DIGITAL ACCESS TO SCHOLARSHIP AT HARVARD
DASH.HARVARD.EDU

HARVARD LIBRARY
Office for Scholarly Communication

SPARLS: The Sparse RLS Algorithm

Citation
Babadi, Behtash, Nicholas Kalouptsidis, and Vahid Tarokh. 2010. SPARLS: The sparse RLS algorithm. IEEE Transactions on Signal Processing 58(8): 4013-4025.

Published Version
doi:10.1109/TSP.2010.2048103

Permanent link
http://nrs.harvard.edu/urn-3:HUL.InstRepos:4687196

Terms of Use
This article was downloaded from Harvard University's DASH repository, and is made available under the terms and conditions applicable to Open Access Policy Articles, as set forth at http://nrs.harvard.edu/urn-3:HUL.InstRepos:dash.current.terms-of-use#OAP

Share Your Story

The Harvard community has made this article openly available.
Please share how this access benefits you. Submit a story .

Accessibility

SPARLS: The Sparse RLS Algorithm

Behtash Babadi, Nicholas Kalouptsidis and Vahid Tarokh

***Abstract*—We develop a Recursive $\mathcal{L}_1$-Regularized Least Squares (SPARLS) algorithm for the estimation of a sparse tap-weight vector in the adaptive filtering setting. The SPARLS algorithm exploits noisy observations of the tap-weight vector output stream and produces its estimate using an Expectation-Maximization type algorithm. We prove the convergence of the SPARLS algorithm to a near-optimal estimate in a stationary environment and present analytical results for the steady state error. Simulation studies in the context of channel estimation, employing multi-path wireless channels, show that the SPARLS algorithm has significant improvement over the conventional widely-used Recursive Least Squares (RLS) algorithm in terms of mean squared error (MSE). Moreover, these simulation studies suggest that the SPARLS algorithm (with slight modifications) can operate with lower computational requirements than the RLS algorithm, when applied to tap-weight vectors with fixed support.**

I. Introduction

Adaptive filtering is an important part of statistical signal processing, which is highly appealing in estimation problems based on streaming data in environments with unknown statistics [17]. In particular, it is widely used for echo cancellation in speech processing systems and for equalization or channel estimation in wireless systems.

A wide range of signals of interest admit sparse representations. Furthermore various input output systems are described by sparse models. For example, the multi-path wireless channel has only a few significant components [6]. Other examples include echo components of sound in indoor environments and natural images. However, the conventional adaptive filtering algorithms, such as Least Mean Squares (LMS) and Recursive Least Squares (RLS) algorithms, which are widely used in practice, do not exploit the underlying sparseness in order to improve the estimation process.

There has been a lot of focus on the estimation of sparse signals based on noisy observations among the researchers in the fields of signal processing and information theory (Please see [1], [9], [10], [14], [18], [28] and [30]). Although the above-mentioned works contain fundamental theoretical results, most of the proposed estimation algorithms are not tailored to time varying environments with real time requirements; they suffer from high complexity and are not appropriate for implementation purposes.

Copyright (c) 2010 IEEE. Personal use of this material is permitted. However, permission to use this material for any other purposes must be obtained from the IEEE by sending a request to pubs-permissions@ieee.org.

B. Babadi and V. Tarokh are with the School of Engineering and Applied Sciences, Harvard University, Cambridge, MA, 02138. (e-mails: {behtash , vahid}@seas.harvard.edu)

N. Kalouptsidis is with the Department of Informatics and Telecommunications, National and Kapodistrian University of Athens, Athens, Greece (e-mail: kalou@di.uoa.gr)

This work has been presented in part at the 2009 IEEE Sarnoff Symposium, March 30 - April 1, 2009, Princeton, NJ [4].

Recently, Bajwa et. al [6] used the Dantzig Selector (presented by Candes and Tao [10]) and Least Squares (LS) estimates for the problem of sparse channel sensing. Although the Dantzig Selector and the LS method produce sparse estimates with improved MSE, they do not exploit the sparsity of the underlying signal in order to reduce the computational complexity. Moreover, they are not appropriate for the setting of streaming data.

Chen et. al [11] have also presented a Sparse LMS algorithm for system identification, which takes advantage of the sparsity of the underlying signal in order to improve the MSE performance of the LMS algorithm. This is done by incorporating two different sparsity constraints into the quadratic cost function of the LMS algorithm. However, it does not make use of the sparseness in order to reduce the computational complexity. Moreover, LMS type algorithms suffer from slow convergence and hence poor tracking properties when used for estimation of time-varying signals [17]. In [3], Angelosante et al. introduced an algorithm which recursively retrieves the weighted LASSO estimates using a system of normal equations or via iterative sub-gradient methods.

In this paper, we introduce a Recursive $\mathcal{L}_1$-Regularized Least Squares (SPARLS) algorithm for adaptive filtering setup. The SPARLS algorithm is based on an Expectation-Maximization (EM) type algorithm presented in [15] and produces successive improved estimates based on streaming data. We present analytical results for the convergence and the steady state Mean Squared Error (MSE), which reveal the significant MSE gain of the SPARLS algorithm. Simulation studies show that the SPARLS algorithm significantly outperforms the RLS algorithm in terms of MSE, for both static (with finite samples) and time-varying signals. Moreover, these simulation results suggest that the computational complexity of the SPARLS algorithm (with slight modifications) can be less than that of the RLS algorithm, for tap-weight vectors with fixed support. In particular, for estimating a time-varying Rayleigh fading wireless channel with 5 nonzero coefficients, the SPARLS algorithm gains about 7dB over the RLS algorithm in MSE and has about 80% less computational complexity.

The outline of the paper is as follows: we will present the mathematical preliminaries and problem statement in Section II. We will formally define the SPARLS algorithm in Section III, followed by analytical results regarding convergence, steady state error, error performance comparison between SPARLS and RLS, complexity and storage issues, and parameter adjustments in Section IV. Simulation studies are presented in Section V, followed by conclusion in Section VI.

II. Mathematical Preliminaries and Problem Statement

A. Adaptive Filtering Setup

Consider the conventional adaptive filtering setup, consisting of a transversal filter followed by an adaptation block. The tap-input vector at time i is defined by

$$\mathbf{x}(i) := [x(i), x(i-1), \cdots, x(i-M+1)]^T \tag{1}$$

where $x(k)$ is the input at time k, $k = 1, \cdots, n$. The tap-weight vector at time n is defined by

$$\hat{\mathbf{w}}(n) := [\hat{w}_0(n), \hat{w}_1(n), \cdots, \hat{w}_{M-1}(n)]^T. \tag{2}$$

The output of the filter at time i is given by

$$y(i) := \hat{\mathbf{w}}^*(n)\mathbf{x}(i). \tag{3}$$

where $(\cdot)^*$ denotes the conjugate transpose operator. Let $d(i)$ be the desired output of the filter at time i. We can define the instantaneous error of the filter by

$$e(i) := d(i) - y(i) = d(i) - \hat{\mathbf{w}}^*(n)\mathbf{x}(i). \tag{4}$$

The operation of the adaptation block at time n can therefore be stated as the following optimization problem:

$$\min_{\hat{\mathbf{w}}(n)} \quad f\big(e(1), e(2), \cdots, e(n)\big), \tag{5}$$

where $f \geq 0$ is a certain cost function. In particular, if $d(i)$ is generated by an unknown tap-weight $\mathbf{w}(n)$, *i.e.*, $d(i) = \mathbf{w}^*(n)\mathbf{x}(i)$, with an appropriate choice of f, one can possibly obtain a good approximation to $\mathbf{w}(n)$ by solving the optimization problem given in (5). This is, in general, an estimation problem and is the topic of interest in this paper[1].

As an example, one can define the cost function as follows:

$$f_{RLS}\big(e(1), e(2), \cdots, e(n)\big) := \sum_{i=1}^{n} \lambda^{n-i}|e(i)|^2. \tag{6}$$

with λ a non-negative constant. The parameter λ is commonly referred to as *forgetting factor*. The solution to the optimization problem in Eq. (5) with f_{RLS} gives rise to the well-known Recursive Least Squares (RLS) algorithm (See, for example, [17]). The cost function f_{RLS} given in (6) corresponds to a least squares identification problem. Let

$$\mathbf{D}(n) := \mathrm{diag}(\lambda^{n-1}, \lambda^{n-2}, \cdots, 1), \tag{7}$$

$$\mathbf{d}(n) := [d^*(1), d^*(2), \cdots, d^*(n)]^T \tag{8}$$

and $\mathbf{X}(n)$ be an $n \times M$ matrix whose ith row is $\mathbf{x}^*(i)$, *i.e.*,

$$\mathbf{X}(n) := \begin{pmatrix} \mathbf{x}^*(1) \\ \vdots \\ \mathbf{x}^*(n-1) \\ \mathbf{x}^*(n) \end{pmatrix}. \tag{9}$$

The RLS cost function can be written in the following form:

$$\begin{aligned} &f_{RLS}\big(e(1), e(2), \cdots, e(n)\big) \\ &= \big\|\mathbf{D}^{1/2}(n)\mathbf{d}(n) - \mathbf{D}^{1/2}(n)\mathbf{X}(n)\hat{\mathbf{w}}(n)\big\|_2^2 \end{aligned} \tag{10}$$

[1]Our discussion will focus on single channel complex valued signals. The extension to the multi-variable case presents no difficulties.

where $\mathbf{D}^{1/2}(n)$ is a diagonal matrix with entries $D_{ii}^{1/2}(n) := \sqrt{D_{ii}(n)}$.

The canonical form of the problem typically assumes that the input-output sequences are generated by a time varying system with parameters represented by $\mathbf{w}(n)$. In most applications however, stochastic uncertainties are also present. Thus a more pragmatic data generation process is described by the noisy model

$$d(i) = \mathbf{w}^*(n)\mathbf{x}(i) + \eta(i) \tag{11}$$

where $\eta(i)$ is the observation noise. Note that $\mathbf{w}(n)$ reflects the true parameters which may or may not vary with time. The noise will be assumed to be i.i.d. Gaussian, i.e., $\eta(i) \sim \mathcal{N}(0, \sigma^2)$. The estimator has only access to the streaming data $x(i)$ and $d(i)$.

B. Estimation of Sparse Vectors

Let $\mathbf{x}$ be a vector in $\mathbb{C}^M$. We define the $\mathcal{L}_0$ quasi-norm of $\mathbf{x}$ as follows:

$$\|\mathbf{x}\|_0 = |\{x_i | x_i \neq 0\}| \tag{12}$$

A vector $\mathbf{x} \in \mathbb{C}^M$ is called *sparse*, if $\|\mathbf{x}\|_0 \ll M$. A wide range of interesting estimation problems deal with the estimation of sparse vectors. Many signals of interest can naturally be modeled as sparse. For example, the wireless channel usually has a few significant multi-path components. One needs to estimate such signals for various purposes.

Suppose that $\|\mathbf{w}(n)\|_0 = L \ll M$. Also, let $\mathcal{I} := \mathrm{supp}(\mathbf{w}(n))$. Given a matrix $\mathbf{A} \in \mathbb{C}^{N \times M}$ and an index set $\mathcal{J} \subseteq \{1, 2, \cdots, M\}$, we denote the sub-matrix of $\mathbf{A}$ with columns corresponding to the index set $\mathcal{J}$ by $\mathbf{A}_{\mathcal{J}}$. Similarly, we denote the sub-vector of $\mathbf{x} \in \mathbb{C}^M$ corresponding to the index set $\mathcal{J}$ by $\mathbf{x}_{\mathcal{J}}$.

A sparse approximation to $\mathbf{w}(n)$ can be obtained by solving the following optimization problem:

$$\min_{\hat{\mathbf{w}}(n)} \|\hat{\mathbf{w}}(n)\|_0 \quad \text{s.t.} \quad f\big(e(1), e(2), \cdots, e(n)\big) \leq \epsilon \tag{13}$$

where ϵ is a positive constant controlling the cost error in (5). The above optimization problem is computationally intractable. A considerable amount of recent research in statistical signal processing is focused on efficient estimation methods for estimating an unknown sparse vector based on noiseless/noisy observations (Please see [9], [10], [14], [16] and [18]). In particular, convex relaxation techniques provide a viable alternative, whereby the $\mathcal{L}_0$ quasi-norm in (13) is replaced by the convex $\mathcal{L}_1$ norm so that (13) becomes

$$\min_{\hat{\mathbf{w}}(n)} \|\hat{\mathbf{w}}(n)\|_1 \quad \text{s.t.} \quad f\big(e(1), e(2), \cdots, e(n)\big) \leq \epsilon \tag{14}$$

A convex problem results when f is convex, as in the RLS case. Note that we employ the following definition of the $\mathcal{L}_1$ norm on the complex vector space $\mathbb{C}^M$:

$$\|\mathbf{w}\|_1 := \sum_{i=1}^{M} \big(|\Re\{w_i\}| + |\Im\{w_i\}|\big) \tag{15}$$

The Lagrangian formulation shows that if $f = f_{RLS}$, the optimum solution can be equivalently derived from the following optimization problem

$$\min_{\hat{\mathbf{w}}(n)} \frac{1}{2\sigma^2}\left\|\mathbf{D}^{1/2}(n)\mathbf{d}(n)-\mathbf{D}^{1/2}(n)\mathbf{X}(n)\hat{\mathbf{w}}(n)\right\|_2^2+\gamma\|\hat{\mathbf{w}}(n)\|_1. \tag{16}$$

The parameter γ represents a trade off between estimation error and sparsity of the parameter coefficients. Sufficient as well as necessary conditions for the existence and uniqueness of a global minimizer are derived in [28]. These conditions require that the input signal must be properly chosen so that the matrix $\mathbf{D}^{1/2}(n)\mathbf{X}(n)$ is sufficiently incoherent (we will explicitly use some of these results later on in Section IV-B). Suitable probing signals for exact recovery in a multi-path environment are analyzed in [6] and [7].

