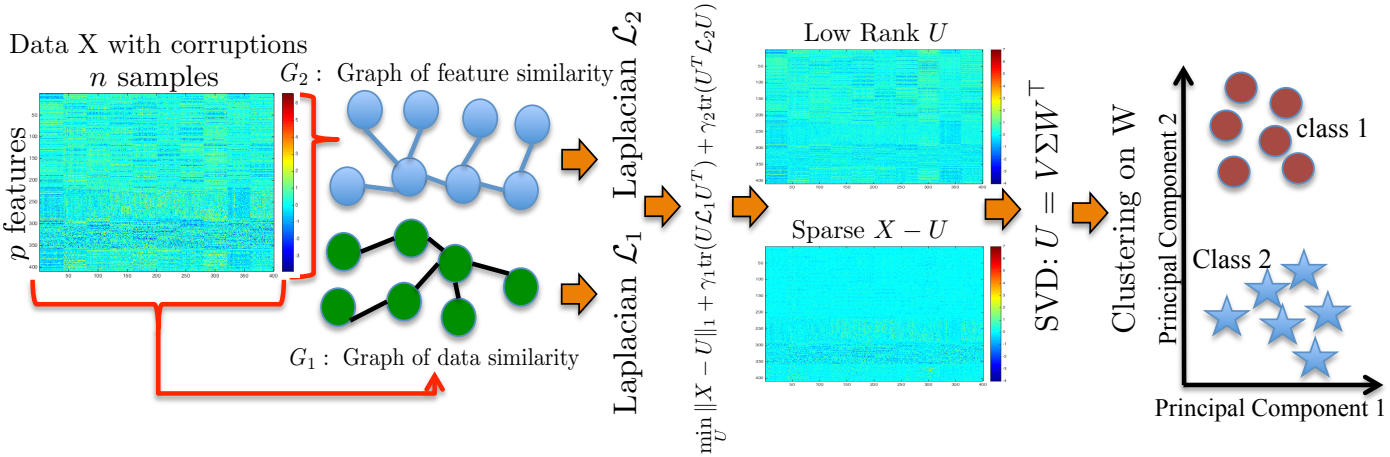
Fast Robust PCA on Graphs

Nauman Shahid*, Nathanael Perraudin, Vassilis Kalofolias, Pierre Vandergheynst

Email: {nauman.shahid, nathanael.perraudin, vassilis.kalofolias, pierre.vandergheynst}@epfl.ch

Signal Processing Laboratory 2 (LTS2), EPFL STI IEL, Lausanne, CH-1015, Switzerland.



Main idea of Fast Robust PCA on Graphs

Abstract

Mining useful clusters from high dimensional data has received significant attention of the computer vision and pattern recognition community in the recent years. Linear and non-linear dimensionality reduction has played an important role to overcome the curse of dimensionality. However, often such methods are accompanied with three different problems: high computational complexity (usually associated with the nuclear norm minimization), non-convexity (for matrix factorization methods) and susceptibility to gross corruptions in the data. In this paper we propose a principal component analysis (PCA) based solution that overcomes these three issues and approximates a low-rank recovery method for high dimensional datasets. We target the low-rank recovery by enforcing two types of graph smoothness assumptions, one on the data samples and the other on the features by designing a convex optimization problem. The resulting algorithm is fast, efficient and scalable for huge datasets with $\mathcal{O}(n\log(n))$ computational complexity in the number of data samples. It is also robust to gross corruptions in the dataset as well as to the model parameters. Clustering experiments on 7 benchmark datasets with different types of corruptions and background separation experiments on 3 video datasets show that our proposed model outperforms 10 state-of-the-art dimensionality reduction models.

Keywords

robust PCA, structured low-rank representation, spectral graph theory, graph regularized PCA

I. Introduction

In the modern era of data explosion, many problems in signal and image processing, machine learning and pattern recognition require dealing with very high dimensional datasets, such as images, videos and web content. The data mining community often strives to reveal natural associations or hidden structures in the data. Over the past couple of decades matrix factorization has

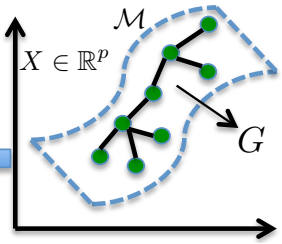
been adopted as one of the key methods in this context. Given a data matrix $X \in \mathbb{R}^{p\times n}$ with n p-dimensional data vectors, the matrix factorization can be stated as determining $V \in \mathbb{R}^{p\times c}$ and $W \in \mathbb{R}^{c\times n}$ such that $X \approx VW$ under different constraints on V and W.

How can matrix factorization extract structures in the data? The answer to this question lies in the intrinsic association of linear dimensionality reduction with matrix factorization. For years, dimensionality reduction has served as an important tool to overcome the curse of dimensionality and extract meaningful clusters or communities from the data. Consider a set of gray-scale images of the same object captured under fixed lighting condition with a moving camera or a set of hand-written numerals with different rotations. Given that the image has m^2 pixels, each such data sample is represented by a point in $\mathbb{R}^{m^2}$. However, the intrinsic dimensionality of the space of all images of the same object captured with small perturbations is much lower than m^2. Thus, dimensionality reduction comes into play. Depending on the application and the type of dataset, one can either use a single linear subspace to approximate the data of different classes using the standard Principal Component Analysis (PCA) [1], a union of low dimensional subspaces where each class belongs to a different subspace (LRR and SSC) [10], [18], [33], [29], or a positive subspace to extract a positive low-rank representation of the data (NMF) [17]. The clustering or community detection can then be performed on the retrieved data representation in the low dimensional space. Not surprisingly, all the above mentioned problems can be stated in the standard matrix factorization manner as shown in the models 1 to 3 of Fig. 1. Alternatively, the clustering quality for non-linearly separable datasets can be improved by using non-linear dimensionality reduction tools such as Laplacian Eigenmaps [4] or Kernel PCA [28].

Standard Matrix Factorization

	Method	Model	Constraints	Dimension Reduction
1	NMF	$\min_{V,W} \|X - VW\|_F^2$	$V \geq 0, W \geq 0, (c < p)$	Positive subspace
2	Sparse Coding (SC)	$\min_{V,W} \|X - VW\|_F^2 + \lambda\|W\|_1$	$\|v_i\|_2 \leq 1 \forall i, (c > p)$	Union of subspaces
3	PCA	$\min_{V,W} \|X - VW\|_F^2$	$V^\top V = I, (c < p)$	Linear subspace
4	RPCA [7]	$\min_{U,S} \|U\|_* + \lambda\|S\|_1$	$X = U + S$	

PCA + Manifold information in Graph Laplacian $\mathcal{L}$



Graph Regularized PCA

	Method	Model	Constraints	Parameters	Graph?	Factors?	Convex?
5	GLPCA [17]	$\min_{V,W} \|X - VW\|_F^2 + \gamma \operatorname{tr}(W \mathcal{L} W^\top)$	$WW^\top = I$				
6	MMF [45]	$\min_{V,W} \|X - VW\|_F^2 + \gamma \operatorname{tr}(W \mathcal{L} W^\top)$	$V^\top V = I$	γ	+	+	-
7	RPCAG [35]	$\min_{U,S} \|U\|_* + \lambda\|S\|_1 + \gamma \operatorname{tr}(U \mathcal{L} U^\top)$	$X = U + S$	λ, γ			
8	**PROPOSED**	$\min_U \|X - U\|_1 + \gamma_1 \operatorname{tr}(U \mathcal{L}_1 U^\top) + \gamma_2 \operatorname{tr}(U^\top \mathcal{L}_2 U)$		γ_1, γ_2	+	-	+

Fig. 1. A summary of the matrix factorization methods with and without graph regularization. $X \in \mathbb{R}^{p\times n}$ is the matrix of n p-dimensional data vectors, $V \in \mathbb{R}^{p\times c}$ and $W \in \mathbb{R}^{c\times n}$ are the learned factors. $U \in \mathbb{R}^{p\times n}$ is the low-rank matrix and $S \in \mathbb{R}^{p\times n}$ is the sparse matrix. $\|\cdot\|_F$, $\|\cdot\|_*$ and $\|\cdot\|_1$ denote the Frobenius, nuclear and ℓ_1 matrix norms respectively. The data manifold $\mathcal{M}$ information can be leveraged in the form of a discrete graph G using the graph Laplacian $\mathcal{L} \in \mathbb{R}^{n\times n}$ resulting in various Graph Regularized PCA models.

Often the low-dimensional data follows some structure which can be leveraged using structured priors [12], [20], [34] and boost the matrix factorization framework. Ever-since the introduction of graph-based priors to represent the data, matrix factorization has emerged as a much powerful tool for the data mining community. Representation of a signal on a graph supported by the well-defined notions of spectral graph theory enables automatic exploitation of additional community structure in the data [31]. The underlying assumption is the same as mentioned previously, i.e, high-dimensional data often lies on or close to a smooth low-dimensional manifold. Interestingly, the low-dimensional manifold information can be exploited via a graph structure between

the data samples. Let $G = (\mathcal{V}, \mathcal{E})$ be a graph between the samples of X, where $\mathcal{E}$ is the set of edges and $\mathcal{V}$ is the set of vertices (data samples). Let A be the symmetric matrix that encodes the adjacency information between the samples of X and D is the diagonal degree matrix of G ($D_{ii} = \sum_j A_{ij}$). Then the normalized graph Laplacian $\mathcal{L}$ to characterize the graph G is defined as $\mathcal{L} = D^{-1/2}(D - A)D^{-1/2}$. Exploiting the manifold information in the form of a graph may be considered as a method of incorporating locality information of the data samples into the dimensionality reduction framework, thus enhancing the clustering quality in low-dimensional space.

In this paper, we focus on the application of PCA to clustering, thus projecting the data on a single linear subspace. We first describe PCA and its related models and then elaborate how the data manifold information in the form of a graph can be used to enhance the standard PCA. Finally, we present a novel, convex, fast and scalable method for PCA that leverages two graph structures.

A. PCA and Related Work

For a dataset $X \in \mathbb{R}^{p \times n}$ with n p-dimensional data vectors, standard PCA learns the projections or principal components $W \in \mathbb{R}^{c \times n}$ of X on a c-dimensional orthonormal basis $V \in \mathbb{R}^{p \times c}$, where $c < p$ by solving model 3 in Fig. 1. Though non-convex, this problem has a global minimum that can be computed using Singular Value Decomposition (SVD), giving a unique low-rank representation $U = VW$.

