Sparse Recovery of Streaming Signals Using ℓ_1-Homotopy

M. Salman Asif and Justin Romberg

Abstract

Most of the existing methods for sparse signal recovery assume a *static* system: the unknown signal is a finite-length vector for which a fixed set of linear measurements and a sparse representation basis are available and an ℓ_1-norm minimization program is solved for the reconstruction. However, the same representation and reconstruction framework is not readily applicable in a *streaming* system: the unknown signal changes over time, and it is measured and reconstructed sequentially over small time intervals. A streaming framework for the reconstruction is particularly desired when dividing a streaming signal into disjoint blocks and processing each block independently is either infeasible or inefficient.

In this paper, we discuss two such streaming systems and a homotopy-based algorithm for quickly solving the associated weighted ℓ_1-norm minimization programs: 1) Recovery of a smooth, time-varying signal for which, instead of using block transforms, we use lapped orthogonal transforms for sparse representation. 2) Recovery of a sparse, time-varying signal that follows a linear dynamic model. For both the systems, we iteratively process measurements over a sliding interval and solve a weighted ℓ_1-norm minimization problem for estimating sparse coefficients. Since we estimate overlapping portions of the streaming signal while adding and removing measurements, instead of solving a new ℓ_1 program from scratch at every iteration, we use an available signal estimate as a starting point in a homotopy formulation. Starting with a warm-start vector, our homotopy algorithm updates the solution in a small number of computationally inexpensive homotopy steps as the system changes. The homotopy algorithm presented in this paper is highly versatile as it can update the solution for the ℓ_1 problem in a number of dynamical settings. We demonstrate with numerical experiments that our proposed streaming recovery framework outperforms the methods that represent and reconstruct a signal as independent, disjoint blocks, in terms of quality of reconstruction, and that our proposed homotopy-based updating scheme outperforms current state-of-the-art solvers in terms of the computation time and complexity.

M. S. Asif and J. Romberg are with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA. Email: {sasif,jrom}@gatech.edu.

I. Introduction

In this paper we discuss the problem of estimating a sparse, time-varying signal from incomplete, streaming measurements—a problem that arises in a variety of signal processing applications; see [1–9] for examples. Most of the existing sparse recovery methods are *static* in nature: they assume that the unknown signal is a finite-length vector, for which a fixed set of linear measurements and a sparse representation basis are available, and solve an ℓ_1-norm minimization problem, which encourages the solution to be sparse while maintaining fidelity toward the measurements [10–12]. However, the same representation and reconstruction framework is not readily applicable in a *streaming* system in which the unknown signal varies over time and has no clear beginning and end. Instead of measuring the entire signal or processing the entire set of measurements at once, we perform these task sequentially over short, shifting time intervals [13, 14].

We consider the following time-varying linear observation model for a discrete-time signal $x[n]$:

$$y_t = \Phi_t x_t + e_t, \tag{1}$$

where x_t is a vector that represents $x[n]$ over an interval of time, y_t is a vector that contains measurements of x_t, Φ_t is a measurement matrix, and e_t is noise in the measurements. We use subscript t to indicate that the system in (1) represents a small part of an infinite-dimensional streaming system, in which for any t, x_t precedes x_{t+1} in $x[n]$ and the two may overlap. If we treat the x_t independent from the rest of the streaming signal $(x[n])$, we can solve (1) as a stand-alone system for every t as follows. Suppose we can represent each x_t as $\Psi_t \alpha_t$, where Ψ_t denotes a representation matrix (e.g., a discrete cosine or a wavelet transform) for which α_t is a sparse vector of transform coefficients. We write the equivalent system for (1) as

$$y_t = \Phi_t \Psi_t \alpha_t + e_t \tag{2}$$

and solve the following weighted ℓ_1-norm minimization problem for a sparse estimate of α_t:

$$\underset{\alpha_t}{\text{minimize}} \ \|W_t \alpha_t\|_1 + \frac{1}{2}\|\Phi_t \Psi_t \alpha_t - y_t\|_2^2. \tag{3}$$

The ℓ_1 term promotes sparsity in the estimated coefficients; W_t is a diagonal matrix of positive weights that can be adapted to promote a certain sparse structure in the solution [15, 16]; and the ℓ_2 term ensures that the solution remains close to the measurements. The optimization problem in (3) is convex and can be solved using a variety of solvers [17–21].

The method described above represents and reconstructs the signal blocks (x_t) independently, which is natural if both the measurement system in (1) and the representation system in (2) are block-diagonal; that is, the x_t are non-overlapping in (1) and each x_t is represented as a sparse vector using a block transform in (2). However, estimating the x_t independently is not optimal if the streaming system for (1) or (2) is not block diagonal, which can happen if Φ_t, Ψ_t, or both of them overlap across the x_t. An illustration of such an overlapping measurement and representation system is presented in Fig. 1. Figure 1a depicts a measurement system in which the Φ_t overlap (the x_t, which are not labeled in the figure, are the overlapping portions of $x[n]$ that constitute the y_t). Figure 1b depicts a representation of $x[n]$ using lapped orthogonal transform (LOT) bases [22, 23] in which the Ψ_t overlap (and multiple $\Psi_t\alpha_t$ may add up to constitute a portion of $x[n]$).

In this paper we present ℓ_1-norm minimization based sparse recovery algorithms for the following two types of overlapping, streaming systems:

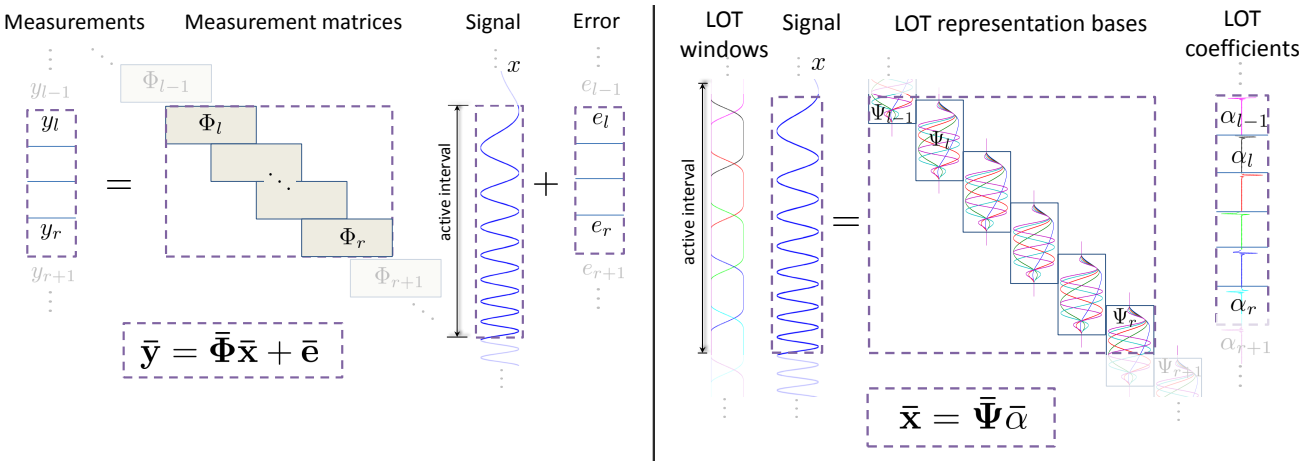
1. Recovery of smooth, time-varying signals from streaming measurements in (1) using sparse representation bases with compact but overlapping supports.

2. Recovery of time-varying signals from streaming measurements in (1) in the presence of the following linear dynamic model:

$$x_{t+1} = F_t x_t + f_t, \tag{4}$$

where F_t is a prediction matrix that couples x_t and x_{t+1} and f_t is the error in the prediction, which we assume has a bounded ℓ_2 norm.

In both these systems, we assume that sets of measurements are sequentially recorded over short, shifting (possibly overlapping) intervals of the streaming signal according to the system in (1). Instead of estimating each block (x_t) independently, we iteratively estimate the signal $(x[n])$ over small, sliding intervals, which allows us to link together the blocks that share information. At every iteration, we build a system model that describes the measurements and the sparse coefficients of the streaming signal over an active interval (one such example is depicted in Fig. 1). We estimate the sparse coefficients for the signal over the active interval by solving a weighted ℓ_1-norm minimization problem, and then we shift the active interval by removing the oldest set of measurements from the system and adding a new one. For instance, to jointly solve the systems in (1) and (4) over an active interval that includes P instances of x_t, say $x_1, \ldots, x_P$, which are non-overlapping and represented as $x_t = \Psi_t\alpha_t$, we solve the following



(a) Overlapping measurements of a signal.

(b) LOT-based representation of a signal.

Fig. 1: Illustration of an overlapping measurement system **(a)** and a lapped orthogonal transform (LOT)-based representation system **(b)**. Boxed regions represent the system over the active interval. Subscripts l and r indicate the left and the right border of the active interval.

modified form of (3):

$$\underset{\alpha_1,\ldots,\alpha_P}{\text{minimize}} \sum_{p=1}^{P} \|W_p\alpha_p\|_1 + \frac{1}{2}\|\Phi_p\Psi_p\alpha_p - y_p\|_2^2 + \frac{\lambda_p}{2}\|F_{p-1}\Psi_{p-1}\alpha_{p-1} - \Psi_p\alpha_p\|_2^2, \tag{5}$$

where the $\lambda_p > 0$ denote the regularization parameters. To separate α_0 from the active system, we fix its value to an estimate $\widehat{\alpha}_0$. At the next streaming iteration, we may have $x_2, \ldots, x_{P+1}$ in the active interval and an estimate of all but x_{P+1}. However, before solving the optimization problem, an estimate of the signal over the entire active interval can be predicted. We use the available signal estimate to aide the recovery process in two ways: We update the W_t using available estimates of the sparse coefficients α_t (in the same spirit as iterative reweighting [16]), and we use the available estimates of the α_t as a starting point to expedite the solution of the ℓ_1 problem.

One key contribution in this paper is a homotopy algorithm for quickly solving the weighted ℓ_1-norm minimization problems for the streaming signal recovery; see [24, 25] for the background on the LASSO homotopy. Since we sequentially estimate overlapping components of a streaming signal while adding and removing measurements, instead of solving a new optimization problem at every iteration, we use the existing signal estimate as a starting point (warm-start) in our homotopy formulation. The homotopy algorithm that we present in this paper extends and unifies previous work on dynamic ℓ_1 updating [26–32]: Our newly proposed ℓ_1 homotopy algorithm can quickly update the solution of the ℓ_1 programs

of the form (3) for arbitrary changes in y_t, W_t, Φ_t, Ψ_t. For instance, adding or removing sequential measurements [26, 27, 29, 30], changes in the measurements (y_t) as the signal (x_t) changes with a fixed measurement matrix [28, 29], arbitrary changes in the measurement matrix (Φ_t) or the representation matrix (Ψ_t) [30, 31], or changes in the weights (W_t) [32]. Unlike previous approaches, we do not impose any restriction on the warm-start vector to be a solution of an ℓ_1 problem; as we can accommodate an arbitrary vector to initialize the homotopy update. Of course, the update will be quicker when the starting vector is close to the final solution. For solving a weighted ℓ_1 program at every streaming iteration, we will use a thresholded and transformed version of the solution from the previous iteration as a warm-start in a homotopy program. We will sequentially add and remove multiple measurements in the system, update the weights in the ℓ_1 term, and update the solution in a sequence of a small number of computationally inexpensive homotopy steps.

The problem formulations and the recovery methods presented in this paper compare with some of the existing signal estimation schemes. Recursive least squares (RLS) and the Kalman filter are two classical estimation methods that are oblivious to any sparse structure in the signal and solve the systems in (1) and (4) in the least-squares sense [33–35]. One attractive feature of these methods is that their solutions admit closed form representation that can be updated recursively. The homotopy algorithm solves an ℓ_1 problem and its updating scheme is not as simple as that of the RLS and the Kalman filter, but the update has the same recursive spirit and reduces to a series of low-rank updates. Moreover, under certain conditions, the recursive estimate of the standard Kalman filter that is computed using only a new set of measurements, a previous estimate of the signal, and the so-called information matrix is optimal for all the previous measurements—as if it were computed by solving a least-squares problem for all the measurements simultaneously [34, 36]. In contrast, the signal estimate for the ℓ_1 problems (such as (5)) is optimal only for the measurements available in the system over the active signal interval.

Sparse recovery of smooth, time-varying signal from streaming, overlapping measurements has been discussed in [14, 37], but the sparse recovery algorithm used there is a streaming variant of a greedy matching pursuit algorithm [38]. Recently, several methods have been proposed to incorporate signal dynamics into the sparse signal estimation framework [39–45]. The method in [39] identifies the support of the signal by solving an ℓ_1 problem and modifies the Kalman filter to estimate the signal on the identified support; [40] embeds additional steps within the original Kalman filter algorithm for promoting a sparse solution; [44] uses a belief propagation algorithm to identify the support and update the signal estimate; [42] compares different types of sparse dynamics by solving an ℓ_1 problem for one signal block; [43] assumes that the prediction error is sparse and jointly estimates multiple signal blocks using

a homotopy algorithm; and [41] solves a group-sparse ℓ_1 problem for a highly restrictive dynamic signal model in which locations of nonzero components of the signal do not change. In our problem formulation, we consider a general dynamic model in (4) and solve a problem of the form in (5) over a sliding, active interval. Our emphasis is on an efficient updating scheme for moving from one solution to the next as new measurements are added and old ones are removed.

