ONLINE LOW-RANK SUBSPACE LEARNING FROM INCOMPLETE DATA USING RANK REVEALING ℓ_2/ℓ_1 REGULARIZATION

Paris V. Giampouras, Athanasios A. Rontogiannis and Konstantinos D. Koutroumbas

IAASARS, National Observatory of Athens, GR-15236, Penteli, Greece

ABSTRACT

Massive amounts of data (also called *big data*) generated by a wealth of sources such as social networks, satellite sensors etc., necessitate the deployment of efficient processing tools. In this context, online subspace learning algorithms that aim at retrieving low-rank representations of data constitute a mainstay in many applications. Working with incomplete (partially observed) data has recently become commonplace. Moreover, the knowledge of the real rank of the sought subspace is rarely at our disposal *a priori*. Herein, a novel low-rank subspace learning algorithm from incomplete data is presented. Its main premise is the online processing of incomplete data along with the imposition of low-rankness on the sought subspace via a sophisticated utilization of the group sparsity inducing ℓ_2/ℓ_1 norm. As is experimentally shown, the resulting scheme is efficient in accurately learning the subspace as well as in unveiling its real rank.

***Index Terms*—** Subspace learning, low-rank, online, incomplete data, ℓ_2/ℓ_1 regularization

1. INTRODUCTION

Learning the underlying subspace where high-dimensional data "live" is a popular task arising in numerous problems in the machine learning and signal processing fields, [1]. The majority of the methods that have been developed for confronting those problems, suppose that the dimensionality of the sought subspace (i.e., the rank of the subspace matrix) is known in advance. This is however a rather strong assumption in many applications. As a result, traditional approaches might present an unstable behavior in this realistic scenario. Moreover, the development of algorithms that process large-scale incomplete datasets has become imperative nowadays [2]. That said, *online* low-rank subspace learning algorithms from incomplete data, recently proposed in the literature [3], constitute a valuable tool for handling those challenging problems.

A recursive least squares (RLS)-type subspace learning algorithm from incomplete data, called PETRELS, was put forth in [4]. PETRELS builds upon alternatingly estimating the subspace matrix and the projection of the incomplete data on its column space, by minimizing an exponentially weighted LS cost function. Since the knowledge of the rank of the subspace is at the heart of PETRELS, it becomes unstable when this condition is not met. Capitalizing on this, Algorithm 1 of [5] proposes the regularization of the PETRELS' cost function by adding a low-rank promoting term that robustifies the algorithm in the lack of this knowledge. That said, Algorithm 1 of [5] amounts to solving separate ridge-regression type problems for estimating the subspace matrix and the projection coefficients. In doing so, it becomes adept at dealing with the unawareness of the real rank of the sought subspace, albeit without revealing its true rank.

In this paper we propose a novel low-rank subspace estimation algorithm that learns from incomplete data . Allowing for the unawareness of the true rank of the subspace, a scheme is derived that efficiently promotes its low-rankness. Towards this and inspired by relevant Bayesian subspace learning ideas [6], [7], [8], we propose to impose jointly column sparsity on the subspace and the projection coefficient matrix via the group sparsity inducing ℓ_2/ℓ_1 norm. The novel cost function formed, which is the sum of an exponentially weighted (LS) term and a low-rank promoting ℓ_2/ℓ_1 term, is then solved via an alternating minimization strategy. The effectiveness of the proposed online column sparsity subspace learning algorithm in accurately estimating the subspace and revealing its true rank is manifested at experiments conducted on both simulated and real data.