C. Low-Complexity Expectation Maximization Algorithm

The convex program in Eq. (16) can be solved with the conventional convex programming methods. Here, we adopt an efficient solution presented by Figueirado and Nowak [15] in the context of Wavelet-based image restoration, which we will modify to an online and adaptive setting. Consider the noisy observation model:

$$\mathbf{d}(n) = \mathbf{X}(n)\mathbf{w}(n) + \boldsymbol{\eta}(n). \tag{17}$$

where $\boldsymbol{\eta}(n) \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$, with the following cost function

$$\begin{aligned} f_n(\mathbf{w}) &= \frac{1}{2\sigma^2}\left\|\mathbf{D}^{1/2}(n)\mathbf{d}(n) - \mathbf{D}^{1/2}(n)\mathbf{X}(n)\mathbf{w}\right\|_2^2 + \gamma\|\mathbf{w}\|_1 \\ &= \frac{1}{2\sigma^2}\Big(\mathbf{d}(n) - \mathbf{X}(n)\mathbf{w}\Big)^*\mathbf{D}(n)\Big(\mathbf{d}(n) - \mathbf{X}(n)\mathbf{w}\Big) \\ &\quad + \gamma\|\mathbf{w}\|_1 \end{aligned} \tag{18}$$

If we consider the alternative observation model:

$$\mathbf{d}(n) = \mathbf{X}(n)\mathbf{w}(n) + \boldsymbol{\xi}(n). \tag{19}$$

with $\boldsymbol{\xi}(n) \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{D}^{-1}(n))$, the convex program in Eq. (16) can be identified as the following penalized Maximum Likelihood (ML) problem:

$$\max_{\mathbf{w}(n)} \Big\{ \log p(\mathbf{d}(n)|\mathbf{w}(n)) - \gamma\|\mathbf{w}(n)\|_1 \Big\} \tag{20}$$

where $p(\mathbf{d}(n)|\mathbf{w}(n)) := \mathcal{N}(\mathbf{X}(n)\mathbf{w}(n), \sigma^2\mathbf{D}^{-1}(n))$. This ML problem is in general hard to solve. The clever idea of [15] is to decompose the noise vector $\boldsymbol{\xi}(n)$ in order to divide the optimization problem into a denoising and a filtering problem. We adopt the same method with appropriate modifications for the cost function given in Eq. (20). Consider the following decomposition for $\boldsymbol{\xi}(n)$:

$$\boldsymbol{\xi}(n) = \alpha\mathbf{X}(n)\boldsymbol{\xi}_1(n) + \mathbf{D}^{-1/2}(n)\boldsymbol{\xi}_2(n) \tag{21}$$

where $\boldsymbol{\xi}_1(n) \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and $\boldsymbol{\xi}_2(n) \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I} - \alpha^2\mathbf{D}^{1/2}(n)\mathbf{X}(n)\mathbf{X}^*(n)\mathbf{D}^{1/2}(n))$. We need to choose $\alpha^2 \leq \sigma^2/s_1$, where s_1 is the largest eigenvalue of $\mathbf{D}^{1/2}(n)\mathbf{X}(n)\mathbf{X}^*(n)\mathbf{D}^{1/2}(n)$, in order for $\boldsymbol{\xi}_2(n)$ to have a positive semi-definite covariance matrix (we will talk about how to choose the parameter α in practice in Section IV-E). We can therefore rewrite the model in Eq. (19) as

$$\begin{cases} \mathbf{v}(n) = \mathbf{w}(n) + \alpha\boldsymbol{\xi}_1(n) \\ \mathbf{d}(n) = \mathbf{X}(n)\mathbf{v}(n) + \mathbf{D}^{-1/2}(n)\boldsymbol{\xi}_2(n) \end{cases} \tag{22}$$

The Expectation Maximization (EM) algorithm can be used to solve the penalized ML problem of (20), with the help of the following alternative penalized ML problem

$$\max_{\mathbf{w}(n)} \Big\{ \log p(\mathbf{d}(n), \mathbf{v}(n)|\mathbf{w}(n)) - \gamma\|\mathbf{w}(n)\|_1 \Big\}, \tag{23}$$

which is easier to solve, employing $\mathbf{v}(n)$ as the auxiliary variable. The ℓth iteration of the EM algorithm is as follows:

$$\begin{cases} \text{E-step: } Q(\mathbf{w}|\hat{\mathbf{w}}(n)) := -\frac{1}{2\alpha^2}\|\mathbf{r}^{(\ell)} - \mathbf{w}\|_2^2 - \gamma\|\mathbf{w}\|_1, \\ \qquad \text{where } \mathbf{r}^{(\ell)}(n) := \big(\mathbf{I} - \frac{\alpha^2}{\sigma^2}\mathbf{X}^*(n)\mathbf{D}(n)\mathbf{X}(n)\big)\hat{\mathbf{w}}^{(\ell)}(n) \\ \qquad\qquad + \frac{\alpha^2}{\sigma^2}\mathbf{X}^*(n)\mathbf{D}(n)\mathbf{d}(n) \\ \\ \text{M-step: } \hat{\mathbf{w}}^{(\ell+1)}(n) := \arg\max_{\mathbf{w}} Q(\mathbf{w}|\hat{\mathbf{w}}(n)) = \mathcal{S}(\mathbf{r}^{(\ell)}) \end{cases} \tag{24}$$

where $\mathcal{S}(\cdot) : \mathbb{C}^M \mapsto \mathbb{C}^M$ is the element-wise *soft thresholding* function defined as

$$\begin{aligned} \big(\mathcal{S}(\mathbf{w})\big)_i &:= \operatorname{sgn}\big(\Re\{w_i\}\big)\big(|\Re\{w_i\}| - \gamma\alpha^2\big)_+ \\ &\quad + i\operatorname{sgn}\big(\Im\{w_i\}\big)\big(|\Im\{w_i\}| - \gamma\alpha^2\big)_+ \end{aligned} \tag{25}$$

for all $i = 1, 2, \cdots, M$ and the $(\cdot)_+$ operator is defined as $(x)_+ := \max(x, 0)$.

Note that the above algorithm belongs to a class of pursuit algorithms denoted by iterated shrinkage algorithms (See [8] for a detailed discussion). It is known that the EM algorithm converges to a local maximum (See for example, [13], [23] and [29]). Moreover, under the hypothesis of $\mathbf{X}_{\mathcal{I}}(n)$ being maximal rank, the maximizer is unique and therefore the EM algorithm converges to the unique maximizer of the cost function [28]. The latter hypothesis can be satisfied by appropriately designing the input sequence $x(n)$. For example, a Gaussian i.i.d. input sequence $x(n)$ (as well as the designs given in [6] and [7]) will guarantee this property with high probability.

The soft thresholding function is plotted in Fig. 1. Note that the soft thresholding function tends to decrease the support of the estimate $\hat{\mathbf{w}}(n)$, since it shrinks the support to those elements whose absolute value is greater than $\gamma\alpha^2$. We can use this observation to express the double iteration given in Eq. (24) in a low complexity fashion. Note that the M-step applies soft thresholding independently on the real and imaginary parts of the vector $\mathbf{r}^{(\ell)}(n)$. In order to simplify the notation in

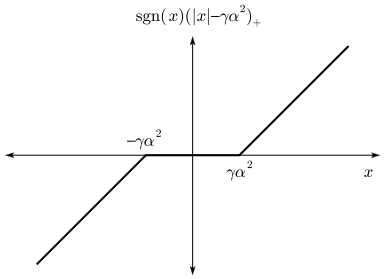

Fig. 1. Soft thresholding function

what follows, we present the low complexity implementation of the EM algorithm for $\mathbf{r}^{(\ell)}(n) \in \mathbb{R}^M$. Generalization to $\mathbf{r}^{(\ell)}(n) \in \mathbb{C}^M$ is straightforward, since the low complexity implementation can be applied to the real and imaginary parts of $\mathbf{r}^{(\ell)}(n)$ independently.

Let $\mathcal{I}^{(\ell)}$ be the support of $\mathbf{r}^{(\ell)}(n)$ at the ℓth iteration. Let

$$\begin{cases} \mathcal{I}_+^{(\ell)} := \{i : r_i^{(\ell)}(n) > \gamma\alpha^2\} \subseteq \mathcal{I}^{(\ell)} \\ \mathcal{I}_-^{(\ell)} := \{i : r_i^{(\ell)}(n) < -\gamma\alpha^2\} \subseteq \mathcal{I}^{(\ell)} \end{cases}, \tag{26}$$

$$\mathbf{B}(n) := \mathbf{I} - \frac{\alpha^2}{\sigma^2}\mathbf{X}^*(n)\mathbf{D}(n)\mathbf{X}(n), \tag{27}$$

and

$$\mathbf{u}(n) := \frac{\alpha^2}{\sigma^2}\mathbf{X}^*(n)\mathbf{D}(n)\mathbf{d}(n). \tag{28}$$

Note that the second iteration in Eq. (24) can be written as

$$\hat{w}_i^{(\ell+1)}(n) = \begin{cases} r_i^{(\ell)}(n) - \gamma\alpha^2 & i \in \mathcal{I}_+^{(\ell)} \\ r_i^{(\ell)}(n) + \gamma\alpha^2 & i \in \mathcal{I}_-^{(\ell)} \\ 0 & i \notin \mathcal{I}_+^{(\ell)} \cup \mathcal{I}_-^{(\ell)} \end{cases} \tag{29}$$

for $i = 1, 2, \cdots, M$. We then have

$$\begin{aligned} \mathbf{B}(n)\hat{\mathbf{w}}^{(\ell+1)}(n) = {} & \mathbf{B}_{\mathcal{I}_+^{(\ell)}}(n)\big(\mathbf{r}^{(\ell)}_{\mathcal{I}_+^{(\ell)}}(n) - \gamma\alpha^2\mathbf{1}_{\mathcal{I}_+^{(\ell)}}\big) \\ & + \mathbf{B}_{\mathcal{I}_-^{(\ell)}}(n)\big(\mathbf{r}^{(\ell)}_{\mathcal{I}_-^{(\ell)}}(n) + \gamma\alpha^2\mathbf{1}_{\mathcal{I}_-^{(\ell)}}\big) \end{aligned} \tag{30}$$

which allows us to express the EM iteration as follows:

$$\begin{cases} \mathbf{r}^{(\ell+1)}(n) = \mathbf{B}_{\mathcal{I}_+^{(\ell)}}(n)\big(\mathbf{r}^{(\ell)}_{\mathcal{I}_+^{(\ell)}}(n) - \gamma\alpha^2\mathbf{1}_{\mathcal{I}_+^{(\ell)}}\big) \\ \qquad\qquad + \mathbf{B}_{\mathcal{I}_-^{(\ell)}}(n)\big(\mathbf{r}^{(\ell)}_{\mathcal{I}_-^{(\ell)}}(n) + \gamma\alpha^2\mathbf{1}_{\mathcal{I}_-^{(\ell)}}\big) + \mathbf{u}(n) \\ \mathcal{I}_+^{(\ell+1)} = \{i : r_i^{(\ell+1)}(n) > \gamma\alpha^2\} \\ \mathcal{I}_-^{(\ell+1)} = \{i : r_i^{(\ell+1)}(n) < -\gamma\alpha^2\} \end{cases} \tag{31}$$

This new set of iteration has a lower computational complexity, since it restricts the matrix multiplications to the instantaneous support of the estimate $\mathbf{r}^{(\ell)}(n)$, which is expected to be close to the support of $\mathbf{w}(n)$ [28]. We denote the iterations given in Eq. (31) by Low-Complexity Expectation Maximization (LCEM) algorithm.

III. The SPARLS Algorithm

A. The Main Algorithm

Upon the arrival of the nth input, $\mathbf{B}(n)$ and $\mathbf{u}(n)$ can be obtained via the following rank-one update rules:

$$\begin{cases} \mathbf{B}(n) = \lambda\mathbf{B}(n-1) - \frac{\alpha^2}{\sigma^2}\mathbf{x}(n)\mathbf{x}^*(n) + (1-\lambda)\mathbf{I} \\ \mathbf{u}(n) = \lambda\mathbf{u}(n-1) + \frac{\alpha^2}{\sigma^2}d^*(n)\mathbf{x}(n) \end{cases} \tag{32}$$

Upon the arrival of the nth input, $x(n)$, the LCEM algorithm computes the estimate $\hat{\mathbf{w}}(n)$ given $\mathbf{B}(n)$, $\mathbf{u}(n)$ and $\mathbf{s}^{(0)}(n)$. The LCEM algorithm is summarized in Algorithm 1. Note that the input argument K denotes the number of EM iterations.