The paper is organized as follows. We discuss signal representation using bases with overlapping supports in Section II, the recovery framework for the two systems in Section III, and the homotopy algorithm in Section IV. We present experimental results to demonstrate the performance of our algorithms, in terms of the quality of reconstructed signals and the computational cost of the recovery process, in Section V.

II. Signal representation using compactly supported bases

We will represent a discrete-time signal $x[n]$ as

$$x[n] = \sum_{p\in\mathbb{Z}} \sum_{0\le k<l_p} \alpha_{p,k}\psi_{p,k}[n], \tag{6}$$

where the set of functions $\psi_{p,k}$ forms an orthogonal basis of $\ell_2(\mathbb{Z})$ and the $\alpha_{p,k} = \langle \psi_{p,k}, x\rangle$ denote the corresponding basis coefficients that we expect to be sparse or compressible. For a fixed $p \in \mathbb{Z}$, $\{\psi_{p,k}\}_k$ denotes a set of orthogonal basis vectors that have a compact support over an interval I_p. The supports of the $\psi_{p,k}$ and the $\psi_{p',k}$ (i.e., I_p and $I_{p'}$) may overlap if $p \neq p'$. An example of such a signal representation using lapped orthogonal bases is depicted in Fig. 1b, where a Ψ_p denotes the basis functions in $\{\psi_{p,k}\}_k$ supported on I_p, an α_p denotes the respective $\{\alpha_{p,k}\}_k$, and the overlapping windows denote the intervals I_p.

While we assume a general framework for the signal representation using (6) in the derivations of the recovery problems, we use the lapped orthogonal transform (LOT) [22] in most of our explanations and numerical experiments. A LOT decomposes a signal into orthogonal components with compact, overlapping supports. Orthogonality between the components in the overlapping regions is maintained due to projections with opposite (i.e., even and odd) symmetries. LOT basis functions can be designed using modified cosine-IV basis functions that are multiplied by smooth, overlapping windows. The advantage of using the LOT instead of a simple block-based discrete cosine or Fourier transform is that block-based transforms use rectangular windows to divide a signal into disjoint blocks and that can introduce artificial discontinuities at the boundaries of the blocks and ruin the sparsity [23].

A discrete LOT basis can be designed as follows. Divide the support of the signal into consecutive, overlapping intervals $I_p = [a_p - \eta_p, a_{p+1} + \eta_{p+1}]$, where $\{a_p\}_{p\in\mathbb{Z}}$ is a sequence of half integers (i.e., $a_p + 1/2 \in \mathbb{Z}$) and $\{\eta_p\}_{p\in\mathbb{Z}}$ is a sequence of transition width parameters such that $l_p \stackrel{\text{def}}{=} a_{p+1} - a_p \geq \eta_p + \eta_{p+1}$. The LOT basis function, $\psi_{p,k}$ in (6), for every p, k is defined as

$$\psi_{p,k}[n] = g_p[n]\sqrt{\frac{2}{l_p}}\cos\left[\pi\left(k+\frac{1}{2}\right)\frac{n-a_p}{l_p}\right], \tag{7}$$

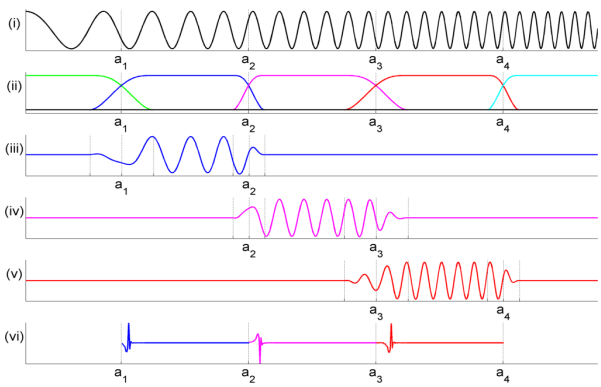
which is a translated and dilated cosine-IV basis function, multiplied by a smooth window g_p that is supported on I_p. For a careful choice of g_p, coupled with the even and odd symmetry of cosine-IV basis functions with respect to a_p and a_{p+1}, respectively, the set of functions $\psi_{p,k}$ forms an orthonormal basis of $\ell_2(\mathbb{Z})$ (see [23, Sec. 8.4] for further details).

To compute the LOT coefficients of $x[n]$ over an arbitrary interval Π, we assume a partition of Π into appropriate LOT subintervals I_p. Figure 2a depicts an example with such a partition of a time interval using LOT windows and the decomposition of a linear chirp signal into overlapping components using LOT bases and their respective coefficients. Since the set of functions $\{\psi_{p,k}\}_k$ defines an orthogonal basis for a LOT subspace on respective I_p, the corresponding LOT projection of $x[n]$ can be written as

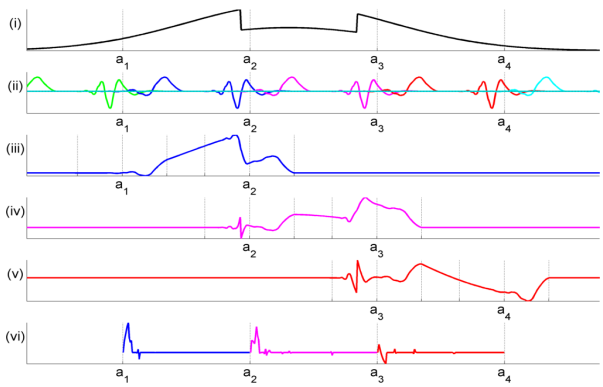
$$\widetilde{x}_p[n] = \sum_{k=0}^{l_p-1} \underbrace{\langle x, \psi_{p,k}\rangle}_{\alpha_{p,k}} \psi_{p,k}[n], \tag{8}$$

where $\widetilde{x}_p[n]$ is supported on I_p. We can represent the restriction of $\widetilde{x}_p[n]$ on I_p as $\Psi_p\alpha_p$, where Ψ_p is a synthesis matrix whose k^{th} column consists of $\psi_{p,k}[n]$ restricted to I_p and α_p is an l_p-length vector of LOT coefficients that consists of $\{\alpha_{p,k}\}_k$ for $0 \leq k < l_p$. Note that the $\widetilde{x}_p[n]$ are the overlapping, orthogonal components of $x[n]$, and to synthesize $x[n]$ over Π, we have to add all the $\widetilde{x}_p[n]$ that overlap Π. Referring to Fig. 1b, $\bar{\mathbf{x}}$ denotes $x[n]$ over the active interval Π, $\bar{\alpha}$ denotes a vector that contains all the α_p that contribute to $\bar{\mathbf{x}}$ stacked on top of one another, $\bar{\mathbf{\Psi}}$ contains the corresponding Ψ_p (in part or full) at appropriate columns and rows, and the columns of Ψ_p and Ψ_{p+1} overlap in $2\eta_{p+1}$ rows.

Another example of an orthogonal basis that can be naturally separated into overlapping, compact intervals is the wavelet transform. A wavelet transform decomposes a signal into orthogonal components with overlapping supports at different resolutions in time and frequency [23, 46]. The scaling and wavelet functions used for this purpose overlap one another while maintaining orthogonality. Although commonly used filter-bank implementations assume that the finite-length signals are symmetrically or periodically extended during convolution, which yields a block-based wavelet transform, we can write wavelet bases



(a) LOT projections and coefficients. (i) A discrete-time linear chirp signal ($x[n]$). (ii) LOT windows over different intervals (I_p); distance between dotted lines around a_p represent η_p. (iii–v) LOT projections $\tilde{x}_p[n]$ over respective intervals. (vi) Sparse coefficients ($\alpha_{p,k}$).



(b) Wavelet projections and coefficients. (i) A piece-wise smooth signal ($x[n]$). (ii) A subset of scaling and wavelet functions at the coarsest scale. Dotted lines denote I_p. (iii–v) Wavelet projections $\tilde{x}_p[n]$ over respective intervals. (vi) Sparse coefficients ($\alpha_{p,k}$).

Fig. 2: Signal decomposition in **(a)** LOT and **(b)** wavelet bases.

in terms of shifted, dilated wavelet and scaling functions that overlap across adjacent blocks. Figure 2b depicts an example of the decomposition of a piece-wise smooth signal into overlapping components using wavelet bases and their respective coefficients.

III. Sparse signal recovery from streaming measurements

In a streaming system, we iteratively estimate sparse coefficients of the signal over an active, sliding interval. We describe a system for the measurements and the sparse representation of the signal over the active interval and solve a weighted ℓ_1-norm minimization problem for estimating the sparse coefficients. At every iteration of the streaming recovery process, we shift the active interval by removing a few oldest measurements and adding a few new ones in the system. Estimate of the sparse coefficients and the signal portion that leave the active interval are committed to the output. The length of the active interval determines the delay, memory, and computational complexity of the system.

Consider the linear system in (1): $y_t = \Phi_t x_t + e_t$, where x_t denotes a portion of $x[n]$ over a short interval and the consecutive x_t may also overlap. We denote $x[n]$ over the active interval Π as $\bar{\mathbf{x}}$ and assume that $\bar{\mathbf{x}}$ consists of a small number of x_t. We describe the equivalent system for $\bar{\mathbf{x}}$ in the following compact form:

$$\bar{\mathbf{y}} = \bar{\mathbf{\Phi}}\bar{\mathbf{x}} + \bar{\mathbf{e}}, \tag{9}$$

where $\bar{\mathbf{y}}$ denotes a vector that contains y_t for the x_t that belong to $\bar{\mathbf{x}}$, $\bar{\mathbf{\Phi}}$ denotes a matrix that contains the corresponding Φ_t, and $\bar{\mathbf{e}}$ denotes the noise vector. At every iteration of the streaming recovery algorithm, we shift Π by removing the oldest y_t in the system and adding a new one and update the system in (9) accordingly. An example of such a measurement system in depicted in Fig. 1a, where the active system is represented in a boxed region. To represent the signal $\bar{\mathbf{x}}$ using the model in (6), we use the following compact form:

$$\bar{\mathbf{x}} = \bar{\mathbf{\Psi}}\bar{\alpha}, \tag{10}$$

where $\bar{\alpha}$ contains the $\alpha_{p,k}$ that synthesize $\bar{\mathbf{x}}$ and the synthesis matrix $\bar{\mathbf{\Psi}}$ contains the corresponding $\psi_{p,k}$ restricted to Π as its columns. An example of such a representation system in depicted in Fig. 1b.

In the following two sections, we discuss the problem formulation for the recovery of a streaming signal from streaming measurements when 1) the signal is represented using lapped orthogonal bases and 2) the signal changes according to a linear dynamic model.

A. Streaming signal with lapped orthogonal bases

1) System model: Given the system in (9) for active interval Π, we use lapped orthogonal bases for signal representation in (10) and describe the system in the following equivalent form:

$$\bar{\mathbf{y}} = \bar{\mathbf{\Phi}}\bar{\mathbf{\Psi}}\bar{\alpha} + \bar{\mathbf{e}}. \tag{11}$$

An example of such a system is depicted in Fig. 3a. Note that even when $\bar{\mathbf{\Phi}}$ is a block diagonal matrix, the system in (11) cannot be separated into independent blocks if $\bar{\mathbf{\Psi}}$ has overlapping columns.

One important consideration in our system is the design of $\bar{\mathbf{\Psi}}$ with respect to the decomposition of Π into overlapping intervals I_p. Our motivation is to have as few unknown coefficients in $\bar{\alpha}$ as possible. Note that if an interval I_p overlaps with Π (partially or fully), we have to include its corresponding coefficient vector α_p of length l_p into $\bar{\alpha}$. Since we can divide the interior of Π in an arbitrary fashion, the special consideration is only for the I_p that partially overlap with Π on its left and right borders.

On the right end of Π, we align the right-most interval, say I_r, such that it partially overlaps with Π but the interval after that, say I_{r+1}, lies completely outside Π. In such a case $\bar{\alpha}$ would contain α_r but not α_{r+1}. Such a relationship between the active interval (Π) and the subintervals (I_p) is depicted in Fig. 1b, where we adjusted the right-most interval such that the overlapping region on its right side lies outside the active interval Π.

