Sparsity Regularized RLS Adaptive Filtering

Ender M. Eksioglu

Electronics and Communications Engineering Department

Istanbul Technical University

Istanbul, Turkey

e-mail: eksioglue@itu.edu.tr

Abstract

We propose a new approach for the adaptive identification of sparse systems. This approach improves on the Recursive Least Squares (RLS) algorithm by adding a sparsity inducing weighted ℓ_1 norm penalty to the RLS cost function. Subgradient analysis is utilized to develop the recursive update equations for the calculation of the optimum system estimate which minimizes the regularized cost function. Two new algorithms are introduced by considering two different weighting scenarios for the ℓ_1 norm penalty. These new ℓ_1 relaxation based RLS algorithms emphasize sparsity during the adaptive filtering process, and they allow for faster convergence than standard RLS when the system under consideration is sparse. We test the performance of the novel algorithms and compare it to standard RLS and other adaptive algorithms for sparse system identification. Simulations demonstrate that the new algorithms exploit the inherent sparse structure effectively.

I. INTRODUCTION

Sparse representations have acquired considerable interest recently. The sparsity prior tells us that the object to be recovered is known to be structured such that in a certain parameterization its representation is sparse. By sparse, it is meant that the number of significant parameters is much less than the total dimensionality of the object. Sparse estimation can be recast as the following combinatorial optimization problem.

$$\min_{\boldsymbol{h}\in\mathbb{R}^N} \|\boldsymbol{h}\|_0, \text{ such that } \boldsymbol{d} = \Phi\boldsymbol{h} \tag{1}$$

Here $\|(\cdot)\|_0$ denotes the ℓ_0 pseudonorm, which equals the count of the nonzero elements of $\boldsymbol{h}$. The ℓ_0 formulation for the optimization problem is nonconvex. This combinatorial problem is transformed into an easier convex form by replacing the ℓ_0 count by the ℓ_1 norm.

$$\min_{\boldsymbol{h}\in\mathbb{R}^N} \|\boldsymbol{h}\|_1, \text{ such that } \boldsymbol{d} = \Phi\boldsymbol{h} \tag{2}$$

This kind of ℓ_1 relaxation has been utilized in sparse model selection in statistics via the least-absolute shrinkage and selection operator (Lasso) algorithm [1]. Another important application of ℓ_1 relaxation is the Basis Pursuit algorithm, which searches for sparse signal representations in overcomplete dictionaries [2]. The emerging compressive sensing literature dwells on the use of ℓ_1 minimization for the recovery of sparse signals from few linear measurements. These recent applications of ℓ_1 minimization consider batch optimization, where the ℓ_1 minimization is solved for a fixed set of measurements by effective iterative linear programming algorithms. Another popular approach to the batch ℓ_1 minimization has been greedy methods including the Matching Pursuit algorithm. In this paper, rather the online sparse optimization problem is considered, where the solution is updated as new measurements arrive. This streaming data formulation for sparse optimization has been considered in [3] and [4]. Another attempt at online ℓ_1 minimization for sparse representation is proposed in [5], where a homotophy based scheme for dynamical update of the ℓ_1 optimization solution is developed.

Adaptive filtering is an important tool for estimation problems where data becomes available sequentially. There are LMS adaptive algorithms specifically adapted to sparse system identification, such as the Proportionate Normalized LMS (PNLMS) algorithm [6] and the improved Proportionate Normalized LMS (IPNLMS) algorithm. PNLMS approach is more successful than the plain LMS algorithm in the context of sparse system identification. The proportionate

updating idea is based on utilizing an adaptation gain proportional to its magnitude for each coefficient. These algorithms do not utilize the sparse estimation concepts based on ℓ_0 or ℓ_1 norms. Recently, Least Mean Squares (LMS) variants specifically designed for sparse system identification have been proposed. These adaptive algorithms utilize the novel results from the sparse reconstruction literature and incorporate the sparsity condition directly into the cost function via a sparsity inducing penalty term. First such attempt was given in [7]. Here, the authors employ the ℓ_p, $0 < p \leqslant 1$ norm of the weight vector as a regularization term for diversity minimization. In [8], the LMS cost function is modified by adding an ℓ_0 norm penalty. Subsequently the ℓ_0 norm is replaced by an analytic approximation to it, and the minimization problem for the cost function becomes tractable. The authors call the resulting adaptive algorithm as the ℓ_0-LMS. In [9], both an ℓ_1 norm term and a log-sum term are considered as the penalty to be added to the regular LMS cost function. The log-sum penalty was utilized in [10] in connection with reweighted ℓ_1 minimization, and the log-sum penalty was shown to be more sparsity-inducing then the plain ℓ_1 norm penalty. Analysis conducted in [9] results in the Zero-Attracting LMS (ZA-LMS) and the Reweighted Zero-Attracting LMS (RZA-LMS) algorithms. One other attempt at online sparse system identification is given in [11], where a projection based algorithm is utilized, rather than sparse regularizing a cost function.

Recursive least squares (RLS) adaptive algorithms constitute another important class of adaptive algorithms. However, contrary to the comparatively long list of LMS based sparse algorithms listed above, there have been few RLS based algorithms designed for sparse signal estimation and specifically for sparse system identification. [12] introduces an online version of the batch least-squares based MOD algorithm for dictionary identification. Dictionary identification considers the construction of a suitable dictionary for sparse representation from data, and it is a problem distinct from sparse system identification or sparse signal estimation. [4] considers a weighted recursive Lasso algorithm for sparse signal estimation. [3] proposes time-weighted (TW) and time- and norm-weighted (TNW) Lasso approaches for recursive, real-time sparse signal estimation. The solution to the ℓ_1-norm regularized least-squares cost function is calculated using an online coordinate descent algorithm [3]. Another sparsity based adaptive algorithm is presented in [13]. Here, the wavelet based Expectation-Maximization (EM) approach for image restoration in [14] is adapted to the sparse system identification setting. The resulting recursive ℓ_1-regularized least squares algorithm is denoted as SPARLS and its performance in sparse system identification is

compared to regular RLS. The SPARLS algorithm is algorithmically quite involved compared to the plain RLS algorithm. SPARLS update equations are not intuitively related to RLS, whereas the sparsity regularized LMS algorithms such as ℓ_0-LMS in [8] and RZA-LMS, ZA-LMS pair in [9] are quite similar to the standard LMS algorithm. In this paper, we regularize the RLS algorithm in a way comparable to the methods applied in [8] and [9]. We add a regularizing penalty term to the RLS cost function, and we minimize this regularized cost function with respect to the system impulse response estimate using subdifferential calculus, since the regularized cost function is convex but not differentiable with respect to the filter tap weights. The sparsifying penalty is taken to be the ℓ_1 norm weighted by a general weighting matrix. The ensuing adaptive algorithm is in a form akin to the conventional RLS scheme. We call this novel algorithm with general weighting as the Weighted ℓ_1-RLS (ℓ_1-WRLS) algorithm. Considering different weighting matrices results in different adaptive update equations for the system impulse response estimate. The simplest weighting scheme is the use of no weighting at all. The use of the non-weighted ℓ_1 norm culminates in the ℓ_1-RLS algorithm. The development of the ℓ_1-RLS algorithm with preliminary results was previously presented in [15] as a brief precursor to this paper. As a second, more advanced weighting method we utilize the reweighting as given in [10], where reweighted ℓ_1 minimization was proposed. This weighting scheme results in an adaptive algorithm which we denote as the Reweighted ℓ_1-RLS (ℓ_1-RRLS).

