GLOBALLY CONVERGENT ACCELERATED PROXIMAL ALTERNATING MAXIMIZATION METHOD FOR L1-PRINCIPAL COMPONENT ANALYSIS

Peng Wang *Huikang Liu* *Anthony Man-Cho So*

Department of Systems Engineering and Engineering Management, CUHK, Hong Kong

ABSTRACT

In this paper, we consider a ℓ_1-PCA problem under the large-scale data sample scenario, which has extensive applications in science and engineering. Previous algorithms for the problem either are not scalable or do not have good convergence guarantees. Our contribution is threefold. First, we develop a novel accelerated version of the proximal alternating maximization method to solve the ℓ_1-PCA problem. Second, by exploiting the Kurdyka-Łojasiewicz property of the problem, we show that our proposed method enjoys global convergence to a critical point, which improves upon existing convergence guarantees of other first-order methods for the ℓ_1-PCA problem. Third, we demonstrate via numerical experiments on both real-world and synthetic datasets that our proposed method is scalable and more efficient and accurate than other methods in the literature.

Index Terms— ℓ_1-PCA, extrapolation, accelerated proximal alternating maximization, Kurdyka-Łojasiewicz inequality, global convergence

1. INTRODUCTION

Principal Component Analysis (PCA) is a fundamental data analytic tool that has found many applications in various areas of science and engineering [8]. Roughly speaking, PCA aims to find a non-trivial lower-dimensional subspace that can explain most of the variance in the data, and a common formulation is given by

$$\max_{\boldsymbol{B}\in\mathbb{R}^{D\times K}} \|\boldsymbol{X}^T\boldsymbol{B}\|_F^2 \quad \text{s.t.} \quad \boldsymbol{B}^T\boldsymbol{B}=\boldsymbol{I}, \tag{1.1}$$

where $\boldsymbol{X}\in\mathbb{R}^{D\times N}$ is the data matrix consisting of N data samples $\boldsymbol{x}_1,\ldots,\boldsymbol{x}_N\in\mathbb{R}^D$ and $\|\boldsymbol{A}\|_F=\left(\sum_{i,j}|A_{i,j}|^2\right)^{1/2}$ denotes the Frobenius norm of the matrix $\boldsymbol{A}$. Problem (1.1), which we shall refer to as the ℓ_2-*PCA*, can be solved by performing a singular value decomposition (SVD) of the data matrix $\boldsymbol{X}$ [7]. Unfortunately, ℓ_2-PCA is sensitive to the outliers in the data and can lead to a poor performance in the presence of outliers. In practice, the outliers can arise as a result of transmission error, bursty-noise effect, etc. This motivates the development of variants of PCA that are robust against outliers. One such variant is the ℓ_1-*PCA*, which was proposed and studied in, e.g., [3, 6, 20, 9]. Instead of maximizing the Frobenius norm of $\boldsymbol{X}^T\boldsymbol{B}$, ℓ_1-PCA seeks to maximize its ℓ_1-norm; i.e.,

$$\max_{\boldsymbol{B}\in\mathbb{R}^{D\times K}} \|\boldsymbol{X}^T\boldsymbol{B}\|_1 \quad \text{s.t.} \quad \boldsymbol{B}^T\boldsymbol{B}=\boldsymbol{I}, \tag{1.2}$$