A main drawback of PCA is its sensitivity to heavy-tailed noise due to the Frobenius norm in the above formulation. Thus, a few strong outliers can result in erratic principal components. Robust PCA (RPCA) proposed by Candes et al. [7] overcomes this problem by recovering the clean low-rank representation U from grossly corrupted X by solving model 4 in Fig. 1. Here S represents the sparse matrix containing the errors and $\|U\|_*$ denotes the nuclear norm of U, the tightest convex relaxation of rank(U).

Recently, many works related to low-rank or sparse representation recovery have been proposed to incorporate the data manifold information in the form of a discrete graph into the dimensionality reduction framework [13], [37], [11], [6], [32], [15], [14], [25], [9]. In fact, for PCA, this can be considered as a method of exploiting the local smoothness information in order to improve clustering quality. The graph smoothness of the principal components W using the graph Laplacian $\mathcal{L}$ has been exploited in various works that explicitly learn W and the basis V. We refer to such models as *factorized models*. In this context Graph Laplacian PCA (GLPCA) was proposed in [13] (model 5 in Fig. 1) and Manifold Regularized Matrix Factorization (MMF) in [37] (model 6 in Fig. 1). Note that the orthonormality constraint in this model is on V, instead of the principal components W.

Later on, the authors of [29] have generalized robust PCA by incorporating the graph smoothness (model 7 in Fig. 1). In their model, they propose a simple graph smoothness regularization term directly on the low-rank matrix instead of principal components and improve the clustering and low-rank recovery properties. They call it Robust PCA on Graphs (RPCAG).

Models 4 to 8 can be used for clustering in the low dimensional space. However, each of them comes with its own weaknesses. GLPCA [13] and MMF [37] improve upon the classical PCA by incorporating graph smoothness but they are non-convex and susceptible to data corruptions. Moreover, the rank c of the subspace has to be specified upfront. RPCAG [29] is convex and builds on the robustness property of RPCA [7] by incorporating the graph smoothness directly on the low-rank matrix and improves both the clustering and low-rank recovery properties of PCA. However, it uses the nuclear norm relaxation that involves an expensive SVD step in every iteration of the algorithm. Although fast methods for the SVD have been proposed, based on randomization [35], [19], [23], Frobenius norm based representations [36], [24] or structured RPCA [2], its use in each iteration makes it hard to scale to large datasets.

B. Our Contributions

In this paper we propose a fast, scalable, robust and convex clustering and low-rank recovery method for potentially corrupted low-rank signals. Our contributions are:

1) We propose an approximate low-rank recovery method for corrupted data by utilizing graph smoothness assumptions both between the samples and between the features.
2) Our model is convex and although non-smooth it can be solved efficiently (in linear time in the number of samples) with a few iterations of the well-known FISTA algorithm. The construction of the two graphs costs $\mathcal{O}(n \log n)$ time, where n is the number of data samples.
3) The resulting algorithm is highly parallelizable and scalable for large datasets since it requires only the multiplication of two sparse matrices with full vectors and elementwise soft-thresholding operations.
4) The recovered close-to-low-rank matrix is a good approximation of the low-rank matrix obtained by solving the expensive state-of-the-art method [29] which uses the much more expensive nuclear norm. This is true even in the presence of gross corruptions in the data.
5) Only one inexpensive SVD computation is needed in the end of our method, in order to obtain the principal components.

The idea of using two graph regularization terms has previously appeared in the work of matrix completion [16]. However, to the best of our knowledge the idea of approximating a low-rank representation with graphs has surfaced for the very first time in this paper. A summary of the notations used in this paper is presented in Tab. II of the supplementary material Section A.

II. Fast Robust PCA on Graphs (FRPCAG)

Let $\mathcal{L}_1 \in \mathbb{R}^{n \times n}$ be the graph Laplacian of the graph G_1 connecting the different samples of X and $\mathcal{L}_2 \in \mathbb{R}^{p \times p}$ the Laplacian of graph G_2 that connects the features of X. The construction of these two graphs is described in Section III. We denote by $U \in \mathbb{R}^{p \times n}$ the low-rank noiseless matrix that needs to be recovered from the measures X, then our proposed model can be written as:

$$\min_U \|X - U\|_1 + \gamma_1 \operatorname{tr}(U \mathcal{L}_1 U^\top) + \gamma_2 \operatorname{tr}(U^\top \mathcal{L}_2 U). \tag{1}$$

This problem can be reformulated in the equivalent split form

$$\begin{aligned} &\min_{U,S} \|S\|_1 + \gamma_1 \operatorname{tr}(U \mathcal{L}_1 U^\top) + \gamma_2 \operatorname{tr}(U^\top \mathcal{L}_2 U), \\ &\text{s.t. } X = U + S, \end{aligned} \tag{2}$$

where S models the sparse outliers in the data X. Model (2) has close connections with the RPCAG [29]. In fact the nuclear norm term in RPCAG has been replaced by another graph Tikhonov term. The two graph regularization terms help in retrieving an approximate low-rank representation U by encoding graph smoothness assumptions on U without using the expensive nuclear norm of RPCAG, therefore we call it Fast Robust PCA on Graphs (FRPCAG). The main idea of our work is summarized in the fig. of the first page of this paper.

The graph terms $\operatorname{tr}(U \mathcal{L}_1 U^\top)$ and $\operatorname{tr}(U^\top \mathcal{L}_2 U^\top)$ encode a weighted penalization in the Laplacian basis. Let $\mathcal{L}_1 = Q \Lambda Q^\top$ and $\mathcal{L}_2 = P \Omega P^\top$ be the spectral decompositions of $\mathcal{L}_1$ and $\mathcal{L}_2$ respectively, where the eigenvalues Λ and Ω are sorted in increasing

order. Let $U = V\Sigma W^\top$ be the SVD of U, then

$$\begin{aligned}
&\gamma_1 \operatorname{tr}(U\,\mathcal{L}_1\,U^\top) + \gamma_2 \operatorname{tr}(U^\top\,\mathcal{L}_2\,U)\\
=&\gamma_1 \operatorname{tr}(V\Sigma W^\top Q\Lambda Q^\top W\Sigma V^\top) + \gamma_2 \operatorname{tr}(W\Sigma V^\top P\Omega P^\top V\Sigma W^\top)\\
=&\gamma_1 \operatorname{tr}(\Sigma W^\top Q\Lambda Q^\top W\Sigma) + \gamma_2 \operatorname{tr}(\Sigma V^\top P\Omega P^\top V\Sigma)\\
=&\gamma_1 \operatorname{tr}(W^\top Q\Lambda Q^T W\Sigma^2) + \gamma_2 \operatorname{tr}(V^\top P\Omega P^\top V\Sigma^2)\\
=&\sum_{i,j=1}^{\min\{n,p\}} \sigma_i^2(\gamma_1\lambda_j(w_i^\top q_j)^2 + \gamma_2\omega_j(v_i^\top p_j)^2),
\end{aligned} \tag{3}$$

where λ_j and ω_j are the eigenvalues in the matrices Λ and Ω respectively. The second step follows from $V^\top V = I$ and the cyclic permutation invariance of the trace. In the standard terminology w_i and v_i are the principal components and principal directions of of the low-rank matrix U. Thus, the first term in the sum encodes a smooth non-linear map towards the principal directions w_i of the underlying manifold defined by the graph $\mathcal{L}_1$. In other words, minimization with respect to the first term forces u_i to be aligned with the eigenvectors q_j of the graph $\mathcal{L}_1$ which correspond to the smallest eigenvalues λ_j. This is the heart of many algorithms in clustering [22] and dimensionality reduction [4]. In our present application, it will force that the principal components of the signals are well represented by a few Laplacian eigenvectors associated to low λ_j.

The role of the second graph can be defined in a similar manner. It uses the underlying structure between the features encoded in $\mathcal{L}_2$. The prior $\operatorname{tr}(U^\top\,\mathcal{L}_2\,U)$then enforces smoothness of the principal directions v_i in the Laplacian basis of $\mathcal{L}_2$ defined by p_j. Thus, the principal directions v_i will be aligned with the first few eigenvectors of $\mathcal{L}_2$. Used together, these two priors push towards a matrix $U = V\Sigma W^\top$ that is close to low rank.We demonstrate this phenomenon experimentally in Section VII.

III. Graphs Construction

We use two types of graphs G_1 and G_2 in our proposed model. The graph G_1 is constructed between the data samples and the graph G_2 is constructed between the features.

A. General Strategy

The graphs are built using a standard K-nearest neighbor strategy. The first step consists of searching the closest neighbours for all the samples using Euclidean distances. We connect each x_i to its K nearest neighbors x_j, resulting in $|\mathcal{E}|$ number of connections. The second step consists of computing the graph weight matrix A as

$$A_{ij} = \begin{cases} \exp\left(-\frac{\|B_{ij}\circ(x_i - x_j)\|_2^2}{\|B_{ij}\|_1\sigma^2}\right) & \text{if } x_j \text{ is connected to } x_i \\ 0 & \text{otherwise.} \end{cases}$$

where $B_{ij} \in \{0,1\}^p$ is the vector mask corresponding to the intersection of uncorrupted values in x_i and x_j. The use of mask B makes the graph robust to gross corruptions in the dataset. Thus

$$B_{ij}^d = \begin{cases} 1 & \text{if features } x_i^d \ \&\ x_j^d \text{ observed,} \\ 0 & \text{otherwise} \end{cases}$$

In this case, we assume that the mask of missing values is known. The parameter σ can be set experimentally as the average distance of the connected samples. Provided that this parameter is not big, it does not effect the final quality of our algorithm.

Finally, in the third step, the normalized graph Laplacian $\mathcal{L} = I - D^{-1/2}AD^{-1/2}$ is calculated, where D is the degree matrix. This procedure has a complexity of $\mathcal{O}(ne)$ and each A_{ij} can be computed in parallel.

Depending on the values of n and p, the presence of corruptions and the knowledge of the mask, the above computation can be done in two different ways.

Strategy 1: For small n, p we can use the above strategy directly for both G_1 and G_2 even if the dataset is corrupted and we know the mask for corruptions. Although, the computation of A is $\mathcal{O}(n^2)$, it should be noted that with sufficiently small n and p, the graphs G_1 and G_2 can still be computed in the order of a few seconds.

Strategy 2: For big or high dimensional datasets, i.e, large n or large p or both, we can use a similar strategy but the computations can be made efficient using the FLANN library (Fast Library for Approximate Nearest Neighbors searches in high dimensional spaces) [21]. However, currently, the FLANN library does not support the mask operator so the corruptions are included in the graphs construction. Therefore, the quality of the graphs constructed using this strategy is slightly lower as compared to strategy 1 due to 1) the approximate nearest neighbor search method and 2) corruptions (if any) even if the mask information is known. We describe the complexity of FLANN in Section V.