II. Weighted $\boldsymbol{\ell_1}$-RLS Algorithm

We consider the input-output system identification setting given by the following relation.

$$y(n) = \boldsymbol{h}^T \boldsymbol{x}(n) + \eta(n) \tag{3}$$

Here, the $N \times 1$ vectors $\boldsymbol{h} = [h_0,\ h_1, \ldots, h_{N-1}]^T \in \mathbb{R}^N$ and $\boldsymbol{x}(n) = [x(n), x(n-1), \ldots x(n-N+1)]^T \in \mathbb{R}^N$ are the system tap weight vector and the input signal vector, respectively. $y(n)$ is the output signal, $x(n)$ is the input signal and $\eta(n)$ denotes the observation noise. Adaptive system identification methods seek to estimate the system parameter vector $\boldsymbol{h}$ from the input and output signals in a sequential manner. We denote the estimate for the system tap weight vector at time n as $\boldsymbol{h}(n) = [h_0(n),\ h_1(n), \ldots, h_{N-1}(n)]^T$. The definition for the standard recursive least squares cost is

$$\mathcal{E}(n) = \sum_{m=0}^{n} \lambda^{n-m} |e(m)|^2, \tag{4}$$

where λ is the exponential forgetting factor. $e(n)$ is the instantaneous error term given by

$$e(n) = y(n) - \boldsymbol{h}^T(n)\boldsymbol{x}(n). \tag{5}$$

In this paper, we pursue an adaptive estimation algorithm for the case when the underlying system coefficient vector $\boldsymbol{h}$ has a sparse form. The system vector is said to be sparse when $\|\boldsymbol{h}\|_0 \ll N$. This definition of sparsity tells us that the number of strictly non-zero terms in the system vector are much less than the vector dimension. Other definitions of sparsity are possible where the number of strictly non-zero terms is replaced with the number of significant terms. We seek to modify the RLS cost function in a manner that emphasizes this sparsity assumption. As introduced in [10] and exemplified by the RZA-LMS algorithm in [9], a tractable way to force sparsity is by using the weighted ℓ_1-norm of the parameter vector estimate. Hence, we regularize the RLS cost function in (4) by adding the weighted ℓ_1 norm of the current tap vector estimate to it.

$$J(n) = \frac{1}{2}\mathcal{E}(n) + \gamma\|\mathbf{W}\boldsymbol{h}(n)\|_1 \tag{6}$$

$\|\mathbf{W}\boldsymbol{h}(n)\|_1$ stands for the weighted ℓ_1 norm of the tab vector estimate.

$$\|\mathbf{W}\boldsymbol{h}(n)\|_1 = \sum_{k=0}^{N-1} w_k|h_k(n)| \tag{7}$$

Here, w_k, $k = 0, 1, 2, \ldots, N-1$ are positive valued weighting parameters. $\mathbf{W}$ denotes the $N \times N$ weighting matrix, which is a diagonal matrix with the w_k values on the main diagonal. The parameter γ governs the compromise between the sparsity of the system estimate and the estimation error. The aim of an adaptive algorithm is to find the system coefficient vector which minimizes the regularized cost function $J(n)$. Let $\widehat{\boldsymbol{h}}(n)$ denote the optimal estimate for the tab vector which minimizes the cost function in (6). For the standard RLS case the cost function under consideration is simply $\mathcal{E}(n)$. The condition for the optimum $\widehat{\boldsymbol{h}}(n)$ which minimizes $\mathcal{E}(n)$ is written in terms of the gradient of $\mathcal{E}(n)$ with respect to $\boldsymbol{h}(n)$ [16].

$$\nabla\mathcal{E}(n)\,|_{\widehat{\boldsymbol{h}}(n)} = 2\frac{\partial\mathcal{E}(n)}{\partial\boldsymbol{h}^*(n)}\Big|_{\widehat{\boldsymbol{h}}(n)} = \mathbf{0} \tag{8}$$

However, when we consider the sparsity regularized cost fuction $J(n)$ in (6), the ℓ_1 norm term $\|\mathbf{W}\boldsymbol{h}(n)\|_1$ is nondifferentiable at any point where $h_k(n) = 0$. Hence, the gradient $\nabla J(n)$ does not exist at any point where $h_k(n) = 0$. A substitute for the gradient in the case of nondifferentiable convex functions such as $\|\mathbf{W}\boldsymbol{h}(n)\|_1$ here is offered by the definition of the

subgradient [17, p. 227]. Let $f(\boldsymbol{\varphi}) : \mathbb{R}^N \to \mathbb{R}$ denote a convex function. At any point where this function is not differentiable there may exist many valid subgradient vectors. The set of all the subgradients for the convex function $f(\boldsymbol{\varphi})$ is called as the subdifferential of $f(\boldsymbol{\varphi})$. The subdifferential is denoted by $\partial f(\boldsymbol{\varphi})$. The subdifferential for $\|\mathbf{W}\boldsymbol{h}(n)\|_1$ can be calculated using results from subdifferential calculus. Consider the case when the convex function $f(\boldsymbol{\varphi})$ can be written as the pointwise maximum of a set of differentiable and convex functions $\phi(\boldsymbol{\varphi}, z)$, $z \in \mathbb{Z}$.

$$f(\boldsymbol{\varphi}) = \max_{z \in \mathbb{Z}} \phi(\boldsymbol{\varphi}, z) \tag{9}$$

The subdifferential at any point for $f(\boldsymbol{\varphi})$ as defined in (9) is calculated by forming the convex hull of the union of the differentials of the functions achieving the maximum at this point [17, p.245]. This result can be written in the following form.

$$\partial f(\boldsymbol{\varphi}) = \operatorname{conv}\Big\{\nabla\phi(\boldsymbol{\varphi}, z) \mid \phi(\boldsymbol{\varphi}, z) = f(\boldsymbol{\varphi})\Big\} \tag{10}$$

The ℓ_1 norm function, that is $f(\boldsymbol{\varphi}) = \|\boldsymbol{\varphi}\|_1$ can be expressed as the maximum of a total of 2^N linear functions.

$$\|\boldsymbol{\varphi}\|_1 = \max\Big\{\boldsymbol{s}^T\boldsymbol{\varphi} \mid \boldsymbol{s} \in \mathbb{R}^N, \boldsymbol{s}(i) \in \{\pm 1\}\Big\} \tag{11}$$

When we apply the result in (10) onto (11), it follows that the subdifferential for $\|\boldsymbol{\varphi}\|_1$ is calculated as below.

$$\partial\|\boldsymbol{\varphi}\|_1 = \Big\{\boldsymbol{d} \mid \|\boldsymbol{d}\|_\infty \leqslant 1, \boldsymbol{d}^T\boldsymbol{\varphi} = \|\boldsymbol{\varphi}\|_1\Big\} \tag{12}$$

Hence, the k^{th} element of the subdifferential for $\|\boldsymbol{\varphi}\|_1$ can be written in the below form.