where $\|\boldsymbol{A}\|_1=\sum_{i,j}|A_{ij}|$ denotes the ℓ_1-norm of the matrix $\boldsymbol{A}$. Unlike the ℓ_2-PCA problem (1.1), which admits an essentially closed-form solution that can be computed in polynomial time, the ℓ_1-PCA problem (1.2) is nonsmooth nonconvex and is NP-hard in general [15]. As such, various heuristics have been proposed to tackle Problem (1.2). Kwak [10] presented a first-order method with fixed-point iterations to solve (1.2) when $K=1$. Subsequently, Nie *et al.* [19] extended the above first-order method to handle the case where $K\geq 2$. In both works, the authors showed that the iterates generated by their respective algorithms have a convergent subsequence (i.e., the so-called *subsequential convergence* property), and that any limit point of the iterates is a critical point of Problem (1.2). Recently, Markopoulos *et al.* [15] proposed an exact algorithm for solving (1.2) with complexity $\mathcal{O}(N^{DK-K+1})$. Note that the algorithm has polynomial-time complexity when D and K are fixed. However, in practice, it can only handle a small number of samples (i.e., N is small) and low-dimensional data (i.e., $D\ll N$). Later, Markopoulos *et al.* [16] proposed a sub-optimal algorithm based on the notion of bit-flipping for tackling Problem (1.2). They showed that their proposed algorithm also enjoys subsequential convergence, and that the set of limit points is a subset of those of the algorithms in [10, 19]. It is worth noting that subsequential convergence is a strictly weaker property than convergence—the latter refers to the property that the sequence of iterates converge to a *single* point, while the former allows for the possibility that the sequence has multiple subsequences that converge to *different* points, which makes it numerically difficult to identify which point is a limit point of the algorithm.

In practice, lightweight methods such as first-order methods are often preferred when solving the ℓ_1-PCA problem (1.2), as the number of data samples N and the data dimension D are often large. In view of the above discussion, we are thus motivated to develop fast first-order methods for tackling (1.2) with strong convergence guarantees. In

this work, we propose a novel accelerated version of the proximal alternating maximization (PAM) method in [1] for solving Problem (1.2). Our method, which we call accelerated PAM (APAM), performs a linear extrapolation on one block-variable of the ℓ_1-PCA formulation, which can not only accelerate the original PAM method but also return solution with better quality than existing methods empirically. It is worth mentioning that such acceleration technique is mainly developed for methods that solve convex optimization problems in the literature, such as heavy-ball method [21, 23], Nesterov's accelerated gradient descent [17, 18] and FISTA [4]. Recently, there have been some works that aim to extend this technique to methods that solve nonconvex optimization problems. In particular, Li and Lin [11] proposed monotone and nonmonotone accelerated proximal gradient methods to solve general nonsmooth nonconvex problems, while Li *et al.* [12] analyzed the convergence of the accelerated proximal gradient method with momentum for nonconvex problems. However, since the proximal operator associated with the ℓ_1-PCA problem (1.2) is not known to be efficiently computable, the methods in [11, 12] are ineffective for solving Problem (1.2). By contrast, our proposed (A)PAM method can take advantage of the structure of the ℓ_1-PCA problem (1.2), thus resulting in extremely efficient computations. Moreover, by utilizing the so-called *Kurdyka-Łojasiewicz* (KŁ) property (see [1, 2]) of a carefully constructed Lyapunov function associated with Problem (1.2), we are able to show that the iterates generated by the (A)PAM method converge globally to a single critical point of (1.2). To the best of our knowledge, this is the first global convergence result for an accelerated version of the PAM method and improves upon the subsequential convergence results in [10, 19, 16] for the ℓ_1-PCA problem (1.2).

Besides the notations introduced earlier, we use $-\mathbf{1} \leq \boldsymbol{A} \leq \mathbf{1}$ to denote $-1 \leq A_{ij} \leq 1,\ \forall\ i, j$ and $\delta_{\mathcal{A}}$ to denote the indicator function of the set $\mathcal{A}$; i.e., $\delta_{\mathcal{A}}(A)$ is 0 if $A \in \mathcal{A}$ and $+\infty$ otherwise. Furthermore, the subdifferential of a proper lower semicontinuous function $f : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ is denoted by ∂f; see, e.g., [22, Chapter 8].

