TENSOR ROBUST PCA ON GRAPHS

Nauman Shahid, Francesco Grassi, Pierre Vandergheynst

shahidn@utrc.utc.com, grassifr@utrc.utc.com, pierre.vandergheysnt@epfl.ch
* United Technologies Research Center, Penrose Business Centre, Penrose Wharf, Cork, Ireland,
STI IEL LTS2, Ecole Polytechnique Federale de Lausanne, CH-1015, Switzerland

ABSTRACT

We propose a graph signal processing framework to overcome the computational burden of Tensor Robust PCA (TRPCA). Our framework also serves as a convex alternative to graph regularized tensor factorization methods. Our method is based on projecting a tensor onto a lower-dimensional graph basis and benefits from significantly smaller SVDs as compared to TRPCA. Qualitative and computational experiments on several 2D and 3D tensors reveal that for the same reconstruction quality, our method attains up to 100 times speed-up on a low-rank and sparse decomposition application.

Index Terms— low-rank matrix factorization, tensor low-rank and sparse decomposition, graph signal processing

1. INTRODUCTION & RELATED WORK

Tensor Robust PCA (TRPCA) [1], [2], or low-rank and sparse decomposition (L+S) spans a broad variety of applications such as dynamic MRI, 3D point cloud denoising, background separation from videos and many others. It decomposes a tensor $\mathcal{Y}$ into sum of low-rank $\mathcal{X}$ and sparse tensors $\mathcal{S} = \mathcal{Y} - \mathcal{X}$, by solving the following convex optimization problem:

$$\min_{\mathcal{X}} \left\| \text{vec}(\mathcal{Y}) - \text{vec}(\mathcal{X}) \right\|_1 + \gamma \sum_{\mu} \|X^{\mu}\|_*, \qquad (1)$$

where $\| \cdot \|_1$ denotes the element-wise l_1 norm of the tensor, $\sum_{\mu} \|X^{\mu}\|_*$ denotes the tensor nuclear norm, which is the sum of nuclear norms (sum of singular values) of the matrices X^{μ} formed by matricizing $\mathcal{X}$ along the μ^{th} mode [3] and γ is a tunable regularization parameter. However, for a tensor $\mathcal{Y} \in \Re^{n \times n \times n}$, TRPCA via iterative SVDs of 3 matrices $X \in \Re^{n \times n^2}$ [1] costs $O(n^4)$ per iteration which is unfeasible even for n as small as 100.

Several techniques have been proposed in the past to overcome the computational cost by sampling and randomization [4, 5, 6] along the *data dimension* p. However, the problem of large number of samples n remains unresolved.

In a different context, i.e., to exploit inherent data structure, the notion of graph Laplacian L regularization [7] has been extensively used in the form of low-rank tensor factor priors [8], [9], [10], [11], [12], [13]. Tailoring these models for L+S application, they can be written as:

$$\min_{U_r^{\mu}, \mathcal{C}, \mathcal{X}} \left\| \text{vec}(\mathcal{Y}) - \text{vec}(\mathcal{X}) \right\|_1 + \gamma \sum_{\mu} \text{tr}(U^{\mu}{}_r^{\top} L^{\mu} U_r^{\mu})$$
$$\text{s.t} \;\; \text{vec}(\mathcal{X}) = (U_r^1 \otimes U_r^2 \otimes U_r^3)\, \text{vec}(\mathcal{C}) \qquad (2)$$

where $U_r^{\mu} \in \Re^{n \times r}$ is the μ^{th}-mode basis, $\mathcal{C} \in \Re^{r \times r \times r}$ is the core tensor, where $r \ll n$, $\text{vec}(\cdot)$ denotes the vectorization operation and and $\otimes$ denotes the matrix Kronecker product. This model avoids the computation of expensive SVDs due to its factorized nature but this advantage comes at the cost of non-convexity.

To the best of our knowledge, the only effort to jointly exploit *graph structure* and *reduce the computation cost* of matrix L+S in a *convex* manner is by the authors of [14], [15]. However, their methods cannot be directly extended for tensors because of computational cost of regularizer $\text{tr}(XLX^{\top})$ on large matrices X.

Contributions: Our goal in this paper is to understand if we can achieve the best of both, i.e, convexity of eq.(1) and graph structure of eq.(2) for tensor L+S while avoiding their corresponding shortcomings. We target this via a two-step methodology. First we define the notion of "low-frequency power concentration" in graph frequencies, which significantly reduces the size of a low-rank tensor by projecting on the graph basis. Second, we use this notion to derive from eq.(2), a convex L+S model which enjoys graph regularization and reduces the computation time by orders of magnitude as compared to TRPCA (1).

