Column Subset Selection with Missing Data

Laura Balzano and Robert Nowak
University of Wisconsin-Madison
{sunbeam, nowak}@ece.wisc.edu

Waheed U. Bajwa
Duke University, Durham, NC
w.bajwa@duke.edu

Abstract

An important problem in massive data collection systems is to identify a representative subset of collection points from a given set of measurements. This problem is well cast in the literature as selecting representative columns from a low-rank data matrix. However, a challenge that arises immediately is that complete data are typically not available or are infeasible to collect in very large systems. This paper addresses the challenge of selecting representative columns from a low-rank matrix with missing data by formulating the problem as the solution to a system of linear equations with a "block-sparsity" constraint. The key distinguishing characteristic of our formulation is that it operates only on the matrix entries that have been observed. Simulation results confirm that our problem formulation together with a block variant of the *orthogonal matching pursuit* algorithm often outperforms *rank-revealing QR* factorization, the standard choice for column subset selection, run on a zero-filled data matrix.

1 Introduction

The problem of selecting k representative columns from a low-rank $m \times n$ matrix $\boldsymbol{Y}$, or *column subset selection* (CSS), arises frequently in data applications involving large data sets. Consider, for example, the case of an internet traffic measurement system in which a data matrix is constructed by collecting router traffic loads for various traffic types at different times into columns. In this application, CSS is akin to identifying representative routers that can lead to quick tracking down of network anomalies. Similarly, consider the case of a movie recommendation system in which a data matrix is constructed by collecting movie ratings of users into columns. In this application, CSS leads to identification of users who can reliably predict the ratings of newly-released movies. There are numerous other uses of the CSS problem in data applications and we refer the reader to some of the references within [2] for further motivation.

The CSS problem has been well-studied in the literature. The main algorithm that is put forth to solve this problem is *rank-revealing QR* (RRQR) decomposition [4, 6]. In particular, it has been shown that RRQR and its variants solve the CSS problem in a near-optimal fashion; see, e.g., [2, 6] and Table 1 within [2]. Unfortunately, it is nearly impossible in many data applications involving massive data sets to have access to the complete data. In the case of the internet traffic measurement system, for example, it is quite common for the routers to lose traffic data at various points in time. Similarly, in the case of the movie recommendation system, no user can be expected to rate every movie in the database. Existing formulations of the CSS problem in the literature seem incapable of handling the case of missing data in an intelligible fashion.

In this paper, we take a unified approach that naturally leads to algorithms for CSS with or without having access to full data. Our approach is in part motivated by recent results that show that incomplete data matrices can be reliably completed if they meet certain low-rank and incoherence assumptions [3, 7]. The problem of CSS is in some sense a detection problem, rather than an estimation problem, and therefore we expect it to remain well-posed even when a larger fraction of data is missing. Our main contribution in this regard is that we adapt the original optimization-based formulation of the CSS problem to accommodate situations when data are missing. The key distin-

guishing characteristic of our formulation is that it operates only on the matrix entries that have been observed, which is justified by insights from our recent work on subspace detection with missing data [1]. In addition, the work presented in here also opens the door to low-complexity CSS algorithms in terms of both computation and memory requirements for the case when a large fraction of data are missing.

2 Formulation of Column Subset Selection with Missing Data

Consider an $m \times n$ matrix $\boldsymbol{Y}$ whose n columns represent n sensors or measurement points and whose m rows represent measurement times. In this paper, we consider that $\boldsymbol{Y}$ is exactly low-rank and we are interested in choosing a subset of the columns of $\boldsymbol{Y}$ to represent the entire matrix. If we know the number of columns we would like to select, k, we can state CSS formally by saying that we want to find a matrix $\boldsymbol{X}$ such that $\boldsymbol{Y} = \boldsymbol{Y}\boldsymbol{X}$ while only k or fewer rows of $\boldsymbol{X}$ are non-zero. This problem is combinatorial; we may need to check all size-k subsets to find a suitable set. However, there are greedy algorithms that aim to solve this problem, and the one we use in this paper is *Block Orthogonal Matching Pursuit*, or BOMP. Block OMP can refer to an algorithm which solves a problem set up where dictionary elements are blocked [5]; in our case we have individual dictionary elements (the columns of our matrix) but require that the support on the dictionary is the same for every other column [8].

