Recursive Recovery of Sparse Signal Sequences from Compressive Measurements: A Review

Namrata Vaswani and Jinchun Zhan

***Abstract*—In this article, we review the literature on the design and analysis of recursive algorithms for reconstructing a time sequence of sparse signals from compressive measurements. The signals are assumed to be sparse in some transform domain or in some dictionary. Their sparsity patterns can change with time, although in many practical applications, the changes are gradual. An important class of applications where this problem occurs is dynamic projection imaging, e.g., dynamic magnetic resonance imaging for real-time medical applications such as interventional radiology, or dynamic computed tomography.**

I. Introduction

In this paper, we review the literature on the design and analysis of recursive algorithms for causally reconstructing a time sequence of sparse signals from a limited number of linear measurements (compressive measurements). The signals are assumed to be sparse, or approximately sparse, in some transform domain referred to as the sparsity basis, and their sparsity pattern (support set of the sparsity basis coefficients' vector) can change with time. The signals could also be sparse in a known dictionary and everything described in this article will apply. The term "recursive algorithms", refers to algorithms that only use past signals' estimates and the current measurements' vector to get the current signal's estimate [1].

The problem of recovering a sparse signal from a small number of its linear measurements has been studied for a long time. In the signal processing literature, the works of Mallat and Zhang [1] (matching pursuit), Chen and Donoho (basis pursuit) [2], Feng and Bresler [3], [4] (spectrum blind recovery of multi-band signals), Gorodnistky and Rao [5], [6] (a reweighted minimum 2-norm algorithm for sparse recovery) and Wipf and Rao [7] (sparse Bayesian learning for sparse recovery) were among the first works on this topic. The papers by Candes, Romberg, Tao and by Donoho [8]–[10] introduced the compressive sensing (CS) problem. The idea of CS is to compressively sense signals that are sparse in some known domain and then use sparse recovery techniques to recover them. The most important contribution of [8]–[10] was that they provided practically meaningful conditions for exact sparse recovery using basis pursuit which is a commonly used sparse recovery technique. In the last decade since these papers appeared, this problem has received a lot of attention. Often the terms "sparse recovery" and "CS" are used interchangeably. We also do this in this article. The CS problem occurs in a large class of applications. Examples include magnetic resonance imaging (MRI), computed tomography (CT), and various other projection imaging applications, where measurements are acquired one linear projection at a time. The ability to reconstruct from fewer measurements is useful for these applications since it means that less time is needed for completing an MR or a CT scan.

Consider the dynamic CS problem, i.e., the problem of recovering a time sequence of sparse signals. Most of the initial solutions for this problem consisted of batch algorithms. These can be split into two categories depending on what assumption they use on the time sequence. The first category is batch algorithms that solve the multiple measurements' vectors (MMV) problem. These use the assumption that the support set of the sparse signals *does not* change with time [11]–[15]. The second category is batch algorithms that treat the entire time sequence as a single sparse spatiotemporal signal by assuming Fourier sparsity along the time axis [16]–[18]. However, in many situations neither of these assumptions is valid. Moreover, even when these are valid assumptions, batch algorithms are offline, slower, and their memory requirement increases linearly with the sequence length. In this work, we focus on recursive algorithms for solving the dynamic CS problem. Their computational and storage complexity is much lower than that of the batch techniques and is, in fact, comparable to that of simple-CS solutions. At the same time, the number of measurements required by these algorithms for exact or accurate recovery is significantly smaller than what simple-CS solutions need, as long as an accurate estimate of the first sparse signal is available. Hereafter "simple-CS" refers to dynamic CS solutions that recover each sparse signal in the sequence independently without using any information from past or future frames.

The above problem occurs in multiple applications. In fact, it occurs in virtually every application where CS is useful and there exists a sequence of sparse signals. For a comprehensive list of CS applications, see [19], [20]. One common class of applications is dynamic MRI or dynamic CT. We show an example of a vocal tract (larynx) MR image sequence in Fig. 1. Notice that the images are piecewise smooth and hence wavelet sparse. As shown in Fig. 2, their sparsity pattern in the wavelet transform domain changes with time, but the changes are slow. We discuss this and other applications in Sec. III-B.

A. Paper Organization

The rest of this article is organized as follows. We summarize the notation and provide a short overview of some of the approaches for solving the static sparse recovery or

N. Vaswani and J. Zhan are with the Dept. of Electrical and Computer Engineering, Iowa State University, Ames, IA, USA. Email: namrata@iastate.edu.
This work was supported by NSF grant CCF-0917015 and CCF-1117125.

[1] In some other works, "recursive estimation" is also used to refer to recursively estimating a single signal as more of its measurements come in. This should not be confused with our definition.

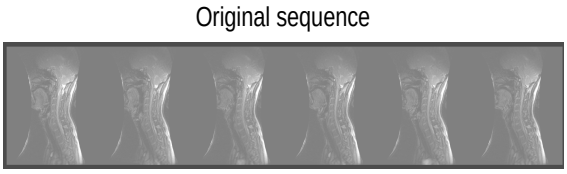
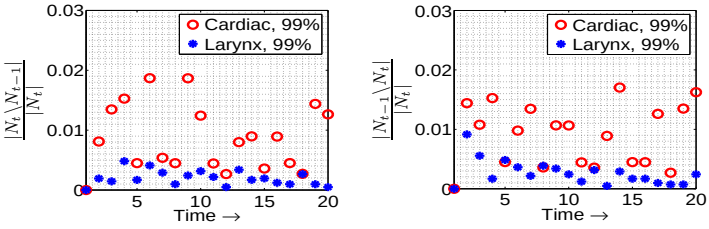


Fig. 1. We show a dynamic MRI sequence of the vocal tract (larynx) that was acquired when the person was speaking a vowel.



(a) slow support changes (adds) (b) slow support changes (removals)

Fig. 2. In these figures, $\mathcal{N}_t$ refers to the 99%-energy support of the 2D discrete wavelet transform (two-level Daubechies-4 2D DWT) of the larynx sequence shown in Fig. 1 and of a cardiac sequence. The 99%-energy support size, $|\mathcal{N}_t|$, varied between 6-7% of the image size in both cases. We plot the number of additions (top) and the number of removals (bottom) as a fraction of the support size. *Notice that all support change sizes are less than 2% of the support size.*

CS problem in Section II. Next, in Section III, we define the recursive dynamic CS problem, discuss its applications and explain why new approaches are needed to solve it. We split the discussion of the proposed solutions into three sections. In Section IV, we discuss algorithms that only exploit slow support change. Under this assumption and if the first signal can be recovered accurately, our problem can be reformulated as one of sparse recovery with partial support knowledge. We describe solutions to this reformulated problem and their guarantees (which is also of independent interest). In Section V, we discuss algorithms that also exploit slow signal value change, by again reformulating a static problem first. In Section VI, we first briefly explain how the solutions from the previous two sections apply to the recursive dynamic CS problem. Next, we describe solutions that were designed in the recursive dynamic CS context. Algorithm pseudo-code and the key ideas of how to set parameters automatically are also given. Error stability over time results are also discussed here. In Section VII, we explain tracking-based and adaptive-filtering-based solutions and their pros and cons compared with previously described solutions. Numerical experiments comparing the various approaches both on simulated data and on dynamic MRI sequences are discussed in Section VIII. In Section IX, we describe work on related problems and how it can be used in conjunction with recursive dynamic CS. Open questions for future work are also summarized. We conclude in Section X.

II. Notation and Background

A. Notation

We define $[1, m] := [1, 2, \ldots m]$. For a set $\mathcal{T}$, we use $\mathcal{T}^c$ to denote the complement of $\mathcal{T}$ w.r.t. $[1, m]$, i.e., $\mathcal{T}^c := \{i \in [1, m] : i \notin \mathcal{T}\}$. The notation $|\mathcal{T}|$ denotes the size (cardinality) of the set $\mathcal{T}$. The set operation $\setminus$ denotes set set difference, i.e., for two sets $\mathcal{T}_1, \mathcal{T}_2$, $\mathcal{T}_1 \setminus \mathcal{T}_2 := \mathcal{T}_1 \cap \mathcal{T}_2^c$. We use $\emptyset$ to denote the empty set. For a vector, v, and a set, $\mathcal{T}$, $v_{\mathcal{T}}$ denotes the $|\mathcal{T}|$ length sub-vector containing the elements of v corresponding to the indices in the set $\mathcal{T}$. Also, $\|v\|_k$ denotes the ℓ_k norm of a vector v. If just $\|v\|$ is used, it refers to $\|v\|_2$. When $k = 0$, $\|v\|_0$ counts the number of nonzero elements in the vector v. This is referred to the ℓ_0 norm even though it does not satisfy the properties of a norm. A vector v is sparse if it has many zero entries. We use $\text{supp}(v) := \{i : v_i \neq 0\}$ to denote the support set of a sparse vector v, i.e., the set of indices of its nonzero entries. For a matrix M, $\|M\|_k$ denotes its induced k-norm, while just $\|M\|$ refers to $\|M\|_2$. M' denotes the transpose of M and $M^\dagger$ denotes its Moore-Penrose pseudo-inverse. For a tall matrix, M, $M^\dagger := (M'M)^{-1}M'$. For a matrix A, $A_{\mathcal{T}}$ denotes the sub-matrix obtained by extracting the columns of A corresponding to the indices in $\mathcal{T}$. We use I to denote the identity matrix.

B. Sparse Recovery or Compressive Sensing (CS)

The goal of sparse recovery or CS is to reconstruct an m-length sparse signal, x, with support $\mathcal{N}$, from an n-length measurement vector, $y := Ax$ or from $y := Ax + w$, with $\|w\|_2 \leq \epsilon$ (noisy case) when A has more columns than rows, i.e., $n < m$. Consider the noise-free case. Let $s = |\mathcal{N}|$. It is easy to show that this problem is solved if we can find the sparsest vector satisfying $y = A\beta$, i.e., if we can solve

$$\min_{\beta} \|\beta\|_0 \text{ subject to } y = A\beta, \tag{1}$$

and if any set of $2s$ columns of A are linearly independent [9]. However doing this is impractical since it requires a combinatorial search. The complexity of solving (1) is exponential in the support size. In the last two decades, many practical (polynomial complexity) approaches have been developed. Most of the classical approaches can be split as follows (a) convex relaxation approaches, (b) greedy algorithms, (c) iterative thresholding methods, and (d) sparse Bayesian learning (SBL) based methods. Besides these, there are many more solution approaches that have appeared more recently.

The convex relaxation approaches are also referred to as ℓ_1 minimization programs since these replace the ℓ_0 norm in (1) by the ℓ_1 norm which is the closest norm to ℓ_0 that is convex. Thus, in the noise-free case, one solves

$$\min_{\beta} \|\beta\|_1 \text{ subject to } y = A\beta \tag{2}$$

The above program is referred to as *basis pursuit (BP)* [2]. Clearly, it is a convex program. In fact it is easy to show that it is actually a linear program[2]. *Since this was the program analyzed in the papers that introduced the term Compressed Sensing (CS) [8]–[10], some later works wrongly refer to this program itself as "CS".* In the noisy case, the constraint is

[2]Let x^+ denote a sparse vector whose i-th index is zero wherever $x_i \leq 0$ and is equal to x_i otherwise. Similarly let x^- be a vector whose i-th index is zero wherever $x_i \geq 0$ and is equal to $-x_i$ otherwise. Then, clearly, $x = x^+ - x^-$ and $\|x\|_1 = \sum_{i=1}^m (x^+)_i + \sum_{i=1}^m (x^-)_i$. Thus (2) is equivalent to the linear program $\min_{\beta^+, \beta^-} \sum_{i=1}^m (\beta^+)_i + \sum_{i=1}^m (\beta^-)_i$ subject to $y = A(\beta^+ - \beta^-)$.

replaced by $\|y - A\beta\|_2 \leq \epsilon$ where ϵ is the bound on the ℓ_2 norm of the noise, i.e.,

$$\min_{\beta} \|\beta\|_1 \text{ subject to } \|y - A\beta\|_2 \leq \epsilon. \tag{3}$$

This is referred to as *BP-noisy*. In problems where the noise bound is not known, one can solve an unconstrained version of this problem by including the data term as a soft constraint (the "Lagrangian version"):

$$\min_{\beta} \gamma\|\beta\|_1 + 0.5\|y - A\beta\|_2^2. \tag{4}$$

The above is referred to as *BP denoising (BPDN)* [2]. BPDN solves an unconstrained convex optimization problem which is faster to solve than BP-noisy which has constraints. Of course, both are equivalent in the sense that for given ϵ, there exists a $\gamma(\epsilon)$ such that the solutions to both coincide. However it is hard to find such a mapping.

Another class of solutions for the CS problem consists of greedy algorithms. These get an estimate of the support of x in a greedy fashion. This is done by finding one, or a set of, indices of the columns of A that have the largest correlation with the measurement residual from the previous iteration. The first known greedy algorithms were Matching Pursuit [1] and later orthogonal matching pursuit (OMP) [21], [22]. MP and OMP add only one index at a time. Later algorithms such as subspace pursuit [23] and CoSaMP [24] add multiple indices at a time, but also include a step to delete indices. The greedy algorithms are designed assuming that the columns of A have unit ℓ_2 norm. However when this does not hold, it is easy to reformulate the problem in order to ensure this.

Remark 2.1 (matrices A without unit ℓ_2-norm columns): Any matrix can be converted into a matrix with unit ℓ_2-norm columns by right multiplying it with a diagonal matrix D that contains $\|A_i\|_2^{-1}$ as its entries. Here A_i is the i-th column of A. Thus, for any matrix A, $A_{\text{normalized}} = AD$. Whenever normalized columns of A are needed, one can rewrite $y = Ax$ as $y = ADD^{-1}x = A_{\text{normalized}}\tilde{x}$ and first recover $\tilde{x}$ from y and then obtain $x = D\tilde{x}$.

A third solution approach for CS is Iterative Hard Thresholding (IHT) [25], [26]. This is an iterative algorithm that proceeds by hard thresholding the current "estimate" of x to s largest elements. Let $H_s(a)$ denote the hard thresholding operator which zeroes out all but the s largest magnitude elements of the vector a. Let $\hat{x}^k$ denote the estimate of x at the k^{th} iteration. It proceeds as follows.

$$\hat{x}^0 = 0,\ \hat{x}^{i+1} = H_s(\hat{x}^i + A'(y - A\hat{x}^i)). \tag{5}$$

Another commonly used approach to solving the sparse recovery problem is sparse Bayesian learning (SBL) [7], [27]. In SBL, one models the sparse vector x as consisting of independent Gaussian components with zero mean and variances γ_i for the i-th component. The observation noise is assumed to be independent identically distributed (i.i.d.) Gaussian with zero mean and variance σ^2. SBL consists of an expectation maximization (EM)-type algorithm to estimate the hyper-parameters $\{\sigma^2, \gamma_1, \gamma_2, \ldots \gamma_m\}$ from the observation vector y using evidence maximization (type-II maximum likelihood). Since the true x is sparse, it can be argued that the estimates of a lot of the γ_i's will be zero or nearly zero (and can be zeroed out). Once the hyper-parameters are estimated, SBL computes the maximum a posteriori (MAP) estimate of x. This has a simple closed form expression under the assumed joint Gaussian model.

C. Restricted Isometry Property and Null Space Property

In this section we describe some of the properties introduced in recent work that are either sufficient or necessary and sufficient to ensure exact sparse recovery.

Definition 2.2: The *restricted isometry constant (RIC)*, $\delta_s(A)$, for a matrix A, is the smallest real number satisfying

$$(1 - \delta_s)\|b\|_2^2 \leq \|Ab\|_2^2 \leq (1 + \delta_s)\|b\|_2^2 \tag{6}$$

for all s-sparse vectors b [9]. A matrix A satisfies the *RIP of order s* if $\delta_s(A) < 1$.

It is easy to see that $(1 - \delta_s) \leq \|A_{\mathcal{T}}{}'A_{\mathcal{T}}\| \leq (1 + \delta_s)$, $\|(A_{\mathcal{T}}{}'A_{\mathcal{T}})^{-1}\| \leq 1/(1 - \delta_s)$ and $\|A_{\mathcal{T}}{}^{\dagger}\| \leq 1/\sqrt{(1 - \delta_s)}$ for sets $\mathcal{T}$ with $|\mathcal{T}| \leq s$.

Definition 2.3: The *restricted orthogonality constant (ROC)*, $\theta_{s,\tilde{s}}$, for a matrix A, is the smallest real number satisfying

$$|b_1{}'A_{\mathcal{T}_1}{}'A_{\mathcal{T}_2}b_2| \leq \theta_{s,\tilde{s}}\ \|b_1\|_2\ \|b_2\|_2 \tag{7}$$

for all disjoint sets $\mathcal{T}_1, T_2 \subseteq [1, m]$ with $|\mathcal{T}_1| \leq s$, $|\mathcal{T}_2| \leq \tilde{s}$, $s + \tilde{s} \leq m$, and for all vectors b_1, b_2 of length $|\mathcal{T}_1|$, $|\mathcal{T}_2|$ [9]. It is not hard to show that $\|A_{\mathcal{T}_1}{}'A_{\mathcal{T}_2}\| \leq \theta_{s,\tilde{s}}$ [28] and that $\theta_{s,\tilde{s}} \leq \delta_{s+\tilde{s}}$ [9].

The following result was proved in [29].

Theorem 2.4 (Exact recovery and error bound for BP and BP-noisy): Denote the solution of BP, (2), by $\hat{x}$. In the noise-free case, i.e., when $y := Ax$, if $\delta_s(A) < \sqrt{2} - 1$, then $\hat{x} = x$ (BP achieves exact recovery). In the noisy case, i.e., when $y := Ax + w$ with $\|w\|_2 \leq \epsilon$, if $\delta_{2s}(A) < 0.207$, then

$$\|x - \hat{x}\|_2 \leq C_1(2s)\epsilon \leq 7.50\epsilon \text{ where } C_1(k) := \frac{4\sqrt{1 + \delta_k}}{1 - 2\delta_k}$$

where $\hat{x}$ is the the solution of BP-noisy, (3).

With high probability (whp), random Gaussian matrices and various other random matrix ensembles satisfy the RIP of order s whenever the number of measurements n is of the order of $s \log m$ and m is large enough [9].

The null space property (NSP) is another property used to prove results for exact sparse recovery [30], [31]. NSP ensures that every vector v in the null space of A is not too sparse. NSP is known to be a necessary and sufficient condition for exact recovery of s-sparse vectors [30], [31].

Definition 2.5: A matrix A satisfies the *null space property (NSP)* of order s if, for any vector v in the null space of A,

$$\|v_S\|_1 < 0.5\|v\|_1, \text{ for all sets } S \text{ with } |S| \leq s$$

III. The Problem, Applications and Motivation

A. Problem Definition

Let t denote the discrete time index. We would like to recover a sparse n-length vector sequence $\{x_t\}$ from under-sampled and possibly noisy measurements $\{y_t\}$ satisfying

$$y_t := A_t x_t + w_t,\ \|w_t\|_2 \leq \epsilon, \tag{8}$$

where $A_t := H_t\Phi$ is an $n_t \times m$ matrix with $n_t < m$ and w_t is bounded noise. Here H_t is the measurement matrix and Φ is an orthonormal matrix for the sparsity basis. Alternatively, it can be a dictionary matrix. In the above formulation, $z_t := \Phi x_t$ is actually the signal (or image arranged as a 1D vector) whereas x_t is its representation in the sparsity basis or dictionary Φ. For example, in MRI of wavelet sparse images, H_t is a partial Fourier matrix and Φ is the matrix corresponding to the inverse 2D discrete wavelet transform (DWT).

We use $\mathcal{N}_t$ to denote the support set of x_t, i.e.,

$$\mathcal{N}_t := \text{supp}(x_t) = \{i : (x_t)_i \neq 0\}.$$

When we say x_t is sparse, it means that $|\mathcal{N}_t| \ll m$. Let $\hat{x}_t$ denote an estimate of x_t. The goal is to recursively reconstruct x_t from $y_0, y_1, \ldots y_t$, i.e., use only $\hat{x}_0, \hat{x}_1, \ldots, \hat{x}_{t-1}$ and y_t for reconstructing x_t. It is assumed that the first signal, x_0, can be accurately recovered from y_0. The simplest way to ensure this is by using more measurements at $t = 0$ and using a simple-CS solution to recover x_0. Other possible approaches, such as using other prior knowledge or using a batch CS approach for an initial short sequence, are described in Section VI-A.

In order to solve the above problem, one can leverage the practically valid assumption of slow support (sparsity pattern) change [28], [32], [33]:

$$|\mathcal{N}_t \setminus \mathcal{N}_{t-1}| \approx |\mathcal{N}_{t-1} \setminus \mathcal{N}_t| \ll |\mathcal{N}_t|. \tag{9}$$

Notice from Fig. 2 that this is valid for dynamic MRI sequences. A second assumption that can also be exploited is that of slow signal value change:

$$\|(x_t - x_{t-1})_{\mathcal{N}_{t-1} \cup \mathcal{N}_t}\|_2 \ll \|(x_t)_{\mathcal{N}_{t-1} \cup \mathcal{N}_t}\|_2. \tag{10}$$

This, of course, is a commonly used assumption in almost all past work on tracking algorithms as well as in work on adaptive filtering algorithms. Notice that we can also write the above assumption as $\|x_t - x_{t-1}\|_2 \ll \|x_t\|_2$. This is true too since for $i \notin \mathcal{N}_{t-1} \cup \mathcal{N}_t$, $(x_t - x_{t-1})_i = 0$.

Henceforth we refer to the above problem as the *recursive dynamic CS* problem. Also, as noted in Sec. I, *"simple-CS solutions"* refers to dynamic CS solutions that recover each sparse signal in the sequence independently without using any information from past or future frames.

B. Applications

An important application where the above problem occurs is undersampled dynamic MRI for applications such as interventional radiology [34], [35], MRI-guided surgery and functional MRI based tracking of brain activations in response to changing stimuli [36], [37]. Since MR data is acquired one Fourier projection at a time, the ability to accurately reconstruct using fewer measurements directly translates into reduced scan times. Shorter scan times along with online reconstruction can potentially enable real-time[3] imaging of fast changing physiological phenomena, thus making many interventional MRI applications such as MRI-guided surgery feasible in the future [34]. As explained in [34], these are currently not feasible due to the extremely slow image acquisition speed of MR systems. To understand the MRI application, assume that all images are rearranged as 1D vectors. The MRI measurement vector at time t, y_t, satisfies $y_t = H_t z_t + w_t$ where z_t is the $m_1 \times m_2$ image at time t arranged as an $m = m_1 m_2$ length vector and

$$H_t = {I_{\mathcal{O}_t}}'(F_{m_1} \otimes F_{m_2}).$$

Here $\mathcal{O}_t$ is the set of indices of the observed discrete frequencies at time t, F_m is the m-point discrete Fourier transform (DFT) matrix and $\otimes$ denotes Kronecker product. Observe that the notation ${I_{\mathcal{O}_t}}'M$ creates a sub-matrix consisting of the rows of M with indices in the set $\mathcal{O}_t$.

Cross-sectional images of the brain, heart, larynx or other human organs are usually piecewise smooth, e.g., see Fig. 1, and thus well modeled as being sparse in the wavelet domain. Hence, in this case, Φ is the inverse 2D discrete wavelet transform (DWT) matrix. If W_m is the inverse DWT matrix corresponding to a chosen 1D wavelet, e.g. Daubechies-4, then

$$\Phi = W_{m_1} \otimes W_{m_2}.$$

Slow support change of the wavelet coefficients vector, $x_t := \Phi^{-1} z_t$, is verified in Fig. 2. For large-sized images, the matrix H_t or the matrix Φ as expressed above become too big to store in memory. Thus, in practice, one does not compute the DFT or DWT using matrix-vector multiplies but directly as explained in Sec. VIII-C. This ensures that one never needs to store H_t or Φ in memory, only $\mathcal{O}_t$ needs to be stored.

Another potential application of recursive dynamic CS solutions is single-pixel camera (SPC) based video imaging. The SPC is not a very practical tool for optical imaging since much faster cameras exist, but it is expected to be useful for imaging applications where the imaging sensors are very expensive, e.g., for short-wave-infrared (SWIR) imaging [38]–[40] or for compressive depth acquisition [41]. In fact many companies including start-ups such as InView and large research labs such as MERL and Bell-Labs have built their own SPCs [42]. In this application, z_t is again the vectorized image of interest that can often be modeled as being wavelet sparse. The measurements are random-Gaussian or Rademacher, i.e., each element of H_t is either an independent and identically distributed (i.i.d.) Gaussian with zero mean and unit variance or is i.i.d. ± 1. This problem formulation is also used for CS-based (or dynamic CS based) online video compression/decompression. This can be useful in applications where the compression end needs to be implemented with minimal hardware (just matrix-vector multiplies) while significantly higher computational power is available for decompression, e.g., in a sensor network based sensing setup as in Cevher et al. [43] (this involved a camera network and a CS-based background-subtraction solution).

A completely different application is online denoising of image sequences that are sparsifiable in a known sparsity basis or dictionary. In fact, denoising was the original problem for which basis pursuit denoising (BPDN) was introduced [2]. Let Φ denote the given dictionary or sparsity basis. For example, one could let it be the inverse Discrete Cosine

[3]None of the solutions that we describe in this article are currently able to run in "real-time". The fastest method still needs about 5 seconds per frame of processing when implemented using MATLAB code on a standard desktop (see Table V).