Notation and Definitions: Throughout we will use the notions of mode, matricization and multilinear rank of tensors, which are well defined in [3]. We will use subscripts r, k, k^2, etc., to represent the number of columns of a matrix, superscript μ to refer to the μ^{th} mode and $-\mu$ to refer to the kronecker product of all matrices except the μ^{th} mode. We work with 3D tensors of size n and rank r along each mode but the analysis is valid for general dimensions. We will also use the notion of k_{nn}-nearest neighbor graphs [7].

[0]*This work was done during the PhD of authors at Ecole Polytechniuqe Federlae de Lausanne (Switzerland). It was supported by the Swiss National Foundation (SNF) grant 200021_154350/1 for the project "Towards Signal Processing on Graphs".

A k_{nn}-graph constructed between the columns of a matrix $X \in \Re^{p\times n}$ is a tuple $G = \{V, E, W\}$ where V is a set of vertices, E a set of edges, and $W : V \times V \to R_+$ a weight function. Each vertex v_i is connected on average to k_{nn} nearest neighbor vertices in Euclidean distance. Each entry of the weight matrix $W \in \Re_+^{|V|\times|V|}$ is assigned via Gaussian kernel function of the signals x_i, x_j on the vertices v_i, v_j. Throughout, we use combinatorial Laplacian $L = D - W$, where D is the degree matrix, to represent G. The eigenvector decomposition of L is given as $L = P\Lambda P^\top$.

2. PROPOSED MODEL

It is a well-known fact that the graph regularization term of the form $\mathrm{tr}(U_r^\top L U_r)$ in eq.(2) ensures the smoothness of factors U_r w.r.t to the graph Laplacian. More specifically, let P_k be the first k eigenvectors of the graph Laplacian L of the k_{nn}-graph G, then the regularization ensures that the columns of U_r belong to the span of P_k [14]. We show here that it is also possible to ensure this smoothness using a specific property of low-rank tensors, which we call as "Low Frequency Power Concentration". We use this property to 1) derive a computationally tractable nuclear norm regularizer from $\mathrm{tr}(U_r^\top L U_r)$ and 2) fix the basis U_r in eq.(2) to P_k to avoid updating U_r, hence eliminating the non-convexity.

Low Frequency Power Concentration: Consider the face low-rank tensor $\mathcal{Y} \in \Re^{542\times 333\times 148}$ as shown in Fig.1. The singular values of all the 3 modes clearly show that it is a low-rank tensor. We now show that the modes of this tensor have most of their power concentrated in the top few graph eigenvectors. We matricize the tensor $\mathcal{Y}$ along mode 1 to get the matrix $Y^1 \in \Re^{542\times(333\cdot 148)}$ and then construct a k_{nn}-graph Laplacian $L^1 \in \Re^{542\times 542}$ between the rows of Y^1. Next, we compute the eigenvalue decomposition of $L^1 = P^1\Lambda^1 {P^1}^\top$ and the Graph Fourier Transform $\hat{Y^1} = {P^1}^\top Y^1$ [7]. Finally, we compute the covariance matrix in graph spectral domain $10\log(\hat{Y^1}\hat{Y^1}^\top)$, which shows the power distribution across different graph frequencies. It can be seen from the 3rd plot in Fig. 1 that most of the power for mode 1 is concentrated in the first $k \ll 542$ graph frequencies. The same holds for modes 2 and 3, as shown in the normalized power concentration plots. We can use this observation to state for the low-rank tensor:

$$\mathrm{vec}(\mathcal{X}) = (P_k^1 \otimes P_k^2 \otimes P_k^3)\,\mathrm{vec}(\mathcal{Z}), \tag{3}$$

where $\mathcal{Z} \in \Re^{k\times k\times k}$ is the low-dimensional projection of $\mathcal{X}$. We call this projection as *graph core tensor (GCT)*. P_k^μ denote the first k Laplacian eigenvectors of the k_{nn}-graph G^μ constructed between the rows of matricized tensor Y^μ.