The SPARLS algorithm is formally defined in Algorithm 2. Without loss of generality, we can set the time index $n = 1$ such that $x(1) \neq 0$, in order for the initialization to be well-defined. The schematic realizations of the SPARLS and RLS algorithm are depicted in Fig. 2. Both algorithms perform in an online fashion and update the estimate $\hat{\mathbf{w}}(n)$ upon the arrival of the new data input $x(n)$.

Algorithm 1 LCEM $\big(\mathbf{B}, \mathbf{u}, \hat{\mathbf{w}}, \mathcal{I}_+^{(K-1)} \cup \mathcal{I}_-^{(K-1)}, K\big)$

Inputs: $\mathbf{B}$, $\mathbf{u}$, $\hat{\mathbf{w}}$, $\mathcal{I}_+^{(K-1)} \cup \mathcal{I}_-^{(K-1)}$, and K.
Outputs: $\hat{\mathbf{w}}$, $\mathcal{I}_+^{(K-1)}$ and $\mathcal{I}_-^{(K-1)}$.

$\mathbf{r}^{(0)} = \mathbf{B}_{\mathcal{I}_+^{(K-1)}}\hat{\mathbf{w}}_{\mathcal{I}_+^{(K-1)}} + \mathbf{B}_{\mathcal{I}_-^{(K-1)}}\hat{\mathbf{w}}_{\mathcal{I}_-^{(K-1)}} + \mathbf{u}$.
$\mathcal{I}_+^{(0)} = \{i : r_i^{(0)} > \gamma\alpha^2\}$.
$\mathcal{I}_-^{(0)} = \{i : r_i^{(0)} < -\gamma\alpha^2\}$.
for $\ell = 1, 2, \cdots, K-1$ **do**
  $\mathbf{r}^{(\ell)} = \mathbf{B}_{\mathcal{I}_+^{(\ell-1)}}\big(\mathbf{r}^{(\ell-1)}_{\mathcal{I}_+^{(\ell-1)}} - \gamma\alpha^2\mathbf{1}_{\mathcal{I}_+^{(\ell-1)}}\big) + \mathbf{B}_{\mathcal{I}_-^{(\ell-1)}}\big(\mathbf{r}^{(\ell-1)}_{\mathcal{I}_-^{(\ell-1)}} + \gamma\alpha^2\mathbf{1}_{\mathcal{I}_-^{(\ell-1)}}\big) + \mathbf{u}$.
  $\mathcal{I}_+^{(\ell)} = \{i : r_i^{(\ell)} > \gamma\alpha^2\}$.
  $\mathcal{I}_-^{(\ell)} = \{i : r_i^{(\ell)} < -\gamma\alpha^2\}$.
end for
for $i = 1, 2, \cdots, M$ **do**
  $\hat{w}_i = \begin{cases} r_i^{(K-1)} - \gamma\alpha^2 & i \in \mathcal{I}_+^{(K-1)} \\ r_i^{(K-1)} + \gamma\alpha^2 & i \in \mathcal{I}_-^{(K-1)} \\ 0 & i \notin \mathcal{I}_+^{(K-1)} \cup \mathcal{I}_-^{(K-1)} \end{cases}$.
end for

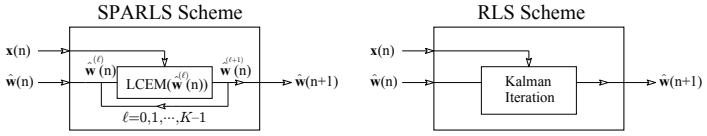

Fig. 2. Schematic realizations of SPARLS and RLS algorithms.

Algorithm 2 SPARLS

Inputs: $\mathbf{B}(1) = \mathbf{I} - \frac{\alpha^2}{\sigma^2}\mathbf{x}(1)\mathbf{x}^*(1)$, $\mathbf{u}(1) = \frac{\alpha^2}{\sigma^2}\mathbf{x}(1)d^*(1)$ and K.
Output: $\hat{\mathbf{w}}(n)$.

for all Input $x(n)$ **do**
  $\mathbf{B}(n) = \lambda\mathbf{B}(n-1) - \frac{\alpha^2}{\sigma^2}\mathbf{x}(n)\mathbf{x}^*(n) + (1-\lambda)\mathbf{I}$.
  $\mathbf{u}(n) = \lambda\mathbf{u}(n-1) + \frac{\alpha^2}{\sigma^2}d^*(n)\mathbf{x}(n)$.
  Run LCEM $\big(\mathbf{B}(n), \mathbf{u}(n), \hat{\mathbf{w}}(n-1), \mathcal{I}_+^{(K-1)}(n-1) \cup \mathcal{I}_-^{(K-1)}(n-1), K\big)$.
  Update $\hat{\mathbf{w}}(n)$.
end for

B. The Low Complexity Update Scheme

The update equation for $\mathbf{B}(n)$ can be implemented in a low complexity fashion. This is due to the fact that the LCEM algorithm only needs the columns of $\mathbf{B}(n)$ corresponding to the index set $\mathcal{I}_+ \cup \mathcal{I}_-$. Thus, given the hypothesis that the subset $\mathcal{I}^{(0)}(n)$ does not vary much with n, *i.e.*, $|\mathcal{I}^{(0)}(n) \backslash \mathcal{I}^{(0)}(n-1)| \ll |\mathcal{I}^{(0)}(n)|$, one can implement the update step for $\mathbf{B}(n)$ in a low complexity fashion as follows.

First, we consider the updating procedure for $\mathbf{B}(n)$ when the new input data $x(n)$ has arrived. Clearly, $\mathcal{I}^{(0)}(n) = \mathcal{I}_+^{(K-1)}(n-1) \cup \mathcal{I}_-^{(K-1)}(n-1)$, if we run the LCEM algorithm a total of K times for each new input $x(n)$. The columns of $\mathbf{B}(n)$ required for the LCEM algorithm clearly correspond to $\mathcal{I}^{(0)}(n)$. We also assign a variable $t_i \in \{1, 2, \cdots, n-1\}$

to each column of $\mathbf{B}(n)$, which denotes the last time index when the ith column of the matrix $\mathbf{B}$ was in the index set $\mathcal{I}^{(0)}$. Upon the arrival of $x(n)$, we only update the columns of $\mathbf{B}(n)$ corresponding to the index set $\mathcal{I}^{(0)}(n)$ and denote the resulting matrix by $\tilde{\mathbf{B}}(n)$:

$$\begin{aligned}\tilde{\mathbf{B}}_i(n) &= \lambda^{n-t_i}\tilde{\mathbf{B}}_i(n-1) \\ &- \frac{\alpha^2}{\sigma^2}\sum_{m=0}^{n-t_i-1}\lambda^m\Big((\mathbf{x}(n-m)\mathbf{x}^*(n-m))_i \\ &\qquad +(1-\lambda)\mathbf{I}_i\Big) \end{aligned} \tag{33}$$

for all $i \in \mathcal{I}^{(0)}(n)$. For example, if the ith column of $\tilde{\mathbf{B}}(n)$ has been last updated at time $n-3$, then $t_i = n-3$, hence the update equation simply becomes:

$$\begin{aligned}\tilde{\mathbf{B}}_i(n) &= \lambda^3\tilde{\mathbf{B}}_i(n-1) \\ &- \frac{\alpha^2}{\sigma^2}\Big(\mathbf{x}(n)\mathbf{x}^*(n)+\lambda\mathbf{x}(n-1)\mathbf{x}^*(n-1) \\ &\qquad +\lambda^2\mathbf{x}(n-2)\mathbf{x}^*(n-2)\Big)_i \\ &+ (1-\lambda)(1+\lambda+\lambda^2)\mathbf{I}_i\end{aligned}$$

Algorithm 3 $\mathrm{LCU}(\tilde{\mathbf{B}}(n-1), \mathcal{J}, \{t_i\}_{i=1}^M)$

Inputs: $\tilde{\mathbf{B}}(n-1)$, $\mathcal{J}$ and $\{t_i\}_{i=1}^M$.
Output: $\mathbf{B}_{\mathcal{J}}$ and $\{t_i\}_{i=1}^M$.

1: **for all** i in $\mathcal{J}$ **do**
2: $\tilde{\mathbf{B}}_i(n) = \lambda\Big\{\tilde{\mathbf{B}}_i(n-1) - \frac{\alpha^2}{\sigma^2}\sum_{m=0}^{n-t_i-1}\lambda^m\Big((\mathbf{x}(n-m)\mathbf{x}^*(n-m))_i + (1-\lambda)\mathbf{I}_i\Big)\Big\}$.
3: $t_i \leftarrow n$.
4: **end for**
5: $\mathbf{B}_{\mathcal{J}} \leftarrow \tilde{\mathbf{B}}_{\mathcal{J}}$

Subsequently, the time indices t_i will be updated as $t_i = n$ for all $i \in \mathcal{I}^{(0)}(n)$ and remain unchanged otherwise. We can formally define the sub-routine Low Complexity Update (LCU) for updating $\mathbf{B}(n)$ as in Algorithm 3. Note that if $\mathcal{I}^{(0)}(n) = \{1, 2, \cdots, M\}$ for all times, then the above update equation for $\tilde{\mathbf{B}}(n)$ is equivalent to the update equation in Eq. (32). But, due to the sparsifying nature of the estimator, the index set $\mathcal{I}^{(0)}(n)$ is expected to be very close to the true index set $\mathcal{I}$. In that case the number of column updates at each time is $\mathcal{I}^{(0)}(n)$. Moreover, these updates are usually very simple in the steady state, since most of the t_is are equal to n, for all $i \in \mathcal{I}^{(0)}(n)$. This is due to the hypothesis that the subset $\mathcal{I}^{(0)}(n)$ does not vary much with n, *i.e.*, $|\mathcal{I}^{(0)}(n) \backslash \mathcal{I}^{(0)}(n-1)| \ll |\mathcal{I}^{(0)}(n)|$. This way, we can exploit the sparseness of the estimate in order to reduce the complexity of the update process for $\mathbf{B}(n)$. Therefore, one can use the LCU subroutine $\mathrm{LCU}(\mathbf{B}(n-1), \mathcal{I}_+^{(K-1)}(n-1) \cup \mathcal{I}_-^{(K-1)}(n-1), \{t_i\}_{i=1}^M)$ on line 2 of the SPARLS algorithm. Similarly, the LCU subroutine can be used in the LCEM algorithm (right before lines 1 and 5), when the algorithm needs to access sub-matrices such as $\mathbf{B}_{\mathcal{I}_+^{(\ell)}}(n)$ or $\mathbf{B}_{\mathcal{I}_-^{(\ell)}}(n)$. Nevertheless, the hypothesis of $|\mathcal{I}^{(0)}(n) \backslash \mathcal{I}^{(0)}(n-1)| \ll |\mathcal{I}^{(0)}(n)|$ may be violated, in which case using the LCU subroutine might result in drawbacks (See Section IV-D for a detailed discussion). Nevertheless, one can always resort to the original form of the SPARLS algorithm.

IV. Analysis of the SPARLS Algorithm

In this section, we will study the convergence of SPARLS to a fixed point in a stationary environment in Section IV-A, the steady state error of the SPARLS in Section IV-B, comparison of the error performance of SPARLS and RLS in a stationary environment for finite sample size, i.e., $n < \infty$ in Section IV-C, the complexity and storage issues of SPARLS (with and without the LCU subroutine) in Section IV-D, and finally, adjusting the parameters of SPARLS in Section IV-E.

A. Convergence Analysis

In order to study the convergence of the SPARLS algorithm, we need to make a number of additional assumptions. First of all, we consider the case of constant unknown vector $\mathbf{w}(n)$, *i.e.*, $\mathbf{w}(n) = \mathbf{w}_0$ for all $n = 1, 2, \cdots$. Moreover, we analyze the convergence in a stationary environment: the input sequence $\{x(n)\}_{n=1}^{\infty}$ and the output sequence $\{d(n)\}_{n=1}^{\infty}$ are realizations of a jointly stationary random process.

Before moving on to the convergence analysis of SPARLS, we briefly overview the convergence properties of the EM algorithm. The global and componentwise convergence of the EM algorithm has been widely studied in the statistics literature (See, for example, [13] and [23]). According to the original paper of Dempster et al. [13], the EM algorithm can be represented by a mapping $\mathcal{M}_n : \mathbb{C}^M \longmapsto \mathbb{C}^M$, defined as

$$\hat{\mathbf{w}}^{(\ell+1)}(n) = \mathcal{M}_n(\hat{\mathbf{w}}^{(\ell)}(n)) \tag{34}$$

where the mapping $\mathcal{M}_n$ is the composition of the E and M steps at time n. Moreover, if the minimizer of the objective function

$$f_n(\mathbf{w}) := \frac{1}{2\sigma^2}\|\mathbf{D}^{1/2}\mathbf{d}(n) - \mathbf{D}^{1/2}\mathbf{X}(n)\mathbf{w}\|_2^2 + \gamma\|\mathbf{w}\|_1 \tag{35}$$

is unique, we have

$$f_n(\mathbf{w}^{(\ell+1)}(n)) < f_n(\mathbf{w}^{(\ell)}(n)). \tag{36}$$

From Lemma 3 of Tropp [28], we know that the minimizer of the objective function given in Eq. (35) is unique if $\mathbf{X}_{\mathcal{I}}(n)$ is maximal rank, where $\mathcal{I} = \mathrm{supp}(\mathbf{w}_0)$. We denote this minimizer by $\tilde{\mathbf{w}}(n)$. The hypothesis of $\mathbf{X}_{\mathcal{I}}(n)$ being maximal rank can be achieved if the input sequence is persistently exciting (In other words, the input must be sufficiently rich to properly excite all modes of the system). For example, if the input sequence $x(n)$ is drawn from an i.i.d. random process, the columns of $\mathbf{X}_{\mathcal{I}}(n)$ form an orthogonal set with probability 1. Hence, we can assume throughout the analysis that the minimizer of the objective function is unique.

