Caliphate Journal of Science & Technology (CaJoST)

ISSN: 2705-313X (**PRINT**); 2705-3121 (**ONLINE**) **Research Article**

Open Access Journal available at: https://cajostssu.com and https://www.ajol.info/index.php/cajost/index

DOI: https://dx.doi.org/10.4314/cajost.v4i1.13

Article Info

Received: 9th July 2021

Revised: 18th November 2021

Accepted: 23rd November 2021

[1]Department of Mathematics Education, School of Secondary Education Sciences, Isa Kaita College of Education, P.M.B. 5007, Dutsin-Ma, Katsina State
[2]Department of Mathematics, Sokoto State University Sokoto, P.M.B, 2134 Sokoto State
[3]Department of Statistics, Abubakar Tatari Ali Polytechnic, Bauch State

***Corresponding author's email:**

anisultan11@gmail.com

Cite this: *CaJoST*, 2022, 1, 101-108

Online Stochastic Principal Component Analysis

Nuraddeen Y. Adamu[1]*, [2]Samaila Abdullahi[2] and Sani Musa[3]

This paper studied Principal Component Analysis (PCA) in an online. The problem is posed as a subspace optimization problem and solved using gradient based algorithms. One such algorithm is the Variance-Reduced PCA (VR-PCA). The VR-PCA was designed as an improvement to the classical online PCA algorithm known as the Oja's method where it only handled one sample at a time. The paper developed Block VR-PCA as an improved version of VR-PCA. Unlike prominent VR-PCA, the Block VR-PCA was designed to handle more than one dimension in subspace optimization at a time and it showed good performance. The Block VR-PCA and Block Oja method were compared experimentally in MATLAB using synthetic and real data sets, their convergence results showed Block VR-PCA method appeared to achieve a minimum steady state error than Block Oja method.

Keywords: Online Stochastic, Principal Component Analysis, Block Variance-Reduced, Block Oja.

1. Introduction

Principal component analysis (PCA) can be described as an efficient technique as well as a strategic tool for data analysis and dimensionality reduction. PCA mainly used in determining a subspace that explains most of the variance of the data. By projecting data into this subspace, one can operate a dimensionality reduction procedure and extract the structure of the data. This is classically achieved by diagonalizing the covariance matrix of the data, as its eigenvectors associated with the largest eigenvalues correspond to the principal axes defining this subspace [1]. It is a fundamental tool in data analysis and visualization, designed to find the subspace of largest variance in a given dataset (a set of points in Euclidean space)[2]. A good hope behind employing this method is that the variance along a small number of principal components (i.e. less than the number of measurement types) provides a reasonable feature of the complete dataset [3].

PCA have long been an excellent model for capturing intrinsic low-dimensional structures in large data sets. It has been successfully applied to many signal processing applications including medical, imaging, communications, source localization and clutter tracking in radar and sonar, computer vision for object tracking, system identification, traffic data analysis, and speech recognition, to name just a few. The calculated principal components and best-fit subspaces to a data set not only allow dimensionality reduction but also provide intermediate means for signal estimation, noise removal, and anomaly detection [4].

2. Problem Formulation

Let us focus on stochastic setting by considering the basic matrix optimization problem, given: $X \in \mathbb{R}^{d\times n}$, here we wish to find top k (number of principal subspace) left singular vectors ($k \ll d \text{ and } k > 1$) by solving the following optimization problem:

$$\max_{W\in R^{d\times k}: W^T W = I} \frac{1}{n}\|X^T W\|_F^2 \qquad (1)$$

where X is a data matrix whose columns contains n instances in $\mathbb{R}^d$, W is a transformation matrix specified by k instances in $\mathbb{R}^d$, $\|.\|_F$ is the Frobenius norm and I is the identity matrix. Considering the problem, we are interested in finding the k-dimensional subspace of W on which the projection data has largest possible variance. This is the extension of technique introduced by [5] focusing on simplest

problem for $k = 1$ with a goal of finding w_1 largest eigenvalue. The extension is on finding $k > 1$ i.e. $w_1, w_2, w_3, \cdots$ simultaneously.