Model Derivation: Now we derive our model starting from eq.(2). First note that using $L^\mu = P^\mu\Lambda^\mu {P^\mu}^\top$, we can write $\sum_\mu \mathrm{tr}({U_r^\mu}^\top L^\mu U_r^\mu) = \sum_\mu \|\sqrt{\Lambda}P^\top U_r\|_F^2$. Using the results from [16], which state that the Frobenius norm minimization of the above form is equivalent to a weighted nuclear norm and $\mathrm{vec}(\mathcal{X}) = (U_r^1 \otimes U_r^2 \otimes U_r^3)\,\mathrm{vec}(\mathcal{C})$, we can write eq. (2) as:

$$\min_{\mathcal{X}} \Big\| \mathrm{vec}(\mathcal{Y}) - \mathrm{vec}(\mathcal{X}) \Big\|_1 + \gamma \sum_\mu \Big\| \sqrt{\Lambda^\mu} {P^\mu}^\top X^\mu P^{-\mu} \sqrt{\Lambda^{-\mu}} \Big\|_*,$$

where $X \in \Re^{n\times n^2}$, $P^{-\mu}, \Lambda^{-\mu}$ is the Kronecker product of the eigenvectors and eigenvalues of all modes except μ. The weighted nuclear norm is still on the big matrix of size $n \times n^2$ and the weights are a function of graph eigenvalues.

Let $P^\mu = [P_k^\mu, \bar{P}_k^\mu]$, where bar denotes the set of complement graph eigenvectors. Using ${P_k^\mu}^\top \bar{P}_k^\mu = 0$ and the power concentration property from eq.(3), we can reduce the size of nuclear norm on X^μ significantly and get our proposed model:

$$\min_{\mathcal{Z}} \Big\| \mathrm{vec}(\mathcal{Y}) - (P_k^1 \otimes P_k^2 \otimes P_k^3)\,\mathrm{vec}(\mathcal{Z}) \Big\|_1 + \gamma \sum_\mu \Big\| \sqrt{\Lambda_k^\mu} Z^\mu \sqrt{\Lambda_{k^2}^{-\mu}} \Big\|_*. \tag{4}$$

We call our proposed model Tensor Robust PCA on Graphs (TRPCAG). Note that in contrast to eq.(1), which reduces the rank of each mode iteratively by starting from a full-rank $\mathcal{Y}$ and performing expensive SVDs, we fix the rank $k \ll n$ upfront by initializing the basis U_r^μ with Laplacian eigenvectors P_k^μ. Then, we iteratively 1) reduce the rank from k to r and 2) refine the basis P_k^μ to approximate U_r^μ via nuclear norm minimization of $\mathcal{Z}$. This reduces the cost of SVD from $O(n^4)$ to $O(k^4)$, where $k \ll n$ and also makes the problem convex.

Note that TRPCAG (eq.(4)) can be easily extended for matrices (2D tensors) as well:

$$\min_Z \Big\| Y - P_k^1 Z {P_k^{-1}}^\top \Big\|_1 + \gamma \| \sqrt{\Lambda_k^1} Z \sqrt{\Lambda_k^{-1}} \|_*. \tag{5}$$

Optimization Solution: We use Parallel Proximal Splitting method [17] to solve this problem. First, we re-write $s_0 = \|\mathrm{vec}(\mathcal{Y}) - (P_k^1 \otimes P_k^2 \otimes P_k^3)\,\mathrm{vec}(\mathcal{Z})\|_1 = \|P_k^1 Z^1 {P_{k^2}^{-\mu}}^\top - Y^1\|_1$, $\sum_\mu \|Z^\mu\|_* = s_\mu$ and $f_\mu(\Lambda_k) = \sqrt{\Lambda_k^\mu} \circ \sqrt{\Lambda_{k^2}^{-\mu}}$.

Let $\Omega(Z, \tau)$ denote the element-wise soft-thresholding matrix operator: $\Omega(Z, \tau) = \mathrm{sgn}(Z)\max(|Z| - \tau, 0)$, then we can define $D_\mu(Z^\mu, \tau) = A_k^1 \Omega(R^\mu, \tau) {A_{k^2}^{-\mu}}^\top$, as the singular value thresholding operator for matrix Z^μ, where $Z^\mu = A_k^1 R^\mu {A_{k^2}^{-\mu}}^\top$ is any singular value decomposition of Z^μ. Clearly, $\mathrm{prox}_{\gamma s_\mu}(Z^\mu) = D_\mu(Z^\mu, \gamma)$, and

$$\mathrm{prox}_{s_0}(Z^1) = Z^1 + {P_k^1}^\top \Big(\Omega\big(P_k^1 Z^1 {P_{k^2}^{-\mu}}^\top - Y^1, \alpha\gamma f_\mu(\Lambda_k)\big) - P_k^1 Z^1 {P_{k^2}^{-\mu}}^\top - Y^1 \Big) P_{k^2}^{-\mu},$$

where α is the step size. These proximal operators can be used in the Parallel Proximal Splitting Algorithm 1 to obtain the solution to eq.(4).

