Neural Networks 85 (2017) 33–50

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

An online incremental orthogonal component analysis method for dimensionality reduction

Tao Zhu [a], Ye Xu [b,1], Furao Shen [a,*], Jinxi Zhao [a]

[a] National Key Laboratory for Novel Software Technology, Department of Computer Science and Technology, Nanjing University, Nanjing, PR China
[b] 6211 Sudikoff Lab, Dartmouth College Hanover, NH 03755, United States

ARTICLE INFO

Article history:
Received 29 May 2016
Received in revised form 17 August 2016
Accepted 4 October 2016
Available online 14 October 2016

Keywords:
Dimensionality reduction
Orthogonal component
Incremental learning
Automatic target dimension estimation
Online learning

ABSTRACT

In this paper, we introduce a fast linear dimensionality reduction method named incremental orthogonal component analysis (IOCA). IOCA is designed to automatically extract desired orthogonal components (OCs) in an online environment. The OCs and the low-dimensional representations of original data are obtained with only one pass through the entire dataset. Without solving matrix eigenproblem or matrix inversion problem, IOCA learns incrementally from continuous data stream with low computational cost. By proposing an adaptive threshold policy, IOCA is able to automatically determine the dimension of feature subspace. Meanwhile, the quality of the learned OCs is guaranteed. The analysis and experiments demonstrate that IOCA is simple, but efficient and effective.

1. Introduction

How to efficiently and effectively extract useful information from high-dimensional data is an open problem worth studying. One of the biggest challenges is "the curse of dimensionality". It is caused by the high dimensional form of original data and the lack of training samples (Bishop, 2006). On the other hand, with the developing of information science, the amount of fresh data being produced increases exponentially, the world is entering the age of "Big Data" (Lohr, 2008). The collected datasets are so large and complex that it becomes almost impossible to process them directly. Dimensionality reduction (DR) is an important tool for us to address both of these two problems which are contradictory and interrelated with each other.

In data mining and machine learning areas, researchers have paid great attention on finding or creating methods to obtain the most essential information from original data while discarding noise information (Fodor, 2002; Huo & Smith, 2008; Sarveniazi, 2014).

A great sum of dimensionality reduction algorithms have been proposed, such as principal component analysis (PCA) (Jolliffe, 1986), linear discriminant analysis (LDA) (Fukunaga, 1990), non-negative matrix factorization (NMF) (Lee & Seung, 1999), locally linear embedding (LLE) (Roweis & Saul, 2000), isometric mapping (Isomap) (Tenenbaum, de Silva, & Langford, 2000), and neural networks (NN) (Bishop, 1996). These methods and their variants have been proved to be efficient and widely used in face recognition (Lee, Ho, & Kriegman, 2005; Lu, Tan, & Wang, 2011), image compression (Ye, Janardan, & Li, 2004), biometrics (Lee & Zhang, 2006; Zhang, Li, Tao, & Yang, 2008), text processing (Torkkola, 2001), pose estimation and tracking (Tao, Li, Wu, & Maybank, 2007; Wang, Xu, & Ai, 2003; Weinberger & Saul, 2004), etc. By applying various objective functions, they convert original high-dimensional data space into low-dimensional feature space in different ways.

Unfortunately, the classical methods are designed to deal with off-line data and they are not suitable for online learning. Furthermore, they need user to predetermine the dimension of the feature subspace (the target dimension).

In this paper, we introduce a high-speed dimensionality reduction method named incremental orthogonal components analysis (IOCA) to handle the above mentioned problems. By proposing an adaptive threshold policy, IOCA is able to (1) keep learning from continually input data; (2) achieve high-speed orthogonal component (OC) learning; (3) automatically estimate and update the target dimension; (4) obtain numerically orthogonal components.

The rest of this paper is structured as follows. Some important related works are introduced Section 2. In Section 3, we introduce IOCA and discuss some related problems. Moreover, theoretical

* Corresponding author.
E-mail addresses: tao144@gmail.com (T. Zhu), ye@cs.dartmouth.edu (Y. Xu), frshen@nju.edu.cn (F. Shen), jxzhao@nju.edu.cn (J. Zhao).