Thus for our work the overall quality of graphs can be divided into 3 types.

- **Type A**: Good sample graph G_1 and good feature graph G_2, both constructed using strategy 1. This case corresponds to small n and p.
- **Type B**: Good sample graph G_1 using strategy 1 and noisy feature graph G_2 using strategy 2. This case corresponds to small n but large p.
- **Type C**: Noisy sample graph G_1 and noisy feature graph G_2 both constructed using strategy 2 for large n and p.

We report the performance of FRPCAG for these three combinations of graph types, thus the acronyms FRPCAG(A), FRPCAG(B) and FRPCAG(C). Although the graph quality is lower if FLANN is used for corrupted data, our experiments for MNIST dataset show that our proposed model attains better results than other state-of-the-art models even with low quality graphs. A summary of the graph construction methodology using the above two strategies is presented in Fig. 8 of the supplementary material Section B.

B. Special case: features' graph G_2 for image datasets

For the non-image datasets we simply construct a graph between the features of X. However, for the case of images it is more reasonable to enforce smoothness on the patch level rather than on the pixel level. Indeed comparing patches on the images allows the use of local information of the image. As a first step we vectorize the patches that correspond to the same position for all the images in the dataset. Let l^2 be the size of each square patch which is centered at the pixel under consideration, then we form p data samples each of size nl^2 as shown in Fig. 2. These transformed data samples are then fed into the graph construction algorithm described in Section III-A.

IV. Optimization Solution

A. Fast Iterative Soft Thresholding

We use the Fast Iterative Soft Thresholding Algorithm (FISTA) [3] to solve problem (1). Let $g : \mathbb{R}^N \to \mathbb{R}$ be a convex, differentiable function with a β-Lipschitz continuous gradient ∇g and $h : \mathbb{R}^{\mathbb{N}} \to \mathbb{R}$ a convex function with a proximity operator $\text{prox}_h : \mathbb{R}^N \to \mathbb{R}^N$ defined as:

$$\text{prox}_{\lambda h}(y) = \underset{x}{\text{argmin}} \frac{1}{2}\|x - y\|_2^2 + \lambda h(x).$$

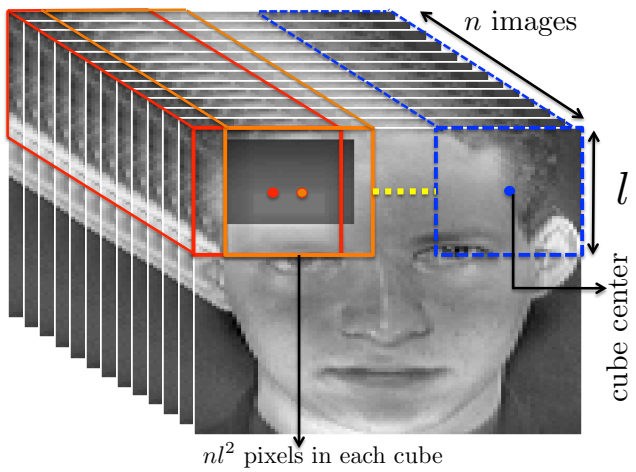


Fig. 2. For the construction of graph G_2, all the images are arranged in a cube and a patch of size l^2, at the same position for all n images and centered at every pixel is vectorized. This results in p vectors with nl^2 pixels each. The FLANN algorithm is then used to compute the approximate $p \times p$ K-nearest neighbors graph from the p vectorized patches.

Our goal is to minimize the sum $g(x) + h(x)$, which is done efficiently with proximal splitting methods. More information about proximal operators and splitting methods for non-smooth convex optimization can be found in [8]. For model (1), $g(U) = \gamma_1 \operatorname{tr}(U\,\mathcal{L}_1\,U^\top) + \gamma_2 \operatorname{tr}(U^\top\,\mathcal{L}_2\,U)$ and $h(U) = \|X - U\|_1$. The gradient of g becomes

$$\nabla_g(U) = 2(\gamma_1 U\,\mathcal{L}_1 + \gamma_2\,\mathcal{L}_2\,U). \tag{4}$$

We define an upper bound on the Lipschitz constant β as $\beta \leq \beta' = 2\gamma_1 \| \mathcal{L}_1 \|_2 + 2\gamma_2 \| \mathcal{L}_2 \|_2$ where $\| \mathcal{L} \|_2$ is the spectral norm (or maximum eigenvalue) of $\mathcal{L}$. Moreover, the proximal operator of the function h is the ℓ_1 soft-thresholding given by the elementwise operations (here $\circ$ is the Hadamard product)

$$\operatorname{prox}_{\lambda h}(U) = X + \operatorname{sgn}(U - X) \circ \max(|U - X| - \lambda, 0). \tag{5}$$

The FISTA algorithm [3] can now be stated as Algorithm 1, where λ is the step size (we use $\lambda = \frac{1}{\beta'}$), ϵ the stopping tolerance

Algorithm 1 FISTA for FRPCAG

INPUT: $Y_1 = X$, $U_0 = X$, $t_1 = 1$, $\epsilon > 0$

for $j = 1, \ldots J$ **do**

$U_{j+1} = \operatorname{prox}_{\lambda_j h}(Y_j - \lambda_j \nabla g(Y_j))$

$t_{j+1} = \frac{1+\sqrt{1+4t_j^2}}{2}$

$Y_{j+1} = U_j + \frac{t_j - 1}{t_{j+1}}(U_j - U_{j-1})$

if $\frac{\|Y_{j+1} - Y_j\|_F^2}{\|Y_j\|_F^2 + \delta} < \epsilon$ **then**

BREAK

end if

end for

and J the maximum number of iterations. δ is a very small number to avoid a possible division by 0.

V. Computational Complexity

A. Complexity of Graph Construction

For n p-dimensional vectors, the computational complexity of the FLANN algorithm is $\mathcal{O}(pnK(\log(n)/\log(K)))$ for the graph G_1 between the samples and $\mathcal{O}(pnK(\log(p)/\log(K)))$ for the graph G_2 between the features, where K is the number of nearest neighbors. For the image databases, the graph G_2 is constructed between the patches of size nl^2 so the computational complexity of the FLANN algorithm with this strategy is $\mathcal{O}(pnl^2K(\log(p)/\log(K)))$. For a fixed K and l^2 the complexity of G_1 is $\mathcal{O}(pn\log(n))$ and that of G_2 is $\mathcal{O}(np\log(p))$. We use $K = 10$ and $l^2 = 25$ for all the experiments reported in this work.

B. Algorithm Complexity

1) FISTA: Let I denote the number of iterations for the algorithm to converge, p is the data dimension, n is the number of samples and c is the rank of the low-dimensional space. The computational cost of our algorithm per iteration is linear in the number of data samples n, i.e. $\mathcal{O}(Ipn)$ for I iterations.

2) Final SVD: Our model, in order to preserve convexity, finds an approximately low-rank solution U without explicitly factorizing it. While this gives a great advantage, depending on the application we have in hand, we might need to provide explicitly the low dimensional representation in a factorized form. This can be done by computing an "economic" SVD of U after our algorithm has finished.

Most importantly, this computation can be done in time that scales linearly with the number of samples for a fixed number of features $p \ll n$. Let $U = V\Sigma W^\top$ the SVD of U. The orthonormal basis V can be computed by the eigenvalue decomposition of the small $p \times p$ matrix $UU^\top = VEV^\top$ that also reveals the singular values $\Sigma = \sqrt{E}$ since $UU^\top$ is s.p.s.d. and therefore E is non-negative diagonal. Here we choose to keep only the c biggest singular values and corresponding vectors according to the application in hand (the procedure for determining c is explained in Section VI). Given V and Σ the sample projections are computed as $W = \Sigma^{-1}V^\top U$.

The complexity of this SVD is $\mathcal{O}(np^2)$ –due to the multiplication $UU^\top$– and does not change the asymptotic complexity of our algorithm. Note that the standard economic SVD implementation in numerical analysis software typically does not use this simple trick, in order to achieve better numerical error. However, in most machine learning applications like the ones of interest in this paper, the compromise in terms of numerical error is negligible compared to the gains in terms of scalability.

C. Overall Complexity

The complexity of FISTA is $\mathcal{O}(Ipn)$, the graph G_1 is $\mathcal{O}(pn\log(n))$, G_2 is $\mathcal{O}(pn\log(p))$ and the final SVD step is $\mathcal{O}(np^2)$. Given that $p \ll n$, the overall complexity of our algorithm is $\mathcal{O}(pn(\log(n) + I + p + \log(p)))$. Table III in the supplementary material Section C presents the computational complexities of all the models considered in this work (discussed in Section VI).

D. Scalability

The construction of graphs G_1 and G_2 is highly scalable. For small n and p the strategy 1 of Section III can be used for the graphs construction and each of the entries of the adjacency matrix A can be computed in parallel once the nearest neighbors have been found. For large n an p the approximate K-nearest neighbors scheme (FLANN) is used for graphs construction which is highly scalable as well. Next, our proposed FISTA algorithm for FRPCAG requires two important computations at every iteration: 1) computation of proximal operator $\text{prox}_{\lambda h}(U)$ and 2) the gradient $\nabla g(Y)$. The former computation is given by the

element-wise soft-thresholding (eq. (5)) that can be performed in parallel for all the entries of a matrix. The gradient computation, as given by eq. (4), involves matrix-matrix multiplications that involve sparse matrices $\mathcal{L}_1$ and $\mathcal{L}_2$ and can be performed very efficiently in parallel as well.

VI. Results

Experiments were done using two open-source toolboxes: the UNLocBoX [27] for the optimization part and the GSPBox [26] for the graph creation. The complete demo, code and datasets used for this work are available at https://lts2.epfl.ch/research/reproducible-research/frpcag/. We perform two types of experiments corresponding to two applications of PCA.

1) Data clustering in the low-dimensional space.
2) Low-rank recovery: Static background separation from videos.

We present extensive quantitative results for clustering but currently our experiments for low-rank recovery are limited to qualitative analysis only. This is because our work on approximating the low-rank representation using graphs is the first of its kind. Theoretical investigation for this approximate method has to be carried out. The experiments on the low-rank background extraction from videos suffice as an initial proof-of-concept for the working of this model.