$$\Big\{\partial\|\boldsymbol{\varphi}\|_1\Big\}_k = \begin{cases} \big\{\varphi_k/|\varphi_k|\big\} & \varphi_k \neq 0 \\ \big\{d \mid |d| \leqslant 1\big\} & \varphi_k = 0 \end{cases} \tag{13}$$

For points with no zero element, that is when $\varphi_k \neq 0\ \forall k = 0, \ldots, N-1$, the subdifferential is a single vector. For any point with some $\varphi_k = 0$, there is a valid subgradient vector with its k^{th} entry equal to zero, since $|d| = 0$ is allowed on the second line of (13). Using these results we can state that one valid subgradient vector for $\|\boldsymbol{\varphi}\|_1$ is as given below.

$$\nabla^S\|\boldsymbol{\varphi}\|_1 = \operatorname{sgn}\big(\boldsymbol{\varphi}\big) \tag{14}$$

We utilize $\nabla^S f(\boldsymbol{\varphi})$ to denote a subgradient vector of the function $f(\boldsymbol{\varphi})$. A subgradient vector is an element of the subdifferential set, hence $\nabla^S f(\boldsymbol{\varphi}) \in \partial f(\boldsymbol{\varphi})$. $\operatorname{sgn}(\cdot)$, acting possibly on a

vector, denotes the componentwise sign function.

$$\Big\{\mathrm{sgn}\big(\boldsymbol{\varphi}\big)\Big\}_k = \begin{cases} \varphi_k/|\varphi_k| & \varphi_k \neq 0 \\ 0 & \varphi_k = 0 \end{cases} \tag{15}$$

Using (14) and the chain rule for subdifferential of an affine transformation of a convex function [17, p.233], one valid subgradient vector for $\|\mathbf{W}\boldsymbol{h}(n)\|_1$ can be written as follows.

$$\nabla^S \|\mathbf{W}\boldsymbol{h}(n)\|_1 = \mathbf{W}^T \mathrm{sgn}\big(\mathbf{W}\boldsymbol{h}(n)\big) \tag{16}$$

Assuming that the weighting matrix $\mathbf{W}$ is a diagonal matrix with positive entries, the subgradient becomes

$$\nabla^S \|\mathbf{W}\boldsymbol{h}(n)\|_1 = \mathbf{W}\mathrm{sgn}\big(\boldsymbol{h}(n)\big). \tag{17}$$

Accordingly, one valid subgradient vector of the penalized cost function $J(n)$ in (6) with respect to the weight vector $\boldsymbol{h}(n)$ is written by using (17) and the fact that $\mathcal{E}(n)$ is differentiable everywhere.

$$\nabla^S J(n) = \frac{1}{2}\nabla\mathcal{E} + \gamma\mathbf{W}\mathrm{sgn}\big(\boldsymbol{h}(n)\big) \tag{18}$$

The i^{th} element of this subgradient vector is calculated as below [16].

$$\Big\{\nabla^S J(n)\Big\}_i = -\sum_{m=0}^{n} \lambda^{n-m} e(m)x(m-i) + \gamma w_i \mathrm{sgn}\big(h_i(n)\big) \tag{19}$$

Another result from the subdifferential calculus is about the argument value which minimizes a convex function. This result states that a point $\widehat{\boldsymbol{\varphi}}$ is a minimizer of a convex function $f(\boldsymbol{\varphi})$ if and only if $\mathbf{0} \in \partial f(\widehat{\boldsymbol{\varphi}})$ [17, p.257]. This means that for $\widehat{\boldsymbol{\varphi}}$ to be a minimizer, $\boldsymbol{d} = \mathbf{0}$ should be a subgradient of f at $\widehat{\boldsymbol{\varphi}}$. This result suggests that we set the subgradient term in (19) equal to zero to find the optimal least squares solution, namely $\widehat{\boldsymbol{h}}(n)$ which minimizes $J(n)$.

$$\sum_{m=0}^{n} \lambda^{n-m}\Big\{y(m) - \sum_{k=0}^{N-1} \hat{h}_k(n)x(m-k)\Big\}x(m-i) = \gamma w_i \mathrm{sgn}\big(\hat{h}_i(n)\big) \tag{20}$$

After some manipulation (20) assumes the form given below.

$$\sum_{k=0}^{N-1} \hat{h}_k(n)\Big\{\sum_{m=0}^{n} \lambda^{n-m}x(m-k)x(m-i)\Big\} = \sum_{m=0}^{n} \lambda^{n-m}y(m)x(m-i) - \gamma w_i \ \mathrm{sgn}\big(\hat{h}_i(n)\big) \tag{21}$$

(21) can be written for all $i = 0, \ldots, N-1$ together in a matrix form. Vectorizing (21) results in the following matrix equation, which we call as the modified deterministic normal equations.

$$\boldsymbol{\Phi}(n)\widehat{\boldsymbol{h}}(n) = \boldsymbol{r}(n) - \gamma\mathbf{W}\mathrm{sgn}\big(\widehat{\boldsymbol{h}}(n)\big) \tag{22}$$

In (22), $\boldsymbol{\Phi}(n)$ is the $N \times N$ exponentially weighted deterministic autocorrelation matrix estimate for the input signal $x(n)$ [16].

$$\boldsymbol{\Phi}(n) = \sum_{m=0}^{n} \lambda^{n-m} \boldsymbol{x}(m) \boldsymbol{x}^T(m) \tag{23}$$

Similarly, $\boldsymbol{r}(n)$ defines the $N \times 1$ exponentially weighted deterministic cross-correlation estimate vector between the output signal $y(n)$ and $x(n)$.

$$\boldsymbol{r}(n) = \sum_{m=0}^{n} \lambda^{n-m} y(m) \boldsymbol{x}(m) \tag{24}$$

$\mathbf{W}$ in (22) is again the weighting matrix utilized in the weighted ℓ_1 norm penalty of (7). We define a new variable $\boldsymbol{\theta}(n)$ for the right hand side of (22).

$$\boldsymbol{\theta}(n) = \boldsymbol{r}(n) - \gamma \mathbf{W} \mathrm{sgn}\big(\widehat{\boldsymbol{h}}(n)\big) \tag{25}$$

With the introduction of $\boldsymbol{\theta}(n)$ the normal equation (22) transforms into a simpler form.

$$\boldsymbol{\Phi}(n) \widehat{\boldsymbol{h}}(n) = \boldsymbol{\theta}(n) \tag{26}$$

On a par with the development of standard RLS algorithm, the autocorrelation and cross-correlation estimates have corresponding rank-one update equations pertaining to them.

$$\boldsymbol{\Phi}(n) = \lambda \boldsymbol{\Phi}(n-1) + \boldsymbol{x}(n) \boldsymbol{x}^T(n) \tag{27}$$

$$\boldsymbol{r}(n) = \lambda \boldsymbol{r}(n-1) + y(n) \boldsymbol{x}(n) \tag{28}$$

The $\boldsymbol{\theta}(n)$ also has a recursive update equation considering its definition (25) and (28). The update equation for $\boldsymbol{\theta}(n)$ is calculated as

$$\boldsymbol{\theta}(n) = \lambda \boldsymbol{\theta}(n-1) + y(n) \boldsymbol{x}(n) - \Big\{ \gamma \mathbf{W} \mathrm{sgn}\big(\widehat{\boldsymbol{h}}(n)\big) - \lambda \gamma \mathbf{W} \mathrm{sgn}\big(\widehat{\boldsymbol{h}}(n-1)\big) \Big\}. \tag{29}$$

