State Estimation of Linear Systems With Sparse Inputs and Markov-Modulated Missing Outputs

Joseph, Geethu; Varshney, Pramod K.

Publication date
2022

Document Version
Final published version

Published in
30th European Signal Processing Conference, EUSIPCO 2022 - Proceedings

Citation (APA)
Joseph, G., & Varshney, P. K. (2022). State Estimation of Linear Systems With Sparse Inputs and Markov-Modulated Missing Outputs. In *30th European Signal Processing Conference, EUSIPCO 2022 - Proceedings* (pp. 837-841). (European Signal Processing Conference; Vol. 2022-August). European Signal Processing Conference, EUSIPCO.

State Estimation of Linear Systems With Sparse Inputs and Markov-Modulated Missing Outputs

Geethu Joseph
Faculty of EEMCS
Delft University of Technology
Delft 2628 CD, The Netherlands
Email: g.joseph@tudelft.nl

Pramod K. Varshney *Life Fellow, IEEE*
Department of EECS
Syracuse University
Syracuse 13244, USA
Email: varshney@syr.edu

***Abstract*—In this paper, we consider the problem of estimating the states of a linear dynamical system whose inputs are jointly sparse and outputs at a few unknown time instants are missing. We model the missing data mechanism using a Markov chain with two states representing the missing and non-missing data. This mechanism with memory governed by the Markov chain models intermittent outages due to communication channels and occlusions corresponding to moving objects. We rely on the sparse Bayesian learning framework to derive an estimation algorithm that uses Kalman smoothing to handle temporal correlation and the Viterbi algorithm to handle missing data. Further, we demonstrate the utility of our algorithm by applying it to the frequency division duplexed multiple input multiple output downlink channel estimation problem.**

***Index Terms*—Compressed sensing, missing data, sparsity, intermittent observations, sparse Bayesian learning, Viterbi algorithm, FDD MIMO channel estimation.**

I. Introduction

The problem of missing data is relatively common in almost all data acquisition systems and can significantly affect the inferences from the data. This problem can be handled to some extent by exploiting the underlying structure in the data and interpolating the missing part. We study the problem of missing data in the context of state estimation of a first-order discrete linear dynamical system. The inputs to the systems are jointly sparse vectors and linear noisy measurements corresponding to some states are missing. Further, we consider intermittent outages or a missing data mechanism "with memory". This mechanism is governed by a two-state (missing or not missing) Markov process. Our goal is to utilize the inter-vector correlation in the state vectors, the joint support of the inputs, and the Markov-modulated missing mechanism to estimate the states of the system at all time instants.

The problem of sparse recovery with bursty missing data arises in several applications. One important application domain is internet of things (IoT). For example, some use cases are the monitoring of a temporally correlated and sparse processes such as motion tracking [1], network traffic reconstruction [2], localization refinement [3], urban traffic sensing improvement [4], and structural health monitoring [5], [6]. Here, intermittent data outages may occur due to nonlinear energy harvesting or environmental factors. Similarly, in a satellite imaging system, unknown natural images are sparse in the discrete cosine transformation or wavelet basis. Also, bursty outages occur in satellite communication affected by space weather due to perturbations of the ionosphere [7], [8]. Another example is the frequency division duplexing (FDD) multiple-input multiple-output (MIMO) downlink channel estimation at the base station via feedback from the mobile users [9]. The wireless channel in consecutive time slots are jointly sparse in the angular domain and temporally correlated. Further, the uplink channel can be bursty, and thus, some feedback signals can be missing. Motivated by these applications, in this paper, we study the recovery algorithm for temporally correlated sparse vectors, which handles missing data.

Sparse signal recovery when there is missing data was introduced in [10] for a single measurement vector model. Following this work, several other works studied sparse recovery problems with missing data in different applications [2]–[6]. The measurement bounds for single sparse vector recovery with missing data have also been investigated [11], [12]. For the multiple measurement model, the recovery of temporally correlated sparse vectors with missing data was studied in [13]. This study made a strong assumption that the indices of the missing measurements are known to the algorithm. However, for many practical applications like FDD MIMO downlink channel estimation problem, this assumption does not hold. Furthermore, as we mentioned above, these systems can have intermittent outages (with memory governed by a Markov process). Therefore, in this paper, we relax this assumption and consider the recovery of temporally correlated sparse vectors when the missing indices are Markov-modulated and unknown.

The specific contributions of the paper are two-fold.

- We devise a state estimation algorithm by combining the framework of sparse Bayesian learning (SBL) with the Viterbi algorithm to identify the unknown indices of Markov-modulated missing measurements and the Kalman smoothing algorithm to exploit temporal correlation.
- We apply the algorithm to the FDD MIMO downlink channel estimation problem. Using Monte Carlo simulation results, we illustrate the performance of the algorithm and compare it with competing algorithms.

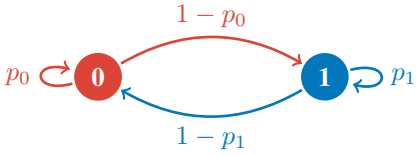


Figure 1. State transition diagram for the missing data mechanism showing the transition probabilities given by (3).