On the left end of Π, we align the left-most interval, say I_l, such that it is fully included in Π. However, in such a setting I_{l-1} will partially overlap with Π and the corresponding coefficient vector α_{l-1} of length l_{l-1} will be included in $\bar{\alpha}$. Suppose we have committed the estimate of α_{l-1} to the output, and we want to remove it from the system in (11). If the system in (11) were block-diagonal, we could simply update the system by removing α_{l-1} from $\bar{\alpha}$ and the corresponding rows from $\bar{\mathbf{y}}$ and $\bar{\mathbf{\Phi}}\bar{\mathbf{\Psi}}$. But if the system in (11) has overlapping rows, where the rows are coupled with more than one set of variables, instead of removing the rows, we remove the columns. Thus, removing α_{l-1} is equivalent to removing the first l_{l-1} coefficients from the vector $\bar{\alpha}$, removing the first l_{l-1} columns from the matrix $\bar{\mathbf{\Phi}}\bar{\mathbf{\Psi}}$ in (11), and modifying the measurement vector $\bar{\mathbf{y}}$ accordingly. To do this we divide $\bar{\mathbf{x}}$ into two parts as

$$\bar{\mathbf{x}} = \bar{\mathbf{\Psi}}\bar{\alpha} = \begin{bmatrix} \breve{\mathbf{\Psi}} & \tilde{\mathbf{\Psi}} \end{bmatrix} \begin{bmatrix} \breve{\alpha} \\ \tilde{\alpha} \end{bmatrix} = \breve{\mathbf{\Psi}}\breve{\alpha} + \tilde{\mathbf{\Psi}}\tilde{\alpha}, \tag{12}$$

where we divided $\bar{\mathbf{\Psi}}$ into two matrices $\breve{\mathbf{\Psi}}$ and $\tilde{\mathbf{\Psi}}$ and $\bar{\alpha}$ into the corresponding vectors $\breve{\alpha}$ and $\tilde{\alpha}$. An example of such a decomposition is depicted in Fig. 3b. To remove α_{l-1} from the system in (11), we modify $\bar{\mathbf{y}}$ as follows. Since we only have an estimate of α_{l-1}, which we denote as $\widehat{\alpha}_{l-1}$, we remove its expected contribution from the system by modifying $\bar{\mathbf{y}}$ as

$$\tilde{\mathbf{y}} \stackrel{\text{def}}{=} \bar{\mathbf{y}} - \bar{\mathbf{\Phi}}\breve{\mathbf{\Psi}}\breve{\alpha}, \tag{13}$$

where we use $\breve{\mathbf{\Psi}}$ to denote the first l_{l-1} columns in $\bar{\mathbf{\Psi}}$, which contains a part of Ψ_{l-1}, and $\breve{\alpha}$ to denote $\widehat{\alpha}_{l-1}$. We write the resultant, modified form of the system in (11) as

$$\tilde{\mathbf{y}} = \bar{\mathbf{\Phi}}\tilde{\mathbf{\Psi}}\tilde{\alpha} + \tilde{\mathbf{e}}, \tag{14}$$

where $\tilde{\mathbf{e}}$ denotes combined error in the system and $\tilde{\alpha}$ denotes the unknown vector of coefficients that we estimate by solving a weighted ℓ_1-norm minimization problem.

2) Recovery problem: To estimate $\tilde{\alpha}$ from the system in (14), we solve the following optimization problem:

$$\underset{\alpha}{\text{minimize}}\ \|\mathbf{W}\alpha\|_1 + \frac{1}{2}\|\bar{\mathbf{\Phi}}\tilde{\mathbf{\Psi}}\alpha - \tilde{\mathbf{y}}\|_2^2, \tag{15}$$

where $\mathbf{W}$ is a diagonal matrix that consists of positive weights. We select the weights using prior knowledge about the estimate of $\tilde{\alpha}$ from the previous streaming iteration. Let us denote $\widehat{\alpha}$ as our prior estimate of $\tilde{\alpha}$. Since there is a significant overlap between the active intervals at the present and the

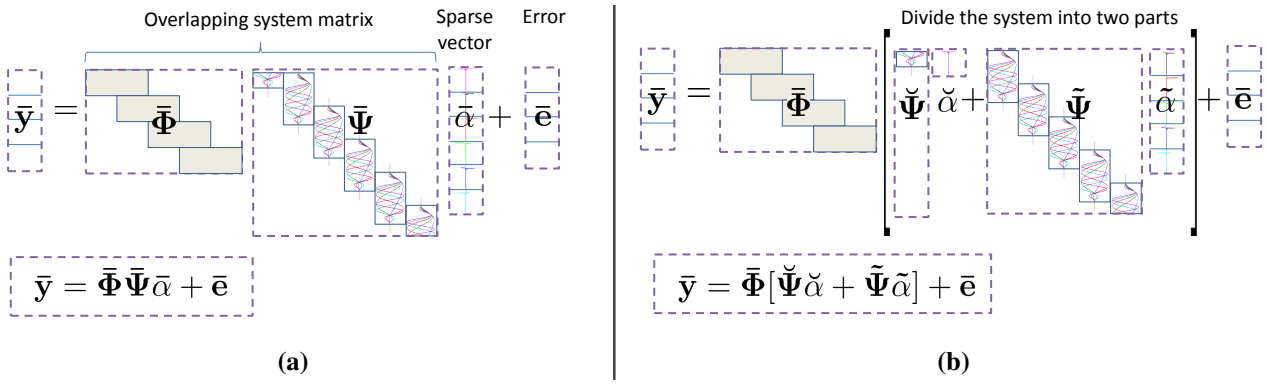


Fig. 3: Illustration of the system used for the signal reconstruction. **(a)** System over the active interval. **(b)** System divided into two parts so that $\breve{\alpha}$ can be removed.

previous iterations, we expect $\widehat{\alpha}$ to be very close to the solution of (15). We compute i^{th} diagonal entry in $\mathbf{W}$ as

$$\mathbf{w}_i \leftarrow \frac{\tau}{\beta|\widehat{\alpha}_i| + 1}, \tag{16}$$

where $\tau > 0$ and $\beta >> 1$ are two parameters that can be used to tune the weights according to the problem. Instead of solving (15) from scratch, we can speed up the recovery process by providing $\widehat{\alpha}$ as a warm-start (initialization) vector to an appropriate solver.

We compute $\widehat{\alpha}$ using the signal estimate from the previous streaming iteration and the available set of measurements. Since we have an estimate of $\bar{\mathbf{x}}$ for the part of Π that is common between the current and the previous iteration, our main task is to predict the signal values that are new to the system. Let us denote the available signal estimate for $\bar{\mathbf{x}}$ as $\widehat{\mathbf{x}}$. We can assign values to the new locations in $\widehat{\mathbf{x}}$ using zero padding, periodic extension, or symmetric extension, and compute $\widehat{\alpha}$ from $\widehat{\mathbf{x}} = \bar{\mathbf{\Psi}}\widehat{\alpha}$. In our experiments, first we update $\widehat{\mathbf{x}}$ by symmetric signal extension onto new locations and identify a candidate support for the new coefficients in $\widehat{\alpha}$; then we calculate magnitudes of the new coefficients by solving a least-squares problem, restricted to the chosen support, using the corresponding measurements in (11); and finally we truncate extremely small values of the least-squares solution and update $\widehat{\alpha}$.

B. Streaming signals with linear dynamic model

1) System model: To include a linear dynamic model for the time-varying signal into the system, we append equations for the dynamic model to the system in (9). Consider the dynamic model in (4): $x_{t+1} = F_t x_t + f_t$, where the consecutive x_t are non-overlapping. At every streaming iteration, we describe a combined system of prediction equations for the x_t that belong to $\bar{\mathbf{x}}$ as follows. Suppose $\bar{\mathbf{x}}$ contains

$x_l, \ldots, x_r$, and an estimate of x_{l-1}, which was removed from $\bar{\mathbf{x}}$ and committed to the output, is given as $\widehat{x}_{l-1}$. We rearrange the equations in (4) for $t = l$ as $-F_{l-1}\widehat{x}_{l-1} = -x_l + f_{l-1}$ and for the rest of t as $0 = F_t x_t - x_{t+1} + f_t$. We stack these equations on top of one another to write the following compact form:

$$\bar{\mathbf{q}} = \bar{\mathbf{F}}\bar{\mathbf{x}} + \bar{\mathbf{f}}. \tag{17}$$

$\bar{\mathbf{F}}$ denotes a banded matrix that consists of negative identity matrices in the main diagonal and $F_l, \ldots, F_r$ below the diagonal; $\bar{\mathbf{f}}$ denotes the combined prediction error; $\bar{\mathbf{q}}$ denotes a vector that contains $-F_{l-1}\widehat{x}_{l-1}$ followed by zeros.

Combining the systems in (9) and (17) with the sparse representation ($\bar{\mathbf{x}} = \bar{\mathbf{\Psi}}\bar{\alpha}$), we write the modified system over active interval Π as

$$\begin{bmatrix} \bar{\mathbf{y}} \\ \bar{\mathbf{q}} \end{bmatrix} = \begin{bmatrix} \bar{\mathbf{\Phi}} \\ \bar{\mathbf{F}} \end{bmatrix} \bar{\mathbf{\Psi}}\bar{\alpha} + \begin{bmatrix} \bar{\mathbf{e}} \\ \bar{\mathbf{f}} \end{bmatrix}. \tag{18}$$

As we discussed in Sec. III-A1 that using (12) we can remove those components of $\bar{\alpha}$ from the system that are committed to the output. Following the same procedure, if we want to remove a vector α_{l-1} that belongs to $\bar{\alpha}$ from the system, we modify the system in (18) as

$$\begin{bmatrix} \tilde{\mathbf{y}} \\ \tilde{\mathbf{q}} \end{bmatrix} \stackrel{\text{def}}{=} \begin{bmatrix} \bar{\mathbf{y}} \\ \bar{\mathbf{q}} \end{bmatrix} - \begin{bmatrix} \bar{\mathbf{\Phi}} \\ \bar{\mathbf{F}} \end{bmatrix} \breve{\mathbf{\Psi}}\breve{\alpha}, \tag{19}$$

where $\breve{\alpha}$ denotes $\widehat{\alpha}_{l-1}$ that is the estimate of α_{l-1} and $\breve{\mathbf{\Psi}}$ denotes the columns in $\bar{\mathbf{\Psi}}$ that correspond to the locations of α_{l-1} in $\bar{\alpha}$. We represent the modified form of the system as

$$\begin{bmatrix} \tilde{\mathbf{y}} \\ \tilde{\mathbf{q}} \end{bmatrix} = \begin{bmatrix} \bar{\mathbf{\Phi}} \\ \bar{\mathbf{F}} \end{bmatrix} \tilde{\mathbf{\Psi}}\tilde{\alpha} + \begin{bmatrix} \tilde{\mathbf{e}} \\ \tilde{\mathbf{f}} \end{bmatrix}. \tag{20}$$

2) Recovery problem: To estimate $\tilde{\alpha}$ from the system in (20), we solve the following optimization problem:

$$\underset{\alpha}{\text{minimize}}\ \|\mathbf{W}\alpha\|_1 + \frac{1}{2}\|\bar{\mathbf{\Phi}}\tilde{\mathbf{\Psi}}\alpha - \tilde{\mathbf{y}}\|_2^2 + \frac{\lambda}{2}\|\bar{\mathbf{F}}\tilde{\mathbf{\Psi}}\alpha - \tilde{\mathbf{q}}\|_2^2, \tag{21}$$

where $\mathbf{W}$ is a diagonal matrix that consists of positive weights and $\lambda > 0$ is a regularization parameter that controls the effect of the dynamic model on the solution. We select the weights using a prior estimate of $\tilde{\alpha}$, which we denote as $\widehat{\alpha}$. Estimate of a significant portion of $\widehat{\alpha}$ is known from the previous streaming iteration, and only a small portion is new to the system. We predict the incoming portion of the signal, say x_r, using the prediction matrix in (4) and the signal estimate from the previous iteration

as $\widehat{x}_{r|r-1} \stackrel{\text{def}}{=} F_{r-1}\widehat{x}_{r-1}$. We update the coefficients in $\widehat{\alpha}$ accordingly and set very small coefficients in $\widehat{\alpha}$ to zero. We compute $\mathbf{W}$ according to (16) using $\widehat{\alpha}$. Similarly, instead of solving (21) from scratch, we use $\widehat{\alpha}$ as a warm-start. In the next section we describe a homotopy algorithm for such a warm-start update.

IV. ℓ_1-HOMOTOPY: A UNIFIED HOMOTOPY ALGORITHM

In this section, we present a general homotopy algorithm that we will use to dynamically update the solutions of the ℓ_1 problems, described in (15) and (21), for the recovery of streaming, time-varying signals.

Suppose $\mathbf{y}$ is a vector that obeys the following linear model: $\mathbf{y} = \mathbf{A}\bar{\mathbf{x}} + \mathbf{e}$, where $\bar{\mathbf{x}}$ is a sparse, unknown signal of interest, $\mathbf{A}$ is an $M \times N$ system matrix, and $\mathbf{e}$ is a noise vector. We want to solve the following ℓ_1-norm minimization problem to recover $\bar{\mathbf{x}}$:

$$\underset{\mathbf{x}}{\text{minimize}}\ \|\mathbf{W}\mathbf{x}\|_1 + \frac{1}{2}\|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2^2, \tag{22}$$

where $\mathbf{W}$ is a diagonal matrix that contains positive weights $\mathbf{w}$ on its diagonal. Instead of solving (22) from scratch, we want to expedite the process by using some prior knowledge about the solution of (22). In this regard, we assume that we have a sparse vector, $\widehat{\mathbf{x}}$, with support[1] $\widehat{\Gamma}$ and sign sequence $\widehat{\mathbf{z}}$ that is close to the original solution of (22). The homotopy algorithm we present can be initialized with an arbitrary vector $\widehat{\mathbf{x}}$, given the corresponding matrix $\mathbf{A}_{\widehat{\Gamma}}^T\mathbf{A}_{\widehat{\Gamma}}$ is invertible; however, the update will be quick if $\widehat{\mathbf{x}}$ is close to the final solution.

Homotopy methods provide a general framework to solve an optimization program by continuously transforming it into a related problem for which the solution is either available or easy to compute. Starting from an available solution, a series of simple problems are solved along the so-called *homotopy path* towards the final solution of the original problem [24, 25, 47]. The progression along the homotopy path is controlled by the *homotopy parameter*, which usually varies between 0 and 1, corresponding to the two end points of the homotopy path.