In a similar vein to the conventional RLS paradigm, we desire a gradual iterative procedure for finding the optimal least squares solution, instead of solving the modified normal equations (26) directly for $\widehat{\boldsymbol{h}}(n)$. The iterative solution should have the following structure.

$$\widehat{\boldsymbol{h}}(n) = \widehat{\boldsymbol{h}}(n-1) + \boldsymbol{\Delta} \widehat{\boldsymbol{h}}(n-1). \tag{30}$$

Here, $\boldsymbol{\Delta} \widehat{\boldsymbol{h}}(n-1)$ is an instantaneous corrective step applied to the estimate vector, and it is written as a function depending on the prior estimate $\widehat{\boldsymbol{h}}(n-1)$. To reach an update equation in the form of (30), we need to convert (29) into a recursion with only the prior estimate $\widehat{\boldsymbol{h}}(n-1)$

terms on the right hand side. We assume that the signs of the tab estimate values do not change significantly in a single time step, hence $\mathrm{sgn}\big(\widehat{\boldsymbol{h}}(n)\big) \approx \mathrm{sgn}\big(\widehat{\boldsymbol{h}}(n-1)\big)$. Therefore, we approximate (29) with

$$\boldsymbol{\theta}(n) \approx \lambda\boldsymbol{\theta}(n-1) + y(n)\boldsymbol{x}(n) + \gamma(\lambda-1)\mathbf{W}\mathrm{sgn}\big(\widehat{\boldsymbol{h}}(n-1)\big). \tag{31}$$

The inverse of the autocorrelation matrix is given a specific name to further the analysis.

$$\mathbf{P}(n) = \boldsymbol{\Phi}^{-1}(n) \tag{32}$$

Using the matrix inversion lemma on (27), the recursive time update for the correlation matrix estimate inverse $\mathbf{P}(n)$ is performed using the Riccati equation for the RLS algorithm.

$$\mathbf{P}(n) = \lambda^{-1}\Big\{\mathbf{P}(n-1) - \boldsymbol{k}(n)\boldsymbol{x}^T(n)\mathbf{P}(n-1)\Big\} \tag{33}$$

Here, $\boldsymbol{k}(n)$ is the so-called gain vector defined as follows.

$$\boldsymbol{k}(n) = \frac{\mathbf{P}(n-1)\boldsymbol{x}(n)}{\lambda + \boldsymbol{x}^T(n)\mathbf{P}(n-1)\boldsymbol{x}(n)} \tag{34}$$

With the advent of the $\mathbf{P}(n)$, the normal equation (26) becomes

$$\widehat{\boldsymbol{h}}(n) = \mathbf{P}(n)\boldsymbol{\theta}(n). \tag{35}$$

When we insert the recursions (27) and (31) into (35), the update for the tab estimate assumes the following structure.

$$\widehat{\boldsymbol{h}}(n) = \mathbf{P}(n-1)\boldsymbol{\theta}(n-1) - \boldsymbol{k}(n)\boldsymbol{x}^T(n)\mathbf{P}(n-1)\boldsymbol{\theta}(n-1) + y(n)\boldsymbol{k}(n) + \gamma\Big(\frac{\lambda-1}{\lambda}\Big)\times \Big\{\mathbf{P}(n-1)\mathbf{W}\mathrm{sgn}\big(\widehat{\boldsymbol{h}}(n-1)\big) - \boldsymbol{k}(n)\boldsymbol{x}^T(n)\mathbf{P}(n-1)\mathbf{W}\mathrm{sgn}\big(\widehat{\boldsymbol{h}}(n-1)\big)\Big\} \tag{36}$$

Apprehending that $\widehat{\boldsymbol{h}}(n-1) = \mathbf{P}(n-1)\boldsymbol{\theta}(n-1)$ from (35), the recursive update equation for the tab vector estimate becomes as follows.

$$\widehat{\boldsymbol{h}}(n) = \widehat{\boldsymbol{h}}(n-1) + \boldsymbol{k}(n)\Big\{y(n) - \widehat{\boldsymbol{h}}^T(n-1)\boldsymbol{x}(n)\Big\} + \gamma\Big(\frac{\lambda-1}{\lambda}\Big)\Big\{\mathbf{I}_N - \boldsymbol{k}(n)\boldsymbol{x}^T(n)\Big\}\times \mathbf{P}(n-1)\mathbf{W}\mathrm{sgn}\big(\widehat{\boldsymbol{h}}(n-1)\big) \tag{37}$$

Here, $\mathbf{I}_N$ is the $N \times N$ identity matrix. The update equation (37) finalizes the adaptive algorithm for the estimation of the sparse system tab vector. We call this novel adaptive sparsity based algorithm as the "Weighted ℓ_1-RLS" (ℓ_1-WRLS). When we compare ℓ_1-WRLS with the regular RLS algorithm, we see that the main difference occurs in the update equation for $\widehat{\boldsymbol{h}}(n)$, that is in (37). The last term in (37) starting with $\gamma\Big(\frac{\lambda-1}{\lambda}\Big)$ constitutes the difference from the regular RLS. If we set $\lambda = 1$ or $\gamma = 0$, the ℓ_1-WRLS algorithm reduces to regular RLS.

III. CHOOSING THE WEIGHTS: $\boldsymbol{\ell_1}$-RRLS ALGORITHM

After the development of the ℓ_1-WRLS, we are confronted with deciding on the weighting matrix $\mathbf{W}$. One obvious choice for the weighting matrix is using the identity matrix $\mathbf{W} = \mathbf{I}_N$. In this case the penalty term in (7) equals the straight ℓ_1 norm, $\|\boldsymbol{h}(n)\|_1 = \sum_{k=0}^{N-1}|h_k(n)|$. We will denote the resulting algorithm as ℓ_1-RLS, as this is a non-weighted special case of the ℓ_1-WRLS.

An intelligent way to chose the weights is to aim at making the weighted ℓ_1 norm have values as similar as possible to the ℓ_0 norm, as ℓ_0 norm is the true measure of sparseness. The weighted ℓ_1 norm value becomes similar to ℓ_0 by choosing the weights inversely proportional to the magnitude of the actual tab values of the system under consideration. Hence, the weights are to be chosen as given below.

$$w_k = \begin{cases} \dfrac{1}{|h_k|}, & h_k \neq 0 \\ \infty, & h_k = 0 \end{cases} \tag{38}$$

However, the true system tab values are the unknowns the adaptive system strives to infer. Therefore, we utilize the current adaptive tab estimate inverses as the weighting values. Hence, the time-varying weights become as follows.

$$w_k(n) = \frac{1}{|h_k(n-1)| + \epsilon} \tag{39}$$

The resulting weighting matrix $\mathbf{W}(n)$ is a diagonal matrix with the $w_k(n)$ values from (39) on the diagonal. The parameter $\epsilon > 0$ in (39) is included in the denominator to enhance stability in the case of a zero-valued instantaneous tab estimate. In [10], it is demonstrated that values slightly less than the magnitude of the actual nonzero system tab weights are proper choices for ϵ. By the insertion of the instantaneous weight values in (39) into the tab estimate update equation (37), the resulting weighted update equation is written as follows.