Rank-Revealing QR (RRQR) [4, 6] decompositions have been studied extensively in the literature for CSS [2]. These algorithms aim to find $\boldsymbol{Q}$, $\boldsymbol{R}$ and a permutation matrix $\boldsymbol{\Pi}$ such that $\boldsymbol{Y}\boldsymbol{\Pi} = \boldsymbol{Q}\boldsymbol{R}$. The permutation matrix essentially permutes the most well-conditioned columns to the front of the matrix; given k, the first k are the selected columns when RRQR is used for column subset selection.

An adaptation of RRQR for matrices with missing data has not been developed; it is not clear to the authors whether a non-heuristic algorithm could be developed. A naïve approach would be to simply fill the matrix with zeros where data are missing. However, zero-filling a column vector changes the subspace of that vector [1] and can cause problems in selection, as we show in Section 3.

Missing Data Problem Formulation Often in applications data are missing and the matrix $\boldsymbol{Y}$ is incomplete. We can manipulate the problem given above for the situation when not all the data are observed. First we note that $\exists \boldsymbol{X}$ such that $\boldsymbol{Y} = \boldsymbol{Y}\boldsymbol{X}$ iff $\|\boldsymbol{Y} - \boldsymbol{Y}\boldsymbol{X}\|_F^2 = 0$. We now will manipulate this norm for the case where there are missing entries in $\boldsymbol{Y}$.

Let $\mathcal{C}_j = \{i : \text{The } i^{th} \text{ entry of column } j \text{ of } \boldsymbol{Y} \text{ is observed}\}$ and $\mathcal{R}_j = \{i : \text{The } i^{th} \text{ entry of row } j \text{ of } \boldsymbol{Y} \text{ is observed}\}$. Also let $\boldsymbol{Y}_{ij}$ represent the element in the i^{th} row and j^{th} column of $\boldsymbol{Y}$. Then we can look at $\|\boldsymbol{Y} - \boldsymbol{Y}\boldsymbol{X}\|_F^2$ considering only elements of $\boldsymbol{Y}$ that have been observed:

$$\sum_{j=1}^{n} \sum_{k \in \mathcal{C}_j} \left(\boldsymbol{Y}_{kj} - \sum_{i \in \mathcal{R}_k} \boldsymbol{X}_{ij} \boldsymbol{Y}_{ki} \right)^2 . \tag{1}$$

Define the zero-filled version of $\boldsymbol{Y}$, and indicator function $\boldsymbol{w}$ which indicates observed entries in $\boldsymbol{Y}$:

$$\widetilde{\boldsymbol{Y}}_{ij} = \begin{cases} \boldsymbol{Y}_{ij} & \text{if } \boldsymbol{Y}_{ij} \text{ is observed;} \\ 0 & \text{if } \boldsymbol{Y}_{ij} \text{ is unobserved.} \end{cases} \qquad \boldsymbol{w}_{ij} = \begin{cases} 1 & \text{if } \boldsymbol{Y}_{ij} \text{ is observed;} \\ 0 & \text{if } \boldsymbol{Y}_{ij} \text{ is unobserved.} \end{cases}$$