Let $x_1, \ldots, x_n$ represent the columns in $X \in R^d$ and Equation (1) can be equivalently written as

$$\min_{W \in R^{d \times k}: W^T W = I} -W^T \left(\tfrac{1}{n}\sum_{i=1}^{n} x_i x_i^T\right) W, \qquad (2)$$

The solution to this also gives top *k*-eigenvectors of covariance matrix $\frac{1}{n}\sum_{i=1}^{n} x_i x_i^T$.

3. Analysis

The following theorem by [5] highlights the importance of eigengap and how it affects the performance of algorithm 2. Eigen-gap is defined as the difference between successive eigenvalues when sorted in ascending order of magnitude

Theorem 1 [5]. *Define the* $d \times d$ *matrix A as* $\frac{1}{n}XX^T = \frac{1}{n}\sum_{i=1}^{n} x_i x_i^T$ *and let* v_1 *be an eigenvector corresponding to its largest eigenvalue. Suppose tha*t

i. $max_i \|x_i\|^2 \leq r \text{ for some } r > 0.$

ii. $A \text{ has eigenvalues } s_1 > s_2 \geq \cdots s_d, \text{where } s_1 - s_2 = \lambda \text{ for some } \lambda > 0.$

iii. $\langle \widetilde{w}_0, v_1 \rangle \geq \frac{1}{\sqrt{2}}.$

let $\delta, \epsilon \in (0,1)$ *be fixed. If we run the algorithm with any epoch parameter* m *and* η*, such that*

$$\eta \leq \frac{c_1 \delta^2}{r^2}\lambda, \quad m \geq \frac{c_2 \log(2/\delta)}{\eta^\lambda}, \quad m\eta^2 r^2 + rk\sqrt{m\eta^2 \log(2/\delta)} \leq c_3 \qquad (3)$$

(where c_1, c_2 *and* c_3 *designate certain positive numerical constants), and for* $T = \left[\frac{log(1/\epsilon)}{log(2/\delta)}\right]$ *epochs, then with probability at least* $1 - 2\,log(1/\epsilon)\,\delta$*, holds that*

$$\langle \widetilde{w}_0, v_1 \rangle \geq \frac{1}{\sqrt{2}}.$$

In [5], pointed out a practical algorithm, Variance-Reduced Principal Component Analysis (VR-PCA), for solving Equation (1) in which the algorithm was based on cheap stochastic iterations and the algorithm's runtime is logarithmic in the required accuracy $\epsilon \in (0,1)$. More precisely, for the case $k = 1$, x_i of bounded norm, and when there is an eigengap of λ between the first and second leading eigenvalues of the covariance matrix $\frac{1}{n}\sum_{i=1}^{n} x_i x_i^T$, the required runtime was proven to be on the order of

$$d_s\left(n + \frac{1}{\lambda^p}\right)\log\left(1/\epsilon\right)$$

This is suitable for getting high accuracy results as the runtime depends on ϵ, which is logarithmic, the runtime scales as the sum of the data size n and the factor involving the eigengap parameter λ, instead of their product. However, he showed the algorithm can still work even when λ is relatively small.

4. Solution Method

The following sections present the techniques that will be employed in solving Equation (1).

4.1. Block Oja and Block VR-PCA Methods

The Oja's method for handling the case $k > 1$, we let $W_t \in \mathbb{R}^{d \times k}$ denotes the algorithm's estimate of the eigenvectors of covariance matrix at time t. We then let η_t to denote the learning rate, and $X \in \mathbb{R}^{d \times n}$ be a random sample matrix with $x_1, x_2, \ldots \ldots, x_{n-1}, x_n$ columns $X \in \mathbb{R}^d$, given an optimization function

$$\min_{W \in R^{d \times k}: W^T W = I} -W^T \left(\frac{1}{n}\sum_{i=1}^{n} x_i x_i^T\right) W,$$

Then the block Oja's algorithm has the following update rule for tracking $k > 1$ subspace:

$$W'_t = W_{t-1} + \eta_t (x_i x_i^T W_{t-1}),\ W_t = \frac{W'_t}{\|W'_t\|}, \qquad (4)$$

Algorithm 1 Block Oja: for $k > 1$

- **Initialize** W_1 randomly on unit sphere
- **For** $t = 1, 2, \ldots \ldots, n$ **do**