$$\widehat{\boldsymbol{h}}(n) = \widehat{\boldsymbol{h}}(n-1) + \boldsymbol{k}(n)\Big\{y(n) - \widehat{\boldsymbol{h}}^T(n-1)\boldsymbol{x}(n)\Big\} + \gamma\Big(\frac{\lambda-1}{\lambda}\Big)\Big\{\mathbf{I}_N - \boldsymbol{k}(n)\boldsymbol{x}^T(n)\Big\}\times \mathbf{P}(n-1)\frac{\operatorname{sgn}\big(\widehat{\boldsymbol{h}}(n-1)\big)}{|\widehat{\boldsymbol{h}}(n-1)| + \epsilon} \tag{40}$$

The vector division operation in this equation denotes a simple componentwise division. We will refer to the resulting adaptive algorithm as ℓ_1-Reweighted RLS (ℓ_1-RRLS), to underline the connection to the reweighted ℓ_1 minimization approach as introduced in [10]. The ℓ_1-RRLS

algorithm can also be developed using a log-sum penalty term instead of the weighted ℓ_1 norm penalty in (6). The log-sum penalty is calculated as follows.

$$\sum_{k=0}^{N-1} \log(|h_k(n)| + \epsilon) \tag{41}$$

If we replace the ℓ_1 norm penalty term $\|\mathbf{W}\boldsymbol{h}(n)\|_1$ in (6) with the log-sum cost given in (41) and than do the minimization analysis, the resulting adaptive algorithm is the ℓ_1-RRLS algorithm. As stated in [10], establishing a connection with the log-sum penalty is important. Utilizing the log-sum cost term as a penalty is potentially more sparsity-inducing than the simple ℓ_1 norm [10]. Hence, we can develop the ℓ_1-RRLS algorithm via two different approaches. One approach is to utilize a diagonal weighting matrix $\mathbf{W}(n)$ constructed from the weight values (39), in the ℓ_1-WRLS update equation (37). A second approach is to employ a log-sum term as the sparsity-inducing penalty in (6) and than do the minimization analysis using subgradients. The complete ℓ_1-RRLS algorithm is outlined in Algorithm 1.

The general ℓ_1-WRLS algorithm is obtained simply by replacing the tab vector update step (step 6) in Algorithm 1 with the general update equation given in (37). The non-weighted ℓ_1-RLS algorithm is obtained by setting $\mathbf{W} = \mathbf{I}_N$.

Algorithm 1 ℓ_1-Reweighted RLS (ℓ_1-RRLS) algorithm.

$\lambda,\ \gamma,\ \epsilon,\ x(n),\ y(n)$ ▷ inputs

$\boldsymbol{h}(-1) = \mathbf{0}, \quad \mathbf{P}(-1) = \delta^{-1}\mathbf{I}_N$ ▷ initial values

1: **for** $n := 0, 1, 2, \ldots$ **do** ▷ time recursion

2: $\quad \boldsymbol{k}_\lambda(n) = \mathbf{P}(n-1)\boldsymbol{x}(n)$

3: $\quad \boldsymbol{k}(n) = \dfrac{\boldsymbol{k}_\lambda(n)}{\lambda + \boldsymbol{x}^T(n)\boldsymbol{k}_\lambda(n)}$

4: $\quad \xi(n) = y(n) - \boldsymbol{h}^T(n-1)\boldsymbol{x}(n)$

5: $\quad \mathbf{P}(n) = \dfrac{1}{\lambda}\Big[\mathbf{P}(n-1) - \boldsymbol{k}(n)\boldsymbol{k}_\lambda^T(n)\Big]$

6: $\quad \boldsymbol{h}(n) = \boldsymbol{h}(n-1) + \boldsymbol{k}(n)\xi(n) + \gamma\Big(\dfrac{\lambda-1}{\lambda}\Big)\Big\{\mathbf{I}_N - \boldsymbol{k}(n)\boldsymbol{x}^T(n)\Big\}\mathbf{P}(n-1)\dfrac{\operatorname{sgn}\big(\boldsymbol{h}(n-1)\big)}{|\boldsymbol{h}(n-1)| + \epsilon}$

7: **end for** ▷ end of recursion

IV. Comparison with other online algorithms

The developed ℓ_1-WRLS algorithm presents a method for sparse system identification which is intuitively related to the standard RLS approach. [4] also considers the weighted, ℓ_1-regularized least-squares cost function as presented in (6). The authors employ a subgradient-based iterative minimizing approach, where the iterations simply update the estimate in the direction of the current subgradient iterate. Since the update is simply a possibly time-varying constant times the subgradient iterate, the algorithm as presented in [4] is a an LMS-like first order algorithm with relatively slow convergence as stated in [3]. ℓ_1-WRLS presented in this paper on the other hand, employs a different minimization approach which results in the second-order update equation (37).

[3] proposes TW and TNW Lasso approaches for real-time sparse signal estimation, again starting with the possibly weighted ℓ_1-regularized least-squares cost function similar to the one presented in (6). The online minimization for this cost function is realized by solving a succession of convex optimization problems. A convex optimization problem is solved for each measurement value. A simplified version of these algorithms which employs minimization with respect to only one coordinate per iteration cycle is also developed, and this is called as the Online Coordinate Descent (OCD) algorithm. In ℓ_1-WRLS, all coordinates of the estimate vector are updated at every iteration cycle without resorting to solving a full convex program per measurement.

[13] introduces the SPARLS algorithm for online sparse system identification. The problem is again formulated as the convex program of minimizing the ℓ_1-regularized least-squares cost function. The optimization at each time point is reformulated as a maximum-likelihood (ML) problem which is solved by an iterative EM algorithm. Hence, for each measurement value an iterative EM algorithm is run to completion. The update step of ℓ_1-WRLS requires no iterative algorithm per time index and is realized simply by (37). The SPARLS algorithm has a computational complexity of $\mathcal{O}(N^2)$ multiplications per time step [13]. The ℓ_1-WRLS algorithm has this same general complexity, which coincides with the multiplicational complexity of the standard RLS algorithm. ℓ_1-WRLS distinguishes from the standard RLS algorithm only in the last term of the final update step. Hence, it has the same $\mathcal{O}(N^2)$ complexity as the RLS algorithm.