The SPARLS algorithm only performs the EM algorithm a finite (K) number of times for each n. Hence, it does not exactly solve the minimization problem in (16). Furthermore, the cost function varies at each step (with n). Hence, it is not trivial that performing the EM algorithm a finite number of times ($K < \infty$) at each step, results in convergence to the unique minimizer of $f_n(\mathbf{x})$, as $n \to \infty$. Indeed, the following

theorem establishes the convergence of the SPARLS algorithm under the above assumptions:

Theorem 4.1 (Convergence): Given a stationary environment and a constant target sparse vector $\mathbf{w}_0$, the SPARLS algorithm (with $K < \infty$) converges almost surely to the unique minimizer of the cost function $f_n(\mathbf{w})$, as $n \to \infty$.

Idea of proof: The idea of proof is to relate the convergence behavior of the EM algorithm along one specific function $f_n(\mathbf{w})$ to the convergence of the SPARLS algorithm across different functions $f_n(\mathbf{w})$. The proof is formally given in Appendix A.

Note that the case of $n \to \infty$ is not of particular interest in our analysis of the stationary scenario, since it defeats the purpose of compressive sampling. However, the convergence proof guarantees that we can get to an arbitrarily small neighborhood of the fixed point (i.e., limit of the unique minimizer of $f_n(\mathbf{w})$) for finite n. This fact will be used later in the performance comparison of SPARLS and RLS (See Theorem 4.2). Next, we study the steady state error of the SPARLS algorithm.

B. Steady State Error Analysis

We define the average instantaneous error of the SPARLS algorithm as follows:

$$\epsilon(n) := \mathbb{E}_{\eta}\Big\{ \|\hat{\mathbf{w}}(n) - \mathbf{w}(n)\|_2 \Big\}. \tag{37}$$

As it is shown in Appendix B, $\epsilon(n)$ obeys the following recurrence relation:

$$\begin{aligned} \epsilon(n+1) &\leq \rho(n)^K \epsilon(n) \\ &+ \mathbb{E}_{\eta}\Big\{ \big\| (\mathbf{D}^{1/2}(n)\mathbf{X}_{\mathcal{I}}(n))^{+} \boldsymbol{\eta}_{\mathcal{I}}(n) \big\|_2 \Big\} \\ &+ \gamma\sigma^2 \Big\| \Big(\mathbf{X}^*_{\mathcal{I}}(n)\mathbf{D}(n)\mathbf{X}_{\mathcal{I}}(n) \Big)^{-1} \Big\|_{2,\infty} \\ &+ \|\mathbf{w}(n+1) - \mathbf{w}(n)\|_2 \end{aligned} \tag{38}$$

where $\mathbf{A}^+$ is the Moore-Penrose pseudo-inverse of matrix $\mathbf{A}$ and $\rho(n)$ is defined as $\rho(n) := 1 - \frac{\alpha^2}{\sigma^2} s_M(n)$, with $s_M(n)$ being the minimum eigenvalue of $\mathbf{X}^*(n)\mathbf{D}(n)\mathbf{X}(n)$ and the $(2,\infty)$-norm of a matrix $\mathbf{A}$ is defined as $\|\mathbf{A}\|_{2,\infty} := \max_{\mathbf{x}:\mathbf{x}\neq\mathbf{0}} \frac{\|\mathbf{A}\mathbf{x}\|_2}{\|\mathbf{x}\|_\infty}$.

The first term on the right hand side corresponds to the linear convergence of the EM algorithm, the second term corresponds to the observation noise, the third term corresponds to the error bias with respect to the genie-aided solution, and the fourth term corresponds to the evolution of the true vector $\mathbf{w}(n)$. Note that we are allowing the target $\mathbf{w}(n)$ to change with time in the steady state. A popular model to describe the evolution of the parameter vector in statistical signal processing is the random walk model of the form:

$$\mathbf{w}(n+1) = \mathbf{w}(n) \ + \kappa\boldsymbol{\delta}(n) \tag{39}$$

where $\boldsymbol{\delta}(n)$ is a white Gaussian random vector with covariance matrix $\Delta(n)$ and κ is a scaling constant (See, for example, [21]). The scaling constant κ represents the speed of the time evolution of $\mathbf{w}(n)$. In order for the error recurrence relation to remain valid, we need to assume $\kappa \ll 1$, so that the estimate $\hat{\mathbf{w}}(n)$ remains in a small neighborhood of the target $\tilde{\mathbf{w}}(n)$.

If we further assume that the last three terms on the right hand side do not change rapidly with n, using the Cauchy-Schwarz inequality and averaging over $\boldsymbol{\delta}(n)$ (assuming independence between $\boldsymbol{\delta}(n)$ and $\boldsymbol{\eta}(n)$), we get:

$$\begin{aligned} \epsilon(n) \ \lessapprox \ \frac{1}{1-\rho(n)^K} \Bigg(& \frac{\sigma\sqrt{\mathrm{Tr}\left((\mathbf{X}^*_{\mathcal{I}}(n)\mathbf{D}(n)\mathbf{X}_{\mathcal{I}}(n))^{-1} \right)} + \gamma\alpha^2}{s_{\min}\big(\mathbf{X}^*_{\mathcal{I}}(n)\mathbf{D}(n)\mathbf{X}_{\mathcal{I}}(n)\big)} \\ & + \kappa\sqrt{\mathrm{Tr}(\Delta(n))} \Bigg) \end{aligned} \tag{40}$$

where $s_{\min}(\mathbf{A})$ denotes the minimum eigenvalue of the matrix $\mathbf{A} \in \mathbb{C}^{M\times M}$. The first term on the right hand side demonstrates the trade-off between the denoising of the estimate and the additional cost due to $\mathcal{L}_1$-regularization. The second term corresponds to the regeneration of the unknown vector $\mathbf{w}(n)$. Finally, the factor of $1/(1-\rho(n)^K)$ in the error bound is due to the linear convergence of the EM algorithm.

C. Error Performance Comparison of SPARLS and RLS

In the time-invariant scenario, choosing $\lambda < 1$, will result in a persistent steady state MSE error as $n \to \infty$, unlike RLS which converges to the true vector as the number of measurements tend to infinity (with $\lambda = 1$). However, the steady state MSE error of SPARLS can be sufficiently reduced by choose λ close enough to 1 in the low sparsity regime. In fact, in the following theorem, we show that for L/M small enough and for large enough but *finite* number of measurements$n$, $\lambda < 1$ sufficiently close to 1, and an appropriate choice of γ, the MSE performance of SPARLS is superior to that of RLS (with $\lambda = 1$). This is indeed in line with the premises of compressive sampling, which guarantee superior performance with significantly lower number of measurements:

Theorem 4.2: Consider a stationary environment, for which the RLS algorithm operates with $\lambda = 1$ and recovers the true tap-weight vector $\mathbf{w}_0$ as $n \to \infty$. Let $\epsilon(n)$ and $\epsilon_{RLS}(n)$ denote the average instantaneous errors of the SPARLS and RLS algorithms at the nth iteration, respectively. Then, for a given n_0 large enough, there exist constants $0 < a < 1$, $\lambda_0 \in (0,1)$ sufficiently close to 1 and γ_0 such that for $\lambda = \lambda_0$ and $\gamma = \gamma_0$ we have

$$\epsilon(n_0) < \epsilon_{RLS}(n_0), \tag{41}$$

for $L/M < a$.

Idea of proof: The proof uses basic ideas regarding the Basis Pursuit algorithms in compressed sensing (See, for example, [28] and [5]) and is given in Appendix C.

In fact, the MSE of SPARLS can be significantly lower than that of RLS for finite n in the low sparsity regime, i.e., $L \ll M$. This is evident in the fact that only the components of noise corresponding to the index set $\mathcal{I}$ appear in the error expression of SPARLS in Eq. (38), whereas all the noise coordinates contribute to the MSE of RLS. This can also be observed from Fig. 5. Here, we have $L = 5$ and $M = 100$. For $n_0 \approx 120$, SPARLS achieves its steady state error level, while

it takes a much longer time for RLS to achieve the same MSE (in about 500 iterations). Finally, as simulation studies reveal, the SPARLS algorithm has significant MSE advantages over the RLS algorithm, especially in low SNR and low sparsity regimes.

D. Complexity and Storage Issues

The SPARLS algorithm has a computational complexity of $\mathcal{O}(M^2)$ multiplications per step, which coincides with the order of complexity of the RLS algorithm [17]. In what follows, we motivate the use of the LCU subroutine and its role in potentially decreasing the computational complexity of the SPARLS algorithm under the hypothesis that the index set $\mathcal{I}^{(0)}(n)$ does not vary much across different n in the steady state, *i.e.*, $|\mathcal{I}^{(0)}(n) \backslash \mathcal{I}^{(0)}(n-1)| \ll |\mathcal{I}^{(0)}(n)|$.

If the LCU sub-routine is used in lines 1 and 5 of the LCEM algorithm, it will be called a total of K times for each new input $x(n)$ and requires a total of $\sum_{\ell=0}^{K-1} \left(|\mathcal{I}_+^{(\ell)}(n)|+|\mathcal{I}_-^{(\ell)}(n)|\right)$ column updates overall. For each $i \in \mathcal{I}_+^{(\ell)}(n) \cup \mathcal{I}_-^{(\ell)}(n)$, the ith column of $\tilde{\mathbf{B}}(n)$ requires a total of $M(n-t_i)+2$ multiplications. Hence, the total number of multiplications required for K runs of the LCU sub-routine is given by $\sum_{\ell=0}^{K-1} \sum_{i \in \mathcal{I}_+^{(\ell)}(n) \cup \mathcal{I}_-^{(\ell)}(n)} (M(n-t_i)+2)$. The hypothesis of $|\mathcal{I}^{(0)}(n) \backslash \mathcal{I}^{(0)}(n-1)| \ll |\mathcal{I}^{(0)}(n)|$, implies that the indices t_i are very close to n. In other words, $n - t_i \approx \mathcal{O}(1)$, for all $t_i \in \mathcal{I}^{(0)}(n)$. Therefore, the total number of multiplications will be $\mathcal{O}(KMN)$, where $N := \frac{1}{K} \sum_{\ell=0}^{K-1} \left(|\mathcal{I}_+^{(\ell)}(n)| + |\mathcal{I}_-^{(\ell)}(n)|\right)$.

Moreover, the LCEM algorithm requires $M\left(|\mathcal{I}_+^{(\ell)}(n)| + |\mathcal{I}_-^{(\ell)}(n)|\right)$ multiplications at the ℓth iteration in order to perform the E step. Thus, for a total of K iterations, the number of multiplications carried out by the LCEM algorithm will be KMN. For a sparse signal $\mathbf{w}(n)$, one expects to have $N \approx \mathcal{O}(\|\mathbf{w}(n)\|_0) = \mathcal{O}(L)$. Therefore, the overall complexity of the LCEM algorithm is roughly of the order $\mathcal{O}(KLM)$. Thus under the hypothesis of $|\mathcal{I}^{(0)}(n) \backslash \mathcal{I}^{(0)}(n-1)| \ll |\mathcal{I}^{(0)}(n)|$, the SPARLS algorithm has a lower computational complexity than the RLS algorithm, which requires $\mathcal{O}(M^2)$ multiplications for each step.

Note that the assumption of $|\mathcal{I}^{(0)}(n) \backslash \mathcal{I}^{(0)}(n-1)| \ll |\mathcal{I}^{(0)}(n)|$ may be violated at some steps of the algorithm. This can, for example, happen when the support of the true vector changes over time. However, even when the support of the true vector is constant over time, a new component, say i, may arise in $\mathcal{I}^{(0)}(n)$ after a long time ($t_i \ll n$). Therefore, the LCU routine needs to update the corresponding column of $\tilde{\mathbf{B}}(n)$ using all the previous regressors from time t_i to n. Moreover, the LCU subroutine requires storing all the regressors $x(j)$ from time $j = \min_i t_i$ to n. However, simulation studies reveal that such events are very rare (a component being inactive for a long time which suddenly arises in $\mathcal{I}^{(0)}(n)$). Although this is a drawback compared to RLS (in terms of storage requirements), the cost of storing a finite number of regressors is traded off with potential computational complexity reduction. Finally, note that in any case the cumulative computational complexity of SPARLS using the LCU subroutine (from time 1 to n) will always be lower or equal to that of RLS.