2. PROBLEM STATEMENT

Let n be the time index and $\mathbf{z}(n)$ a sequence of *incomplete* $K \times 1$ data vectors of observations that can be expressed as,

$$\mathbf{z}(n) = \boldsymbol{\phi}(n) \odot \mathbf{y}(n) = \boldsymbol{\Phi}_n \mathbf{y}(n). \tag{1}$$

In (1), $\mathbf{y}(n)$ is the full $K \times 1$ data vector, $\boldsymbol{\phi}(n)$ is a $\{0,1\}$-binary $K \times 1$ vector having 0's at the positions where $\mathbf{y}(n)$ has missing entries and 1's elsewhere, $\boldsymbol{\Phi}_n = \mathrm{diag}(\boldsymbol{\phi}(n))$ and $\odot$ denotes Hadamard multiplication. In this work we consider that the data $\mathbf{y}(n)$ lie in a low-dimensional linear subspace of (unknown) rank $r(n)$ that may be time-varying. Based on this, we assume that our data are produced according to the

This work was funded by the PHySIS project, contract no. 640174, within the H2020 Framework Program of the European Commission.

following linear regression model

$$\mathbf{y}(n) = \mathbf{W}(n)\mathbf{x}(n) + \mathbf{e}(n), \tag{2}$$

where $\mathbf{W}(n)$ is a $K \times L$ subspace matrix with $K \gg L \geq r(n)$, $\mathbf{x}(n)$ is the $L \times 1$ low-dimensional representation of $\mathbf{y}(n)$ in the subspace spanned by the columns of $\mathbf{W}(n)$ and $\mathbf{e}(n)$ is zero-mean additive white Gaussian noise. It is clear from the above that since the true subspace rank is unknown, an overestimate L of it is implicitly assumed in (2). If we now stack together incomplete observation vectors up to time n as *rows* in a matrix $\mathbf{Z}(n)$, we easily get from (1) and (2)

$$\mathbf{Z}(n) = \mathbf{\Phi}(n) \odot \big(\mathbf{X}(n)\mathbf{W}^T(n) + \mathbf{E}(n)\big) \tag{3}$$

where

$$\mathbf{Z}(n) = [\mathbf{z}(1), \mathbf{z}(2), \ldots, \mathbf{z}(n)]^T = [\boldsymbol{z}_1(n), \boldsymbol{z}_2(n), \ldots, \boldsymbol{z}_K(n)], \tag{4}$$

$$\mathbf{\Phi}(n) = [\boldsymbol{\phi}(1), \boldsymbol{\phi}(2), \ldots, \boldsymbol{\phi}(n)]^T = [\boldsymbol{\varphi}_1(n), \boldsymbol{\varphi}_2(n), \ldots, \boldsymbol{\varphi}_K(n)], \tag{5}$$

$$\mathbf{X}(n) = [\mathbf{x}(1), \mathbf{x}(2), \ldots, \mathbf{x}(n)]^T = [\boldsymbol{x}_1(n), \boldsymbol{x}_2(n), \ldots, \boldsymbol{x}_L(n)] \tag{6}$$

and $\mathbf{E}(n) = [\mathbf{e}(1), \mathbf{e}(2), \ldots, \mathbf{e}(n)]^T$. In addition, we define the subspace matrix $\mathbf{W}(n)$ row- and columnwise as[1]

$$\mathbf{W}(n) = [\mathbf{w}_1(n), \mathbf{w}_2(n), \ldots, \mathbf{w}_K(n)]^T = [\boldsymbol{w}_1(n), \boldsymbol{w}_2(n), \ldots, \boldsymbol{w}_L(n)]. \tag{7}$$

It can be noticed from Eqs. (4)-(7) that the rowsize of matrices $\mathbf{Z}(n), \mathbf{\Phi}(n)$ and $\mathbf{X}(n)$ increases with time, while $\mathbf{W}(n)$ is a fixed size $K \times L$ matrix.

In this work we deal with the problem of online low-rank subspace learning and matrix completion. That is, given the incomplete data vector $\mathbf{z}(n)$ at each time instant n, we aim at estimating a) the subspace matrix $\mathbf{W}(n)$, b) its low-rank representation $\mathbf{x}(n)$ and c) the complete data vector $\mathbf{y}(n)$ as a by-product. In the following, such an online algorithm is presented, which besides the above, it also achieves to reveal the true rank of the unknown data subspace, thus leading to very accurate estimates.