V. Simulation Results

In this section, we compare the performance of the novel ℓ_1-WRLS algorithms to the regular RLS, regular LMS and other sparsity oriented adaptive algorithm. The first experiment considers the tracking capabilities of ℓ_1-RRLS, ℓ_1-RLS, RLS, RZA-LMS [9], ZA-LMS [9], ℓ_0-LMS [8] and LMS. The sparse system to be identified has a total of 64 tabs and 4 of them are nonzero. The positions and amplitudes of the nonzero tab weights are chosen randomly. Input $x(n)$ is assumed to be white and AWGN observation noise is added to the system output with an SNR value of 20 dB. The parameters for the different algorithm are chosen as below:

- ℓ_1-RRLS, ℓ_1-RLS, and RLS: $\lambda = 0.99$
- ℓ_1-RRLS: $\gamma = 1.2$, $\epsilon = 0.1$
- ℓ_1-RLS: $\gamma = 3$
- RZA-LMS, ZA-LMS, ℓ_0-LMS and LMS: $\mu = 0.008$
- RZA-LMS: $\rho = 8 \times 10^{-4}$, $\sigma = 10$
- ZA-LMS: $\rho = 3 \times 10^{-4}$, $\sigma = 10$
- ℓ_0-LMS: $\kappa = 2 \times 10^{-4}$, $\beta = 5$, $Q = 1$

The λ parameter for the RLS algorithm and the μ parameter for the LMS algorithm are chosen as to result in roughly equal steady-state error values for the standard RLS and LMS algorithms. The RLS and LMS variants utilize these same λ and μ values. The remaining parameters are found by repeated trials as to cause the minimum steady-state error for their respective adaptive algorithm. The normalized mean square deviation (MSD) of the system impulse response estimates versus time iteration index are plotted in Fig. 1. The normalized MSD is defined as

$$\text{MSD} = \frac{\mathbb{E}\{\|\widehat{\boldsymbol{h}} - \boldsymbol{h}\|_2^2\}}{\mathbb{E}\{\|\boldsymbol{h}\|_2^2\}} \tag{42}$$

The MSD's for all the algorithms are averaged over a total of 500 runs. The ℓ_0-LMS and RZA-LMS algorithms have almost equivalent performance, which is to be expected from the similarity between their respective tab estimate update equations [8], [9]. They have better convergence than ZA-LMS. ℓ_1-RRLS and ℓ_1-RLS present convergence and steady-state error improvements over the regular RLS algorithm, just as RZA-LMS, ℓ_0-LMS and ZA-LMS work better than the regular LMS algorithm. It is interesting to note that the steady-state error for the ℓ_1-RRLS coincides with the RZA-LMS, ℓ_0-LMS pair, and the steady-state error for the ℓ_1-RLS algorithm coincides with the ZA-LMS algorithm. The novel ℓ_1-WRLS algorithms also maintain the faster convergence

of RLS over LMS. Hence, the ℓ_1-WRLS algorithms do indeed present RLS counterparts to the sparsity based LMS algorithms of [8] and [9].

The second experiment dwells on the effect of sparsity on ℓ_1-RRLS and RLS performance. The resulting learning curves for this experiment are presented in Fig.2. The system to be identified is assumed to have a total of 64 coefficients. The number of nonzero coefficients varies from 4 to 64, hence finally attaining a totally non-sparse system. As in the first experiment, the positions and amplitudes of the nonzero tabs are random variables. AWGN observation noise with an SNR of 20 dB is present. The algorithm parameters stay the same as in the first experiment, except varying γ values are used for different sparsity levels.

- ℓ_1-RRLS, and RLS: $\lambda = 0.99$
- ℓ_1-RRLS: $\gamma = [1.2\ 1.5\ 2.0\ 2.5\ 3]$, $\epsilon = 0.1$

The results presented in Fig.2 demonstrate that RLS performance is independent from the system sparsity. On the other hand, ℓ_1-RRLS steady-state error performance degrades with a decline in sparsity. The ℓ_1-RRLS steady-state error is the least for the most sparse system with only 4 nonzero tabs. The steady-state error gradually increases as the number of nonzero terms increases. Finally for a nonsparse system with 64 nonzero terms, the performance curves of ℓ_1-RRLS and straight RLS coincide.

In the third experiment, we analyze the effect of the choice for γ on the ℓ_1-RRLS performance. The system to be identified has a total of 64 coefficients where 4 are nonzero, and SNR is 20 dB. The γ parameter changes from 0 up to 3.5 in steps of 0.5. ℓ_1-RRLS with $\gamma = 0$ corresponds to the standard RLS algorithm. As seen in Fig.3, the steady-state error makes a dip around $\gamma = 1$. However, the ℓ_1-RRLS performance is not overly sensitive to the γ value.

The fourth experiment compares the performance of the ℓ_1-RRLS algorithm to RLS under different SNR values. The underlying system has again impulse response length of 64 with 4 nonzero tabs. The learning curves for SNR values of 10, 20, 30 and 40 dB are presented in Fig.4. The λ value and the γ parameter for ℓ_1-RRLS chosen as follows:

- ℓ_1-RRLS, ℓ_1-RLS and RLS: $\lambda = 0.99$
- ℓ_1-RRLS: $\gamma = 3.5$ for 10 dB, $\gamma = 1.2$ for 20 dB, $\gamma = 0.3$ for 30 dB, $\gamma = 0.1$ for 40 dB
- ℓ_1-RLS: $\gamma = 5$ for 10 dB, $\gamma = 3$ for 20 dB, $\gamma = 0.5$ for 30 dB, $\gamma = 0.3$ for 40 dB

TABLE I

OPTIMAL VALUES OF γ VERSUS NOISE VARIANCE.

σ^2	ℓ_1-RRLS	ℓ_1-RLS	SPARLS
0.0001	0.02	0.06	100
0.0005	0.04	0.08	50
0.001	0.08	0.15	35
0.005	0.12	0.3	15
0.01	0.18	0.5	13
0.05	0.35	1	3

These γ values are found by repeated trials as to minimize the corresponding steady-state error value. As can be inferred from Fig.4, ℓ_1-RRLS and ℓ_1-RLS have better convergence and steady-state properties than the regular RLS for all SNR values.

As a final experiment, we compare the performance of ℓ_1-WRLS to another recently proposed adaptive sparse system identification algorithm, namely SPARLS [13]. [1] For this experiment we repeat the experimental setup as described in [13] in the time-invariant scenario ($f_d = 0$). We realize ℓ_1-RRLS, ℓ_1-RLS, regular RLS and SPARLS. The input data is Gaussian distributed with length 500, and the sparse system to be identified has a total of 100 tabs where 5 of them are nonzero. The simulation results are averaged over 50 trials. For RLS $\lambda = 1$, and for the other algorithms $\lambda = 0.999$. The optimal γ values for the SPARLS are taken from [13], and the γ values for ℓ_1-RRLS and ℓ_1-RLS are obtained via repeated simulations. The γ values utilized for different noise variance levels σ^2 are listed in Table 1. Fig. 5 demonstrates the final MSD of the four algorithms versus SNR. ℓ_1-RRLS has the best performance among the four algorithms and presents a gain of about 2 dB in MSD against SPARLS. ℓ_1-RLS performs better than RLS, but is slightly inferior to SPARLS.

Fig. 6 displays the time variation of the MSD for the four algorithms when SNR is 30 dB. In consistence with the results of Fig. 5, the ℓ_1-RRLS has the best performance among the four algorithms. These results suggest that the proposed ℓ_1-RRLS sparse system adaptive algorithm outperforms both SPARLS and RLS, whereas the unweighted ℓ_1-RLS variant is inferior

[1]The authors of [13] have generously shared the code for their simulations.

to SPARLS but superior to the regular RLS.

VI. Conclusions

This paper introduced a novel approach for adaptive identification of sparse systems. RLS algorithm is regularized by adding a weighted ℓ_1 norm penalty to the cost function. The update equations for this new approach are developed by utilizing subgradient analysis on the nondifferentiable ℓ_1 norm term. Two new adaptive algorithms result for two different weighting scenarios of the ℓ_1 norm, namely ℓ_1-RRLS and ℓ_1-RLS. Numerical simulations demonstrate that these algorithms do indeed bring about better convergence and steady-state performance than regular RLS when the system to be identified is sparse. The new ℓ_1 regularization based algorithms improve on the standard RLS, just as the recent sparsity regularization based LMS algorithms improve on the standard LMS algorithm in the sparse setting.

