RIEMANNIAN CUR DECOMPOSITIONS FOR ROBUST PRINCIPAL COMPONENT ANALYSIS

Keaton Hamm & Mohamed Meskini
Department of Mathematics
University of Texas at Arlington
Arlington, TX 76019, USA
keaton.hamm@uta.edu
mxm6670@mavs.uta.edu

HanQin Cai
Department of Statistics and Data Science
University of Central Florida
Orlando, FL 32816, USA
hqcai@ucf.edu

ABSTRACT

Robust Principal Component Analysis (PCA) has received massive attention in recent years. It aims to recover a low-rank matrix and a sparse matrix from their sum. This paper proposes a novel nonconvex Robust PCA algorithm, coined Riemannian CUR (RieCUR), which utilizes the ideas of Riemannian optimization and robust CUR decompositions. This algorithm has the same computational complexity as Iterated Robust CUR, which is currently state-of-the-art, but is more robust to outliers. RieCUR is also able to tolerate a significant amount of outliers, and is comparable to Accelerated Alternating Projections, which has high outlier tolerance but worse computational complexity than the proposed method. Thus, the proposed algorithm achieves state-of-the-art performance on Robust PCA both in terms of computational complexity and outlier tolerance.

1 INTRODUCTION

Principal Component Analysis (PCA) is one of the most fundamental tools to uncover low-dimensional structure in high-dimensional data. Robust PCA proposed by (Wright et al., 2009; Candès et al., 2011) seeks to find an incoherent, low-rank matrix from observations of it corrupted by sparse outliers (Bouwmans et al., 2018). That is, given observations of the form $D = L + S$, where L is the underlying low-rank signal and S is a matrix of sparse, but arbitrary magnitude outliers, one seeks to find a good approximation of L. Robust PCA has a wide range of applications, for instance, video static background subtraction (Jang et al., 2016), singing-voice separation (Huang et al., 2012), ultrasound imaging (Cai et al., 2021d), face recognition (Wright et al., 2008), image alignment (Peng et al., 2012), community detection (Chen et al., 2012), and NMR spectrum recovery (Cai et al., 2021a; 2022a).

To get around the computational cost of the semidefinite program for solving the convex relaxation to Robust PCA, many recent methods attack the nonconvex optimization problem directly

$$\begin{aligned} \underset{L,S}{\text{minimize}} \quad & \|D - L - S\|_{\mathrm{F}} \\ \text{subject to} \quad & \text{rank}(L) \leq r, \\ & \|S\|_0 \leq \Delta. \end{aligned} \tag{1}$$

Several recent algorithms use an iterative procedure in which one forms alternating approximations of L and S, in which L is updated by projecting $D - S$ onto the Riemannian manifold of rank-r matrices (Vandereycken, 2013), denoted $\mathcal{M}_r$, and S is updated by projecting $D - L$ onto the set of sparse matrices that have at most Δ non-zero entries, denoted $\mathcal{S}_\Delta$. This technique is called Alternating Projections (AltProj) (Netrapalli et al., 2014). One drawback of this algorithm is that it requires computation of the SVD of the full $n \times n$ matrix $D - S$, which carries complexity $\mathcal{O}(n^2 r)$. (Cai et al., 2019) propose Accelerated Alternating Projections (AccAltProj) which instead projects $D - S$ onto the tangent space of manifold $\mathcal{M}_r$ at the current estimation of L, which is significantly faster, while remaining robust to outliers. The trick of tangent space projection in AccAltProj coincides with the idea of Riemannian optimization (Wei et al., 2020), thus AccAltProj can be viewed as a manifold method.

Subsequently, (Cai et al., 2020) proposed the use of CUR decompositions (Drineas et al., 2006; Mahoney & Drineas, 2009; Hamm & Huang, 2020) to provide an even faster nonconvex Robust PCA solver, Iterated Robust CUR (IRCUR). IRCUR reduced the computational complexity of AccAltProj and previous methods from $\mathcal{O}(n^2 r)$ to $\mathcal{O}(r^2 n \log^2 n)$. However, one drawback of IRCUR is that it is not capable of handling as many outliers as the previous methods (i.e., Δ must be smaller for IRCUR to be successful compared to AccAltProj). To bridge this gap, this paper proposes ***Riemannian CUR*** (RieCUR), which combines the idea of using CUR decompositions with the tangent space projections of AccAltProj.