We perform our clustering experiments on 7 benchmark databases: CMU PIE[1], ORL[2], YALE[3], COIL20[4], MNIST[5], USPS and MFEAT[6]. CMU PIE, ORL and YALE are face databases with small pose variations. COIL20 is a dataset of objects with significant pose changes so we select the images for each object with less than 45 degrees of pose change. USPS and MNIST contain images of handwritten digits and MFeat consists of features extracted from handwritten numerals. The details of all datasets used are provided in Table IV in the supplementary material Section D.

In order to evaluate the robustness of our model to gross corruptions we corrupt the datasets with two different types of errors 1) block occlusions and 2) random missing pixels. Block occlusions of three different sizes, i.e, 15%, 25% and 40% of the total size of the image are placed uniformly randomly in all the images of the datasets. Similarly, all the images of the datasets are also corrupted by removing 10%, 20%, 30% and 40% pixels uniformly randomly. Separate clustering experiments are performed for each of the different types of corruptions.

We compare the clustering performance of our model with 10 other models including the state-of-art: 1) k-means on original data 2) Normalized Cut (NCut) [30] 3) Laplacian Eigenmaps (LE) [4] 4) Standard PCA 5) Graph Laplacian PCA (GLPCA) [13] 6) Manifold Regularized Matrix Factorization (MMF) [37] 7) Non-negative Matrix Factorization (NMF) [17] 8) Graph Regularized Non-negative Matrix Factorization (GNMF) [6] 9) Robust PCA (RPCA) [7] and 10) Robust PCA on Graphs (RPCAG) [29]. For ORL, CMU PIE, COIL20, YALE and USPS datasets we compare three different versions of our model corresponding to the three types of graphs G_1 and G_2.

As mentioned in Section III, FRPCAG(A) corresponds to our model using good quality sample and feature graphs, FRPCAG(B) to the case using a good quality sample graph and approximate feature graph, and FRPCAG(C) to the case where approximate graphs were used both between samples and features. All other models for these 5 datasets are evaluated using a good quality sample graph G_1. Due to the large size of the MNIST dataset, we use FLANN (strategy 2) to construct both graphs, therefore

[1] http://vasc.ri.cmu.edu/idb/html/face/
[2] http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html
[3] http://vision.ucsd.edu/content/yale-face-database
[4] http://www.cs.columbia.edu/CAVE/software/softlib/coil-20.php
[5] http://yann.lecun.com/exdb/mnist/
[6] https://archive.ics.uci.edu/ml/datasets/Multiple+Features

we get approximate versions in the presence of corruptions. Thus the experiments on MNIST dataset are kept separate from the rest of the datasets to emphasize the difference in the graph construction strategy. We also perform a separate set of experiments on the ORL dataset and compare the performance of our model with state-of-the-art nuclear norm based models, RPCA [7] and RPCAG [29], both with a good quality and an approximate graph. We perform this set of experiments only on the ORL dataset (due to its small size) as the nuclear norm based models are computationally expensive. Finally, the experiments on MFeat dataset are only performed with missing values because block occlusions in non-image datasets correspond to an unrealistic assumption. The computational complexities of all these models are presented in Table III in the supplementary material Section C.

Pre-processing: All datasets are transformed to zero-mean and unit standard deviation along the features for the RPCA, RPCAG and FRPCAG. For MMF the samples are additionally normalized to unit-norm. For NMF and GNMF only the unit-norm normalization is applied to all the samples of the dataset.

Evaluation: We use *clustering error* as a metric to compare the clustering performance of various models. NCut, LE, PCA, GLPCA, MMF, NMF and GNMF are matrix factorization models that explicitly learn the principal components W. The clustering error for these models is evaluated by performing k-means on the principal components. RPCA, RPCAG and FRPCAG learn the low-rank matrix U. The clustering error for these models is evaluated by performing k-means on the principal components W obtained by the SVD of the low-rank matrix $U = V\Sigma W^\top$. Moreover, RPCA and RPCAG determine the exact low-rank representation U, whereas our model only shrinks singular values and therefore only recovers an approximate low-rank representation U. Thus, the dimension of the subspace (number of columns of W) for FRPCAG is decided by selecting the number of singular values such that the lowest selected singular value is 10% of the maximum singular value. Due to the non-deterministic nature of k-means, it is run 10 times and the minimum error over all runs is reported.

Parameter selection for various models: Each model has several parameters which have to be selected in the validation stage of the experiment. To perform a fair validation for each of the models we use a range of parameter values as presented in Table V in the supplementary material Section E. For a given dataset, each of the models is run for each of the parameter tuples in this table and the parameters corresponding to minimum clustering error are selected for testing purpose. Furthermore, PCA, GLPCA, MMF, NMF and GNMF are non-convex models so they are run 10 times for each of the parameter tuple. RPCA, RPCAG and FRPCAG are convex so they are run only once.

Parameter selection for Graphs: For all the experiments reported in this paper we use the following parameters for graphs G_1 and G_2. K-nearest neighbors = 10, $\sigma^2 = 1$ and patch size $l^2 = 25$. It is important to point out here that different types of data might call for slightly different parameters for graphs. However, for a given dataset, the use of same graph parameters (same graph quality) for all the graph regularized models ensures a fair comparison.

A. Clustering

1) Comparison with Matrix Factorization Models: Fig. 3 presents the clustering error for various matrix factorization and our proposed model. NMF and GNMF are not evaluated for the USPS and MFeat datasets as they are not originally non-negative. It can be seen that our proposed model FRPCAG(A) with the two good quality graphs performs better than all the other models in most of the cases both in the presence and absence of data corruptions. Even FRPCAG(B) with a good sample graph G_1 and a noisy feature graph G_2 performs reasonably well. This shows that our model is quite robust to the quality of graph G_2. However, as expected FRPCAG(C) performs worse for ORL, CMU PIE, COIL20, YALE and USPS datasets as compared to other models evaluated with a good sample graph G_1. Finally, our model outperforms others in most of the cases for the interesting case of MNIST dataset where both graphs G_1 and G_2 are noisy for all models under consideration. It is worth mentioning here that

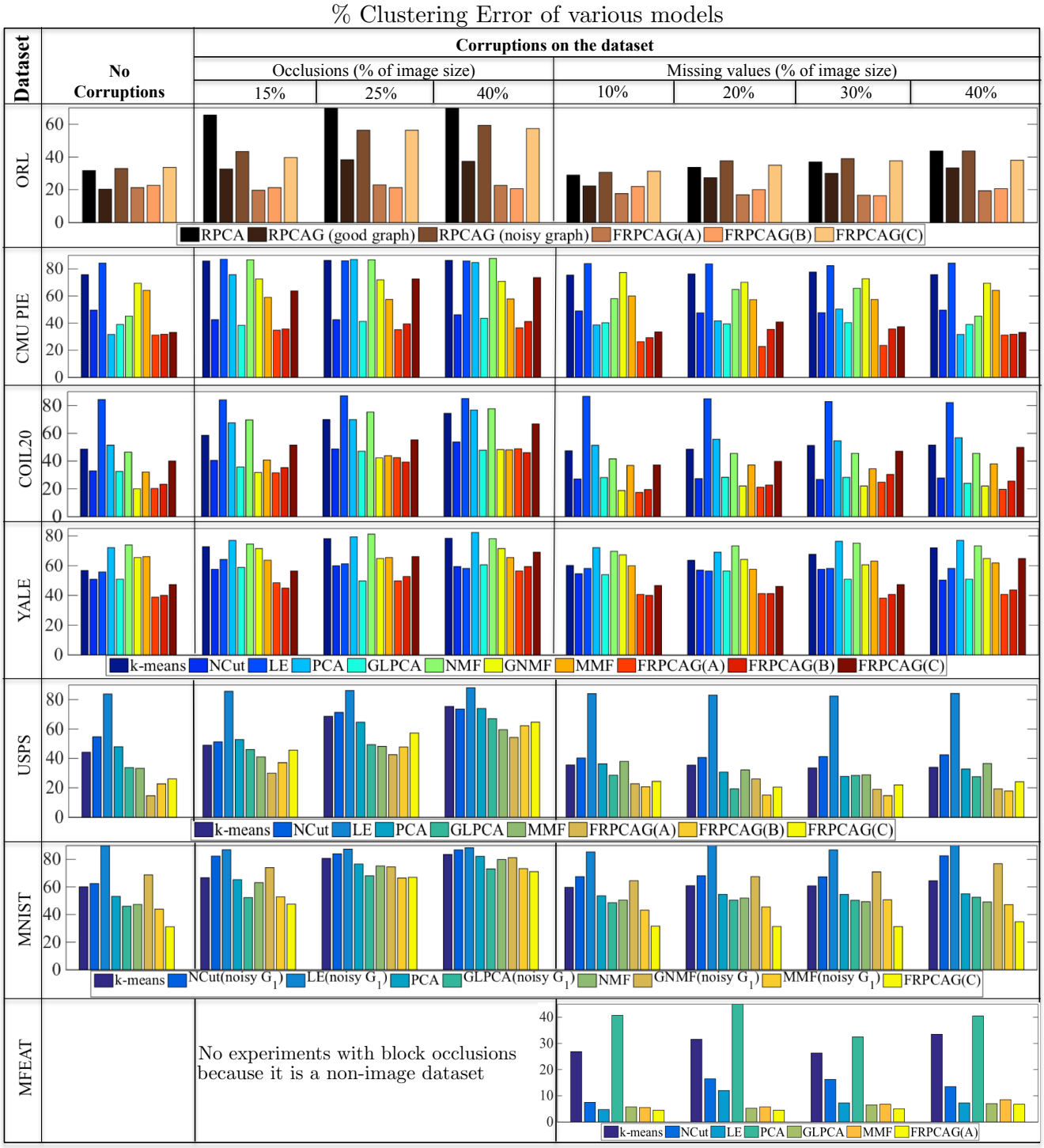


Fig. 3. A comparison of clustering error of our model with various dimensionality reduction models. The image data sets include: 1) ORL 2) CMU PIE 3) COIL20 and 4) YALE. The compared models are: 1) k-means 2) Normalized Cut (NCut) 3) Laplacian Eigenmaps (LE) [4] 4) Standard Principal Component Analysis (PCA) 5) Graph Laplacian PCA (GLPCA) [13] 6) Non-negative Matrix Factorization [17] 7) Graph Regularized Non-negative Matrix Factorization (GNMF) [6] 8) Manifold Regularized Matrix Factorization (MMF) [37] 9) Robust PCA (RPCA) [7] 10) Fast Robust PCA on Graphs (A) 11) Fast Robust PCA on Graphs (B) and 12) Fast Robust PCA on Graphs (C). Two types of corruptions are introduced in the data: 1) Block occlusions and 2) Random missing values. NCut, LE, GLPCA, MMF and GNMF are evaluated with a good sample graph G_1. FRPCA(A) corresponds to our model evaluated with a good sample and a good feature graph, FRPCA(B) to a good sample graph and a noisy feature graph and FRPCA(C) to a noisy sample and feature graph. NMF and GNMF require non-negative data so they were not evaluated for the USPS and MFeat datasets because they are negative as well. MFeat is a non-image dataset so it is not evaluated with block occlusions. Due to the large size of the MNIST dataset, we use FLANN algorithm (strategy 2) to construct the graphs, therefore we get noisy graphs in the presence of corruptions.

even though the absolute errors are quite high for FRPCAG on the MNIST dataset, it performs relatively better than the other models. As PCA is mostly used as a feature extraction or a pre-processing step for a variety of machine learning algorithms, a better absolute classification performance can be obtained for these datasets by using FRPCAG as a pre-processing step for supervised algorithms as compared to other PCA models.