  Pick $i_t \in \{1, 2, \ldots \ldots, n\}$ uniformly at random

$$W'_t = W_{t-1} + \eta_t x_{i_t} x_{i_t}^T W_{t-1}$$

$$W_t = \frac{W'_t}{\|W_t'^T W'_t\|}$$

For handling the case when $k > 1$ subspace in VR-PCA, more than one eigenvector should be required, where one wish to recover the leading eigenvectors $v_1, v_2, \ldots, v_k$ one after the other, each time using the $k = 1$ algorithm as shown by [5], this method is refers to as deflation method. However, to guarantee the convergence of this approach, positive eigengap between all top k eigenvalues is required, otherwise, the algorithm is not guaranteed to converge. Therefore, an algorithm which simultaneously recovers all k leading eigenvectors is preferable.

A block variant of Oja's method has been developed in the literature [6]–[8], in this, the method partitions the input into blocks and each time processes one block in a way similar to Oja's method. These methods are referred to as the block power method, or block Oja's method, or the noisy power method. Nevertheless, the block power method will be on VR-PCA introduced for by [5] for $k = 1$.

However, we need a block method of algorithm 1 to develop algorithm 2 by focusing on the following:

From the work of [5] we can apply an update by replacing the d-dimensional vectors w_{t-1}, $\widetilde{w}_{s-1}$, $\tilde{u}$ by $d \times k$ matrices W_{t-1}, $\widetilde{W}_{s-1}$, $\tilde{U}$, and normalization $w_t = \frac{w'_t}{\|w'_t\|}$ is replaced by $W_t = W'_t / \|W_t'^T W'_t\|$ makes W_t to orthonormal columns, this is what makes algorithms 1 and 2 respectively. The choice of η is always sufficiently small so that $W_t'^T W'_t$ invertible always.

$$W'_t = W_{t-1} + \underbrace{\eta A W_{t-1}}_{power\ /\ gradient\ step} + \underbrace{\eta\left(x_{i_t} x_{i_t}^T - A\right)\left(W_{t-1} - \widetilde{W}_{s-1}\right)}_{stochastic\ zero-mean\ noise}, \quad W_t = \frac{W'_t}{\|W'_t\|}$$

where A is a positive semi-definite matrix i.e. in PCA case $A = \frac{1}{n} XX^T = \frac{1}{n}\sum_{i=1}^{n} x_i x_i^T$

Algorithm 2 VR-PCA $k > 1$

Parameters: Step size η, epoch length m

Input: Data matrix $X = (x_1, \ldots, x_n)$; Initial $d \times k$ matrix $\widetilde{W}_0$

for $s = 1, 2, \cdots$ **do**

$\tilde{U} = \frac{1}{n}\sum_{i=1}^{n} x_i \left(x_i^T \widetilde{W}_{s-1}\right)$

$W_0 = \widetilde{W}_{s-1}$

for $t = 1, 2, \ldots, m$ **do**

$B_{t-1} = \left\| W_{t-1} - \widetilde{W}_{s-1} B \right\|$

Pick $i_t \in \{1, \ldots, n\}$ uniformly at random

$W'_t = W_{t-1} + \eta\left(x_{t_i}\left(x_{i_t}^T W_{t-1} - x_{i_t}^T \widetilde{W}_{s-1}\right) + \tilde{u} B_{t-1}\right)$

$W_t = \frac{W'_t}{\|W_t'^T W'_t\|}$

` **end for**

$\widetilde{W}_s = W_m$

Note, this method process the dataset to recover more than one eigenvectors simultaneously.

5. Simulation and Analysis

For the synthetic approach, we used similar approach used by [5] where for each choice of λ, we constructed a $d \times d$ diagonal matrix Σ, with diagonal $(1, 1 - \lambda, 1 - 1.1\lambda, \ldots, 1 - 1.4\lambda, p_1, p_2, \ldots)$ where $p_i = |q_i|/d$ and each q_i was chosen according to a standard Gaussian distribution, where $i = 1,2,3, \ldots$. We then let $X = U\Sigma V^T$, where U and V are random $d \times d$ and $n \times d$ orthogonal matrices. This results in a data matrix X whose spectrum is the same as Σ.