Transform (DCT) matrix. Or, as suggested in [44], one can let $\Phi = [I\ D]$ where I is an identity matrix and D is the inverse DCT matrix. The DCT basis is a good sparsifying basis for textures in an image while the canonical basis (I) is a good sparsifying basis for edges [44]. For this problem, $H_t = I$ and so $y_t = z_t + w_t = \Phi x_t + w_t$ and x_t is the sparse vector at time t. Since image sequences are correlated over time, slow support change and slow signal value change are valid assumptions on x_t. A generalization of this problem is the dictionary learning or the sparsifying transform learning problem where the dictionary or sparsifying transform is also unknown [45], [46]. This is briefly discussed in Sec. IX.

Another application of recursive dynamic CS is speech/audio reconstruction from time-undersampled measurements. This was studied by Friedlander et al. [47]. In a traditional Analog-to-Digital converter, speech is uniformly sampled at 44.1kHz. With random undersampling and dynamic CS based reconstruction, one can sample speech at an average rate of $b * 44.1$kHz where b is the undersampling factor. In [47], $b = 0.25$ was used. Speech is broken down into segments of time duration $m\tau$ seconds where $\tau = 1/44100$ seconds. Then z_t corresponds to the time samples of the t-th speech segment uniformly sampled using τ as the sampling interval. The measurement vector for the t-th segment satisfies $y_t := I_{\mathcal{O}_t}{}' z_t$ where $\mathcal{O}_t$ contains $\lceil b * m \rceil$ uniformly randomly selected indices out of the set $\{1, 2, \dots, m\}$. The DCT is known to form a good sparsifying basis for speech/audio [47]. Thus, for this application, Φ corresponds to the inverse DCT matrix. As explained in [47], the support set corresponding to the largest DCT coefficients in adjacent blocks of speech does not change much from one block to the next and thus slow support change holds. Of course, once the z_t's are recovered, they can be converted to speech using the usual analog interpolation filter.

Besides the above, the recursive dynamic CS problem has also been explored for various other applications such as dynamic optical coherence tomography (OCT) [48]; dynamic spectrum sensing in cognitive radios, e.g., [49]; and sparse channel estimation and data detection in orthogonal frequency division multiplexing (OFDM) systems [50].

Lastly, consider a solution approach to recursive robust principal components' analysis (RPCA). In [51], RPCA was defined as a problem of separating a low-rank matrix L and a sparse matrix X from the data matrix $Y := X + L$ [51][4]. Recursive RPCA is then the problem of recovering a time sequence of sparse vectors, x_t, and vectors lying in a low-dimensional subspace, ℓ_t, from $y_t := x_t + \ell_t$ in a recursive fashion, starting with an accurate knowledge of the subspace from which ℓ_0 was generated. A solution approach for recursive RPCA, called Recursive Projected CS (ReProCS), was introduced in recent work [52]–[54]. As we explain, one of two key steps of ReProCS involves solving a recursive dynamic CS problem. ReProCS assumes that the subspace from which ℓ_t is generated either remains fixed or changes slowly over time, and any set of basis vectors for this subspace are dense (non-sparse) vectors. At time t, suppose that an accurate estimate of the subspace from which ℓ_{t-1} is generated is available. Let $\hat{P}_{t-1}$ be a matrix containing its basis vectors. ReProCS then projects y_t orthogonal to the range of $\hat{P}_{t-1}$ to get $\tilde{y}_t := (I - \hat{P}_{t-1}\hat{P}_{t-1}{}')y_t$. Because of the slow subspace change assumption, doing this approximately nullifies ℓ_t and gives projected measurements of x_t. Notice that $\tilde{y}_t = A_t x_t + \beta_t$ where $A_t := (I - \hat{P}_{t-1}\hat{P}_{t-1}{}')$. Here $\beta_t := A_t \ell_t$ is small "noise" due to ℓ_t not being fully nullified. The problem of recovering x_t from $\tilde{y}_t$ is clearly a CS problem in small noise. If one considers the sequence of x_t's, and if slow support or signal value change holds for them, then this becomes a recursive dynamic CS problem. One example situation where these assumptions hold is when x_t is the foreground image sequence in a video analytics application[5]. The denseness assumption on the subspace basis vectors ensures that RIP holds for the matrix A_t [53]. This implies that x_t can be accurately recovered from $\tilde{y}_t$. One can then recover $\hat{\ell}_t = y_t - \hat{x}_t$ and use this to update its subspace estimate, $\hat{P}_t$. A more recent recursive RPCA algorithm, GRASTA [55], also uses the above idea although both the projected CS and subspace update steps are solved differently.

In all the applications described above except the last one, the signal of interest is only compressible (approximately sparse). Whenever we say "slow support change", we are referring to the changes in the b%-energy-support (the largest set containing at most b% of the total signal energy). In the last application, the outlier sequence, e.g., the foreground image sequence for the video analytics application, is an exactly sparse image sequence.

C. Motivation: why are new techniques needed?

One question that comes to mind is why are new techniques needed to solve the recursive dynamic CS problem and why can we not use ideas from the adaptive filtering or the tracking literature applied to simple-CS solutions? The reason is as follows. Adaptive filtering and tracking solutions rely on slow signal value change which is the only thing one can use for dense signal sequences. This can definitely be done for sparse, and approximately sparse, signal sequences as well, and does often result in good experimental results. We explain these ideas in Sec. VII. However sparse signal sequences have more structure, e.g., slow support change, that can be exploited to (i) get better algorithms and (ii) prove stronger theoretical results. For example, neither tracking-based solutions nor adaptive-filtering-based solutions allow for exact recovery using fewer measurements than what simple-CS solutions need. For a detailed example refer to Sec. VII-A1. Similarly, one cannot obtain stability over time results for these techniques under weaker assumptions on the measurement matrix than what simple-CS solutions need.

[4]Here L corresponds to the noise-free data matrix that is, by definition, low rank (its left singular vectors form the desired principal components); and X models the outliers (since outliers occur infrequently they are well modeled as forming a sparse matrix).

[5]The background image sequence of a typical static camera video is well modeled as lying in a low-dimensional subspace (forms ℓ_t) and the foreground image sequence is well-modeled as being sparse since it often consists of one or more moving objects (forms x_t) [51].

IV. Exploiting slow support change: Sparse recovery with partial support knowledge

When defining the recursive dynamic CS problem, we introduced two assumptions that are often valid in practice - slow support change and slow nonzero signal value change. In this section, we describe solutions that only exploit the slow support change assumption, i.e., (9). If the initial signal, x_0, can be recovered accurately, then, under this assumption, recursive dynamic CS can be reformulated as a problem of *sparse recovery using partial support knowledge* [33], [56]. We can use the support estimate of $\hat{x}_{t-1}$, denoted $\hat{\mathcal{N}}_{t-1}$, as the "partial support knowledge". We give the reformulated problem below followed by the proposed solutions for it. Their dynamic counterparts are summarized later in Sec. VI.

If the support does not change slowly enough, but the change is still highly correlated and the correlation model is known, one can get an accurate support prediction by using the correlation model information applied to the previous support estimate. An algorithm based on this idea is described in [57].

A. Reformulated problem: Sparse recovery using partial support knowledge

The goal is to recover a sparse vector, x, with support $\mathcal{N}$, from noise-free measurements, $y := Ax$, or from noisy measurements, $y := Ax + w$, when partial and possibly erroneous support knowledge, $\mathcal{T}$, is available [33], [56].

The true support $\mathcal{N}$ can be rewritten as

$$\mathcal{N} = \mathcal{T} \cup \Delta_u \setminus \Delta_e \quad \text{where} \quad \Delta_u := \mathcal{N} \setminus \mathcal{T},\ \Delta_e := \mathcal{T} \setminus \mathcal{N}$$

Here Δ_u denotes the set of unknown (or missing) support entries while Δ_e denotes the set of extra entries in $\mathcal{T}$. Let

$$s := |\mathcal{N}|, k := |\mathcal{T}|, u := |\Delta_u|, e := |\Delta_e|$$

It is easy to see that

$$s = k + u - e$$

We say the *support knowledge is accurate if* $u \ll s$ and $e \ll s$.

This problem is also of independent interest, since in many static sparse recovery applications, partial support knowledge is often available. For example, when using wavelet sparsity for an image with very little black background (most of its pixel are nonzero), most of its wavelet scaling coefficients will also be nonzero. Thus, the set of indices of the wavelet scaling coefficients could serve as accurate partial support knowledge.

B. Least Squares CS-residual (LS-CS)

The Least Squares CS-residual (LS-CS) algorithm [28], [58] can be interpreted as the first solution for the above problem. It starts by computing an initial LS estimate of x by assuming that its support set is equal to $\mathcal{T}$:

$$\hat{x}_{\text{init}} = I_{\mathcal{T}} (A_{\mathcal{T}}{}' A_{\mathcal{T}})^{-1} A_{\mathcal{T}}{}' y_t.$$

Using this, it computes

$$\hat{x} = \hat{x}_{\text{init}} + [\arg\min_b \|b\|_1 \text{ s.t. } \|y - A\hat{x}_{\text{init}} - Ab\|_2 \le \epsilon]. \quad (11)$$

This is followed by support estimation and computing a final LS estimate as described later in Sec. VI-B. The signal residual $\beta := x - \hat{x}_{\text{init}}$ satisfies

$$\beta = I_{\mathcal{T}} (A_{\mathcal{T}}{}' A_{\mathcal{T}})^{-1} A_{\mathcal{T}}{}' (A_{\Delta_u} x_{\Delta_u} + w) + I_{\Delta_u} x_{\Delta_u}$$

If A satisfies RIP of order at least $|\mathcal{T}| + |\Delta_u|$, $\|A_{\mathcal{T}}{}' A_{\Delta_u}\|_2 \le \theta_{|\mathcal{T}|,|\Delta_u|}$ is small. If the noise w is also small, clearly $\beta_{\mathcal{T}} := I_{\mathcal{T}}{}' \beta$ will be small and hence β will be approximately supported on Δ_u. When $|\Delta_u| \ll |\mathcal{N}|$, the approximate support of β is much smaller than that of x. Thus, one expects LS-CS to have smaller reconstruction error than BP-noisy when fewer measurements are available. This statement is quantified for the error upper bounds in [28] (see Theorem 1 and its corollary and the discussion that follows).

However, notice that the support size of β is $|\mathcal{T}| + |\Delta_u| \ge |\mathcal{N}|$. Since the number of measurements required for exact recovery is governed by the exact support size, LS-CS is not able to achieve exact recovery using fewer noiseless measurements than those needed by BP-noisy.

C. Modified-CS (mod-CS)

The search for a solution that achieves exact reconstruction using fewer measurements led to the modified-CS idea [33], [56]. To understand the approach, suppose first that Δ_e is empty, i.e., $\mathcal{N} = \mathcal{T} \cup \Delta_u$. Then the sparse recovery problem becomes one of trying to find the vector b that is sparsest outside the set $\mathcal{T}$ among all vectors that satisfy the data constraint. In the noise-free case, this can be written as

$$\min_b \|b_{\mathcal{T}^c}\|_0 \text{ s.t. } y = Ab$$

This is referred to as the *modified-*ℓ_0 problem [33], [56]. It also works if Δ_e is not empty. The following can be shown.

Proposition 4.1 ([33]): Modified-ℓ_0 will exactly recover x from $y := Ax$ if every set of $(k+2u)$ columns of A is linearly independent. Recall $k = |\mathcal{T}|$, $u = |\Delta_u|$.

Notice that $k + 2u = s + u + e = |\mathcal{N}| + |\Delta_e| + |\Delta_u|$. In comparison, the original ℓ_0 program, (1), requires every set of $2s$ columns of A to be linearly independent [9]. This is a stronger requirement when $u \ll s$ and $e \ll s$.

Like simple ℓ_0, the modified-ℓ_0 program also has exponential complexity, and hence we can again replace it by the ℓ_1 program to get

$$\min_b \|b_{\mathcal{T}^c}\|_1 \text{ s.t. } y = Ab \quad (12)$$

The above program was called *modified-CS* in [33], [56] where it was introduced. As shown in the result below, this also works even when Δ_e is not empty. The following was shown by Vaswani and Lu [33], [56].

Theorem 4.2 (RIP-based modified-CS exact recovery [33]): Consider recovering x with support $\mathcal{N}$ from $y := Ax$ by solving modified-CS, i.e., (12).

1) x is the unique minimizer of (12) if
   a) $\delta_{k+u} < 1$ and $\delta_{2u} + \delta_k + \theta^2_{k,2u} < 1$ and
   b) $a_k(2u, u) + a_k(u, u) < 1$ where $a_k(i, \check{i}) := \frac{\theta_{\check{i},i} + \frac{\theta_{i,k}\,\theta_{\check{i},k}}{1-\delta_k}}{1 - \delta_i - \frac{\theta^2_{i,k}}{1-\delta_k}}$

2) A weaker (but simpler) sufficient condition for exact recovery is

$$2\delta_{2u} + \delta_{3u} + \delta_k + \delta_{k+u}^2 + 2\delta_{k+2u}^2 < 1. \quad (13)$$

3) An even weaker (but even simpler) sufficient condition for exact recovery is

$$\delta_{k+2u} \leq 0.2.$$

Recall that $s := |\mathcal{N}|,\ k := |\mathcal{T}|,\ u := |\Delta_u|,\ e := |\Delta_e|$. The above conditions can also be rewritten in terms of s, e, u by substituting $k = s + e - u$ so that $k + 2u = s + e + u$.

Compare this result with that for BP which requires [29], [59], [60] $\delta_{2s} < \sqrt{2} - 1$ or $\delta_{2s} + \delta_{3s} < 1$. To compare the conditions numerically, we can use $u = e = 0.02s$ which is typical for time series applications (see Fig. 2). Using $\delta_{cr} \leq c\delta_{2r}$ [24, Corollary 3.4], it can be show that Modified-CS only requires $\delta_{2u} < 0.008$. On the other hand, BP requires $\delta_{2u} < 0.004$ which is clearly stronger.

An idea similar to modified-CS was independently introduced by von Borries et al. [61]. For noisy measurements, one can relax the data constraint in modified-CS either using the BP-noisy approach or the BPDN approach from Sec. II.

D. Weighted-ℓ_1

The weighted-ℓ_1 program studied in the work of Khajehnejad et al. [62], [63] and the later work of Friedlander et al. [47] can be interpreted as a generalization of modified-CS. The idea is to partition the index set $\{1, 2, \dots m\}$ into sets $\mathcal{T}_1, \mathcal{T}_2, \dots \mathcal{T}_q$ and to assume that the percentage of nonzero entries in each set is known. This knowledge is used to weight the ℓ_1 norm along each of these sets differently. Performance guarantees are obtained for the two set partition case $q = 2$. Using the notation from above, the two set partition can be labeled $\mathcal{T}, \mathcal{T}^c$. In this case, weighted-ℓ_1 solves

$$\min_b \|b_{\mathcal{T}^c}\|_1 + \tau\|b_{\mathcal{T}}\|_1 \text{ s.t. } y = Ab \quad (14)$$

Clearly, modified-CS is a special case of the above with $\tau = 0$. In general, the set $\mathcal{T}$ contains both extra entries Δ_e and missing (unknown) entries Δ_u. As long as the number of extras, $|\Delta_e|$, is small, modified-CS cannot be improved much further by weighted-ℓ_1. However if $|\Delta_e|$ is larger or if the measurements are noisy, the weighted-ℓ_1 generalization has smaller recovery error. This has been demonstrated experimentally in [47], [64]. For noisy measurements, one can relax the data constraint in (14) either using the BP-noisy approach or the BPDN approach from Sec. II.

Khajehnejad et al. [62], [63] obtained "weak thresholds" on the number of measurements required to ensure exact recovery with high probability. We state and discuss this result in Sec. IV-D1 below. We first give here the RIP based exact recovery condition from Friedlander et al. [47] since this can be easily compared with the modified-CS result from Theorem 4.2.

Theorem 4.3 (RIP-based weighted-ℓ_1 exact recovery [47]): Consider recovering x with support $\mathcal{N}$ from $y := Ax$ by solving weighted-ℓ_1, i.e., (14). Let $\alpha = \frac{|\mathcal{T} \cap \mathcal{N}|}{|\mathcal{T}|} = \frac{(s-u)}{(s+e-u)}$ and $\rho = \frac{|\mathcal{T}|}{|\mathcal{N}|} = \frac{(s+e-u)}{s}$. Let $\mathbb{Z}$ denote the set of integers and let $\frac{1}{s}\mathbb{Z}$ denote the set $\{\dots, \frac{-2}{s}, \frac{-1}{s}, 0, \frac{1}{s}, \frac{2}{s}, \dots\}$.

n/m	s/n	δ_s	δ_{2s}	u/s
0.1	0.0029	0.4343	0.6153	0.0978
0.2	0.0031	0.4343	0.6139	0.0989
0.3	0.003218	0.4343	0.61176	0.1007
0.4	0.003315	0.4343	0.61077	0.1015
0.5	0.003394	0.4343	0.60989	0.1023

TABLE I
THIS IS TABLE I OF [47] TRANSLATED INTO THE NOTATION USED IN THIS WORK. IT SHOWS FIVE SETS OF VALUES OF $(n/m), (s/n)$ FOR WHICH (13) DOES NOT HOLD, BUT (15) HOLDS AS LONG AS u/s IS BOUNDED BY THE VALUE GIVEN IN THE LAST COLUMN.

1) Pick an $a \in \frac{1}{s}\mathbb{Z}$ that is such that $a > \max(1, (1-\alpha)\rho)$. Weighted-$\ell_1$ achieves exact recovery if

$$\delta_{as} + \frac{a}{\gamma^2}\delta_{(a+1)s} < \frac{a}{\gamma^2} - 1$$

for $\gamma = \tau + (1-\tau)\sqrt{1 + \rho - 2\alpha\rho}$.

2) A weaker (but simpler) sufficient condition for exact recovery is

$$\delta_{2s} \leq \frac{1}{\sqrt{2}(\tau + (1-\tau)\sqrt{1+\rho-2\alpha\rho}) + 1}.$$

Corollary 4.4: By setting $\tau = 0$ in the above result, we get an exact recovery result for modified-CS. By setting $\tau = 1$, we obtain the original results for exact recovery using BP.

Let us compare Corollary 4.4 with the result from Theorem 4.2. To reduce the number of variables, suppose that $u = e$ so that $k = |\mathcal{T}| = s = |\mathcal{N}|$ and $\rho = 1$. By Corollary 4.4, modified-CS achieves exact recovery if

$$\delta_{2s} \leq \frac{1}{1 + 2\sqrt{\frac{u}{s}}}. \quad (15)$$

As demonstrated in [47], the above condition from Corollary 4.4 is weaker than the simplified condition (13) from Theorem 4.2. We summarize their discussion here. When $k = s$, it is clear that (13) will hold *only if*

$$\delta_s + 3\delta_s^2 < 1. \quad (16)$$

This, in turn, will hold *only if* $\delta_s < 0.4343$ [47]. Using the upper bounds for the RIC of a random Gaussian matrix from [65], one can bound δ_{2s} and find the set of u's for which (15) will hold. The upper bounds on u/s for various values of n/m and s/n for which (15) holds are displayed in Table I (this is Table 1 of [47]). For these values of n/m and s/n, $\delta_s = 0.4343$ and hence (13) of Theorem 4.2 does not hold.

1) Weak thresholds for high probability exact recovery for weighted-ℓ_1 and Modified-CS: In very interesting work, Khajehnejad et al. [63] obtained "weak thresholds" on the minimum number of measurements, n, (as a fraction of m) that are sufficient for exact recovery with overwhelming probability: the probability of not getting exact recovery decays to zero as the signal length m increases. The weak threshold was first defined by Donoho in [66] for BP.

Theorem 4.5 (weighted-ℓ_1 weak threshold [63, Theorem 4.3]): Consider recovering x with support $\mathcal{N}$ from $y := Ax$ by solving weighted-ℓ_1, i.e., (14). Let $\omega := 1/\tau$, $\gamma_1 := \frac{|\mathcal{T}|}{m}$ and $\gamma_2 := \frac{|\mathcal{T}^c|}{m} = 1 - \gamma_1$. Also let p_1, p_2 be the sparsity fractions

on the sets $\mathcal{T}$ and $\mathcal{T}^c$, i.e., let $p_1 := \frac{|\mathcal{T}|-|\Delta_e|}{|\mathcal{T}|}$ and $p_2 := \frac{|\Delta_u|}{|\mathcal{T}^c|}$. Then there exists a critical threshold

$$\delta_c = \delta_c(\gamma_1, \gamma_2, p_1, p_2, \omega)$$

such that for all $\frac{n}{m} > \delta_c$, the probability that a sparse vector x is not recovered decays to zero exponentially with m. In the above, $\delta_c(\gamma_1, \gamma_2, p_1, p_2, \omega) = \min\{\delta \mid \psi_{com}(\tau_1, \tau_2) - \psi_{int}(\tau_1, \tau_2) - \psi_{ext}(\tau_1, \tau_2) < 0 \ \forall \ 0 \leq \tau_1 \leq \gamma_1(1-p_1), 0 \leq \tau_2 \leq \gamma_2(1-p_2), \tau_1 + \tau_2 > \delta - \gamma_1 p_1 - \gamma_2 p_2\}$ where ψ_{com}, ψ_{int} and ψ_{ext} are obtained from the following expressions: Define $g(x) = \frac{2}{\sqrt{\pi}} e^{-x^2}$, $G(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-y^2} dy$ and let $\varphi(.)$ and $\Phi(.)$ be the standard Gaussian pdf and cdf functions respectively.

1) (Combinatorial exponent)

$$\psi_{com}(\tau_1, \tau_2) = \Bigg(\gamma_1(1-p_1) H\left(\frac{\tau_1}{\gamma_1(1-p_1)}\right) + \gamma_2(1-p_2) H\left(\frac{\tau_2}{\gamma_2(1-p_2)}\right) + \tau_1 + \tau_2 \Bigg) \log 2$$

where $H(\cdot)$ is the entropy function defined by $H(x) = -x \log x - (1-x)\log(1-x)$.

2) (External angle exponent) Define $c = (\tau_1 + \gamma_1 p_1) + \omega^2(\tau_2 + \gamma_2 p_2)$, $\alpha_1 = \gamma_1(1-p_1) - \tau_1$ and $\alpha_2 = \gamma_2(1-p_2) - \tau_2$. Let x_0 be the unique solution to x of the following:

$$2c - \frac{g(x)\alpha_1}{xG(x)} - \frac{\omega g(\omega x)\alpha_2}{xG(\omega x)} = 0$$

Then

$$\psi_{ext}(\tau_1, \tau_2) = c x_0^2 - \alpha_1 \log G(x_0) - \alpha_2 \log G(\omega x_0)$$

3) (Internal angle exponent) Let $b = \frac{\tau_1 + \omega^2 \tau_2}{\tau_1 + \tau_2}$, $\Omega' = \gamma_1 p_1 + \omega^2 \gamma_2 p_2$ and $Q(s) = \frac{\tau_1 \varphi(s)}{(\tau_1+\tau_2)\Phi(s)} + \frac{\omega \tau_2 \varphi(\omega s)}{(\tau_1+\tau_2)\Phi(\omega s)}$. Define the function $\hat{M}(s) = -\frac{s}{Q(s)}$ and solve for s in $\hat{M}(s) = \frac{\tau_1+\tau_2}{(\tau_1+\tau_2)b+\Omega'}$. Let the unique solution be s^* and set $y = s^*(b - \frac{1}{\hat{M}(s^*)})$. Compute the rate function $\Lambda^*(y) = sy - \frac{\tau_1}{\tau_1+\tau_2}\Lambda_1(s) - \frac{\tau_2}{\tau_1+\tau_2}\Lambda_1(\omega s)$ at the point $s = s^*$, where $\Lambda_1(s) = \frac{s^2}{2} + \log(2\Phi(s))$. The internal angle exponent is then given by:

$$\psi_{int}(\tau_1, \tau_2) = (\Lambda^*(y) + \frac{\tau_1 + \tau_2}{2\Omega'} y^2 + \log 2)(\tau_1 + \tau_2)$$

As explained in [63], the above result can be used to design an intelligent heuristic for picking τ as follows. Create a discrete set of possible values of τ. Use the above result to compute the weak threshold δ_c for each value of τ from this set and pick the τ that needs the smallest weak threshold. The weak threshold for the best τ also specifies the required number of measurements, n.

Corollary 4.6 (modified-CS and BP weak threshold [63]): Consider recovering x from $y := Ax$. Assume all notation from Theorem 4.5.

The weak threshold for BP is given by $\delta_c(0, 1, 0, \frac{s}{m}, 1)$.

The weak threshold for modified-CS is given by $\delta_c(\gamma_1, \gamma_2, p_1, p_2, \infty)$. In the special case when $\mathcal{T} \subseteq \mathcal{N}$, $p_1 = 1$. In this case, the weak threshold satisfies

$$\delta_c(\gamma_1, \gamma_2, 1, p_2, \infty) = \gamma_1 + \gamma_2 \delta_c(0, 1, 0, p_2, 1).$$

As explained in [63], when $\mathcal{T} \subseteq \mathcal{N}$, the above has a nice intuitive interpretation. In this case, the number of measurements n needed by modified-CS is equal to $|\mathcal{T}|$ plus the number of measurements needed for recovering the remaining $|\Delta_u|$ entries from $\mathcal{T}^c$ using BP.