The result is an algorithm that combines the best of both methods in the following way:

- RieCUR has complexity $\mathcal{O}(r^2 n \log^2 n)$, which is the same as IRCUR (although the constant appears to be larger for RieCUR based on experiments in the sequel).
- RieCUR appears to tolerate outliers as well as AccAltProj in terms of reconstruction vs. sparsity based on our numerical experiments, whereas IRCUR degrades as the amount of outliers increases.

Additionally, many of the nonconvex solvers initialize guesses of L and S via a single truncated SVD of D. We show in our experiments here that one can use fast matrix sketching approaches for approximating the SVD during initialization while not sacrificing recovery of L and S. This allows for significant speedups in practical settings.

2 Proposed Approach

Here we detail our algorithm, coined RieCUR, as well as its implementation details. First, let us collect some notation and assumptions.

2.1 Notation

We focus on square matrices $D, L, S \in \mathbb{R}^{n \times n}$, but note the proposed algorithm works for rectangular matrices (see (Cai et al., 2019) for details on converting rectangular Robust PCA problems to square ones). The SVD of $A \in \mathbb{R}^{n \times n}$ is $W \Sigma V^\top$ (or $W_A \Sigma_A V_A^\top$ if it is unclear from context), where W and V are orthogonal (called the left and right singular vectors, respectively) and $\Sigma = \operatorname{diag}(\sigma_1, \dots, \sigma_n)$ with $\sigma_1 \geq \cdots \geq \sigma_n \geq 0$ (called the singular values of A. The truncated SVD of A with rank parameter r is denoted $A_r = W_r \Sigma_r V_r^\top$, where W_r and V_r are the $n \times r$ submatrices of W and V, respectively corresponding to choosing the first r columns of each, and Σ_r is the $r \times r$ leading principal minor of Σ. The Frobenius norm of a matrix is $\|A\|_{\mathrm{F}} := (\sum_{i,j} |A_{ij}|^2)^{\frac{1}{2}}$, the ℓ_0-"norm" of a matrix is the number of nonzero entries of it, i.e., $\|S\|_0 := |\{(i,j) : S_{ij} \neq 0\}|$ (here $|\cdot|$ denotes set cardinality), and $\|\cdot\|_2$ will here always denote the Euclidean norm of a vector.

Given an index set $I \subset \{1, \dots, n\}$, $A_{I,:}$ denotes the $|I| \times n$ submatrix of A corresponding to choosing the rows of A indexed by I, and similar meanings are assigned to $A_{I,J}$ and $A_{:,J}$. Finally, if $A = W \Sigma V^\top$, then its Moore-Penrose pseudoinverse (Penrose, 1956) is $A^\dagger = V \Sigma^\dagger W^\top$, where $\Sigma^\dagger$ contains diagonal entries $1/\sigma_i$ if $\sigma_i > 0$ or 0 if $\sigma_i = 0$ along its diagonal.

Finally, given $\zeta > 0$, we define the hard thresholding operator by

$$[\mathcal{T}_\zeta A]_{i,j} := \begin{cases} A_{i,j}, & |A_{i,j}| \geq \zeta; \\ 0, & \text{otherwise.} \end{cases}$$

2.2 Assumptions

We utilize common assumptions for Robust PCA problems to be tractable, namely incoherence and sparsity. A matrix $L \in \mathbb{R}^{n \times n}$ with truncated SVD $W_r \Sigma_r V_r^\top$ is called μ-incoherent provided

$$\max_i \|(W_r)_{i,:}\|_2 \leq \sqrt{\frac{\mu r}{n}}, \quad \max_j \|(V_r)_{j,:}\|_2 \leq \sqrt{\frac{\mu r}{n}}.$$

A matrix $S \in \mathbb{R}^{n \times n}$ is said to be α-sparse provided

$$\|S_{i,:}\|_0 \leq \alpha n, \quad \|S_{:,j}\|_0 \leq \alpha n, \quad \text{for all } i, j = 1, \dots, n.$$

Note that this notion of sparsity requires that S not have an overly large concentration of outliers on any given row or column.