II. System Model

We consider a discrete-time linear dynamical system whose state $\boldsymbol{x}_k \in \mathbb{R}^N$ obeys the following model:

$$\boldsymbol{x}_k = \boldsymbol{D}\boldsymbol{x}_{k-1} + \boldsymbol{u}_k. \tag{1}$$

Here, $k > 0$ is the discrete time index, $\boldsymbol{D} \in \mathbb{R}^{N\times N}$ is the known state transition matrix, and $\boldsymbol{u}_k \in \mathbb{R}^N$ is the input. We assume that the inputs are jointly sparse, i.e., they have a few nonzero elements compared to their dimension N, and their supports (the locations of the nonzero indices) are the same across all the time instants. We also assume that the initial state $\boldsymbol{x}_0 = \boldsymbol{0} \in \mathbb{R}^N$. Here, $\boldsymbol{D}$ is an arbitrary matrix and as a result, the state $\boldsymbol{x}_k$ is not necessarily jointly sparse.

Further, the output $\boldsymbol{y}_k \in \mathbb{R}^m$ of the system at time k is given by

$$\boldsymbol{y}_k = \boldsymbol{z}_k^* \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{w}_k, \tag{2}$$

where $\boldsymbol{A} \in \mathbb{R}^{m\times N}$ is the output matrix, and $\boldsymbol{w}_k \in \mathbb{R}^m$ is the output noise with $\boldsymbol{w}_k \sim \mathcal{N}(\boldsymbol{0}, \sigma^2\boldsymbol{I})$. Also, $\boldsymbol{z}_k^* \in \{0,1\}$ models the missing data mechanism, i.e., if $\boldsymbol{z}_k^* = 0$, the corresponding state information is missing at the output, and $\boldsymbol{y}_k$ is independent of $\boldsymbol{x}_k$. The missing data indices $\boldsymbol{z}_k^*, k = 1, 2, \ldots$ follow a *hidden Markov model* defined by the state transition probabilities as given in Fig. 1. For any integer $k > 0$ and $i, j \in \{0, 1\}$,

$$\mathbb{P}\left\{\boldsymbol{z}_k^* = i | \boldsymbol{z}_{k-1}^* = j\right\} = \begin{cases} p_j & \text{if } i = j \\ 1 - p_j & \text{if } i \neq j. \end{cases} \tag{3}$$

For a given integer value of $K < \infty$, our goal is to recover the state matrix $\boldsymbol{X}$ using the output matrix $\boldsymbol{Y}$ when $\boldsymbol{z}^*$ is unknown, where we define

$$\boldsymbol{X} \triangleq \begin{bmatrix} \boldsymbol{x}_1 & \boldsymbol{x}_2 & \ldots & \boldsymbol{x}_K \end{bmatrix} \in \mathbb{R}^{N\times K} \tag{4}$$

$$\boldsymbol{Y} \triangleq \begin{bmatrix} \boldsymbol{y}_1 & \boldsymbol{y}_2 & \ldots & \boldsymbol{y}_K \end{bmatrix} \in \mathbb{R}^{m\times K} \tag{5}$$

$$\boldsymbol{z}^* \triangleq \begin{bmatrix} \boldsymbol{z}_1^* & \boldsymbol{z}_2^* & \ldots & \boldsymbol{z}_K^* \end{bmatrix}^\mathsf{T} \in \{0,1\}^K. \tag{6}$$

The main challenge here is that the missing data indices given by the locations of zeros in $\boldsymbol{z}^*$ are unknown to the algorithm. We recall that if the data is missing at time instant k (i.e., $\boldsymbol{z}_k^* = 0$), then the output $\boldsymbol{y}_k$ has no information about the corresponding state $\boldsymbol{x}_k$. However, it is possible to recover the state $\boldsymbol{x}_k$ by exploiting the temporal correlation modeled using the first-order autoregressive model in (1). To this end, the recovery algorithm needs to exploit the different underlying signal structures: the joint sparsity of the inputs, the temporal correlation of the states, and the temporal correlation of the missing data indices.

The next section presents a Bayesian algorithm that accounts for all these structures to identify the missing data indices and recover the sparse inputs and system states.

III. Bayesian Recovery Algorithm

We use the SBL framework to exploit the sparse structure of the inputs [14], [15]. The SBL formulation imposes a fictitious Gaussian prior $\boldsymbol{u}_k \sim \mathcal{N}(\boldsymbol{0}, \text{Diag}\{\boldsymbol{\gamma}^*\}), k = 1, 2, \ldots, K$ on the sparse vectors, where $\text{Diag}\{\boldsymbol{\gamma}^*\}$ is the common diagonal covariance matrix with $\boldsymbol{\gamma}^* \in \mathbb{R}_+^N$ along the diagonal. The Gaussian prior promotes sparsity and using a common covariance matrix captures the jointly sparse structure [16]. Using this framework, we compute the states $\boldsymbol{x}_k$ via type II maximum likelihood (ML) estimation. To be specific, we first compute the ML estimate of the parameters $\boldsymbol{\gamma}^*$ and $\boldsymbol{z}^*$ using the output $\boldsymbol{Y}$. Let $\bar{\boldsymbol{\theta}}$ be the ML estimate of $\boldsymbol{\theta}^* = \begin{bmatrix} \boldsymbol{\gamma}^* & \boldsymbol{z}^* \end{bmatrix} \in \mathbb{R}_+^N \times \{0,1\}^K$. Then, the state estimate at time k is given by

$$\widehat{\boldsymbol{x}}_k = \mathbb{E}\left\{\boldsymbol{x}_k \middle| \boldsymbol{Y}, \bar{\boldsymbol{\theta}}\right\}. \tag{7}$$