We build the homotopy formulation for (22), using $\epsilon \in [0, 1]$ as the homotopy parameter, as follows. We treat the given warm-start vector $\widehat{\mathbf{x}}$ as a starting point and solve the following optimization problem:

$$\underset{\mathbf{x}}{\text{minimize}}\ \|\mathbf{W}\mathbf{x}\|_1 + \frac{1}{2}\|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2^2 + (1-\epsilon)\mathbf{u}^T\mathbf{x} \tag{23}$$

[1]We use the terms support and active set interchangeably for the index set of nonzero coefficients.

by changing ϵ from 0 to 1. We define $\mathbf{u}$ as

$$\mathbf{u} \stackrel{\text{def}}{=} -\mathbf{W}\,\widehat{\mathbf{z}} - \mathbf{A}^T(\mathbf{A}\widehat{\mathbf{x}} - \mathbf{y}), \tag{24}$$

where $\widehat{\mathbf{z}}$ can be any vector that is defined as $\text{sign}(\widehat{\mathbf{x}})$ on $\widehat{\Gamma}$ and strictly smaller than one elsewhere. Using the definition of $\mathbf{u}$ in (24) and the conditions in (26) below, we can establish that $\widehat{\mathbf{x}}$ is the optimal solution of (23) at $\epsilon = 0$. As ϵ changes from 0 to 1, the optimization problem in (23) gradually transforms into the one in (22), and the solution of (23) follows a piece-wise linear homotopy path from $\widehat{\mathbf{x}}$ toward the solution of (22). To demonstrate these facts and derive the homotopy algorithm, we analyze the optimality conditions for (23) below.

The optimality conditions for (23) can be derived by setting the subdifferential of its objective function to zero [17, 48]. We can describe the conditions that a vector $\mathbf{x}^*$ needs to satisfy to be an optimal solution as

$$\mathbf{W}\mathbf{g} + \mathbf{A}^T(\mathbf{A}\mathbf{x}^* - \mathbf{y}) + (1-\epsilon)\mathbf{u} = 0, \;\; \|\mathbf{g}\|_\infty \leq 1, \;\; \mathbf{g}^T\mathbf{x}^* = \|\mathbf{x}^*\|_1, \tag{25}$$

where $\mathbf{g} = \partial\|\mathbf{x}^*\|_1$ denotes the subdifferential of the ℓ_1 norm of $\mathbf{x}^*$ [49, 50]. This implies that for any given value of $\epsilon \in [0,\ 1]$, the solution $\mathbf{x}^*$ for (23) must satisfy the following optimality conditions:

$$\mathbf{a}_i^T(\mathbf{A}\mathbf{x}^* - \mathbf{y}) + (1-\epsilon)\mathbf{u}_i = -\mathbf{w}_i\mathbf{z}_i \qquad \text{for all } i \in \Gamma \tag{26a}$$

$$|\mathbf{a}_i^T(\mathbf{A}\mathbf{x}^* - \mathbf{y}) + (1-\epsilon)\mathbf{u}_i| \leq \mathbf{w}_i \qquad \text{for all } i \in \Gamma^c, \tag{26b}$$

where $\mathbf{a}_i$ denotes i^{th} column of $\mathbf{A}$, Γ is the support of $\mathbf{x}^*$, and $\mathbf{z}$ is its sign sequence. The optimality conditions in (26) can be viewed as N constraints on $\mathbf{a}_i^T(\mathbf{A}\mathbf{x} - \mathbf{y}) + (1-\epsilon)\mathbf{u}_i$ that the solution $\mathbf{x}^*$ needs to satisfy with equality (in terms of the magnitude and the sign) on the active set Γ and strict inequality (in terms of the magnitude) elsewhere. The only exception is at the critical values of ϵ when the support changes and the constraint on the incoming or outgoing index holds with equality. Equivalently, the locations of the active constraints in (26) determine the support of $\mathbf{x}^*$, Γ, and their signs determine the signs of $\mathbf{x}^*$, $\mathbf{z}$, which in our formulation are opposite to the signs of the active constraints. Note that, the definition of $\mathbf{u}$ in (24) ensures that $\widehat{\mathbf{x}}$ satisfies the optimality conditions in (26) at $\epsilon = 0$; hence, it is a valid initial solution. It is also evident from (26a) that at any value of ϵ the solution $\mathbf{x}^*$ is completely described by the support Γ and the sign sequence $\mathbf{z}$ (assuming that $(\mathbf{A}_\Gamma^T\mathbf{A}_\Gamma)^{-1}$ exists). The support changes only at certain critical values of ϵ, when either a new element enters the support or an existing nonzero element shrinks to zero. These critical values of ϵ are easy to calculate at any point along the

homotopy path, and the entire path (parameterized by ϵ) can be traced in a sequence of computationally inexpensive homotopy steps.

For every homotopy step we jump from one critical value of ϵ to the next while updating the support of the solution, until ϵ is equal to 1. As we increase ϵ by a small value δ, the solution moves in a direction $\partial\mathbf{x}$, which to maintain optimality must obey

$$\mathbf{a}_i^T(\mathbf{A}\mathbf{x}^* - \mathbf{y}) + (1-\epsilon)\mathbf{u}_i + \delta(\mathbf{a}_i^T\mathbf{A}\partial\mathbf{x} - \mathbf{u}_i) = -\mathbf{w}_i\mathbf{z}_i \qquad \text{for all } i \in \Gamma \tag{27a}$$

$$|\underbrace{\mathbf{a}_i^T(\mathbf{A}\mathbf{x}^* - \mathbf{y}) + (1-\epsilon)\mathbf{u}_i}_{\mathbf{p}_i} + \delta\underbrace{(\mathbf{a}_i^T\mathbf{A}\partial\mathbf{x} - \mathbf{u}_i)}_{\mathbf{d}_i}| \leq \mathbf{w}_i \qquad \text{for all } i \in \Gamma^c, \tag{27b}$$

The update direction that keeps the solution optimal as we change δ can be written as

$$\partial\mathbf{x} = \begin{cases} (\mathbf{A}_\Gamma^T\mathbf{A}_\Gamma)^{-1}\mathbf{u}_\Gamma & \text{on } \Gamma \\ 0 & \text{otherwise.} \end{cases} \tag{28}$$

We can move in direction $\partial\mathbf{x}$ until either one of the constraints in (27b) is violated, indicating that we must add an element to the support Γ, or one of the nonzero elements in $\mathbf{x}^*$ shrinks to zero, indicating that we must remove an element from Γ. The smallest step-size that causes one of these changes in the support can be easily computed as $\delta^* = \min(\delta^+, \delta^-)$, where[2]

$$\delta^+ = \min_{i\in\Gamma^c}\left(\frac{\mathbf{w}_i - \mathbf{p}_i}{\mathbf{d}_i}, \frac{-\mathbf{w}_i - \mathbf{p}_i}{\mathbf{d}_i}\right)_+ \tag{29a}$$

and

$$\delta^- = \min_{i\in\Gamma}\left(\frac{-\mathbf{x}_i^*}{\partial\mathbf{x}_i}\right)_+, \tag{29b}$$

and $\min(\cdot)_+$ means that the minimum is taken over only positive arguments. δ^+ is the smallest step-size that causes an inactive constraint to become active at index γ^+, indicating that γ^+ should enter the support and $\mathbf{z}_{\gamma^+}$ should be opposite to the sign of the active constraint at γ^+, and δ^- is the smallest step-size that shrinks an existing element at index γ^- to zero, indicating that γ^- should leave the support. The new critical value of ϵ becomes $\epsilon + \delta^*$ and the new signal estimate $\mathbf{x}^*$ becomes $\mathbf{x}^* + \delta^*\partial\mathbf{x}$, and its support and sign sequence are updated accordingly. If γ^+ is added to the support, at the next iteration we check whether the value of $\partial\mathbf{x}_{\gamma^+}$ has the same sign as $\mathbf{z}_{\gamma^+}$; if the signs mismatch, we immediately remove γ^+ from the support and recompute the update direction $\partial\mathbf{x}$.

[2]To include the positivity constraint in the optimization problem (22), we initialize the homotopy with a non-negative (feasible) warm-start vector and define $\delta^+ = \min_{i\in\Gamma^c}\left(\frac{-\mathbf{w}_i - \mathbf{p}_i}{\mathbf{d}_i}\right)_+$ [51].

Algorithm 1 ℓ_1-HOMOTOPY

Input: $\mathbf{A}$, $\mathbf{y}$, $\mathbf{W}$, $\widehat{\mathbf{x}}$, and $\mathbf{u}$ (optional: inverse or decomposition factors of $\mathbf{A}_{\widehat{\Gamma}}^T\mathbf{A}_{\widehat{\Gamma}}$)
Output: $\mathbf{x}^*$

Initialize: $\epsilon = 0$, $\mathbf{x}^* \leftarrow \widehat{\mathbf{x}}$
Repeat:
  Compute $\partial\mathbf{x}$ in (28) ▷ Update direction
  Compute $\mathbf{p}$ and $\mathbf{d}$ in (27b)
  Compute $\delta^* = \min(\delta^+, \delta^-)$ in (29) ▷ Step size
  if $\epsilon + \delta^* > 1$ **then**
    $\delta^* \leftarrow 1 - \epsilon$ ▷ Last iteration
    $\mathbf{x}^* \leftarrow \mathbf{x}^* + \delta^*\partial\mathbf{x}$ ▷ Final solution
    `break`
  end if
  $\mathbf{x}^* \leftarrow \mathbf{x}^* + \delta^*\partial\mathbf{x}$ ▷ Update the solution
  $\epsilon \leftarrow \epsilon + \delta^*$ ▷ Update the homotopy parameter
  if $\delta^* = \delta^-$ **then**
    $\Gamma \leftarrow \Gamma \backslash \gamma^-$ ▷ Remove an element from the support
  else
    $\Gamma \leftarrow \Gamma \cup \gamma^+$ ▷ Add a new element to the support
  end if
until $\epsilon = 1$

At every step along the homotopy path, we compute the update direction, the step-size, and the consequent one-element change in the support. We repeat this procedure until ϵ is equal to 1. The pseudocode outlining the homotopy procedure is presented in Algorithm 1.

The main computational cost of every homotopy step comes from computing $\partial\mathbf{x}$ by solving an $S \times S$ system of equations in (28) (where S denotes the size of Γ) and from computing the vector $\mathbf{d}$ in (26b) that is used to compute the step-size δ in (29). Since we know the values of $\mathbf{d}$ on Γ by construction and $\partial\mathbf{x}$ is nonzero only on Γ, the cost for computing $\mathbf{d}$ is same as one application of an $M \times N$ matrix. Moreover, since Γ changes by a single element at every homotopy step, instead of solving the linear system in (28) from scratch, we can efficiently compute $\partial\mathbf{x}$ using a rank-one update at every step:

- ▷ *Update matrix inverse:* We can derive a rank-one updating scheme by using matrix inversion lemma to explicitly update the inverse matrix $(\mathbf{A}_\Gamma^T\mathbf{A}_\Gamma)^{-1}$, which has an equivalent cost of performing one matrix-vector product with an $M \times S$ and an $S \times S$ matrix each and adding a rank-one matrix to $(\mathbf{A}_\Gamma^T\mathbf{A}_\Gamma)^{-1}$. The update direction $\partial\mathbf{x}$ can be recursively computed with a vector addition. The total cost for rank-one update is approximately $MS + 2S^2$ flops.

- ▹ *Update matrix factorization:* Updating the inverse of matrix often suffers from numerical stability issues, especially when S becomes closer to M (i.e, the number of columns in $\mathbf{A}_\Gamma$ becomes closer to the number of rows). In general, a more stable approach is to update a Cholesky factorization of $\mathbf{A}_\Gamma^T\mathbf{A}_\Gamma$ (or a QR factorization of $\mathbf{A}_\Gamma$) as the support changes [33, Chapter 12], [52, Chapter 3]. The computational cost for updating Cholesky factors and $\partial\mathbf{x}$ involves nearly $MS + 3S^2$ flops.

As such, the computational cost of a homotopy step is close to the cost of one application of each $\mathbf{A}$ and $\mathbf{A}^T$ (that is, close to $MN + MS + 3S^2 + O(N)$ flops, assuming S elements in the support). If the inverse or factors of $\mathbf{A}_{\widehat{\Gamma}}^T\mathbf{A}_{\widehat{\Gamma}}$ are not readily available during initialization, then updating or computing that would incur an additional one-time cost.