2.3 Review of CUR decompositions and tangent space projections

Here we recall the background of both CUR decompositions and tangent space projections which are used in our main algorithm.

The CUR decomposition of a matrix comes from the observation that if one chooses a column submatrix $C = A_{:,J}$ whose columns span the column space of A and a row matrix $R = A_{I,:}$ whose rows span the row space of A, and let $U = A_{I,J}$, then $A = CU^{\dagger}R$. See (Goreinov et al., 1997; Hamm & Huang, 2020; Strang & Moler, 2022) for more details. Based on random sampling methods for CUR decompositions of incoherent low-rank matrices (Chiu & Demanet, 2013), we sample $\mathcal{O}(r \log n)$ columns and rows for each CUR decomposition done in this work.

During the iterative procedure of Riemannian CUR, if we have estimate L_k and S_k of L and S at iteration k, and L_k has truncated SVD of order r given by $W_k \Sigma_k V_k^{\top}$, then the singular vectors U_k and V_k form a $(2n-r)r$-dimensional subspace of $\mathcal{M}_r$, which is the tangent space at L_k, defined by

$$T_k := \{W_k A^{\top} + B V_k^{\top} : A, B \in \mathbb{R}^{n \times r}\}.$$

One can verify (Vandereycken, 2013) that the projection onto T_k is given by

$$P_{T_k} X = W_k W_k^{\top} X + X V_k V_k^{\top} - W_k W_k^{\top} X V_k V_k^{\top}. \tag{2}$$

The main idea of AccAltProj (Cai et al., 2019) is that $D - L_k$ is projected onto T_k and then projected onto $\mathcal{M}_r$ to obtain S_{k+1}; then L_{k+1} is obtained by hard thresholding $D - S_{k+1}$. The reason this speeds up the procedure of AltProj is that one can compute the SVD of $P_{T_k}(D - S_k)$ much more efficiently than the SVD of $D - S_k$ itself.

The main idea of IRCUR (Cai et al., 2020) and its variants (Cai et al., 2021c;b; 2022b) is that one only every works with column or row submatrices of D, L_k, and S_k in the iterative process. That is, one is only trying to recover $L_{I,:}$, $L_{:,J}$, $S_{I,:}$, and $S_{:,J}$ for given row and column subsets I and J, and at the end, one approximates L by a CUR decomposition of the form $L \approx CU^{\dagger}R$. Since only a few columns and rows of D are utilized and held in memory, this algorithm is significantly faster than previous ones that required operations on the entire $n \times n$ matrices in question.

2.4 Riemannian CUR algorithm

We are now ready to state our algorithm below. The stopping criterion is in terms of an empirical relative error defined by

$$e_k = \frac{\|(D - L_k - S_k)_{I,:}\|_{\mathrm{F}} + \|(D - L_k - S_k)_{:,J}\|_{\mathrm{F}}}{\|D_{I,:}\|_{\mathrm{F}} + \|D_{:,J}\|_{\mathrm{F}}},$$

where I and J are the indices of the submatrices of $P_{T_k}(D - S_k)$ formed at each iteration. This is the same error term as in IRCUR, and is used because our algorithm does not form all of L or S during the iterative part, but rather only tracks submatrices of them.

The main difference of Algorithm 1 compared with AccAltProj and IRCUR is that it utilizes submatrices of the tangent space projection $P_{T_k}(D - S_k)$ rather than the entire matrix as AccAltProj does, and IRCUR does not utilize the tangent space projection at all, but rather works with submatrices of $D - S_k$ directly. Use of the tangent space projection in RieCUR adds a small amount of computation time to IRCUR but with the benefit of making the procedure more robust to outliers, as CUR decompositions are known to suffer from outliers.