We solve this ML estimation problem using the expectation-maximization (EM) algorithm. The EM algorithm is an iterative procedure for ML estimation of parameters of a probabilistic model that depends on unobserved or hidden data. In our case, the parameter is $\boldsymbol{\theta}^*$, and the hidden variable is the state matrix $\boldsymbol{X}$ that is unknown while estimating $\boldsymbol{\theta}^*$. The EM algorithm alternates between an expectation step (E-step) and a maximization step (M-step). In every iteration, the E-step computes the marginal log-likelihood $Q(\cdot)$ of the observed data $\boldsymbol{Y}$ using the estimates of parameters obtained in the previous iteration. The M-step computes the estimates of $\boldsymbol{\theta}^*$ that maximizes $Q(\cdot)$. Let $\boldsymbol{\theta}^{(r)} \in \mathbb{R}_+^N \times \{0,1\}^K$ be the parameter estimates in the r^{th} iteration. Then, the E and M-steps in the $(r+1)^{\text{th}}$ iteration is given by

$$\textbf{E-step:} \quad Q\left(\boldsymbol{\theta}; \boldsymbol{\theta}^{(r)}\right) = \mathbb{E}_{\boldsymbol{X}|\boldsymbol{Y},\boldsymbol{\theta}^{(r)}}\left\{\log p(\boldsymbol{Y}, \boldsymbol{X}, \boldsymbol{\theta})\right\} \tag{8}$$

$$\textbf{M-step:} \quad \boldsymbol{\theta}^{(r+1)} \in \underset{\boldsymbol{\theta}\in\mathbb{R}_+^N\times\{0,1\}^K}{\arg\max}\, Q\left(\boldsymbol{\theta}; \boldsymbol{\theta}^{(r)}\right), \tag{9}$$

where $\boldsymbol{\theta} = \begin{bmatrix} \boldsymbol{\gamma} \in \mathbb{R}_+^N & \boldsymbol{z} \in \{0,1\}^K \end{bmatrix}$. Simplifying $Q(\cdot)$ using (1) and (2),

$$Q\left(\boldsymbol{\theta}; \boldsymbol{\theta}^{(r)}\right) = \mathbb{E}_{\boldsymbol{X}|\boldsymbol{Y},\boldsymbol{\theta}^{(r)}}\left\{\log\left[p\left(\boldsymbol{Y}|\boldsymbol{X}, \boldsymbol{z}\right) p\left(\boldsymbol{z}\right) p\left(\boldsymbol{X}, \boldsymbol{\gamma}\right)\right]\right\}. \tag{10}$$

From the above relation, we see that the optimization problem in the M-step is separable in the variables $\boldsymbol{\gamma}$ and $\boldsymbol{z}$. Thus, we have

$$\boldsymbol{\gamma}^{(r+1)} \in \underset{\boldsymbol{\gamma}\in\mathbb{R}_+^N}{\arg\max}\, \mathbb{E}_{\boldsymbol{X}|\boldsymbol{Y},\boldsymbol{\theta}^{(r)}}\left\{\log p\left(\boldsymbol{X}, \boldsymbol{\gamma}\right)\right\} \tag{11}$$

$$\boldsymbol{z}^{(r+1)} \in \underset{\boldsymbol{z}\in\{0,1\}^K}{\arg\max}\, \mathbb{E}_{\boldsymbol{X}|\boldsymbol{Y},\boldsymbol{\theta}^{(r)}}\{\log p\left(\boldsymbol{Y}|\boldsymbol{X}, \boldsymbol{z}\right)\} + \log p\left(\boldsymbol{z}\right). \tag{12}$$

We describe the solution to the above two optimization problems in the following subsections.

A. Estimation of $\boldsymbol{\gamma}^{(r+1)}$

To estimate $\boldsymbol{\gamma}^{(r+1)}$, we need to compute the quantity $\mathbb{E}_{\boldsymbol{X}|\boldsymbol{Y},\boldsymbol{\theta}^{(r)}}\left\{\log p\left(\boldsymbol{X},\boldsymbol{\gamma}\right)\right\}$ in (11). We note that

$$\log p\left(\boldsymbol{X},\boldsymbol{\gamma}\right)=\sum_{k=1}^{K}\log p\left(\boldsymbol{x}_k|\boldsymbol{x}_{k-1},\boldsymbol{\gamma}\right). \tag{13}$$

We derive the distribution $p\left(\boldsymbol{x}_k|\boldsymbol{x}_{k-1},\boldsymbol{\gamma}\right)$ using the Gaussian prior assumption of the SBL framework and the first-order autoregressive model in (1). Thus, the distribution of $\boldsymbol{x}_k$ conditioned on $\boldsymbol{x}_{k-1}$ is given as follows:

$$\boldsymbol{x}_k|\boldsymbol{x}_{k-1}\sim\mathcal{N}(\boldsymbol{D}\boldsymbol{x}_{k-1},\mathsf{Diag}\left\{\boldsymbol{\gamma}\right\}). \tag{14}$$