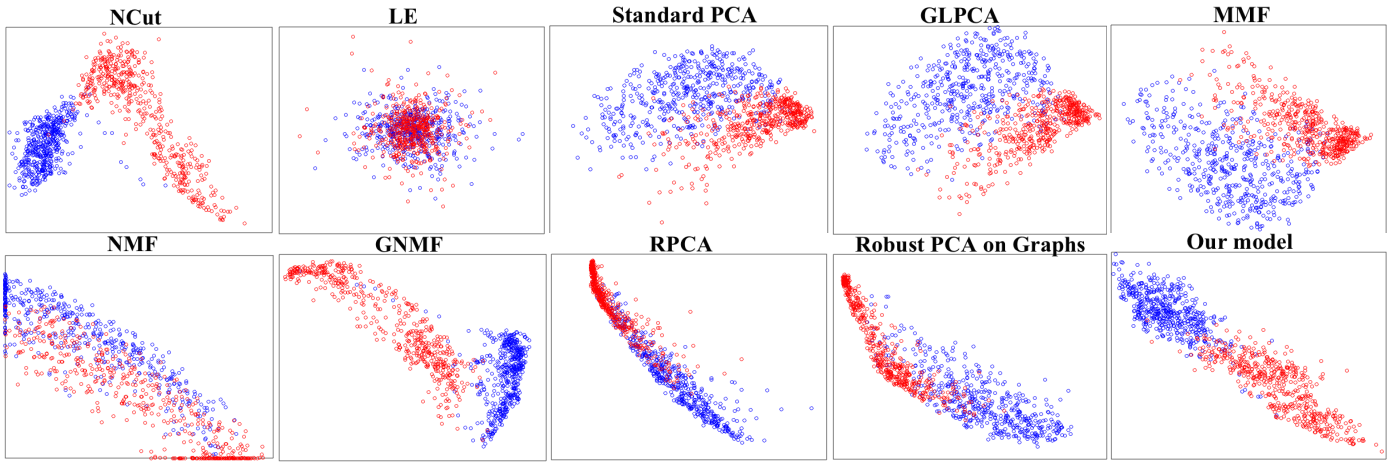


Fig. 4. Principal Components of 1000 samples of digits 0 and 1 of the MNIST dataset in 2D space. For this experiment all the digits were corrupted randomly with 15% missing pixels. Our proposed model (lower right) attains a good separation between the digits which is comparable and even better than other state-of-the-art dimensionality reduction models.

2) Comparison with Nuclear Norm based Models: Fig. 3 also presents a comparison of the clustering error of our model with nuclear norm based models, i.e, RPCA and RPCAG for ORL dataset. This comparison is of specific interest because of the convexity of all the algorithms under consideration. As these models require an expensive SVD step on the whole low-rank matrix at every iteration of the algorithm, these experiments are performed on small ORL dataset. Clearly, our proposed model FRPCAG(A) performs better than the nuclear norm based models even in the presence of large fraction of gross errors. Interestingly, even FRPCAG(B) with a noisy graph G_2 performs better than RPCAG with a good graph G_1. Furthermore, the performance of FRPCAG(C) with two noisy graphs is comparable to RPCAG with noisy graph, but still better than RPCA.

B. Principal Components

Fig. 4 shows the principal components of 1000 samples of MNIST dataset in two dimensional space obtained by various dimensionality reduction models. 500 samples of digit 0 and 1 each are chosen and randomly corrupted by 15% missing pixels for this experiment. Clearly, our proposed model attains a good separation between the digits 0 and 1 (represented by blue and red points respectively) comparable with other state-of-the-art dimensionality reduction models.

C. Static background separation from videos

In order to demonstrate the effectiveness of our model to recover low-rank static background from videos we perform experiments on 1000 frames of 3 videos available online [7]. The graph G_1 is constructed between the 1000 frames of the

[7] https://sites.google.com/site/backgroundsubtraction/test-sequences

Fig. 5. Static background separation from three videos. Each row shows the actual frame (left), recovered static low-rank background using RPCA, RPCAG and our proposed model. The first row corresponds to the video of a restaurant food counter, the second row to the shopping mall lobby and the third to an airport lobby. In all the three videos the moving people belong to the sparse component. Thus, our model is able to accurately separate the static portion from the three frames as good as the RPCAG. Our model converged in less than 2 minutes for each of the three videos, whereas RPCA and RPCAG converged in more than 45 minutes.

video and the graph G_2 is constructed between the patches of frames following the methodology of Section III. Both graphs for all the videos are constructed without the prior knowledge of the mask of sparse errors (moving people). Fig. 5 shows the recovery of low-rank frames for one actual frame of each of the videos. The leftmost plot in each row shows the actual frame, the other three show the recovered low-rank representations using RPCA, RPCAG and our proposed model (FRPCAG). The first row corresponds to a frame from the video of a restaurant food counter, the second row to the shopping mall lobby and the third row to an airport lobby. In each of the three plots it can be seen that our proposed model is able to separate the static backgrounds very accurately from the moving people which do not belong to the static ground truth. Our model converged in less than 2 minutes for each of the three videos, whereas RPCA and RPCAG converged in more than 45 minutes. We make the complete videos available online[8] [9] [10]. Due to the unavailability of low-rank ground truth for these videos we do not present

[8] https://vid.me/ixS3

[8] https://vid.me/UtDZ

[10] https://vid.me/Ad1w

any quantitative results. However, in future we will work on foreground extraction and present quantitative results as in [5].

D. Computational Time

TABLE I. COMPUTATION TIMES (IN SECONDS) FOR GRAPHS G_1, G_2, FRPCAG, RPCAG, RPCA AND THE NUMBER OF ITERATIONS TO CONVERGE FOR DIFFERENT DATASETS. THE COMPUTATION IS DONE ON A SINGLE CORE MACHINE WITH A 3.3 GHZ PROCESSOR WITHOUT USING ANY DISTRIBUTED OR PARALLEL COMPUTING TRICKS. ∞ INDICATES THAT THE ALGORITHM DID NOT CONVERGE IN 4 HOURS.

Dataset	**Samples**	**Features**	**Classes**	**Graphs**		**FRPCAG**		**RPCAG**		**RPCA**	
				G_1	G_2	**time**	**Iters**	**time**	**Iters**	**time**	**Iters**
MNIST	5000	784	10	10.8	4.3	13.7	27	1345	325	1090	378
MNIST	15000	784	10	32.5	13.3	35.4	23	3801	412	3400	323
MNIST	25000	784	10	40.7	22.2	58.6	24	∞	∞	∞	∞
ORL	300	10304	30	1.8	56.4	4.7	12	360	301	240	320
USPS	3500	256	10	5.8	10.8	1.76	16	900	410	790	350
US census	2.5 million	68	-	540	42.3	3900	200	∞	∞	∞	∞

Table I presents the computational time and number of iterations for the convergence of FRPCAG, RPCAG and RPCA on different sizes and dimensions of the datasets. We also present the time needed for the graph construction. The computation is done on a single core machine with a 3.3 GHz processor without using any distributed or parallel computing tricks. An ∞ in the table indicates that the algorithm did not converge in 4 hours. It is notable that our model requires a very small number of iterations to converge irrespective of the size of the dataset. Furthermore, the model is orders of magnitude faster than RPCA and RPCAG. This is clearly observed from the experiments on MNIST dataset where our proposed model is 100 times faster than RPCAG. Specially for MNIST dataset with 25000 samples, RPCAG and RPCA did not converge even in 4 hours whereas FRPCAG converged in less than a minute.

To demonstrate the scalability of our model for big datasets, we perform an experiment on the US census 1990 [11] dataset available at the UCI machine learning repository. This dataset consists of approximately 2.5 million samples and 68 features. The approximate K-nearest neighbors graph construction strategy using the FLANN algorithm took only 540 secs to construct G_1 between 2.5 million samples and 42.3 secs. to construct G_2 between 68 features. We do not compare the performance of this model with other state-of-the-art models as the ground truth for this dataset is not available. However, we run our algorithm in order to see how long it takes to recover a low-rank representation for this dataset. It took 65 minutes and 200 iterations for the algorithm to converge on a single core machine with 3.3 GHz of CPU power.

VII. APPROXIMATE LOW-RANK RECOVERY

As already discussed, even though our proposed model targets approximate low-rank recovery, it still attains state-of-the-art results for data clustering in low-dimensional space and background separation from videos. We take a step back here and present an experimental justification for the success of this model. In summary we illustrate that:

1) The model recovers a close-to-low-rank representation.

[11] https://archive.ics.uci.edu/ml/datasets/US+Census+Data+(1990)

2) The principal components and principal directions of U align with the first few eigenvectors of their respective graphs, automatically revealing a low-rank and enhanced class structure.
3) The singular values of the low-rank matrix obtained using our model closely approximate those obtained by nuclear norm based models even in the presence of corruptions.

Our justification relies mostly on the quality of the singular values of the low-rank representation and the alignment of the singular vectors with their respective graphs. A detailed quantitative evaluation of the low-rank approximation along with the theoretical investigation is left for future work.

We perform an experiment with 1000 samples of the MNIST dataset belonging to two different classes (digits 0 and 1). As argued in [18], if the data is arranged according to the classes, the matrix $WW^\top$ (where W are the principal components of the data) reveals the subspace structure. If the subspaces are orthogonal then it should acquire a block diagonal structure. Furthermore, as explained in Section II our model tends to align the first few principal components w_i and principal directions v_i of U to the first few eigenvectors q_j and p_j of $\mathcal{L}_1$ and $\mathcal{L}_2$ respectively. Thus, it is interesting to observe the matrices $\Sigma W^\top Q$ and $\Sigma V^\top P$ scaled with the singular values Σ of the low-rank matrix U, as justified by eq. (3). This scaling takes into account the importance of the eigenvectors that are associated to bigger singular values.