E. Modified greedy algorithms and IHT-PKS

The modified-CS idea can also be used to modify other approaches for sparse recovery. This has been done in recent work by Stankovic et al. and Carillo et al. [67], [68] with encouraging results. They have developed and evaluated OMP with partially known support (OMP-PKS) [67], Compressive Sampling Matching Pursuit (CoSaMP)-PKS and IHT-PKS [68]. The greedy algorithms (OMP and CoSaMP) are modified as follows. Instead of starting with an initial empty support set, one starts with $\mathcal{T}$ as being the initial support set.

IHT is modified as follows. Let $k = |\mathcal{T}|$ and $s = |\mathcal{N}|$. IHT-PKS iterates as [68]:

$$\hat{x}^0 = 0, \ \hat{x}^{i+1} = (\hat{x}^i)_{\mathcal{T}} + H_{s-k}((\hat{x}^i + A'(y - A\hat{x}^i))_{\mathcal{T}^c}). \quad (17)$$

The authors also bound its error at each iteration for the special case when $\mathcal{T} \subseteq \mathcal{N}$. The bound shows geometric convergence as $i \to \infty$ to the true solution in the noise-free case and to within noise level of the true solution in the noisy case. We summarize this result in Section IV-H along with the other noisy case results.

F. An interesting interpretation of modified-CS

The following has been shown by Bandeira et al. [69]. Assume that $A_{\mathcal{T}}$ is full rank (this is a necessary condition in any case). Let $\mathcal{P}_{\mathcal{T},\perp}$ denote a projection onto the space perpendicular to $A_{\mathcal{T}}$, i.e., let

$$\mathcal{P}_{\mathcal{T},\perp} := (I - A_{\mathcal{T}}(A_{\mathcal{T}}{}' A_{\mathcal{T}})^{-1} A_{\mathcal{T}}').$$

Let $\tilde{y} := \mathcal{P}_{\mathcal{T},\perp} y$, $\tilde{A} := \mathcal{P}_{\mathcal{T},\perp} A_{\mathcal{T}^c}$ and $\tilde{x} := x_{\mathcal{T}^c}$. It is easy to see that $\tilde{y} = \tilde{A}\tilde{x}$ and $\tilde{x}$ is $|\Delta_u|$ sparse. Thus modified-CS can be interpreted as finding a $|\Delta_u|$-sparse vector $\tilde{x}$ of length $m - |\mathcal{T}|$ from $\tilde{y} := \tilde{A}\tilde{x}$. One can then recover $x_{\mathcal{T}}$ as the (unique) solution of $A_{\mathcal{T}} x_{\mathcal{T}} = y - A_{\mathcal{T}^c} x_{\mathcal{T}^c}$. More precisely let $\hat{x}_{modCS}$ denote the solution of modified-CS, i.e., (12). Then,

$$(\hat{x}_{modCS})_{\mathcal{T}^c} \in \arg\min_b \|b\|_1 \ s.t. \ (\mathcal{P}_{\mathcal{T},\perp} y) = (\mathcal{P}_{\mathcal{T},\perp} A_{\mathcal{T}^c}) b,$$
$$(\hat{x}_{modCS})_{\mathcal{T}} = (A_{\mathcal{T}})^{\dagger}(y - A_{\mathcal{T}^c}(\hat{x}_{modCS})_{\mathcal{T}^c}).$$

This interpretation can then be used to define a partial RIC.

Definition 4.7: We refer to δ_u^k as the *partial RIC* for a matrix A if, for any set $\mathcal{T}$ of size $|\mathcal{T}| = k$, $A_{\mathcal{T}}$ is full column rank and δ_u^k is the smallest real number so that

$$(1 - \delta_u^k)\|b\|_2^2 \leq \|(\mathcal{P}_{\mathcal{T},\perp} A_{\mathcal{T}^c}) b\|_2^2 \leq (1 + \delta_u^k)\|b\|_2^2$$

for all u-sparse vectors b of length $(m-k)$ and all sets $\mathcal{T}$ of size k.

With this, any of the results for BP or BP-noisy can be directly applied to get a result for modified-CS. For example, using Theorem 2.4 we can conclude the following

Theorem 4.8: If $\delta_{2u}^k < \sqrt{2} - 1$, then modified-CS achieves exact recovery.

G. Related ideas: Truncated BP and Reweighted ℓ_1

1) Truncated BP: The modified-CS program has been used in the work of Wang and Yin [70] for developing a new algorithm for simple sparse recovery. They call it *truncated basis pursuit (BP)* and use it iteratively to improve the recovery error for regular sparse recovery. In the zeroth iteration, they solve the BP program and compute the estimated signal's support by thresholding. This is then used to solve modified-CS in the second iteration and the process is repeated with a specific support threshold setting scheme.

2) Reweighted ℓ_1: The reweighted ℓ_1 approach developed by Candes et al. in much earlier work [71] is similarly related to weighted ℓ_1 described above. In this case again, in the zeroth iteration, one solves BP. Denote the solution by $\hat{x}^0$. In the next iteration one uses the entries of $\hat{x}^0$ to weight the ℓ_1 norm in an intelligent fashion and solves the weighted ℓ_1 program. This procedure is repeated until a stopping criterion is met.

H. Error bounds for the noisy case

It is easy to bound the error of modified-CS-noisy or weighted-ℓ_1-noisy by modifying the corresponding RIP-based results for BP-noisy. Weighted-ℓ_1-noisy solves

$$\min_b \|b_{\mathcal{T}^c}\|_1 + \tau\|b_{\mathcal{T}}\|_1 \text{ s.t. } \|y - Ab\|_2 \leq \epsilon \tag{18}$$

and modified-CS-noisy solves the above with $\tau = 0$.

Let x be a sparse vector with support $\mathcal{N}$ and let $y := Ax+w$ with $\|w\|_2 \leq \epsilon$.

Theorem 4.9 (modified-CS error bound [72, Lemma 2.7]): Let $\hat{x}$ be the solution of modified-CS-noisy, i.e., (18) with $\tau = 0$. If $\delta_{|\mathcal{T}|+3|\Delta_u|} < (\sqrt{2}-1)/2$, then

$$\|x - \hat{x}\| \leq C_1(|\mathcal{T}| + 3|\Delta_u|)\epsilon \leq 7.50\epsilon,\ C_1(k) := \frac{4\sqrt{1+\delta_k}}{1-2\delta_k}.$$

Notice that $\delta_{|\mathcal{T}|+3|\Delta_u|} = \delta_{|\mathcal{N}|+|\Delta_e|+2|\Delta_u|}$. A similar result for a compressible (approximately sparse) signal and weighted-ℓ_1-noisy was proved in [47].

Theorem 4.10 (weighted-ℓ_1 error bound [47]): Let $\hat{x}$ be the solution of (18). Let x_s denote the best s term approximation for x and let $\mathcal{N} = \text{support}(x_s)$. Let α, ρ, and a be as defined in Theorem 4.3. Assume the exact recovery condition given in Theorem 4.3 holds for $\mathcal{N}$ as defined here. Then,

$$\|\hat{x} - x\|_2 \leq C_0'\epsilon + C_1' s^{-1/2}(\tau\|x - x_s\|_1 + (1-\tau)\|x_{(N\cup T)^c}\|_1)$$

where $C_0' = C_0'(\tau, s, a, \delta_{(a+1)s}, \delta_{as})$ and $C_1' = C_1'(\tau, s, a, \delta_{(a+1)s}, \delta_{as})$ are constants specified in Remark 3.2 of [47].

The following was proved for IHT-PKS. It only analyzes the special case $\mathcal{T} \subseteq \mathcal{N}$ (no extras in $\mathcal{T}$).

Theorem 4.11: Recall the IHT-PKS algorithm from (17). If $e := |\Delta_e| = 0$, i.e., if $\mathcal{T} \subseteq \mathcal{N}$, if $\|A\|_2 < 1$ and if $\delta_{3s-2k} < 1/\sqrt{32}$, then the i-th IHT-PKS iterate satisfies

$$\|x - \hat{x}^i\|_2 \leq (\sqrt{8}\delta_{3s-2k})^i\|x\|_2 + C\epsilon, \tag{19}$$

where $C = \sqrt{1+\delta_{2s-k}}\left(\frac{1-\alpha^t}{1-\alpha}\right)$. Recall that $s = |\mathcal{N}|$ and $k = |\mathcal{T}|$. With $e = 0$, $3s - 2k = s + 3u$ and $2s - k = s + 2u$.

V. Exploiting slow support and slow signal value change: Sparse recovery with partial support and signal value knowledge

In the previous section, we discussed approaches that only exploit slow support change. In many recursive dynamic CS applications, slow signal value change also holds. We discuss here how to design improved solutions that use this knowledge also. We begin by first stating the reformulated static problem.

A. Reformulated problem: Sparse recovery with partial support and signal value knowledge

The goal is to recover a sparse vector x, with support set $\mathcal{N}$, either from noise-free undersampled measurements, $y := Ax$, or from noisy measurements, $y := Ax+w$, when a signal value estimate, $\hat{\mu}$, is available. We let the partial support knowledge $\mathcal{T}$ be equal to the support of $\hat{\mu}$.

The true support $\mathcal{N}$ can be written as

$$\mathcal{N} = \mathcal{T} \cup \Delta_u \setminus \Delta_e \text{ where } \Delta_u := \mathcal{N} \setminus \mathcal{T},\ \Delta_e := \mathcal{T} \setminus \mathcal{N}$$

and the true signal x can be written as

$$(x)_{\mathcal{N}\cup\mathcal{T}} = (\hat{\mu})_{\mathcal{N}\cup\mathcal{T}} + \nu,\ (x)_{\mathcal{N}^c} = 0. \tag{20}$$

By definition, $(\hat{\mu})_{\mathcal{T}^c} = 0$. The error ν in the prior signal estimate is assumed to be small, i.e., $\|\nu\| \ll \|x\|$.

B. Regularized modified-BPDN

Regularized modified-BPDN adds the slow signal value change constraint to modified-BPDN. In general, this can be imposed as either a bound on the max norm or on the 2-norm [73] or it can be included into the cost function as a soft constraint. *Regularized modified-BPDN (reg-mod-BPDN)* does the latter [64]. It solves:

$$\min_b \gamma\|b_{\mathcal{T}^c}\|_1 + 0.5\|y - Ab\|_2^2 + 0.5\lambda\|b_{\mathcal{T}} - \hat{\mu}_{\mathcal{T}}\|_2^2. \tag{21}$$

Setting $\lambda = 0$ in (21) gives modified-BPDN (mod-BPDN).

Remark 5.1: Notice that the third term in (21) acts as a regularizer on $b_{\mathcal{T}}$. It ensures that (21) has a unique solution at least when b is constrained to be supported on $\mathcal{T}$. In the absence of this term, one would need $A_{\mathcal{T}}$ to be full rank to ensure this. The first term is a regularizer on $b_{\mathcal{T}^c}$. As long as γ is large enough, i.e., $\gamma \geq \gamma^*$, and an incoherence condition holds on the columns of the matrix A, one can show that (21) will have a unique minimizer even without a constraint on b. We state a result of this type below.

In [64], Lu and Vaswani analyzed the reg-mod-BPDN solution and obtained a computable error bound on it that holds without any sufficient conditions. We discuss this below.

1) Computable error bounds: In [74], Tropp introduced the Exact Recovery Coefficient (ERC) to quantify the incoherence between columns of the matrix A and used this to obtain a computable error bound for BPDN. By modifying this approach, one can get a similar result for reg-mod-BPDN and hence also for mod-BPDN (which is a special case) [64]. With some extra work, one can obtain a computable bound that holds always. The main idea is to maximize over all choices of the sparsified approximation of x that satisfy the required

sufficient conditions [64]. We state this result below. Since the bound is computable and holds without sufficient conditions and, from simulations, is also fairly tight (see [64, Fig. 4]), it provides a good heuristic for setting γ for mod-BPDN and γ and λ for reg-mod-BPDN. We explain how to do this in Sec. VI-H.

Theorem 5.2 (computable bound for reg-mod-BPDN): Let x be a sparse vector with support $\mathcal{N}$ and let $y := Ax + w$ with $\|w\|_2 \leq \epsilon$. Assume that the columns of A have unit 2-norm. When they do not have unit 2-norm, we can use Remark 2.1. If $\gamma = \gamma^*_{\mathcal{T},\lambda}(\tilde{\Delta}^*_u(k_{\min}))$, then reg-mod-BPDN (21) has a unique minimizer, $\hat{x}$, that is supported on $\mathcal{T} \cup \tilde{\Delta}^*_u(k_{\min})$, and that satisfies

$$\|x - \hat{x}\|_2 \leq g_\lambda(\tilde{\Delta}^*_u(k_{\min})).$$

The quantities $k_{\min}$, $\tilde{\Delta}^*_u(k_{\min})$, $\gamma^*_{\mathcal{T},\lambda}(\tilde{\Delta}^*_u(k_{\min}))$, $g(\tilde{\Delta}^*_u(k_{\min}))$ are defined in Appendix A (have very long expressions).

Remark 5.3: As we explain in Appendix A, everything needed for the above result can be computed in polynomial time if x, $\mathcal{T}$ and a bound on $\|w\|_\infty$ are available.

Remark 5.4: One can actually get a unique minimizer of the reg-mod-BPDN program using any $\gamma \geq \gamma^*_{\mathcal{T},\lambda}(\tilde{\Delta}^*_u(k_{\min}))$. By setting $\gamma = \gamma^*$, we get the smallest error bound and this is why we state the theorem as above.

Corollary 5.5 (computable bound for mod-BPDN, BPDN): If $A_\mathcal{T}$ is full rank, and $\gamma = \gamma^*_{\mathcal{T},0}(\tilde{\Delta}^*_u(k_{\min}))$, then the solution of modified-BPDN satisfies $\|x - \hat{x}\|_2 \leq g_0(\tilde{\Delta}^*_u(k_{\min}))$.
The corollary for BPDN follows by setting $\mathcal{T} = \emptyset$ and $\Delta_u = \mathcal{N}$ in the above result.

C. Modified-BPDN-residual

Another way to use both slow support and slow signal value knowledge is as follows. Replace y by $(y - A\hat{\mu})$ and solve the modified-BPDN program. Add back its solution $\hat{\mu}$. We refer to this as *modified-BPDN-residual.* It computes [75]

$$\hat{x} = \hat{\mu} + [\arg\min_b \gamma\|b_{\mathcal{T}^c}\|_1 + 0.5\|y - A\hat{\mu} - Ab\|_2^2]. \quad (22)$$

VI. Recursive dynamic CS

To solve the original problem described in Section III, it is easy to develop dynamic versions of the approaches described so far. To do this, two things are required. First, an accurate estimate of the sparse signal at the initial time instant is required and second an accurate support estimation technique is required. We discuss these two issues in the next two subsections. After this we briefly summarize the dynamic versions of the approaches from the previous two sections, followed by algorithms that were directly designed for the dynamic problem. In Sec. VI-H, we discuss general parameter setting strategies. Finally, in Sec. VI-I, we summarize error stability over time results.

A. Initialization

To initialize any recursive dynamic CS solution, one needs an accurate estimate of x_0. This can be computed using any sparse recovery technique as long as enough measurements are available to allow for exact or accurate enough reconstruction. For example, BP or BP-noisy can be used.

Alternatively, one can obtain partial support knowledge for the initial signal using other types of prior knowledge, e.g., for a wavelet-sparse image with a very small black region, most of the wavelet scaling coefficients will be nonzero. Thus the indices of these coefficients can serve as partial support knowledge at $t = 0$ [33]. If this knowledge is not as accurate as the support estimate from the previous time instant, more measurements would be required at $t = 0$.

In most applications where sparse recovery is used, such as MRI, it is possible to use different number of measurements at different times. In applications where this is not possible, one can use a batch sparse recovery technique for the first short batch of time instants. For example, one could solve the multiple measurement vectors' (MMV) problem that attempts to recover a batch of sparse signals with common support from a batch of their measurements [11]–[14], [76], [77]. Or one could use the approach of Zhang and Rao [78] (called temporal SBL) that solves the MMV problem with the extra assumption that the elements in each nonzero row of the solution matrix are temporally correlated.

B. Support estimation

The simplest way to estimate the support is by thresholding, i.e., by computing

$$\hat{\mathcal{N}} = \{i : |(\hat{x})_i| > \alpha\}$$

where $\alpha \geq 0$ is the zeroing threshold. If $\hat{x} = x$ (exact recovery), we use $\alpha = 0$. Exact recovery is possible only for sparse x and noise-free (and enough) measurements. In other situations, we need a nonzero value. In case of accurate reconstruction of a sparse signal, we can set α to be a little smaller than an estimate of its smallest magnitude nonzero entry [33]. For compressible signals, one should use an estimate of the smallest magnitude nonzero entry of a sparsified version of the signal, e.g., sparsified to retain $b\%$ of the total signal energy. In general α should also depend on the noise level. We discuss the setting of α further in Section VI-H.

For all of LS-CS, modified-CS and weighted-ℓ_1, it can be argued that $\hat{x}$ is a biased estimate of x: it is biased towards zero along Δ_u and away from zero along $\mathcal{T}$ [28], [72]. As a result, the threshold α is either too small and does not delete all extras (subset of $\mathcal{T}$) or is too large and does not detect all misses (subset of $\mathcal{T}^c$). A partial solution to this issue is provided by a two step support estimation approach that we call Add-LS-Del. The Add-LS part of this approach is motivated by the Gauss-Dantzig selector idea [60] that first demonstrated that support estimation with a nonzero threshold followed by computing an LS estimate on the support significantly improves the sparse recovery error because it reduces the bias in the solution estimate. Add-LS-Del proceeds as follows.

$$\begin{aligned}
\mathcal{T}_{\text{add}} &= \mathcal{T} \cup \{i : |(\hat{x})_i| > \alpha_{\text{add}}\} \\
\hat{x}_{\text{add}} &= I_{\mathcal{T}_{\text{add}}} {A_{\mathcal{T}_{\text{add}}}}^\dagger y \\
\hat{\mathcal{N}} &= \mathcal{T}_{\text{add}} \setminus \{i : |(\hat{x}_{\text{add}})_i| \leq \alpha_{\text{del}}\} \\
\hat{x}_{final} &= I_{\hat{\mathcal{N}}} (A_{\hat{\mathcal{N}}})^\dagger y
\end{aligned} \quad (23)$$

Algorithm 1 Dynamic modified-BPDN [64]

At $t = 0$: Solve BPDN with sufficient measurements, i.e., compute $\hat{x}_0$ as the solution of $\min_b \gamma\|b\|_1+0.5\|y_0-A_0b\|_2^2$ and compute its support by thresholding: $\hat{\mathcal{N}}_0 = \{i : |(\hat{x}_0)_i| > \alpha\}$.
For each $t > 0$ do

1) Set $\mathcal{T} = \hat{\mathcal{N}}_{t-1}$
2) *Mod-BPDN* Compute $\hat{x}_t$ as the solution of

$$\min_b \gamma\|b_{\mathcal{T}^c}\|_1 + 0.5\|y_t - A_t b\|_2^2$$

3) *Support Estimation (Simple):* $\hat{\mathcal{N}}_t = \{i : |(\hat{x}_t)_i| > \alpha\}$

Parameter setting: see Section VI-H step 2.

Algorithm 2 Dynamic weighted-ℓ_1 [63]

At $t = 0$: Solve BPDN with sufficient measurements, i.e., compute $\hat{x}_0$ as the solution of $\min_b \gamma\|b\|_1 + 0.5\|y_0 - A_0b\|_2^2$ and compute its support $\hat{\mathcal{N}}_0 = \{i : |(\hat{x}_0)_i| > \alpha\}$. For each $t > 0$ do

1) Set $\mathcal{T} = \hat{\mathcal{N}}_{t-1}$
2) *Weighted-ℓ_1* Compute $\hat{x}_t$ as the solution of

$$\min_b \gamma\|b_{\mathcal{T}^c}\|_1 + \gamma\tau\|b_{\mathcal{T}}\|_1 + 0.5\|y_t - A_t b\|_2^2$$

3) *Support Estimation (Simple):* $\hat{\mathcal{N}}_t = \{i : |(\hat{x}_t)_i| > \alpha\}$

Parameter setting: see Section VI-H step 2.

The addition step threshold, α_{add}, needs to be just large enough to ensure that the matrix used for LS estimation, $A_{\mathcal{T}_{\text{add}}}$ is well-conditioned. If α_{add} is chosen properly and if the number of measurements, n, is large enough, the LS estimate on $\mathcal{T}_{\text{add}}$ will have smaller error, and will be less biased, than $\hat{x}$. As a result, deletion will be more accurate when done using this estimate. This also means that one can use a larger deletion threshold, α_{del}, which will ensure deletion of more extras.

C. Dynamic LS-CS, modified-CS, weighted-ℓ_1, IHT-PKS

It is easy to develop dynamic versions of all the approaches described in Sec. IV by (i) recovering x_0 as described above, (ii) for $t > 0$, using $\mathcal{T} = \hat{\mathcal{N}}_{t-1}$ where $\hat{\mathcal{N}}_{t-1}$ is computed from $\hat{x}_{t-1}$ as explained above, and (iii) by using $y = y_t$ and $A = A_t$ at time t. We summarize dynamic weighted-ℓ_1, dynamic modified-BPDN and dynamic IHT-PKS in Algorithms 1, 2, 3 respectively. In these algorithms, we have used simple BPDN at the initial time, but this can be replaced by the other approaches described in Sec. VI-A.

D. Streaming ell-1 homotopy and streaming modified weighted-ℓ_1 (streaming mod-wl1)

In [79], Asif et al. developed a fast homotopy based solver for the general weighted ℓ_1 minimization problem:

$$\min_b \|Wb\|_1 + 0.5\|y - Ab\|_2^2 \tag{24}$$

where W is a diagonal weighting matrix. They also developed a homotopy for solving (24) with a third term $0.5\|b - F\mu\|_2^2$. Here F can be an arbitrary square matrix. For the dynamic problem, it is usually the state transition matrix of the dynamical system [79]. We use these solvers for some of our numerical experiments.

Algorithm 3 Dynamic IHT-PKS

At $t = 0$: Compute $\hat{x}_0$ using (5) and compute its support as $\hat{\mathcal{N}}_0 := \{i : |(\hat{x}_0)_i| > 0\}$.
For each $t > 0$ do

1) Set $\mathcal{T} = \hat{\mathcal{N}}_{t-1}$
2) *IHT-PKS* Compute $\hat{x}_t$ using (17).
3) *Support Estimation:* $\hat{\mathcal{N}}_t = \{i : |(\hat{x}_t)_i| > 0\}$

Another key contribution of this work was a weighting scheme for solving the recursive dynamic CS problem that can be understood as a generalization of the modified-CS idea. At time t, one solves (24) with $W_{i,i} = \frac{\gamma}{\beta|(\hat{x}_{t-1})_i|+1}$ with $\beta \gg 1$. The resulting algorithm, that can be called *streaming modified weighted-ℓ_1 (streaming mod-wl1)*, is summarized in Algorithm 4. When x_t's, and hence $\hat{x}_t$'s, are exactly sparse, if we set $\beta = \infty$, we recover the modified-CS program (12). To understand this, let $\mathcal{T} = \text{supp}(\hat{x}_{t-1})$. Thus, if $i \in \mathcal{T}$, then $W_{i,i} = 0$ while if $i \notin \mathcal{T}$, then $W_{i,i}$ equals a constant nonzero value. This gives the modified-BPDN cost function.

The above is a very useful generalization since it uses weights that are inversely proportional to the magnitude of the entries of $\hat{x}_{t-1}$. Thus it incorporates support change knowledge in a soft fashion while also using signal value knowledge. It also naturally handles compressible signals for which no entry of $\hat{x}_{t-1}$ is zero but many entries are small. The weighting scheme is similar to that introduced in the reweighted ℓ_1 approach of [71] to solve the simple CS problem.

E. Dynamic regularized-modified-BPDN

It is easy to develop a dynamic version of reg-mod-BPDN by using $y = y_t$ and $A = A_t$ and by recovering x_0 as described in Sec. VI-A above. We set $\mathcal{T} = \hat{\mathcal{N}}_{t-1}$ where $\hat{\mathcal{N}}_{t-1}$ is computed from $\hat{x}_{t-1}$ as explained above. For $\hat{\mu}$, one can either use $\hat{\mu} = \hat{x}_{t-1}$, or, in certain applications, e.g., functional MRI reconstruction [37], where the signal values do not change much w.r.t. the first frame, using $\hat{\mu} = \hat{x}_0$ is a better idea. This is because $\hat{x}_0$ is estimated using many more measurements and hence is much more accurate. We summarize the complete algorithm in Algorithm 5.

F. Kalman filtered Modified-CS (KF-ModCS)

Kalman Filtered CS-residual (KF-CS) was introduced for solving the recursive dynamic CS problem in [32] and in fact this was the first solution to this problem. With the modified-CS approach and results now known, a much better idea than KF-CS is *Kalman Filtered Modified-CS-residual (KF-ModCS).* This can be understood as modified-BPDN-residual but with $\hat{\mu}$ obtained as a Kalman filtered estimate on the previous support estimate $\mathcal{T} = \hat{\mathcal{N}}_{t-1}$. For the KF step, one needs to assume a model on signal value change. In the absence of specific information, a Gaussian random walk model with equal change

Algorithm 4 Streaming modified weighted-ℓ_1 (streaming mod-wl1) [79]

1: At $t = 0$: Solve BPDN with sufficient measurements, i.e., compute $\hat{x}_0$ as the solution of $\min_b \gamma\|b\|_1 + 0.5\|y_0 - A_0 b\|_2^2$ and compute its support $\hat{\mathcal{N}}_0 = \{i : |(\hat{x}_0)_i| > \alpha\}$.