2. ACCELERATED PAM FOR ℓ_1-PCA

Our proposed (A)PAM method for the ℓ_1-PCA problem is based on the following two-block reformulation of Problem (1.2):

$$\max_{\boldsymbol{B}^T\boldsymbol{B}=\boldsymbol{I}} \|\boldsymbol{X}^T\boldsymbol{B}\|_1 = \max_{\boldsymbol{A}\in\mathcal{A},\,\boldsymbol{B}\in\mathcal{B}} H(\boldsymbol{A},\boldsymbol{B}), \tag{2.1}$$

where $H(\boldsymbol{A},\boldsymbol{B}) := \mathrm{tr}(\boldsymbol{A}^T\boldsymbol{X}^T\boldsymbol{B})$, $\mathcal{A} := \{\boldsymbol{A} \in \mathbb{R}^{N\times K} : -\mathbf{1} \leq \boldsymbol{A} \leq \mathbf{1}\}$, and $\mathcal{B} := \{\boldsymbol{B} \in \mathbb{R}^{D\times K} : \boldsymbol{B}^T\boldsymbol{B} = \boldsymbol{I}\}$. Specifically, given an initial point $(\boldsymbol{A}^0, \boldsymbol{B}^0)$, our (A)PAM method generates the sequence $\{(\boldsymbol{A}^k, \boldsymbol{B}^k)\}$, $k = 0, 1, 2, \ldots$, via the scheme

$$\boldsymbol{A}^{k+1} = \arg\max_{\boldsymbol{A}\in\mathcal{A}} \left\{ H(\boldsymbol{A}, \boldsymbol{Y}^k) - \frac{1}{2\alpha}\|\boldsymbol{A} - \boldsymbol{A}^k\|_F^2 \right\}, \tag{2.2}$$

$$\boldsymbol{B}^{k+1} = \arg\max_{\boldsymbol{B}\in\mathcal{B}} \left\{ H(\boldsymbol{A}^{k+1}, \boldsymbol{B}) - \frac{1}{2\beta}\|\boldsymbol{B} - \boldsymbol{B}^k\|_F^2 \right\}, \tag{2.3}$$

where $\boldsymbol{Y}^k = \boldsymbol{B}^k + \theta(\boldsymbol{B}^k - \boldsymbol{B}^{k-1})$ and $\theta \in [0, 1]$ when $k \geq 1$ and $\boldsymbol{Y}^0 = \boldsymbol{B}^0$. The novelty here is that we perform an extrapolation step for the variable $\boldsymbol{B}$ by adding a momentum term that involves the previous and current iterates. When $\theta = 0$, we recover the PAM updates in [1]. The upshot of the updates (2.2) (2.3) is that they both have closed-form solutions. Indeed, it can be easily verified that

$$\boldsymbol{A}^{k+1} = \Pi_{\mathcal{A}}\left[\boldsymbol{A}^k + \alpha\boldsymbol{X}^T\boldsymbol{Y}^k\right], \tag{2.4}$$

where $\Pi_{\mathcal{A}}$ is the projector onto the box $\mathcal{A}$; i.e.,

$$\Pi_{\mathcal{A}}(\boldsymbol{D}) = \begin{cases} -1 & \text{if } D_{ij} < -1, \\ D_{ij} & \text{if } -1 \leq D_{ij} \leq 1, \\ 1 & \text{if } D_{ij} > 1. \end{cases}$$

On the other hand, we have $\boldsymbol{B}^{k+1} = \boldsymbol{U}^{k+1}{\boldsymbol{V}^{k+1}}^T$, where

$$\boldsymbol{B}^k + \beta\boldsymbol{X}\boldsymbol{A}^{k+1} = \boldsymbol{U}^{k+1}\boldsymbol{\Lambda}^{k+1}{\boldsymbol{V}^{k+1}}^T \tag{2.5}$$

is a compact SVD of $\boldsymbol{B}^k + \beta\boldsymbol{X}\boldsymbol{A}^{k+1}$. Note that since $\boldsymbol{B}^k + \beta\boldsymbol{X}\boldsymbol{A}^{k+1} \in \mathbb{R}^{D\times K}$, the complexity of computing the compact SVD to solve (2.3) is $\mathcal{O}(D^2K)$, which is very cheap under the large-scale setting $D \ll N$. The total complexity of each iteration is $\mathcal{O}(NDK)$. We now summarize the APAM method for solving the ℓ_1-PCA problem (1.2) in Algorithm 1.