[1] This work was done when the author was in Nanjing University.

http://dx.doi.org/10.1016/j.neunet.2016.10.001
0893-6080/

analysis is given in Section 4. In Section 5, experiments are taken and experimental results are reported. Finally, in Section 6, we conclude the paper and briefly discuss our further works.

2. Related work

In recent years, incremental learning has attracted great attention due to the increasing demand for systems have the ability of learning and evolving. When new high-dimensional data are continually input, incremental learning methods updated the learned model without recalculating the whole model repeatedly. Obviously, these methods enjoy a great advantage: their computational and storage cost is greatly reduced while the performance is improved.

Most of the existing linear incremental dimensionality reduction methods focus on efficiently adjusting the existing eigenspace model with new samples.

PCA and LDA are the most widely-used unsupervised DR algorithm and supervised DR algorithm, respectively, a great number of PCA-based or LDA-based incremental algorithms have been proposed. Incremental PCA (IPCA) is described in Hall, Marshall, and Manin (1998) by Hall et al. Then Artač et al. successfully employed it in online object learning and recognition (Artač, Jogan, & Leonardis, 2002). Furthermore, Hall et al. presented a approach that not only can merge two eigenspace models but also can split two eigenspace models in learning process (Hall, Marshall, & Martin, 2000). Ren and Dai proposed an incremental method BDPCA which is based on singular value decomposition (SVD) (Ren & Dai, 2010). Weng et al. proposed a covariance-free incremental PCA algorithm for online principal components computation (Weng, Zhang, & Hwang, 2003). Pang et al. proposed an incremental LDA algorithm that solved the problem of scatter matrix's updating (Pang, Ozawa, & Kasabov, 2005). With the help of fast SVD updating technique, Zhao and Yuen proposed an incremental supervised learning method called GSVD-ILDA (Zhao & Yuen, 2008). Ye et al. achieved incremental dimension reduction via QR decomposition (Ye, Li, Xiong, & Park, 2005). Kim et al. applied the concept of sufficient spanning set in approximation and proposed incremental LDA algorithm that is successfully employed in different applications (Kim, Stenger, Kittler, & Cipolla, 2011).

Moreover, researchers also have proposed incremental DR algorithms that employed different strategies. Guan et al. proposed an online NMF algorithm named OR-NMF that employed robust stochastic approximation in online basis matrix updating (Guan, Tao, Luo, & Yuan, 2012). Based on orthogonality constrains and the assumption that the coefficients of old data do not change in learning progress, Wang et al. proposed IOPNMF (Wang & Lu, 2013). Law et al. proposed an incremental version of Isomap by employing the previous computation results to update the geodesic distances and eigenvectors (Law & Jain, 2006). Mairal et al. proposed an online dictionary learning algorithm based on the concept of sparse coding (Mairal & Bach, 2010).

All those methods successfully achieve incremental learning while their computational cost is acceptable.

How to set the target dimension is another important problem for dimensionality reduction. Ideally, we hope the target dimension is set equal to the intrinsic dimension of the dataset. According to Fukunaga's definition (Fukunaga, 1982), a dataset $X \subseteq \Omega^d$ is said to have intrinsic dimension equal to k if its elements lie entirely within a k-dimensional subspace (where $k < d$). The intrinsic dimension estimation approaches can be grouped into two categories (Fan, Qiao, & Zhang, 2009): the geometric approaches and the projection approaches. The geometric approaches employ the geometric structure of data for intrinsic dimension estimation (Fan, Zhang, Chen, Bao, & Maybank, 2013). However, they usually need a lot of available training samples. Projection approaches determine the target dimension when the projection of original data has been done. For these approaches, the target dimension is usually determined by counting the number of significant eigenvalues (Hall et al., 2000). Usually projection approaches need an absolute threshold that is empirically set. However, Bishop proposed Bayesian PCA that is able to automatically determine the target dimension by employing EM algorithm (Bishop, 1999). Bayesian PCA is particularly advantageous for small data sets in high dimensions. Unfortunately, due to the great computational cost in the iterations, Bayesian PCA would not be suitable for large-scale or online data.

In this paper, we try to solve the intrinsic dimension estimation problem while achieving incremental learning. In online environment, the available data set dynamically changes, the proper target dimension may also change. Therefore, during the learning process, the proposed IOCA method prefers to adaptively expands the feature space rather than adjusting the feature space whose dimension is fixed. The comparison between IOCA and the existing non-incremental and incremental dimensionality reduction methods is illustrated in Fig. 1.

