison with the graph Lasso.

Figure 2 demonstrates that Graph-CoSaMP yields consistently competitive phase transitions and exhibits the best sample complexity for images with "long" connected clusters, such as the angiogram image (b) and the text image (c). While StructOMP performs well on "blob"-like images such as the background-subtracted image (a), its performance is poor in our other test cases. For example, it can successfully recover the text image only for oversampling ratios $n/s > 15$. Note that the performance of Graph-CoSaMP is very consistent: in all three examples, the phase transition occurs between oversampling ratios 3 and 4. Other methods show significantly more variability.

We also investigate the computational efficiency of Graph-CoSaMP. We consider resized versions of the angiogram image and record $n = 6s$ observations for each image size d. Figure 2(d) displays the recovery times (averaged over 50 trials) as a function of d. We observe that the runtime of Graph-CoSaMP scales nearly linearly with d, comparable to the conventional sparse recovery methods. Moreover, Graph-CoSaMP is about $20\times$ faster than StructOMP.

7. Further applications

We expect our algorithms to be useful beyond sparse recovery and now briefly describe two promising applications.

Seismic feature extraction In (Schmidt et al., 2015), the authors use Steiner tree methods for a seismic feature extraction task. Our new algorithms for PCSF give a principled way of choosing tuning parameters for their proposed optimization problem.. Moreover, our fast algorithms for PCSF can speed-up their method.

Event detection in social networks (Rozenshtein et al., 2014) introduce a method for event detection in social networks based on the PCST problem. Their method performs well but produces spurious results in the presence of multiple disconnected events because their PCST algorithm produces only a *single* tree instead of a forest. Our new algorithm for PCSF gives exact control over the number of trees in the solution and hence directly addresses this issue. Furthermore, the authors quote a running time of $O(|V|^2 \log|V|)$ for their GW scheme, so our nearly-linear time algorithm allows their method to scale to larger data.

Acknowledgements

We thank Jayadev Acharya, Stefanie Jegelka, Youssef Mroueh, Devavrat Shah, and the anonymous reviewers for many helpful comments on earlier versions of this paper. This work was supported by grants from the MITEI-Shell program, the MADALGO center, and the Simons Investigator Award.

References

Bach, Francis, Jenatton, Rodolphe, Mairal, Julien, and Obozinski, Guillaume. Optimization with sparsity-inducing penalties. *Foundations and Trends in Machine Learning*, 4(1):1–106, 2012a.

Bach, Francis, Jenatton, Rodolphe, Mairal, Julien, and Obozinski, Guillaume. Structured sparsity through convex optimization. *Statistical Science*, 27(4):450–468, 11 2012b.

Baraniuk, Richard G., Cevher, Volkan, Duarte, Marco F., and Hegde, Chinmay. Model-based compressive sensing. *IEEE Transactions on Information Theory*, 56(4): 1982–2001, 2010.

Bateni, MohammadHossein, Chekuri, Chandra, Ene, Alina, Hajiaghayi, Mohammad T., Korula, Nitish, and Marx, Daniel. Prize-collecting steiner problems on planar graphs. In *Proceedings of the Twenty-second Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 1028–1049, 2011.

Bi, Wei and Kwok, James T. Multi-label classification on tree- and DAG-structured hierarchies. In *Proceedings of the 28th International Conference on Machine Learning (ICML)*, pp. 17–24, 2011.

Candès, Emmanuel J. and Tao, Terence. Decoding by linear programming. *IEEE Transactions on Information Theory*, 51(12):4203–4215, 2005.

Candès, Emmanuel J., Romberg, Justin, and Tao, Terence. Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52(2):489–509, 2006.

Cevher, Volkan, Duarte, Marco F., Hegde, Chinmay, and Baraniuk, Richard. Sparse signal recovery using markov random fields. In *Advances in Neural Information Processing Systems 21 (NIPS)*, pp. 257–264. 2009a.

Cevher, Volkan, Indyk, Piotr, Hegde, Chinmay, and Baraniuk, Richard G. Recovery of clustered sparse signals from compressive measurements. In *International Conference on Sampling Theory and Applications (SAMPTA)*, 2009b.

Cole, Richard, Hariharan, Ramesh, Lewenstein, Moshe, and Porat, Ely. A faster implementation of the Goemans-Williamson clustering algorithm. In *Proceedings of the Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 17–25, 2001.

Do Ba, Khanh, Indyk, Piotr, Price, Eric, and Woodruff, David P. Lower bounds for sparse recovery. In *Proceedings of the 21st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 1190–1197, 2010.