As a result, we simplify the optimization problem in (11) as follows:

$$\begin{aligned}\boldsymbol{\gamma}^{(r+1)}&\in\underset{\boldsymbol{\gamma}\in\mathbb{R}_+^N}{\arg\max}\sum_{k=1}^{K}\mathbb{E}_{\boldsymbol{X}|\boldsymbol{Y},\boldsymbol{\theta}^{(r)}}\left\{\log p\left(\boldsymbol{x}_k|\boldsymbol{x}_{k-1},\boldsymbol{\gamma}\right)\right\}\\&=\underset{\boldsymbol{\gamma}\in\mathbb{R}_+^N}{\arg\min}\;K\log|\mathsf{Diag}\left\{\boldsymbol{\gamma}\right\}|+\sum_{k=1}^{K}\mathsf{Tr}\left\{\mathsf{Diag}\left\{\boldsymbol{\gamma}\right\}^{-1}\boldsymbol{C}_k^{(r)}\right\},\end{aligned} \tag{15}$$

where $\mathsf{Tr}\left\{\cdot\right\}$ denotes the trace and we define the matrix $\boldsymbol{C}_k^{(r)}\in\mathbb{R}^{N\times N}$ in (17) as

$$\begin{aligned}\boldsymbol{C}_k^{(r)}&=\mathbb{E}_{\boldsymbol{X}|\boldsymbol{Y},\boldsymbol{\theta}^{(r)}}\left\{\left(\boldsymbol{x}_k-\boldsymbol{D}\boldsymbol{x}_{k-1}\right)\left(\boldsymbol{x}_k-\boldsymbol{D}\boldsymbol{x}_{k-1}\right)^{\mathsf{T}}\right\} &(16)\\&=\boldsymbol{P}_k^{(r)}+\widehat{\boldsymbol{x}}_k^{(r)}\left(\widehat{\boldsymbol{x}}_k^{(r)}\right)^{\mathsf{T}}-2\boldsymbol{D}\left[\boldsymbol{P}_{k,k-1}^{(r)}+\widehat{\boldsymbol{x}}_k^{(r)}\left(\widehat{\boldsymbol{x}}_{k-1}^{(r)}\right)^{\mathsf{T}}\right]\\&\quad+\boldsymbol{D}\left[\boldsymbol{P}_{k-1}^{(r)}+\widehat{\boldsymbol{x}}_{k-1}^{(r)}\left(\widehat{\boldsymbol{x}}_{k-1}^{(r)}\right)^{\mathsf{T}}\right]\boldsymbol{D}^{\mathsf{T}}. &(17)\end{aligned}$$

Here, $\widehat{\boldsymbol{x}}_{k-1}^{(r)}\in\mathbb{R}^N$, $\boldsymbol{P}_k^{(r)}\in\mathbb{R}^{N\times N}$, and $\boldsymbol{P}_{k,k-1}^{(r)}\in\mathbb{R}^{N\times N}$ denote the state statistics computed using the current estimate of the hyperparameter $\boldsymbol{\theta}^{(r)}$:

$$\widehat{\boldsymbol{x}}_k^{(r)}=\mathbb{E}\left\{\boldsymbol{x}_k\middle|\boldsymbol{Y},\boldsymbol{\theta}^{(r)}\right\} \tag{18}$$

$$\boldsymbol{P}_k^{(r)}=\mathbb{E}\left\{\left(\boldsymbol{x}_k-\widehat{\boldsymbol{x}}_k^{(r)}\right)\left(\boldsymbol{x}_k-\widehat{\boldsymbol{x}}_k^{(r)}\right)^{\mathsf{T}}\middle|\boldsymbol{Y},\boldsymbol{\theta}^{(r)}\right\} \tag{19}$$

$$\boldsymbol{P}_{k,k-1}^{(r)}=\mathbb{E}\left\{\left(\boldsymbol{x}_k-\widehat{\boldsymbol{x}}_k^{(r)}\right)\left(\boldsymbol{x}_{k-1}-\widehat{\boldsymbol{x}}_{k-1}^{(r)}\right)^{\mathsf{T}}\middle|\boldsymbol{Y},\boldsymbol{\theta}^{(r)}\right\}. \tag{20}$$

The optimization problem (15) is separable in the entries of $\boldsymbol{\gamma}$, and its closed form solution takes the following form:

$$\boldsymbol{\gamma}_k^{(r+1)}=\frac{1}{K}\sum_{k=1}^{K}\mathsf{Diag}\left\{\boldsymbol{C}_k^{(r)}\right\}. \tag{21}$$

Further, relying on the Gaussian assumption, we compute the quantities $\widehat{\boldsymbol{x}}_{k-1}$, $\boldsymbol{P}_k^{(r)}$, and $\boldsymbol{P}_{k,k-1}^{(r)}$ using fixed interval Kalman smoothing [17] and its pseudo-code is given in Algorithm 1.