As a linear DR algorithm, IOCA is designed to extract numerically orthogonal components. Many researches have indicated that compared with non-orthogonal components, orthogonal components are more desired in dimensionality reduction (Cai & He, 2005; Liu & Yu, 2005).

The Gram–Schmidt (GS) (Golub & Loan, 1996) process is the most famous method for orthonormalizing a finite and linearly independent set of vectors. Given vector set $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\}$, the orthogonal set $\{\boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_N\}$ is generated as follows:

$$\boldsymbol{r}_1 = \boldsymbol{x}_1 \quad \boldsymbol{b}_1 = \frac{\boldsymbol{r}_1}{\|\boldsymbol{r}_1\|_2} \tag{1}$$

$$\boldsymbol{r}_t = \boldsymbol{x}_t - \sum_{j=1}^{t-1} \boldsymbol{b}_j \boldsymbol{b}_j^\top \boldsymbol{x}_t \quad \boldsymbol{b}_t = \frac{\boldsymbol{r}_t}{\|\boldsymbol{r}_t\|_2}. \tag{2}$$

GS processes the vectors in sequence. Obviously, it has the excellent property of low computational complexity. However, classical GS has a serious drawback: due to round-off error, it is numerically unstable. Round-off error is the difference between the calculated approximation of a number and its exact mathematical value. It may accumulate through a sequence of calculations. Sometimes significant error has accumulated and it dominates the calculations (Chartier, 2006). For GS process, when the ℓ_2-norm of the obtained $\boldsymbol{r}_t$ is very small, the normalize operation $\frac{\boldsymbol{r}_t}{\|\boldsymbol{r}_t\|_2}$ may greatly magnify the round-off error occurs in the calculation of $\boldsymbol{r}_t$. In this case, suppose $\boldsymbol{b}_i$ is one of the previously obtained basis, though the value of $\|\boldsymbol{r}_t\|_2$ is so small that $\boldsymbol{r}_t^\top \boldsymbol{b}_i$ is very close to 0, $\frac{\boldsymbol{r}_t^\top \boldsymbol{b}_i}{\|\boldsymbol{r}_t\|_2}$ may be much larger than 0, i.e. $\boldsymbol{b}_t$ is not orthogonal to $\boldsymbol{b}_i$. Take the $n \times n$ Hilbert matrix $\boldsymbol{H}(n)$ (Choi, 1983) for example. Theoretically speaking, the dimension of the space that is spanned by $\boldsymbol{H}(n)$'s column vectors should be n. In a case study, we employed GS on $\boldsymbol{H}(100)$'s 100 column vectors in Matlab. Assume the columns of matrix $\boldsymbol{B}$ are calculated by (1) and (2), theoretically, we should have $\|\boldsymbol{I}_{\boldsymbol{100}} - \boldsymbol{B}^\top \boldsymbol{B}\|_2 = 0$, $\boldsymbol{I}_{\boldsymbol{100}}$ is the 100×100 identity matrix. However, in practice, we obtained $\|\boldsymbol{I}_{\boldsymbol{100}} - \boldsymbol{B}^\top \boldsymbol{B}\|_2 \approx 92.11$, this means that the mutual orthogonality of the learned basis is completely destroyed. Fortunately, researchers have paid great attention to solve this problem and fruitful achievements have been made (Leon, Björck, & Gander, 2013).

Selecting sample data that are fairly linear independent is a common strategy to ensure the quality of GS's output. Data selection is also an effective technology for improving DR algorithms' performance (Gheyas & Smith, 2010; Hua, Tembe, & Dougherty, 2008; Jin, Xu, Bie, & Guo, 2006; Peng, Long, & Ding,