2: For $t \geq 2$, set

$$(W_t)_{ii} = \frac{\gamma}{\beta|(\hat{x}_{t-1})_i| + 1},$$

where $\beta = n\frac{\|\hat{x}_{t-1}\|_2^2}{\|\hat{x}_{t-1}\|_1^2}$, and compute $\hat{x}$ as the solution of

$$\min_x \|W_t x\|_1 + \frac{1}{2}\|y_t - A_t x\|_2^2$$

3: $t \leftarrow t + 1$, go to step 2.

Algorithm 5 Dynamic Regularized Modified-BPDN [64]

At $t = 0$: Solve BPDN with sufficient measurements, i.e., compute $\hat{x}_0$ as the solution of $\min_b \gamma\|b\|_1 + 0.5\|y_0 - A_0 b\|_2^2$ and compute its support $\hat{\mathcal{N}}_0 = \{i : |(\hat{x}_0)_i| > \alpha\}$.
For each $t > 0$ do

1) Set $\mathcal{T} = \hat{\mathcal{N}}_{t-1}$
2) *Reg-Mod-BPDN* Compute $\hat{x}_t$ as the solution of

$$\min_b \gamma\|b_{\mathcal{T}^c}\|_1 + 0.5\|y_t - A_t b\|_2^2 + 0.5\lambda\|b_{\mathcal{T}} - \hat{\mu}_{\mathcal{T}}\|_2^2$$

3) *Support Estimation (Simple):* $\hat{\mathcal{N}}_t = \{i : |(\hat{x}_t)_i| > \alpha\}$

Parameter setting: see Section VI-H step 2.

variance can be used [32]:

$$\begin{aligned}
&(x_0)_{\mathcal{N}_0} \sim \mathcal{N}(0, \sigma_{sys,0}^2 I),\\
&(x_t)_{\mathcal{N}_t} = (x_{t-1})_{\mathcal{N}_t} + \nu_t,\ \nu_t \sim \mathcal{N}(0, \sigma_{sys}^2 I)\\
&(x_t)_{\mathcal{N}_t^c} = 0
\end{aligned} \tag{25}$$

Here $\mathcal{N}(a, \Sigma)$ denotes a Gaussian distribution with mean a and covariance matrix Σ. Assume for a moment that the support does not change with time, i.e., $\mathcal{N}_t = \mathcal{N}_0$, and $\mathcal{N}_0$ is known or can be perfectly estimated using BP-noisy followed by support estimation. With the above model on x_t, if $y_t = A_t x_t + w_t$ with the observation noise w_t being Gaussian, the KF provides a causal minimum mean squared error (MMSE) solution, i.e., it returns $\hat{x}_{t|t}$ which solves

$$\arg \min_{\hat{x}_{t|t}:(\hat{x}_{t|t})_{\mathcal{N}_0^c}=0} \mathbb{E}_{x_t|y_1,y_2,\ldots y_t}[\|x_t - \hat{x}_{t|t}(y_1, y_2, \ldots y_t)\|_2^2]$$

Here $\mathbb{E}_{x|y}[q(x)]$ denotes expectation of $q(x)$ given y.

Our problem is significantly more difficult because the support set $\mathcal{N}_t$ changes with time and is unknown. To solve it, KF-ModCS is a practical heuristic that combines the modified-BPDN-residual idea for tracking the support with an adaptation of the regular KF algorithm to the case where the set of entries of x_t that form the state vector for the KF change with time: the KF state vector at time t is $(x_t)_{\mathcal{N}_t}$ at time t. Unlike the regular KF for a fixed dimensional linear Gaussian state space model, KF-CS or KF-ModCS do not enjoy any optimality properties. One expects KF-ModCS to outperform modified-BPDN when accurate prior knowledge of the signal values is available. We summarize it in Algorithm 6.

Algorithm 6 KF-ModCS: use of mod-BPDN-residual to replace BPDN in the KF-CS algorithm of [32]

Parameters: σ_{sys}^2, σ_{obs}^2, α, γ
At $t = 0$: Solve BPDN with sufficient measurements, i.e., compute $\hat{x}_0$ as the solution of $\min_b \gamma\|b\|_1 + 0.5\|y_0 - A_0 b\|_2^2$. Denote the solution by $\hat{x}_0$. Estimate its support, $\mathcal{T} = \hat{\mathcal{N}}_0 = \{i : |(\hat{x}_0)_i| > \alpha\}$.
Initialize $P_0 = \sigma_{sys}^2 I_{\mathcal{T}} I_{\mathcal{T}}{}'$.
For each $t > 0$ do

1) Set $\mathcal{T} = \hat{\mathcal{N}}_{t-1}$
2) *Mod-BPDN-residual:*

$$\hat{x}_{t,mod} = \hat{x}_{t-1} + [\arg\min_b \gamma\|b_{\mathcal{T}^c}\|_1 + \|y_t - A_t\hat{x}_{t-1} - A_t b\|^2]$$

3) *Support Estimation - Simple Thresholding:*

$$\hat{\mathcal{N}}_t = \{i : |(\hat{x}_{t,mod})_i| > \alpha\} \tag{26}$$

4) *Kalman Filter:*

$$\begin{aligned}
&\hat{Q}_t = \sigma_{sys}^2 I_{\hat{\mathcal{N}}_t} I_{\hat{\mathcal{N}}_t}{}'\\
&K_t = (P_{t-1} + \hat{Q}_t)A_t' \left(A_t(P_{t-1} + \hat{Q}_t)A_t' + \sigma_{obs}^2 I\right)^{-1}\\
&P_t = (I - K_t A_t)(P_{t-1} + \hat{Q}_t)\\
&\hat{x}_t = (I - K_t A_t)\hat{x}_{t-1} + K_t y_t
\end{aligned} \tag{27}$$

Parameter setting: see Section VI-H step 2.

An open question is how to analyze KF-CS or KF-ModCS and get a meaningful performance guarantee? This is a hard problem because the KF state vector is $(x_t)_{\mathcal{N}_t}$ at time t and $\mathcal{N}_t$ changes with time. In fact, even if we assume that the sets $\mathcal{N}_t$ are given, there are no known results for the resulting "genie-aided KF".

G. Dynamic CS via approximate message passing(DCS-AMP)

Another approximate Bayesian approach was developed in very interesting recent work by Ziniel and Schniter [80], [81]. They introduced the dynamic CS via approximate message passing (DCS-AMP) algorithm by developing the recently introduced AMP approach of Donoho et al. [82] for the dynamic CS problem. The authors model the dynamics of the sparse vectors over time using a stationary Bernoulli Gaussian prior as follows: for all $i = 1, 2, \ldots m$,

$$(x_t)_i = (s_t)_i(\theta_t)_i$$

where $(s_t)_i$ is a binary random variable that forms a stationary Markov chain over time and $(\theta_t)_i$ follows a stationary first order autoregressive (AR) model with nonzero mean. Independence is assumed across the various indices i. Let $\lambda := \Pr((s_t)_i = 1)$ and $p_{10} := \Pr((s_t)_i = 1|(s_{t-1})_i = 0)$. Using stationarity, this tells us that $p_{01} := \Pr((s_t)_i = 0|(s_{t-1})_i = 1) = \frac{\lambda p_{10}}{1-\lambda}$. The following stationary AR model with nonzero mean is imposed on θ_t.

$$(\theta_t)_i = (1 - \alpha)((\theta_{t-1})_i - \zeta_i) + \alpha(v_t)_i$$

with $0 < \alpha \leq 1$. In the above model, v_t is the i.i.d. white Gaussian perturbation, ζ is the mean, and $(1 - \alpha)$ is the AR

coefficient. The choice of α controls how slowly or quickly the signal values change over time. The choice of p_{10} controls the likelihood of new support addition(s). This model results in a Bernoulli-Gaussian or "spike-and-slab" distribution of $(x_t)_i$, which is known to be an effective sparsity promoting prior [80], [81].

Exact computation of the minimum mean squared error (MMSE) estimate of x_t cannot be done under the above model. On one end, one can try to use sequential Monte Carlo techniques (particle filtering) to approximate the MMSE estimate. Some attempts to do this are described in [83], [84]. But these can get computationally expensive for high dimensional problems and it is never clear what number of particles is sufficient to get an accurate enough estimate. The AMP approach developed in [81] is also approximate but is extremely fast and hence is useful. The complete DCS-AMP algorithm taken from [81, Table II] is summarized in Table II (with permission from the authors). Its code is available at http://www2.ece.ohio-state.edu/~schniter/DCS/index.html.

H. Parameter Setting

In this section we discuss parameter setting for the algorithms described above.

1) For DCS-AMP, an expectation maximization (EM) algorithm is proposed for parameter learning [81]. We summarize this in Table III.
2) For the rest of the algorithms, the following procedure is proposed. This assumes that, for the first two time instants, enough measurements are available so that a simple CS technique such as BPDN can be used to recover x_1 and x_2 accurately.
   a) Let $\hat{x}_1, \hat{x}_2$ denote the estimates of x_1, x_2 obtained using BPDN (and enough measurements).
   b) Consider the simple single-threshold based support estimation scheme. Let α_0 be the smallest real number so that $\sum_{i:|(\hat{x}_1)_i|>\alpha_0} (\hat{x}_1)_i^2 < 0.999 \sum_i (\hat{x}_1)_i^2$. We set the threshold $\alpha = c\alpha_0$ with $c = 1/12$. In words, α is a fraction of the smallest magnitude nonzero entry of $\hat{x}_1$ sparsified to 99.9% energy. We used 99.9% energy for sparsification and $c = 1/12$ in our experiments, but these numbers can be changed.
   c) Consider dynamic reg-mod-BPDN (Algorithm 5). It has three parameters α, γ, λ. We set α as in step 2b. We set γ and λ using a heuristic motivated by Theorem 5.2. This theorem can be used because, for a given choice of λ, it gives us an error bound that is computable in polynomial time and that holds always (without any sufficient conditions). To use this heuristic, we need $\mathcal{T}, \mathcal{N}, \hat{\mu}, x, \Delta_u, \Delta_e$ and an estimate of $\|w\|_\infty$. We compute $\hat{\mathcal{N}}_1 = \{i : |(\hat{x}_1)_i| > \alpha\}$ and $\hat{\mathcal{N}}_2 = \{i : |(\hat{x}_2)_i| > \alpha\}$. We set $\mathcal{T} = \hat{\mathcal{N}}_1$, $\mathcal{N} = \hat{\mathcal{N}}_2$, $\hat{\mu} = \hat{x}_1$, $x = \hat{x}_2$, $\Delta_u = \mathcal{N} \setminus \mathcal{T}$, and $\Delta_e = \mathcal{T} \setminus \mathcal{N}$. We use $\|y_2 - A\hat{x}_2\|_\infty$ as an estimate of $\|w\|_\infty$. With these, for a given value of λ, one can compute $g_\lambda(\tilde{\Delta}_u^*(k_{\min}))$ defined in Theorem 5.2. We pick λ by selecting the one that minimizes $g_\lambda(\tilde{\Delta}_u^*(k_{\min}))$ out of a discrete set of possible values. In our experiments we selected λ out of the set $\{0.5, 0.2, 0.1, 0.05, 0.01, 0.005, 0.001, 0.0001\}$. We then use this value of λ to compute $\gamma^*_{\mathcal{T},\lambda}(\tilde{\Delta}_u^*(k_{\min}))$ given in Theorem 5.2. This gives us the value of γ.
   d) Consider modified-BPDN. It has two parameters α, γ. We set α as in step 2b. Modified-BPDN is reg-mod-BPDN with $\lambda = 0$. Thus one would expect to be able to just compute $\gamma = \gamma^*_{\mathcal{T},0}$, i.e., use the above approach with $\lambda = 0$. However with $\lambda = 0$, certain matrices used in the computation of $\gamma^*_{\mathcal{T},\lambda}$ become ill-conditioned when $A_\mathcal{T}$ is ill-conditioned (either it is not full rank or is full rank but is ill-conditioned). Hence we instead compute $\gamma = \gamma^*_{\mathcal{T},0.0001}$, i.e., we use $\lambda = 0.0001$ and step 2c.
   e) Consider weighted-ℓ_1. It has three parameters α, γ, τ. We set α as in step 2b. Even though Theorem 5.2 does not apply for it directly, we still just use the heuristic of step 2d for setting γ. As suggested in [47], we set τ equal to the ratio of the estimate of the number of extras to the size of the known support, i.e., $\tau = \frac{|\Delta_e|}{|\mathcal{N}|}$.
   f) Consider KF-ModCS. It has parameters $\alpha, \gamma, \sigma^2_{sys}, \sigma^2_{sys,0}, \sigma^2_{obs}$. We set α as in step 2b. We set γ as in step 2d. We assume $\sigma^2_{sys,0} = \sigma^2_{sys}$ and we set σ^2_{sys} by maximum likelihood estimation, i.e., we compute it as $\frac{1}{|\hat{\mathcal{N}}_2|} \sum_{i \in \hat{\mathcal{N}}_2} (\hat{x}_2 - \hat{x}_1)_i^2$. We use $\|y_2 - A\hat{x}_2\|_2^2/m$ as an estimate for σ^2_{obs}.
   g) For initial sparse recovery, we use BPDN with γ set as suggested in [79]: $\gamma = \max\{10^{-2}\|A_1^T[y_1\ y_2]\|_\infty, \sigma_{\text{obs}}\sqrt{m}\}$.

Lastly consider the add-LS-del support estimation approach. This is most useful for exactly sparse signal sequences. This needs two thresholds α_{add}, α_{del}. We can set $\alpha_{\text{add},t}$ using the following heuristic. It is not hard to see (see [72, Lemma 2.8]) that $(x_t - \hat{x}_{t,\text{add}})_{\mathcal{T}_{\text{add},t}} = (A_{\mathcal{T}_{\text{add},t}}{}' A_{\mathcal{T}_{\text{add},t}})^{-1}[A_{\mathcal{T}_{\text{add},t}}{}' w_t + A_{\mathcal{T}_{\text{add},t}}{}' A_{\Delta_{\text{add},t}}(x_t)_{\Delta_{\text{add},t}}]$. To ensure that this error is not too large, we need $A_{\mathcal{T}_{\text{add},t}}$ to be well conditioned. To ensure this we pick $\alpha_{\text{add},t}$ as the smallest number such that $\sigma_{\min}(A_{\mathcal{T}_{\text{add},t}}) \geq 0.4$. If one could set α_{del} equal to the lower bound on $x_{\min,t} - \|(x_t - \hat{x}_{t,\text{add}})_{\mathcal{T}_{\text{add},t}}\|_\infty$, there will be zero misses. Here $x_{\min,t} = \min_{i \in \mathcal{N}_t} |(x_t)_i|$ is the minimum magnitude nonzero entry. Using this idea, we let $\alpha_{\text{del},t}$ be an estimate of the lower bound of this quantity. As explained in [72, Section VII-A], this leads to $\alpha_{\text{del},t} = 0.7\hat{x}_{\min,t} - \|A^\dagger_{\mathcal{T}_{\text{add},t}}(y_t - A\hat{x}_{t,\text{modcs}})\|_\infty$ where $\hat{x}_{\min,t} = \min_{i \in \hat{\mathcal{N}}_{t-1}} |(\hat{x}_{t-1})_i|$. An alternative approach useful for videos is explained in [54, Algorithm 1].

I. Error stability over time

It is easy to apply the results from the previous two sections to obtain exact recovery conditions or the noisy case error bounds at a given time t. In the noise-free case, these directly also imply error stability at all times. For example, consider dynamic Modified-CS given in Algorithm 7. A direct corollary of Theorem 4.2 is the following.

Corollary 6.1 (dynamic Modified-CS exact recovery): Consider recovering x_t with support $\mathcal{N}_t$ from $y_t := A_t x_t$ using Algorithm 7 with $\epsilon = 0$ and $\alpha = 0$. Let $u_t := |\mathcal{N}_t \setminus \mathcal{N}_{t-1}|$,

% Define soft-thresholding functions:

$$F_{nt}(\phi;c) \triangleq (1+\gamma_{nt}(\phi;c))^{-1}\Big(\frac{\overleftarrow{\psi}_n^{(t)}\phi+\overleftarrow{\xi}_n^{(t)}c}{\overleftarrow{\psi}_n^{(t)}+c}\Big) \quad \text{(D1)}$$

$$G_{nt}(\phi;c) \triangleq (1+\gamma_{nt}(\phi;c))^{-1}\Big(\frac{\overleftarrow{\psi}_n^{(t)}c}{\overleftarrow{\psi}_n^{(t)}+c}\Big)+\gamma_{nt}(\phi;c)|F_{nt}(\phi;c)|^2 \quad \text{(D2)}$$

$$F'_{nt}(\phi;c) \triangleq \tfrac{\partial}{\partial\phi}F_{nt}(\phi,c) = \tfrac{1}{c}G_{nt}(\phi;c) \quad \text{(D3)}$$

$$\gamma_{nt}(\phi;c) \triangleq \Big(\frac{1-\overleftarrow{\pi}_n^{(t)}}{\overleftarrow{\pi}_n^{(t)}}\Big)\Big(\frac{\overleftarrow{\psi}_n^{(t)}+c}{c}\Big) \times \exp\Big(-\Big[\frac{\overleftarrow{\psi}_n^{(t)}|\phi|^2+\overleftarrow{\xi}_n^{(t)*}c\phi+\overleftarrow{\xi}_n^{(t)}c\phi^*-c|\overleftarrow{\xi}_n^{(t)}|^2}{c(\overleftarrow{\psi}_n^{(t)}+c)}\Big]\Big) \quad \text{(D4)}$$

% Begin passing messages . . .

for $t = 1, \ldots, T$, :

% Execute the **(into)** phase . . .

$$\overleftarrow{\pi}_n^{(t)} = \frac{\overrightarrow{\lambda}_n^{(t)}\cdot\overleftarrow{\lambda}_n^{(t)}}{(1-\overrightarrow{\lambda}_n^{(t)})\cdot(1-\overleftarrow{\lambda}_n^{(t)})+\overrightarrow{\lambda}_n^{(t)}\cdot\overleftarrow{\lambda}_n^{(t)}} \quad \forall n \quad \text{(A1)}$$

$$\overleftarrow{\psi}_n^{(t)} = \frac{\overrightarrow{\kappa}_n^{(t)}\cdot\overleftarrow{\kappa}_n^{(t)}}{\overrightarrow{\kappa}_n^{(t)}+\overleftarrow{\kappa}_n^{(t)}} \quad \forall n \quad \text{(A2)}$$

$$\overleftarrow{\xi}_n^{(t)} = \overleftarrow{\psi}_n^{(t)}\cdot\Big(\frac{\overrightarrow{\eta}_n^{(t)}}{\overrightarrow{\kappa}_n^{(t)}}+\frac{\overleftarrow{\eta}_n^{(t)}}{\overleftarrow{\kappa}_n^{(t)}}\Big) \quad \forall n \quad \text{(A3)}$$

% Initialize AMP-related variables . . .

$\forall m: z_{mt}^1 = y_m^{(t)}, \forall n: \mu_{nt}^1 = 0$, and $c_t^1 = 100\cdot\sum_{n=1}^N \psi_n^{(t)}$

% Execute the **(within)** phase using AMP . . .

for $i = 1, \ldots, I$, :

$$\phi_{nt}^i = \textstyle\sum_{m=1}^M A_{mn}^{(t)*} z_{mt}^i + \mu_{nt}^i \quad \forall n \quad \text{(A4)}$$

$$\mu_{nt}^{i+1} = F_{nt}(\phi_{nt}^i; c_t^i) \quad \forall n \quad \text{(A5)}$$

$$v_{nt}^{i+1} = G_{nt}(\phi_{nt}^i; c_t^i) \quad \forall n \quad \text{(A6)}$$

$$c_t^{i+1} = \sigma_e^2 + \tfrac{1}{M}\textstyle\sum_{n=1}^N v_{nt}^{i+1} \quad \text{(A7)}$$

$$z_{mt}^{i+1} = y_m^{(t)} - \boldsymbol{a}_m^{(t)\mathsf{T}}\boldsymbol{\mu}_t^{i+1} + \tfrac{z_{mt}^i}{M}\textstyle\sum_{n=1}^N F'_{nt}(\phi_{nt}^i; c_t^i) \quad \forall m \quad \text{(A8)}$$

end

$$\hat{x}_n^{(t)} = \mu_{nt}^{I+1} \quad \forall n \quad \text{\% Store current estimate of } x_n^{(t)} \quad \text{(A9)}$$

% Execute the **(out)** phase . . .

$$\overrightarrow{\pi}_n^{(t)} = \Big(1+\Big(\frac{1-\overleftarrow{\pi}_n^{(t)}}{\overleftarrow{\pi}_n^{(t)}}\Big)\gamma_{nt}(\phi_{nt}^I; c_t^{I+1})\Big)^{-1} \quad \forall n \quad \text{(A10)}$$

$$(\overrightarrow{\xi}_n^{(t)}, \overrightarrow{\psi}_n^{(t)}) = \begin{cases}(\phi_n^I/\epsilon, c_t^{I+1}/\epsilon^2), & \overleftarrow{\pi}_n^{(t)} \le \tau \\ (\phi_n^I, c_t^{I+1}), & \text{o.w.}\end{cases} \quad \forall n \quad (\epsilon \ll 1) \quad \text{(A11)}$$

% Execute the **(across)** phase forward in time . . .

$$\overrightarrow{\lambda}_n^{(t+1)} = \frac{p_{10}(1-\overrightarrow{\lambda}_n^{(t)})(1-\overrightarrow{\pi}_n^{(t)})+(1-p_{01})\overrightarrow{\lambda}_n^{(t)}\overrightarrow{\pi}_n^{(t)}}{(1-\overrightarrow{\lambda}_n^{(t)})(1-\overrightarrow{\pi}_n^{(t)})+\overrightarrow{\lambda}_n^{(t)}\overrightarrow{\pi}_n^{(t)}} \quad \forall n \quad \text{(A12)}$$

$$\overrightarrow{\eta}_n^{(t+1)} = (1-\alpha)\Big(\frac{\overrightarrow{\kappa}_n^{(t)}\overrightarrow{\psi}_n^{(t)}}{\overrightarrow{\kappa}_n^{(t)}+\overrightarrow{\psi}_n^{(t)}}\Big)\Big(\frac{\overrightarrow{\eta}_n^{(t)}}{\overrightarrow{\kappa}_n^{(t)}}+\frac{\overrightarrow{\xi}_n^{(t)}}{\overrightarrow{\psi}_n^{(t)}}\Big)+\alpha\zeta \quad \forall n \quad \text{(A13)}$$

$$\overrightarrow{\kappa}_n^{(t+1)} = (1-\alpha)^2\Big(\frac{\overrightarrow{\kappa}_n^{(t)}\overrightarrow{\psi}_n^{(t)}}{\overrightarrow{\kappa}_n^{(t)}+\overrightarrow{\psi}_n^{(t)}}\Big)+\alpha^2\rho \quad \forall n \quad \text{(A14)}$$

end

TABLE II

THE DCS-AMP ALGORITHM (FILTERING MODE) FROM [81]. USING NOTATION FROM [81], $n = 1, 2, \ldots, N$ DENOTES THE INDICES OF x_t AND $m = 1, 2, \ldots, M$ DENOTES THE INDICES OF y_t. AS EXPLAINED IN [81], $\epsilon = 10^{-7}$, $\tau = 0.99$, $I = 25$. THE MODEL PARAMETERS ARE ESTIMATED USING THE EM ALGORITHM GIVEN IN TABLE III. THE ALGORITHM OUTPUT AT TIME t IS $\hat{x}^{(t)}$.