E. Adjusting the Parameters of SPARLS

Parameter α: As mentioned earlier in Section II-C, the parameter α in the SPARLS algorithm must be chosen such that $\alpha^2 \leq \sigma^2/s_1$, where s_1 is the largest eigenvalue of $\mathbf{D}^{1/2}(n)\mathbf{X}(n)\mathbf{X}^*(n)\mathbf{D}^{1/2}$. This constraint clearly depends on the underlying statistical characteristics of the input sequence $x(n)$. Here, we investigate this constraint for a Gaussian i.i.d. input sequnce, *i.e.*, $x(i) \sim \mathcal{N}(0, \nu^2)$, for $i = 1, 2, \cdots, n$, for simplicity. Generalization to other stationary input sequences is possible.

First, note that the maximum eigenvalue of the above matrix is equal to the maximum eigenvalue of $\mathbf{C}(n) := \mathbf{X}^*(n)\mathbf{D}(n)\mathbf{X}(n)$. Recall that the rows of the matrix $\mathbf{X}(n)$ are the tap inputs at times $1, 2, \cdots, n$. Hence, we have

$$\mathbf{C}(n) = \sum_{k=1}^{n} \lambda^{n-k} \mathbf{x}(k)\mathbf{x}^*(k) \tag{42}$$

where $\mathbf{x}(k)$ is the tap input at time k. Hence, the (i,j)th element of the $\mathbf{C}(n)$ can be expressed as $C_{ij}(n) = \sum_{k=1}^{n} \lambda^{n-k} x_i(k) x_j^*(k)$. Next, we invoke the *independence assumption* (See, for example, [17], [22] and [32]). The independence assumption implies that the tap input vectors $\mathbf{x}(1), \mathbf{x}(2), \cdots, \mathbf{x}(n)$ form a sequence of statistically independent vectors. Moreover, the elements of each input vector are distributed i.i.d. and according to $\mathcal{N}(0, \nu^2)$. Hence, the set $\{x_i(k)\}$ for $i = 1, 2, \cdots, M$ and $k = 1, 2, \cdots, n$ consists of i.i.d. zero mean Gaussian random variables with variance ν^2.

The exponentially weighted random matrix $\mathbf{C}(n)$ formed by the set $\{x_i(k)\}$, can be identified as the empirical estimate of the covariance matrix through an exponentially weighted moving average. Such random matrices often arise in portfolio optimization techniques (See, for example, [24]). In [24], the eigen-distribution of such matrices is studied and compared to those of Wishart ensembles. Using the resolvent technique (See, for example, [27]), it is shown in [24] that in the limit of $M \to \infty$ and $\lambda \to 1$, with $Q := 1/M(1-\lambda)$ fixed, and $n \to \infty$, the eigenvalues of the matrix $(1-\lambda)\mathbf{C}(n)$ are distributed according to the density

$$\rho(s) = \frac{Qv}{\pi} \tag{43}$$

where v is the solution to the non-algebraic equation $\frac{s}{\nu^2} - \frac{vs}{\tan(vs)} + \log(v\nu^2) - \log\sin(vs) - \frac{1}{Q} = 0$.

For example, by solving the above equation numerically for $Q = 2$ and $\nu = 1$, the minimum and maximum eigenvalues of $(1-\lambda)\mathbf{C}(n)$ are found to be 0.30 and 2.37, respectively. As it is shown in [24], for finite but large values of M, the empirical eigen-distribution is very similar to the asymptotic case. Therefore, it is possible to obtain an estimate of s_1, and choose α such that $\alpha^2/\sigma^2 \leq 1/s_1$ with high probability. Moreover, the asymptotic value of $\rho(n) = 1 - \alpha^2/\sigma^2 s_M(n)$ as $n \to \infty$, can be estimated using the minimum eigenvalue of $\mathbf{C}(n)$. Note that the above concentration result can be extended to the case of correlated input sequences, which is studied in [27].

Parameter γ: The parameter γ is an additional degree of freedom which controls the trade-off between sparseness of

the output (computational complexity) and the MSE. For very small values of γ, the SPARLS algorithm coincides with the RLS algorithm. For very large values of γ, the output will be the zero vector. Thus, there are intermediate values for γ which result in low MSE and sparsity level which is desired. The parameter γ can be fine-tuned according to the application we are interested in. For example, for estimating the wireless multi-path channel, γ can be optimized with respect to the number of channel taps (sparsity), temporal statistics of the channel and noise level via exhaustive simulations or experiments. Note that γ can be fine-tuned offline for a certain application. Theoretical bounds on γ for near-oracle recovery are discussed in [5] and [28]. There are also some heuristic methods for choosing γ which are discussed in [15]. The noise variance σ^2 can be estimated in various ways, which are discussed in [15] and [20].

Parameter λ: The parameter λ can be fine-tuned based on the time-variation rate of the true vector, as it is done for the RLS algorithm. However, for the SPARLS algorithm we assume that $\lambda \in (0, 1)$, in the cost function given in Eq. (16), even when the true vector is constant over time. This is due to the fact that with $\lambda = 1$, which is used for RLS algorithm when the true vector is constant over time, for large values of n, the quadratic term in Eq. (16) grows unboundedly and dominates the ℓ_1-penalty term. Hence, the minimizer of the cost function, for large values of n, coincides with that obtained by the RLS algorithm, which is not necessarily sparse. Restricting λ to lie in the open interval $(0, 1)$ maintains a proper scaling between the quadratic and ℓ_1-penalty terms, since the quadratic term will remain bounded over time. The lack of scalability of the Laplacian prior induced by the ℓ_1-penalty term, has led some researchers to employ the Gaussian Scale Mixture (GSM) densities, which are known to be scale invariant (See [2] and [25]). However, there are a number of well-established performance results that show potential near-oracle performance when the Laplacian prior is used (See [5] and [28]). In this regard, we have chosen to use the Laplacian prior. Nevertheless, generalization of the SPARLS algorithm equipped with other penalization schemes (such as the GSM prior) is possible.

V. Simulation Studies

We consider the estimation of a sparse multi-path wireless channel generated by the Jake's model [19]. In the Jake's model, each component of the tap-weight vector is a sample path of a Rayleigh random process with autocorrelation function given by

$$R(n) = \mathrm{J}_0(2\pi n f_d T_s) \tag{44}$$

where $\mathrm{J}_0(\cdot)$ is the zeroth order Bessel function, f_d is the Doppler frequency shift and T_s is the channel sampling interval. The dimensionless parameter $f_d T_s$ gives a measure of how fast each tap is changing over time. Note that the case $f_d T_s = 0$ corresponds to a constant tap-weight vector. Thus, the Jake's model covers constant tap-weight vectors as well. For the purpose of simulations, T_s is normalized to 1.

We consider two different input sequences $\{x(i)\}_{i=1}^{\infty}$ for simulations: Gaussian i.i.d. input sequence, where each $x(i)$ is distributed according to $\mathcal{N}(0, 1/M)$, and i.i.d. random Rademacher input sequence, where each $x(i)$ takes the values $\pm 1/\sqrt{M}$ with equal probability. The SNR is defined as $\mathbb{E}\{\|\mathbf{w}\|_2^2\}/\sigma^2$, where σ^2 is the variance of the Gaussian zero-mean observation noise. The locations of the nonzero elements of the tap-weight vector are randomly chosen in the set $\{1, 2, \cdots, M\}$ and the SPARLS algorithm has no knowledge of these locations. Also, all the simulations are done with $K = 1$, *i.e.*, a single LCEM iteration per new data and the column updates are performed using the LCU subroutine. Finally, a choice of $\alpha = \sigma/2$ has been used (Please see Section IV-E).

We compare the performance of the SPARLS and RLS with respect to two performance measures. The first measure is the MSE defined as

$$\text{MSE} := \frac{\mathbb{E}\{\|\hat{\mathbf{w}} - \mathbf{w}\|_2^2\}}{\mathbb{E}\{\|\mathbf{w}\|_2^2\}} \tag{45}$$

where the averaging is carried out by 50000 Monte Carlo samplings. The number of samples has been chosen large enough to ensure that the uncertainty in the measurements is less than 1%. The second measure is the computational complexity ratio (CCR) which is defined by

$$\text{CCR} := \frac{\text{average number of multiplications for SPARLS}}{\text{average number of multiplications for RLS}} \tag{46}$$

A. Time-invariant Scenario: $f_d = 0$

In this case, the best choice of λ for the RLS algorithm is $\lambda = 1$. As mentioned earlier in Section IV-E, in order to maintain the scaling between the quadratic and ℓ_1-penalty terms of the cost function, we choose $\lambda < 1$ for SPARLS. A value of $\lambda = 0.999$ has been chosen for the SPARLS algorithm. The corresponding values of γ are obtained by exhaustive simulations and are listed in Tables I and II. Moreover, we have $L = 5$ and $M = 100$, and both RLS and SPARLS algorithms are run for Gaussian and Rademacher i.i.d. input sequences of length 500.

Figures 3 and 4 show the mean squared error and computational complexity ratio of the SPARLS and RLS algorithm for Gaussian and Rademacher i.i.d. sequences, respectively. The SPARLS algorithm gains about 5 dB in MSE and about 75% less computational complexity.

Figure 5 shows the time-domain behvior of the SPARLS and RLS algorithms for three different SNR levels of 10 dB, 20 dB and 30 dB, with Gaussian i.i.d. input (the case of Rademacher i.i.d. input is very similar, and thus omitted for brevity). As it is clear from the figure, for low number of measurements, the SPARLS algorithm significantly outperforms the RLS algorithm in terms of MSE.

B. Time-varying Scenario: $f_d \neq 0$

In order to compare the performance of the SPARLS and RLS algorithms, we first need to optimize the RLS algorithm for the given time-varying channel. By exhaustive simulations,

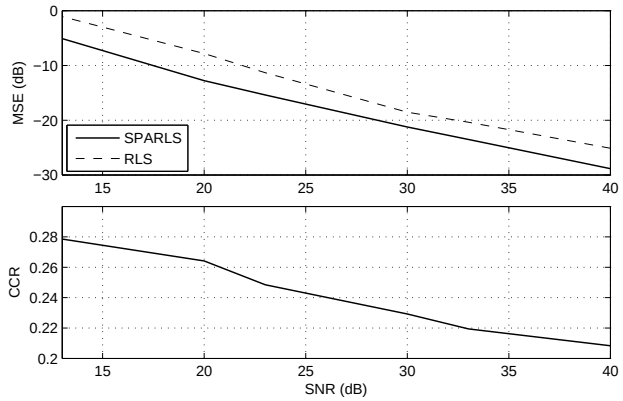

Fig. 3. MSE of RLS and SPARLS vs. SNR for $f_dT_s = 0$, for i.i.d. Gaussian input sequence.

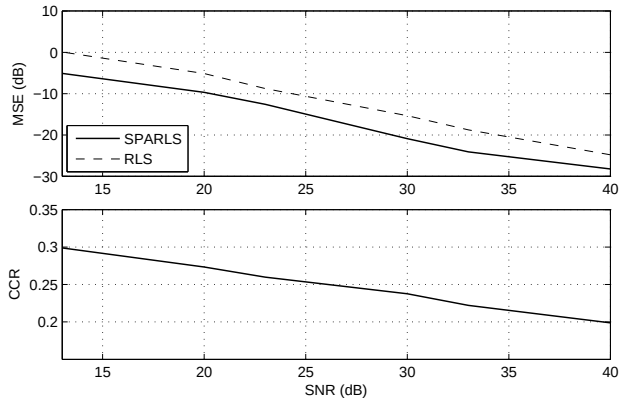

Fig. 4. MSE of RLS and SPARLS vs. SNR for $f_dT_s = 0$, for i.i.d. Rademacher input sequence.

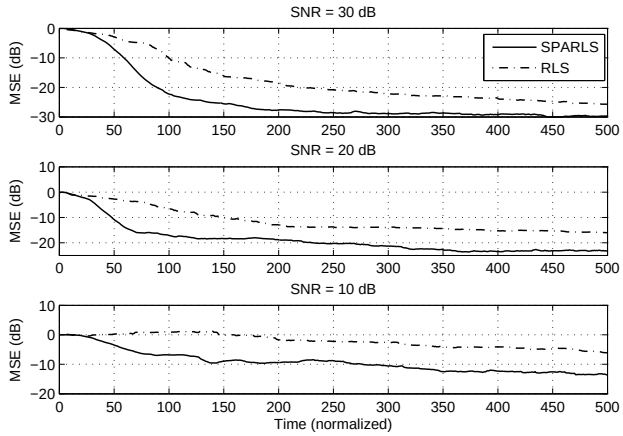

Fig. 5. MSE of RLS and SPARLS vs. time for SNR $= 10, 20$ and 30 dB and i.i.d. Gaussian input sequence. The time scale is normalized to the signaling interval of the input sequence.

TABLE I
OPTIMAL VALUES OF λ FOR THE RLS ALGORITHM AND THE CORRESPONDING VALUES OF γ FOR THE SPARLS ALGORITHM VS. σ^2 AND f_dT_s, FOR I.I.D. GAUSSIAN INPUT.