The homotopy method described above is a versatile algorithm that can dynamically update the solution of ℓ_1 problem in (22) for various changes. For instance, adding or removing sequential measurements, updating solution for a time-varying signal, updating the weights, or making arbitrary changes in the system matrix. Almost all these variations appear in the ℓ_1 problems for the recovery of streaming signals described in (15) and (21). Similar to the homotopy formulation in (23), we use the given $\widehat{\alpha}$ as a warm-start vector and solve (15) at every streaming iteration using the following homotopy program:

$$\underset{\alpha}{\text{minimize}}\ \|\mathbf{W}\alpha\|_1 + \frac{1}{2}\|\bar{\mathbf{\Phi}}\tilde{\mathbf{\Psi}}\alpha - \tilde{\mathbf{y}}\|_2^2 + (1-\epsilon)\mathbf{u}^T\alpha, \tag{30}$$

by changing ϵ from 0 to 1. To solve (30), we provide the following parameters to Algorithm 1: the warm-start vector $\widehat{\alpha}$, the system matrix $\mathbf{A} \leftarrow \bar{\mathbf{\Phi}}\tilde{\mathbf{\Psi}}$, and the measurement vector $\mathbf{y} \leftarrow \tilde{\mathbf{y}}$. We define $\mathbf{u}$ as

$$\mathbf{u} \stackrel{\text{def}}{=} -\mathbf{W}\,\widehat{\mathbf{z}} - (\bar{\mathbf{\Phi}}\tilde{\mathbf{\Psi}})^T(\bar{\mathbf{\Phi}}\tilde{\mathbf{\Psi}}\widehat{\alpha} - \tilde{\mathbf{y}}), \tag{31}$$

where $\widehat{\mathbf{z}}$ can be any vector that is defined as $\text{sign}(\widehat{\alpha})$ on the support (nonzero indices) of $\widehat{\alpha}$ and strictly smaller than one elsewhere. Similarly, to solve (21), we use the given warm-start vector $\widehat{\alpha}$ and solve the following homotopy formulation:

$$\underset{\alpha}{\text{minimize}}\ \|\mathbf{W}\alpha\|_1 + \frac{1}{2}\|\bar{\mathbf{\Phi}}\tilde{\mathbf{\Psi}}\alpha - \tilde{\mathbf{y}}\|_2^2 + \frac{\lambda}{2}\|\bar{\mathbf{F}}\tilde{\mathbf{\Psi}}\alpha - \tilde{\mathbf{q}}\|_2^2 + (1-\epsilon)\mathbf{u}^T\alpha, \tag{32}$$

by changing ϵ from 0 to 1, using system matrix $\mathbf{A} \leftarrow [\bar{\mathbf{\Phi}}\tilde{\mathbf{\Psi}}\ ;\ \sqrt{\lambda}\bar{\mathbf{F}}\tilde{\mathbf{\Psi}}]$ and measurement vector $\mathbf{y} \leftarrow [\tilde{\mathbf{y}}\ ;\ \sqrt{\lambda}\tilde{\mathbf{q}}]$ in Algorithm 1. We define $\mathbf{u}$ as

$$\mathbf{u} \stackrel{\text{def}}{=} -\mathbf{W}\,\widehat{\mathbf{z}} - (\bar{\mathbf{\Phi}}\tilde{\mathbf{\Psi}})^T(\bar{\mathbf{\Phi}}\tilde{\mathbf{\Psi}}\widehat{\alpha} - \tilde{\mathbf{y}}) - \lambda\,(\bar{\mathbf{F}}\tilde{\mathbf{\Psi}})^T(\bar{\mathbf{F}}\tilde{\mathbf{\Psi}}\widehat{\alpha} - \tilde{\mathbf{q}}), \tag{33}$$

where $\widehat{\mathbf{z}}$ is defined as before.

V. Numerical experiments

We present experiments for the recovery of two types of time-varying signals from streaming, compressive measurements: 1) signals that have sparse representation in LOT bases and 2) signals that vary according to a linear dynamic model in (4) and have sparse representation in wavelet bases. We demonstrate the performance of our proposed recovery algorithms for these signals at different compression factors. We compare the performance of ℓ_1-homotopy algorithm against two state-of-the-art ℓ_1 solvers and demonstrate that ℓ_1-homotopy requires significantly lesser computation operations and time.

A. Signals with LOT representation

1) Experiment setup: In these experiments, we used the following two discrete-time signals, $x[n]$, from the Wavelab toolbox [53], that have sparse representation in LOT bases: 1) `LinChirp`, which is a critically sampled sinusoidal chirp signal and its frequency increases linearly from zero to one-half of the sampling frequency. 2) `MishMash`, which is a summation of a quadratic and a linear chirp with increasing frequencies and a sinusoidal signal. For both the signals, we generated 2^{15} samples and prepended them with $N = 256$ zeros. Snapshots of `LinChirp` and `MishMash` and their LOT coefficients are presented in Fig. 4a and Fig. 5a, respectively. We estimated sparse LOT coefficients of these signals from streaming, compressive measurements using the system model and the recovery procedure outlined in Sec. III-A.

We selected the parameters for compressive measurements and the signal representation as follows. To simulate streaming, compressive measurements of a given time-varying signal, $x[n]$, at a compression rate R, we followed the model in (1): $y_t = \Phi_t x_t + e_t$. We used non-overlapping x_t of length N to generate a set of $M = N/R$ measurements in y_t. We generated entries in Φ_t independently at random as $\pm 1/\sqrt{M}$ with equal probability. We added Gaussian noise in the measurements by selecting every entry in e_t according to $\mathcal{N}(0, \sigma^2)$ distribution. We selected the variance σ^2 such that the expected SNR with respect to the measurements $\Phi_t x_t$ becomes 35 dB. To represent $x[n]$ using LOT bases, according to (6), we selected the overlapping intervals, I_p, of the same length $N + 2\eta_p = 2N$, where we fixed $\eta_p = N/2$, $a_p = pN + 1/2$, and $l_p = N$ for all $p \in \mathbb{Z}$. We divided $x[n]$ into overlapping intervals, I_p, and computed the corresponding LOT coefficients α_p.

At every streaming iteration, we built the system in (14) for $P = 5$ consecutive x_t in $\bar{\mathbf{x}}$. We updated the system in (9), from the previous iteration, by shifting the active interval, removing old measurements, and adding new measurements. We computed $\tilde{\mathbf{y}}$ in (13), committed a portion of $\widehat{\alpha}$ to the output. The combined system in (14), corresponding to the unknown vector $\bar{\mathbf{x}}$ of length PN, thus, consists of a

measurement vector $\tilde{\mathbf{y}}$ of length PM, a block diagonal $PM \times PN$ measurement matrix $\bar{\mathbf{\Phi}}$, a $PN \times PN$ LOT representation matrix $\tilde{\mathbf{\Psi}}$ in which adjacent pairs of columns overlap in N rows, the unknown LOT coefficient vector $\tilde{\alpha}$ of length PN, and a noise vector $\tilde{\mathbf{e}}$. An example of such a system in depicted in Fig. 3. We predicted the new coefficients in $\widehat{\alpha}$, updated the weights $\mathbf{W}$, and solved (15) using $\widehat{\alpha}$ as a warm-start. We updated the weights according to (16) using $\beta = M\frac{\|\widehat{\alpha}\|_2^2}{\|\widehat{\alpha}\|_1^2}$ and $\tau = \max\{10^{-2}\|\mathbf{A}^T\mathbf{y}\|_\infty, \sigma\sqrt{\log(PN)}\}$, where $\mathbf{A}$ and $\mathbf{y}$ denote the system matrix and the measurement vector in (15), respectively, and σ denotes the standard deviation of the measurement noise. For the first streaming iteration, we initialized α as zero and solved (15) as an iterative reweighted ℓ_1 problem, starting with $\mathbf{W} = \tau$, using five reweighting iterations [16, 32].

We solved (15) using our proposed ℓ_1-homotopy algorithm and two state-of-the-art ℓ_1 solvers: YALL1 [19] and SpaRSA [18], with identical initialization (warm-start) and weight selection procedure. Further description of these algorithms is as follows.

ℓ_1-homotopy[3] : We solved (30) following the procedure outlined in Algorithm 1. The main computational cost at every step of ℓ_1-homotopy involves one matrix-vector multiplication for identifying a change in the support and a rank-one update for computing the update direction. We used the matrix inversion lemma-based scheme to perform the rank-one updates.

YALL1[4]: YALL1 is a first-order algorithm that uses an alternating direction minimization method for solving various ℓ_1 problems, see [19] for further details. We solved (15) using weighted-ℓ_1/ℓ_2 solver in YALL1 package by selecting the initialization vector and weights according to the procedure described in Sec. III-A2. At every streaming iteration, we used previous YALL1 solution to predict the initialization vector and the weights according to (16). We fixed the tolerance parameter to 10^{-4} in all the experiments. The main computational cost of every step in the YALL1 solver comes from applications of $\mathbf{A}$ and $\mathbf{A}^T$.

SpaRSA[5]: SpaRSA is also a first-order method that uses a fast variant of iterative shrinkage and thresholding for solving various ℓ_1-regularized problems, see [18] for further details. Similar to YALL1, we solved (15) using SpaRSA at every streaming iteration by selecting the initialization vector and the weights from the solution of previous iteration. We used the SpaRSA code with default adaptive continuation procedure in the Safeguard mode using the duality gap-based termination criterion for which we fixed

[3] ℓ_1-homotopy code: http://users.ece.gatech.edu/∼sasif/homotopy. Additional experimental results are also available on the same page.

[4] YALL1 code: http://yall1.blogs.rice.edu

[5] SpaRSA code: http://lx.it.pt/∼mtf/SpaRSA

the tolerance parameter to 10^{-4} and modified the code to accommodate weights in the evaluation. The main computational cost for every step in the SpaRSA solver also involves applications of $\mathbf{A}$ and $\mathbf{A}^T$.

To summarize, ℓ_1-homotopy solves homotopy formulation of (15), given in (30), while YALL1 and SpaRSA solve (15) using a warm-start vector for the initialization.

We used MATLAB implementations of all the algorithms and performed all the experiments on a standard laptop computer. We used a single computational thread for all the experiments, which involved recovery of a sparse signal from a given set of streaming measurements using all the candidate algorithms. In every experiment, we recorded three quantities for each algorithm: 1) the quality of reconstructed signal in terms of signal-to-error ratio (SER) in dB, defined as

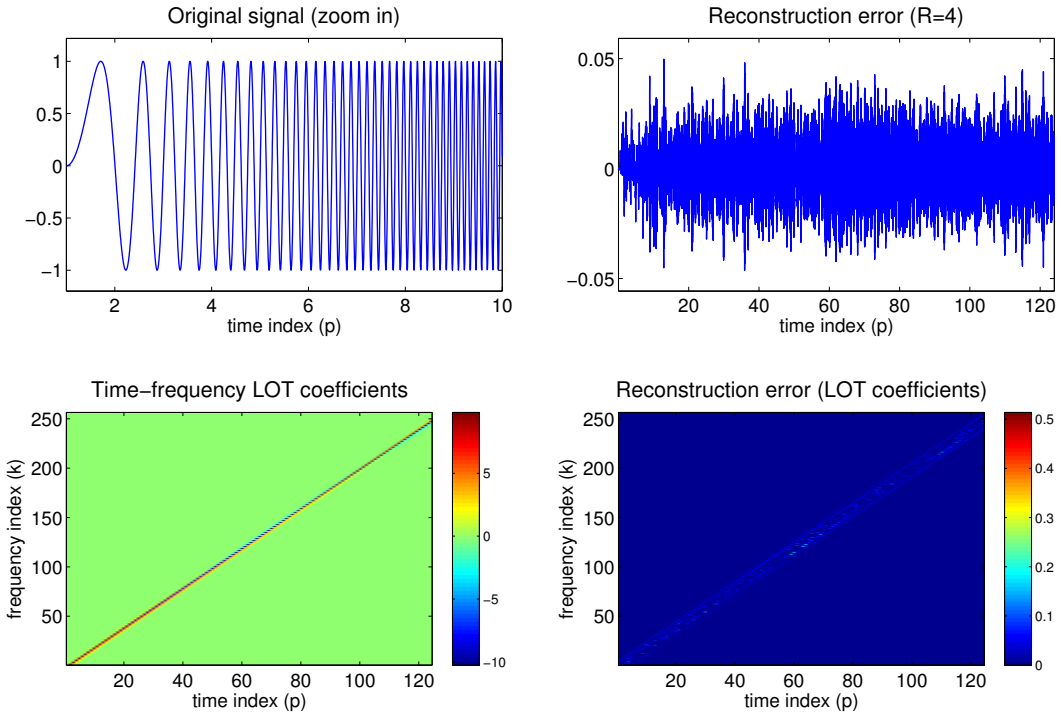
$$\text{SER} = -10 \log_{10} \frac{\|x - \widehat{x}\|_2^2}{\|x\|_2^2},$$

where x and $\widehat{x}$ denote the original and the reconstructed streaming signal, respectively, 2) the number of matrix-vector products with $\mathbf{A}$ and $\mathbf{A}^T$, and 3) the execution time in MATLAB.

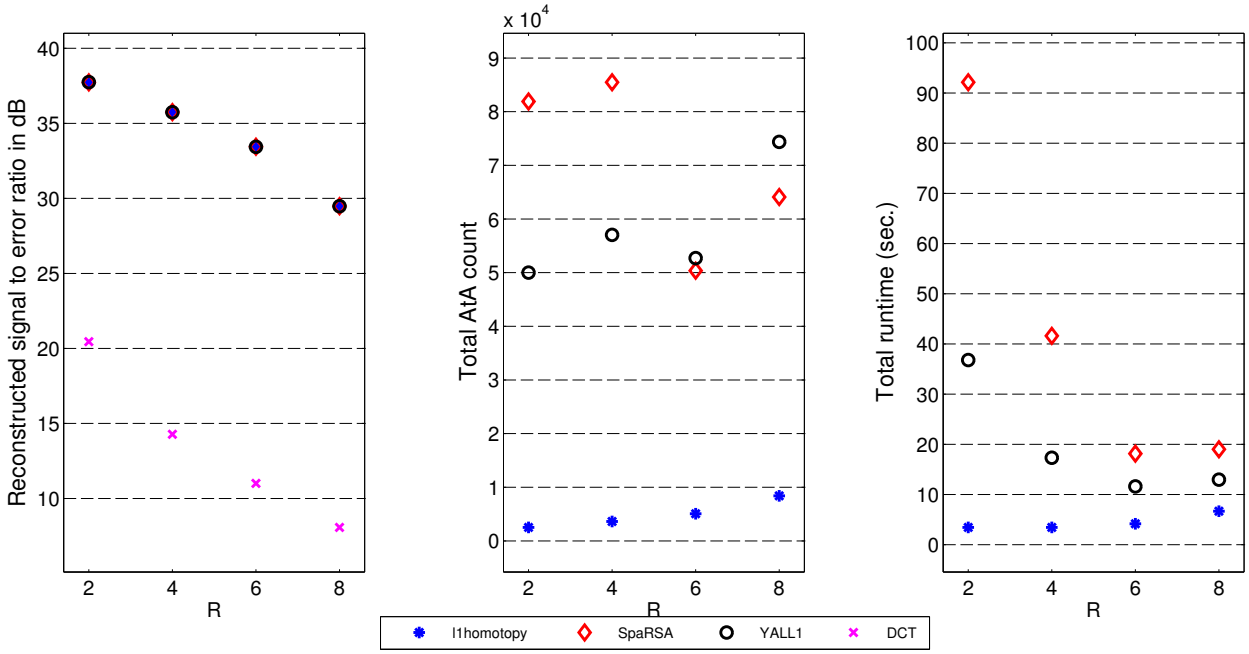
B. Results

We compared performances of ℓ_1-homotopy, YALL1, and SpaRSA for the recovery of `LinChirp` and `MishMash` signals from streaming, compressive measurements. We performed 5 independent trials for the recovery of the streaming signal from random, streaming measurements at different values of compression factor R. The results, averaged over all the trials are presented in Figures 4–5.