3. THE PROPOSED MINIMIZATION PROBLEM

A direct way to solve the previously described problem is via alternating LS, as in [4]. In that context the following minimization problem is first defined,

$$\min_{\mathbf{x}(n),\mathbf{W}(n)} \|\mathbf{\Lambda}^{\frac{1}{2}}(n)\left(\mathbf{Z}(n) - \mathbf{\Phi}(n) \odot \left(\mathbf{X}(n)\mathbf{W}^T(n)\right)\right)\|_F^2 \equiv$$

$$\min_{\mathbf{x}(n),\mathbf{W}(n)} \sum_{i=1}^{n} \lambda^{n-i} \|\mathbf{z}(i) - \boldsymbol{\phi}(i) \odot (\mathbf{W}(n)\mathbf{x}(i))\|_2^2 \tag{8}$$

where λ is the usual forgetting factor with $0 \ll \lambda < 1$ and $\mathbf{\Lambda}(n) = \mathrm{diag}([\lambda^{n-1}, \lambda^{n-2}, \ldots, \lambda, 1]^T)$. Then $\mathbf{x}(n), \mathbf{W}(n)$ are estimated in an alternating fashion, a process which can also easily be adjusted in the online scenario. As shown in Section 5, such a procedure is sensitive to the lack of knowledge of the true rank of the underlying data subspace and diverges if an overestimate L of it is used. Motivated by this, we seek a specific mechanism that is able to reveal the true subspace rank $r(n)$ in time by gradually reducing the initially set rank value L as the learning task progresses.

To achieve this we first recall that the matrix product $\mathbf{X}(n)\mathbf{W}^T(n)$ appearing in (8) can be written as the sum of the outer products between the columns $\boldsymbol{x}_l(n)$ and $\boldsymbol{w}_l(n)$ of $\mathbf{X}(n)$ and $\mathbf{W}(n)$ respectively i.e.,

$$\mathbf{X}(n)\mathbf{W}^T(n) = \sum_{l=1}^{L} \boldsymbol{x}_l(n)\boldsymbol{w}_l^T(n). \tag{9}$$

It is obvious from (9) that the rank of $\mathbf{X}(n)\mathbf{W}^T(n)$ (and that of $\mathbf{W}(n)$) equals to the number of the rank-one terms existing into the summation. Having said that, it becomes clear that a possible way for reducing the subspace rank is by somehow discarding a few of those terms. In view of this, we propose the regularization of the LS cost function in (8) by utilizing the *column sparsity* promoting ℓ_2/ℓ_1 norm [9] applied on an appropriately formed matrix. To be more specific, we define the $(n + K + 1) \times L$ matrix $\mathbf{A}(n) = [\mathbf{X}^T(n)\mathbf{\Lambda}^{\frac{1}{2}}(n), \mathbf{W}^T(n), \boldsymbol{\epsilon}]^T$, i.e. we concatenate matrices $\mathbf{\Lambda}^{\frac{1}{2}}(n)\mathbf{X}(n)$ and $\mathbf{W}(n)$ columnwise and the $L \times 1$ vector $\boldsymbol{\epsilon}$ with entries small constants $\epsilon^{\frac{1}{2}}$, so that any column sparsity inducing mechanism applied on $\mathbf{A}(n)$ to annihilate jointly the corresponding columns of $\mathbf{X}(n)$ and $\mathbf{W}(n)$. The ℓ_2/ℓ_1 norm of $\mathbf{A}(n)$ is defined as

$$\|\mathbf{A}(n)\|_{2,1} = \sum_{l=1}^{L} \|\mathbf{a}_l(n)\|_2, \tag{10}$$

where $\mathbf{a}_l(n) = [\boldsymbol{x}_l^T(n)\mathbf{\Lambda}^{\frac{1}{2}}(n), \boldsymbol{w}_l^T(n), \epsilon^{\frac{1}{2}}]^T$ is the lth column of $\mathbf{A}(n)$ and

$$\|\mathbf{a}_l(n)\|_2 = \sqrt{\boldsymbol{x}_l^T(n)\mathbf{\Lambda}(n)\boldsymbol{x}_l(n) + \|\boldsymbol{w}_l(n)\|_2^2 + \epsilon}. \tag{11}$$