σ^2 \ f_dT_s	0	0.0001	0.0005	0.001	0.005
0.0001	(0.999, 100)	(0.97, 100)	(0.96, 100)	(0.97, 100)	(0.99, 200)
0.0005	(0.999, 50)	(0.97, 50)	(0.97, 50)	(0.98, 40)	(0.99, 100)
0.001	(0.999, 35)	(0.98, 35)	(0.98, 30)	(0.99, 25)	(0.99, 60)
0.005	(0.999, 15)	(0.99, 15)	(0.99, 15)	(0.99, 10)	(0.99, 30)
0.01	(0.999, 13)	(0.99, 10)	(0.99, 8)	(0.99, 8)	(0.99, 15)
0.05	(0.999, 3)	(0.99, 3)	(0.99, 3)	(0.99, 3)	(0.99, 5)

TABLE II
OPTIMAL VALUES OF λ FOR THE RLS ALGORITHM AND THE CORRESPONDING VALUES OF γ FOR THE SPARLS ALGORITHM VS. σ^2 AND f_dT_s, FOR I.I.D. RADEMACHER INPUT.

σ^2 \ f_dT_s	0	0.0001	0.0005	0.001	0.005
0.0001	(0.999, 100)	(0.97, 90)	(0.96, 90)	(0.97, 90)	(0.99, 250)
0.0005	(0.999, 50)	(0.97, 50)	(0.97, 45)	(0.98, 45)	(0.99, 100)
0.001	(0.999, 35)	(0.98, 35)	(0.98, 35)	(0.99, 20)	(0.99, 70)
0.005	(0.999, 10)	(0.99, 10)	(0.99, 10)	(0.99, 10)	(0.99, 30)
0.01	(0.999, 8)	(0.99, 5)	(0.99, 5)	(0.99, 5)	(0.99, 10)
0.05	(0.999, 5)	(0.99, 4)	(0.99, 4)	(0.99, 4)	(0.99, 7)

the optimum forgetting factor, λ, of the RLS algorithm can be obtained for various choices of SNR and f_dT_s.

As for the SPARLS algorithm, we perform a partial optimization as follows: we use the values of Table 1 for λ and optimize over γ with exhaustive simulations. Note that with such choices of parameters λ and γ, we are comparing a near-optimal parametrization of SPARLS with the optimal parametrization of RLS. The performance of the SPARLS can be further enhanced by simultaneous optimization over both λ and γ. The pairs of (λ, γ) corresponding to the optimal values of γ and λ vs. σ^2 and f_dT_s are summarized in Tables 1 and 2, for i.i.d. Gaussian and Rademacher input sequences, respectively.

Figures 6 and 7 show the mean squared error and computational complexity ratio of the RLS and SPARLS algorithms for $f_dT_s = 0.0001, 0.0005, 0.001$ and 0.005, with $L = 5$ and $M = 100$ and i.i.d. Gaussian input, respectively. Similarly, Figures 8 and 9 show the corresponding curves for i.i.d. Rademacher inputs. In both cases, the SPARLS algorithm outperforms the RLS algorithm with about 7 dB gain in the MSE performance. Moreover, the computational complexity of the SPARLS (using the LCU subroutine) is about 80% less than that of RLS on average.

VI. CONCLUSION

We have developed a Recursive $\mathcal{L}_1$-Regularized Least Squares (SPARLS) algorithm for the estimation of a sparse tap-weight vector in the adaptive filtering setting. The SPARLS algorithm estimates the tap-weight vector based on noisy observations of the output stream, using an Expectation-Maximization type algorithm. We have presented analytical results regarding the convergence, steady state error and parameter adjustments of the SPARLS algorithm. Simulation studies, in the context of multi-path wireless channel estimation, show that the SPARLS algorithm has significant improvement over the conventional widely-used Recursive Least Squares (RLS) algorithm in terms of mean squared error

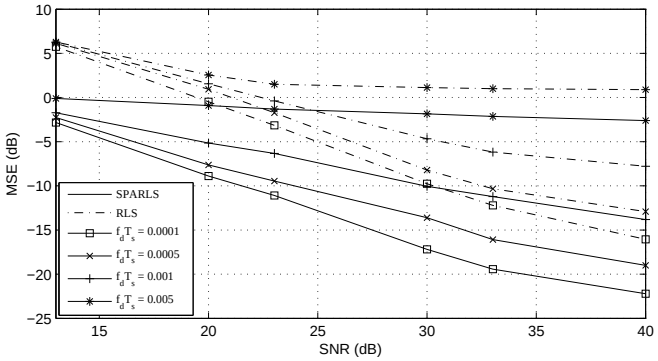

Fig. 6. MSE of RLS and SPARLS vs. SNR for $f_dT_s = 0.0001, 0.0005, 0.001$ and 0.005, for i.i.d. Gaussian input sequence.

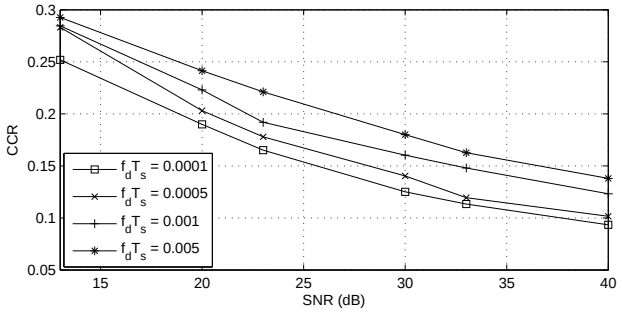

Fig. 7. CCR vs. SNR for $f_dT_s = 0.0001, 0.0005, 0.001$ and 0.005, for i.i.d. Gaussian input sequence.

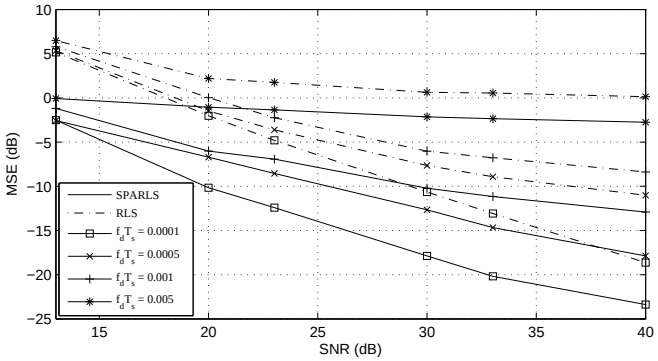

Fig. 8. MSE of RLS and SPARLS vs. SNR for $f_dT_s = 0.0001, 0.0005, 0.001$ and 0.005, for i.i.d. Rademacher input sequence.

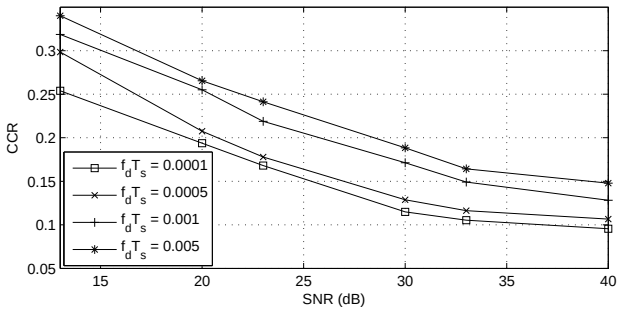

Fig. 9. CCR vs. SNR for $f_dT_s = 0.0001, 0.0005, 0.001$ and 0.005, for i.i.d. Rademacher input sequence.

(MSE). Moreover, these simulation results suggest that the SPARLS algorithm (using the LCU subroutine) has a lower computational complexity than the RLS algorithm, when the underlying tap-weight vector has a fixed support.

APPENDIX A
PROOF OF THEOREM 4.1

Suppose that we perform the LCEM algorithm a total of K times in each step. The estimate at time $n+1$ can be written as

$$\hat{\mathbf{w}}(n+1) = \underbrace{\mathcal{M}_n \circ \mathcal{M}_n \circ \cdots \circ \mathcal{M}_n}_{K \text{ times}}(\hat{\mathbf{w}}(n)) = \mathcal{M}_n^K(\hat{\mathbf{w}}(n)). \tag{47}$$

Now, consider the objective function $f_n(\mathbf{w})$:

$$\begin{aligned} f_n(\mathbf{w}) \quad = \quad \text{const.} + \frac{1}{2\sigma^2}\Big\{ & \mathbf{w}^*\mathbf{X}^*(n)\mathbf{D}(n)\mathbf{X}(n)\mathbf{w} \\ & -2\,\mathrm{Re}\left\{\mathbf{w}_0^*\mathbf{X}^*(n)\mathbf{D}(n)\mathbf{X}(n)\mathbf{w}\right\} \\ & -2\,\mathrm{Re}\left\{\boldsymbol{\eta}^*(n)\mathbf{D}(n)\mathbf{X}(n)\mathbf{w}\right\}\Big\} \\ & + \gamma\|\mathbf{w}\|_1 \end{aligned} \tag{48}$$

Using the stationarity hypothesis, we assume that the input vector $\mathbf{x}(i)$ at time i is a random vector with zero mean entries and covariance $\mathbf{R}_x$. For n large enough, the entries of the matrix $\mathbf{X}^*(n)\mathbf{D}(n)\mathbf{X}(n)$ can be written as

$$(\mathbf{X}^*(n)\mathbf{D}(n)\mathbf{X}(n))_{ij} = \sum_{k=0}^{n-1} \lambda^k x_i(k)x_j^*(k) \to \frac{1}{1-\lambda}(\mathbf{R}_x)_{ij}, \tag{49}$$

where we have invoked the strong law of large numbers for weighted sums [12]. If we take the expectation of the objective function with respect to $\boldsymbol{\eta}(n)$, we get:

$$\begin{aligned} f(\mathbf{w}) \quad := \quad \text{const.} + \frac{1}{2\sigma^2(1-\lambda)}\Big\{ & \mathbf{w}^*\mathbf{R}_x\mathbf{w} - 2\,\mathrm{Re}\left\{\mathbf{w}_0^*\mathbf{R}_x\mathbf{w}\right\}\Big\} \\ & + \ \gamma\|\mathbf{w}\|_1 \end{aligned} \tag{50}$$

as $n \to \infty$. Note that $f(\mathbf{w})$ is independent of n. From the continuity of the minimizer of $f_n(\mathbf{w})$ in $\boldsymbol{\eta}(n)$, we conclude that

$$\mathbb{E}_\eta\{\tilde{\mathbf{w}}(n)\} \to \tilde{\mathbf{w}}_0 \tag{51}$$

as $n \to \infty$ almost surely. The above limit process implies the existence of a limit genie-aided estimate as the number of observations n tends to infinity.

We want to show that the SPARLS algorithm converges to $\tilde{\mathbf{w}}_0$ almost surely. Throughout the rest of the proof, we drop the expectation with respect to $\boldsymbol{\eta}$ for notational simplicity and assume it implicitly in our derivations.

Consider Kn_0 successive iterations of the EM algorithm on a single cost function $f_n(\mathbf{w})$ at time n, resulting in the set of estimates $\{\mathcal{M}_n^i(\hat{\mathbf{w}}(n))\}_{i=1}^{Kn_0}$. It is possible to choose n_0 large enough such that

$$\left|f_n\big(\mathcal{M}_n^{Kn_0}(\hat{\mathbf{w}}(n))\big) - f_n(\tilde{\mathbf{w}}(n))\right| < \epsilon/3 \tag{52}$$

due to the guaranteed convergence of the EM algorithm applied to a single cost function $f_n(\mathbf{w})$ [13]. In other words, due to the continuity of $f_n(\mathbf{w})$, we can reach an arbitrarily

small neighborhood of $\tilde{\mathbf{w}}(n)$ in finite time by successively applying the EM iteration across the curve $f_n(\mathbf{w})$.