Fig. 6 plots the matrix $WW^\top$, the corresponding clustering error, the matrices Σ, $\Sigma W^\top Q$, and $\Sigma V^\top P$ for different values of γ_1 and γ_2 from left to right. Increasing γ_1 and γ_2 from 1 to 30 leads to 1) the thresholding of the singular values in Σ resulting in a lower rank 2) alignment of the first few principal components w_i and principal directions v_i in the direction of the first few eigenvectors q_j and p_j of $\mathcal{L}_1$ and $\mathcal{L}_2$ respectively 3) an enhanced subspace structure in $WW^\top$ and 4) a lower clustering error. Together the two graphs help in acquiring a low-rank structure that is suitable for clustering applications as well. As already described in Section VI the final rank of U is decided by using the threshold $0.1\sigma_{max}$.

Next we demonstrate that for data with or without corruptions, FRPCAG is able to acquire singular values as good as the nuclear norm based models, RPCA and RPCAG. We perform three clustering experiments on 30 classes of ORL dataset with no block occlusions, 15% block occlusions and 25% block occlusions. Fig. 7 presents a comparison of the singular values of the original data with the singular values of the low-rank matrix obtained by solving RPCA, RPCAG and our model. The parameters for all the models are selected corresponding to the lowest clustering error for each model. It is straightforward to conclude that the singular values of the low-rank representation using our fast method closely approximate those of the nuclear norm based models irrespective of the level of corruptions.

Finally, in Section F of the supplementary material we show the validation error of our model for ORL dataset with different levels of corruptions. We also demonstrate that the choice of parameters is robust to the level of corruptions in the dataset.

VIII. Applications & Future Work

The idea of using a graph of samples and another graph of features to extract an approximate low-rank representation is quite novel. As demonstrated experimentally in this work, it leads to an improvement in the clustering quality of the data in the low-dimensional space. This methodology is quite general and can easily be extended for other clustering methods. Typical examples include NMF, where the two graph regularization, one on each of the positive factors may lead to an enhanced positive low-rank representation. Another important application for this work is the fast decomposition of a data matrix into a low-rank and sparse matrix. Many applications in signal and image processing, such as matrix completion, background separation from the videos, foreground separation for tracking, analyzing brain activities using dynamic FMRI and EEG signals can benefit from this methodology.

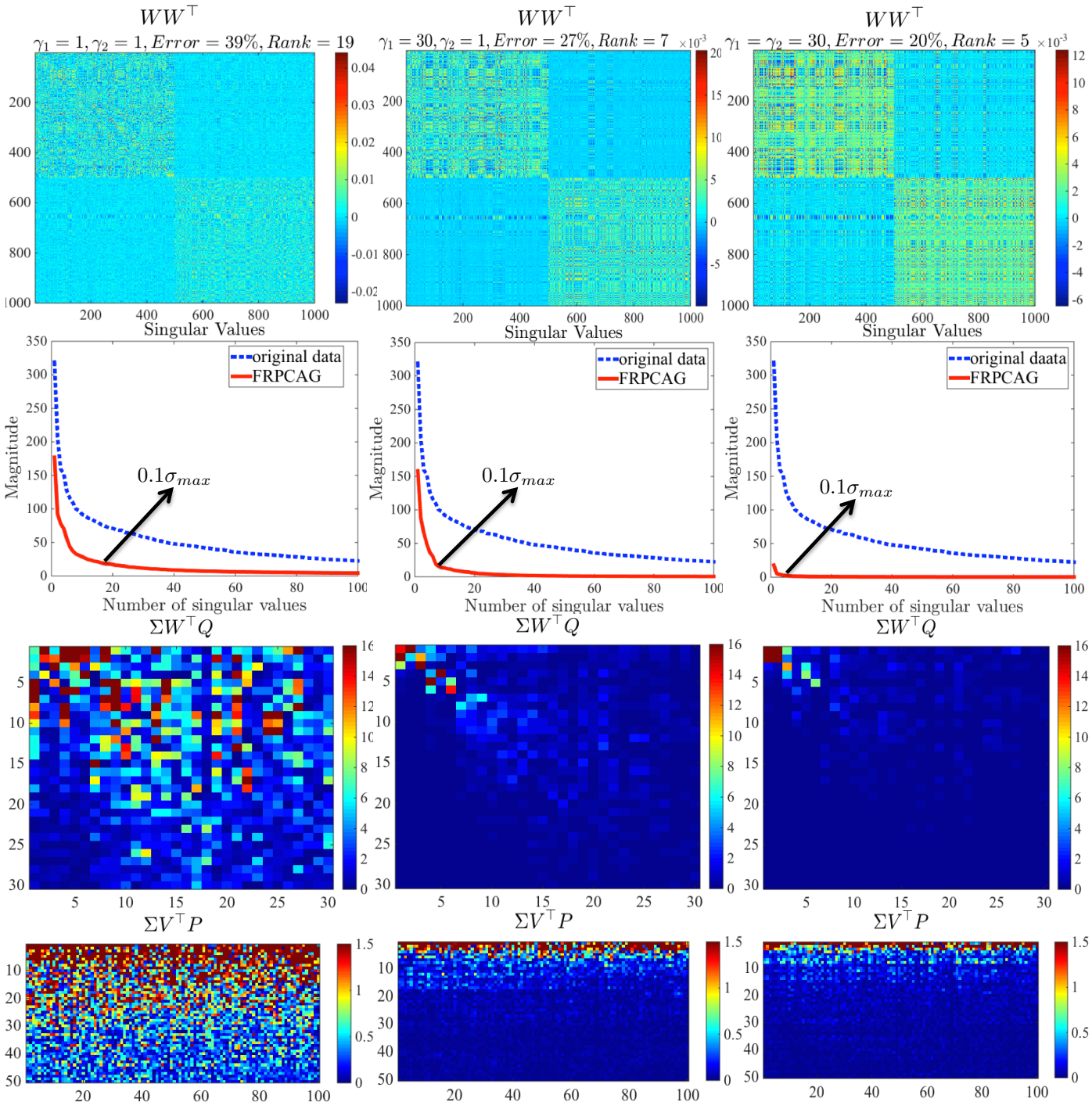


Fig. 6. The matrices $WW^\top$, Σ, $\Sigma W^\top Q$, $\Sigma V^\top P$ and the corresponding clustering errors obtained for different values of the weights on the two graph regularization terms for 1000 samples of MNIST dataset (digits 0 and 1). If $U = V\Sigma W^\top$ is the SVD of U, then W corresponds to the matrix of principal components (right singular vectors of U) and V to the principal directions (left singular vectors of U). Let $\mathcal{L}_1 = Q\Lambda Q^\top$ and $\mathcal{L}_2 = P\Omega P^\top$ be the eigenvalue decompositions of $\mathcal{L}_1$ and $\mathcal{L}_2$ respectively then Q and P correspond to the eigenvectors of Laplacians $\mathcal{L}_1$ and $\mathcal{L}_2$. The block diagonal structure of $WW^\top$ becomes more clear by increasing γ_1 and γ_2 with a thresholding of the singular values in Σ. Further, the sparse structure of $\Sigma W^\top Q$ and $\Sigma V^\top P$ towards the rightmost corners show that the number of left and right singular vectors which align with the eigenvectors of the Laplacians $\mathcal{L}_1$ and $\mathcal{L}_2$ go on decreasing with increasing γ_1 and γ_2. This shows that the two graphs help in attaining a low-rank structure with a low clustering error.

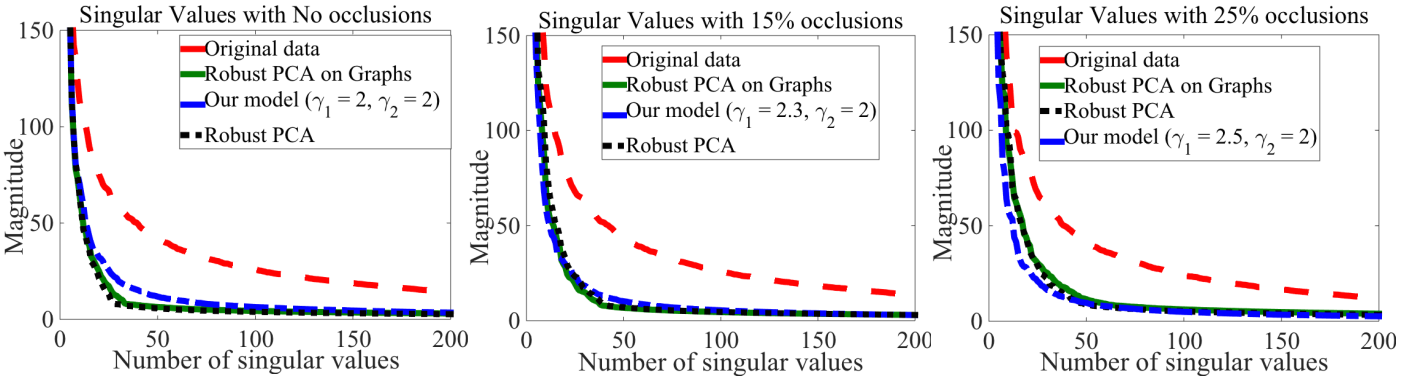


Fig. 7. A comparison of singular values of the low-rank matrix obtained via our model, RPCA and RPCAG. The experiments were performed on the ORL dataset with different levels of block occlusions. The parameters corresponding to the minimum validation clustering error for each of the model were used.

In this paper we present the quantitative and qualitative evaluation of our low-rank approximation method for clustering task. One essential question remains unanswered: What is the quality of low-rank representation using this method? In fact this constitutes an important application of PCA, such as background extraction from videos. Our future work will be dedicated to a thorough theoretical investigation of the approximate low-rank model that we have presented in this paper. Another interesting direction would be to explore the effect of various graph parameters on the properties of the low-rank approximation. We would also like to adapt this model for online applications, such as online Robust PCA and dynamic data such as EEG and FMRI brain signals.

IX. Conclusion

We present Fast Robust PCA on Graphs (FRPCAG), a fast dimensionality reduction algorithm for mining clusters from high dimensional and large datasets. The power of the model lies in its ability to effectively exploit the hidden information about the intrinsic dimensionality of the smooth low-dimensional manifolds on which reside the signals and features of the data. Therefore, it targets an approximate recovery of low-rank signals by exploiting the local smoothness assumption of the samples and features of the data via graph structures. In short our method leverages 1) smoothness of the samples on a sample graph and 2) smoothness of the features on a feature graph. The proposed method is convex, scalable and efficient and tends to outperform several other state-of-the-art exact low-rank recovery methods in clustering tasks that use the expensive nuclear norm. In an ordinary clustering task FRPCAG is approximately 100 times faster than nuclear norm based methods. The double graph structure also plays an important role towards the robustness of the model to gross corruptions. Furthermore, the singular values of the low-rank matrix obtained via FRPCAG closely approximate those obtained via nuclear norm based methods.