Favorably, the inclusion of the small constants ϵ's in $\mathbf{A}(n)$, renders smooth the inherently non-smooth ℓ_2/ℓ_1 norm, [10]. We are now in place to define the following minimization problem for low-rank subspace estimation,

$$\min_{\mathbf{x}(n),\mathbf{W}(n)} \sum_{i=1}^{n} \lambda^{n-i} \|\mathbf{z}(i) - \mathbf{\Phi}(i) \odot (\mathbf{W}(n)\mathbf{x}(i))\|_2^2$$
$$+\delta \sum_{l=1}^{L} \sqrt{\boldsymbol{x}_l^T(n)\mathbf{\Lambda}(n)\boldsymbol{x}_l(n) + \|\boldsymbol{w}_l(n)\|_2^2 + \epsilon}, \tag{12}$$

where δ is a low-rank regularization parameter. The proposed minimization problem consists of two terms: a) an exponentially weighted LS term which performs the fitting between the data and the model and b) the previously defined ℓ_2/ℓ_1

[1] In (4)-(7), small bold calligraphic letters have been used to denote columns of matrices and regular bold letters to denote rows.

norm which is used as a regularizing term for imposing low-rankness.

It is worthy to underline that albeit the first term of (12) decouples over both the rows and the columns of $\mathbf{X}(n)$ and $\mathbf{W}(n)$, this is not the case with the second low-rank promoting term. Hence, getting closed form expressions for estimating $\mathbf{x}(n)$ and $\mathbf{W}(n)$ at time n is rendered infeasible, while also the lack of such a decoupling seems to hinder the derivation of an online scheme. However, in the next section we show that by adopting an alternating minimization strategy that combines regularized LS and cyclic coordinate-descent type steps, an efficient online subspace learning algorithm can be obtained.

4. THE PROPOSED ALGORITHM

Since we adhere to the online scenario we assume that incomplete datums $\mathbf{z}(n)$'s are processed sequentially over time. At each time instance n, our subspace learning task amounts to: a) getting an estimate $\hat{\mathbf{x}}(n)$ of the projection coefficients vector $\mathbf{x}(n)$ of $\mathbf{z}(n)$ on the column space of the subspace matrix estimate $\hat{\mathbf{W}}(n-1)$ (computed in the previous time instance $n-1$) and b) updating the subspace matrix estimate to $\hat{\mathbf{W}}(n)$.

As mentioned previously, the approach that we follow is based on minimizing (12) alternatingly w.r.t. $\mathbf{x}(n)$ and $\mathbf{W}(n)$. Defining the $L \times L$ diagonal matrix $\mathbf{D}(n)$ with

$$d_l(n) = \frac{\delta}{\sqrt{\hat{\boldsymbol{x}}_l^T(n)\boldsymbol{\Lambda}(n)\hat{\boldsymbol{x}}_l(n) + \|\hat{\boldsymbol{w}}_l(n)\|_2^2 + \epsilon}}, \quad (13)$$

its diagonal entries, we first minimize (12) w.r.t. $\mathbf{x}(n)$ and get an *approximate* closed-form solution[2] for $\hat{\mathbf{x}}(n)$, i.e.,

$$\hat{\mathbf{x}}(n) = \left(\hat{\mathbf{W}}^T(n-1)\boldsymbol{\Phi}_n\hat{\mathbf{W}}(n-1) + \mathbf{D}(n-1)\right)^{-1} \times \hat{\mathbf{W}}(n-1)^T\mathbf{z}(n). \quad (14)$$