Also denote the j^{th} column vector with subscript $*j$ and the Hadamard product with $\circ$. Using these we can rewrite (1) as

$$\sum_{j=1}^{n} \sum_{k=1}^{n} \boldsymbol{w}_{kj} \left(\tilde{\boldsymbol{Y}}_{kj} - \sum_{i=1}^{n} \boldsymbol{X}_{ij} \tilde{\boldsymbol{Y}}_{ki} \right)^2 = \sum_{j=1}^{n} \| \boldsymbol{w}_{*j} \circ \tilde{\boldsymbol{Y}}_{*j} - \boldsymbol{w}_{*j} \circ \left(\tilde{\boldsymbol{Y}} \boldsymbol{X}_{*j} \right) \|_2^2 .$$

We can manipulate this into matrix-vector form. Let $\boldsymbol{W}_i$ be the diagonal matrix with column $\boldsymbol{w}_{*i}$ on the diagonal.

$$\sum_{j=1}^{n} \| \boldsymbol{w}_{*j} \circ \tilde{\boldsymbol{Y}}_{*j} - \boldsymbol{w}_{*j} \circ \left(\tilde{\boldsymbol{Y}} \boldsymbol{X}_{*j} \right) \|_2^2 = \left\| \tilde{\boldsymbol{Y}} - \left[\boldsymbol{W}_1 \tilde{\boldsymbol{Y}} \boldsymbol{X}_{*1}, \boldsymbol{W}_2 \tilde{\boldsymbol{Y}} \boldsymbol{X}_{*2}, \ldots, \boldsymbol{W}_n \tilde{\boldsymbol{Y}} \boldsymbol{X}_{*n} \right] \right\|_F^2 . \tag{2}$$

Now we will use $\boldsymbol{x} = \mathsf{vec}(\boldsymbol{X})$ for the operation of stacking the columns of $\boldsymbol{X}$into a single column vector; $\mathsf{vec}^{-1}(\boldsymbol{x})$ will be defined to undo this operation. Apply vec to the term inside the norm of (2). We first have $\mathsf{vec}(\tilde{\boldsymbol{Y}}) = \tilde{\boldsymbol{y}}$. Let $\boldsymbol{W}_i$ be the diagonal matrix with column $\boldsymbol{w}_{*i}$ on the diagonal. Define the block-diagonal matrix

$$\boldsymbol{A} = \mathsf{diag} \left(\boldsymbol{W}_1 \widetilde{\boldsymbol{Y}}, ..., \boldsymbol{W}_n \widetilde{\boldsymbol{Y}} \right) . \tag{3}$$

Thus (2) becomes $\|\widetilde{\boldsymbol{y}} - \boldsymbol{A}\boldsymbol{x}\|_2^2$. This is 0 iff $\exists \boldsymbol{X}$ such that $\widetilde{\boldsymbol{y}} = \boldsymbol{A}\boldsymbol{x}$. The problem becomes:

$$\text{Find } \boldsymbol{X} \text{ such that } \mathsf{vec}(\widetilde{\boldsymbol{Y}}) = \boldsymbol{A}\mathsf{vec}(\boldsymbol{X}), \text{ requiring that } \boldsymbol{X} \text{ has exactly } k \text{ non-zero rows.} \quad (4)$$

As is the full-data formulation, this problem is combinatorial, but many algorithms can be applied to solve it efficiently. One possible choice for such an algorithm is Block OMP for column subset selection with missing data, which would operate as shown in Algorithm 1. In the case where noise is added to the low-rank matrix $\boldsymbol{Y}$, the Group Lasso [9] can be used to formulate the problem as $\min_{\boldsymbol{x}} \|\widetilde{\boldsymbol{y}} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \lambda \sum_{i=1}^{n} \|x_i\|_2$ where x_i is the i^{th} row of $\boldsymbol{X}$. The parameter λ can then be chosen in such a way to enforce that k rows of $\boldsymbol{X}$ are non-zero.

Algorithm 1 Block OMP for Column Subset Selection with missing data

Require: Samples from an $m \times n$ matrix $\boldsymbol{Y}$ and sample locations, from which we construct $\boldsymbol{A}$ as in (3) and $\mathsf{vec}(\widetilde{\boldsymbol{Y}})$. An integer k, the number of columns to be selected.