Figure 4 presents results for experiments with `LinChirp` signal. Figure 4a presents a snapshot of the `LinChirp` signal, its LOT coefficients, and the reconstruction error at $R = 4$. Three plots in Fig. 4b present results for the three solvers: ℓ_1-homotopy ($*$), SpaRSA ($\diamond$), and YALL1 ($\circ$). The left plot in Fig. 4b compares SER for the three solvers. Since all of them solve the same convex program, SERs for the reconstructed signals are almost identical. To gauge the advantage of LOT-based reconstruction over a block transform-based reconstruction, we repeated the same experiment by replacing the LOT bases with the DCT bases for signal representation (results shown as $\times$). We can see a significant degradation (more than 20 dB loss in SER) in the results for the DCT-based representation as compared to the results for the LOT-based representation. The middle plot in Fig. 4b compares the computational cost of all the algorithms in terms of the total number of matrix-vector multiplications used in the signal reconstruction.

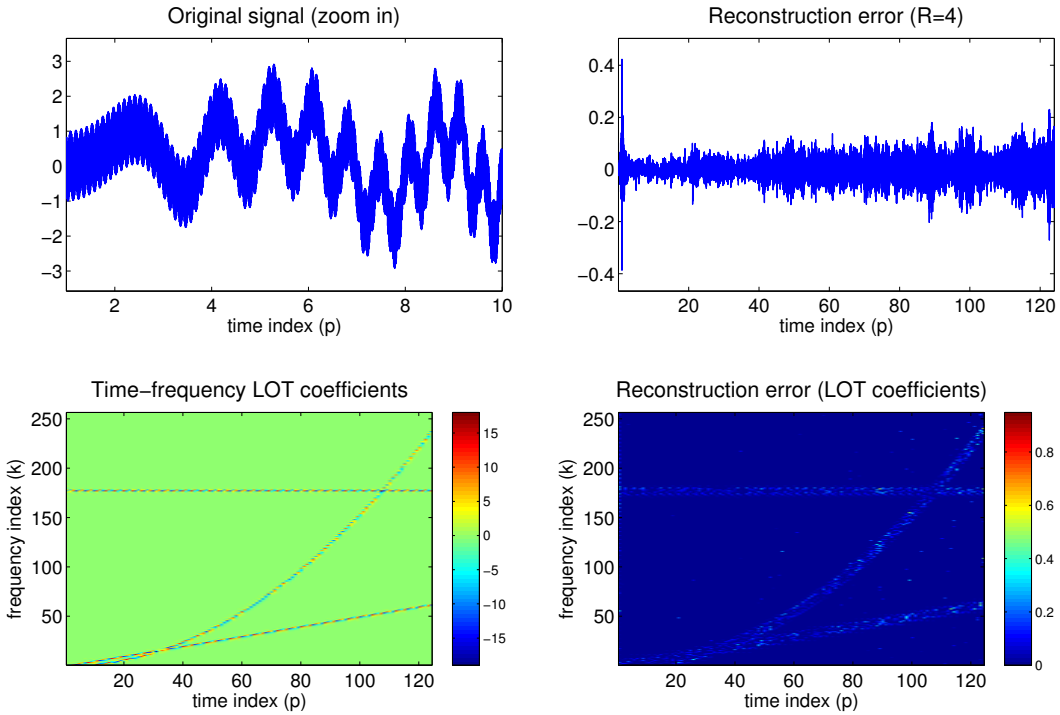


(a) Snapshot of `LinChirp` signal, LOT coefficients, and errors in the reconstruction. **Top left:** Signal $x[n]$ (zoomed in over first 2560 samples). **Bottom left:** LOT coefficients α_p. **Top right:** Error in the reconstructed signal at $R = 4$. **Bottom right:** Error in the reconstructed LOT coefficients

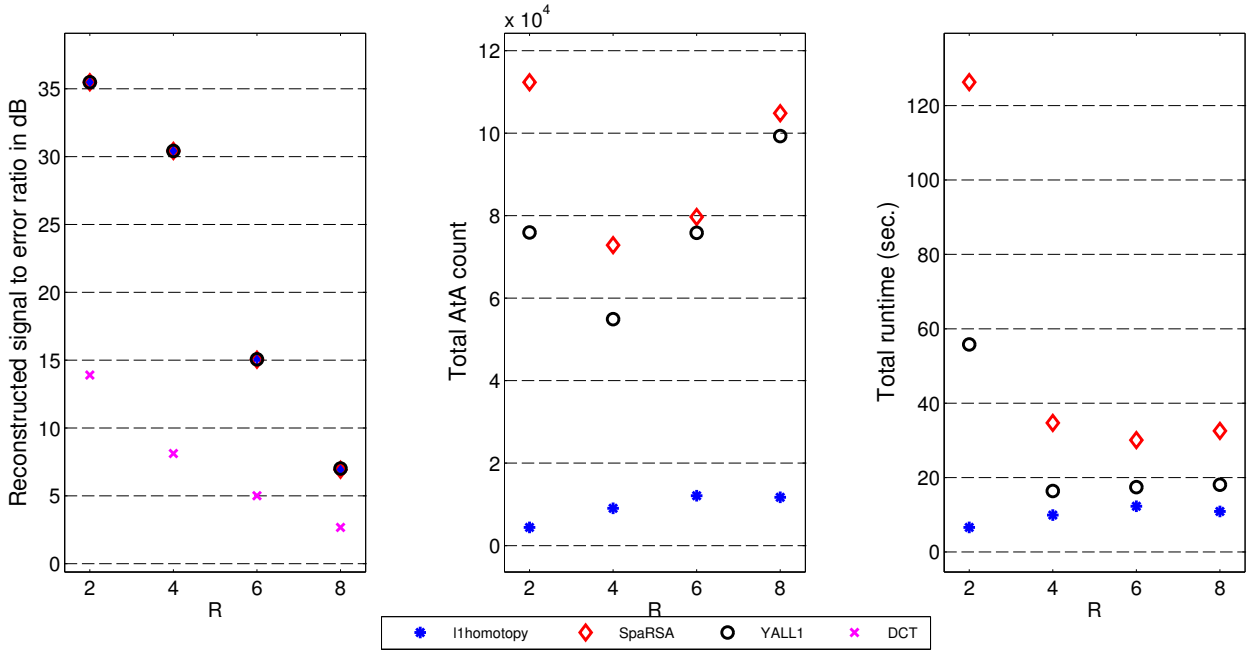


(b) Results for the recovery of `LinChirp` signal from random, compressive measurements in the presence of 35dB noise. **Left:** SER at different R. **Middle:** Approximate count of matrix-vector multiplications. **Right:** Matlab execution time in seconds.

Fig. 4: Experiments on the `LinChirp` signal reconstruction from streaming, compressed measurements using LOT representation.



(a) Snapshot of `MishMash` signal, LOT coefficients, and errors in the reconstruction. **Top left:** Signal $x[n]$ (zoomed in over first 2560 samples). **Bottom left:** LOT coefficients α_p. **Top right:** Error in the reconstructed signal at $R = 4$. **Bottom right:** Error in the reconstructed LOT coefficients



(b) Results for the recovery of `MishMash` signal from random, compressive measurements in the presence of 35dB noise. **Left:** SER at different R. **Middle:** Approximate count of matrix-vector multiplications. **Right:** Matlab execution time in seconds.

Fig. 5: Experiments on the `MishMash` signal reconstruction from streaming, compressed measurements using LOT representation.

We counted an application of each $\mathbf{A}$ and $\mathbf{A}^T$ as one application of $\mathbf{A}^T\mathbf{A}$[6]. We can see that, out of the three solvers, ℓ_1-homotopy required the least number of $\mathbf{A}^T\mathbf{A}$ applications in all the experiments. The right plot in Fig. 4b compares the MATLAB execution time for each solver. We can see that, compared to YALL1 and SpaRSA, ℓ_1-homotopy consumed distinctly lesser time for the reconstruction.

Figures 5 presents similar results for experiments with `MishMash` signal. Figure 5a presents a snapshot of the `MishMash` signal, its LOT coefficients, and the reconstruction error at $R = 4$. Three plots in Fig. 5b compare the performance of the three solvers. In these plots we see similar results that the reconstruction error for (15) using all the solvers is almost identical, but ℓ_1-homotopy performs significantly better in terms of the computational cost and execution time.

A brief summary of the results for our experiments is as follows. We observed that the signals reconstructed using the LOT-based representation had significantly better quality compared to those reconstructed using the DCT-based signal representation. Computational cost and execution time for ℓ_1-homotopy is significantly smaller than that for SpaRSA and YALL1.

C. Linear dynamic model

1) Experiment setup: In these experiments, we simulated time-varying signal $x[n]$ according to the linear dynamic model defined in (4): $x_{t+1} = F_t x_t + f_t$. We generated a seed signal of length $N = 256$, which we will denote as x_0. Starting with x_0, we generated a sequence of signal instances x_t for $t = 1,\ 2, \ldots$ as follows. For each t, we generated x_{t+1} by applying a non-integer, left-circular shift $\epsilon_t \sim \text{uniform}(0.5,\ 1.5)$ to x_t (i.e., $x_{t+1}[n] = x_t[(n + \epsilon_t)_{\text{mod } N}]$, where ϵ_t is drawn uniformly, at random from interval $[0.5,\ 1.5]$). We computed the x_t at non-integer locations using linear interpolation. To define the dynamic model, we assumed that the individual shifts (ϵ_t) are unknown and only their average value is known, which is one in our experiments. Therefore, we defined F_t, for all t, as a matrix that applies left-circular shift of one, whereas f_t accounts for the prediction error in the model because of the unaccounted component of the shift ϵ_t.

We used the following two signals from the Wavelab toolbox [53] as x_0: 1) `HeaviSine`, which is a summation of a sinusoidal and a rectangular signal and 2) `Piece-Regular`, which is a piecewise smooth signal. `HeaviSine` and `Piece-Regular` signals along with examples of their shifted copies are presented in Fig. 6a and Fig. 7a, respectively. We concatenated the x_t for $t = 1,\ 2, \ldots, 128$ to build the time-varying signal $x[n]$ of length 2^{15}. We estimated sparse wavelet coefficients of $x[n]$ from

[6]For the homotopy algorithms, we approximated the cost of one step as one application of $\mathbf{A}^T\mathbf{A}$.

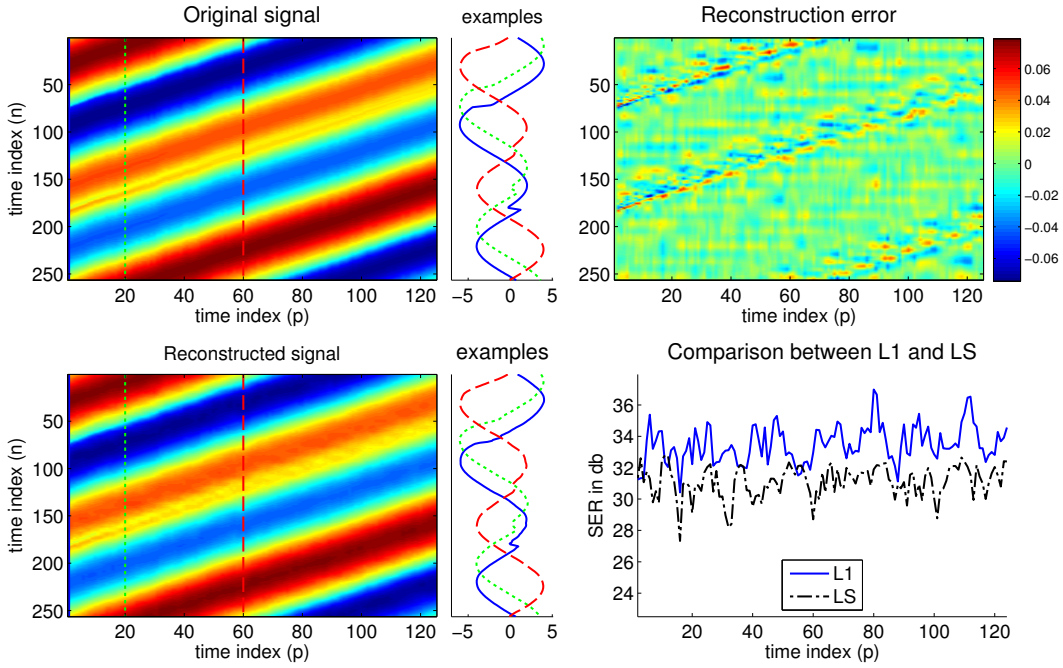
streaming, compressive measurements using the system model and the recovery procedure outlined in Sec. III-B.