Algorithm 1 Accelerated Proximal Alternating Maximization Method (APAM) for ℓ_1-PCA

1: **Input:** Step sizes $\alpha > 0$, $\beta > 0$; extrapolation parameter $\theta \in [0, 1]$; initial points $\boldsymbol{A}^0 \in \mathcal{A}$, $\boldsymbol{Y}^0 = \boldsymbol{B}^0 \in \mathcal{B}$
2: **for** $k = 0, 1, 2, \ldots$ **do**
3: $\quad\boldsymbol{A}^{k+1} = \Pi_{\mathcal{A}}\left[\boldsymbol{A}^k + \alpha\boldsymbol{X}^T\boldsymbol{Y}^k\right]$
4: $\quad\boldsymbol{B}^{k+1} = \boldsymbol{U}^{k+1}{\boldsymbol{V}^{k+1}}^T$, where $(\boldsymbol{U}^{k+1}, \boldsymbol{V}^{k+1})$ is given by (2.5)
5: $\quad\boldsymbol{Y}^{k+1} = \boldsymbol{B}^{k+1} + \theta(\boldsymbol{B}^{k+1} - \boldsymbol{B}^k)$
6: **end for**

3. CONVERGENCE ANALYSIS OF APAM

To facilitate our convergence analysis of the APAM method, let us rewrite (2.1) as the following unconstrained minimization problem:

$$\min_{\boldsymbol{A}\in\mathbb{R}^{N\times K},\,\boldsymbol{B}\in\mathbb{R}^{D\times K}} \Psi(\boldsymbol{A},\boldsymbol{B}), \tag{3.1}$$

where $\Psi(\boldsymbol{A},\boldsymbol{B}) := -H(\boldsymbol{A},\boldsymbol{B}) + \delta_{\mathcal{A}}(\boldsymbol{A}) + \delta_{\mathcal{B}}(\boldsymbol{B})$. The upshot of the form (3.1) is that it allows us to apply the powerful machinery in [1, 2] (particularly [2, Theorem 2.9]) to analyze the convergence behavior of the APAM method. To begin, let

us construct the following Lyapunov function associated with Problem (3.1):

$$\Theta(\boldsymbol{A},\boldsymbol{B},\boldsymbol{B}') = \Psi(\boldsymbol{A},\boldsymbol{B}) + \frac{\gamma}{2}\|\boldsymbol{B}-\boldsymbol{B}'\|_F^2,\ \gamma \in [L\theta, \frac{1}{\beta}], \tag{3.2}$$

where $L = \|\boldsymbol{X}\|$ is the spectral norm of data matrix $\boldsymbol{X}$. For simplicity, let us write $\boldsymbol{C}^k := (\boldsymbol{A}^k, \boldsymbol{B}^k, \boldsymbol{B}^{k-1}),\ \forall k = 1,2,\ldots$. We then have the following preparatory results:

Lemma 3.1 (Sufficient Decrease of Θ)**.** *Let $\{\boldsymbol{C}^k\}_{k\geq 0}$ be the sequence generate by Algorithm 1, where the step size $\alpha < \frac{1}{L\theta}$ is a constant and the penalty parameter γ satisfies $\gamma \in [L\theta, \frac{1}{\beta}]$. Then, the sequence $\{\Theta(\boldsymbol{C}^k)\}_{k\geq 0}$ satisfies*

$$\Theta(\boldsymbol{C}^{k+1}) - \Theta(\boldsymbol{C}^k) \leq -\rho_1\|\boldsymbol{C}^{k+1}-\boldsymbol{C}^k\|_F^2$$

for some constant $\rho_1 > 0$.