The situation when $k > 1$, i.e. where our target is to compute more than one leading eigenvector i.e. $v_1, v_2, \ldots, v_k$ simultaneously to handle the case for the tracking of a subspace with dimension greater than one using block matrix update of Oja's method and VR-PCA method respectively. A comparative study will be used to compare the performance of these two methods in terms of convergence rate that will be

generated from the synthetic data and the two real world datasets: Mixed National Institute of Standards and Technology (MNIST) dataset and wine dataset using MATLAB software.

6. Experiment and Discussion of Results

We are now going to present a comparison of some experiments based on the performance of the prominent Oja's method with its variant VR-PCA in algorithm 1 and as well the block Oja's method with block VR-PCA method in algorithm 2. However, rather than tuning its parameters, we used the following fixed heuristic: The epoch length m was set to n (number of data points, or columns in the data matrix), and η was set $= \frac{1}{\tilde{r}\sqrt{n}}$, where $\tilde{r} = \frac{1}{n}\sum_{i=1}^{n}\|X_i\|^2$ is the average squared norm of the data. The choice of $m = n$ ensures that at each epoch, the runtime is almost equal between the stochastic updates and the computations of $\tilde{u}$. The choice of η is motivated by the theoretical analysis, which requires η on the order of $1/(max_i\|x_i\|^2\sqrt{n})$ in the place where m should be on the order of n. Likewise, we can have the choice of η to be readily computed from the data, and doesn't require knowledge of λ.

Firstly, we performed experiments on a synthetic datasets (where $n = 60$, $d = 60$) different small-value of eigengap λ as discussed in the simulation set-up to compare the performance in term of convergence rate on block VR-PCA with block Oja's methods for more than one subspace tracking, all the algorithms are initialized from the same random matrix, chosen uniformly from the unit ball. The results are displayed in the Figure 1, Figure 2, Figure 3, Figure 4 and Figure 5.