References

[1] R. Tibshirani, "Regression shrinkage and selection via the lasso," *J. Roy. Statist. Soc. Ser. B*, vol. 58, no. 1, pp. 267–288, 1996.

[2] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM J. Sci. Comput.*, vol. 20, no. 1, pp. 33–61, 1998.

[3] D. Angelosante, J. Bazerque, and G. Giannakis, "Online adaptive estimation of sparse signals: Where RLS meets the ℓ_1-norm," *IEEE Transactions on Signal Processing*, vol. 58, no. 7, pp. 3436 –3447, 2010.

[4] D. Angelosante and G. B. Giannakis, "RLS-weighted Lasso for adaptive estimation of sparse signals," in *Proc. ICASSP*, 19–24 April 2009, pp. 3245–3248.

[5] M. S. Asif and J. Romberg, "Streaming measurements in compressive sensing: ℓ_1 filtering," in *Proc. 42nd Asilomar Conference on Signals, Systems and Computers*, Oct. 26–29, 2008, pp. 1051–1058.

[6] D. L. Duttweiler, "Proportionate normalized least-mean-squares adaptation in echo cancelers," *IEEE Transactions on Speech and Audio Processing*, vol. 8, no. 5, pp. 508–518, Sep. 2000.

[7] B. D. Rao and B. Song, "Adaptive filtering algorithms for promoting sparsity," in *Proc. ICASSP*, vol. 6, 6–10 April 2003, pp. VI361–VI364.

[8] Y. Gu, J. Jin, and S. Mei, "l_0 norm constraint LMS algorithm for sparse system identification," *IEEE Signal Processing Letters*, vol. 16, no. 9, pp. 774–777, Sept. 2009.

[9] Y. Chen, Y. Gu, and A. O. Hero, "Sparse LMS for system identification," in *Proc. ICASSP*, 19–24 April 2009, pp. 3125–3128.

[10] 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, pp. 877–905, 2008.

[11] K. Slavakis, Y. Kopsinis, and S. Theodoridis, "Adaptive algorithm for sparse system identification using projections onto weighted ℓ_1 balls," in *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*, 2010, pp. 3742 –3745.

[12] K. Skretting and K. Engan, "Recursive least squares dictionary learning algorithm," *IEEE Transactions on Signal Processing*, vol. 58, no. 4, pp. 2121–2130, April 2010.

[13] B. Babadi, N. Kalouptsidis, and V. Tarokh, "SPARLS: The sparse RLS algorithm," *IEEE Transactions on Signal Processing*, vol. 58, no. 8, pp. 4013 –4025, 2010.

[14] M. A. T. Figueiredo and R. D. Nowak, "An EM algorithm for wavelet-based image restoration," *IEEE Transactions on Image Processing*, vol. 12, pp. 906–916, 2003.

[15] E. M. Eksioglu, "RLS adaptive filtering with sparsity regularization," in *10th International Conference on Information Sciences Signal Processing and their Applications (ISSPA)*, May 2010, pp. 550 –553.

[16] M. H. Hayes, *Statistical Digital Signal Processing and Modeling*. Wiley, March 1996.

[17] D. Bertsekas, A. Nedic, and A. Ozdaglar, *Convex Analysis and Optimization*. Athena Scientific, Cambridge, Massachusetts, 2003.

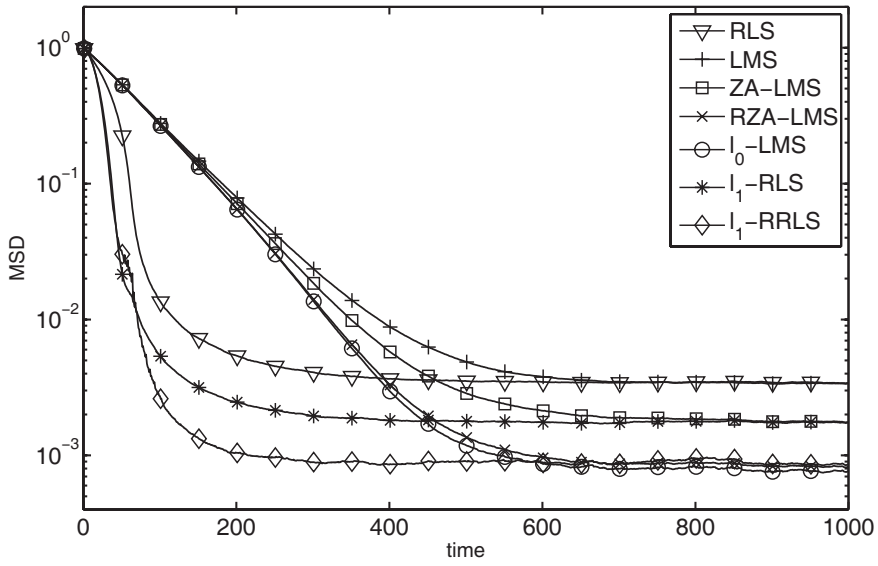

Fig. 1. Learning curves for ℓ_1-RRLS, ℓ_1-RLS, RLS, LMS, ℓ_0-LMS, RZA-LMS and LMS.

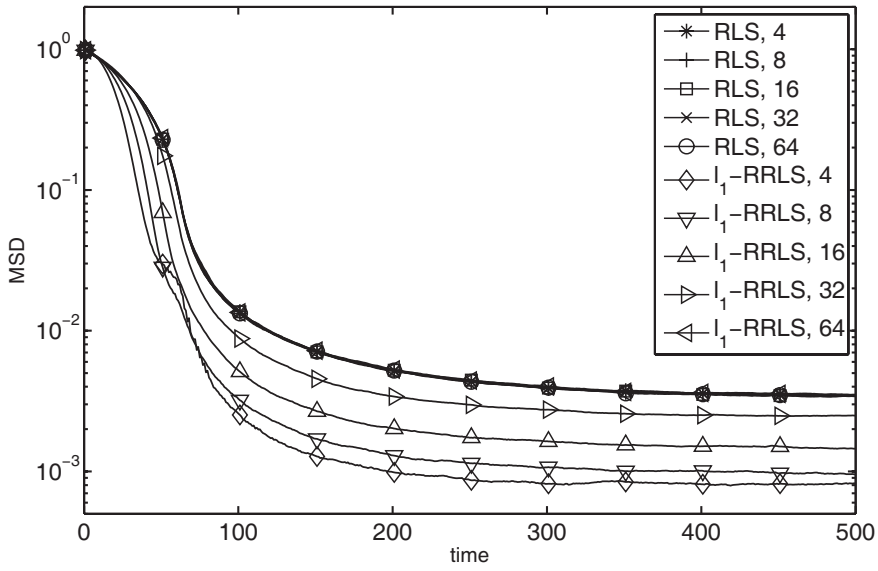

Fig. 2. Performance of ℓ_1-RRLS and RLS under different sparsity conditions.