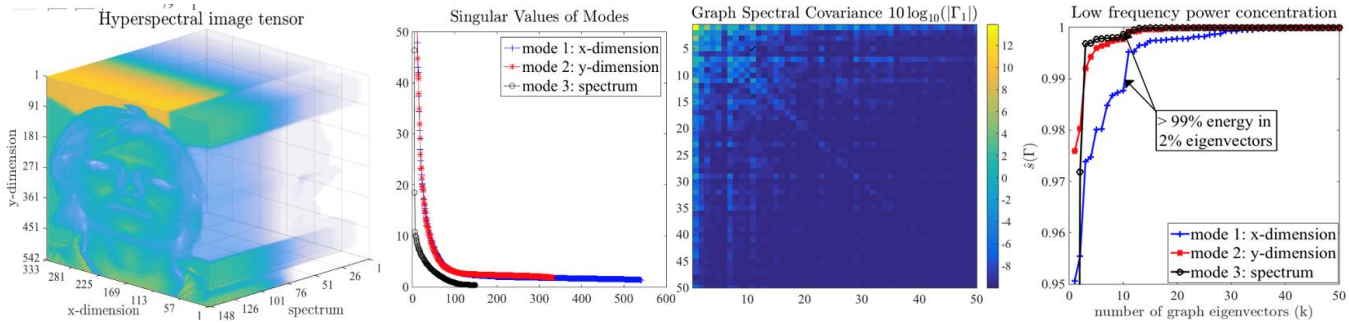

Fig. 1. Hyperspectral face tensor, its singular values, graph spectral covariance and low frequency power concentration.

Algorithm 1 Parallel Proximal Algorithm for TRPCAG

IN: Y^1, $f_\mu(\Lambda_k) = \sqrt{\Lambda_k^\mu} \circ \sqrt{\Lambda_{k^2}^{-\mu}}$, $\gamma, \epsilon \in (0,1), \alpha \geq 0$.
$Z^1_{0,0}, \cdots, Z^1_{0,3} \in R^{k \times k^2}$, all matricized along mode 1.
Set $H^1_0 = \sum_{i=0}^{3} Z^1_{0,i}$
for $j = 1, \cdots, J$ **do**
  $F^1_{j,0} = \text{prox}_{s_0}(H^1_j, \alpha)$
  for $\mu = 1, \cdots, 3$ **do**
    $F^1_{j,\mu} = reshape(\text{prox}_{(\gamma s_\mu)}(H^\mu_j, \gamma(f_\mu(\Lambda_k))))$
  end for
  $\epsilon \leq \beta_j \leq 2 - \epsilon$
  $F^1_j = \sum_\mu F^1_{j,\mu}$
  for $\mu = 0, \cdots, 3$ **do**
    $Z^1_{j+1,\mu} = Z^1_{j,\mu} + \beta_j(2F^1_j - H^1_j - F^1_{j,\mu})$
  end for
  $H^1_{j+1} = H^1_j + \beta_j(F^3_j - H^1_j)$
end for
OUTPUT: H^1_{j+1}

Complete Algorithm: 1) Matricize the tensor $\mathcal{Y}$ along each of the modes $Y^\mu \in \Re^{n \times n^2}$ and construct a k_{nn}-graph G^μ between the rows for each matrix. 2) Determine the first k eigenvectors of the combinatorial graph Laplacians L^μ. 3) Solve the optimization problem in eq. (4) using Algorithm 1.