Acknowledgements

We would like to thank Benjamin Ricaud for his valuable insight and suggestions for improving the experimental sections of this paper.

References

[1] H. Abdi and L. J. Williams. Principal component analysis. Wiley Interdisciplinary Reviews: Computational Statistics, 2(4):433–459, 2010. 2

[2] M. Ayazoglu, M. Sznaier, O. Camps, et al. Fast algorithms for structured robust principal component analysis. In Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, pages 1704–1711. IEEE, 2012. 3

[3] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM Journal on Imaging Sciences, 2(1):183–202, 2009. 6, 7

[4] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. Neural computation, 15(6):1373–1396, 2003. 2, 5, 9, 11, 20, 22, 23

[5] T. Bouwmans and E. H. Zahzah. Robust pca via principal component pursuit: a review for a comparative evaluation in video surveillance. Computer Vision and Image Understanding, 122:22–34, 2014. 14

[6] D. Cai, X. He, J. Han, and T. S. Huang. Graph regularized nonnegative matrix factorization for data representation. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 33(8):1548–1560, 2011. 3, 9, 11, 20, 22, 23

[7] E. J. Candès, X. Li, Y. Ma, and J. Wright. Robust principal component analysis? Journal of the ACM (JACM), 58(3):11, 2011. 3, 9, 10, 11, 20, 22, 23

[8] P. L. Combettes and J.-C. Pesquet. Proximal splitting methods in signal processing. In Fixed-point algorithms for inverse problems in science and engineering, pages 185–212. Springer, 2011. 7

[9] H. Du, X. Zhang, Q. Hu, and Y. Hou. Sparse representation-based robust face recognition by graph regularized low-rank sparse representation recovery. Neurocomputing, 164:220–229, 2015. 3

[10] E. Elhamifar and R. Vidal. Sparse subspace clustering: Algorithm, theory, and applications. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 35(11):2765–2781, 2013. 2

[11] S. Gao, I.-H. Tsang, and L.-T. Chia. Laplacian sparse coding, hypergraph laplacian sparse coding, and applications. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 35(1):92–104, 2013. 3

[12] R. Jenatton, G. Obozinski, and F. Bach. Structured sparse principal component analysis. arXiv preprint arXiv:0909.1440, 2009. 2

[13] B. Jiang, C. Ding, and J. Tang. Graph-laplacian pca: Closed-form solution and robustness. In Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on, pages 3492–3498. IEEE, 2013. 3, 9, 11, 20, 22, 23

[14] T. Jin, J. Yu, J. You, K. Zeng, C. Li, and Z. Yu. Low-rank matrix factorization with multiple hypergraph regularizers. Pattern Recognition, 2014. 3

[15] T. Jin, Z. Yu, L. Li, and C. Li. Multiple graph regularized sparse coding and multiple hypergraph regularized sparse coding for image representation. Neurocomputing, 2014. 3

[16] V. Kalofolias, X. Bresson, M. Bronstein, and P. Vandergheynst. Matrix completion on graphs. arXiv preprint arXiv:1408.1717, 2014. 4

[17] D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. Nature, 401(6755):788–791, 1999. 2, 9, 11, 20, 22, 23

[18] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma. Robust recovery of subspace structures by low-rank representation. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 35(1):171–184, 2013. 2, 15

[19] A. Lucas, M. Stalzer, and J. Feo. Parallel implementation of fast randomized algorithms for low rank matrix decomposition. Parallel Processing Letters, 24(01), 2014. 3

[20] J. Mairal, F. Bach, and J. Ponce. Sparse modeling for image and vision processing. arXiv preprint arXiv:1411.3230, 2014. 2

[21] M. Muja and D. Lowe. Scalable nearest neighbour algorithms for high dimensional data. 2014. 6

[22] A. Y. Ng, M. I. Jordan, Y. Weiss, et al. On spectral clustering: Analysis and an algorithm. Advances in neural information processing systems, 2:849–856, 2002. 5

[23] T.-H. Oh, Y. Matsushita, Y.-W. Tai, and I. S. Kweon. Fast randomized singular value thresholding for nuclear norm minimization. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 4484–4493, 2015. 3

[24] X. Peng, C. Lu, Z. Yi, and H. Tang. Connections between nuclear norm and frobenius norm based representation. arXiv preprint arXiv:1502.07423, 2015. 3

[25] Y. Peng, B.-L. Lu, and S. Wang. Enhanced low-rank representation via sparse manifold adaption for semi-supervised learning. Neural Networks, 2015. 3

[26] N. Perraudin, J. Paratte, D. Shuman, V. Kalofolias, P. Vandergheynst, and D. K. Hammond. GSPBOX: A toolbox for signal processing on graphs. ArXiv e-prints, Aug. 2014. 9

[27] N. Perraudin, D. Shuman, G. Puy, and P. Vandergheynst. UNLocBoX A matlab convex optimization toolbox using proximal splitting methods. ArXiv e-prints, Feb. 2014. 9

[28] B. Schölkopf, A. Smola, and K.-R. Müller. Kernel principal component analysis. In Artificial Neural Networks—ICANN'97, pages 583–588. Springer, 1997. 2

[29] N. Shahid, V. Kalofolias, X. Bresson, M. Bronstein, and P. Vandergheynst. Robust principal component analysis on graphs. arXiv preprint arXiv:1504.06151, 2015. 2, 3, 4, 9, 10, 20, 22, 23

[30] J. Shi and J. Malik. Normalized cuts and image segmentation. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 22(8):888–905, 2000.

9, 20, 22, 23

[31] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. Signal Processing Magazine, IEEE, 30(3):83–98, 2013. 2

[32] L. Tao, H. H. Ip, Y. Wang, and X. Shu. Low rank approximation with sparse integration of multiple manifolds for data representation. Applied Intelligence, pages 1–17, 2014. 3

[33] R. Vidal and P. Favaro. Low rank subspace clustering (lrsc). Pattern Recognition Letters, 43:47–61, 2014. 2

[34] Y.-X. Wang and Y.-J. Zhang. Nonnegative matrix factorization: A comprehensive review. Knowledge and Data Engineering, IEEE Transactions on, 25(6):1336–1353, 2013. 2

[35] R. Witten and E. Candes. Randomized algorithms for low-rank matrix factorizations: sharp performance bounds. Algorithmica, pages 1–18, 2013. 3

[36] H. Zhang, Z. Yi, and X. Peng. flrr: fast low-rank representation using frobenius-norm. Electronics Letters, 50(13):936–938, 2014. 3

[37] Z. Zhang and K. Zhao. Low-rank matrix approximation with manifold regularization. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 35(7):1717–1729, 2013. 3, 9, 11, 20, 22, 23

APPENDIX

A. Notation and Terminology

TABLE II. A SUMMARY OF NOTATIONS USED IN THIS WORK

Notation	Terminology
$\|\cdot\|_F$	matrix frobenius norm
$\|\cdot\|_1$	matrix ℓ_1 norm
n	number of data samples
p	number of features / pixels
c	dimension of the subspace
k	number of classes in the data set
$X \in \mathbb{R}^{p\times n}$	data matrix
$U \in \mathbb{R}^{p\times n}$	low-rank noiseless approximation of X
$U = V\Sigma W^\top$	SVD of the low-rank matrix U
$V \in \mathbb{R}^{p\times c}$	left singular vectors of U / principal directions of U
Σ	singular values of U
$W \in \mathbb{R}^{n\times c}$	right singular vectors of U / principal components of U
$A \in \mathbb{R}^{n\times n}$ or $\mathbb{R}^{p\times p}$	adjacency matrix between samples / features of X
$D = diag(\sum_j A_{ij})\forall i$	diagonal degree matrix
$B \in \{0,1\}^p$	mask for data corruptions
σ	smoothing parameter / window width of the Gaussian kernel
G_1	graph between the samples of X
G_2	graph between the features of X
$(\mathcal{V}, \mathcal{E})$	set of vertices, edges for graph
γ_1	penalty for G_1 Tikhonov regularization term
γ_2	penalty for G_2 Tikhonov regularization term
K	number of nearest neighbors for the construction of graphs
$\mathcal{L}_1 \in \mathbb{R}^{n\times n}$	Laplacian for graph G_1
$\mathcal{L}_2 \in \mathbb{R}^{p\times p}$	Laplacian for graph G_2
$\mathcal{L}_1 = Q\Lambda Q^\top$	eigenvalue decomposition of $\mathcal{L}_1$
$\mathcal{L}_2 = P\Omega P^\top$	eigenvalue decomposition of $\mathcal{L}_2$
NCut [30]	Normalized Cut
LE [4]	Laplacian Eigenmaps
PCA	Principal Component Analysis
GLPCA [13]	Graph Laplacian PCA
NMF [17]	Non-negative matrix factorization
GNMF [6]	Graph regularized non-negative matrix factorization
MMF [37]	Manifold matrix factorization
RPCA [7]	Robust PCA
RPCAG [29]	Robust PCA on graphs
FRPCAG	Fast robust PCA on graphs