Algorithm 1 Fixed interval Kalman smoothing algorithm

Input: $\boldsymbol{\gamma}^{(r)},\boldsymbol{z}^{(r)},\boldsymbol{Y},\boldsymbol{D},\boldsymbol{A},\sigma^2$
initialize: $\widehat{\boldsymbol{x}}_{0|0}=\boldsymbol{0},\boldsymbol{P}_{0|0}=\boldsymbol{0},\boldsymbol{P}_{1,0}^{(r)}=\boldsymbol{0}$
for $k=1,2,\ldots,K$ **do**
#Prediction:
  $\widehat{\boldsymbol{x}}_{k|k-1}=\boldsymbol{D}\widehat{\boldsymbol{x}}_{k-1|k-1}$
  $\boldsymbol{P}_{k|k-1}=\boldsymbol{D}\boldsymbol{P}_{k-1|k-1}\boldsymbol{D}+\mathsf{Diag}\left\{\boldsymbol{\gamma}^{(r)}\right\}$
#Filtering:
  $\boldsymbol{J}=\boldsymbol{z}_k^{(r)}\boldsymbol{P}_{k|k-1}\boldsymbol{A}^{\mathsf{T}}\left(\boldsymbol{A}\boldsymbol{P}_{k|k-1}\boldsymbol{A}^{\mathsf{T}}+\sigma^2\boldsymbol{I}\right)^{-1}$
  $\widehat{\boldsymbol{x}}_{k|k}=(\boldsymbol{I}-\boldsymbol{J}\boldsymbol{A})\widehat{\boldsymbol{x}}_{k|k-1}+\boldsymbol{J}\boldsymbol{y}_k$
  $\boldsymbol{P}_{k|k}=(\boldsymbol{I}-\boldsymbol{J}\boldsymbol{A})\boldsymbol{P}_{k|k-1}$
end for
$\boldsymbol{P}_{K,K-1}^{(r)}=(\boldsymbol{I}-\boldsymbol{J}\boldsymbol{A})\,\boldsymbol{D}\boldsymbol{P}_{K-1|K-1}$
for $k=K,K-1,\ldots,2$ **do**
#Smoothing:
  $\boldsymbol{G}=\boldsymbol{P}_{k-1|k-1}\boldsymbol{D}\boldsymbol{P}_{k|k-1}^{-1}$
  $\widehat{\boldsymbol{x}}_{k-1}^{(r)}=\widehat{\boldsymbol{x}}_{k-1|k-1}+\boldsymbol{G}(\widehat{\boldsymbol{x}}_k-\widehat{\boldsymbol{x}}_{k|k-1})$
  $\boldsymbol{P}_{k-1}^{(r)}=\boldsymbol{P}_{k-1|k-1}+\boldsymbol{G}(\boldsymbol{P}_k-\boldsymbol{P}_{k|k-1})\boldsymbol{G}^{\mathsf{T}}$
#Computing cross-covariance:
  if $k\neq K$ **then**
    $\boldsymbol{P}_{k,k-1}^{(r)}=\boldsymbol{P}_{k|k}\boldsymbol{G}^{\mathsf{T}}+\boldsymbol{G}\left(\boldsymbol{P}_{k+1,k}-\boldsymbol{D}\boldsymbol{P}_{k|k}\right)\boldsymbol{G}^{\mathsf{T}}$
  end if
end for
Output: $\left\{\widehat{\boldsymbol{x}}_k^{(r)},\boldsymbol{P}_k^{(r)},\boldsymbol{P}_{k,k-1}^{(r)}\right\}_{k=1}^{K}$

B. Estimation of $\boldsymbol{z}^{(r+1)}$

We next simplify the optimization problem in (12), and we begin with the distribution $p\left(\boldsymbol{Y}|\boldsymbol{X},\boldsymbol{z}\right)$. From (2), we note that

$$\log p\left(\boldsymbol{Y}|\boldsymbol{X},\boldsymbol{z}\right)=\sum_{k=1}^{K}\log p\left(\boldsymbol{y}_k|\boldsymbol{x}_k,\boldsymbol{z}_k\right). \tag{22}$$

Also, the conditional distribution of $\boldsymbol{y}_k$ given $\boldsymbol{X}$ and $\boldsymbol{z}_k$ is $\boldsymbol{y}_k|\boldsymbol{X},\boldsymbol{z}\sim\mathcal{N}(\boldsymbol{z}_k\boldsymbol{D}\boldsymbol{x}_k,\sigma^2\boldsymbol{I})$. Thus, (12) is equivalent to

$$\boldsymbol{z}^{(r+1)}\in\underset{\boldsymbol{z}\in\{0,1\}^K}{\arg\max}\;\log p\left(\boldsymbol{z}\right)+\sum_{k=1}^{K}\boldsymbol{z}_k\lambda_k^{(r)} \tag{23}$$

$$=\underset{\boldsymbol{z}\in\{0,1\}^K}{\arg\max}\;p\left(\boldsymbol{z}\right)\prod_{k=1}^{K}\exp\left(\boldsymbol{z}_k\lambda_k^{(r)}\right). \tag{24}$$

Here, we define $\lambda_k^{(r)}\in\mathbb{R}$ as

$$\lambda_k^{(r)}=\boldsymbol{y}_k^{\mathsf{T}}\boldsymbol{D}\widehat{\boldsymbol{x}}_k^{(r)}-\frac{1}{2}\mathsf{Tr}\left\{\boldsymbol{D}\left[\boldsymbol{P}_{k-1}^{(r)}+\widehat{\boldsymbol{x}}_{k-1}^{(r)}\left(\widehat{\boldsymbol{x}}_{k-1}^{(r)}\right)^{\mathsf{T}}\right]\boldsymbol{D}^{\mathsf{T}}\right\}, \tag{25}$$

where $\widehat{\boldsymbol{x}}_{k-1}^{(r)}$ and $\boldsymbol{P}_{k-1}^{(r)}$ are defined in (18) and (19), respectively, and they are obtained using Algorithm 1.