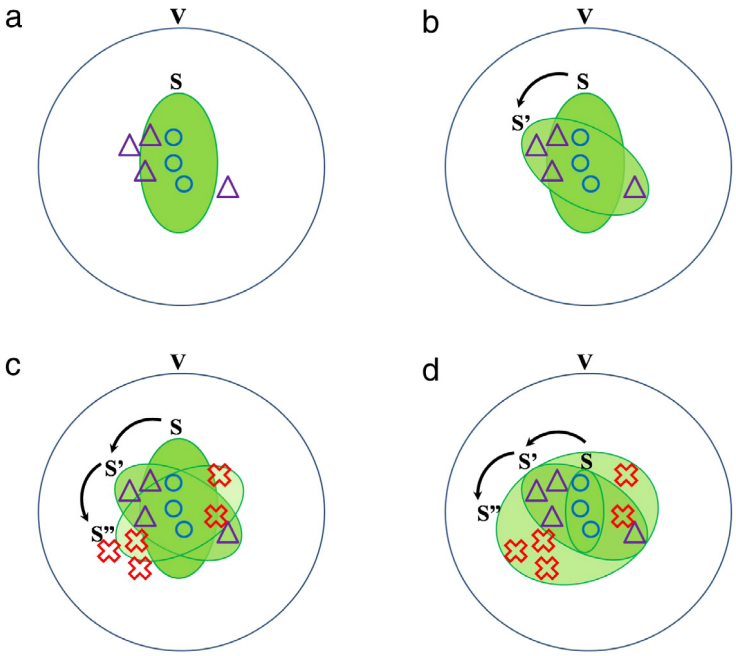

Fig. 1. V is a linear space, S is its feature subspace. In these figures, the larger the area of S is, the higher the dimension of S is. The blue circles are the original training data; the purple triangles and red crosses are newly input data. (a) Non-incremental methods cannot adjust S without completely recalculating it. (b) Incremental methods have the ability of applying new samples to update S. (c) Nevertheless, most incremental methods cannot automatically determine the dimension of S (the area of S cannot change). Meanwhile, the user may predetermine an unsuitable target dimension. (d) IOCA is able to automatically and adaptively expand S (the area of S increases) and estimate its intrinsic dimension during incremental learning. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

2005). The concept of GS with pivoting has been employed by many algorithms for selection purposes (Araújo et al., 2001; Gillis & Vavasis, 2014; Ren & Chang, 2003), although their final goal is not extracting orthogonal components. Especially, in Gillis and Vavasis (2014), the authors theoretically proved that under certain assumptions, this family of algorithms are robust under any small perturbations of the input data. It should be emphasized that these algorithms are not able to automatically estimate the number of extracted vectors.

Based on the previous studies, in IOCA, we employed GS-based method for OC extraction while proposing an adaptive threshold policy that is the key of the algorithm. With the help of the threshold policy, only the ideal candidate components will be accepted and thus IOCA is able to automatically determine the proper number of components in online environment. Meanwhile the numerical orthogonality of learned OCs is guaranteed. Although the threshold policy is simple, it needs little prior knowledge and extra computational cost. Therefore, like classical GS process, IOCA avoids time consuming solving matrix eigenproblem or matrix inversion problem and enjoys low time complexity.

3. Incremental orthogonal components analysis

The main principle of IOCA is "entities should not be multiplied unnecessarily". Assume S is the obtained feature subspace, $\boldsymbol{x}$ is input data. As illustrated in Fig. 2, IOCA does not extract new OCs from the data that are highly linear dependent on the learned feature subspace S. An adaptive threshold T is employed in decision making. If $\boldsymbol{x}$ is strongly linear independent with S, IOCA believes that at this time the dimension of S is too small and S cannot represent the original dataset well; therefore, a new basis vector of S is extracted from $\boldsymbol{x}$ and S is enlarged. Otherwise, IOCA believes it is not necessary to update S; then, S is maintained. Meanwhile, $\boldsymbol{x}$'s low-dimensional representation is obtained.