2.5 Implementation details

There are three primary details to consider in the implementation of Algorithm 1: initialization, forming the column and row submatrices of $P_{T_k}(D - S_k)$, and how to choose the initial threshold γ, which here is a hyperparameter of the algorithm, but we note that one can use the choices of (Cai et al., 2019), and the algorithm regularly converges in thorough experimentation; however, at present

Algorithm 1 Riemannian CUR (RieCUR) for Robust PCA

Input: D: observed corrupted data matrix; r: rank; ε: target precision level; ζ_0: initial threshold; γ: threshold decay rate; $|I|, |J|$: number of rows and columns to sample.
Uniformly sample row indices I and column indices J
Initialize L_0 and S_0
$k = 0$
while $e_k > \varepsilon$ **do**
  (Optional:) Resample row and column indices
  $C_{k+1} = (P_{T_k}(D - S_k))_{:,J}$
  $R_{k+1} = (P_{T_k}(D - S_k))_{I,:}$
  $U_{k+1} = (P_{T_k}(D - S_k))_{I,J}$
  $L_{k+1} = C_{k+1}U_{k+1}^{\dagger}R_{k+1}$
  $\zeta_{k+1} = \gamma^k \zeta_0$
  $(S_{k+1})_{:,J} = \mathcal{T}_{\zeta_{k+1}}(D - L_{k+1})_{:,J}$
  $(S_{k+1})_{I,:} = \mathcal{T}_{\zeta_{k+1}}(D - L_{k+1})_{I,:}$
  $k = k + 1$
end while
Output: C_k, U_k, R_k: CUR decomposition of L

we leave this as a general parameter and leave the analysis of choosing an adaptive threshold value to future work.

For initialization, we use a modification of Algorithm 3 of (Cai et al., 2019), which uses two steps of Alternating Projections (Netrapalli et al., 2014) to obtain L_0 and S_0.

Algorithm 2 Initialization via CUR Decomposition

Input: D: observed corrupted data matrix; r: target rank; β_1, β_2: thresholding parameters; $|I|, |J|$: number of rows and columns to sample.
Uniformly sample row indices I and column indices J
$S_{-1} = \mathcal{T}_{\beta_1}(D)$
$C = (D - S_{-1})_{:,J}$
$R = (D - S_{-1})_{I,:}$
$U = (D - S_{-1})_{I,J}$
$L_0 = CU_r^{\dagger}R$
$S_0 = \mathcal{T}_{\beta_2}(D - L_0)$
Output: L_0, S_0

We do allow one significant change in our experimental examination of Algorithm 1: when we initialize via Algorithm 2, rather than computing a full SVD of $D - S_{-1}$ in line 4 as in Cai et al. (2019), we instead compute a CUR decomposition of it (lines 4-6). This change reduces the computational cost significantly. As noted above, we sample $\mathcal{O}(r \log n)$ columns and rows of $D - S_{-1}$ in forming the CUR decomposition. Thus, computing the pseudoinverse of the intersection matrix requires only $\mathcal{O}(r^3 \log^2 n)$ flops as opposed to $\mathcal{O}(n^2 r)$ to compute the SVD of $D - S_{-1}$.

Finally, one of the speed advantages of IRCUR is that one need only operate on small submatrices of $D - S_k$ at a time. This is not quite the case for Riemannian CUR, as one cannot form a column submatrix of $P_{T_k}(D - S_k)$ by only using the corresponding columns of $D - S_k$. Nonetheless, we are not required to form all of $P_{T_k}(D - S_k)$ as is done in AccAltProj here. Indeed, at time instance k, we have $D - S_k = C_k U_k^{\dagger} R_k$, so

$$P_{T_k}(D - S_k) = W_k W_k^{\top} C_k U_k^{\dagger} R_k + C_k U_k^{\dagger} R_k V_k V_k^{\top} - W_k W_k^{\top} C_k U_k^{\dagger} R_k V_k V_k^{\top}.$$

Now let $\widetilde{R}_k := R_k V_k V_k^{\top}$ and $\widetilde{C}_k = W_k W_k^{\top} C_k$, and notice that after some combining of terms, we have

$$(P_{T_k}(D - S_k))_{I,:} = (\widetilde{C}_k)_{I,:} U_k^{\dagger} R_k + (C_k - \widetilde{C}_k)_{I,:} U_k^{\dagger} \widetilde{R}_k,$$

and

$$(P_{T_k}(D - S_k))_{:,J} = \widetilde{C}_k U_k^{\dagger} (R_k - \widetilde{R}_k)_{:,J} + C_k U_k^{\dagger} (\widetilde{R}_k)_{:,J}.$$

Thus, the submatrices of $P_{T_k}(D - S_k)$ may be computed by keeping track of C_k, U_k, and R_k from the previous iteration, and of computing only submatrices of $\widetilde{C}_k$ and $\widetilde{R}_k$ defined as above. Thus we never form all of $P_{T_k}(D - S_k)$ at any stage, but only its constituent submatrices according to the index sets I and J.