Now, consider applying the SPARLS iterations from time n to $n+n_0-1$, resulting in the estimates $\{\hat{\mathbf{w}}(n+i)\}_{i=1}^{n_0}$, where $\hat{\mathbf{w}}(n+i) := \mathcal{M}_{n+i-1}^K(\hat{\mathbf{w}}(n+i-1))$. By the continuity of the mapping $\mathcal{M}_n$ in the linear and quadratic coefficients of $\mathbf{w}$, and by the continuity of the function $f_n(\mathbf{w})$ in $\mathbf{w}$, we can choose n large enough such that

$$\begin{aligned}&\big|f_{n+n_0}(\hat{\mathbf{w}}(n+n_0)) - f_{n+n_0}(\mathcal{M}_n^{Kn_0}(\hat{\mathbf{w}}(n)))\big| \\ &= \Big|f_{n+n_0}\Big(\mathcal{M}_{n+n_0-1}^K \circ \mathcal{M}_{n+n_0-2}^K \circ \cdots \circ \mathcal{M}_n^K(\hat{\mathbf{w}}(n))\Big) \\ &\quad - f_{n+n_0}\big(\mathcal{M}_n^{Kn_0}(\hat{\mathbf{w}}(n))\big)\Big| \\ &\leq \epsilon/3 \end{aligned} \tag{53}$$

Since the coefficients of the linear and quadratic terms in $f_n(\mathbf{w})$ are independent of n in the limit of $n \to \infty$, $f_n(\mathbf{w})$ tends to $f(\mathbf{w})$ in a point-wise fashion. Let

$$W := \mathcal{B}_{2\|\tilde{\mathbf{w}}_0\|_2}(0) := \big\{\mathbf{w} \in \mathbb{C}^M : \|\mathbf{w}\|_2 \leq 2\|\tilde{\mathbf{w}}_0\|_2\big\} \tag{54}$$

Since $\mathbb{C}^M$ is a separable metric space, by the Egorov's theorem [31], the point-wise convergence of the continuous bounded functions $f_n(\mathbf{w})$ to $f(\mathbf{w})$ in the compact set W, implies uniform convergence everywhere except on some subset of arbitrarily small measure. Hence, for any positive $\epsilon > 0$, there exists an integer N such that for all $n > N$ we have

$$\max_{\mathbf{w}\in W} |f_n(\mathbf{w}) - f(\mathbf{w})| < \epsilon/12. \tag{55}$$

By Eqs. (52) and (53), it is implied that for ϵ small enough, $\hat{\mathbf{w}}(n+n_0)$ and $\mathcal{M}_n^{Kn_0}(\hat{\mathbf{w}}(n))$ are in a small neighborhood of $\tilde{\mathbf{w}}(n)$ (due to the continuity of $f_n(\cdot)$ and $f_{n+n_0}(\cdot)$). Hence, by choosing n large enough, the points $\hat{\mathbf{w}}(n+n_0)$ and $\mathcal{M}_n^{Kn_0}(\hat{\mathbf{w}}(n))$ lie inside the set W. We thus have

$$\begin{aligned}&|f_n(\hat{\mathbf{w}}(n+n_0)) - f_n(\mathcal{M}_n^{Kn_0}(\hat{\mathbf{w}}(n)))| \\ &\leq |f_{n+n_0}(\hat{\mathbf{w}}(n+n_0)) - f_n(\hat{\mathbf{w}}(n+n_0))| \\ &\quad + |f_{n+n_0}(\mathcal{M}_n^{Kn_0}(\hat{\mathbf{w}}(n))) - f_n(\mathcal{M}_n^{Kn_0}(\hat{\mathbf{w}}(n)))| \\ &\quad + |f_{n+n_0}(\mathcal{M}_n^{Kn_0}(\hat{\mathbf{w}}(n))) - f_{n+n_0}(\hat{\mathbf{w}}(n+n_0))| \\ &\leq 4\epsilon/12 + \epsilon/3 = 2\epsilon/3 \end{aligned} \tag{56}$$

Hence, after n_0 iterations of the SPARLS algorithm, we have

$$\begin{aligned}&|f_n(\hat{\mathbf{w}}(n+n_0)) - f_n(\tilde{\mathbf{w}}(n))| \\ &\leq |f_n(\hat{\mathbf{w}}(n+n_0)) - f_n(\mathcal{M}_n^{Kn_0}(\hat{\mathbf{w}}(n)))| \\ &\quad + |f_n(\mathcal{M}_n^{Kn_0}(\hat{\mathbf{w}}(n))) - f_n(\tilde{\mathbf{w}}(n))| \\ &< 2\epsilon/3 + \epsilon/3 = \epsilon. \end{aligned}$$

Therefore, after n_0 iterations, we can reach an arbitrarily small neighborhood of $\tilde{\mathbf{w}}(n)$ for all n, due to the continuity of $f_n(\mathbf{w})$. Since $\tilde{\mathbf{w}}(n) \to \tilde{\mathbf{w}}_0$, we can reach an arbitrarily small neighborhood of $\tilde{\mathbf{w}}_0$ in finite time for all n. Therefore, the SPARLS algorithm converges to $\tilde{\mathbf{w}}_0$ almost surely.

APPENDIX B
STEADY STATE ERROR ANALYSIS: DERIVATIONS

First, we briefly overview the convergence properties of the EM algorithm. The global and componentwise convergence of the EM algorithm has been widely studied in the statistics literature (See, for example, [13] and [23]). Suppose, for the moment, that the mapping $\mathcal{M}_n$ is differentiable at $\tilde{\mathbf{w}}(n)$, the maximizer of the objective function in Eq. (18). We can therefore write the Taylor expansion as follows:

$$\begin{aligned}\hat{\mathbf{w}}^{(\ell+1)}(n) - \tilde{\mathbf{w}}(n) &= D\mathcal{M}_n(\tilde{\mathbf{w}}(n))\big(\hat{\mathbf{w}}^{(\ell)}(n) - \tilde{\mathbf{w}}(n)\big) \\ &\quad + \mathcal{O}\big(\|\hat{\mathbf{w}}^{(\ell)}(n) - \tilde{\mathbf{w}}(n)\|^2\big), \end{aligned} \tag{57}$$

where $D\mathcal{M}_n$ is the Jacobian of the mapping $\mathcal{M}_n$ and we have used the fact that $\tilde{\mathbf{w}}(n)$ is a fixed point for the mapping $\mathcal{M}_n$. Hence, in a sufficiently small neighborhood of $\tilde{\mathbf{w}}(n)$, the EM algorithm is simply a linear mapping. However, in our case the mapping $\mathcal{M}_n$ is not differentiable, since the soft thresholding function is not differentiable at points $-\gamma\alpha^2$ and $\gamma\alpha^2$. We can therefore use the sub-differential of the mapping $\mathcal{M}_n$ in order to study its behavior in a neighborhood of $\tilde{\mathbf{w}}(n)$. Let $\mathcal{E} : \mathbb{C}^M \longmapsto \mathbb{C}^M$ be a mapping defined as:

$$\mathcal{E}(\mathbf{w}) := \big(\mathbf{I} - \frac{\alpha^2}{\sigma^2}\mathbf{X}^*\mathbf{D}\mathbf{X}\big)\mathbf{w} + \frac{\alpha^2}{\sigma^2}\mathbf{X}^*\mathbf{D}\mathbf{d}. \tag{58}$$

Note that we have dropped the dependence on n for notational convenience. The mapping $\mathcal{M}$ is then simply given by $\mathcal{M}(\mathbf{w}) = \mathcal{S} \circ \mathcal{E}(\mathbf{w})$, where $\mathcal{S}(\cdot)$ is the elementwise soft thresholding function, defined in Eq. (25). Although the mapping $\mathcal{E}$ is differentiable, the mapping $\mathcal{S}$ is not. However, as we will see later on, the restriction on the convergence properties of the EM algorithm does not arise from the M step and is mainly due to the E step. Here, we take the approach of working with sub-differentials to avoid introducing smoothing parameters to our setting. In order to simplify the notational presentation, we assume that $\mathbf{w} \in \mathbb{R}^M$. Due to the trivial isomorphism of the vector spaces $\mathbb{C}^M$ and $\mathbb{R}^{2M}$ over the field of real numbers, generalization to $\mathbf{w} \in \mathbb{C}^M$ is straightforward. We can define the sub-differential of the mapping $\mathcal{S}$ as follows (See, for example, [26]):

$$\partial\mathcal{S}(\mathbf{w}) = \mathrm{diag}(h_1, h_2, \cdots, h_M) \tag{59}$$

where

$$h_i := \begin{cases} 1 & |w_i| > \gamma\alpha^2 \\ 0 \leq h_i \leq 1 & |w_i| = \gamma\alpha^2 \\ 0 & |w_i| < \gamma\alpha^2 \end{cases} \tag{60}$$

In addition, from the chain rule for sub-differentials [26], we have

$$\partial\mathcal{M}(\mathbf{w}) = \partial(\mathcal{S} \circ \mathcal{E}(\mathbf{w})) = \big(\partial\mathcal{S}(\mathcal{E}(\mathbf{w}))\big)^*\big(\mathbf{I} - \frac{\alpha^2}{\sigma^2}\mathbf{X}^*\mathbf{D}\mathbf{X}\big) \tag{61}$$

Therefore, by an appropriate choice of the sub-differential of $\mathcal{S}$ at $\tilde{\mathbf{w}}(n)$, we can locally approximate the EM iteration by

$$\begin{aligned}&\hat{\mathbf{w}}^{(\ell+1)}(n) - \tilde{\mathbf{w}}(n) \\ &\approx \big(\partial\mathcal{S}(\mathcal{E}(\tilde{\mathbf{w}}(n)))\big)^*\big(\mathbf{I} - \frac{\alpha^2}{\sigma^2}\mathbf{X}^*(n)\mathbf{D}(n)\mathbf{X}(n)\big) \\ &\quad \times\big(\hat{\mathbf{w}}^{(\ell)}(n) - \tilde{\mathbf{w}}(n)\big) \end{aligned} \tag{62}$$

From the convergence results of [13] and [23], it is known that the linear convergence rate of the EM algorithm is governed by the maximum eigenvalue of the Jacobian $D\mathcal{M}$.

In our case, we need to consider the maximum eigenvalue of $\partial(\mathcal{S}\circ\mathcal{E}(\tilde{\mathbf{w}}(n)))$. Clearly, the maximum eigenvalue of the diagonal matrix $\partial\mathcal{S}(\mathcal{E}(\tilde{\mathbf{w}}(n))$ is bounded above by 1, since all its diagonal elements h_i are bounded as $0 \leq h_i \leq 1$. In fact, the maximum eigenvalue of $\partial\mathcal{S}(\mathcal{E}(\tilde{\mathbf{w}}(n))$ is equal to 1, unless all the elements of $\tilde{\mathbf{w}}(n)$ are in the range $-\gamma\alpha^2 \leq w_i \leq \gamma\alpha^2$, which is very unlikely to happen. This account for the earlier claim that the maximum eigenvalue of $\partial\mathcal{S}$ does not play a significant role in the convergence rate, since it most likely is equal to 1. Therefore, the rate of convergence is governed by the maximum eigenvalue of the matrix $\mathbf{I}-\frac{\alpha^2}{\sigma^2}\mathbf{X}^*(n)\mathbf{D}(n)\mathbf{X}(n)$, which is given by

$$\rho(n) := 1-\frac{\alpha^2}{\sigma^2}s_M(n), \tag{63}$$

where $s_M(n)$ is the minimum eigenvalue of $\mathbf{X}^*(n)\mathbf{D}(n)\mathbf{X}(n)$ (there is more to say about the asymptotic behavior of $\rho(n)$, as $n\to\infty$, in Section IV-E). If we perform the EM iteration a total of K times, we can write:

$$\begin{aligned}\|\hat{\mathbf{w}}(n+1)-\tilde{\mathbf{w}}(n)\|_2 &\leq \Big\|\Big(\partial\mathcal{M}(\tilde{\mathbf{w}}(n))\Big)^K\Big\|_2 \\ &\quad\times\|\hat{\mathbf{w}}(n)-\tilde{\mathbf{w}}(n)\|_2 \\ &\leq \rho(n)^K\|\hat{\mathbf{w}}(n)-\tilde{\mathbf{w}}(n)\|_2\end{aligned}$$

for $\hat{\mathbf{w}}(n)$ in a small neighborhood of $\tilde{\mathbf{w}}(n)$.

Recall that from Lemma 3 of [28], we know that the maximizer of the objective function given in Eq. (18) is unique if $\mathbf{X}_{\mathcal{I}}(n)$ is maximal rank, where $\mathcal{I}=\mathrm{supp}(\mathbf{w}(n))$. Moreover, Lemma 6 of Tropp in [28] establishes that if γ satisfies

$$\gamma \geq \frac{\|\mathbf{X}^*(n)\mathbf{D}^{1/2}(n)(\mathbf{D}^{1/2}(n)\mathbf{X}_{\mathcal{I}}(n))^{+}\mathbf{D}^{1/2}(n)\boldsymbol{\eta}_{\mathcal{I}}(n)\|_\infty}{1-\max\limits_{i\notin\mathcal{I}}|\mathbf{X}_i^*(n)\mathbf{D}(n)\mathbf{X}(n)\mathbf{g}(n)|}, \tag{64}$$

we have

$$\hat{\mathbf{w}}_g(n)-\tilde{\mathbf{w}}(n)=\gamma\sigma^2\Big(\mathbf{X}_{\mathcal{I}}^*(n)\mathbf{D}(n)\mathbf{X}_{\mathcal{I}}(n)\Big)^{-1}\mathbf{g}(n) \tag{65}$$

where $\hat{\mathbf{w}}_g(n)$ is the *genie-aided* estimate of $\mathbf{w}(n)$ given by

$$\begin{aligned}\hat{\mathbf{w}}_g(n) &:= \big(\mathbf{D}^{1/2}(n)\mathbf{X}_{\mathcal{I}}(n)\big)^{+}\mathbf{D}^{1/2}(n)\mathbf{d}_{\mathcal{I}}(n) \\ &= \mathbf{w}(n)+\big(\mathbf{D}^{1/2}(n)\mathbf{X}_{\mathcal{I}}(n)\big)^{+}\boldsymbol{\eta}_{\mathcal{I}}(n)\end{aligned} \tag{66}$$

and $\mathbf{g}(n)$ is in the sub-gradient set of $\|\tilde{\mathbf{w}}(n)\|_1$. The genie-aided estimate corresponds to the least square solution when a genie has provided the support of $\mathbf{w}(n)$ to the estimator and is considered to be a theoretical performance benchmark for the estimation of sparse vectors. Using the relations between $\hat{\mathbf{w}}_g(n)$, $\tilde{\mathbf{w}}(n)$ and $\mathbf{w}(n)$ and triangle inequality we can write:

$$\begin{aligned}\epsilon(n+1) &= \mathbb{E}_\eta\Big\{\|\hat{\mathbf{w}}(n+1)-\tilde{\mathbf{w}}(n)+\tilde{\mathbf{w}}(n)-\mathbf{w}(n) \\ &\qquad +\mathbf{w}(n)-\mathbf{w}(n+1)\|_2\Big\} \\ &\leq \rho(n)^K\epsilon(n)+\mathbb{E}_\eta\Big\{\big\|(\mathbf{D}^{1/2}(n)\mathbf{X}_{\mathcal{I}}(n))^{+}\boldsymbol{\eta}_{\mathcal{I}}(n)\big\|_2\Big\} \\ &+ \gamma\sigma^2\Big\|\Big(\mathbf{X}_{\mathcal{I}}^*(n)\mathbf{D}(n)\mathbf{X}_{\mathcal{I}}(n)\Big)^{-1}\Big\|_{2,\infty} \\ &+ \|\mathbf{w}(n+1)-\mathbf{w}(n)\|_2\end{aligned} \tag{67}$$

where the $(2,\infty)$-norm of a matrix $\mathbf{A}$ is defined as $\|\mathbf{A}\|_{2,\infty} := \max_{\mathbf{x}:\mathbf{x}\neq\mathbf{0}}\frac{\|\mathbf{A}\mathbf{x}\|_2}{\|\mathbf{x}\|_\infty}$.