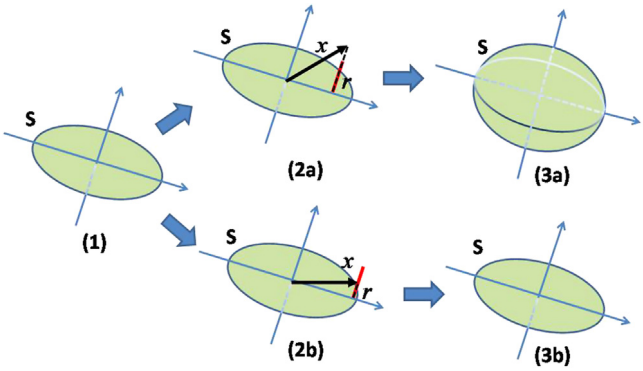

Fig. 2. S is the feature subspace. Input vector $\boldsymbol{x}$ is projected onto S and its complemented subspace $S^{\perp}$. $\|\boldsymbol{r}\|_2$ measures the linear dependence between $\boldsymbol{x}$ and S. The adaptive threshold T is represented by the red line. (2a) If $\|\boldsymbol{r}\|_2$ is larger than T, (3a) IOCA will extract a new base vector and enlarge S. (2b) Otherwise, (3b) no new base vector will be extracted and S remains unchanged. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

3.1. Linear independence measure

The learning process of IOCA can be seemed as a process of continually extracting necessary new basis vectors and enlarging the feature subspace S based on the input data sequence. S is represented by the extracted OCs.

The blind pursuit of finding out all the precise OCs of one certain dataset without component selecting will not only increase the computational burden but also poison the data analysis process. Therefore, IOCA employs a straightforward strategy to guarantee the numerical orthogonality between the learned OCs $\boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_k$ and the new candidate component $\boldsymbol{b}_{k+1}$ extracted from input data $\boldsymbol{x}_t$: If the $\boldsymbol{x}$ is not strongly linear independent of learned components, $\boldsymbol{b}_{k+1}$ will be directly discarded.

The linear dependence theorem (Gillis, 2001) indicates that the linear independence between a space $S = span\{\boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_k\}$ and data vector $\boldsymbol{x}_t$ can be measured by the projection distance:

$$\|\boldsymbol{r}_t\|_2 = \min_{\boldsymbol{y}'} \|\boldsymbol{x}_t - \boldsymbol{B}^{(k)}\boldsymbol{y}'\|_2. \tag{3}$$

Here, matrix $\boldsymbol{B}^{(k)} = [\boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_k] \in \mathbb{R}^{d\times k}$, $\boldsymbol{y}' = (y_1, y_2, \ldots, y_k)^\top$ is the coordinate vector.

Due to the orthogonality of learned components, $\boldsymbol{r}$ can be calculated rapidly by following steps: Let $\boldsymbol{r}_t = \boldsymbol{x}_t$, for $i = 1, 2, \ldots, k$, $\boldsymbol{r}_t = \boldsymbol{r}_t - \boldsymbol{b}_i\boldsymbol{b}_i^\top\boldsymbol{r}_t$. Theoretically speaking, the above operations that are employed by modified Gram–Schmidt (MGS) are equivalent to directly calculating $\boldsymbol{r}_t = \boldsymbol{x}_t - \sum_{i=1}^{k}\boldsymbol{b}_i\boldsymbol{b}_i^\top\boldsymbol{x}_t$. However, each time the operation $\boldsymbol{r}_t = \boldsymbol{r}_t - \boldsymbol{b}_i\boldsymbol{b}_i^\top\boldsymbol{r}_t$ finds a residual vector $\boldsymbol{r}_t$ that is orthogonal to the obtained $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_i$, thus $\boldsymbol{r}_t$ is also orthogonalized against any errors introduced in computation of $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_i$. By this small modification, the numerical stability of the algorithm can be improved. Then $\|\boldsymbol{r}\|_2$ is employed in the proposed adaptive threshold policy.

3.2. Adaptive threshold policy

The adjustment of feature subspace S is influenced by two aspects. On the one hand, we hope S to contain as much information as possible, thus S tends to expand itself. On the other hand, as a dimensionality reduction task, it is natural that the compression ratio (i.e. the dimension of basis dividing the dimension of input data ($\frac{k}{d}$)) should be as low as possible. Therefore, when feature subspace S is small, it tends to learn more information from input data; when S is large, preventing its blind expansion is more important than learning new knowledge. Expansion or maintenance, IOCA should automatically take balance between them by threshold T. Once the balance is achieved, the target dimension is determined.

As described above, the proposed threshold T should satisfy the following two constraints: (i) The strong orthogonality between the learned components should be ensured. (ii) The difficulty of accepting components increases with the growth of the feature subspace S.