Lemma 3.2 (Safeguard)**.** *The sequence $\{\boldsymbol{C}^k\}_{k\geq 0}$ is bounded and $\boldsymbol{W}^{k+1} \in \partial\Theta(\boldsymbol{C}^{k+1})$, where*

$$\boldsymbol{W}^{k+1} := \left(-\boldsymbol{X}^T\boldsymbol{\Delta}^{k+1} + \theta\boldsymbol{X}^T\boldsymbol{\Delta}^k + \frac{1}{\alpha}(\boldsymbol{A}^k - \boldsymbol{A}^{k+1}),\right.$$
$$\left.\left(\gamma - \frac{1}{\beta}\right)\Delta^{k+1}, -\gamma\boldsymbol{\Delta}^{k+1}\right)$$

and $\boldsymbol{\Delta}^k := \boldsymbol{B}^k - \boldsymbol{B}^{k-1}$. Furthermore, there exists a constant $\rho_2 > 0$ such that

$$\|\boldsymbol{W}^{k+1}\|_F \leq \rho_2\|\boldsymbol{C}^{k+1}-\boldsymbol{C}^k\|_F.$$

Due to space limitation, we defer the proofs of Lemmas 3.1 and 3.2 to the full version of this paper.

Lemma 3.3 (Relationship between Ψ and Θ)**.** *Let $\boldsymbol{A} \in \mathcal{A}$ and $\boldsymbol{B} \in \mathcal{B}$. Then,*

$$\mathbf{0} \in \partial\Psi(\boldsymbol{A},\boldsymbol{B}) \Leftrightarrow \mathbf{0} \in \partial\Theta(\boldsymbol{A},\boldsymbol{B},\boldsymbol{B}).$$

Lemma 3.3 can be easily verified using the properties of the subdifferential; see, e.g., [22, Chapter 8].

Lemma 3.4 (KŁ Property of the Lyapunov Function)**.** *The Lyapunov function Θ has the KŁ property (see [2, Definition 2.4] for the definition).*

Lemma 3.4 follows from the fact that $\Theta(\boldsymbol{A},\boldsymbol{B},\boldsymbol{B}')$ consists of polynomials (i.e., $-H(\boldsymbol{A},\boldsymbol{B}) + \frac{\gamma}{2}\|\boldsymbol{B}-\boldsymbol{B}'\|_F^2$) and indicator functions of the polyhedral set $\mathcal{A}$ and the smooth manifold $\mathcal{B}$; see, e.g., [1, 2].

Armed with the above lemmas and invoking [2, Theorem 2.9], we have the following global convergence result for the APAM method.

Theorem 3.5. *For any initialization $\boldsymbol{A}^0 \in \mathcal{A},\ \boldsymbol{B}^0 \in \mathcal{B}$, the sequence generated by the APAM method (Algorithm 1) will converge to a critical point of the ℓ_1-PCA problem* (1.2)*.*

Theorem 3.5 establishes, for the first time, the global convergence of an accelerated version of the PAM method in [1]. This is achieved by a novel construction of the Lyapunov function (3.2) associated with Problem (3.1).

4. NUMERICAL RESULTS

In this section, we demonstrate the superiority of the APAM method[1] for solving the ℓ_1-PCA problem (1.2) via numerical experiments. Our codes are implemented in MATLAB R2018a. The tests are conducted on a standard computing server with 1000GB memory and 4 Intel(R) Xeon(R) CPUs each consisting 10 cores with 2.40GHz.

4.1. Performance of Different First-Order Methods

In this subsection, we compare our APAM method with the PAM method in [1] and the non-greedy first-order fixed-point method (NFFM) in [19]. We utilize the real-world datasets *epsilon, mnist* downloaded from LIBSVM [5].[2] We employ the following stopping criterion: At each iteration k, we compute the function value f^k and terminate the algorithm if there are 10 consecutive iterations such that

$$|f^{k+m} - f^{k+m-1}| \leq \epsilon,\ m = 1,\ldots,10,$$

where we choose $\epsilon = 10^{-7}$.