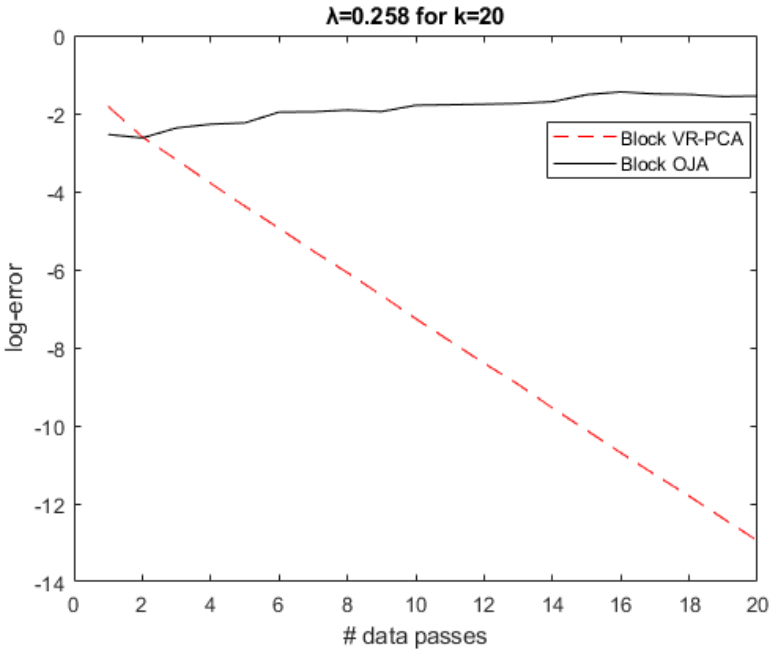

Figure 1: Lambda equals to 0.258

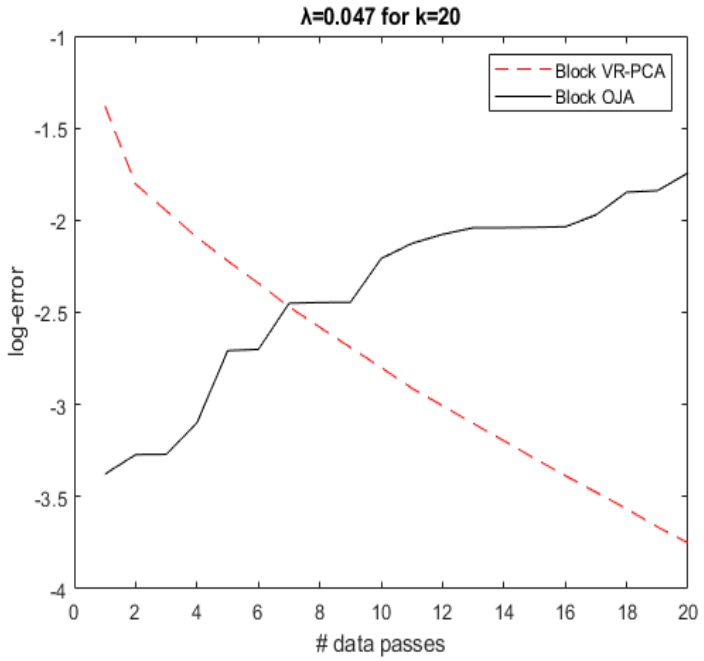

Figure 2: Lambda equals to 0.47

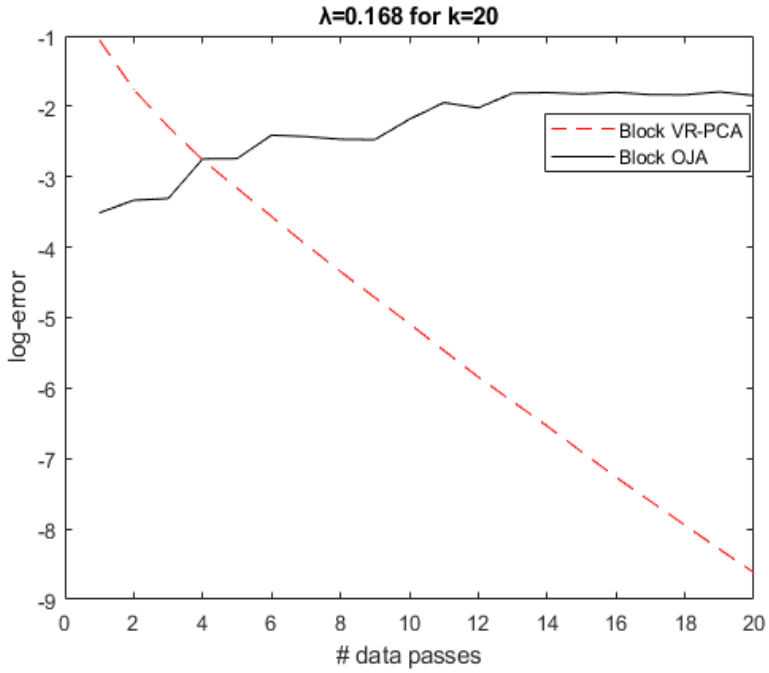

Figure 3: Lambda equals to 0.168

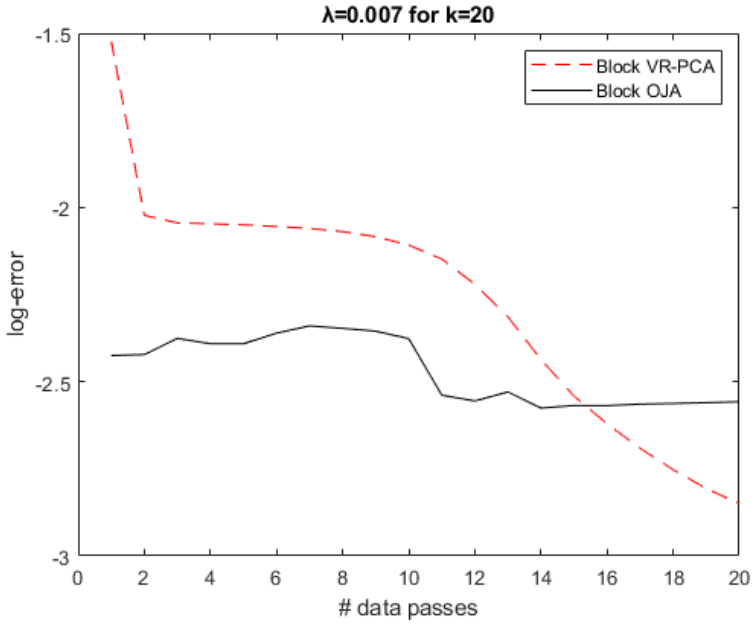

Figure 4: Lambda equals to 0.007

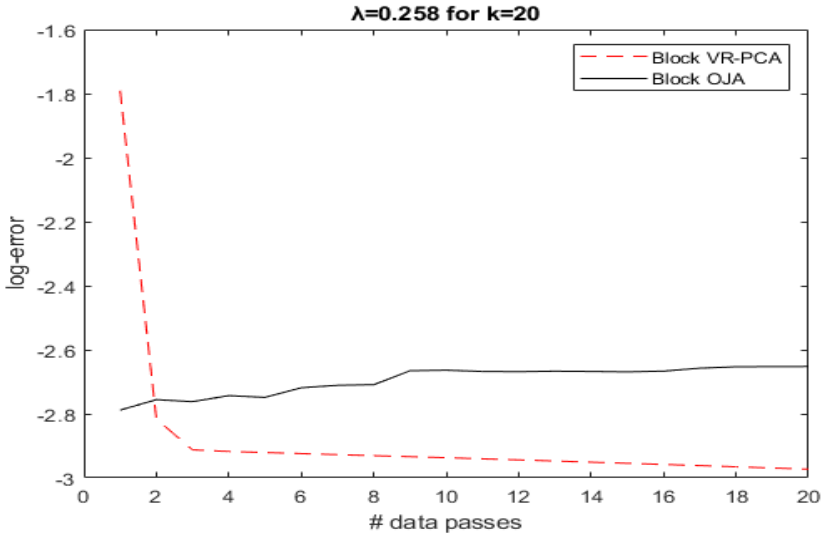

Figure 5: Lambda equals to 0.0017

In each and every plot of Figure 1, Figure 2, Figure 3, Figure 4 and Figure 5 above, x-axis represents the number of effective passes and y-axis represents some errors in logarithmic scale, i.e. y-axis equals to $\log_{10}\left(1-\frac{\|x^T w\|^2}{\max_{V:\|V\|=1}\|x^T v\|^2}\right)$. However, their convergence rate is appeared exponentially in nature. Nevertheless, the block VR-PCA method shows more efficiency in terms of convergence rate with minimum error for even much smaller value of eigengap chosen than the block method of prominent Oja's method, these similar results and approaches but in targeting single vector at a time are found in [5].

Similarly, similar experiments but with y-axis equals to $\log_{10}\left(1-\frac{\|x^T w\|^2}{\max_{V:\|V\|=1}\|x^T v\|^2}\right)$ are conducted using synthetic data on block Oja's method and block VR-PCA method, the results proved block VR-PCA method converges more exponentially than prominent Oja's method with a minimum error as appeared in the Figure 6 and Figure 7 respectively.

However, similar experiment is conducted using the well-known Mixed National Institute of

Standards and Technology (MNIST) training dataset and wine dataset. The MNIST data matrix size is 4000×784, in which we targeted $k = 20$ eigenvectors, and for the wine matrix size 178×13, in which we utilized the whole data by taking $k = 13$. These two data are sourced from [9]. The result of this, showed similar result as appeared in the synthetic dataset of the block VR-PCA and block Oja's methods, by still reflecting good convergence rate with minimum error on block VR-PCA method than block Oja's method. The y-axis here is equals to $\log_{10}\left(1 - \frac{\|x^T w\|^2}{\max_{V:V^TV=I} \|x^T v\|^2}\right)$. The result of this comparison is shown in the Figure 6 and Figure 7.