3 Numerical Experiments

Here we test the validity of the proposed algorithm with respect to other nonconvex Robust PCA solvers. Note that (Cai et al., 2020) demonstrates that both IRCUR and AccAltProj are significantly faster and more accurate than AltProj and the gradient descent method of (Yi et al., 2016). Thus, we only compare our algorithm to IRCUR and AccAltProj.

3.1 Synthetic data

First, Figure 1 illustrates that indeed RieCUR is significantly faster than AccAltProj, while being modestly slower than IRCUR. For each dimension $n \in \{500, 1000, \ldots, 10000\}$, we generate a random Robust PCA problems by first generating a random low-rank matrix L with rank $r = 5$ as the product of two Gaussian random matrices, i.e., $L = AB^\top$ where $A, B \in \mathbb{R}^{n\times 5}$ have i.i.d. $\mathcal{N}(0, 1)$ entries. We generate a sparse random matrix S with $\alpha = 0.3$. We set a convergence threshold of 10^{-6} for all algorithms, and a maximum number of iterations of 40. Figure 1 corresponds to average runtime over 5 random trials for each algorithm for each dimension. In all trials below, we sample $3 \log n$ columns and rows when doing a CUR decomposition in RieCUR or IRCUR.

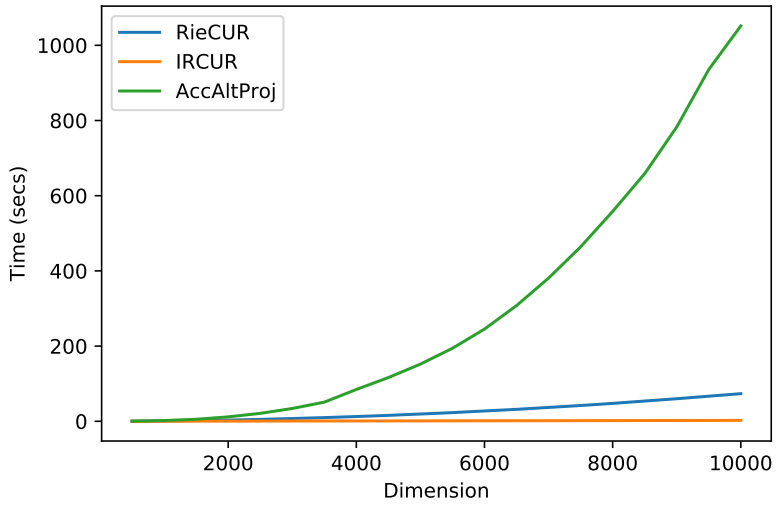

Figure 1: Time vs. Problem dimension (n) for the three algorithms considered. In all trials L has rank 5 and $\alpha = 0.3$. Each algorithm stops once $e_k < 10^{-6}$ or 40 iterations has been reached.

As a second test, we consider the effect of the sparsity of S on the runtime of the algorithms. To do so, we again generate random matrices of size 2000×2000, but allow α to range from 0.5 to 0.95. Each algorithm is allowed to run until either $e_k < 10^{-3}$ or 100 iterations is reached, whichever comes first. The results are shown in Figure 2. One can see that RieCUR, while slower than IRCUR for all sparsity levels, remains significantly faster than AccAltProj.

Next, we consider the effect of the problem dimension n on the final relative error. Figure 3 shows the results for a similar setup to the above, with L being a rank 5, 2500×2500 matrix, and the dimension again goes from 500 to $10,000$. Each algorithm is set to a tolerance of 10^{-6} and maximum iteration of 40. One can see that IRCUR does not achieve as good error in 40 iterations compared to RieCUR which also performs slightly worse than AccAltProj.