We selected the compressive measurements and the signal representation as follows. We simulated streaming, compressive measurements of $x[n]$ according to (1), using the same procedure as described in the previous section. For a desired compression rate R, we generated y_t with $M = N/R$ measurements of non-overlapping x_t, generated entries in Φ_t as $\pm 1/\sqrt{M}$ with equal probability, and added Gaussian noise in the measurements such that the expected SNR becomes $35\,\text{dB}$. To represent $x[n]$ according to the model in (6), we used (block-based) Daubechies-8 orthogonal wavelets [54] with five levels of decomposition. We divided $x[n]$ into consecutive, disjoint components, x_t, of length N and computed wavelet coefficients, α_t, using circular convolution in the wavelet analysis filter bank.

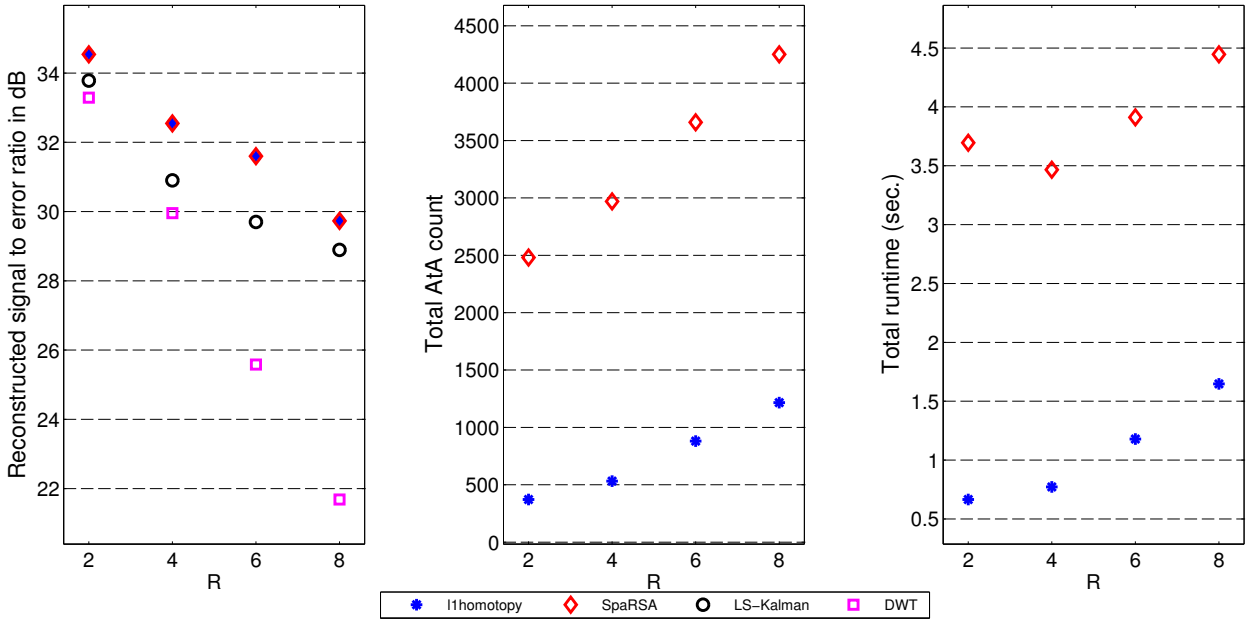
At every streaming iteration, we built the system in (20) for $P = 3$ consecutive x_t in $\bar{\mathbf{x}}$. We updated the system in (18) by shifting the active interval, removing old measurements, and adding new measurements. We computed $\tilde{\mathbf{y}}$ and $\tilde{\mathbf{q}}$ in (19) and committed a portion of $\widehat{\alpha}$ to the output. The combined system in (20), corresponding to the unknown vector $\bar{\mathbf{x}}$ of length PN, thus, consists of measurement vectors $\tilde{\mathbf{y}}, \tilde{\mathbf{q}}$ of length PM and PN, respectively, a block diagonal $PM \times PN$ measurement matrix $\bar{\mathbf{\Phi}}$, a banded $PN \times PN$ prediction matrix $\bar{\mathbf{F}}$, a block-diagonal $PN \times PN$ representation matrix $\tilde{\mathbf{\Psi}}$, the unknown wavelet coefficient vector $\tilde{\alpha}$ of length PN, and error vectors $\tilde{\mathbf{e}}, \tilde{\mathbf{f}}$. We predicted values of the new coefficients in $\widehat{\alpha}$, updated the weights $\mathbf{W}$, and solved (21) using $\widehat{\alpha}$ as a warm-start. We selected $\lambda = 1/2$ and updated the weights according to (16) using $\beta = M\frac{\|\widehat{\alpha}\|_2^2}{\|\widehat{\alpha}\|_1^2}$ and $\tau = \max\{10^{-2}\|\mathbf{A}^T\mathbf{y}\|_\infty, \sigma\sqrt{\log(PN)}\}$, where $\mathbf{A}$ and $\mathbf{y}$ denote the system matrix and the measurements in (21), respectively, and σ denotes the standard deviation of the measurement noise. We truncated the values in $\widehat{\alpha}$ that are smaller than $\tau/\sqrt{\log(PN)}$ to zero. For the first streaming iteration, we initialized $\widehat{x}_{l-1}$ as x_0 and α as zero. We solved (21) as an iterative reweighted ℓ_1 problem, starting with $\mathbf{W} = \tau$, using five reweighting iterations [16, 32].

We solved (21) using our proposed ℓ_1-homotopy algorithm (which in fact solves (32)) and SpaRSA, with identical initialization (warm-start) and weight selection procedure. Since YALL1 only works with under-determined systems, we did not use it in these experiments.

2) Results: We compared the performance of ℓ_1-homotopy and SpaRSA for the recovery of `HeaviSine` and `Piece-Regular` signals from streaming, compressive measurements. We performed 5 independent trials at different values of the compression factor R. In each experiment, we estimated the time-varying signal using all the algorithms, according to the procedures described above, and recorded corresponding



(a) Snapshot of the original and the reconstructed signal, error in the reconstruction, and the comparison of ℓ_1- and ℓ_2-regularized reconstructions. **Top left:** `HeaviSine` signal $x[n]$ drawn as an image; p^{th} column represents x_p; x_1, x_{20} and x_{60} are plotted on the right. **Bottom left:** Reconstructed signal at $R = 4$. **Top right:** Error in the reconstructed signal. **Bottom right:** Comparison between SERs for the solution of the ℓ_1-regularized problem in (21) (solid-blue line, labeled L1) and the solution of the ℓ_2-regularized (Kalman filter smoothing) problem in (34) (broken-black line, labeled LS).



(b) Results for the recovery of `HeaviSine` signal from random, compressive measurements in the presence of 35dB noise. **Left:** SER at different R. **Middle:** Approximate count of matrix-vector multiplications. **Right:** Matlab execution time in seconds.

Fig. 6: Experiments on the time-varying `HeaviSine` signal reconstruction from streaming, compressed measurements when the signal follows a linear dynamic model.

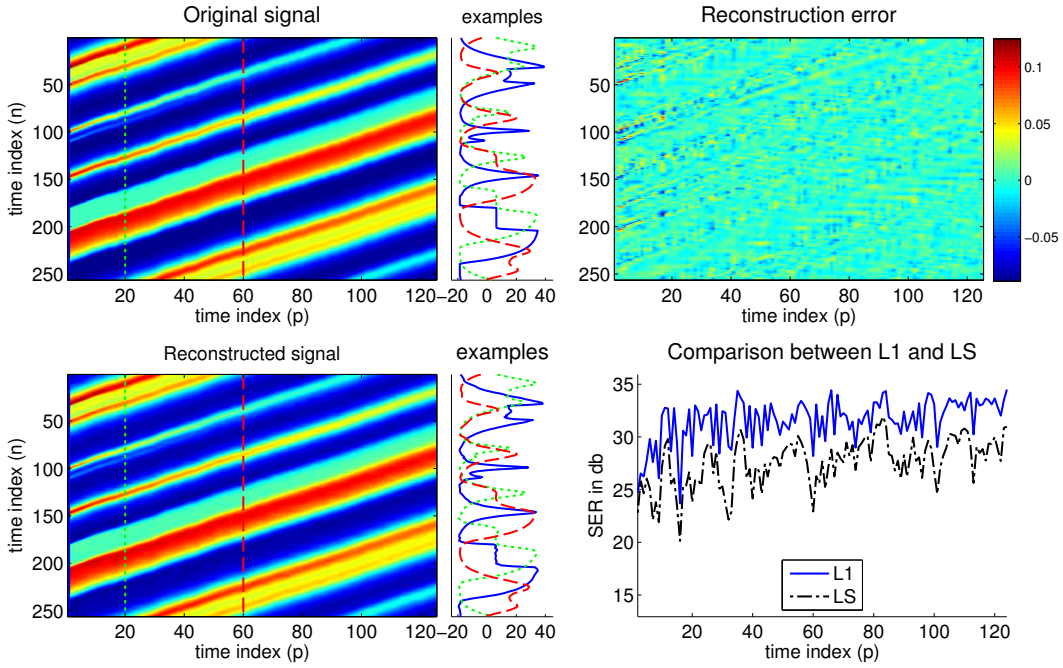
signal-to-error ratio, number of matrix-vector products, and MATLAB runtime. The results, averaged over all the trials, are presented in Figures 6–7.

Figure 6 presents results for experiments with `HeaviSine` signal. Figure 6a, top-left image presents the `HeaviSine` signal, where p^{th} column represents x_p. Next to the image, we have plotted three examples for x_1, x_{20}, x_{60}, as three colored lines. Bottom-left image is the reconstructed signal at $R = 4$ along with the examples of the reconstructed x_p on its right. Top-right image represents errors in the reconstruction. Bottom-right plot presents a comparison between the SER for the solution of the ℓ_1-regularized problem in (21) and the solution of the following ℓ_2-regularized (Kalman filtering and smoothing) problem using the systems in (9) and (17):

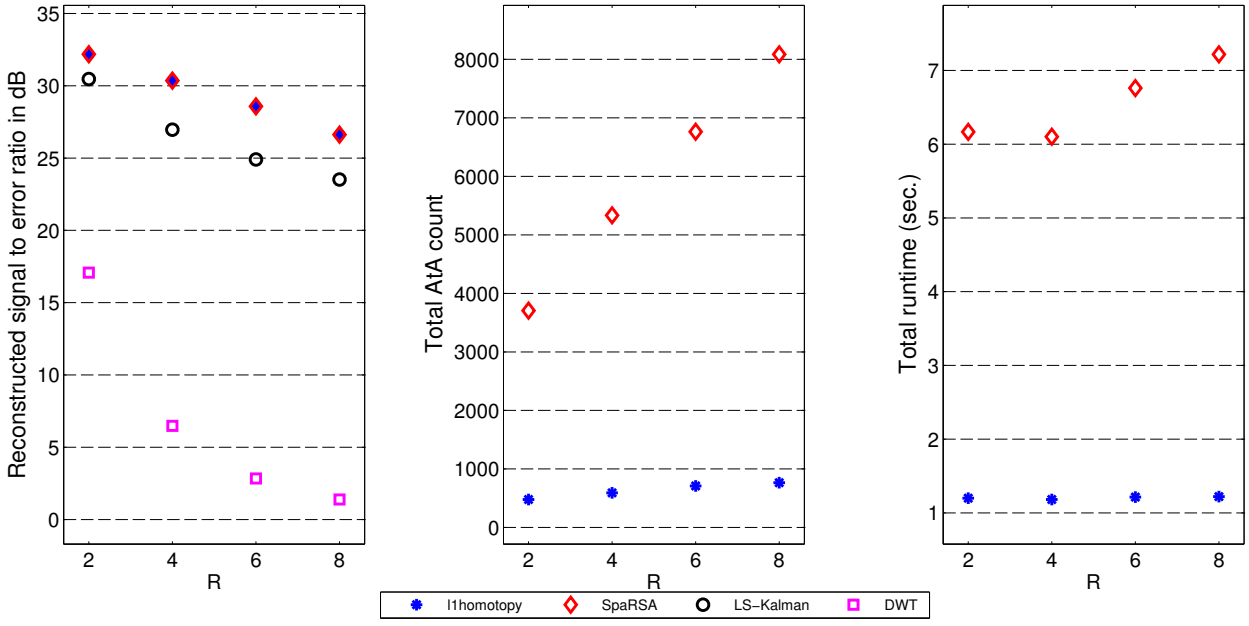
$$\underset{\mathbf{x}}{\text{minimize}}\ (x_p - \widehat{x}_{p|p-1})^T P_{p|p-1}^{-1}(x_p - \widehat{x}_{p|p-1}) + \lambda\|\tilde{\mathbf{F}}\mathbf{x}\|_2^2 + \|\bar{\mathbf{\Phi}}\mathbf{x} - \bar{\mathbf{y}}\|_2^2, \tag{34}$$

where $\mathbf{x}$ denotes a vector that consists of $x_p, \ldots, x_{p+P-1}$, $\tilde{\mathbf{F}}$ denotes a submatrix of $\bar{\mathbf{F}}$ (without its first N rows), and $P_{p|p-1}$ denotes the error covariance matrix for the Kalman filter estimate $\widehat{x}_{p|p-1}$ given all the previous measurements [34, 35]. Three plots in Fig. 6b compare performance of the ℓ_1-homotopy ($*$) and SpaRSA ($\diamond$). The left plot in Fig. 6b compares the SER for the two solvers. Since both of them solve the same convex program, SERs for the reconstructed signals are almost identical. To demonstrate the advantage of our proposed recovery framework (21), we present results for the solution of two related recovery problems, for identical signal representation and measurement settings: 1) Kalman filtering and smoothing problem (34) (labeled as LS-Kalman and plotted as $\circ$), which does not take into account the sparsity of the signal. As the results indicate, the Kalman filter estimate is not as good as the one for the ℓ_1-regularized problem in (21). 2) Weighted ℓ_1-regularized problem in (21) without the dynamic model, which is equivalent to solving (21) with $\lambda = 0$, and it exploits only the sparse representation of each x_p in wavelets (results labeled as DWT and plotted as $\times$). We observed a significant degradation in the signals reconstructed without the dynamic model; the results are indeed inferior to the LS-Kalman. The middle plot in Fig. 6b compares the computational cost of all the algorithms in terms of the total number of matrix-vector multiplications used in the signal reconstruction, and the right plot in Fig. 6b compares the MATLAB execution time for each solver. We observed that ℓ_1-homotopy consumed distinctly fewer matrix-vector multiplications and lesser computation time for the signal reconstruction.