Appendix C
Proof of Theorem 4.2

For the RLS algorithm (with $\lambda=1$), the error expression is given by

$$\begin{aligned}\epsilon_{RLS}(n+1) &:= \mathbb{E}_\eta\Big\{\big\|\hat{\mathbf{w}}_{RLS}(n+1)-\mathbf{w}(n+1)\big\|_2\Big\} \\ &= \mathbb{E}_\eta\Big\{\big\|\mathbf{X}^{+}(n)\boldsymbol{\eta}(n)\big\|_2\Big\}.\end{aligned} \tag{68}$$

According to Eq. (38) the corresponding error expression for the SPARLS algorithm in a stationary environment is upper bounded as

$$\begin{aligned}\epsilon(n+1) &\leq \rho^K(n)\epsilon(n)+\mathbb{E}_\eta\Big\{\big\|(\mathbf{D}^{1/2}(n)\mathbf{X}_{\mathcal{I}}(n))^{+}\boldsymbol{\eta}_{\mathcal{I}}(n)\big\|_2\Big\} \\ &+ \gamma\alpha^2\Big\|\big(\mathbf{X}_{\mathcal{I}}^*(n)\mathbf{D}(n)\mathbf{X}_{\mathcal{I}}(n)\big)^{-1}\Big\|_{2,\infty}.\end{aligned} \tag{69}$$

Let $\mu(n)$ be the coherence of the matrix $\mathbf{X}^*(n)\mathbf{D}(n)\mathbf{X}(n)$. Now, we claim that for $n_0<\infty$ and $L<1/(3\mu(n_0))$, one can choose γ_0 and $\lambda_0<1$ such that

$$\begin{aligned}&\mathbb{E}_\eta\Big\{\big\|(\mathbf{D}^{1/2}(n_0)\mathbf{X}_{\mathcal{I}}(n_0))^{+}\boldsymbol{\eta}_{\mathcal{I}}(n_0)\big\|_2\Big\} \\ &\quad+\gamma\alpha^2\Big\|\big(\mathbf{X}_{\mathcal{I}}^*(n_0)\mathbf{D}(n_0)\mathbf{X}_{\mathcal{I}}(n_0)\big)^{-1}\Big\|_{2,\infty} \\ &<\mathbb{E}_\eta\Big\{\big\|\mathbf{X}^{+}(n_0)\boldsymbol{\eta}(n_0)\big\|_2\Big\}.\end{aligned} \tag{70}$$

First, note that the claim is obviously true for $\lambda=1$, for an appropriate choice of γ and a sufficiently incoherent measurement matrix $\mathbf{X}(n)$, thanks to the results of Tropp [28] and Ben-Haim et al. [5] on the near-oracle performance of Subspace Pursuit. Next, by the continuity of the pseudo-inverse operator in the argument $\mathbf{D}^{1/2}(n)$, the continuity of the coherence $\mu(n)$ in λ, and finally the continuity of the lower bound on γ in λ (See Eq. (64) or Lemma 6 of [28]), there exist $\lambda_0<1$ and γ_0 such that the above inequality holds.

Note that with the appropriate choice of γ_0 as in [28] and [5], $|\mathcal{I}|\leq L$ with high probability. Hence, for $L\ll M$ (low sparsity regime), the left hand side of Eq. (70) can be significantly smaller than the right hand side. Now, given that the SPARLS algorithm converges to a fixed point (Theorem 4.1), for n_0 large enough, the average instantaneous error of SPARLS, $\epsilon(n_0)$, is a factor of $1/(1-\rho(n_0)^K)$ away from the left hand side of Eq. (70). By choosing K appropriately, one can guarantee that $\rho(n_0)^K \ll 1$. Hence, there exists $0<a<\min\{\frac{1}{3\mu_0 M},1\}$ such that for $L/M<a$, we have $\epsilon(n_0)<\epsilon_{RLS}(n_0)$. This establishes the statement of the theorem.

References

[1] M. Akçakaya, and V. Tarokh, "Shannon Theoretic Limits on Noisy Compressive Sampling", IEEE Trans. on Information Theory, Vol. 56, No. 1, pp. 492-504, Jan 2010.

[2] D. Andrews and C. Mallows, "Scale mixtures of normal distributions," J. R. Stat. Soc., vol. 36, pp. 99 - 102, 1974.

[3] D. Angelosante and G. Giannakis, "RLS-weighted LASSO for adaptive estimation of sparse signals", IEEE Int. Conf. Acounstics, Speech, Signal Process. (ICASSP), April 19 - 24, 2009, Taipei, Taiwan.

[4] B. Babadi, N. Kalouptsidis, and V. Tarokh, "Comparison of SPARLS and RLS algorithms for Adaptive Filtering", 2009 IEEE Sarnoff Symposium, March 30 - April 1, 2009, Princeton, NJ.

[5] Z. Ben-Haim, Y. C. Eldar, and M. Elad, "Near-Oracle Performance of Basis Pursuit Under Random Noise," submitted to IEEE Transactions on Signal Processing, Mar. 2009 (available at http://arxiv.org/abs/0903.4579).

[6] W. Bajwa, J. Haupt, G. Raz and R. Nowak, "Compressed Channel Sensing", in Proc. CISS '08.

[7] J. Haupt, W. Bajwa, G. Raz and R. Nowak, "Toeplitz Compressed Sensing Matrices with Applications to Sparse Channel Estimation", submitted.

[8] A. Bruckstein, D. 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. 3481, 2009.

[9] E. Candès, and T. Tao, "Decoding By Linear Programming", IEEE Trans. on Inf. Theory, Vol. 51, no. 12, pp. 4203-4215, Dec. 2005.

[10] E. Candès, and T. Tao, "The Dantzig selector: Statistical estimation when p is much larger than n", Ann. Statist., pp. 23132351, Dec. 2007.

[11] Y. Chen, Y. Gu, and A. O. Hero III, "Sparse LMS for System Identification", IEEE Int. Conf. Acounstics, Speech, Signal Process. (ICASSP), April 19-24, 2009, Taipei, Taiwan.

[12] Y. S. Chow and T. L. Lai, "Limiting Behavior of Weighted Sums of Independent Random Variables", Ann. Probab. Volume 1, Number 5 (1973), pp. 810-824.

[13] A. P. Dempster, N. M. Laird and D. B. Rubin , "Maximum Likelihood from Incomplete Data via the EM Algorithm", Journal of the Royal Statistical Society. Series B (Methodological), Vol. 39, No. 1. (1977), pp. 1-38.

[14] D. Donoho, "Compressed Sensing", IEEE Trans. on Inf. Theory, Vol. 52, no. 4, pp. 1289-1306, Apr. 2006.

[15] M. Figueirado and R. Nowak, "An EM Algorithm for Wavelet-Based Image Restoration", IEEE Transactions on Image Processing, vol.12, no.8, pp. 906-916, August 2003.

[16] M. Figueiredo, R. Nowak and S. Wright, "Gradient Projection for Sparse Reconstruction: Applications to Compressed Sensing and Other Inverse Problems", IEEE Journal Of Selected Topics In Signal Processing, Vol. 1, NO. 4, Dec. 2007.

[17] S. Haykin, *Adaptive Filter Theory*, 3rd Edition, Prentice Hall, 1996.

[18] J. Haupt, and R. Nowak, "Signal Reconstruction From Noisy Random Projections", IEEE Trans. on Inf. Theory, Vol. 52, No. 9, Sep. 2006.

[19] W. C. Jakes, Editor, *Microwave Mobile Communications*, New York: John Wiley & Sons Inc, 1975.

[20] A. Jazwinski, "Adaptive filtering", Automatica, vol. 5, pp. 475485, 1969.

[21] L. Ljung, "General Structure of Adaptive Algorithms: Adaptation and Tracking", in *Adaptive system identification and signal processing algorithms*, Eds. N. Kalouptsidis and S. Theodoridis, Prentice Hall, Inc., 1993.

[22] J. E. Mazo, "On the Independence Theory of Equalizer Convergence", Bell Syst. Tech. J., Vol. 58, pp. 963-993, 1979.

[23] X.L. Meng and D. B. Rubin, "On the global and componentwise rates of convergence of the EM algorithm", Linear Algebra Appl. v199. 413-425. Schafer, 1997.

[24] S. Pafka, M. Potters, and I. Kondor, "Exponential Weighting and Random-Matrix-Theory-Based Filtering of Financial Covariance Matrices for Portfolio Optimization", Arxiv preprint cond-mat/0402573, 2004 (available at http://arxiv.org/abs/cond-mat/0402573).

[25] C. Robert, *The Bayesian Choice: A Decision Theoretic Motivation*, First Edition, New York, NY, Springer-Verlag, 1994.

[26] R. T. Rockafellar, *Convex Analysis*, Princeton, NJ: Princeton Univ. Press, 1970.

[27] A. M. Sengupta and P. P. Mitra, "Distributions of Singular Values for Some Random Matrices", Physical Review E, Vol. 60, No. 3, 1999.

[28] J. Tropp, "Just Relax: Convex programming methods for identifying sparse signals", IEEE Trans. Info. Theory, vol. 51, num. 3, pp. 1030-1051, Mar. 2006.

[29] H. L. Van Trees, *Detection, Estimation, and Modulation Theory, Part I*, 1st Edition, John Wiley and Sons Inc., 2001.

[30] M. J. Wainwright, "Information-theoretic limits on sparsity recovery in the high-dimensional and noisy setting", Technical report 725, Department of Statistics, UC Berkeley. January 2007 (available at http://arxiv.org/abs/math/0702301).

[31] R. L. Wheeden and A. Zygmund, *Measure and Integral: An Introduction to Real Analysis*, New York: Dekker, 1977.

[32] B. Widrow, J. M. McCool, M. G. Larimore, and C. R. Johnson, Jr., "Stationary and nonstationary learning characteristics of the LMS adaptive filter", Proc. IEEE, vol. 64, pp. 1151-1162, 1976.

Behtash Babadi (S08) received the BSc. degree in Electrical Engineering from Sharif University of Technology, Tehran, Iran and the MSc. in Engineering Sciences from Harvard University, Cambridge, MA, in 2006 and 2008, respectively.

He is currently working towards the PhD degree in Engineering Sciences at the School of Engineering and Applied Sciences, Harvard University, Cambridge, MA. His research interests include dynamic spectrum access networks, adaptive signal processing and compressed sensing.

Nicholas Kalouptsidis (M82-SM85) was born in Athens, Greece, on September 13, 1951. He received the B.Sc. degree in mathematics (with highest honors) from the University of Athens, Athens, Greece, in 1973 and the M.S. and Ph.D. degrees in systems science and mathematics from Washington University, St. Louis, MO, in 1975 and 1976 respectively.

He has held visiting positions at Washington University, St. Louis, MO; Politecnico di Torino, Turin, Italy; Northeastern University, Boston, MA; and CNET Lannion, France. He has been an Associate Professor and Professor with the Department of Physics, University of Athens. In Fall 1998, he was a Clyde Chair Professor with the School of Engineering, University of Utah, Salt Lake City. In Spring 2008, he was a visiting scholar at Harvard university. He is currently a Professor with the Department of Informatics and Telecommunications, University of Athens. He is the author of the textbook *Signal Processing Systems: Theory and Design* (New York: Wiley, 1997) and coeditor, with S. Theodoridis, of the book *Adaptive System Identification and Signal Processing Algorithms* (Englewood Cliffs, NJ: Prentice-Hall, 1993). His research interests are in system theory and signal processing.

Vahid Tarokh (M97-SM02-F09) worked at AT&T Labs-Research until August 2000, where he was the head of the Department of Wireless Communications and Signal Processing. He then joined the Department of Electrical Engineering and Computer Sciences (EECS) at MIT as an associate professor. In 2002, he joined Harvard University as a professor and senior fellow.

Dr. Tarokh's research interest is mainly focused in the areas of information theory, signal processing, communications and networking. He has received a number of awards, and holds 2 honorary degrees.