Next we adopt a block coordinate-descent (CD) type minimization of (12) w.r.t. the columns of $\mathbf{W}(n)$, [11], which results to the following expression for the estimate of its klth element[3],

$$\hat{w}_{kl}(n) = \left(\sum_{i=1}^{n}\lambda^{n-i}\phi_k(i)\hat{x}_l^2(i) + d_l(n-1)\right)^{-1}\sum_{i=1}^{n}\lambda^{n-i}\hat{x}_l(i)\times \left(z_k(i) - \phi_k(i)\left(\sum_{l'<l}\hat{x}_{l'}(i)\hat{w}_{kl'}(n) + \sum_{l'>l}\hat{x}_{l'}(i)\hat{w}_{kl'}(n-1)\right)\right). \quad (15)$$

In Eqs. (14), (15) there is a subtle point that should be accentuated: both $\hat{\mathbf{x}}(n)$ and $\hat{\mathbf{W}}(n)$ aside from their inherent interrelation shown in (14) and (15), they involve matrix $\mathbf{D}(n-1)$ which in turn includes quantities depending on $\hat{\boldsymbol{x}}_l(n-1)$ and $\hat{\boldsymbol{w}}_l(n-1)$. Hence, it arises that $\hat{\mathbf{x}}(n)$ estimated at time n is also influenced by the projection coefficient vectors estimated in the previous time instants. A similar case occurs for the estimate $\hat{\mathbf{W}}(n)$ of the subspace matrix, which due to the presence of $\mathbf{D}(n-1)$ in (15) also relies on its estimate obtained in the previous time instant. It should be noted that this particular characteristic of our method results from the aforementioned *non-decoupling* nature of the low-rank regularizing term utilized. This tricky point gives the proposed algorithm the ability of revealing the true rank of the subspace after convergence, an issue that is further explained below and highlighted in the experimental section.

Eq. (15) can be written more compactly and avoid time-increasing summation terms by incorporating appropriate time-updating formulas. First we define the following fixed size w.r.t. time quantities

$$\mathbf{T}(n) = \hat{\mathbf{X}}^T(n)\boldsymbol{\Lambda}(n)\mathbf{Z}(n), \quad (16)$$

$$\mathbf{P}_k(n) = \hat{\mathbf{X}}^T(n)\boldsymbol{\Lambda}(n)\boldsymbol{\Phi}_k(n)\hat{\mathbf{X}}(n), \quad k = 1, 2, \ldots K, \quad (17)$$

$$q_l(n) = \hat{\boldsymbol{x}}_l^T(n)\boldsymbol{\Lambda}(n)\hat{\boldsymbol{x}}_l(n), \quad l = 1, 2, \ldots L, \quad (18)$$

where $\boldsymbol{\Phi}_k(n) = \mathrm{diag}(\boldsymbol{\varphi}_k(n))$. Eqs. (16), (17), (18) can be easily expressed time-recursively as

$$\mathbf{T}(n) = \lambda\mathbf{T}(n-1) + \hat{\mathbf{x}}(n)\mathbf{z}^T(n), \quad (19)$$

$$\mathbf{P}_k(n) = \lambda\mathbf{P}_k(n-1) + \phi_k(n)\hat{\mathbf{x}}(n)\hat{\mathbf{x}}^T(n), \quad (20)$$

$$q_l(n) = \lambda q_l(n-1) + \hat{x}_l^2(n). \quad (21)$$

The term $\hat{\boldsymbol{x}}_l^T(n)\boldsymbol{\Lambda}(n)\hat{\boldsymbol{x}}_l(n)$ which appears in the expression of $d_l(n)$ (13) coincides with $q_l(n)$ and thus it is efficiently computed via (21). Following the same path Eq. (15) is rewritten by integrating the above-defined quantities yielding