% Define key quantities obtained from AMP-MMV at iteration k:

$$\mathrm{E}[s_n^{(t)}|\bar{\boldsymbol{y}}] = \frac{\big(\overrightarrow{\lambda}_n^{(t)}\overrightarrow{\pi}_n^{(t)}\overleftarrow{\lambda}_n^{(t)}\big)}{\big(\overrightarrow{\lambda}_n^{(t)}\overleftarrow{\pi}_n^{(t)}\overleftarrow{\lambda}_n^{(t)}+(1-\overrightarrow{\lambda}_n^{(t)})(1-\overrightarrow{\pi}_n^{(t)})(1-\overleftarrow{\lambda}_n^{(t)})\big)} \quad \text{(Q1)}$$

$$\mathrm{E}[s_n^{(t)}s_n^{(t-1)}|\bar{\boldsymbol{y}}] = p(s_n^{(t)}=1, s_n^{(t-1)}=1|\bar{\boldsymbol{y}}) \quad \text{(Q2)}$$

$$\tilde{v}_n^{(t)} \triangleq \mathrm{var}\{\theta_n^{(t)}|\bar{\boldsymbol{y}}\} = \Big(\frac{1}{\overrightarrow{\kappa}_n^{(t)}}+\frac{1}{\overrightarrow{\psi}_n^{(t)}}+\frac{1}{\overleftarrow{\kappa}_n^{(t)}}\Big)^{-1} \quad \text{(Q3)}$$

$$\tilde{\mu}_n^{(t)} \triangleq \mathrm{E}[\theta_n^{(t)}|\bar{\boldsymbol{y}}] = \tilde{v}_n^{(t)}\cdot\Big(\frac{\overrightarrow{\eta}_n^{(t)}}{\overrightarrow{\kappa}_n^{(t)}}+\frac{\overrightarrow{\xi}_n^{(t)}}{\overrightarrow{\psi}_n^{(t)}}+\frac{\overleftarrow{\eta}_n^{(t)}}{\overleftarrow{\kappa}_n^{(t)}}\Big) \quad \text{(Q4)}$$

$v_n^{(t)} \triangleq \mathrm{var}\{x_n^{(t)}|\bar{\boldsymbol{y}}\}$ % See (A6) of Table II

$\mu_n^{(t)} \triangleq \mathrm{E}[x_n^{(t)}|\bar{\boldsymbol{y}}]$ % See (A5) of Table II

% EM update equations:

$$\lambda^{k+1} = \tfrac{1}{N}\textstyle\sum_{n=1}^N \mathrm{E}[s_n^{(1)}|\bar{\boldsymbol{y}}] \quad \text{(E1)}$$

$$p_{01}^{k+1} = \frac{\sum_{t=2}^T\sum_{n=1}^N \mathrm{E}[s_n^{(t-1)}|\bar{\boldsymbol{y}}]-\mathrm{E}[s_n^{(t)}s_n^{(t-1)}|\bar{\boldsymbol{y}}]}{\sum_{t=2}^T\sum_{n=1}^N \mathrm{E}[s_n^{(t-1)}|\bar{\boldsymbol{y}}]} \quad \text{(E2)}$$

$$\zeta^{k+1} = \Big(\tfrac{N(T-1)}{\rho^k}+\tfrac{N}{(\sigma^2)^k}\Big)^{-1}\Big(\tfrac{1}{(\sigma^2)^k}\textstyle\sum_{n=1}^N\tilde{\mu}_n^{(1)} + \sum_{t=2}^T\sum_{n=1}^N\tfrac{1}{\alpha^k\rho^k}\big(\tilde{\mu}_n^{(t)}-(1-\alpha^k)\tilde{\mu}_n^{(t-1)}\big)\Big) \quad \text{(E3)}$$

$$\alpha^{k+1} = \tfrac{1}{4N(T-1)}\Big(\mathfrak{b}-\sqrt{\mathfrak{b}^2+8N(T-1)\mathfrak{c}}\Big) \quad \text{(E4)}$$

where:

$$\mathfrak{b} \triangleq \tfrac{2}{\rho^k}\textstyle\sum_{t=2}^T\sum_{n=1}^N \mathfrak{Re}\{E[\theta_n^{(t)*}\theta_n^{(t-1)}|\bar{\boldsymbol{y}}]\} - \mathfrak{Re}\{(\tilde{\mu}_n^{(t)}-\tilde{\mu}_n^{(t-1)})^*\zeta^k\} - \tilde{v}_n^{(t-1)} - |\tilde{\mu}_n^{(t-1)}|^2$$

$$\mathfrak{c} \triangleq \tfrac{2}{\rho^k}\textstyle\sum_{t=2}^T\sum_{n=1}^N \tilde{v}_n^{(t)} + |\tilde{\mu}_n^{(t)}|^2 + \tilde{v}_n^{(t-1)} + |\tilde{\mu}_n^{(t-1)}|^2 - 2\mathfrak{Re}\{E[\theta_n^{(t)*}\theta_n^{(t-1)}|\bar{\boldsymbol{y}}]\}$$

$$\rho^{k+1} = \tfrac{1}{(\alpha^k)^2N(T-1)}\textstyle\sum_{t=2}^T\sum_{n=1}^N \tilde{v}_n^{(t)} + |\tilde{\mu}_n^{(t)}|^2 + (\alpha^k)^2|\zeta^k|^2 - 2(1-\alpha^k)\mathfrak{Re}\{E[\theta_n^{(t)*}\theta_n^{(t-1)}|\bar{\boldsymbol{y}}]\} - 2\alpha^k\mathfrak{Re}\{\tilde{\mu}_n^{(t)*}\zeta^k\} + 2\alpha^k(1-\alpha^k)\mathfrak{Re}\{\tilde{\mu}_n^{(t-1)*}\zeta^k\} + (1-\alpha^k)(\tilde{v}_n^{(t-1)}+|\tilde{\mu}_n^{(t-1)}|^2) \quad \text{(E5)}$$

$$(\sigma_e^2)^{k+1} = \tfrac{1}{TM}\Big(\textstyle\sum_{t=1}^T\|\boldsymbol{y}^{(t)}-\boldsymbol{A}\boldsymbol{\mu}^{(t)}\|^2+\mathbf{1}_N^T\boldsymbol{v}^{(t)}\Big) \quad \text{(E6)}$$

TABLE III

EM UPDATE EQUATIONS [81, TABLE III] FOR THE SIGNAL MODEL PARAMETERS USED FOR THE DCS-AMP ALGORITHM FROM [81]. AS EXPLAINED IN [81], AT ITERATION k, THE DCS-AMP ALGORITHM FROM TABLE II IS FIRST RUN AND ITS OUTPUTS ARE USED FOR THE k-TH EM ITERATION STEPS GIVEN ABOVE. AS EXPLAINED IN [81, SECTION VI], FOR THE FIRST EM ITERATION, ONE INITIALIZES α USING EQUATION (12) OF [81] WHILE THE OTHER PARAMETERS ARE INITIALIZED USING THE APPROACH OF [85, SECTION V].

Algorithm 7 Dynamic Modified-CS -noisy

At $t = 0$: Solve BP-noisy with sufficient measurements, i.e., compute $\hat{x}_0$ as the solution of $\min_b \|b\|_1$ s.t. $\|y - A_0 b\|_2 \le \epsilon$ and compute its support: $\hat{\mathcal{N}}_0 = \{i : |(\hat{x}_0)_i| > \alpha\}$.

For $t > 0$ do

1) Set $\mathcal{T} = \hat{\mathcal{N}}_{t-1}$
2) *Modified-CS -noisy.* Compute $\hat{x}_t$ as the solution of
$$\min_b \|b_{\mathcal{T}^c}\|_1 \text{ s.t. } \|y - A_t b\|_2 \le \epsilon$$
3) *Support Estimation - Simple Thresholding.*
$$\hat{\mathcal{N}}_t = \{i : |(\hat{x}_t)_i| > \alpha\} \quad (28)$$

Dynamic modified-CS-Add-LS-del: replace the support estimation step by the Add-LS-Del procedure of (23).

$e_t = |\mathcal{N}_{t-1} \setminus \mathcal{N}_t|$ and $s_t = |\mathcal{N}_t|$. If $\delta_{2s_0}(A_0) \le 0.2$ and if, for all $t > 0$, $\delta_{s_t+u_t+e_t}(A_t) \le 0.2$, then $\hat{x}_t = x_t$ (exact recovery is achieved) at all times t.

Similar corollaries can be obtained for Theorem 4.3 and Theorem 4.5 for weighted-ℓ_1. For the noisy case, the error bounds are functions of $|\mathcal{T}_t|, |\Delta_{u,t}|$ (or equivalently of $|\mathcal{N}_t|, |\Delta_{u,t}|, |\Delta_{e,t}|$). The bounds are small as long as $|\Delta_{e,t}|$ and $|\Delta_{u,t}|$ are small for a given $|\mathcal{N}_t|$. But, unless we obtain conditions to ensure time-invariant bounds on $|\Delta_{e,t}|$ and $|\Delta_{u,t}|$, the size of these sets may keep growing over time resulting in instability. We obtain these conditions in the error stability over time results described next. So far, such results exist only for LS-CS [28, Theorem 2] and for dynamic modified-CS-noisy summarized in Algorithm 7. [72], [86], [87]. We give here two sample results for the latter from Zhan and Vaswani [72]. The first result below does not assume anything about how signal values change while the second assumes a realistic signal change model.

Theorem 6.2 (modified-CS error stability: no signal model [72]): Consider recovering x_t's from y_t satisfying (8) using Algorithm 7. Assume that the support size of x_t is bounded by s and that there are at most s_a additions and at most s_a removals at all times. If

1) *(support estimation threshold)* $\alpha = 7.50\epsilon$,
2) *(number of measurements)* $\delta_{s+6s_a}(A_t) \le 0.207$,
3) *(number of small magnitude entries)* $|\{i \in \mathcal{N}_t : |(x_t)_i| \le \alpha + 7.50\epsilon\}| \le s_a$
4) *(initial time)* at $t = 0$, n_0 is large enough to ensure that

$$|\mathcal{N}_0 \setminus \hat{\mathcal{N}}_0| = 0,\ |\hat{\mathcal{N}}_0 \setminus \mathcal{N}_0| = 0,$$

then for all t,

- $|\Delta_{u,t}| \le 2s_a$, $|\Delta_{e,t}| \le s_a$, $|\mathcal{T}_t| \le s$,
- $\|x_t - \hat{x}_t\| \le 7.50\epsilon$,
- $|\mathcal{N}_t \setminus \hat{\mathcal{N}}_t| \le s_a$, $|\hat{\mathcal{N}}_t \setminus \mathcal{N}_t| = 0$, $|\tilde{\mathcal{T}}_t| \le s$

The above result is the most general, but it does not give us practical models on signal change that would ensure the required upper bound on the number of small magnitude entries. Next we give one realistic model on signal change followed by a stability result for it. Briefly, this model says the following. At any time t, x_t is a sparse vector with support set $\mathcal{N}_t$ of size s or less. At most s_a elements get added to the support at each time t and at most s_a elements get removed from it. At time t, a new element j gets added at an initial magnitude a_j. Its magnitude increases for the next d_j time units with $d_j \ge d_{\min} > 0$. Its magnitude increase at time τ is $r_{j,\tau}$. Also, at each time t, at most s_a elements out of the "large elements" set (defined in the signal model) leave the set and begin to decrease. These elements keep decreasing and get removed from the support in at most b time units. In the model as stated above, we are implicitly allowing an element j to get added to the support at most once. In general, j can get added, then removed and then added again. To allow for this, we let $\mathrm{addtimes}_j$ be the *set* of time instants at which j gets added; we replace a_j by $a_{j,t}$ and we replace d_j by $d_{j,t}$ (both of which are nonzero only for $t \in \mathrm{addtimes}_j$).

Model 6.3 (Model on signal change over time (parameter ℓ)): Assume the following model on signal change

1) At $t = 0$, $|\mathcal{N}_0| = s_0$.
2) At time t, $s_{a,t}$ elements are added to the support set. Denote this set by $\mathcal{A}_t$. A new element j gets added to the support at an initial magnitude $a_{j,t}$ and its magnitude increases for at least the next $d_{\min}$ time instants. At time τ (for $t < \tau \le t + d_{\min}$), the magnitude of element j increases by $r_{j,\tau}$.
   Note: $a_{j,t}$ is nonzero only if element j got added at time t, for all other times, we set it to zero.
3) For a given scalar ℓ, define the "large set" as
$$\mathcal{L}_t(\ell) := \{j \notin \cup_{\tau=t-d_{\min}+1}^{t} \mathcal{A}_\tau : |(x_t)_j| \ge \ell\}.$$
   Elements in $\mathcal{L}_{t-1}(\ell)$ either remain in $\mathcal{L}_t(\ell)$ (while increasing or decreasing or remaining constant) or decrease enough to leave $\mathcal{L}_t(\ell)$. At time t, we assume that $s_{d,t}$ elements out of $\mathcal{L}_{t-1}(\ell)$ decrease enough to leave it. All these elements continue to keep decreasing and become zero (removed from support) within at most b time units.
4) At all times t, $0 \le s_{a,t} \le s_a$, $0 \le s_{d,t} \le \min\{s_a, |\mathcal{L}_{t-1}(\ell)|\}$, and the support size, $s_t := |\mathcal{N}_t| \le s$ for constants s and s_a such that $s + s_a \le m$.

Notice that an element j could get added, then removed and added again later. Let

$$\mathrm{addtimes}_j := \{t : a_{j,t} \ne 0\}$$

denote the set of time instants at which the index j got added to the support. Clearly, $\mathrm{addtimes}_j = \emptyset$ if j never got added. Let

$$a_{\min} := \min_{j:\mathrm{addtimes}_j \ne \emptyset} \min_{t \in \mathrm{addtimes}_j, t>0} a_{j,t}$$

denote the minimum of $a_{j,t}$ over all elements j that got added at $t > 0$. We are excluding coefficients that never got added and those that got added at $t = 0$. Let

$$r_{\min}(d) := \min_{j:\mathrm{addtimes}_j \ne \emptyset} \min_{t \in \mathrm{addtimes}_j, t>0} \min_{\tau \in [t+1, t+d]} r_{j,\tau}$$

denote the minimum, over all elements j that got added at $t > 0$, of the minimum of $r_{j,\tau}$ over the first d time instants after j got added.

Theorem 6.4 (dynamic Modified-CS error stability): [72] Consider recovering x_t's from y_t satisfies (8) using Algorithm 7. Assume that Model 6.3 on x_t holds with

$$\ell = a_{\min} + d_{\min} r_{\min}(d_{\min})$$

where $a_{\min}$, $r_{\min}$ and the set $\mathrm{addtimes}_j$ are defined above. If there exists a $d_0 \le d_{\min}$ such that the following hold:

1) (algorithm parameters) $\alpha = 7.50\epsilon$,
2) (number of measurements) $\delta_{s+3(b+d_0+1)s_a}(A_t) \le 0.207$,
3) (initial magnitude and magnitude increase rate)
$$\min\{\ell, \min_{j:\mathrm{addtimes}_j \ne \emptyset} \min_{t \in \mathrm{addtimes}_j} (a_{j,t} + \sum_{\tau=t+1}^{t+d_0} r_{j,\tau})\} > \alpha + 7.50\epsilon,$$
4) at $t = 0, n_0$ is large enough to ensure that $|\mathcal{N}_0 \setminus \hat{\mathcal{N}}_0| = 0$, $|\hat{\mathcal{N}}_0 \setminus \mathcal{N}_0| = 0$,

then, for all t,

- $|\Delta_{u,t}| \le bs_a + d_0 s_a + s_a$, $|\Delta_{e,t}| \le s_a$, $|\mathcal{T}_t| \le s$,
- $\|x_t - \hat{x}_t\| \le 7.50\epsilon$,
- $|\mathcal{N}_t \setminus \hat{\mathcal{N}}_t| \le bs_a + d_0 s_a$, $|\hat{\mathcal{N}}_t \setminus \mathcal{N}_t| = 0$, $|\tilde{\mathcal{T}}_t| \le s$

Results similar to the above two results also exist for dynamic modified-CS-Add-LS-Del [72]. Their main advantage is that they require a weaker condition 3).

1) Discussion: Notice that s is a bound on the support size at any time. As long as the number of new additions or removals, $s_a \ll s$, i.e., slow support change holds, the above result shows that the worst case number of misses or extras is also small compared to the support size. This makes it a meaningful result. The reconstruction error bound is also small compared to the signal energy as long as the signal-to-noise ratio is high enough (ϵ^2 is small compared to $\|x_t\|^2$). Observe that both the above results need a bound on the RIC of A_t of order $s + \mathcal{O}(s_a)$. On the other hand, BP-noisy needs the same bound on the RIC of A_t of order $2s$ (see Theorem 2.4). This is stronger when $s_a \ll s$ (slow support change).

As explained in [72], Model 6.3 allows for both slow and fast signal magnitude increase or decrease. Slow magnitude increase and decrease would happen, for example, in an imaging problem when one object slowly morphs into another with gradual intensity changes. Or, in case of brain regions becoming "active" in response to stimuli, the activity level gradually increases from zero to a certain maximum value within a few milliseconds (10-12 frames of fMRI data), and similarly the "activity" level decays to zero within a few milliseconds. In both of the above examples, a new coefficient will get added to the support at time t at a small magnitude $a_{j,t}$ and increase by $r_{j,t}$ per unit time for sometime after that. A similar thing will happen for the decay to zero of the brains

activity level. On the other hand, the model also allows support changes resulting from motion of objects, e.g., translation. In this case, the signal magnitude changes will typically not be slow. As the object moves, a set of new pixels enter the support and another set leave. The entering pixels may have large enough pixel intensity and their intensity may never change. For our model, this means that the pixel enters the support at a large enough initial magnitude $a_{j,t}$ but its magnitude never changes i.e., $r_{j,t} = 0$ for all t. If all pixels exit the support without their magnitude first decreasing, then $b = 1$.

The only thing that the above result requires is that (i) for any element j that is added, either $a_{j,t}$ is large enough or $r_{j,t}$ is large enough for the initial few (d_0) time instants so that condition 3) holds; and (ii) a decaying coefficient decays to zero within a short delay, b. Condition (i) ensures that every newly added support element gets detected either immediately or within a finite delay; while (ii) ensures removal within finite delay of a decreasing element. For the moving object case, this translates to requiring that $a_{j,t}$ be large enough. For the morphing object example or the fMRI activation example, this translates to requiring that $r_{j,t}$ be large enough for the first d_0 frames after index j gets added and b be small enough.

VII. Tracking-based and adaptive-filtering-based solutions for recursive dynamic CS

As explained in Section III-C, it is possible to use ideas from the tracking literature or the adaptive filtering literature along with a sparsity constraint to get solutions for the recursive dynamic CS problem. These only use the slow signal value change assumption and sparsity without explicitly using slow support change. We describe the tracking-based solutions next followed by the adaptive filtering based solutions.

A. Tracking-based recursive dynamic CS solutions

We discuss here two tracking-based solutions.

1) BP-residual: To explain the idea, we use BP-noisy. The exact same approach can also be applied to get BPDN-residual, IHT-residual or OMP-residual. BP-residual replaces y_t by the measurement residual $y_t - A\hat{x}_{t-1}$ in the BP or the BP-noisy program, i.e., it computes

$$\hat{x}_t = \hat{x}_{t-1} + \arg\min_{\beta} [\|\beta\|_1 \text{ s.t. } \|y_t - A\hat{x}_{t-1} - A\beta\|_2 \le \epsilon] \quad (29)$$

with setting $\epsilon = 0$ in the noise-free case. This is related to BP on observation differences idea used in [43]. Observe that this is using the assumption that the difference $(x_t - x_{t-1})$ is small. However, if x_t and x_{t-1} are k-sparse, then $(x_t - x_{t-1})$ will also be at least k-sparse unless the difference is exactly zero along one or more coordinates. There are very few practical situations where one can hope to get perfect prediction along a few dimensions. One possible example is the case of coarsely quantized x_t's. In the results known so far, the number of measurements required for exact sparse recovery depend only on the support size of the sparse vector, e.g., see [9], [25], [26], [29]. Thus, when using BP-residual, this number will be as much or more than what BP needs. This is also observed in Monte Carlo based computations of the probability of exact recovery in Fig. 3. As can be seen, both BP and BP-residual need the same n for exact recovery with Monte Carlo probability equal to one. This number is much larger than what modified-CS and weighted-ℓ_1, that exploit slow support change, need. On the other hand, as we will see in Fig. 5, for compressible signal sequences, its performance is not much worse than that of approaches that do explicitly use slow support change.

Algorithm 8 PM-CS-KF (Algorithm 1 of [88])

1: Prediction

$$\begin{aligned} \hat{z}_{k+1|k} &= A\hat{z}_{k|k} \\ P_{k+1|k} &= AP_{k|k}A^T + Q_k \end{aligned} \quad (30)$$

2: Measurement Update

$$\begin{aligned} K_k &= P_{k+1|k}H'^T \left(H'P_{k+1|k}H'^T + R_k\right)^{-1} \\ \hat{z}_{k+1|k+1} &= \hat{z}_{k+1|k} + K_k(y_k - H'\hat{z}_{k+1|k}) \\ P_{k+1|k+1} &= (I - K_kH')P_{k+1|k} \end{aligned} \quad (31)$$

3: CS Pseudo Measurement: Let $P^1 = P_{k+1|k+1}$ and $\hat{z}^1 = \hat{z}_{k+1|k+1}$.

4: for $\tau = 1, 2, \cdots, N_\tau - 1$ iterations do

5:

$$\begin{aligned} \bar{H}_\tau &= [sign(\hat{z}^\tau(1)), \cdots, sign(\hat{z}^\tau(n))] \\ K^\tau &= P^\tau \bar{H}_\tau^T \left(\bar{H}_\tau P^\tau \bar{H}_\tau^T + R_\epsilon\right)^{-1} \\ \hat{z}^{\tau+1} &= (I - K^\tau \bar{H}_\tau)\hat{z}^\tau \\ P^{\tau+1} &= (I - K^\tau \bar{H}_\tau)P^\tau. \end{aligned} \quad (32)$$

6: end for

7: Set $P_{k+1|k+1} = P^{N_\tau}$ and $\hat{z}_{k+1|k+1} = \hat{z}^{N_\tau}$.

2) Pseudo-measurement based CS-KF (PM-CS-KF): In [88], Carmi et al. introduced the pseudo-measurement based CS-KF (PM-CS-KF) algorithm. It uses an indirect method called the pseudo-measurement (PM) technique [89] to include the sparsity constraint while trying to minimize the estimation error in the KF update step. To be precise, it uses PM to approximately solve the following

$$\min_{\hat{x}_{k|k}} \mathbb{E}_{x_k|y_1,y_2,\ldots y_k}[\|x_k - \hat{x}_{k|k}\|_2^2] \text{ s.t. } \|\hat{x}_{k|k}\|_1 \le \epsilon$$

The idea of PM is to replace the constraint $\|\hat{x}_{k|k}\|_1 \le \epsilon$ by a linear equation of the form

$$\tilde{H}x_k - \epsilon = 0$$

where $\tilde{H} = \text{diag}\left(\text{sgn}((x_k)_1), \text{sgn}((x_k)_2), \ldots \text{sgn}((x_k)_m)\right)$ and ϵ serves as measurement noise. It then uses an extended KF approach iterated multiple times to enforce the sparsity constraint. As explained in [88], the covariance of the pseudo noise ϵ, R_ϵ, is a tuning parameter that can be chosen in the same manner as the process noise covariance is determined for an extended KF. The complete algorithm [88, Algorithm 1] is summarized in Algorithm 8 (with permission from the authors). The code is not posted on the authors' webpages but is available from this article's webpage http://www.ece.iastate.edu/~namrata/RecReconReview.html.

B. Sparsity-aware adaptive filtering and its application to recursive dynamic CS

A problem related to the recursive dynamic CS problem is that of sparsity-aware adaptive filtering. This is, in fact, a

special case of a more general problem of recovering a single sparse signal from sequentially arriving measurements. The general problem is as follows. An unknown sparse vector $\mathbf{h}$ needs to be recovered from sequentially arriving measurements

$$d_i := \mathbf{a}_i{}'\mathbf{h} + v_i, i = 1, 2, \ldots, n \tag{33}$$

To connect this with our earlier notation, $y := [d_1, d_2, \ldots d_n]'$, $A := [\mathbf{a}_1, \mathbf{a}_2, \ldots \mathbf{a}_n]'$, $x := \mathbf{h}$ and $w := [v_1, v_2, \ldots v_n]'$. We use different notation in this section so as to keep it consistent with what is used in the adaptive filtering literature. Let $\hat{\mathbf{h}}^i$ denote the estimate of $\mathbf{h}$ based on the first i measurements. The goal is to obtain $\hat{\mathbf{h}}^i$ from $\hat{\mathbf{h}}^{i-1}$ without re-solving the sparse recovery problem. Starting with the works of Malioutov et al. [90] and Garrigues et al. [91], various homotopy based solutions have been introduced in recent literature to do this [92], [93]. Some of these works also provide a stopping criterion that tells the user when to stop taking new measurements.

A special case of the above problem is sparsity-aware adaptive filtering for system identification (e.g., estimating an unknown communication channel). In this case, a known discrete time signal $x[i]$ is sent through an unknown linear time-invariant system (e.g., a communication channel) with impulse response denoted by $\mathbf{h}$. The impulse response vector $\mathbf{h}$ is assumed to be sparse. One can see the output signal $d[i]$. The goal is to keep updating the estimates of $\mathbf{h}$ on-the-fly so that the output after passing $x[i]$ through the estimated system/channel is close to the observed output $d[i]$. Let $x[i]$ denote a given input sequence and define the vector

$$\mathbf{x}[i] := [x[i], x[i-1], x[i-2], \ldots x[i-m+1]]'$$

Then sparsity-aware adaptive filtering involves recovering $\mathbf{h}$ from sequentially arriving $d[i]$ satisfying (33) with $d_i = d[i]$ and $\mathbf{a}_i = \mathbf{x}[i]$. This problem has been studied in a sequence of recent works. One of the first solutions to this problem, called zero-attracting least mean squares (ZA-LMS) [94], [95], modifies the standard LMS algorithm by including an ℓ_1-norm constraint in the cost function. Let $\hat{\mathbf{h}}^i$ denote the estimate of $\mathbf{h}$ at time i (based on the past i measurements) and let

$$e[i] := d[i] - \mathbf{a}_i{}'\hat{\mathbf{h}}^i.$$

with $\mathbf{a}_i = \mathbf{x}[i]$. At time i, regular LMS takes one gradient descent step towards minimizing $L(i) = 0.5e[i]^2$. ZA-LMS does almost the same thing for $L(i) = 0.5e[i]^2 + \gamma\|\hat{\mathbf{h}}^i\|_1$. Let μ denote the step size. Then, ZA-LMS computes $\hat{\mathbf{h}}^{i+1}$ from $\hat{\mathbf{h}}^i$ using

$$\hat{\mathbf{h}}^{i+1} = \hat{\mathbf{h}}^i + \mu e[i]\mathbf{a}_i - \mu\gamma\,\mathrm{sgn}(\hat{\mathbf{h}}^i)$$

where $\mathrm{sgn}(z)$ is a component-wise sign function: $(\mathrm{sgn}(z))_i = z_i/|z_i|$ if $z_i \neq 0$ and $(\mathrm{sgn}(z))_i = 0$ otherwise. The last term in the above update attracts the coefficients of the estimate of $\mathbf{h}$ to zero. We should point out here that the term $\|\hat{\mathbf{h}}^i\|_1$ is not differentiable and hence it does not have gradient. ZA-LMS is actually replacing its gradient by one possible vector from its sub-gradient set. The work of Jin et al. [95] modified the ZA-LMS algorithm by including a differentiable approximation to the ℓ_0 norm to replace the ℓ_1 norm used in the cost function for ZA-LMS. In particular they used $L(i) = 0.5e[i]^2 + \gamma\sum_{k=1}^m(1-\exp(-\alpha|\mathbf{h}_k|))$ and then derived a similar algorithm. These algorithms were analyzed in [96].