We implement two sets of experiments on different subsets of the datasets *epsilon, mnist* to compare the runtimes and objective values of APAM, PAM, and NFFM, where *epsilon* is a size 100000×500 subset of the original one. Specifically, we set the size of subsets of the full data as $10\%, 20\%, \ldots, 100\%$ in each test. The step sizes for APAM and PAM are both set to $\alpha = 10^6$, $\beta = 1$. To avoid the influence of the starting points and other random factors, we run each algorithm 10 times from different starting points. To be consistent, different algorithms have the same starting point in each test. Finally, for each algorithm in a test, we select the maximum among the 10 obtained objective values as the recorded value. Furthermore, we average the runtimes of the 10 runs as the recorded time. To compare the suboptimal value of each algorithm, we will use the value of APAM as the baseline, and compute the ratio of improvement, *i.e.*, others' recorded value minus APAM's divided by APAM's.

As seen from Fig. 1, APAM is the most efficient and accurate algorithm when compared to other two first-order algorithms, as it consumes much less time and converges to a better solution than PAM and NFFM.

4.2. Computation Accuracy of Different Methods

In this subsection, we investigate how far the objective values returned by APAM, PAM, NFFM, and the bit-flipping method (BFM) in [16] are from the global optimal value of the ℓ_1-PCA problem (1.2). Note that BFM is a high-order method. To do so, we make use of the exact algorithm proposed in [15] to check the optimality of the aforementioned methods. It should be noted that the algorithm in [15] has a polynomial

[1] We set $\theta = 1$ unless specified otherwise.

[2] https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/

complexity of $\mathcal{O}(N^D)$ when D is fixed and $D \ll N$. We set the step sizes for APAM and PAM to $\alpha = 10$, $\beta = 10$. Here, we define the relative gap between an obtained objective value and the global optimal value as

$$\Delta(\boldsymbol{B}, \boldsymbol{B}^\star) = \frac{\|\boldsymbol{X}^T\boldsymbol{B}^\star\|_1 - \|\boldsymbol{X}^T\boldsymbol{B}\|_1}{\|\boldsymbol{X}^T\boldsymbol{B}^\star\|_1}, \qquad (4.1)$$

where $\boldsymbol{B}^\star$ is an optimal solution to Problem (1.2) and $\boldsymbol{B}$ is a solution obtained by any of the aforementioned methods. The quantity (4.1) is called the *performance degradation ratio* in [16]. We generate 1000 data matrices $\{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_{1000}\}$ randomly with $N = 20$, $D = 3$, and $K = 2$. Based on 1000 tests, we calculate the empirical cumulative distribution functions (CDFs) of the performance degradation ratios of the algorithms and plot them in Fig. 2.

According to the above experiments, when the number of initialization is 5 or 15, BFM can return the optimal solution 92% and 97% of the time, respectively, while APAM can return the optimal solution about 84% and 96%, respectively. These numbers are much better than PAM and NFFM. Since APAM is a first-order method and BFM is a local exhaustive search procedure, it is normal that BFM has a better performance than APAM in terms of performance degradation ratio.

4.3. Convergence Performance of APAM with Different Extrapolation Step Size

In this subsection, we demonstrate empirically that APAM is superior to PAM in both runtime and convergence performance. Towards that end, we run APAM with $\theta = 0.25, 0.5, 0.75, 1$ and PAM (which corresponds to APAM with $\theta = 0$) on different subsets of the real-world dataset *protein* downloaded from LIBSVM under the same setting as that in Section 4.1. We then compare the runtimes and function values of these five settings of θ in APAM. As is obvious from Fig. 3, the extrapolation step greatly accelerates the algorithm, with $\theta = 1$ being the most efficient and accurate setting among the five.