B. Summary of Graph Construction Methodology

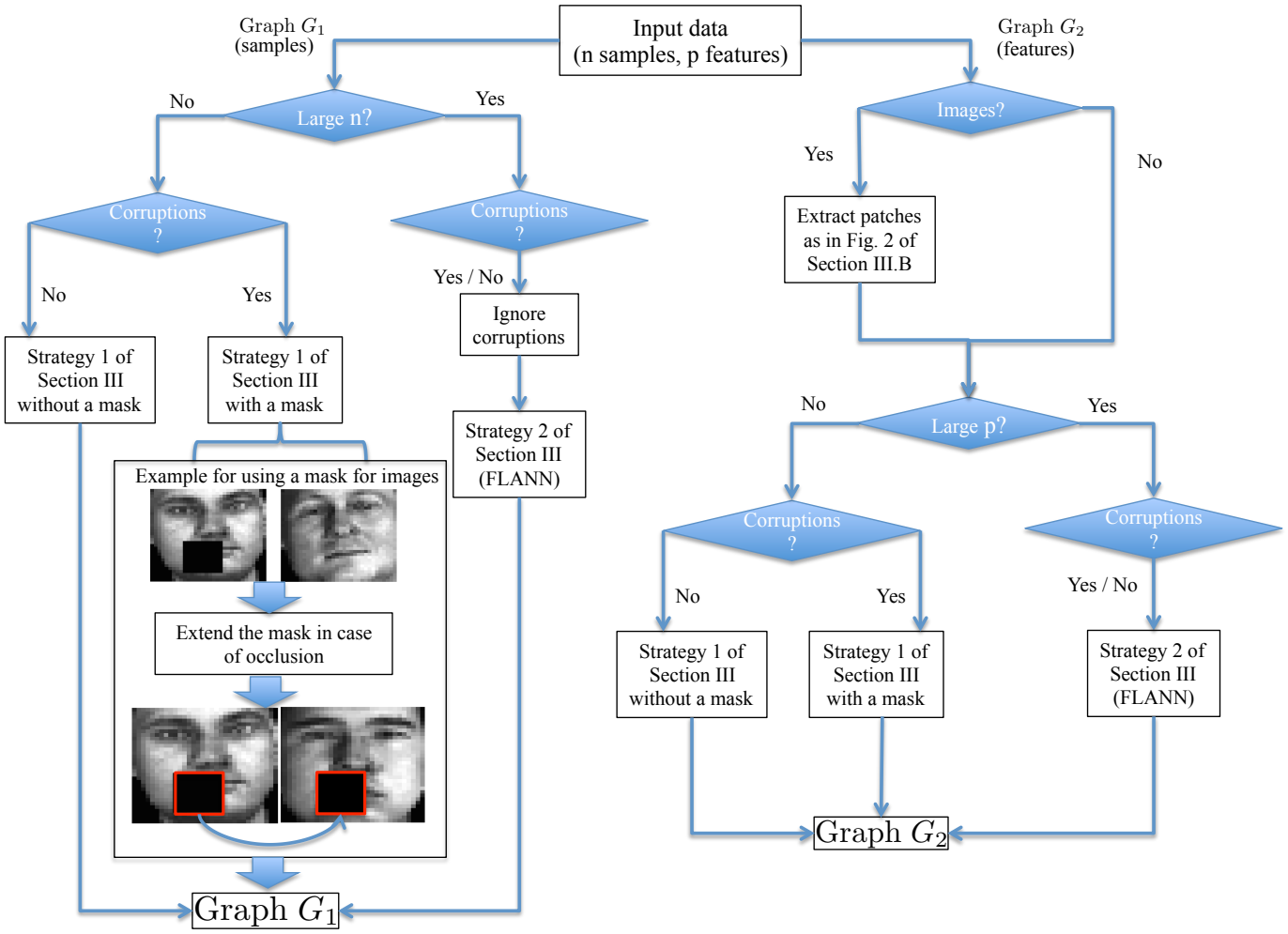


Fig. 8. Methodology for the construction of graphs G_1 (between the samples) and G_2 (between the features).

C. Computational Complexity

TABLE III. COMPUTATIONAL COMPLEXITY OF ALL THE MODELS CONSIDERED IN THIS WORK. I DENOTES THE NUMBER OF ITERATIONS FOR THE ALGORITHM TO CONVERGE, p IS THE DIMENSION, n IS THE NUMBER OF SAMPLES AND c IS THE RANK OF THE LOW-DIMENSIONAL SPACE. ALL THE MODELS WHICH USE THE GRAPH G_1 ARE MARKED BY '+'. THE CONSTRUCTION OF GRAPH G_2 IS INCLUDED ONLY IN OUR MODEL (FRPCAG). NOTE THAT WE HAVE USED COMPLEXITY $\mathcal{O}(np^2)$ FOR ALL SVD COMPUTATIONS AND $\mathcal{O}(In)$ FOR APPROXIMATE EIGENVALUE DECOMPOSITION FOR NCUT AS PROPOSED IN [30], WHILE THE LATTER COULD BE USED FOR THE DECOMPOSITION NEEDED BY LE EVEN THOUGH NOT SPECIFIED IN [4].

Model	**Complexity G_1** $\mathcal{O}(np\log(n))$	**Complexity G_2** $\mathcal{O}(np\log(p))$	**Complexity Algorithm** for $p \ll n$	**Overall Complexity** for $p \ll n$
FRPCAG	+	+	$\mathcal{O}(np(I+p))$	$\mathcal{O}(np(\log(n)+p+I+\log(p)))$
NCut [30]	+	–	$\mathcal{O}(In)$	$\mathcal{O}(n(p\log(n)+I))$
LE [4]	+	–	$\mathcal{O}(n^3)$	$\mathcal{O}(n(p\log(n)+n^2))$
PCA	–	–	$\mathcal{O}(p^2n)$	$\mathcal{O}(np(p\log(n)+p))$
GLPCA [13]	+	–	$\mathcal{O}(n^3)$	$\mathcal{O}(n(p\log(n)+n^2))$
NMF [17]	–	–	$\mathcal{O}(Inpc)$	$\mathcal{O}(Inpc)$
GNMF [6]	+	–	$\mathcal{O}(Inpc)$	$\mathcal{O}(np(Ic+\log(n)))$
MMF [37]	+	–	$\mathcal{O}(((p+c)c^2+pc)I)$	$\mathcal{O}(((p+c)c^2+pc)I+pn\log(n))$
RPCA [7]	–	–	$\mathcal{O}(Inp^2)$	$\mathcal{O}(np(Ip+\log(n)))$
RPCAG [29]	+	–	$\mathcal{O}(Inp^2)$	$\mathcal{O}(np(Ip+\log(n)))$

D. Datasets

TABLE IV. DETAILS OF THE DATASETS USED FOR CLUSTERING EXPERIMENTS IN THIS WORK.

Dataset	**Samples**	**Dimension**	**Classes**
CMU PIE	1200	32×32	30
ORL	400	56×46	40
COIL20	1400	32×32	20
YALE	165	32×32	11
MNIST	50000	28×28	10
USPS	3500	16×16	10
MFEAT	400	409	10

E. Parameter Selection

TABLE V. RANGE OF PARAMETER VALUES FOR EACH OF THE MODELS CONSIDERED IN THIS WORK. c IS THE RANK OR DIMENSION OF SUBSPACE, λ IS THE WEIGHT ASSOCIATED WITH THE SPARSE TERM FOR ROBUST PCA FRAMEWORK [7] AND γ IS THE PARAMETER ASSOCIATED WITH THE GRAPH REGULARIZATION TERM.

Model	**Parameters**	**Parameter Range**
NCut [30]	c	$c \in \{2^1, 2^2, \cdots, \min(n,p)\}$
LE [4]		
PCA		
GLPCA [13]	c, γ	$c \in \{2^1, 2^2, \cdots, \min(n,p)\}$ $\gamma \implies \beta$ using [13] $\beta \in \{0.1, 0.2, \cdots, 0.9\}$
MMF [37]	c, γ	$c \in \{2^1, 2^2, \cdots, \min(n,p)\}$
NMF [17]	c	
GNMF [6]	c, γ	$\gamma \in \{2^{-3}, 2^{-2}, \cdots, 2^{10}\}$
RPCA [7]	λ	$\lambda \in \{\frac{2^{-3}}{\sqrt{\max(n,p)}} : 0.1 : \frac{2^3}{\sqrt{\max(n,p)}}\}$
RPCAG [29]	λ, γ	$\gamma \in \{2^{-3}, 2^{-2}, \cdots, 2^{10}\}$
FRPCAG	γ_1, γ_2	$\gamma_1, \gamma_2 \in \{1, 2, \cdots, 100\}$

F. Validation Error Grid

Table VI presents the rank of the recovered low-rank matrix from different samples and corruptions of the MNIST dataset using $\gamma_1 = \gamma_2 = 2$. For a fixed size of corruptions, the same model parameters recover approximately the same rank for different sizes of the dataset. Fig. 9 presents the validation error over (γ_1, γ_2) grid for the same experimental settings of ORL dataset. It can be observed that our model attains a low validation error for a broad range of the (γ_1, γ_2) grid irrespective of the level of corruptions. The minimum for all levels of corruptions occur approximately at $\gamma_1 = \gamma_2 = 2$.

TABLE VI. RANK OF THE LOW-RANK MATRIX RECOVERED BY USING $\gamma_1 = \gamma_2 = 2$ FOR DIFFERENT SAMPLES AND CORRUPTIONS OF THE MNIST DATASET. THE RECOVERED RANK IS QUITE ROBUST TO PARAMETERS WHICH SHOWS THAT THE MODEL PARAMETERS ARE ROBUST TO THE SIZE OF THE DATASET.

Samples	**No occlusions**	**15% occlusions**	**25% occlusions**
1000	23	16	11
2000	22	15	11
5000	23	17	12

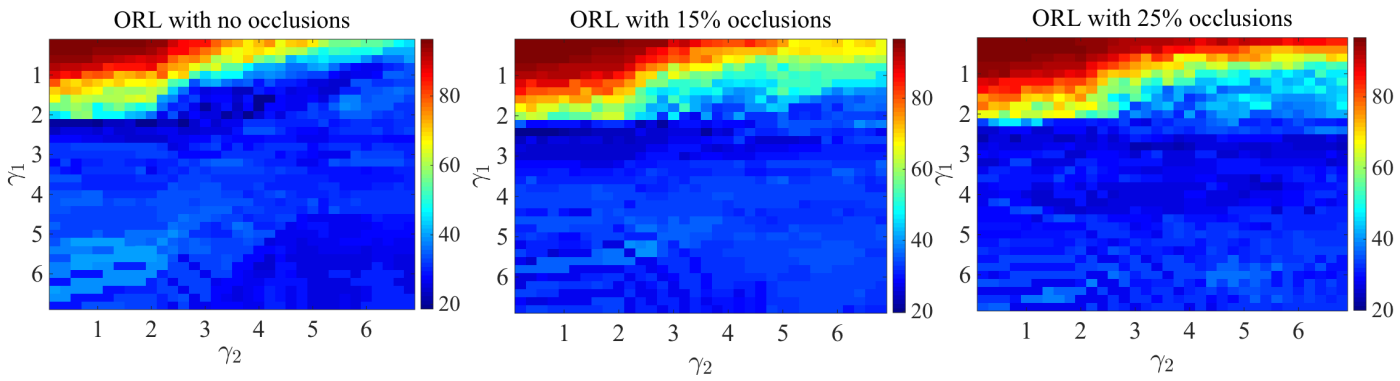


Fig. 9. Validation error variation over (γ_1, γ_2) grid for different levels of corruption for ORL dataset. Similar experimental conditions were used as reported in Fig. 7