In later work by Babadi et al. [97], the SPARLS algorithm was developed for solving the sparse recursive least squares (RLS) problem. This was done for RLS with an exponential forgetting factor. At time i, RLS with an exponential forgetting factor computes $\hat{\mathbf{h}}^i$ as the minimizer of $L(i) = \sum_{j=1}^i \lambda^{i-j}e[j]^2$ with $e[j] := d[j] - \mathbf{a}_j'\hat{\mathbf{h}}^i$ and $\mathbf{a}_j = \mathbf{x}[j]$. By defining the vector $\mathbf{d}[i] := [d_1, d_2, \ldots d_i]'$, the matrix $\mathbf{X}[i] := [\mathbf{a}_1, \mathbf{a}_2, \ldots \mathbf{a}_i]'$, and a diagonal matrix $\mathbf{D}[i] := diag(\lambda^{i-1}, \lambda^{i-2}, \ldots 1)$, $L(i)$ can be rewritten as $L(i) = \|\mathbf{D}[i]^{1/2}\mathbf{d}[i] - \mathbf{D}[i]^{1/2}\mathbf{X}[i]\hat{\mathbf{h}}^i\|_2^2$. In the above definitions, we use $\mathbf{a}_i = \mathbf{x}[i]$. This is now a regular least squares (LS) problem. When solved in a recursive fashion, we get the RLS algorithm for regular adaptive filtering. Sparse RLS adds an ℓ_1 term to the cost function. It solves

$$\min_{\hat{\mathbf{h}}^i} \frac{1}{2\sigma^2}\|\mathbf{D}[i]^{1/2}\mathbf{d}[i] - \mathbf{D}[i]^{1/2}\mathbf{X}[i]\hat{\mathbf{h}}^i\|_2^2 + \gamma\|\hat{\mathbf{h}}^i\|_1 \tag{34}$$

Instead of solving the above using conventional convex programming techniques, the authors of [97] developed a low-complexity Expectation Minimization (EM) algorithm motivated by an earlier work of Figueirado and Nowak [98]. Another parallel work by Angelosante et al. [99] also starts with the RLS cost function and adds an ℓ_1 cost to it to get the sparse RLS cost function. Their solution approach involves developing an online version of the cyclic coordinate descent algorithm. They, in fact, developed sparse RLS algorithms with various types of forgetting factors.

While all of the above algorithms are designed for a recovering a fixed sparse vector $\mathbf{h}$, they also often work well for situations where $\mathbf{h}$ is slow time-varying [95]. In fact, this is how they relate to the problem studied in this article. However, all the theoretical results for sparse adaptive filtering are for the case where $\mathbf{h}$ is fixed. For example, the result from [97] for SPARLS states the following.

Theorem 7.1 ([97, Theorem 1]): Assume that $\mathbf{h}$ is fixed and that the input sequence $x[i]$ and the output sequence $d[i]$ are realizations of a jointly stationary random process. Then the estimates $\hat{\mathbf{h}}^i$ generated by the SPARLS algorithm converge almost surely to the unique minimizer of (34).

Another class of approaches for solving the above problem involves the use of the set theoretic estimation technique [100]–[102]. Instead of recursively trying to minimize a cost function, the goal, in this case, is to find a set of solutions that are in agreement with the available measurements and the sparsity constraints. For example, the work of [101] develops an algorithm that finds all vectors $\mathbf{h}$ that belong to the intersection of the sets $S_j(\epsilon) := \{\mathbf{h} : |\mathbf{x}[j]'\mathbf{h} - d[j]| \leq \epsilon\}$ for all $j \leq i$ and the weighted-ℓ_1 ball $B(\delta) := \{\mathbf{h} : \sum_{k=1}^m \omega_k|\mathbf{h}_k| \leq \delta\}$. Thus at time i, it finds the set of all $\mathbf{h}$'s that belong to $\cap_{j=1}^i S_j(\epsilon) \cap B(\delta)$.

Other work on the topic includes [103] (Kalman filter based sparse adaptive filter), [104] ("proportionate-type algorithms" for online sparsity-aware system identification problems), [105] (combine sparsity-promoting schemes with data-selection mechanisms), [106] (greedy sparse RLS), [107] (re-

cursive $\ell_{1,\infty}$ group LASSO), [108] (variational Bayes framework for sparse adaptive filtering) and [109], [110] (distributed adaptive filtering).

1) Application to recursive dynamic CS: While the approaches described above are designed for a single sparse vector $\mathbf{h}$, as mentioned above, they can also be used to track a time sequence of sparse vectors that are slow time-varying. Instead of using a time-sequence of vector measurements (as in Sec. III), in this case, one uses a single sequence of scalar measurements indexed by i. To use these algorithms for recursive dynamic CS, for time t, the index i will vary from $nt+1$ to $nt+n$. For a given i, define $t(i) = \lfloor \frac{i}{n} \rfloor$ and $k(i) = i - nt$. Then one can use the above algorithms with the mapping $d_i = (y_{t(i)})_{k(i)}$ and with $\mathbf{a}_i$ being the $k(i)$-th row of the matrix A_t. Also, one assumes $\mathbf{h}_i = x_{t(i)}$, i.e., $\mathbf{h}_i$ is is equal to x_t for all $i = nt+1, nt+2, \dots nt+n$. The best estimate of x_t is then given by $\hat{\mathbf{h}}^{nt+n}$, i.e., we use $\hat{x}_t = \hat{\mathbf{h}}^{nt+n}$. We use this formulation to develop ZA-LMS to solve our problem and show experiments with it in the next section.

The advantage of the above approach is that it is very quick and needs very little memory. Its computational complexity is linear in the signal length. The disadvantage is that it is not leveraging the slow support change assumption explicitly and hence does not work well in situations where support changes by a little at every time or often enough. This can be seen from Fig. 4. Moreover, all the theoretical guarantees are for the case where $\mathbf{h}$ is *fixed*.

VIII. Numerical Experiments

We report three sets of experiments. The first studies the noise-free case and compares algorithms that only exploit slow support change of the sparse signal sequence. As explained earlier, this problem can be reformulated as one of sparse recovery with partial support knowledge. In the noise-free case, once the static problem is solved, so is the dynamic one. So we only simulate the static problem. In the second experiment, we study the noisy case and the dynamic problem for a simulated exactly sparse signal sequence and random Gaussian measurements. In the third experiment, we study the same problem but for a real image sequence (the larynx MRI sequence shown in Fig. 1) and simulated MRI measurements. This image sequence is only approximately sparse in the wavelet domain. In the second and third experiment, we compare all types of algorithms - those that exploit none, one or both of slow support change and slow signal value change.

A. Sparse recovery with partial support knowledge: phase transition plots for comparing n needed for exact recovery

We compare BP and BP-residual with all the approaches that only use partial support knowledge – Modified-CS and weighted-ℓ_1 – using phase transition plots shown in Fig. 3. BP-residual is an approach that is motivated by tracking literature and it uses signal value knowledge but not support knowledge. We include this in our comparison to demonstrate that it cannot use a smaller n than BP for exact recovery with probability one. For a given support size, s, number of misses in the support knowledge, u, number of extras, e, and number of measurements, n, we use Monte Carlo to estimate the probability of exact recovery of the various approaches. We generated the true support $\mathcal{N}$ of size s uniformly at random from $\{1, 2, \dots, m\}$. The nonzero entries of x were generated from a Gaussian $\mathcal{N}(0, \sigma_x^2)$ distribution. The support knowledge $\mathcal{T}$ was generated as $\mathcal{T} = \mathcal{N} \cup \Delta_e \setminus \Delta_u$ where Δ_e is generated as a set of size e uniformly at random from $\mathcal{N}^c$ and Δ_u was generated as a set of size of u uniformly at random from $\mathcal{N}$. We generated the observation vector $y = Ax$ where A is an $n \times m$ random Gaussian matrix. Since BP-residual uses signal value knowledge, for it, we generate a "signal value knowledge" $\hat{\mu}$ as follows. Generate $\hat{\mu}_{\mathcal{T}} = x_{\mathcal{T}} + \nu_{\mathcal{T}}$ with $\nu \sim \mathcal{N}(0, \sigma_\nu^2 I)$, and set $\hat{\mu}_{\mathcal{T}^c} = 0$. We generated two types of prior signal knowledge, the good prior case with $\sigma_\nu^2 = 0.0001\sigma_x^2$ and the bad prior case with $\sigma_\nu^2 = \sigma_x^2$.

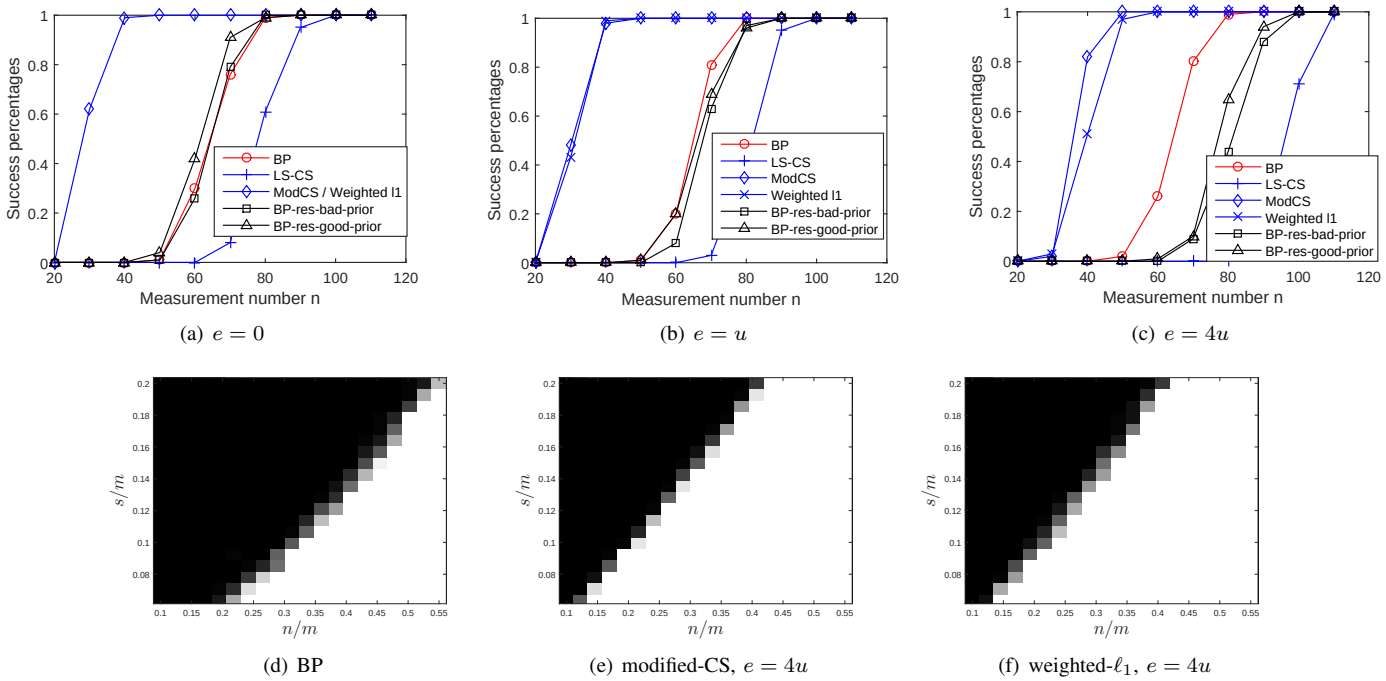
Modified-CS solved (12) which has no parameters. Weighted-ℓ_1 solved (14) with $\tau = e/s$ [47]. BP-residual computed $\hat{x} = \hat{\mu} + [\arg\min_b \|b\|_1 \text{ s.t. } y - A\hat{\mu} = Ab]$, where $\hat{\mu}$ was generated as above. For LS-CS, we did the same thing but $\hat{\mu}$ was now $I_{\mathcal{T}} A_{\mathcal{T}}^{\dagger} y$ (LS estimate on $\mathcal{T}$). All convex programs were solved using CVX.

For the first row figures of Fig. 3, we used $m = 200$, $s = 0.1m$, $u = 0.1s$, $\sigma_x^2 = 5$ and three values of e: $e = 0$ (Fig. 3(a)) and $e = u$ (Fig. 3(b)) and $e = 4u$ (Fig. 3(c)). The plot varies n and plots the Monte Carlo estimate of the probability of exact recovery. The probability of exact recovery was computed by generating 100 realizations of the data and counting the number of times x was exactly recovered by a given approach. We say that x is exactly recovered if $\frac{\|x-\hat{x}\|}{\|x\|} < 10^{-6}$ (precision used by CVX). We show three cases, $e = 0$, $e = u$ and $e = 4u$ in Fig. 3. In all cases we observe the following. (1) LS-CS needs more measurements for exact recovery than either of BP or BP-residual. This is true even in the $e = 0$ case (and this is something we are unable to explain). (2) BP-residual needs as many or more measurements as BP to achieve exact recovery with (Monte Carlo) probability one. This is true even when very good prior knowledge is provided to BP-residual. (3) Weighted-ℓ_1 and modified-CS significantly outperform all other approaches – they need a significantly smaller n for exact recovery with (Monte Carlo) probability one. (4) From this set of simulations, it is hard to differentiate modified-CS and weighted-ℓ_1 for the noise-free case. It has been observed in other works [47], [64] though, that weighted-ℓ_1 has a smaller recovery error than modified-CS when e is larger and the measurements are noisy.

In the second row, we show a 2D phase transition plot that varies s and n and displays a grey-scale intensity proportional to the Monte Carlo estimate of the exact recovery probability. This is done to compare BP, modified-CS and weighted-ℓ_1 in detail. Its conclusions are similar to the ones above. Notice that the white area (the region where an algorithm works with probability nearly one) is smallest for BP.

B. Recursive recovery of a simulated sparse signal sequence from noisy random-Gaussian measurements

We generated a sparse signal sequence, x_t, that satisfied the assumptions of Model 6.3 with $m = 256$, $s = 0.1m = 25$,



(a) $e = 0$ (b) $e = u$ (c) $e = 4u$

(d) BP (e) modified-CS, $e = 4u$ (f) weighted-ℓ_1, $e = 4u$

Fig. 3. Phase transition plots. In all figures we used signal length $m = 200$. In the top row figures, we plot the Monte Carlo estimate of the probability of exact recovery against n for $s = 0.1m$, $u = 0.1s$ and three values for e. Since the plots are 1D plots, we can compare all figures in a single figure. We compare BP, LS-CS, modified-CS, weighted-ℓ_1 with $\tau = e/s$ and BP-residual with two types of prior signal value knowledge - good and bad. In the "images" shown in the second row, we display the Monte Carlo estimate of the probability of exact recovery for various values of n (varied along x-axis) and s (varied along y-axis). The grey scale intensity for a given $(n/m), (s/m)$ point is proportional to the computed probability of exact recovery for that n, s.

$s_a = 1$, $b = 4$, $d_{\min} = 2$, $a_{\min} = 2$, $r_{\min} = 1$. The specific generative model to generate the x_t's was very similar to the one specified in [72, Appendix I] with one change: the magnitude of *all* entries in the support (not just the newly added ones) changed over time. We briefly summarize this here. The support set $\mathcal{N}_t$ consisted of three subsets. The first was the "large set", $\{i : |(x_t)_i| > a_{\min} + d_{\min} r_{\min}\}$. The magnitude of an element of this set increased by $r_{j,t}$ (for element j at time t) until it exceeded $a_{\min} + 6 d_{\min} r_{\min}$. The second set was the "decreasing set", $\{i : 0 < |(x_t)_i| < |(x_{t-1})_i|$ and $|(x_t)_i| < a_{\min} + d_{\min} r_{\min}\}$. At each time, there was a 50% probability that some entry from the large set entered the decreasing set (the entry itself was chosen uniformly at random). All entries of the decreasing set decreased to zero within $b = 4$ time units. The third set was the "increasing set", $\{i : |(x_t)_i| > |(x_{t-1})_i|$ and $|(x_t)_i| < a + d_{\min} r_{\min}\}$. At time t, if $|\mathcal{N}_{t-1}| < s$, then an element out of $\mathcal{N}_{t-1}^c$ was selected uniformly at random to enter the increasing set. Element j entered the increasing set at initial magnitude $a_{j,t}$ at time t. The magnitude of entries in this set increased for at least $d_{\min}$ time units with rate $r_{j,t}$ (for element j at time t). For all j, t, $r_{j,t}$ was i.i.d. uniformly distributed in the interval $[r_{\min}, 2r_{\min}]$ and the initial magnitude $a_{j,t}$ was i.i.d. uniformly distributed in the interval $[a_{\min}, 2a_{\min}]$. With the above model, $s_{d,t}$ was zero roughly half the time and so was $s_{a,t}$ except for the initial few time instants. For example, for a sample realization, $s_{a,t}$ was equal to 1 for 43 out of 100 time instants while being zero for the other 57. From the x_t's we generated measurements, $y_t = Ax_t + w_t$ where w_t was Gaussian noise with zero mean and variance $\sigma^2_{obs} = 0.0004$ and A was a random Gaussian matrix of size $n_1 \times m$ for $t = 1, 2$ and of size $n_3 \times m$ for $t \geq 3$. We used $n_1 = n_2 = 180$, and $n_t = n_3 = 0.234m = 60$ for $t \geq 3$. More measurements were used for the first two time instants because we used simple BPDN to estimate x_1 and x_2 for these time instants. These were used for parameter estimation as explained in Section VI-H.

We compare BPDN, BPDN-residual, PM-CS-KF (Algorithm 8) [88], ZA-LMS [94], [95], modified-BPDN (Algorithm 1), weighted-ℓ_1 (Algorithm 2), streaming modified weighted-ℓ_1 (streaming mod-wl1) [79], reg-mod-BPDN (Algorithm 5) KF-ModCS (Algorithm 6), DCS-AMP [80], [81] (algorithm in Table II), CS-MUSIC [76] and Temporal SBL [78]. BPDN solved (4) with $y = y_t$, $A = A_t$ at time t and $\gamma = \max\{10^{-2}\|A'[y_1 \ y_2]\|_\infty, \sigma_{\text{obs}}\sqrt{\log m}\}$ [79]. BPDN-residual solved (4) with $y = y_t - A_t \hat{x}_{t-1}$, $A = A_t$ at time t. Among these, BPDN uses no prior knowledge; BPDN-residual and PM-CS-KF are tracking-based methods that only use slow signal value change and sparsity; ZA-LMS is an adaptive-filtering-based solution; modified-BPDN and weighted-ℓ_1 use only slow support change; reg-mod-BPDN, KF-ModCS, DCS-AMP use both slow support and slow signal value change; and streaming mod-wl1 enforces a "soft" version of slow support change by also using the previous signal values' magnitudes. CS-MUSIC [76] and temporal SBL [78] are two batch algorithms that solve the MMV problem (assumes the support of x_t does not change with time). Since A is fixed,

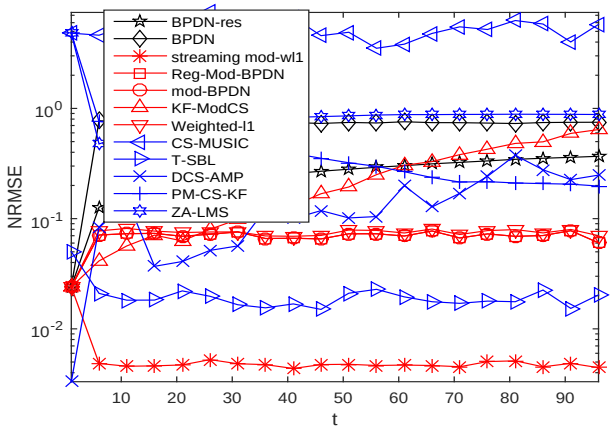


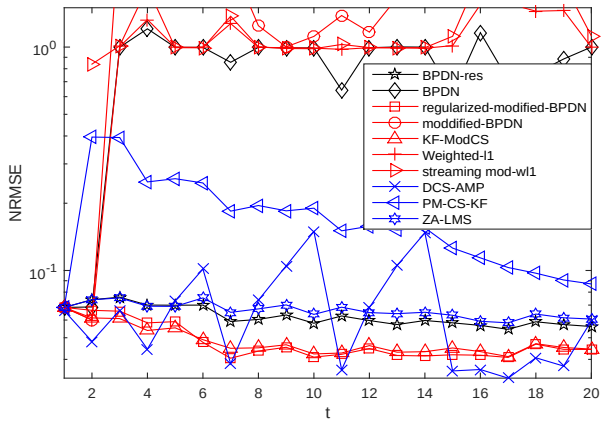
Fig. 4. NMSE plot for a simulated sparse signal sequence generated as explained in Section VIII-B, solved by ℓ_1-Homotopy [79].

we are able to apply these as well. Temporal SBL [78] additionally also uses temporal correlation among the nonzero entries while solving the MMV problem. The MMV problem is discussed in detail in Sec. IX-A2 (part of related work and future directions). We used the authors' code for DCS-AMP, PM-CS-KF, Temporal-SBL and CS-MUSIC. For the others, we wrote our own MATLAB code and used the ℓ_1-homotopy solver of Asif and Romberg [79] to solve the convex programs. With permission from the authors and with links to the authors' own webpages, the code for all algorithms used to generate the figures in the current article is posted at http://www.ece.iastate.edu/~namrata/RecReconReview.html.

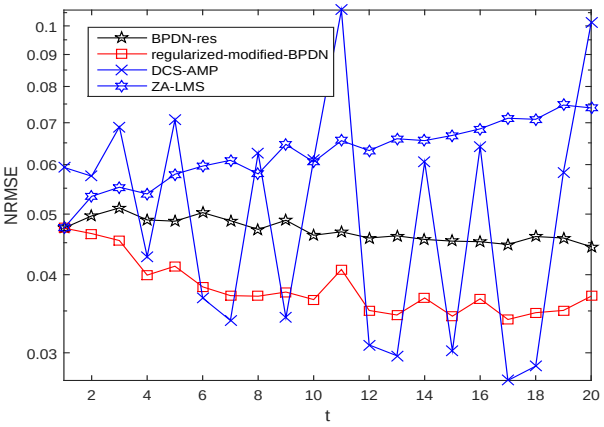
We plot the normalized root mean squared error (NRMSE), $\sqrt{\mathbb{E}[\|x_t - \hat{x}_t\|_2^2]}/\sqrt{\mathbb{E}[\|x_t\|_2^2]}$, in Fig. 4. Here $\mathbb{E}[.]$ denotes the Monte Carlo estimate of the expectation. We averaged over 100 realizations. The average time taken by each algorithm is shown in Table IV. As can be seen from Fig. 4, streaming mod-wl1 has the smallest error followed by temporal SBL and then reg-mod-BPDN, mod-BPDN and weighted-ℓ_1. The errors of all these are also stable (do not increase with time) and are below 10% at all times. Since enough measurements are used ($n = 60$ for $s_t \leq s = 25$), in this figure, one cannot see the advantage of reg-mod-BPDN over mod-BPDN. In terms of speed, ZA-LMS (an adaptive filtering technique) is the fastest, followed by DCS-AMP but both have unstable or large errors. Streaming mod-wl1, mod-BPDN, reg-mod-BPDN and weighted-ℓ_1 take similar amounts of time and all are 5-6 times slower than DCS-AMP and more than 10 times slower than ZA-LMS. However, this difference in speed disappears for large sized problems (see Table V, bottom).

C. Recursive recovery of a real vocal tract dynamic MRI sequence (approximately sparse) from simulated partial Fourier measurements

We show two experiments on the larynx vocal tract dynamic MRI sequence (shown in Fig. 1). In the first experiment we select a 32x32 block of it that contains most of the significant motion. In the second one, we use the entire 256x256 sized



(a) 32x32 piece sequence



(b) full image (256x256) sequence

Fig. 5. NRMSE plot for recovering the larynx image sequence from simulated partial Fourier measurements corrupted by Gaussian noise. We used $n_1 = n_2 = 0.18m$ and $n_t = 0.06m$ for $t > 2$ (a): recovery error plot for a 32x32 sub-image sequence, used the ℓ_1-homotopy solver. (b): recovery error plot for the full-sized image sequence, used the Yall1 solver.

image sequences. In the first experiment, the indices of the selected block were $(60:91, 60:91)$. Thus $m = 1024$. This was done to allow us to compare all algorithms including those that cannot work for large-scale problems. Let z_t denote the image at time t arranged as vector. We simulated MRI measurements as $y_t = H_t z_t + w_t$ where $H_t = I_{\mathcal{O}_t}{}' F_m$ where F_m is an m-point DFT matrix. The observed set $\mathcal{O}_t$ consisted of n_t rows selected using a modification of the low-frequency random undersampling scheme of Lustig et al. [111] and w_t was zero mean i.i.d. Gaussian noise with variance $\sigma_{obs}^2 = 10$. We used $n_1 = n_2 = 0.18m = 184$ and $n_t = 0.06m = 62$ for $t > 2$. To get 6% measurements, we generated three mask matrices with 50%, 40% and 30% measurements each using the low-frequency random undersampling scheme, multiplied them, and selected 62 rows out of the resulting matrix uniformly at random. More measurements were used for the first two time instants because we used simple BPDN to estimate x_1 and x_2 for these time instants. These were used for parameter estimation as explained in Section VI-H. For all algorithms,

reg-mod-BPDN	mod-BPDN	BPDN-res	BPDN	KF-ModCS	ZA-LMS	DCS-AMP	weighted-ell1	PM-CS-KF	str-mod-wl1	T-SBL
1.2340	1.0788	2.0748	4.4624	1.9184	0.0844	0.1816	0.9018	1.6463	0.9933	28.8252

TABLE IV
AVERAGED TIME TAKEN (IN SECONDS) BY DIFFERENT ALGORITHMS TO RECOVER THE SIMULATED SEQUENCE, AVERAGED OVER 100 SIMULATIONS, SOLVED BY ℓ_1-HOMOTOPY.

the sparsity basis that we used was a two-level Daubechies-4 wavelet. Thus Φ was the inverse wavelet transform corresponding to this wavelet written as a matrix. We compare all algorithms from the previous subsection except CS-MUSIC and T-SBL which require the matrix $A_t = H_t\Phi$ to be constant with time. The NRMSE is plotted in Fig. 5(a) and the time comparisons are shown in Table V, top.