The optimization problem in (24) is equivalent to finding the ML estimate of the state sequence of the HMM in Fig. 1 where the probability of observed data $p(\boldsymbol{y}_k|\boldsymbol{X},\boldsymbol{z}_k)\propto\exp\left(\boldsymbol{z}_k\lambda_k^{(r)}\right)$. An efficient method to solve this problem is using the dynamic programming algorithm called the *Viterbi algorithm* [18]. The algorithm computes two quantities $\boldsymbol{v}_{0,k}$

Algorithm 2 Viterbi Algorithm

Input: $p_0, p_1, \lambda_k, k = 1, 2, \ldots, K$

initialize: $\boldsymbol{v}_1^{(0)} = p_0, \boldsymbol{v}_1^{(1)} = p_1 \exp\left(\lambda^{(r)}\right)$

for $k = 2, 3, \ldots, K$ **do**

#Viterbi path probabilities

$\boldsymbol{v}_{0,k} = \max_{j=0,1} \boldsymbol{v}_{j,k-1} p_0^{1-j}(1-p_0)^j$

$\boldsymbol{v}_{1,k} = \exp\left(\lambda_k^{(r)}\right) \max_{j=0,1} \boldsymbol{v}_{j,k-1} p_1^j (1-p_1)^{1-j}$

#Back pointers

$\boldsymbol{b}_{0,k} \in \arg\max_{j=0,1} \boldsymbol{v}_{j,k-1} p_0^{1-j}(1-p_0)^j$

$\boldsymbol{b}_{1,k} \in \arg\max_{j=0,1} \boldsymbol{v}_{j,k-1} p_1^j (1-p_1)^{1-j}$

end for

#Backtracing

$\boldsymbol{z}_K^{(r+1)} = \arg\max_{j=0,1} \boldsymbol{v}_{j,K}$

for $k = K-1, K-2, \ldots, 1$ **do**

$\boldsymbol{z}_k^{(r+1)} = \boldsymbol{b}_{\boldsymbol{z}_{k+1}^{(r+1)}, k+1}$

end for

Output: $\boldsymbol{z}^{(r)}$

and $\boldsymbol{v}_{1,k}$ by recursively taking the most probable path that can lead $\boldsymbol{z}_k$ to 0 and 1, respectively. For $i = 0, 1$, we obtain

$$\boldsymbol{v}_{i,k} = \max_{\{j_l \in \{0,1\}\}_{l=1}^{k-1}} p\left(\{\boldsymbol{y}_l, \boldsymbol{z}_l = j_l\}_{l=1}^{k-1}, \boldsymbol{y}_k, \boldsymbol{z}_k = i \middle| \boldsymbol{X}\right) \tag{26}$$

$$= \max_{j \in \{0,1\}} \boldsymbol{v}_{j,k-1} p\left(\boldsymbol{y}_k | \boldsymbol{z}_k = i, \boldsymbol{X}\right) p(\boldsymbol{z}_k = i | \boldsymbol{z}_{k-1} = j) \tag{27}$$

$$= \exp\left(i\lambda_k^{(r)}\right) \max_{j \in \{0,1\}} \boldsymbol{v}_{j,k-1} p(\boldsymbol{z}_k = i | \boldsymbol{z}_{k-1} = j). \tag{28}$$

Here, the transition probability $p\left(\boldsymbol{z}_k = i | \boldsymbol{z}_{k-1} = j\right)$ is defined by the HMM in Fig. 1. The best sequence $\boldsymbol{z}^{(r)}$ is obtained by keeping track of the path of hidden states that lead to each state (back pointers), and then at the end, backtracing the best path from $k = K$ to 1. The pseudocode of the Viterbi algorithm is given in Algorithm 2.

C. Overall Algorithm and Complexity

Combining Kalman smoothing and the Viterbi algorithm to solve the optimization problem in the M-step, we obtain the overall Bayesian algorithm as follows. In every iteration, we first use Kalman filtering and smoothing as given in Algorithm 1 to compute the statistical measures of the states $\boldsymbol{x}_k$. These statistical measures are used to update the estimates of $\boldsymbol{\gamma}$ and $\boldsymbol{z}$ using (21) and Algorithm 2, respectively. The pseudocode of the overall algorithm is given in Algorithm 3.

We next look at the computational complexity of an iteration of the algorithm (the loop `repeat` in Step 3 of Algorithm 3). The computational complexities of Kalman smoothing in Step 4 and the Viterbi algorithm in Step 8 of Algorithm 3 are $\mathcal{O}(K(N^2m + m^3))$ and $\mathcal{O}(K)$, respectively. The other steps (computation of (17), (25) and (21)) have complexity $\mathcal{O}(KN^2)$. So, the overall complexity of the algorithm is $\mathcal{O}(KN^2m)$ when $m < N$. Further, the memory requirement of the algorithm is $\mathcal{O}(KN^2)$ since we store K covariance matrices of size $N \times N$ in each iteration.

Algorithm 3 Bayesian algorithm for recovering sparse vectors with missing data

Input: $\boldsymbol{Y}, \boldsymbol{D}, \boldsymbol{A}, \sigma^2\boldsymbol{I}, p_0, p_1$

Initialize: $\boldsymbol{\gamma}^{(1)} = \mathbf{1}, \boldsymbol{z}^{(1)} = \mathbf{1}, r = 1$

parameters: $\epsilon > 0$ (stopping time)

repeat

#E-step:

Compute $\left\{\widehat{\boldsymbol{x}}_k^{(r)}, \boldsymbol{P}_k^{(r)}, \boldsymbol{P}_{k,k-1}^{(r)}\right\}_{k=1}^K$ using Algorithm 1 with input as $\left(\boldsymbol{\gamma}^{(r)}, \boldsymbol{z}^{(r)}, \boldsymbol{Y}, \boldsymbol{D}, \boldsymbol{A}, \sigma^2\right)$

Compute $\boldsymbol{C}_k^{(r)}$ using (17) for $k = 1, 2, \ldots, K$

Compute $\lambda_k^{(r)}$ using (25) for $k = 1, 2, \ldots, K$

#M-step:

Update $\boldsymbol{\gamma}_k^{(r+1)}$ using (21)

Update $\boldsymbol{z}^{(r+1)}$ using Algorithm 2 with input as $(p_0, p_1, \lambda_k, k = 1, 2, \ldots, K)$

until $\left\|\boldsymbol{\gamma}_k^{(r+1)} - \boldsymbol{\gamma}_k^{(r)}\right\| + \left\|\boldsymbol{z}^{(r+1)} - \boldsymbol{z}^{(r)}\right\| < \epsilon$

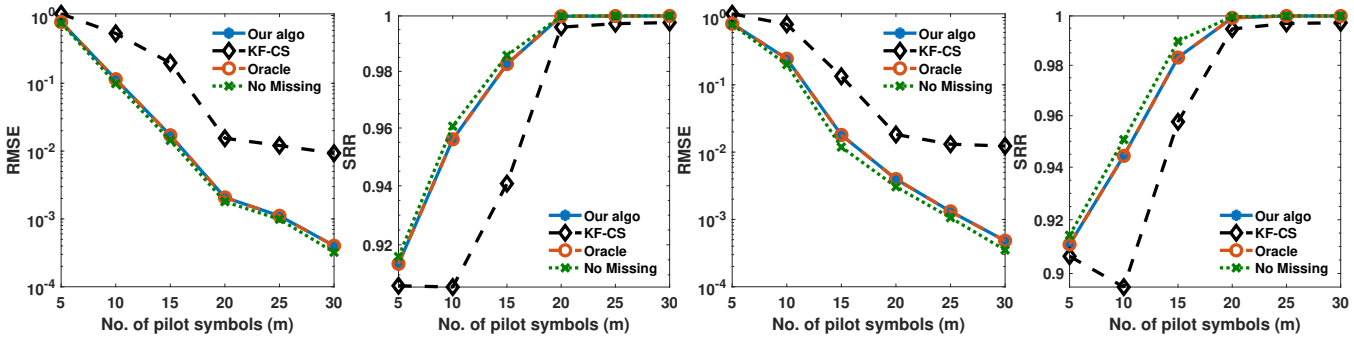
Output: $\widehat{\boldsymbol{x}}_k^{(r)}$

IV. Simulation Results

This section illustrates the utility of our algorithm using the task of FDD MIMO downlink channel estimation via feedback. Our simulation setting is similar to that in [9]. We consider a massive MIMO base station with a uniform linear array of $N = 128$ antennas and spacing of half-wavelength. The base station serves users with a single antenna and employs orthogonal frequency division multiplexing (OFDM) of size 2048. Also, the carrier frequency is $f_c = 2$ GHz.

The wireless channel $\boldsymbol{x}_k$ between the base station and user exhibits a small angle spread of 10°. Therefore, the channel is sparse in the virtual angular domain with sparsity level $\lceil 10^\circ / (180^\circ/N) \rceil = 8$. Moreover, the support of the sparse channels remains unchanged over multiple time blocks. Further, the nonzero entries of the channel vary slowly over time, according to the Jakes model. Therefore, the channel vector $\boldsymbol{x}_k$ obeys (1) where $\boldsymbol{D} = \rho\boldsymbol{I}$ and ρ is the Doppler frequency shift dependent correlation coefficient. We assume that the maximum mobile velocity of the supported users is $v = 27$ km/hr, and thus, the maximum Doppler frequency shift is $vf_c/c = 50$ Hz where c is the speed of light.

In each time slot, the base station transmits a non-orthogonal and randomly placed pilot $\boldsymbol{s}$ of size $m = 30$ to the user. The user directly feeds back the received pilot signal to the base station via a bursty channel. Thus, the feedback signal received at the base station is given by (2) where $\boldsymbol{A} = \mathsf{Diag}\{\boldsymbol{s}\}\boldsymbol{F}$ where $\boldsymbol{F}$ is the $m \times N$ submatrix of discrete Fourier transform matrix with m rows corresponding to the pilot placement. The number of OFDM symbols $K = 150$. The busty feedback channel is modeled using a Markov chain as shown in Fig. 1 with $\boldsymbol{z}_1^* = 1$, $p_0 = 0.3$ and $p_1 = 0.75$ and 0.5.



(a) $p_1 = 0.75$ (b) $p_1 = 0.75$ (c) $p_1 = 0.5$ (d) $p_1 = 0.5$

Figure 2. Comparison of RMSE and SRR of our algorithm with the competing schemes when m and p_1 are varied and $N = 128, K = 150$ and $p_0 = 0.3$.

To the best of our knowledge, ours is the first sparse vector recovery algorithm with data missing at unknown indices. So, we benchmark its performance using three algorithms which assume the knowledge of the missing indices: Kalman filtered compressive sensing (KF-CS) with intermittent observations [13], an oracle version of our algorithm with known missing indices (labeled as `Oracle`), and the oracle version with no missing data (labeled as `No missing`). For comparison, we use two metrics: relative mean square error RMSE $= \frac{\sum_{k=1}^{K} \|\hat{\boldsymbol{x}}_k - \boldsymbol{x}_k\|^2}{\sum_{k=1}^{K} \|\boldsymbol{x}_k\|^2}$ and support recovery rate SRR $= 1 - \frac{1}{KN} \sum_{k=1}^{K} |\text{Supp}\{\hat{\boldsymbol{x}}_k - \boldsymbol{x}_k\}|$, where $\hat{\boldsymbol{x}}_k$ and $\boldsymbol{x}_k$ denote the estimate and true value of the sparse vectors. Here, SRR indicates the quality of support recovery.