1: **Initialize:** $\widetilde{\boldsymbol{y}}_0 \leftarrow \mathsf{vec}(\widetilde{\boldsymbol{Y}})$, $\mathcal{I} \leftarrow \{\}$
2: **for** $t = 1, \ldots, k$ **do**
3: **Back-project and devectorize:** $d = \boldsymbol{A}^T \mathsf{vec}(\widetilde{\boldsymbol{y}}_0)$ and $D = \mathsf{vec}^{-1}(d)$.
4: **Choose max row norm:** $i_t = \arg\min_i \|D_i\|_2$ where D_i refers to the i^{th} row of D.
5: **Add index to selected set:** $\mathcal{I} = \mathcal{I} \cup \{i_t\}$.
6: **Fit to the current index set:** $\theta = \boldsymbol{A}_{\mathcal{I}}^{\dagger} \mathsf{vec}(\widetilde{\boldsymbol{Y}})$ where † denotes the pseudo-inverse.
7: **Update**: $\widetilde{\boldsymbol{y}}_0 = \mathsf{vec}(\widetilde{\boldsymbol{Y}}) - \boldsymbol{A}_{\mathcal{I}}\theta$
8: **end for**
9: **return** $\mathcal{I}$

3 Numerical Results and Discussion

Our new formulation of the CSS problem in (4) operates only on the matrix entries that have been observed and automatically reduces to the traditional problem formulation for the case when no data are missing. In this section, we demonstrate the significance of this formulation by numerically comparing the performance of the BOMP algorithm proposed for solving (4) and the RRQR factorization run naïvely on a zero-filled data matrix. For the sake of this exposition, we implement the RRQR factorization by making use of the implementataion of QR factorization provided in Matlab, which returns an ordering of matrix columns such that all the diagonal entries of $\boldsymbol{R}$ are decreasing.

The numerical experiments reported in here correspond to column selection on a low-rank matrix $\boldsymbol{Y}$ with $m = 150$, $n = 200$, and $k = 4$. In order to built the matrix $\boldsymbol{Y}$, we first generate four standard Gaussian column vectors and then orthonormalize them. Next, we generate four sets of columns from each of these generating vectors such that each column in a set is a random scaling of the corresponding generating vector. The matrix $\boldsymbol{Y}$ then corresponds to collecting these four sets of columns into a matrix followed by a random permutation of the locations of these columns. We define success as picking one column from each of the four sets. The first set of numerical experiments that we carried out corresponded to the case of complete data (not shown in here). In this case, both BOMP and RRQR picked exactly the same columns in every run: the largest-norm column from each set. This remained true over numerous runs regardless of the cardinality of the sets of columns.

Next, we carried out numerical experiments corresponding to the case of missing data. For this purpose, we randomly erased 10% to 90% of the mn entries of $\boldsymbol{Y}$ and also varied the cardinality of the column set sizes. We varied the first set from $0.05n$ to $0.75n$, and then we split the remaining columns evenly among the three remaining sets. The results of these experiments, shown in Fig. 1, demonstrate that our problem formulation together with BOMP outperforms RRQR run on a zero-filled data matrix, especially for the case when 30% to 60% of the data are missing, as long as no more than $\sim 70\%$ of the columns are explained by a single column in the matrix.

Two remarks are now in order concerning the insights gained from the numerical experiments and our proposed approach to CSS in the case of missing data. First, note that it is reasonable to expect that for the case when a single column describes a very large fraction of other columns along with missing data, BOMP would perform worse than RRQR with zero-filled matrix. This is because of the greedy nature of Step 4 in the BOMP algorithm that picks columns that best describe the "residual" energy in the remaining columns. Missing data would always leave some residual energy