Finally, we demonstrate that RieCUR is capable of handling more outliers than IRCUR. To do so, we run all algorithms to 100 iterations and plot the final error versus the sparsity α. We utilize the

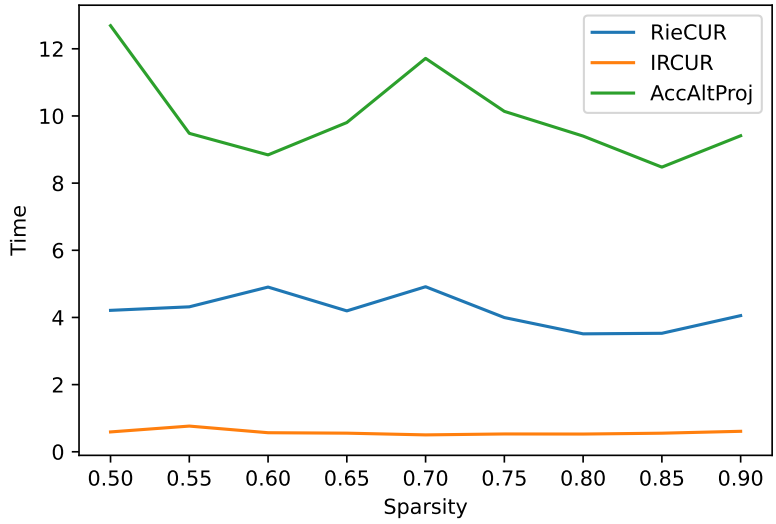

Figure 2: Time vs. Sparsity for the three algorithms considered. In all trials L is 2000×2000 with rank 5. Each algorithm stops once $e_k < 10^{-3}$ or 100 iterations.

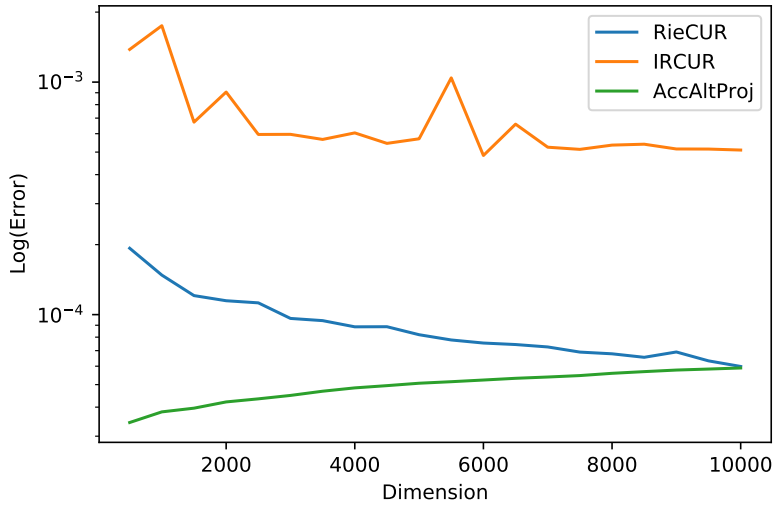

Figure 3: $\log$(Final Relative Error) vs. Problem dimension (n) for the three algorithms. In all trials, L is a 2500×2500 rank 5 matrix. Each algorithm stops once $e_k < 10^{-6}$ or 40 iterations.

same setup as the previous experiment for runtime versus sparsity. Note from Figure 4 that for each sparsity level, AccAltProj attains the lowest error, followed by RieCUR, followed by IRCUR.

3.2 Real data

Here we illustrate the performance of RieCUR on the `restaurant` benchmark dataset. The video is black and white with $3,055$ frames of size 120×160. A single data matrix of the video is obtained as follows: each frame is vectorized to form a column vector of size $19,200$, and columns are concatenated into a data matrix of size $19,200 \times 3,055$.