Computational Complexity: Graph construction between the rows of a matrix of size $Y^\mu \in R^{n \times n^2}$, using FLANN [18] scales as $\mathcal{O}(n \log(n))$ and the eigenvector computation scales as $\mathcal{O}(nk^2)$. Computing the $\text{prox}_{\gamma s_\mu}$ involves the SVD of $Z^\mu \in R^{k \times k^2}$, so it costs $\mathcal{O}(Ik^4)$. The computation of prox_{s_0} involves matrix multiplications and scales as $\mathcal{O}(In^3k + In^2k^2)$. Thus, the overall complexity of our method scales as $\mathcal{O}(n \log(n) + nk^2 + I(n^3k + n^2k^2))$, where $k \ll n$ and I is the number of iterations. The graph construction and eigenvector decomposition are performed only once and the biggest computational cost is incurred by the matrix multiplication $\mathcal{O}(I(n^3k + n^2k^2))$, which can be parallelized on the GPU. This is a significant reduction as compared to TRPCA, which scales as $\mathcal{O}(In^4)$ and cannot be parallelized on a GPU because it involves SVD computations.

3. EXPERIMENTAL RESULTS

We perform L+S experiments on various 2D and 3D artificial and real tensor datasets to evaluate the performance of TRPCAG against other state-of-the-art methods such as Tensor Robust PCA (TRPCA) [1], Robust PCA (RPCA) [19], Robust PCA on Graphs (RPCAG) [20], Fast Robust PCA on Graphs (FRPCAG) [21] and Compressive PCA (CPCA) [15]. We study both the qualitative and computational gains, where the qualitative study involves the visual quality and reconstruction error of recovered low-rank datasets. The details of the datasets used in various experiments, the purpose of experiments and compared methods are summarized in Table 1. Experiments on 2D datasets use eq.(5) and those on 3D datasets use eq.(4).

Parameters: For all the experiments, the k_{nn} graphs are constructed from the rows of each of the flattened modes of the tensor, using $k_{nn} = 10$, a Gaussian kernel for weighting the edges and the FLANN library [18]. For a fair comparison, all the methods are properly tuned for their hyper-parameters and best results are reported. Since TRPCAG requires explicit specification of the rank k for every mode of the tensor, or the size of GCT $\mathcal{Z} \in R^{k \times k \times k}$, we use $k > r$, where r is the actual rank of a given mode. Furthermore, we tune the regularization parameter $\gamma \in (0, 30)$.

Background Separation from 2D video datasets: The 1st and 2nd rows of Fig. 2 present experiments on the 2D real video datasets obtained from an airport lobby (25344×900) and shopping mall (81920×900), where every frame is vectorized and stacked as the columns of a matrix. The goal is to separate the static low-rank component from the sparse part (moving people) in the video. The results of TRPCAG (eq.(5), Algorithm 1) are compared with RPCA, RPCAG, FRPCAG and CPCA (5,1) with a downsampling factor of 5 along the frames. Clearly, TRPCAG recovers a low-rank which is qualitatively equivalent to the other methods in a time which is *100 times less than RPCA and RPCAG and an order of magnitude less as compared to FRPCAG*. Furthermore, TR-

Table 1. Details of the datasets, experiments and methods used for L+S experiments in this work.

Dataset	Dimension	Experiment	Compared Methods
Airport lobby video	2D (25344×900)	quality & timing	RPCA [19], RPCAG [20], FRPCAG [21], CPCA [15]
Shopping mall video	2D (81920×900)	quality & timing	RPCA [19], RPCAG [20], FRPCAG [21], CPCA [15]
Snowfall video	3D ($1920 \times 1080 \times 500$) 1.5GB	timing	TRPCA [1]
artificial dataset	2D (400×400 to 12800×12800)	timing	RPCA [19], RPCAG [20], FRPCAG [21]