To satisfy these two constraints, we set $T = f(\frac{k}{d})\|\boldsymbol{x}_t\|_2$ and write the adaptive threshold policy as follows: if and only if

$$\|\boldsymbol{r}_t\|_2 \geq f\left(\frac{k}{d}\right)\|\boldsymbol{x}_t\|_2 \tag{4}$$

we accept $\boldsymbol{b}_{k+1} = \frac{\boldsymbol{r}_t}{\|\boldsymbol{r}_t\|_2}$ as a new component.

Obviously, $\|\boldsymbol{r}_t\|_2$ will never be larger than $\|\boldsymbol{x}_t\|_2$. We hope that as $\frac{k}{d}$ increases, it becomes more difficult for IOCA to accept $\boldsymbol{b}_{k+1}$. Therefore, we define that $f(\omega)$ is a strictly monotonic increasing function and $0 \leq f(\omega) \leq 1$ when $0 \leq \omega \leq 1$.

As mentioned above, when GS process is implemented on a computer, if $\|\boldsymbol{r}_t\|_2$ is very small, the round-off error may be magnified in the calculations, then $\frac{\boldsymbol{r}_t}{\|\boldsymbol{r}_t\|_2}$ is often not quite orthogonal with the previously obtained components. We have to consider about the situation that when the ℓ_2-norm of input vector $\boldsymbol{x}_t$ is very small. In this case, even though (4) is satisfied, $\|\boldsymbol{r}_t\|_2$ may be still very small, then IOCA algorithm may not obtain a component $\boldsymbol{b}_{k+1}$ which is numerically orthogonal with the learned components.

Inspired by pivot selection scheme, we can solve the problem in an uncomplicated way. We modify (4) as

$$\frac{\|\boldsymbol{r}_t\|_2}{L_{\max}^{(t)}} \geq f\left(\frac{k}{d}\right). \tag{5}$$

$L_{\max}^{(t)}$ is the longest ℓ_2-norm of the t available sample vectors. There is no doubt that the value of $\frac{\|\boldsymbol{r}_t\|_2}{L_{\max}^{(t)}}$ is also between 0 and 1. Through this modification, the threshold of IOCA is a bit stricter, but the algorithm is better at discarding unreliable results and preventing the calculations to magnify the round-off error. Therefore, the numerical stability of the algorithm is enhanced.

In sum, an adaptive threshold is employed for IOCA to estimate the dimension of feature subspace. It is much easier for user to choose a not-so-bad threshold function than to determine the exact value of the threshold. The proposed threshold can automatically adjust itself according to the current situation. The influence of the improper initialization of threshold can be reduced. And the target dimension is determined by IOCA with little prior knowledge and few additional calculations.

3.3. Algorithm of IOCA

In Algorithm 1, we give the detailed IOCA algorithm. In the beginning, feature subspace S is initialized as a zero-dimensional space. Suppose when the tth data $\boldsymbol{x}_t$ is input, $\boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_k$ are the k OCs have been learned, IOCA tries to update S by extracting candidate OC $\boldsymbol{b}_{k+1}$ from $\boldsymbol{x}_t$ and outputs $\boldsymbol{x}_t$'s low-dimensional representation $\boldsymbol{y}_t$. Then, IOCA continues to process the $t+1$th data until there is no new data.

Algorithm 1 Incremental Orthogonal Component Analysis

Initialize basis set $B^{(0)} = \emptyset$ and its dimension $k = 0$.
Initialize $L_{\max} = 0$.
for each input new pattern $\boldsymbol{x}_t$ $(t = 1, 2, \ldots)$ **do**
  if $\|\boldsymbol{x}_t\|_2 > L_{\max}$ **then**
    $L_{\max} = \|\boldsymbol{x}_t\|_2$.
  end if
  Let $\boldsymbol{r}_t = \boldsymbol{x}_t$.
  for $i = 1 : k$ **do**
    Compute $y_{t,i} = \boldsymbol{r}_t{}^\top\boldsymbol{b}_i$, let $y_{t,i}$ be the ith entry of $\boldsymbol{y}_t$.
    Compute $\boldsymbol{r}_t = \boldsymbol{r}_t - y_{t,i}\boldsymbol{b}_i$.
  end for
  Compute $\boldsymbol{b}_{k+1} = \frac{\boldsymbol{r}_t}{\|\boldsymbol{r}_t\|_2}$.
  if $\frac{\|\boldsymbol{r}_t\|_2}{L_{\max}} \geq f\left(\frac{k}{d}\right)$ **then**
    Accept $\boldsymbol{b}_{k+1}$ as a component and let $B^{(k+1)} = B^{(k)} \bigcup \{\boldsymbol{b}_{k+1}\}$.
    Let $y_{t,k+1} = \|\boldsymbol{r}_t\|_2$ be the $(k+1)^{th}$ entry of $\boldsymbol{y}_t$.
    Update basis dimension $k = k + 1$.
  end if
end for