In the second experiment, we used the full 256x256 larynx image sequence (so $m = 65536$) and generated simulated MRI measurements as above. In actual implementation, this was done by computing the 2D FFT of the image at time t followed by retaining coefficients with indices in the set $\mathcal{O}_t$. We again used $n_1 = n_2 = 0.18m = 11797$ and $n_t = 0.06m = 3933$ for $t > 2$ and $\sigma^2_{obs} = 10$. We select the best algorithms from the ones compared in the first experiment and compare their NRMSE in Fig. 5(b). By "best" we mean algorithms that had small error in the previous experiment and that can be implemented for the large-scale problem. We compare reg-mod-BPDN, ZA-LMS, DCS-AMP and BPDN-residual. For reg-mod-BPDN, we used γ, λ from the previous experiment since these cannot be computed for this large problem. For solving the convex programs of reg-mod-BPDN and BPDN-residual, we used the YALL-1 solver [112] which allows the user to work with partial Fourier measurements and with a DWT sparsity basis, without having to ever explicitly store the measurement matrix H_t or the sparsity basis matrix Φ (both are too large to fit in memory for this problem). So, for example, it computes $H_t z$ by computing the FFT of z followed by only keeping the entries with indices in $\mathcal{O}_t$. Similarly, for a vector y, it computes $H_t' y$ by computing the inverse FFT of y and retaining the entries with indices in $\mathcal{O}_t$ to get the row vector $(H_t' y)$. The time comparison for this experiment is shown in Table V, bottom.

As can be seen from Fig. 5 and Table V, reg-mod-BPDN has the smallest error although it is not the fastest. However it is only a little slower than DCS-AMP for the large scale problem (see Table V, bottom). This is, in part, because the YALL-1 solver is being used for it and that is much faster. DCS-AMP error is almost as small for many time instants but not all. Its performance seems to be very dependent on the specific A_t used. Another thing to notice is that algorithms such as ZA-LMS and BPDN-residual also do not have error that is too large since this example consists of compressible signal sequences.

IX. RELATED WORK AND FUTURE DIRECTIONS

We split the discussion in this section into four parts - more general signal models, more general measurement models, open questions for the algorithms described here, and how to do sequential detection and tracking using compressive measurements.

A. Signal Models

1) Structured Sparsity: There has been a lot of recent work on structured sparsity for a single signal. Dynamic extensions of these ideas should prove to be useful in applications where the structure is present and changes slowly. Two common examples of structured sparsity are block sparsity [113], [114] and tree structured sparsity (for wavelet coefficients) [115]. A length m vector is block sparse if it can be partitioned into length k blocks such that a lot of the blocks are entirely zero. One way to recover block sparse signals is by solving the ℓ_2-ℓ_1 minimization problem [113], [114]. Block sparsity is valid for many applications, e.g., for the foreground image sequence of a video consisting of one or a few moving objects, or for the activation regions in brain fMRI. In both of these cases, it is also true that the blocks do not change arbitrarily and hence the block support from the previous time instant should be a good estimate of the current block support. In this case one, can again use a mModified-CS type idea applied to the blocks. This was developed by Stojnic [116]. It solves

$$\min_b \sum_{j=1, j\notin\mathcal{T}}^{m/k} \sqrt{\sum_{i=1}^{k-1} b^2_{jk+i}} \quad \text{s.t.} \quad y = Ab$$

where $\mathcal{T}$ is the set of known nonzero blocks. Similarly, it should be possible to develop dynamic extensions of the tree-structured IHT algorithm of Cevher et al. [115] or of the approximate model-based IHT algorithm of Hegde et al. [117]. Another related work [118] assumes that the earth-mover's distance between the support sets of consecutive signals is small and uses this to design a recursive dynamic CS algorithm.

2) MMV and dynamic MMV: In the MMV problem [11]–[15], [76], [77], the goal is to recover a set of sparse signals with a common support but different nonzero signal values from a set of their measurements (all obtained using the same measurement matrix). This problem can be interpreted as that of block-sparse signal recovery and hence one commonly used solution is the ℓ_2-ℓ_1 program. Another more recent set of solutions is inspired by the MUSIC algorithm and are called

reg-mod-BPDN	mod-BPDN	BPDN-res	BPDN	KF-ModCS	ZA-LMS
16.4040	6.5424	2.0383	2.3205	10.8591	0.0973
DCS-AMP	Weighted ℓ_1	PM-CS-KF			
0.1515	2.4815	9.9332			

reg-mod-BPDN	BPDN-res	ZA-LMS	DCS-AMP
6.6700	17.4458	310.2790	4.9160

TABLE V
AVERAGED TIME TAKEN (IN SECONDS) BY DIFFERENT ALGORITHMS TO RECOVER A 32X32 PIECE OF THE LARYNX SEQUENCE (TOP) AND TO RECOVER THE FULL 256X256 LARYNX SEQUENCE (BOTTOM), AVERAGED OVER 100 SIMULATIONS.

CS-MUSIC [76] or iterative-MUSIC [77]. For signals with time-varying support (the case studied in this article), one can still use the MMV approaches as long as the size of their joint support (union of their supports), $\mathcal{N} := \cup_t \mathcal{N}_t$, is small enough. This may not be true for the entire sequence, but will usually be true for short durations. One could design a modified-MMV algorithm that utilizes the joint support of the previous duration to solve the current MMV problem better. A related idea was explored in a recent work [119].

Another related work is that of Zhang and Rao [78]. In it, the authors develop what they call the temporal SBL (T-SBL) algorithm. This is a batch (but fast) algorithm that solves the MMV problem with temporal correlations. It assumes that the support of x_t *does not* change over time and the signal values are correlated over time. The various indices of x_t are assumed to be independent. For a given index, i, it is assumed that $[(x_1)_i, (x_2)_i, \dots (x_{t_{\max}})_i]' \sim \mathcal{N}(0, \gamma_i B)$. All the x_t's are strung together to get a long $t_{\max} m$ length vector x that is Gaussian with block diagonal covariance matrix $\text{diag}(\gamma_1 B, \gamma_2 B, \dots \gamma_m B)$. T-SBL develops the SBL approach to estimate the hyper-parameters $\{\sigma^2, \gamma_1, \gamma_2, \dots \gamma_m, B\}$ and then compute an MAP estimate of the sparse vector sequence.

Another related work [120] develops and studies a causal but batch algorithm for CS for time-varying signals.

3) Sparse Transform Learning and Dictionary Learning: In certain applications involving natural images, the wavelet transform provides a good enough sparsifying basis. However, for many other applications, while the wavelet transform is one possible sparsifying basis, it can be significantly improved. There has been a large amount of recent work both on dictionary learning, e.g., [45], and more recently on sparsifying transform learning from a given dataset of sparse signals [46]. The advantage of the latter work is that it is much faster than existing dictionary learning approaches. For dynamic sparse signals, an open question of interest is how to learn a sparsifying transform that is optimized for signal sequences with slow support change?

B. Measurement Models

1) Recursive Dynamic CS in Large but Structured Noise: The work discussed in this article solves the sparse recovery problem either in the noise-free case or in the small noise case. Only in this case, one can show that the reconstruction error is small compared to the signal energy. In fact, this is true for almost all work on sparse recovery; one can get reasonable error bounds only for the small noise case.

However, in some applications, the noise energy can be much larger than that of the sparse signal. If the noise is large but has no structure then nothing can be done. But if it does have structure, that can be exploited. This was done for outliers (modeled as sparse vectors) in the work of Wright and Ma [121]. Their work, and then many later works, showed exact sparse recovery from large but sparse noise (outliers) as long as the sparsity bases for the signal and the noise/outlier are "different" or "incoherent" enough. In fact, as noted by an anonymous reviewer, more generally, the earlier works on sparsity in unions of bases, e.g., [122]–[124], can also be interpreted in this fashion. Recent work on robust PCA by Candes et al. [51] and Chandrasekharan et al. [125] posed robust PCA as a problem of separating a low-rank matrix and a sparse matrix from their sum and proposed a convex optimization solution to it. In more recent work [52]–[55], [126], the recursive or online robust PCA problem was solved. This can be interpreted as a problem of recursive dynamic CS in large but structured noise (noise that is dense and lies in a fixed or "slowly changing" low-dimensional subspace of the full space). An open question is how can the work described in this article be used in this context and what we say about performance guarantees for the resulting algorithm? For example, in [54], [127], the authors have attempted to use weighted-ℓ_1 to replace BP-noisy with encouraging results.

2) Recursive Recovery from Nonlinear Measurements and Dynamic Phase Retrieval: The work described in this article focuses on sparse recovery from linear measurements. However, in many applications such as computer vision, the measurement (e.g., image) is a nonlinear function of the sparse signal of interest (e.g., object's boundary which is often modeled as being Fourier sparse). Some recent work that has studied the static version of this "nonlinear CS" problem includes [128]–[132]. These approaches use an iterative linearization procedure along with adapting standard iterative sparse recovery techniques such as IHT. An extension of these techniques for the dynamic case can potentially be developed using an extended Kalman filter and the modified-IHT (IHT-PKS) idea from [68]. There has been other work on solving the dynamic CS problem from nonlinear measurements by using particle filtering based approaches [83], [84], [133], [134].

An important special case of the nonlinear CS problem is sparse phase retrieval, i.e., recover a sparse x from $y := |Ax|$. Here $|.|$ takes the element-wise magnitude of the vector (Ax). The special case of this problem where A is the Fourier matrix occurs in applications such as astronomical imaging, optical imaging and X-ray crystallography where one can only measure the magnitude of the Fourier coefficients of the unknown quantity. This problem has been studied in a series of recent works [135]–[138], [138]–[141]. An open question is how to design and analyze an approach for phase retrieval for a time sequence of sparse signals and how much will the use of past information help? For example, the work of Jaganathan et al. [138] provides a provably correct approach for sparse Fourier phase retrieval. It involves first using a combinatorial algorithm to estimate the signal's support, followed by using a lifting technique to get a convex optimization program for positive semi-definite matrices. As explained in [138], the lifting based convex program cannot be used directly because of the location ambiguity introduced by the Fourier transform. Consider the dynamic sparse phase retrieval problem. An open question is whether the support and the signal value estimates from the previous time instant can help regularize this problem enough to ensure a unique solution when directly solving the resulting lifting-based convex program? If the combinatorial support recovery algorithm can be eliminated, it would make the solution approach a lot faster.

C. Algorithms

The work done on this topic so far consists of good algorithms that improve significantly over simple-CS solutions, and some of them come with provable guarantees. However, it is not clear if any of these are "optimal" in any sense. An open question is, can we develop an "optimal" algorithm or can we show that an algorithm is close enough to an "optimal" one? For sparse recovery techniques, it is not even clear what a tractable measure of "optimality" is?

The original KF-CS algorithm [32] was developed with this question in mind; however so far there has not been any reasonable performance bound for it or for its improved version, KF-ModCS. An open question is, can we show that KF-ModCS comes within a bounded distance of the genie-aided causal MMSE solution for this problem (the causal MMSE solution assuming that the support sets at each time are known)?

In fact, none of the approaches that utilize both slow support and signal value change have stability results so far. This is an important question for future work. In particular, it would be interesting to analyze streaming mod-wl1 (Algorithm 4) [79] since it has excellent performance in simulations.

There is other very recent work on necessary and sufficient conditions for weighted-ℓ_1 [142].

D. Compressive Sequential Signal Detection and Tracking

In many signal processing applications, the final goal is to use the recovered signal for detection, classification, estimation or to filter out a certain component of the signal (e.g. a band of frequencies or some other subspace). The question is can we do this directly with compressive measurements without having to first recover the sparse signal? This has been studied in some recent works such as the work of Davenport et al. [143]. For a time sequence of sparse signals, a question of interest is how to do the same thing for compressive sequential detection/classification or sequential estimation (tracking)? The question to answer would be how to use the previously reconstructed/detected signal to improve compressive detection at the current time and how to analyze the performance of the resulting approach?

X. Conclusions

This article reviewed the literature on recursive recovery of sparse signal sequences or what can be called "recursive dynamic CS". Most of the literature on this topic exploits one or both of the practically valid assumptions of slow support change and slow signal value change. While slow signal value change is commonly used in a lot of previous tracking and adaptive filtering literature, the slow support change is a new assumption introduced for solving this problem. As shown in Fig. 2, this is indeed valid in applications such as dynamic MRI. We summarized both theoretical and experimental results that demonstrate the advantage of using these assumptions. In the section above, we also discussed related problems and open questions for future work.

A key limitation of almost all the work reviewed here is that the algorithms assume more measurements are available at the first time instant. This is needed in order to get an accurate initial signal recovery using simple-CS solutions. In applications such as dynamic MRI or functional MRI, it is possible to use more measurements at the initial time (the scanner can be configured to allow this). Other solutions to this issue have been described in Section VI-A.

Appendix A
Computable error bound for reg-mod-BPDN, mod-BPDN, BPDN

We define here the terms used in Theorem 5.2. Let $I_{\mathcal{T},\mathcal{T}}$ denote the identity matrix on the row, column indices $\mathcal{T},\mathcal{T}$ and let $\mathbf{0}_{\mathcal{T},S}$ be a zero matrix on the row, column indices $\mathcal{T},S$. Define

$$\begin{aligned}
\text{maxcor}(\tilde{\Delta}_u) &:= \max_{i\notin(T\cup\tilde{\Delta}_u)^c} \|A_i{}' A_{\mathcal{T}\cup\tilde{\Delta}_u}\|_2,\\
Q_{\mathcal{T},\lambda}(S) &:= A_{\mathcal{T}\cup S}{}' A_{\mathcal{T}\cup S} + \lambda \begin{bmatrix} I_{\mathcal{T},\mathcal{T}} & \mathbf{0}_{\mathcal{T},S} \\ \mathbf{0}_{S,\mathcal{T}} & \mathbf{0}_{S,S}\end{bmatrix}\\
ERC_{\mathcal{T},\lambda}(S) &:= 1 - \max_{\omega\notin T\cup S} \|P_{\mathcal{T},\lambda}(S) A_S{}' M_{\mathcal{T},\lambda} A_\omega\|_1,\\
P_{\mathcal{T},\lambda}(S) &:= (A_S{}' M_{\mathcal{T},\lambda} A_S)^{-1}\\
M_{\mathcal{T},\lambda} &:= I - A_{\mathcal{T}}(A_{\mathcal{T}}{}' A_{\mathcal{T}} + \lambda I_{\mathcal{T},\mathcal{T}})^{-1} A_{\mathcal{T}}{}'
\end{aligned}$$

and

$$\begin{aligned}
f_1(\tilde{\Delta}_u) &:= \sqrt{\|(A_{\mathcal{T}}{}' A_{\mathcal{T}} + \lambda I_{\mathcal{T}})^{-1} A_{\mathcal{T}}{}' A_{\tilde{\Delta}_u} P_{\mathcal{T},\lambda}(\tilde{\Delta}_u)\|_2^2 + \|P_{\mathcal{T},\lambda}(\tilde{\Delta}_u)\|_2^2},\\
f_2(\tilde{\Delta}_u) &:= \|Q_{\mathcal{T},\lambda}(\tilde{\Delta}_u)^{-1}\|_2\\
f_3(\tilde{\Delta}_u) &:= \|Q_{\mathcal{T},\lambda}(\tilde{\Delta}_u)^{-1} A_{\mathcal{T}\cup\tilde{\Delta}_u}{}'\|_2,\\
f_4(\tilde{\Delta}_u) &:= \sqrt{\|Q_{\mathcal{T},\lambda}(\tilde{\Delta}_u)^{-1} A_{\mathcal{T}\cup\tilde{\Delta}_u}{}' A_{\tilde{\Delta}_u\setminus\tilde{\Delta}_u}\|_2^2 + 1}.\\
g_1(\tilde{\Delta}_u) &:= \lambda f_2(\tilde{\Delta}_u)\left(\frac{\sqrt{|\tilde{\Delta}_u|} f_1(\tilde{\Delta}_u)\text{maxcor}(\tilde{\Delta}_u)}{ERC_{\mathcal{T},\lambda}(\tilde{\Delta}_u)} + 1\right),\\
g_2(\tilde{\Delta}_u) &:= \frac{\sqrt{|\tilde{\Delta}_u|} f_1(\tilde{\Delta}_u) f_3(\tilde{\Delta}_u)\text{maxcor}(\tilde{\Delta}_u)}{ERC_{\mathcal{T},\lambda}(\tilde{\Delta}_u)} + f_3(\tilde{\Delta}_u),\\
g_3(\tilde{\Delta}_u) &:= \frac{\sqrt{|\tilde{\Delta}_u|} f_1(\tilde{\Delta}_u) f_4(\tilde{\Delta}_u)\text{maxcor}(\tilde{\Delta}_u)}{ERC_{\mathcal{T},\lambda}(\tilde{\Delta}_u)} + f_4(\tilde{\Delta}_u),\\
g_4(\tilde{\Delta}_u) &:= \frac{\sqrt{|\tilde{\Delta}_u|}\|A_{(T\cup\tilde{\Delta}_u)^c}\|_\infty \|w\|_\infty f_1(\tilde{\Delta}_u)}{ERC_{\mathcal{T},\lambda}(\tilde{\Delta}_u)}
\end{aligned}$$

For a set $\tilde{\Delta}_u \subseteq \Delta_u$, define

$$\begin{aligned}
\gamma^*_{\mathcal{T},\lambda}(\tilde{\Delta}_u) &:= \frac{\text{maxcor}(\tilde{\Delta}_u)}{\text{ERC}_{\mathcal{T},\lambda}(\tilde{\Delta}_u)}\Big[\lambda f_2(\tilde{\Delta}_u)\|x_{\mathcal{T}} - \hat{\mu}_{\mathcal{T}}\|_2 +\\
&\qquad f_3(\tilde{\Delta}_u)\|w\|_2 + f_4(\tilde{\Delta}_u)\|x_{\Delta_u\setminus\tilde{\Delta}_u}\|_2\Big]\\
&\qquad + \frac{\|w\|_\infty}{\text{ERC}_{\mathcal{T},\lambda}(\tilde{\Delta}_u)}, \qquad (35)\\
g_\lambda(\tilde{\Delta}_u) &:= g_1(\tilde{\Delta}_u)\|x_{\mathcal{T}} - \hat{\mu}_{\mathcal{T}}\|_2 + g_2(\tilde{\Delta}_u)\|w\|_2\\
&\qquad + g_3(\tilde{\Delta}_u)\|x_{\Delta_u\setminus\tilde{\Delta}_u}\|_2 + g_4(\tilde{\Delta}_u) \qquad (36)
\end{aligned}$$

For an integer k, define

$$\tilde{\Delta}^*_u(k) := \arg \min_{\tilde{\Delta}_u\subseteq\Delta_u, |\tilde{\Delta}_u|=k} \|x_{\Delta_u\setminus\tilde{\Delta}_u}\|_2 \qquad (37)$$

This is a subset of Δ_u of size k that contains the k largest magnitude entries of x.

Let $k_{\min}$ be the integer k that results in the smallest error bound $g_\lambda(\tilde{\Delta}_u^*(k))$ out of all integers k for which $\text{ERC}_{\mathcal{T},\lambda}(\tilde{\Delta}_u^*(k)) > 0$ and $Q_{\mathcal{T},\lambda}(\tilde{\Delta}_u^*(k))$ is invertible. Thus,

$$k_{\min} := \arg\min_k B_k \text{ where}$$
$$B_k := \begin{cases} g(\tilde{\Delta}_u^*(k)) & \text{if } \text{ERC}_{\mathcal{T},\lambda}(\tilde{\Delta}_u^*(k)) > 0 \text{ and} \\ & Q_{\mathcal{T},\lambda}(\tilde{\Delta}_u^*(k)) \text{ is invertible} \\ \infty & \text{otherwise} \end{cases}$$

Notice that $\arg\min_{\tilde{\Delta}_u \subseteq \Delta_u, |\tilde{\Delta}_u| = k} \|x_{\Delta_u \setminus \tilde{\Delta}_u}\|_2$ in (37) is computable in polynomial time by just sorting the entries of x_{Δ_u} in decreasing order of magnitude and retaining the indices of its largest k entries. Hence everything in the above result is computable in polynomial time if x, $\mathcal{T}$ and a bound on $\|w\|_\infty$ are available. Thus if training data is available, the above theorem can be used to compute good choices of γ and λ.

REFERENCES

[1] S. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Trans. Sig. Proc.*, vol. 41(12), pp. 3397 – 3415, Dec 1993.

[2] S. Chen, D. Donoho, and M. Saunders, "Atomic decomposition by basis pursuit," *SIAM Journal of Scientific Computing*, vol. 20, pp. 33–61, 1998.

[3] P. Feng and Y. Bresler, "Spectrum-blind minimum-rate sampling and reconstruction of multiband signals," in *IEEE Intl. Conf. Acoustics, Speech, Sig. Proc. (ICASSP)*, vol. 3, 1996, pp. 1688–1691.

[4] Y. Bresler and P. Feng, "Spectrum-blind minimum-rate sampling and reconstruction of 2-d multiband signals," in *Image Processing, 1996. Proceedings., International Conference on*, vol. 1. IEEE, 1996, pp. 701–704.

[5] I. F. Gorodnitsky and B. D. Rao, "Sparse signal reconstruction from limited data using focuss: A re-weighted norm minimization algorithm," *IEEE Trans. Sig. Proc.*, pp. 600 – 616, March 1997.

[6] I. Gorodnitsky and B. Rao, "Sparse signal reconstruction from limited data using focuss: A re-weighted minimum norm algorithm," *IEEE Trans. Sig. Proc.*, vol. 45, no. 3, pp. 600–616, 1997.

[7] D. Wipf and B. Rao, "Sparse bayesian learning for basis selection," *IEEE Trans. Sig. Proc.*, vol. 52, pp. 2153–2164, Aug 2004.

[8] D. Donoho, "Compressed sensing," *IEEE Trans. Info. Th.*, vol. 52(4), pp. 1289–1306, April 2006.

[9] E. Candes and T. Tao, "Decoding by linear programming," *IEEE Trans. Info. Th.*, vol. 51(12), pp. 4203 – 4215, Dec. 2005.

[10] E. Candes, J. Romberg, and T. Tao, "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information," *IEEE Trans. Info. Th.*, vol. 52(2), pp. 489–509, February 2006.

[11] D. P. Wipf and B. D. Rao, "An empirical bayesian strategy for solving the simultaneous sparse approximation problem," *IEEE Trans. Sig. Proc.*, vol. 55, no. 7, pp. 3704–3716, 2007.

[12] J. A. Tropp, "Algorithms for simultaneous sparse approximation. part ii: Convex relaxation," *Signal Processing*, vol. 86, no. 3, pp. 589–602, 2006.

[13] J. Chen and X. Huo, "Theoretical results on sparse representations of multiple-measurement vectors," *IEEE Trans. Sig. Proc.*, 2006.

[14] Y. C. Eldar, P. Kuppinger, and H. Bölcskei, "Compressed sensing of block-sparse signals: Uncertainty relations and efficient recovery," *arXiv preprint arXiv:0906.3173*, 2009.

[15] M. E. Davies and Y. C. Eldar, "Rank awareness in joint sparse recovery," *IEEE Trans. Info. Th.*, vol. 58, no. 2, pp. 1135–1146, 2012.

[16] M. Wakin, J. Laska, M. Duarte, D. Baron, S. Sarvotham, D. Takhar, K. Kelly, and R. Baraniuk, "Compressive imaging for video representation and coding," in *Proc. Picture Coding Symposium (PCS), Beijing, China*, April 2006.

[17] U. Gamper, P. Boesiger, and S. Kozerke, "Compressed sensing in dynamic MRI," *Magnetic Resonance in Medicine*, vol. 59(2), pp. 365–373, January 2008.

[18] H. Jung, K. H. Sung, K. S. Nayak, E. Y. Kim, and J. C. Ye, "k-t focuss: a general compressed sensing framework for high resolution dynamic MRI," *Magnetic Resonance in Medicine*, 2009.

[19] I. Carron, "Nuit blanche," in *http://nuit-blanche.blogspot.com/*.

[20] "Rice compressive sensing resources," in *http://www-dsp.rice.edu/cs*.