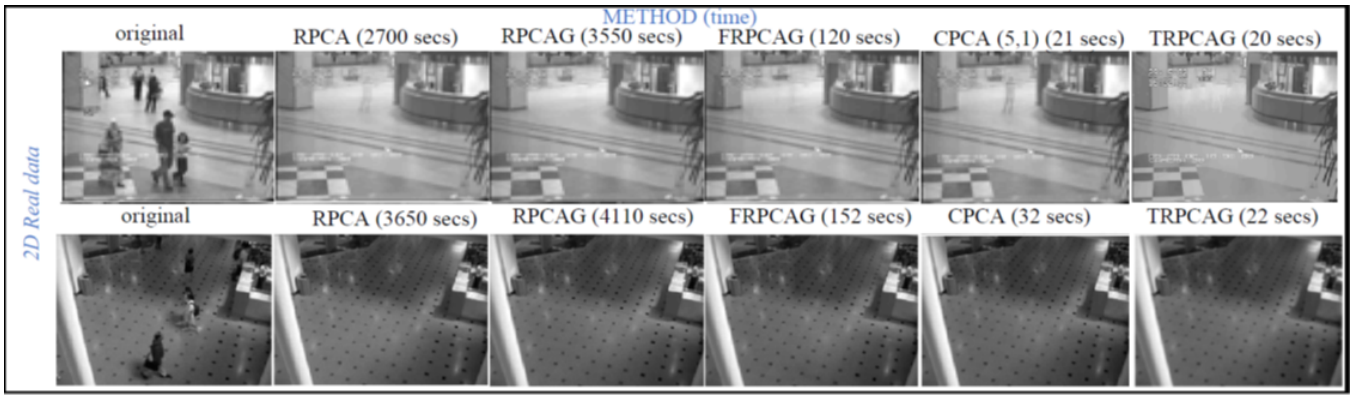

Fig. 2. L+S experimental results of TRPCAG on 2D video datasets.

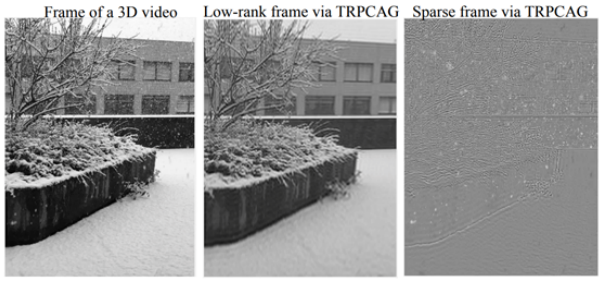

Fig. 3. L+S decomposition on 3D video

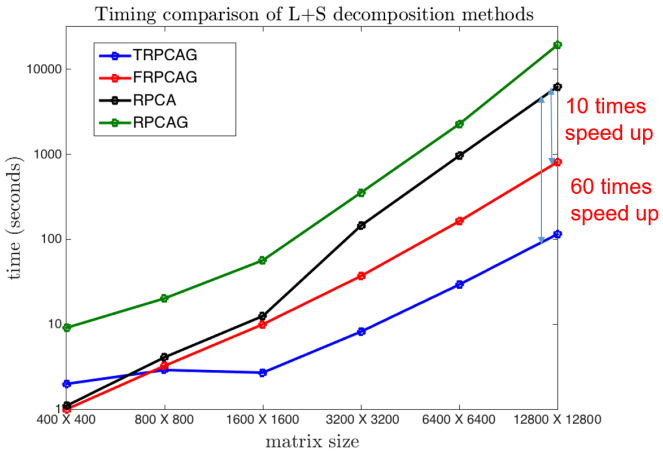

Fig. 4. L+S decomposition timing of various methods. All methods are tuned to give a reconstruction error of 0.01.

PCAG requires the same time as sampling-based CPCA but it recovers a better quality low-rank structure.

Scalability of TRPCAG on 3D video: To show the scalability of TRPCAG as compared to TRPCA, we recorded a video of snow-fall at the campus and tried to separate the snow-fall from the low-rank background via both methods. For this 1.5GB video of dimension $1920 \times 1080 \times 500$, TRPCAG (eq.(4), Algorithm 1) with core tensor size $100 \times 100 \times 50$, took less than 3 minutes, whereas TRPCA did not converge even in 4 hours. The result obtained via TRPCAG is visualized in the right half of the 5th row of Fig. 3.

Timing on artificial datasets: We also generated artificial square low-rank matrices of size ranging from 400×400 to 12800×12800 and added sparse noise with 0.1 standard deviation on 10% of the entries. We tune all the methods to attain a reconstruction error of 0.01 and then compare the timings in Fig. 4. For the same reconstruction quality, TRPCAG is upto 60 times faster than RPCA and 10 times faster than FRPCAG. Clearly, this shows the computational advantage of using our method.

4. CONCLUSION

We propose a convex alternative to graph regularized tensor factorization methods to overcome the computational burden of Tensor Robust PCA (TRPCA). We make use of significantly smaller SVDs as compared to TRPCA by projecting the tensor onto a graph basis. Experiments on several 2D and 3D tensors reveal that for the same reconstruction quality, our method attains upto 100 times speed-up on low-rank and sparse decomposition application.

5. REFERENCES

[1] Donald Goldfarb and Zhiwei Qin, "Robust low-rank tensor recovery: Models and algorithms," *SIAM Journal on Matrix Analysis and Applications*, vol. 35, no. 1, pp. 225–253, 2014.