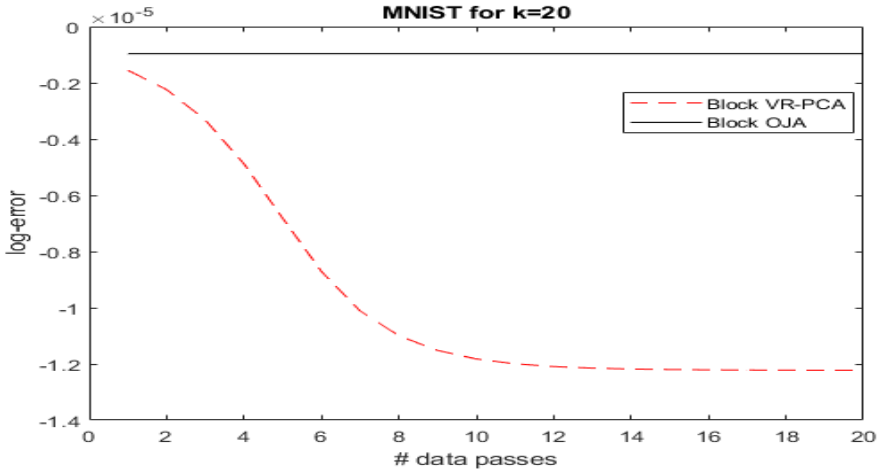

Figure 6: Result from MNIST dataset

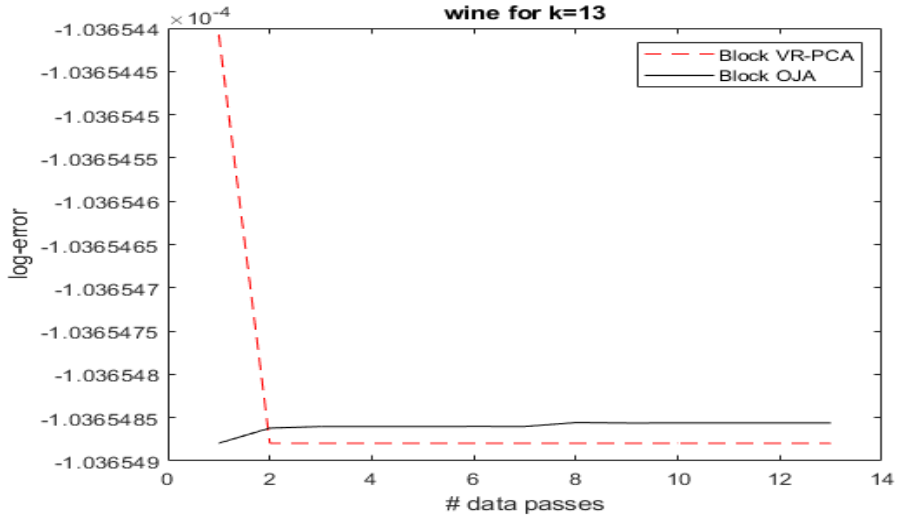

Figure 7: Result from wine dataset

5. Summary and Conclusion

The performance of VR-PCA and Oja gradient methods were studied in handling only one sample at a time; a steady convergence was noticed in their simulation experiments for k = 1 for higher value of eigengap, while for other smaller values of eigengap for k = 1, an exponential convergence was achieved this can be notice in [5]. However, we developed their blocks methods to handled more than one sample at a time, and this showed good performance experimentally. Using these blocks methods on simulation experiment for k = 20 we achieved exponential convergence for the higher value of eigengap while we achieved steady state convergence for the very small value of eigengap. Likewise, for both the MNIST and wine real datasets, we achieved a steady convergence state. The overall results showed Block VR-PCA method achieved minimum convergence with smaller error than Block Oja's method. Hence, these blocks methods were able to compute more than one eigenvalue at a time.

Conflict of interest

The authors declare no conflict of interest.

References

[1] P. Honeine, “Online kernel principal component analysis: A reduced-order model,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 9, pp. 1814–1826, 2012.

[2] O. Shamir, “Convergence of Stochastic Gradient Descent for PCA,” *Int. J. Mach. Learn.*, vol. 48, no. 1, pp. 257–267, 2016.

[3] J. Shlens, “A Tutorial on Principal Component Analysis,” 2014.

[4] L. Balzano, Y. Chi, and Y. M. Lu, “Streaming PCA and Subspace Tracking: The Missing Data Case,” *Proc. IEEE*, vol. 106, no. 8, pp. 1293–1310, 2018.

[5] O. Shamir, “Convergence of Stochastic Gradient Descent for PCA,” vol. 48, no. 1, 2015.

[6] I. Mitliagkas, C. Caramanis, and P. Jain, “Streaming PCA with Many Missing Entries,” 2014.

[7] M. Hardt and E. Price, “The Noisy Power Method : A Meta Algorithm with Applications,” pp. 1–9, 2014.

[8] M. Balcan, S. S. Du, Y. Wang, and A. W. Yu, “An Improved Gap-Dependency Analysis of the Noisy Power Method,” vol. 49, pp. 1–26, 2016.

[9] E. K. Dheeru, D., & Taniskidou, “UCI machine learning repository.,” 2017. [Online]. Available: https://archive.ics.uci.edu/ml/index.php.