$$\hat{w}_{kl}(n) = \frac{t_{lk}(n) - \mathbf{p}_{k\neg l}^T(n)\hat{\mathbf{w}}_{k\neg l}(n)}{p_{k,ll}(n) + d_l(n-1)}, \quad (22)$$

where $t_{lk}(n)$ denotes the lkth entry of the $L\times K$ matrix $\mathbf{T}(n)$, $\mathbf{p}_{k\neg l}^T(n)$ is the lth row of the $L \times L$ autocorrelation matrix $\mathbf{P}_k(n)$ after ignoring its lth entry $p_{k,ll}(n)$, and finally

$$\hat{\mathbf{w}}_{k\neg l}(n) = [\hat{w}_{k1}(n), \hat{w}_{k2}(n), \ldots, \hat{w}_{kl-1}(n), \hat{w}_{kl+1}(n-1), \ldots, \hat{w}_{kL}(n-1)]^T. \quad (23)$$

The proposed online column sparsity promoting subspace learning algorithm (OCSpSL) from incomplete data is summarized in Algorithm 1. By using an element-by-element CD estimation procedure for the unknown subspace matrix, the computational complexity of OCSpSL is reduced to $\mathcal{O}(|\boldsymbol{\phi}(n)|L^2)$ (with $|\boldsymbol{\phi}(n)|$ the number of 1's in $\boldsymbol{\phi}(n)$) which is similar to that of PETRELS and lower than that of Algorithm 1 in [5]. The most important feature of the proposed algorithm though is its ability in retrieving the true rank of the unknown subspace, by zeroing whole columns of $\hat{\mathbf{W}}(n)$

[2]The exact (yet not closed-form) expression for $\hat{\mathbf{x}}(n)$ contains $\mathbf{D}(n)$ in place of $\mathbf{D}(n-1)$ in Eq. (14).

[3]Note that in (15) we consider a single iteration of the CD procedure and map CD iterations to time iterations.

Algorithm 1 The OCSpSL algorithm

Initialize $\mathbf{W}(0), \mathbf{D}(0)$
Set $\mathbf{T}(0) = \mathbf{0}, \mathbf{q}(0) = \mathbf{0}, \delta$
for $n = 1, 2, \ldots$
  Compute $\hat{\mathbf{x}}(n)$ from (14)
  Update $\mathbf{q}(n), \mathbf{D}(n)$ from (21) and (13)
  Update $\mathbf{T}(n)$ from (19)
  for $k = 1, 2, \ldots, K$
    Update $\mathbf{P}_k(n)$ from (20)
    for $l = 1, 2, \ldots, L$
      Compute $\hat{w}_{kl}(n)$ from (22)
    end
  end
end

after convergence. This is readily explained from (22), since for obsolete columns of $\mathbf{A}(n)$, as verified by (11) and (13), $d_l(n-1)$ takes large values that drive all entries of the lth column of $\hat{\mathbf{W}}(n)$ to zero, thus reducing its rank.

5. EXPERIMENTAL RESULTS AND DISCUSSION

In this section the performance of OCSpSL is evaluated on a simulated online matrix completion problem as well as on the reconstruction of a real hyperspectral image.