5. CONCLUSIONS

In this paper, we proposed an efficient and high-accuracy first-order method called APAM to solve the ℓ_1-PCA problem and established its global convergence to a critical point by exploiting the KŁ-property of the problem. Moreover, our experimental results showed that the APAM method has great potential when compared with existing algorithms in terms of both computational efficiency and accuracy. An interesting future direction would be to determine the convergence *rate* of our proposed APAM method by further exploiting the KŁ property; see [11, 12, 14, 13] for some related works.

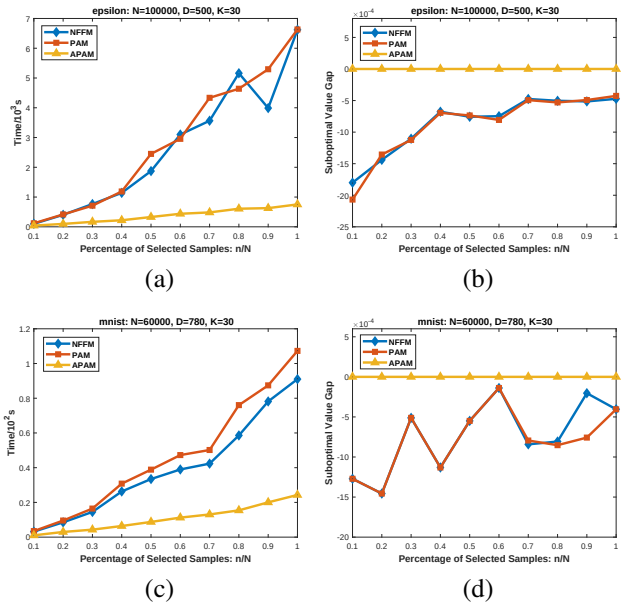

Fig. 1. Different first-order methods for ℓ_1-PCA with different data samples: (a) Time comparison on dataset *epsilon*; (b) function value comparison on dataset *epsilon*; (c) time comparison on dataset *mnist*; (d) function value comparison on dataset *mnist*.

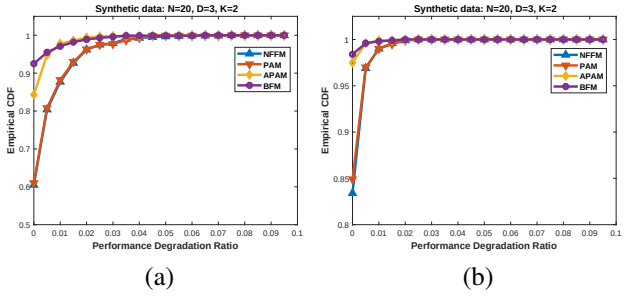

Fig. 2. Performance degradation ratio of four suboptimal algorithms for ℓ_1-PCA: In (a), the number of initialization is 5; in (b), the number of initialization is 15.

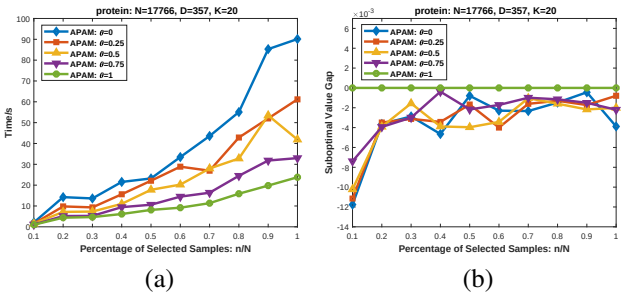

Fig. 3. Performance of different APAM with different extrapolation step sizes θ.

6. REFERENCES

[1] H. Attouch, J. Bolte, P. Redont, and A. Soubeyran. Proximal alternating minimization and projection methods for nonconvex problems: An approach based on the Kurdyka-Łojasiewicz inequality. *Mathematics of Operations Research*, 35(2):438–457, 2010.