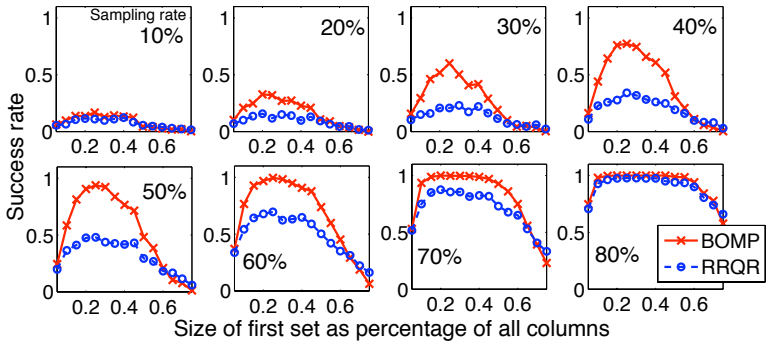

Figure 1: This simulation was done with $n = 200$, $m = 150$, rank of $\boldsymbol{Y} = 4$. Success is defined as choosing exactly one column from each of the four sets.

in the columns, which can accumulate to overshadow the energy in other sets if one single set gets too large. Second, the fact that RRQR with a zero-filled matrix significantly underperforms our problem formulation together with BOMP for the case of 30% to 60% missing data and comparable cardinality of column sets is completely in line [1], which shows that it is important to work only with observed data for subspace detection with missing data, since zero-filling the data in that case leads to false detection of energy outside the true subspace.

We conclude this discussion by pointing out that it is indeed possible to envision alternative, sophisticated approaches to the problem of CSS in the absence of complete data. One immediate choice in this regard could be a two-stage CSS procedure that involves preprocessing of the available data to impute the missing data followed by the RRQR factorization. However, this adds an extra layer of processing that increases the computational complexity of the problem. On the other hand, we have from Figure 1 that perfect CSS can be carried out in the absence of complete data *without* having to impute the missing data. In addition, it is also easy to see that one-stage CSS procedures could in fact take advantage of the missing data to reduce their computational complexity and memory requirements, whereas a two-stage CSS procedure which started with full-matrix imputation would be incapable of doing that. In this regard, the BOMP algorithm proposed in here for solving (4) validates the idea of CSS without missing data imputation. In the future, we plan on making use of our formulation in (4) to devise fast algorithms that are not only uniformly better than RRQR with a zero-filled matrix, but also take advantage of the missing data to reduce overall computational complexity and memory requirements. It is not hard to imagine that this could eventually lead us to situations where one would intentionally throw away data to reduce complexity and storage without sacrificing performance.

References

[1] L. Balzano, B. Recht, and R. Nowak. High-dimensional matched subspace detection when data are missing. In *Proceedings of ISIT*, June 2010.

[2] C. Boutsidis, M. W. Mahoney, and P. Drineas. An improved approximation algorithm for the column subset selection problem. In *Proceedings of SODA*, pages 968–977, 2009.

[3] E. Candès and B. Recht. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, 9(6):717–772, 2009.

[4] T. F. Chan. Rank revealing QR factorizations. *Linear Alg. and its Appl.*, 88-89:67–82, April 1987.

[5] Y. Eldar, P. Kuppinger, and H. Bolcskei. Block-sparse signals: uncertainty relations and efficient recovery. *IEEE Transactions on Signal Processing*, 58:3042–3054, June 2010.

[6] M. Gu and S. C. Eisenstat. Efficient algorithms for computing a strong rank-revealing qr factorization. *SIAM Journal on Scientific Computing*, 17:848–869, July 1996.

[7] B. Recht. A simpler approach to matrix completion. *Journal of Machine Learning Research*. To Appear. Preprint available at `http://arxiv.org/abs/0910.0651`.

[8] J. Tropp, A. Gilbert, and M. Strauss. Algorithms for simultaneous sparse approximation, Part I: Greedy pursuit. *Signal Processing, special issue on Sparse approximations in signal and image processing*, 86:572–588, April 2006.

[9] M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society, Series B*, 68:49–67, 2007.