Online matrix completion. Since our aim is to simulate a matrix completion problem, we generate a low-dimensional subspace matrix $\mathbf{U} \in \mathcal{R}^{K \times r}$ with $K = 500$ and $r = 5$ and i.i.d Gaussian distributed entries i.e., $u_{kl} \sim \mathcal{N}(0, \frac{1}{K})$. Next, we produce $n = 25000$ $r \times 1$ projection coefficient vectors $\mathbf{c}(n)$ that follow a Gaussian distribution, $\mathbf{c}(n) \sim \mathcal{N}(0, \mathbf{I}_r)$. Then, the signal $\mathbf{y}(n)$ at time n is generated as $\mathbf{U}\mathbf{c}(n)$. Additive i.i.d Gaussian noise of variance $\sigma^2 = 10^{-3}$ is assumed to contaminate the data. OCSpSL is compared with two state-of-the-art subspace learning algorithms, i.e., PETRELS [4] and Algorithm 1 of [5]. The low-rank regularization parameters of OCSpSL and Algorithm 1 of [5] are set to 0.1 and $\lambda = 0.99$. The algorithms are tested for two different percentages of missing (at random) data i.e., $\pi = \{0.5, 0.8\}$. As metric performance we utilize the normalized reconstruction average estimation error (NRAEE) $\text{NRAEE}(n) = \frac{1}{n}\sum_{i=1}^{n} \frac{\|\hat{\mathbf{y}}(i) - \mathbf{y}(i)\|_2}{\|\mathbf{y}(i)\|_2}$ and the initial rank of the subspace matrices is set to $L = 20$.

As is shown in Fig. 1, OCSpSL exhibits a robust behavior, outperforming its rivals in terms of NRAEE for both values of π, while PETRELS diverges after a number of iterations. Interestingly, OCSpSL besides its robustness and improved estimation performance, it is also able to retrieve the real subspace rank. After convergence most of the columns of $\hat{\mathbf{W}}(n)$ are zero and its rank is 5. On the contrary, Algorithm 1 of [5] ends up without decreasing the initially set rank value. Note that this favorable behavior of OCSpSL is observed in many different experiments that have been run, but are not presented here due to space limitations.

Pixel-by-pixel hyperspectral image reconstruction. A hyperspectral image (HSI) consists of multiple gray-scale images captured at narrow contiguous spectral bands. Thus,

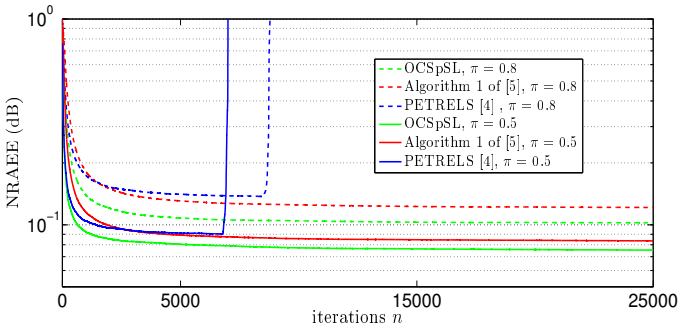

Fig. 1: NRAEE obtained by OCSpSL, PETRELS [4] and Algorithm 1 of [5] on the simulated matrix completion problem.

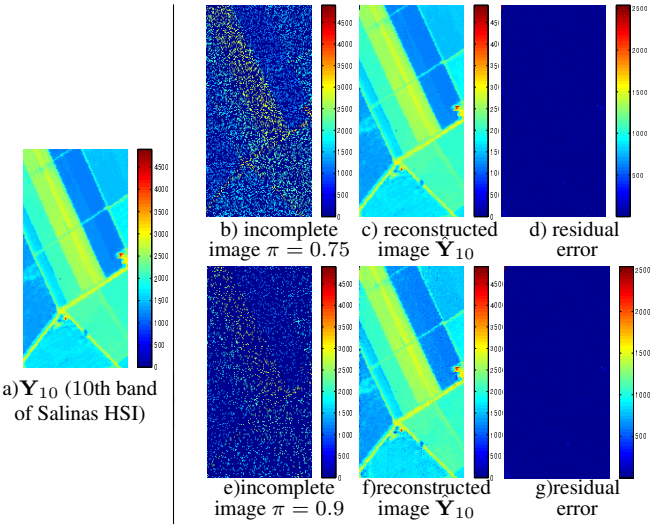

Fig. 2: Reconstruction of Salinas Valley HSI using OCSpSL, for different fractions of observed entries π.

each pixel is represented by a high dimensional vector called *spectral signature*. Each vector entry is radiance captured at a specific spectral band. HSIs inherently present a high degree of correlation both in the spectral and spatial domains [12]. In light of this, if we form a matrix with rows corresponding to the pixels' spectral signatures (i.e., the rowsize of the matrix is the number of pixels), then we can safely assume that the formed matrix can be well approximated by a low-rank one.