[21] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad, "Orthogonal matching pursuit: Recursive function approximat ion with applications to wavelet decomposition," in *Proc. Asilomar Conf. on Signals, Systems and Computers*, 1993, pp. 40–44.

[22] J. Tropp and A. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Trans. on Information Theory*, vol. 53(12), pp. 4655–4666, December 2007.

[23] W. Dai and O. Milenkovic, "Subspace pursuit for compressive sensing signal reconstruction," *IEEE Trans. Info. Th.*, vol. 55(5), pp. 2230 – 2249, May 2009.

[24] D. Needell and J. Tropp., "Cosamp: Iterative signal recovery from incomplete and inaccurate samples," *Appl. Comp. Harmonic Anal.*, vol. 26(3), pp. 301–321, May 2009.

[25] T. Blumensath and M. E. Davies, "Iterative hard thresholding for compressed sensing," *Applied and Computational Harmonic Analysis*, pp. 265–274, 2009.

[26] T. Blumensath and M. Davies, "Normalised iterative hard thresholding; guaranteed stability and performance," *IEEE Journal of Selected Topics in Signal Processing*, vol. 4, no. 2, pp. 298–309, 2010.

[27] M. E. Tipping, "Sparse bayesian learning and the relevance vector machine," *The journal of machine learning research*, vol. 1, pp. 211–244, 2001.

[28] N. Vaswani, "LS-CS-residual (LS-CS): compressive sensing on least squares residual," *IEEE Trans. Sig. Proc.*, vol. 58(8), pp. 4108–4120, August 2010.

[29] E. Candes, "The restricted isometry property and its implications for compressed sensing," *Compte Rendus de l'Academie des Sciences, Paris, Serie I*, pp. 589–592, 2008.

[30] I. Daubechies, R. DeVore, M. Fornasier, and S. Gunturk, "Iteratively re-weighted least squares minimization: Proof of faster than linear rate for sparse recovery," in *Proceedings of the 42nd IEEE Annual Conference on Information Sciences and Systems (CISS 2008)*. IEEE, 2008, pp. 26–29.

[31] A. Cohen, W. Dahmen, and R. DeVore, "Compressed sensing and best k-term approximation," *Journal of the American mathematical society*, vol. 22, no. 1, pp. 211–231, 2009.

[32] N. Vaswani, "Kalman filtered compressed sensing," in *IEEE Intl. Conf. Image Proc. (ICIP)*, 2008.

[33] N. Vaswani and W. Lu, "Modified-CS: Modifying compressive sensing for problems with partially known support," *IEEE Trans. Sig. Proc.*, vol. 58(9), pp. 4595–4607, September 2010.

[34] A. J. Martin, O. M. Weber, D. Saloner, R. Higashida, M. Wilson, M. Saeed, and C. Higgins, "Application of MR Technology to Endovascular Interventions in an XMR Suite," *Medica Mundi*, vol. 46, December 2002.

[35] C. Qiu, W. Lu, and N. Vaswani, "Real-time dynamic MR image reconstruction using Kalman filtered compressed sensing," in *IEEE Intl. Conf. Acoustics, Speech, Sig. Proc. (ICASSP)*. IEEE, 2009, pp. 393–396.

[36] I. C. Atkinson, D. L. J. F. Kamalabadi, and K. R. Thulborn, "Blind estimation for localized low contrast-to-noise ratio bold signals," *IEEE Journal of Selected Topics in Signal Processing*, vol. 8, pp. 350–364, 2008.

[37] W. Lu, T. Li, I. Atkinson, and N. Vaswani, "Modified-CS-residual for recursive reconstruction of highly undersampled functional MRI sequences," in *IEEE Intl. Conf. Image Proc. (ICIP)*. IEEE, 2011, pp. 2689–2692.

[38] H. Chen, M. S. Asif, A. C. Sankaranarayanan, and A. Veeraraghavan, "FPA-CS: Focal Plane Array-based Compressive Imaging in Short-wave Infrared," *arXiv preprint arXiv:1504.04085*, 2015.

[39] M. A. Herman, T. Weston, L. McMackin, Y. Li, J. Chen, and K. F. Kelly, "Recent results in single-pixel compressive imaging using selective measurement strategies," in *SPIE Sensing Technology+ Applications*. International Society for Optics and Photonics, 2015, pp. 94 840A–94 840A.

[40] A. C. Sankaranarayanan, L. Xu, C. Studer, Y. Li, K. Kelly, and R. G. Baraniuk, "Video compressive sensing for spatial multiplexing cameras using motion-flow models," *arXiv preprint arXiv:1503.02727*, 2015.

[41] A. Colaço, A. Kirmani, G. A. Howland, J. C. Howell, and V. K. Goyal, "Compressive depth map acquisition using a single photon-counting detector: Parametric signal processing meets sparsity," in *Computer*

Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on. IEEE, 2012, pp. 96–102.

[42] G. Huang, H. Jiang, K. Matthews, and P. Wilford, "Lensless imaging by compressive sensing," in *Image Processing (ICIP), 2013 20th IEEE International Conference on*. IEEE, 2013, pp. 2101–2105.

[43] V. Cevher, A. Sankaranarayanan, M. Duarte, D. Reddy, R. Baraniuk, and R. Chellappa, "Compressive sensing for background subtraction," in *Eur. Conf. on Comp. Vis. (ECCV)*. Springer, 2008, pp. 155–168.

[44] 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*, vol. 51, no. 1, pp. 34–81, 2009.

[45] M. Aharon, M. Elad, and A. Bruckstein, "The K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Trans. Sig. Proc.*, vol. 54, no. 11, pp. 4311–4322, 2006.

[46] S. Ravishankar and Y. Bresler, "Learning sparsifying transforms," *IEEE Trans. Sig. Proc.*, vol. 61, no. 5, pp. 1072–1086, 2013.

[47] M. Friedlander, H. Mansour, R. Saab, and O. Yilmaz, "Recovering compressively sampled signals using partial support information," *IEEE Trans. Info. Th.*, vol. 58, no. 2, pp. 1122–1134, 2012.

[48] D. Xu, N. Vaswani, Y. Huang, and J. U. Kang, "Modified compressive sensing optical coherence tomography with noise reduction," *Optics letters*, vol. 37, no. 20, pp. 4209–4211, 2012.

[49] L. Liu, Z. Han, Z. Wu, and L. Qian, "Collaborative compressive sensing based dynamic spectrum sensing and mobile primary user localization in cognitive radio networks," in *Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE*. IEEE, 2011, pp. 1–5.

[50] J. Oliver, R. Aravind, and K. Prabhu, "Sparse channel estimation in ofdm systems by threshold-based pruning," *Electronics Letters*, vol. 44, no. 13, pp. 830–832, 2008.

[51] E. J. Candès, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis?" *Journal of ACM*, vol. 58, no. 3, 2011.

[52] C. Qiu and N. Vaswani, "Real-time robust principal components' pursuit," in *Allerton Conference on Communication, Control, and Computing*. IEEE, 2010, pp. 591–598.

[53] C. Qiu, N. Vaswani, B. Lois, and L. Hogben, "Recursive robust pca or recursive sparse recovery in large but structured noise," *IEEE Trans. Info. Th.*, vol. 60, no. 8, pp. 5007–5039, August 2014.

[54] H. Guo, C. Qiu, and N. Vaswani, "An online algorithm for separating sparse and low-dimensional signal sequences from their sum," *IEEE Trans. Sig. Proc.*, vol. 62, no. 16, pp. 4284–4297, 2014.

[55] J. He, L. Balzano, and A. Szlam, "Incremental gradient on the grassmannian for online foreground and background separation in subsampled video," in *IEEE Conf. on Comp. Vis. Pat. Rec. (CVPR)*, 2012.

[56] N. Vaswani and W. Lu, "Modified-CS: Modifying compressive sensing for problems with partially known support," in *IEEE Intl. Symp. Info. Th. (ISIT)*, 2009.

[57] C. Qiu and N. Vaswani, "Support predicted modified-cs for recursive robust principal components' pursuit," in *IEEE Intl. Symp. Info. Th. (ISIT)*, 2011.

[58] N. Vaswani, "Analyzing least squares and kalman filtered compressed sensing," in *IEEE Intl. Conf. Acoustics, Speech, Sig. Proc. (ICASSP)*, 2009.

[59] S. Foucart and M. J. Lai, "Sparsest solutions of underdetermined linear systems via ell-q-minimization for $0 \leq q \leq 1$," *Applied and Computational Harmonic Analysis*, vol. 26, pp. 395–407, 2009.

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

[61] C. J. Miosso, R. von Borries, M. Argez, L. Valazquez, C. Quintero, and C. Potes, "Compressive sensing reconstruction with prior information by iteratively reweighted least-squares," *IEEE Trans. Sig. Proc.*, vol. 57 (6), pp. 2424–2431, June 2009.

[62] A. Khajehnejad, W. Xu, A. Avestimehr, and B. Hassibi, "Weighted ℓ_1 minimization for sparse recovery with prior information," in *IEEE Intl. Symp. Info. Th. (ISIT)*, 2009.

[63] ——, "Weighted ℓ_1 minimization for sparse recovery with prior information," *IEEE Trans. Sig. Proc.*, 2011.

[64] W. Lu and N. Vaswani, "Regularized modified bpdn for noisy sparse reconstruction with partial erroneous support and signal value knowledge," *IEEE Trans. Sig. Proc.*, vol. 60, no. 1, pp. 182–196, 2012.

[65] B. Bah and J. Tanner, "Bounds of restricted isometry constants in extreme asymptotics: formulae for gaussian matrices," *Linear Algebra and its Applications*, vol. 441, pp. 88–109, 2014.

[66] D. Donoho, "For most large underdetermined systems of linear equations, the minimal ell-1 norm solution is also the sparsest solution," *Comm. Pure and App. Math.*, vol. 59(6), pp. 797–829, June 2006.

[67] V. Stankovic, L. Stankovic, and S. Cheng, "Compressive image sampling with side information," in *ICIP*, 2009.

[68] R. Carrillo, L. Polania, and K. Barner, "Iterative algorithms for compressed sensing with patially known support," in *ICASSP*, 2010.

[69] A. S. Bandeira, K. Scheinberg, and L. N. Vicente, "On partial sparse recovery," *arXiv:1304.2809*.

[70] Y. Wang and W. Yin, "Sparse signal reconstruction via iterative support detection," *SIAM Journal on Imaging Sciences*, pp. 462–491, 2010.

[71] E. Candes, M. Wakin, and S. Boyd, "Enhancing sparsity by reweighted l(1) minimization," *Journal of Fourier Analysis and Applications*, vol. 14 (5-6), pp. 877–905, 2008.

[72] J. Zhan and N. Vaswani, "Time invariant error bounds for modified-CS based sparse signal sequence recovery," *IEEE Trans. Info. Th.*, vol. 61, no. 3, pp. 1389–1409, 2015.

[73] W. Lu and N. Vaswani, "Exact reconstruction conditions for regularized modified basis pursuit," *IEEE Trans. Sig. Proc.*, vol. 60, no. 5, pp. 2634–2640, 2012.

[74] J. A. Tropp, "Just relax: Convex programming methods for identifying sparse signals," *IEEE Trans. Info. Th.*, pp. 1030–1051, March 2006.

[75] W. Lu and N. Vaswani, "Modified Compressive Sensing for Real-time Dynamic MR Imaging," in *IEEE Intl. Conf. Image Proc. (ICIP)*, 2009.

[76] J. M. Kim, O. K. Lee, and J. C. Ye, "Compressive music: revisiting the link between compressive sensing and array signal processing," *IEEE Trans. Info. Th.*, vol. 58, no. 1, pp. 278–301, 2012.

[77] K. Lee, Y. Bresler, and M. Junge, "Subspace methods for joint sparse recovery," *IEEE Trans. Info. Th.*, vol. 58, no. 6, pp. 3613–3641, 2012.

[78] Z. Zhang and B. D. Rao, "Sparse signal recovery with temporally correlated source vectors using sparse bayesian learning," *IEEE J. Sel. Topics Sig. Proc., Special Issue on Adaptive Sparse Representation of Data and Applications in Signal and Image Processing*, vol. 5, no. 5, pp. 912–926, Sept 2011.

[79] M. Asif and J. Romberg, "Sparse recovery of streaming signals using ℓ_1 homotopy," *IEEE Trans. Sig. Proc.*, vol. 62, no. 16, pp. 4209–4223, 2014.

[80] J. Ziniel, L. C. Potter, and P. Schniter, "Tracking and smoothing of time-varying sparse signals via approximate belief propagation," in *Asilomar Conf. on Sig. Sys. Comp.*, 2010.

[81] J. Ziniel and P. Schniter, "Dynamic compressive sensing of time-varying signals via approximate message passing," *IEEE Trans. Sig. Proc.*, vol. 61, no. 21, pp. 5270–5284, 2013.

[82] D. L. Donoho, A. Maleko, and A. Montanari, "Message-passing algorithms for compressed sensing," *Proceedings of National Academy of Sciences (PNAS)*, 2009.

[83] R. Sarkar, S. Das, and N. Vaswani, "Tracking sparse signal sequences from nonlinear/non-gaussian measurements and applications in illumination-motion tracking," in *IEEE Intl. Conf. Acoustics, Speech, Sig. Proc. (ICASSP)*, 2013.

[84] S. Das and N. Vaswani, "Particle filtered modified compressive sensing (pafimocs) for tracking signal sequences," in *Asilomar*, 2010.

[85] J. P. Vila and P. Schniter, "Expectation-maximization gaussian-mixture approximate message passing," *IEEE Trans. Sig. Proc.*, vol. 61, no. 19, pp. 4658–4672, 2013.

[86] N. Vaswani, "Stability (over time) of Modified-CS for Recursive Causal Sparse Reconstruction," in *Allerton Conf. Communication, Control, and Computing*, 2010.

[87] J. Zhan and N. Vaswani, "Time invariant error bounds for modified-CS based sparse signal sequence recovery," in *IEEE Intl. Symp. Info. Th. (ISIT)*, 2013.

[88] A. Carmi, P. Gurfil, and D. Kanevsky, "Methods for sparse signal recovery using kalman filtering with embedded pseudo-measurement norms and quasi-norms," *IEEE Trans. Sig. Proc.*, pp. 2405–2409, April 2010.

[89] S. J. Julier and J. J. LaViola, "On kalman filtering with nonlinear equality constraints," *IEEE Trans. Sig. Proc.*, 2007.

[90] D. Malioutov, S. Sanghavi, and A. S. Willsky, "Compressed sensing with sequential observations," in *IEEE Intl. Conf. Acoustics, Speech, Sig. Proc. (ICASSP)*, 2008.

[91] P. J. Garrigues and L. E. Ghaoui, "An homotopy algorithm for the lasso with online observations," in *Adv. Neural Info. Proc. Sys. (NIPS)*, 2008.

[92] M. S. Asif and J. Romberg, "Dynamic updating for sparse time varying signals," in *Conf. Info. Sciences and Systems*, 2009.

[93] D. Angelosante and G. Giannakis, "Rls-weighted lasso for adaptive estimation of sparse signals," in *IEEE Intl. Conf. Acoustics, Speech, Sig. Proc. (ICASSP)*, 2009.

[94] Y. Chen, Y. Gu, and A. O. Hero III, "Sparse lms for system identification," in *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*. IEEE, 2009, pp. 3125–3128.

[95] J. Jin, Y. Gu, and S. Mei, "A stochastic gradient approach on compressive sensing signal reconstruction based on adaptive filtering framework," *Selected Topics in Signal Processing, IEEE Journal of*, vol. 4, no. 2, pp. 409–420, 2010.

[96] G. Su, J. Jin, Y. Gu, and J. Wang, "Performance analysis of l0-norm constraint least mean square algorithm," *IEEE Trans. Sig. Proc.*, vol. 60, no. 5, pp. 2223–2235, 2012.

[97] B. Babadi, N. Kalouptsidis, and V. Tarokh, "Sparls: The sparse rls algorithm," *IEEE Trans. Sig. Proc.*, vol. 58, no. 8, pp. 4013–4025, 2010.

[98] M. A. Figueiredo and R. D. Nowak, "An em algorithm for wavelet-based image restoration," *Image Processing, IEEE Transactions on*, vol. 12, no. 8, pp. 906–916, 2003.

[99] D. Angelosante, J. A. Bazerque, and G. B. Giannakis, "Online adaptive estimation of sparse signals: Where rls meets the ℓ_1-norm," *IEEE Trans. Sig. Proc.*, vol. 58, no. 7, pp. 3436–3447, 2010.

[100] Y. Murakami, M. Yamagishi, M. Yukawa, and I. Yamada, "A sparse adaptive filtering using time-varying soft-thresholding techniques," in *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*. IEEE, 2010, pp. 3734–3737.

[101] Y. Kopsinis, K. Slavakis, and S. Theodoridis, "Online sparse system identification and signal reconstruction using projections onto weighted balls," *IEEE Trans. Sig. Proc.*, vol. 59, no. 3, pp. 936–952, 2011.

[102] K. Slavakis, Y. Kopsinis, S. Theodoridis, and S. McLaughlin, "Generalized thresholding and online sparsity-aware learning in a union of subspaces," *IEEE Trans. Sig. Proc.*, vol. 61, no. 15, pp. 3760–3773, 2013.

[103] G. Mileounis, B. Babadi, N. Kalouptsidis, and V. Tarokh, "An adaptive greedy algorithm with application to nonlinear communications," *IEEE Trans. Sig. Proc.*, vol. 58, no. 6, pp. 2998–3007, 2010.

[104] C. Paleologu, S. Ciochină, and J. Benesty, "An efficient proportionate affine projection algorithm for echo cancellation," *Signal Processing Letters, IEEE*, vol. 17, no. 2, pp. 165–168, 2010.

[105] M. V. Lima, T. N. Ferreira, W. Martins, P. S. Diniz *et al.*, "Sparsity-aware data-selective adaptive filters," *IEEE Trans. Sig. Proc.*, vol. 62, no. 17, pp. 4557–4572, 2014.

[106] B. Dumitrescu, A. Onose, P. Helin, and I. Tăbuş, "Greedy sparse rls," *IEEE Trans. Sig. Proc.*, vol. 60, no. 5, pp. 2194–2207, 2012.

[107] Y. Chen and A. O. Hero, "Recursive ell-1,infinity Group Lasso," *IEEE Trans. Sig. Proc.*, vol. 60, no. 8, pp. 3978–3987, 2012.

[108] K. E. Themelis, A. Rontogiannis, K. D. Koutroumbas *et al.*, "A variational bayes framework for sparse adaptive estimation," *IEEE Trans. Sig. Proc.*, vol. 62, no. 18, pp. 4723–4736, 2014.

[109] R. Abdolee, B. Champagne, and A. H. Sayed, "Estimation of space-time varying parameters using a diffusion lms algorithm," *IEEE Trans. Sig. Proc.*, vol. 62, no. 2, pp. 403–418, 2014.

[110] S. Chouvardas, K. Slavakis, Y. Kopsinis, and S. Theodoridis, "A sparsity promoting adaptive algorithm for distributed learning," *IEEE Trans. Sig. Proc.*, vol. 60, no. 10, pp. 5412–5425, 2012.

[111] M. Lustig, D. Donoho, and J. M. Pauly, "Sparse MRI: The application of compressed sensing for rapid mr imaging," *Magnetic Resonance in Medicine*, vol. 58(6), pp. 1182–1195, December 2007.

[112] J. Yang and Y. Zhang, "Alternating direction algorithms for l1 problems in compressive sensing," Rice University, Tech. Rep., June 2010.

[113] M. Stojnic, F. Parvaresh, and B. Hassibi, "On the reconstruction of block-sparse signals with an optimal number of measurements," in *arXiv:0804.0041*, 2008.

[114] M. Stojnic, "Strong thresholds for ℓ_2/ℓ_1-optimization in block-sparse compressed sensing," in *IEEE Intl. Conf. Acoustics, Speech, Sig. Proc. (ICASSP)*, 2009.

[115] R. Baraniuk, V. Cevher, M. Duarte, and C. Hegde, "Model-based compressive sensing," *IEEE Trans. Info. Th.*, March 2010.

[116] M. Stojnic, "Block-length dependent thresholds for ℓ_2/ℓ_1-optimization in block-sparse compressed sensing," in *ICASSP*, 2010.

[117] C. Hegde, P. Indyk, and L. Schmidt, "Approximation-tolerant model-based compressive sensing," in *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2014.

[118] L. Schmidt, C. Hegde, and P. Indyk, "The constrained earth mover distance model, with applications to compressive sensing," in *CAMSAP*, 2013.

[119] J. Kim, O. K. Lee, and J. C. Ye, "Dynamic sparse support tracking with multiple measurement vectors using compressive music," in *IEEE Intl. Conf. Acoustics, Speech, Sig. Proc. (ICASSP)*, 2012, pp. 2717–2720.

[120] D. Angelosante, G. Giannakis, and E. Grossi, "Compressed sensing of time-varying signals," in *Dig. Sig. Proc. Workshop*, 2009.

[121] J. Wright and Y. Ma, "Dense error correction via ℓ_1-minimization," *IEEE Trans. Info. Th.*, vol. 56, no. 7, pp. 3540–3560, Jul. 2010.

[122] D. L. Donoho and X. Huo, "Uncertainty principles and ideal atomic decomposition," *IEEE Trans. Info. Th.*, vol. 47, no. 7, pp. 2845–2862, 2001.

[123] M. Elad and A. M. Bruckstein, "A generalized uncertainty principle and sparse representation in pairs of bases," *IEEE Trans. Info. Th.*, vol. 48, no. 9, pp. 2558–2567, 2002.

[124] R. Gribonval and M. Nielsen, "Sparse representations in unions of bases," *IEEE Trans. Info. Th.*, vol. 49, no. 12, pp. 3320–3325, 2003.

[125] V. Chandrasekaran, S. Sanghavi, P. A. Parrilo, and A. S. Willsky, "Rank-sparsity incoherence for matrix decomposition," *SIAM Journal on Optimization*, vol. 21, 2011.

[126] J. Feng, H. Xu, and S. Yan, "Online robust pca via stochastic optimization," in *Adv. Neural Info. Proc. Sys. (NIPS)*, 2013.

[127] C. Qiu and N. Vaswani, "Recursive sparse recovery in large but correlated noise," in *Allerton Conf. on Communication, Control, and Computing*, 2011.

[128] T. Blumensath, "Compressed sensing with nonlinear observations and related nonlinear optimisation problems," *arXiv:1205.1650*, 2012.

[129] W. Xu, M. Wang, and A. Tang, "Sparse recovery from nonlinear measurements with applications in bad data detection for power networks," *arXiv:1112.6234 [cs.IT]*, 2011.

[130] L. Li and B. Jafarpour, "An iteratively reweighted algorithm for sparse reconstruction of subsurface flow properties from nonlinear dynamic data," *arXiv:0911.2270*, 2009.

[131] T. Blumensath, "Compressed sensing with nonlinear observations and related nonlinear optimization problems," in *Tech. Rep. arXiv:1205.1650*, 2012.

[132] A. Beck and Y. C. Eldar, "Sparsity constrained nonlinear optimization: Optimality conditions and algorithms," in *Tech. Rep. arXiv:1203.4580*, 2012.

[133] H. Ohlsson, M. Verhaegen, and S. S. Sastry, "Nonlinear compressive particle filtering," in *IEEE Conf. Decision and Control (CDC)*, 2013.

[134] D. Sejdinovic, C. Andrieu, and R. Piechocki, "Bayesian sequential compressed sensing in sparse dynamical systems," in *Allerton Conf. Communication, Control, and Computing*, 2010.

[135] E. J. Candes, T. Strohmer, and V. Voroninski, "Phaselift: Exact and stable signal recovery from magnitude measurements via convex programming," *Communications on Pure and Applied Mathematics*, 2011.

[136] H. Ohlsson, A. Y. Yang, R. Dong, M. Verhaegen, and S. S. Sastry, "Quadratic basis pursuit," in *SPARS*, 2012.

[137] M. L. Moravec, J. K. Romberg, and R. G. Baraniuk, "Compressive phase retrieval," in *Wavelets XII in SPIE International Symposium on Optical Science and Technology*, 2007.

[138] K. Jaganathan, S. Oymak, and B. Hassibi, "Recovery of sparse 1-d signals from the magnitudes of their fourier transform," in *Information Theory Proceedings (ISIT), 2012 IEEE International Symposium on*. IEEE, 2012, pp. 1473–1477.

[139] ——, "Sparse phase retrieval: Convex algorithms and limitations," *arXiv preprint arXiv:1303.4128*, 2013.

[140] E. J. Candes, Y. C. Eldar, T. Strohmer, and V. Voroninski, "Phase retrieval via matrix completion," *SIAM Journal on Imaging Sciences*, vol. 6, no. 1, pp. 199–225, 2013.

[141] E. J. Candes, X. Li, and M. Soltanolkotabi, "Phase retrieval via wirtinger flow: Theory and algorithms," *IEEE Trans. Info. Th.*, vol. 61, no. 4, pp. 1985–2007, 2015.

[142] H. Mansour and R. Saab, "Weighted one-norm minimization with inaccurate support estimates: Sharp analysis via the null-space property," in *IEEE Intl. Conf. Acoustics, Speech, Sig. Proc. (ICASSP)*, 2015.

[143] M. A. Davenport, P. T. Boufounos, M. B. Wakin, and R. G. Baraniuk, "Signal processing with compressive measurements," *IEEE Journal of Selected Topics in Signal Processing*, pp. 445–460, April 2010.