Duarte, Marco F. and Eldar, Yonina C. Structured compressed sensing: From theory to applications. *IEEE Transactions on Signal Processing*, 59(9):4053–4085, 2011.

Eisenstat, David, Klein, Philip, and Mathieu, Claire. An efficient polynomial-time approximation scheme for steiner forest in planar graphs. In *Proceedings of the Twenty-third Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 626–638, 2012.

El Halabi, Marwa and Cevher, Volkan. A totally unimodular view of structured sparsity. In *Proceedings of the 18th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2015.

Feofiloff, Paulo, Fernandes, Cristina G., Ferreira, Carlos E., and de Pina, José Coelho. A note on Johnson, Minkoff and Phillips' algorithm for the prize-collecting Steiner tree problem. *Computing Research Repository (CoRR)*, abs/1004.1437, 2010.

Goemans, Michel X. and Williamson, David P. A general approximation technique for constrained forest problems. *SIAM Journal on Computing*, 24(2):296–317, 1995.

He, Lihan and Carin, Lawrence. Exploiting structure in wavelet-based bayesian compressive sensing. *IEEE Transactions on Signal Processing*, 57(9):3488–3497, 2009.

Hegde, Chinmay, Indyk, Piotr, and Schmidt, Ludwig. Approximation algorithms for model-based compressive sensing. *Computing Research Repository (CoRR)*, abs/1406.1579, 2014a. Conference version appeard in the *Proceedings of the 25th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*.

Hegde, Chinmay, Indyk, Piotr, and Schmidt, Ludwig. Nearly linear-time model-based compressive sensing. In *Automata, Languages, and Programming (ICALP)*, volume 8572 of *Lecture Notes in Computer Science*, pp. 588–599. 2014b.

Huang, Junzhou, Zhang, Tong, and Metaxas, Dimitris. Learning with structured sparsity. *The Journal of Machine Learning Research*, 12:3371–3412, 2011.

Jacob, Laurent, Obozinski, Guillaume, and Vert, Jean-Philippe. Group Lasso with overlap and graph Lasso. In *Proceedings of the 26th Annual International Conference on Machine Learning (ICML)*, pp. 433–440, 2009.

Johnson, David S., Minkoff, Maria, and Phillips, Steven. The prize collecting Steiner tree problem: Theory and practice. In *Proceedings of the Eleventh Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 760–769, 2000.

Jünger, Michael and Pulleyblank, William R. New primal and dual matching heuristics. *Algorithmica*, 13(4):357–380, 1995.

Karp, Richard M. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, The IBM Research Symposia Series, pp. 85–103. 1972.

Kim, Seyoung and Xing, Eric P. Tree-guided group Lasso for multi-task regression with structured sparsity. In *Proceedings of the 27th International Conference on Machine Learning (ICML)*, pp. 543–550, 2010.

Mosci, Sofia, Villa, Silvia, Verri, Alessandro, and Rosasco, Lorenzo. A primal-dual algorithm for group sparse regularization with overlapping groups. In *Advances in Neural Information Processing Systems 23 (NIPS)*, pp. 2604–2612. 2010.

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

Negahban, Sahand N., Ravikumar, Pradeep, Wainwright, Martin J., and Yu, Bin. A unified framework for high-dimensional analysis of m-estimators with decomposable regularizers. *Statistical Science*, 27(4):538–557, 11 2012.

Rao, Nikhil S., Recht, Ben, and Nowak, Robert D. Universal measurement bounds for structured sparse signal recovery. In *Proceedings of the 15th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 22 of *JMLR Proceedings*, pp. 942–950, 2012.

Rozenshtein, Polina, Anagnostopoulos, Aris, Gionis, Aristides, and Tatti, Nikolaj. Event detection in activity networks. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 1176–1185, 2014.

Schmidt, Ludwig, Hegde, Chinmay, Indyk, Piotr, Lu, Ligang, Chi, Xingang, and Hohl, Detlef. Seismic feature extraction using Steiner tree methods. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015.

Simon, Noah, Friedman, Jerome, Hastie, Trevor, and Tibshirani, Robert. A sparse-group Lasso. *Journal of Computational and Graphical Statistics*, 22(2):231–245, 2013.

Wainwright, Martin J. Structured regularizers for high-dimensional problems: Statistical and computational issues. *Annual Review of Statistics and Its Application*, 1 (1):233–253, 2014.

Yuan, Ming and Lin, Yi. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(1):49–67, 2006.