The results are shown in Fig. 2 which indicate that our algorithm performs better than KF-CS that assumes the knowledge of the missing indices. This behavior is due to two reasons: one, the superior sparse recovery performance of the SBL framework compared to the basis pursuit denoising algorithm used in the KF-CS algorithm for sparse recovery; and two, the optimality of the Kalman smoothing scheme used in our algorithm compared to the Kalman filtering scheme used in KF-CS. We also notice that our scheme's performance is similar to that of the oracle version, which implies that the missing indices are correctly identified. Further, the performance degradation due to missing data is not significant because of high temporal correlation (modeled using ρ). However, the performance deteriorates as p_1 decreases, as expected.

V. Conclusion

We estimated the state of a first-order linear system whose inputs are jointly sparse. The output was missing at several unknown locations, wherein the missing mechanism was governed using a Markov-modulated Bernoulli random variable. The estimation algorithm used the SBL framework to model joint sparsity, the Kalman smoothing algorithm to handle temporal correlation, and the Viterbi algorithm to estimate the missing data indices. Our results demonstrated the algorithm's performance compared to the state-of-the-art methods when applied to the wireless channel estimation problem. The algorithm's theoretical analysis and extension to a delay-tolerant setting are promising directions for future work.

References

[1] H. Song, T. Liu, X. Luo, and G. Wang, "Feedback based sparse recovery for motion tracking in RF sensor networks," in *Proc. IEEE Int. Conf. Netw. Archit. Storage*, Jul. 2011, pp. 203–207.

[2] M. Roughan, Y. Zhang, W. Willinger, and L. Qiu, "Spatio-temporal compressive sensing and internet traffic matrices (extended version)," *IEEE/ACM Trans. Networking*, vol. 20, no. 3, pp. 662–676, Oct. 2011.

[3] S. Rallapalli, L. Qiu, Y. Zhang, and Y.-C. Chen, "Exploiting temporal stability and low-rank structure for localization in mobile networks," in *Proc. Int. Conf. Mobile Comput. Netw.*, Sep. 2010, pp. 161–172.

[4] Z. Li, Y. Zhu, H. Zhu, and M. Li, "Compressive sensing approach to urban traffic sensing," in *Proc. Int. Conf. Distrib. Comput. Syst.*, Jun. 2011, pp. 889–898.

[5] S. Ji, Y. Sun, and J. Shen, "A method of data recovery based on compressive sensing in wireless structural health monitoring," *Math. Probl. Eng.*, vol. 2014, Jan. 2014.

[6] V. S. G. Thadikemalla and A. S. Gandhi, "A simple and efficient data loss recovery technique for SHM applications," *Smart Mater. Struct.*, vol. 20, no. 1, pp. 35–42, Jul. 2017.

[7] M. Carlavan, L. Blanc-Féraud, M. Antonini, C. Thiebaut, C. Latry, and Y. Bobichon, "A satellite imaging chain based on the compressed sensing technique," in *Proc. On-Board Payload Data Compress. Workshop*, Oct. 2012.

[8] S. Scalise, H. Ernst, and G. Harles, "Measurement and modeling of the land mobile satellite channel at Ku-band," *IEEE Trans. Veh. Technol.*, vol. 57, no. 2, pp. 693–703, Mar. 2008.

[9] Z. Gao, L. Dai, Z. Wang, and S. Chen, "Spatially common sparsity based adaptive channel estimation and feedback for FDD massive MIMO," *IEEE Trans. Signal Process.*, vol. 63, no. 23, pp. 6169–6183, Jul. 2015.

[10] Z. Charbiwala, S. Chakraborty, S. Zahedi, T. He, C. Bisdikian, Y. Kim, and M. B. Srivastava, "Compressive oversampling for robust data transmission in sensor networks," in *Proc. INFOCOM*, Mar. 2010, pp. 1–9.

[11] G. Joseph and P. K. Varshney, "Measurement bounds for compressed sensing with missing data," in *Proc. IEEE SPAWC*, May 2020, pp. 1–5.

[12] ——, "Measurement bounds for compressed sensing in sensor networks with missing data," *IEEE Trans. Signal Process.*, vol. 69, pp. 905–916, Jan. 2021.

[13] H. S. Karimi and B. Natarajan, "Kalman filtered compressive sensing with intermittent observations," *Signal Process.*, vol. 163, pp. 49–58, Oct. 2019.

[14] M. E. Tipping, "Sparse Bayesian learning and the relevance vector machine," *J. Mach. Learn. Res.*, vol. 1, pp. 211–244, Jun 2001.

[15] D. P. Wipf and B. D. Rao, "Sparse Bayesian learning for basis selection," *IEEE Trans. Signal Process.*, vol. 52, no. 8, pp. 2153–2164, Jul. 2004.

[16] ——, "An empirical Bayesian strategy for solving the simultaneous sparse approximation problem," *IEEE Trans. Signal Process.*, vol. 55, no. 7, pp. 3704–3716, Jun. 2007.

[17] B. D. Anderson and J. B. Moore, *Optimal filtering*. Courier Corporation, 2012.

[18] G. D. Forney, "The Viterbi algorithm," *Proc. IEEE Inst. Electr. Electron. Eng.*, vol. 61, no. 3, pp. 268–278, Mar. 1973.