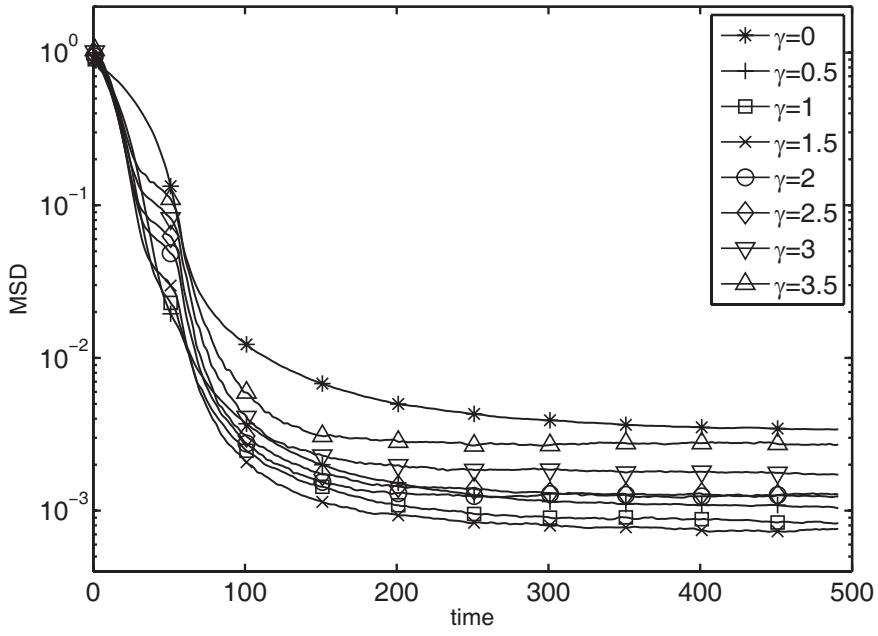

Fig. 3. Performance of ℓ_1-RRLS for different γ values.

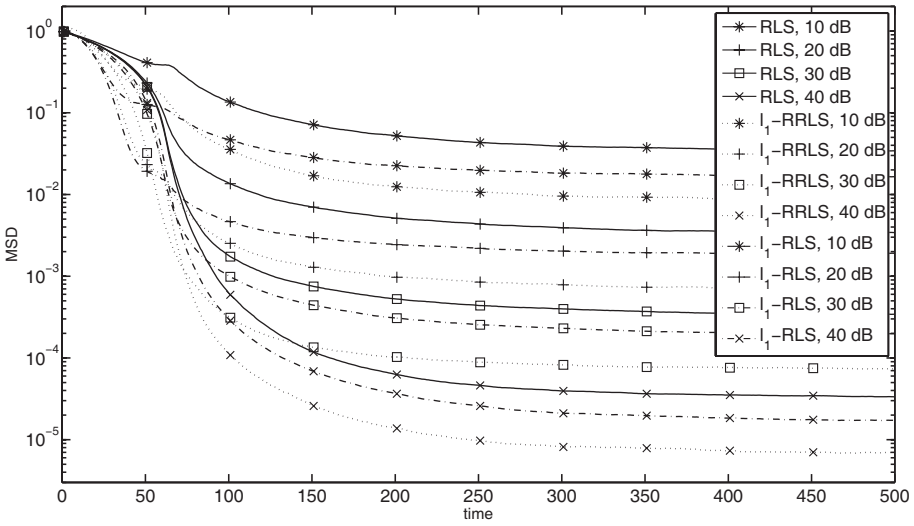

Fig. 4. Performance of ℓ_1-RRLS, ℓ_1-RLS and RLS for different SNR values.

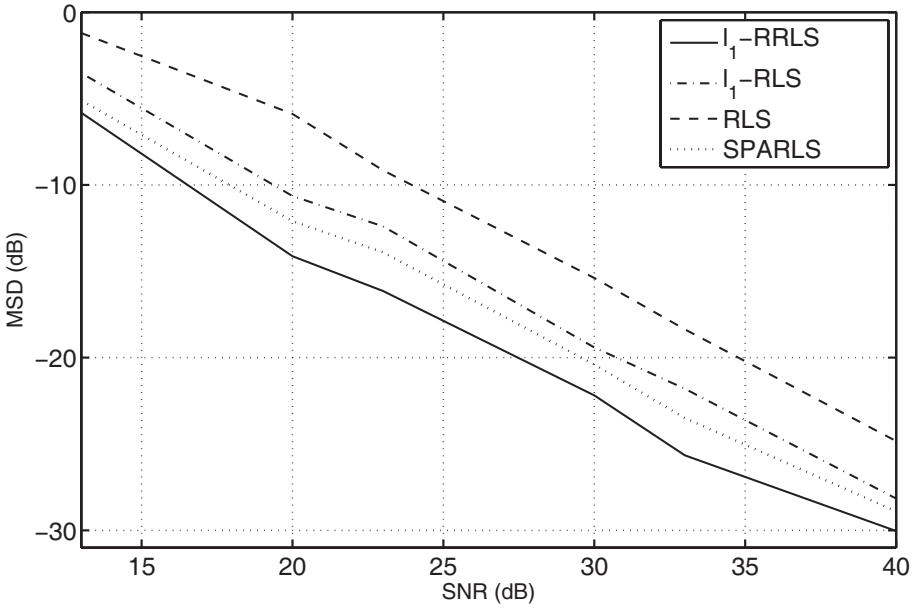

Fig. 5. MSD of ℓ_1-RRLS, ℓ_1-RLS, RLS and SPARLS versus SNR.

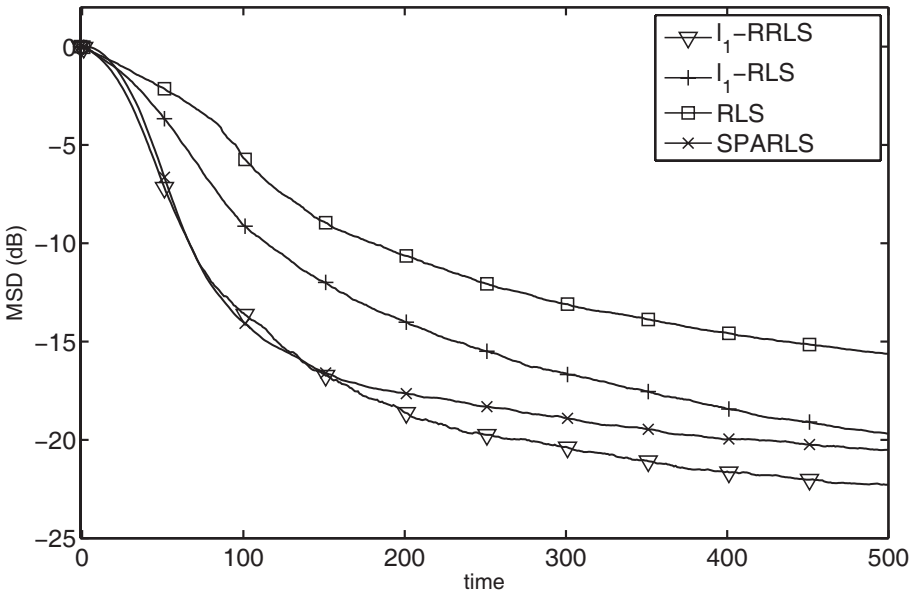

Fig. 6. Time variation of the MSD of ℓ_1-RRLS, ℓ_1-RLS, RLS and SPARLS for SNR=30 dB.