Figures 7 presents similar results for experiments with `Piece-Regular` signal. Figure 7a presents a snapshot of the `Piece-Regular` signal, its reconstruction at $R = 4$ using (21), error in the reconstruction, and comparison between the reconstruction of (21) and (34). Three plots in Fig. 7b compare



(a) Snapshot of the original and the reconstructed signal, error in the reconstruction, and the comparison of ℓ_1- and ℓ_2-regularized reconstructions. **Top left:** `Piece-Regular` signal $x[n]$ drawn as an image; p^{th} column represents x_p; x_1, x_{20} and x_{60} are plotted on the right. **Bottom left:** Reconstructed signal at $R = 4$. **Top right:** Error in the reconstructed signal. **Bottom right:** Comparison between SERs for the solution of the ℓ_1-regularized problem in (21) (solid-blue line, labeled L1) and the solution of the ℓ_2-regularized (Kalman filter smoothing) problem in (34) (broken-black line, labeled LS).



(b) Results for the recovery of `Piece-Regular` signal from random, compressive measurements in the presence of 35dB noise. **Left:** SER at different R. **Middle:** Approximate count of matrix-vector multiplications. **Right:** Matlab execution time in seconds.

Fig. 7: Experiments on the time-varying `Piece-Regular` signal reconstruction from streaming, compressed measurements when the signal follows a linear dynamic model.

performance of the two solvers. In these plots we see similar results that the reconstruction error for (21) using both ℓ_1-homotopy and SpaRSA is almost identical, but ℓ_1-homotopy performs significantly better in terms of computational cost and execution time. For the DWT experiments with `Piece-Regular` signal, we solved a non-weighted version of (21), where we fixed the value of $\mathbf{W}$ as τ.

A brief summary of the results for our experiments is as follows. We observed that combining linear dynamic model with the ℓ_1-norm regularization for sparse signal reconstruction provided much better signal reconstruction compared to the Kalman filter or ℓ_1-regularized problems alone. The computational cost and execution time for ℓ_1-homotopy is significantly smaller than that for SpaRSA. Average number of homotopy steps for updating the solution at every iteration ranges from 3 to 10, and average time for an update ranges from 5 to 13 milliseconds (the results in Fig. 6b–7b are summed over 128 iteration).

REFERENCES

[1] E. Candès, J. Romberg, and T. Tao, "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information," *IEEE Transactions on Information Theory*, vol. 52, no. 2, pp. 489–509, Feb. 2006.

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

[3] E. Candès, "Compressive sampling," *Proceedings of the International Congress of Mathematicians, Madrid, Spain*, vol. 3, pp. 1433–1452, 2006.

[4] R. Baraniuk, V. Cevher, M. Duarte, and C. Hegde, "Model-based compressive sensing," *IEEE Transactions on Information Theory*, vol. 56, no. 4, pp. 1982 –2001, Apr. 2010.

[5] B. Olshausen and D. Field, "Sparse coding with an overcomplete basis set: A strategy employed by V1?" *Vision research*, vol. 37, no. 23, pp. 3311–3325, 1997.

[6] K. Kreutz-Delgado, J. Murray, B. Rao, K. Engan, T. Lee, and T. Sejnowski, "Dictionary learning algorithms for sparse representation," *Neural computation*, vol. 15, no. 2, pp. 349–396, 2003.

[7] M. Lustig, D. Donoho, J. Santos, and J. Pauly, "Compressed Sensing MRI [A look at how CS can improve on current imaging techniques]," *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 72–82, Mar. 2008.

[8] M. S. Lewicki and T. J. Sejnowski, "Coding time-varying signals using sparse, shift-invariant representations," *Advances in neural information processing systems*, pp. 730–736, 1999.

[9] W. Li and J. Preisig, "Estimation of rapidly time-varying sparse channels," *IEEE Journal of Oceanic Engineering*, vol. 32, no. 4, pp. 927–939, 2007.

[10] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society, Series B*, vol. 58, no. 1, pp. 267–288, 1996.

[11] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM Journal on Scientific Computing*, vol. 20, no. 1, pp. 33–61, 1999.

[12] E. Candès and T. Tao, "The Dantzig selector: Statistical estimation when p is much larger than n," *Annals of Statistics*, vol. 35, no. 6, pp. 2313–2351, 2007.

[13] M. S. Asif, D. Reddy, P. T. Boufounos, and A. Veeraraghavan, "Streaming compressive sensing for high-speed periodic videos," in *17th IEEE International Conference on Image Processing (ICIP)*, 2010, pp. 3373–3376.

[14] P. T. Boufounos and M. S. Asif, "Compressive sampling for streaming signals with sparse frequency content," in *44th Annual Conference on Information Sciences and Systems (CISS)*, Mar. 2010, pp. 1–6.

[15] H. Zou, "The adaptive lasso and its oracle properties," *Journal of the American Statistical Association*, vol. 101, no. 476, pp. 1418–1429, 2006.

[16] E. J. Candès, M. B. Wakin, and S. P. Boyd, "Enhancing sparsity by reweighted ℓ_1 minimization," *Journal of Fourier Analysis and Applications*, vol. 14, no. 5-6, pp. 877–905, 2008.

[17] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, March 2004.

[18] S. Wright, R. Nowak, and M. Figueiredo, "Sparse reconstruction by separable approximation," *IEEE Transactions on Signal Processing*, vol. 57, no. 7, pp. 2479–2493, July 2009.

[19] J. Yang and Y. Zhang, "Alternating direction algorithms for ℓ_1-problems in compressive sensing," *SIAM Journal on Scientific Computing*, vol. 33, no. 1-2, pp. 250–278, 2011.

[20] S. Becker, J. Bobin, and E. Candès., "NESTA: A fast and accurate first-order method for sparse recovery," *SIAM Journal on Imaging Sciences*, vol. 4, no. 1, pp. 1–39, 2011.

[21] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM Journal on Imaging Sciences*, vol. 2, no. 1, pp. 183–202, 2009.

[22] H. Malvar and D. Staelin, "The LOT: Transform coding without blocking effects," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 37, no. 4, pp. 553–559, 1989.

[23] S. Mallat, *A Wavelet Tour of Signal Processing*, 2nd ed. Academic Press, 1999.

[24] M. Osborne, B. Presnell, and B. Turlach, "A new approach to variable selection in least squares problems," *IMA Journal of Numerical Analysis*, vol. 20, no. 3, pp. 389–403, 2000.

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

[26] P. J. Garrigues and L. E. Ghaoui, "An homotopy algorithm for the Lasso with online observations," *Neural Information Processing Systems (NIPS) 21*, Dec. 2008.

[27] M. S. Asif and J. Romberg, "Streaming measurements in compressive sensing: ℓ_1 filtering," in *42nd Asilomar conference on Signals, Systems and Computers*, Oct. 2008, pp. 1051–1058.

[28] ——, "Dynamic updating for sparse time varying signals," in *43rd Annual Conference on Information Sciences and Systems (CISS)*, Mar. 2009, pp. 3–8.

[29] ——, "Dynamic updating for ℓ_1 minimization," *IEEE Journal of Selected Topics in Signal Processing*, vol. 4, no. 2, pp. 421–434, Apr. 2010.

[30] ——, "Sparse signal recovery and dynamic update of the underdetermined system," in *44th Asilomar Conference on Signals, Systems and Computers*, Nov. 2010, pp. 798–802.

[31] T.-J. Yang, Y.-M. Tsai, C.-T. Li, and L.-G. Chen, "WarmL1: A warm-start homotopy-based reconstruction algorithm for sparse signals," in *IEEE International Symposium on Information Theory Proceedings (ISIT)*, July 2012, pp. 2226–2230.

[32] M. S. Asif and J. Romberg, "Fast and accurate algorithms for re-weighted ℓ_1-norm minimization," *IEEE Transactions on Signal Processing*, Submitted 2012, [Preprint] Available: http://arxiv.org/abs/1208.0651.

[33] G. Golub and C. Van Loan, *Matrix Computations*. Johns Hopkins University Press, 1996.

[34] R. Kalman, "A new approach to linear filtering and prediction problems," *Journal of basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[35] H. W. Sorenson, "Least-squares estimation: from Gauss to Kalman," *IEEE Spectrum*, vol. 7, no. 7, pp. 63–68, 1970.

[36] T. Kailath, A. H. Sayed, and B. Hassibi, *Linear estimation*. Prentice Hall Upper Saddle River, NJ, 2000.

[37] P. T. Boufounos and M. S. Asif, "Compressive sensing for streaming signals using the streaming greedy pursuit," in *MILCOM*, Nov. 2010, pp. 1205–1210.

[38] D. Needell and J. Tropp, "CoSaMP: Iterative signal recovery from incomplete and inaccurate samples." *Applied and Computational Harmonic Analysis*, vol. 26, pp. 301–321, June 2008.

[39] N. Vaswani, "Kalman filtered compressed sensing," in *15th IEEE International Conference on Image Processing*, Oct. 2008, pp. 893–896.

[40] A. Carmi, P. Gurfil, and D. Kanevsky, "Methods for sparse signal recovery using Kalman filtering pseudo-measuremennt norms and quasi-norms," *IEEE Transactions on Signal Processing*, vol. 58, no. 4, pp. 2405–2409, Apr. 2010.

[41] D. Angelosante, S. I. Roumeliotis, and G. B. Giannakis, "Lasso-Kalman smoother for tracking sparse signals," in *Proc. 43rd Asilomar Conference on Signals, Systems and Computers*, Nov. 2009, pp. 181–185.

[42] A. Charles, M. S. Asif, J. Romberg, and C. Rozell, "Sparsity penalties in dynamical system estimation," in *Proc. Conference on Information and System Sciences (CISS)*, Mar. 2011, pp. 1–6.

[43] M. S. Asif, A. Charles, J. Romberg, and C. Rozell, "Estimation and dynamic updating of time-varying signals with sparse variations," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2011, pp. 3908–3911.

[44] J. Ziniel, L. C. Potter, and P. Schniter, "Tracking and smoothing of time-varying sparse signals via approximate belief propagation," in *44th Asilomar Conference on Signals, Systems and Computers*, Nov. 2010, pp. 808–812.

[45] M. S. Asif, L. Hamilton, M. Brummer, and J. Romberg, "Motion-adaptive spatio-temporal regularization for accelerated dynamic MRI," *Magnetic Resonance in Medicine*, 2012.

[46] M. Vetterli and J. Kovacevic, *Wavelets and subband coding*. Prentice Hall PTR Englewood Cliffs, NJ, 1995.

[47] R. Vanderbei, *Linear Programming: Foundations and Extensions*. Kluwer Academic Publishers, 2001.

[48] D. Bertsekas, *Nonlinear programming*. Athena Scientific Belmont, Mass, 1999.

[49] R. T. Rockafellar, *Convex analysis*. Princeton university press, 1997, vol. 28.

[50] D. L. Donoho and Y. Tsaig, "Fast solution of ℓ_1-norm minimization problems when the solution may be sparse," *IEEE Transactions on Information Theory*, vol. 54, no. 11, pp. 4789–4812, 2008.

[51] M. S. Asif, "Dynamic compressive sensing: Sparse recovery algorithms for streaming signals and video," Doctoral Thesis, Georgia Institute of Technology, 2013.

[52] Å. Björck, *Numerical Methods for Least Squares Problems*. Society for Industrial and Applied Mathematics (SIAM), 1996.

[53] J. Buckheit, S. Chen, D. Donoho, and I. Johnstone, "Wavelab 850, Software toolbox," http://www-stat.stanford.edu/∼wavelab/.

[54] I. Daubechies, *Ten Lectures on Wavelets*. Society for Industrial and Applied Mathematics (SIAM), 1992, vol. 61.