[2] H. Attouch, J. Bolte, and B. F. Svaiter. Convergence of descent methods for semi-algebraic and tame problems: proximal algorithms, forward-backward splitting, and regularized Gauss-Seidel methods. *Mathematical Programming*, 137(1-2):91–129, 2013.

[3] A. Baccini, P. Besse, and A. Falguerolles. A L_1-norm PCA and a heuristic approach. *Ordinal and Symbolic Data Analysis*, 1(1):359–368, 1996.

[4] 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.

[5] C.-C. Chang and C.-J. Lin. LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.

[6] J. S. Galpin and D. M. Hawkins. Methods of L_1 estimation of a covariance matrix. *Computational Statistics & Data Analysis*, 5(4):305–319, 1987.

[7] G. H. Golub and C. F. Van Loan. *Matrix Computations*, volume 3. JHU Press, 2012.

[8] I. T. Jolliffe and J. Cadima. Principal component analysis: a review and recent developments. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 374(2065):20150202, 2016.

[9] Q. Ke and T. Kanade. Robust L_1 norm factorization in the presence of outliers and missing data by alternative convex programming. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 739–746. IEEE, 2005.

[10] N. Kwak. Principal component analysis based on L1-norm maximization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(9):1672–1680, 2008.

[11] H. Li and Z. Lin. Accelerated proximal gradient methods for nonconvex programming. In *Advances in Neural Information Processing Systems*, pages 379–387, 2015.

[12] Q. Li, Y. Zhou, Y. Liang, and P. K. Varshney. Convergence analysis of proximal gradient with momentum for nonconvex optimization. In *International Conference on Machine Learning*, pages 2111–2119, 2017.

[13] X. Li, Z. Zhu, A. M.-C. So, and R. Vidal. Nonconvex robust low-rank matrix recovery. Preprint, available at `https://arxiv.org/abs/1809.09237`, 2018.

[14] H. Liu, A. M.-C. So, and W. Wu. Quadratic optimization with orthogonality constraint: Explicit Łojasiewicz exponent and linear convergence of retraction-based line-search and stochastic variance-reduced gradient methods. Accepted for publication in *Mathematical Programming, Series A*, 2018.

[15] P. P. Markopoulos, G. N. Karystinos, and D. A. Pados. Optimal algorithms for L_1-subspace signal processing. *IEEE Transactions on Signal Processing*, 62(19):5046–5058, 2014.

[16] P. P. Markopoulos, S. Kundu, S. Chamadia, and D. A. Pados. Efficient L1-norm principal-component analysis via bit flipping. *IEEE Transactions on Signal Processing*, 65(16):4252–4264, 2017.

[17] Y. Nesterov. A method of solving a convex programming problem with convergence rate $O(1/k^2)$. In *Soviet Mathematics Doklady*, volume 27, pages 372–376, 1983.

[18] Y. Nesterov. Gradient methods for minimizing composite functions. *Mathematical Programming*, 140(1):125–161, 2013.

[19] F. Nie, H. Huang, C. Ding, D. Luo, and H. Wang. Robust principal component analysis with non-greedy ℓ_1-norm maximization. In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, volume 22, page 1433, 2011.

[20] M. Partridge and M. Jabri. Robust principal component analysis. In *Neural Networks for Signal Processing X, 2000. Proceedings of the 2000 IEEE Signal Processing Society Workshop*, volume 1, pages 289–298. IEEE, 2000.

[21] B. T. Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17, 1964.

[22] R. T. Rockafellar and R. J.-B. Wets. *Variational Analysis*, volume 317 of *Grundlehren der mathematischen Wissenschaften*. Springer–Verlag, Berlin Heidelberg, second edition, 2004.

[23] Y. Xu and W. Yin. A block coordinate descent method for regularized multiconvex optimization with applications to nonnegative tensor factorization and completion. *SIAM Journal on imaging sciences*, 6(3):1758–1789, 2013.