Assume $k = k^{(t)}$ when $\boldsymbol{x}_t$ is processed and the value of final k is $k^{(N)}$ (obviously, $k^{(t)} \leq k^{(N)}$), for each $\boldsymbol{x}_t$, its low-dimensional representation $\boldsymbol{y}_t$ can be written as a $k^{(N)}$-dimensional vector $(y_{t,1}, y_{t,2}, \ldots, y_{t,k^{(t)}}, 0, \ldots, 0)^\top$. In this way, IOCA obtains the OCs and the low-dimensional representation of each input data with only one pass through the entire dataset. This property of IOCA is a great advantage for reducing computational load in large-scale data processing.

The time complexity of IOCA is $O(Ndk)$, N is the training set size, d is the dimension of original data, and $k = k^{(N)}$ is the number of OCs eventually learned by the algorithm.

Note that the algorithm of IOCA is concise and its time complexity is low. IOCA does not need to solve matrix eigenproblem or matrix inversion problem. The adaptive threshold prevents the round-off error in OC calculations. Therefore, the proposed method enjoys a low computational load and high numerical stability. Because of its simplicity, IOCA has few limits in applications and has the potential to be a universal approach in dimensionality reduction.

4. Analysis

4.1. Analysis on the process of IOCA

Initially, the feature subspace is a zero-dimensional subspace. As new data are input, IOCA automatically decides whether to enlarge the feature subspace. Once the feature subspace has stopped expanding, the intrinsic dimension of the original dataset is estimated and the task of dimensionality reduction is fulfilled.

As an online learning method needs little prior knowledge of the original data, IOCA employs a "greedy" strategy in feature subspace learning. Therefore, IOCA has the advantage of low cost. However, different input sequences of the same dataset may result in different outputs of IOCA. In other words, the final determined component number and the orthogonal representation of basis set are not only dependent on the dataset itself but also dependent on the random input order of these data. From this perspective, IOCA is not a deterministic algorithm, but a randomized algorithm.

To analyze the learning process of IOCA, we have to study when the dimension of feature subspace stops increasing. For the convenience of analysis, in this section, we make a natural hypothesis that all the input data are d-dimensional independent and identically distributed (i.i.d.) random vectors follow the same zero-mean isotropic distribution.

Theorem 4.1. *Assuming that all data are i.i.d. d-dimensional random vectors follow the same zero-mean isotropic distribution, when k OCs have been learned by IOCA, the value of $f\left(\frac{k}{d}\right)$ is $\sqrt{\alpha}$, here $k \geq (1-\alpha)d$. For the next input vector $\boldsymbol{x}$, $Pr\left[Accept\ \boldsymbol{b_{k+1}}\right]$ is the possibility for IOCA to accept new component $\boldsymbol{b_{k+1}}$ extracted from it, we have*

$$Pr\left[Accept\ \boldsymbol{b}_{k+1}\right] \leq \exp\left(-\frac{(k-d+\alpha d)^2}{4k}\right). \tag{6}$$

The proof of Theorem 4.1 is given in Appendix A.