In this case, Robust PCA will separate the low-rank part of the collection of video frames, which corresponds to the static background of the video from the sparse outlier part corresponding to the foreground of the video (in this case moving people). In Figure 5, we show the effect of RieCUR for

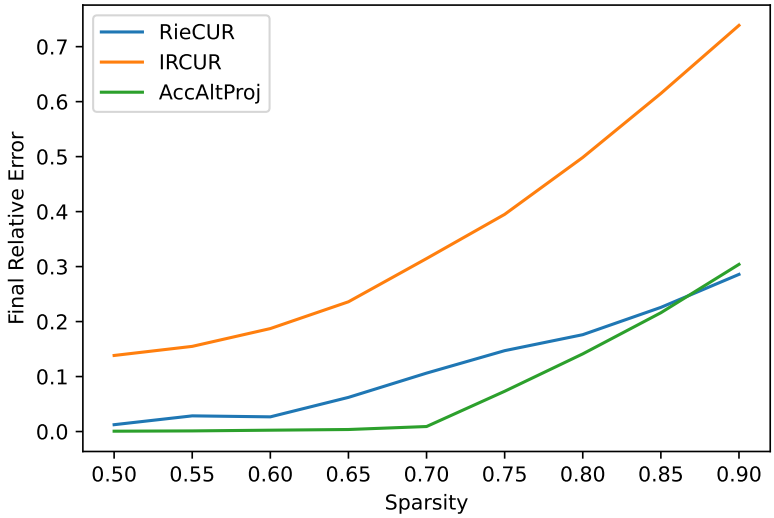

Figure 4: Error vs. Sparsity for the three algorithms considered. In all trials L is 2000×2000 with rank 5. Each algorithm stops after 100 iterations has been reached.

foreground/background separation. One can see that all algorithms tested yield high quality visual separation of the foreground and background of the videos.

Figure 5: (Top row) Original frame from the `restaurant` video. (Center row) Background of the same frame recovered from AccAltProj (left), RieCUR (center), and IRCUR (right). (Bottom row) Foreground of the same frame recovered from AccAltProj (left), RieCUR (center), and IRCUR (right).

4 CONCLUSION

We have proposed Riemannian CUR (RieCUR), a nonconvex, Robust PCA algorithm, which combines ideas from both Accelerated Alternating Projections (AccAltProj) and Iterated Robust CUR (IRCUR). RieCUR has the best of both features of the other two methods, and is state-of-the art in terms of computational complexity, thus providing an alternative method to the others in cases where speed and robustness are both required. The current algorithm carries with it a thresholding decay parameter, but future work will focus on providing an adaptive threshold which can guarantee convergence given initialization within a neighborhood of a local solution. Additionally, we use a fast starting method to initialize the guesses for the low-rank and sparse outlier matrices which uses a CUR decomposition of the corrupted data matrix D rather than taking an SVD of it. This method was shown to still yield good convergence in synthetic trials while being significantly faster than standard SVD initialization.

Future work on RieCUR will consist of proving convergence to a minimizer of the nonconvex Robust PCA problem given good initialization, proof that Algorithm 2 can achieve good initialization for Robust PCA, use of a dynamic threshold rather than a static one in lines 11-13 of Algorithm 1, and more comprehensive experiments on various Robust PCA problems.

ACKNOWLEDGMENTS

H.Q. Cai's work was partially supported by NSF DMS 2208612.

REFERENCES

Thierry Bouwmans, Sajid Javed, Hongyang Zhang, Zhouchen Lin, and Ricardo Otazo. On the applications of robust PCA in image and video processing. *Proceedings of the IEEE*, 106(8): 1427–1457, 2018.

HanQin Cai, Jian-Feng Cai, and Ke Wei. Accelerated alternating projections for robust principal component analysis. *The Journal of Machine Learning Research*, 20(1):685–717, 2019.

HanQin Cai, Keaton Hamm, Longxiu Huang, Jiaqi Li, and Tao Wang. Rapid robust principal component analysis: CUR accelerated inexact low rank estimation. *IEEE Signal Processing Letters*, 28:116–120, 2020.

HanQin Cai, Jian-Feng Cai, Tianming Wang, and Guojian Yin. Accelerated structured alternating projections for robust spectrally sparse signal recovery. *IEEE Transactions on Signal Processing*, 69:809–821, 2021a.

HanQin Cai, Zehan Chao, Longxiu Huang, and Deanna Needell. Fast robust tensor principal component analysis via fiber CUR decomposition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 189–197, 2021b.

HanQin Cai, Keaton Hamm, Longxiu Huang, and Deanna Needell. Robust CUR decomposition: Theory and imaging applications. *SIAM Journal on Imaging Sciences*, 14(4):1472–1503, 2021c.