Herein, the Salinas Valley HSI dataset, [12], is utilized and we consider two different incomplete versions of the dataset corresponding to different fractions of the missing entries, namely $\pi = \{0.75, 0.9\}$. OCSpSL processes sequentially (online) the pixels of the HSI, i.e, it follows a row-by-row processing of the formed matrix. Figs. 2a,2b,2e show the 10th band image of the Salinas HSI and the incomplete versions of it corresponding to the different π's tested. As it can be derived by the reconstructed images and the residual images ($|\mathbf{Y}_{10} - \hat{\mathbf{Y}}_{10}|$) in Figs. 2c,2d,2f,2g, OCSpSL reliably reconstructs the HSI for both π's, thus corroborating its efficiency also on this real data experiment.

6. REFERENCES

[1] S. Theodoridis, *Machine Learning: A Bayesian and Optimization Perspective*, Academic Press, 2015.

[2] K. Slavakis, G.B. Giannakis, and G. Mateos, "Modeling and optimization for big data analytics: (statistical) learning tools for our era of data deluge," *Signal Processing Magazine, IEEE*, vol. 31, no. 5, pp. 18–31, Sept 2014.

[3] L. Balzano, R. Nowak, and B. Recht, "Online identification and tracking of subspaces from highly incomplete information," in *Communication, Control, and Computing (Allerton), 2010 48th Annual Allerton Conference on*, Sept 2010, pp. 704–711.

[4] Y. Chi, Y.C. Eldar, and R. Calderbank, "PETRELS: Parallel subspace estimation and tracking by recursive least squares from partial observations," *Signal Processing, IEEE Transactions on*, vol. 61, no. 23, pp. 5947–5959, Dec. 2013.

[5] M. Mardani, G. Mateos, and G. B. Giannakis, "Subspace learning and imputation for streaming big data matrices and tensors," *Signal Processing, IEEE Transactions on*, vol. 63, no. 10, pp. 2663–2677, May 2015.

[6] V. Tan and C. Févotte, "Automatic relevance determination in nonnegative matrix factorization," in *SPARS'09-Signal Processing with Adaptive Sparse Structured Representations*, 2009.

[7] S.D. Babacan, M. Luessi, R. Molina, and A.K. Katsaggelos, "Sparse bayesian methods for low-rank matrix estimation," *Signal Processing, IEEE Transactions on*, vol. 60, no. 8, pp. 3964–3977, Aug 2012.

[8] P.V. Giampouras, A.A. Rontogiannis, K.E. Themelis, and K.D. Koutroumbas, "Online bayesian low-rank subspace learning from partial observations," in *Signal Processing Conference (EUSIPCO), 2015 23rd European*, Aug 2015, pp. 2526–2530.

[9] F. Bach, R. Jenatton, J. Mairal, and G. Obozinski, "Optimization with sparsity-inducing penalties," *Foundations and Trends® in Machine Learning*, vol. 4, no. 1, pp. 1–106, 2012.

[10] E. M. Eksioglu, "Group sparse RLS algorithms," *International Journal of Adaptive Control and Signal Processing*, vol. 28, no. 12, pp. 1398–1412, 2014.

[11] D. P. Bertsekas, *Nonlinear Programming*, Athena Scientific, 1999.

[12] P. V. Giampouras, K. E Themelis, A. A Rontogiannis, and K. D. Koutroumbas, "Simultaneously sparse and low-rank abundance matrix estimation for hyperspectral image unmixing," *arXiv preprint arXiv:1504.01515*, 2015.