[2] Parikshit Shah, Nikhil Rao, and Gongguo Tang, "Sparse and low-rank tensor decomposition," in *Advances in Neural Information Processing Systems*, 2015, pp. 2548–2556.

[3] Andrzej Cichocki, Danilo Mandic, Lieven De Lathauwer, Guoxu Zhou, Qibin Zhao, Cesar Caiafa, and Huy Anh Phan, "Tensor decompositions for signal processing applications: From two-way to multiway component analysis," *Signal Processing Magazine, IEEE*, vol. 32, no. 2, pp. 145–163, 2015.

[4] Tae-Hyun Oh, Yasuyuki Matsushita, Yu-Wing Tai, and In So Kweon, "Fast randomized singular value thresholding for nuclear norm minimization," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 4484–4493.

[5] Mostafa Rahmani and George Atia, "High dimensional low rank plus sparse matrix decomposition," *arXiv preprint arXiv:1502.00182*, 2015.

[6] Mostafa Rahmani and George K Atia, "Randomized robust subspace recovery for big data," in *Machine Learning for Signal Processing (MLSP), 2015 IEEE 25th International Workshop on*. IEEE, 2015, pp. 1–6.

[7] David I Shuman, Sunil K Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *Signal Processing Magazine, IEEE*, vol. 30, no. 3, pp. 83–98, 2013.

[8] Yi-Lei Chen, Chiou-Ting Hsu, and Hong-Yuan Mark Liao, "Simultaneous tensor decomposition and completion using factor priors," *IEEE transactions on pattern analysis and machine intelligence*, vol. 36, no. 3, pp. 577–591, 2014.

[9] Can Wang, Xiaofei He, Jiajun Bu, Zhengguang Chen, Chun Chen, and Ziyu Guan, "Image representation using laplacian regularized nonnegative tensor factorization," *Pattern Recognition*, vol. 44, no. 10, pp. 2516–2526, 2011.

[10] Yui Man Lui, J Ross Beveridge, and Michael Kirby, "Action classification on product manifolds," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2010, pp. 833–839.

[11] Yui Man Lui, "Human gesture recognition on product manifolds," *Journal of Machine Learning Research*, vol. 13, no. Nov, pp. 3297–3321, 2012.

[12] Guoqiang Zhong and Mohamed Cheriet, "Large margin low rank tensor analysis," *Neural computation*, vol. 26, no. 4, pp. 761–780, 2014.

[13] X. Li, M. K. Ng, G. Cong, Y. Ye, and Q. Wu, "Mr-ntd: Manifold regularization nonnegative tucker decomposition for tensor data dimension reduction and representation," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 8, pp. 1787–1800, Aug 2017.

[14] Nauman Shahid, Nathanael Perraudin, Vassilis Kalofolias, Gilles Puy, and Pierre Vandergheynst, "Fast robust PCA on graphs," *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 4, pp. 740–756, 2016.

[15] Nauman Shahid, Nathanael Perraudin, Gilles Puy, and Pierre Vandergheynst, "Compressive pca for low-rank matrices on graphs," *IEEE transactions on Signal and Information Processing over Networks*, 2016.

[16] Nikhil Rao, Hsiang-Fu Yu, Pradeep K Ravikumar, and Inderjit S Dhillon, "Collaborative filtering with graph information: Consistency and scalable methods," in *Advances in Neural Information Processing Systems*, 2015, pp. 2107–2115.

[17] Patrick L Combettes and Jean-Christophe Pesquet, "Proximal splitting methods in signal processing," in *Fixed-point algorithms for inverse problems in science and engineering*, pp. 185–212. Springer, 2011.

[18] Marius Muja and David Lowe, "Scalable nearest neighbour algorithms for high dimensional data," 2014.

[19] Emmanuel J Candès, Xiaodong Li, Yi Ma, and John Wright, "Robust principal component analysis?," *Journal of the ACM (JACM)*, vol. 58, no. 3, pp. 11, 2011.

[20] Nauman Shahid, Vassilis Kalofolias, Xavier Bresson, Michael Bronstein, and Pierre Vandergheynst, "Robust principal component analysis on graphs," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 2812–2820.

[21] Nauman Shahid, Nathanael Perraudin, Vassilis Kalofolias, Gilles Puy, and Pierre Vandergheynst, "Fast robust pca on graphs," *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 4, pp. 740–756, 2016.