HanQin Cai, Jialin Liu, and Wotao Yin. Learned robust PCA: A scalable deep unfolding approach for high-dimensional outlier detection. In *Advances in Neural Information Processing Systems*, volume 34, pp. 16977–16989, 2021d.

HanQin Cai, Jian-Feng Cai, and Juntao You. Structured gradient descent for fast robust low-rank Hankel matrix completion. *arXiv preprint arXiv:2204.03316*, 2022a.

HanQin Cai, Longxiu Huang, Pengyu Li, and Deanna Needell. Matrix completion with cross-concentrated sampling: Bridging uniform sampling and cur sampling. *arXiv preprint arXiv:2208.09723*, 2022b.

Emmanuel J Candès, Xiaodong Li, Yi Ma, and John Wright. Robust principal component analysis? *Journal of the ACM (JACM)*, 58(3):1–37, 2011.

Yudong Chen, Sujay Sanghavi, and Huan Xu. Clustering sparse graphs. In *Advances in Neural Information Processing Systems*, pp. 2204–2212, 2012.

Jiawei Chiu and Laurent Demanet. Sublinear randomized algorithms for skeleton decompositions. *SIAM Journal on Matrix Analysis and Applications*, 34(3):1361–1383, 2013.

Petros Drineas, Ravi Kannan, and Michael W Mahoney. Fast Monte Carlo algorithms for matrices III: Computing a compressed approximate matrix decomposition. *SIAM Journal on Computing*, 36(1):184–206, 2006.

Sergei A Goreinov, Eugene E Tyrtyshnikov, and Nickolai L Zamarashkin. A theory of pseudoskeleton approximations. *Linear Algebra and its Applications*, 261(1-3):1–21, 1997.

Keaton Hamm and Longxiu Huang. Perspectives on CUR decompositions. *Applied and Computational Harmonic Analysis*, 48(3):1088–1099, 2020.

Po-Sen Huang, Scott Deeann Chen, Paris Smaragdis, and Mark Hasegawa-Johnson. Singing-voice separation from monaural recordings using robust principal component analysis. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 57–60, 2012.

Won-Dong Jang, Chulwoo Lee, and Chang-Su Kim. Primary object segmentation in videos via alternate convex optimization of foreground and background distributions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 696–704, 2016.

Michael W Mahoney and Petros Drineas. CUR matrix decompositions for improved data analysis. *Proceedings of the National Academy of Sciences*, 106(3):697–702, 2009.

Praneeth Netrapalli, UN Niranjan, Sujay Sanghavi, Animashree Anandkumar, and Prateek Jain. Non-convex robust PCA. In *Advances in Neural Information Processing Systems*, pp. 1107–1115, 2014.

Yigang Peng, Arvind Ganesh, John Wright, Wenli Xu, and Yi Ma. RASL: Robust alignment by sparse and low-rank decomposition for linearly correlated images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2233–2246, 2012.

Roger Penrose. On best approximate solutions of linear matrix equations. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 52, pp. 17–19. Cambridge University Press, 1956.

Gilbert Strang and Cleve Moler. LU and CR elimination. *SIAM Review*, 64(1):181–190, 2022.

Bart Vandereycken. Low-rank matrix completion by riemannian optimization. *SIAM Journal on Optimization*, 23(2):1214–1236, 2013.

Ke Wei, Jian-Feng Cai, Tony F Chan, and Shingyu Leung. Guarantees of Riemannian optimization for low rank matrix completion. *Inverse Problems & Imaging*, 14(2):233, 2020.

John Wright, Allen Y Yang, Arvind Ganesh, S Shankar Sastry, and Yi Ma. Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31 (2):210–227, 2008.

John Wright, Arvind Ganesh, Shankar Rao, Yigang Peng, and Yi Ma. Robust principal component analysis: Exact recovery of corrupted low-rank matrices via convex optimization. In *Advances in Neural Information Processing Systems*, pp. 2080–2088, 2009.

Xinyang Yi, Dohyung Park, Yudong Chen, and Constantine Caramanis. Fast algorithms for robust PCA via gradient descent. In *Advances in Neural Information Processing Systems*, pp. 4152–4160, 2016.