If we define $f(\omega) = \omega$, we may have the following corollary:

Corollary 4.1. *Assuming that all data are i.i.d. d-dimensional Gaussian random vectors with independent standard normal entries, $f\left(\frac{k}{d}\right) = \frac{k}{d}$ is the threshold function, when $k \geq \frac{\sqrt{5}-1}{2}d$ OCs has been learned by IOCA, the possibility for IOCA to accept $\boldsymbol{b_{k+1}}$ satisfies*

$$Pr\left[Accept\ \boldsymbol{b}_{k+1}\right] \leq \exp\left(-\frac{(k^2+dk-d^2)^2}{4d^2k}\right). \tag{7}$$

Note that the distribution of d-dimensional Gaussian random vectors with independent standard normal entries is isotropic and zero-mean. Corollary 4.1 is the specific form of Theorem 4.1 and it can be obtained directly from Theorem 4.1 by setting $\sqrt{\alpha} = \frac{k}{d}$: when $\sqrt{\alpha} = \frac{k}{d}$, $k \geq (1-\alpha)d$ and $k \leq d$ are all satisfied, we obtain $k \geq \frac{\sqrt{5}-1}{2}d$.

Based on Corollary 4.1, we can obtain the conclusion: If $k > \frac{\sqrt{5}-1}{2}d$, as the increase of k or d, the upper bound of the possibility for IOCA to accept a new component decreases monotonically. Fig. 3 describes the relationship between the possibilities for IOCA to accept new basis vector when d and k are various. It is obvious when d is small, the calculated upper bound of the possibility does not decrease to nearly 0 until k has been close to d. However, if d is sufficiently large, once k is a little greater than $\frac{\sqrt{5}-1}{2}d$, IOCA accepts new components with a very low possibility. For example, when $d = 5000$ and $k = 3500$, according to Corollary 4.1, we can calculate that: $Pr\left[\text{Accept } \boldsymbol{b}_{k+1}\right] \leq 1.01 \times 10^{-28}$. Suppose P_n is the possibility that IOCA does not extract the 3501th OC from the next n data, we have $P_n \geq (1 - 1.01 \times 10^{-28})^n \geq 1 - 1.01 \times 10^{-28}n$. Note that the upper bound obtained through (7) is very loose. Therefore, we can declare that with a sufficiently large d, even in the environment of "Big Data", with high possibility, k cannot increase to greater than $0.7d$.

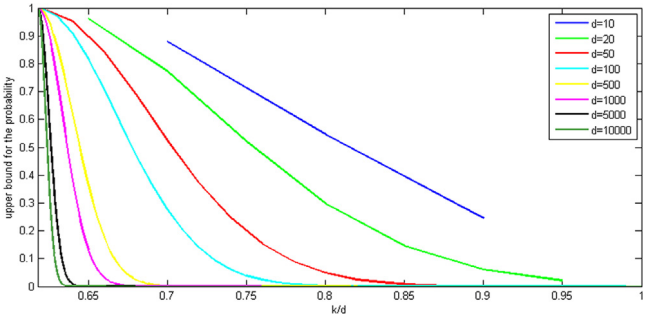

Fig. 3. The relationship between k/d and the upper bound of the possibility for IOCA to accept new component $\boldsymbol{b_{k+1}}$ extracted from the next input data. The possibility upper bound is calculated only when k is an integer. Here, the value of k/d ranges from $\frac{\sqrt{5}-1}{2}$ to 1.

Note that uniformly distributed data may be the most difficult case for component extracting methods. IOCA can still make feature subspace achieve an equilibrium in online learning process and automatically estimate target dimension k for the dataset. If the distribution of the dataset shows certain regularity, intuitively, it is much easier for IOCA to determine target dimension and the obtained k will be usually much smaller than d.

According to Corollary 4.1, if the assumptions are satisfied, when $f(\omega) = \omega$ is employed as threshold variable and d is sufficiently large, even the input dataset size N is very large, with high possibility the final compression ratio of IOCA will be slightly larger but close to $\frac{\sqrt{5}-1}{2}$ which is also known as the "Golden Ratio". Moreover, when the large value of N is fixed ($N \gg d$), as d increases, IOCA's compression ratio will converge to the "Golden Ratio". Therefore, we define the default threshold as $T = \frac{k}{d}L_{\max}$ and employ it in our experiments.

4.2. Analysis on IOCA's effectiveness

In some aspects, IOCA seems like random orthogonal projection (ROP) (Bingham & Mannila, 2001). However, thanks to the proposed adaptive threshold policy, we can clearly declare that IOCA is surely better than the common random projection methods. In IOCA, for each $\boldsymbol{x}_t$, to a certain extent, its low-dimensional representation is quality-assured. It is easy to understand that the performance of random projection will particularly suffer from the problem caused by poor approximation. Given an arbitrary dataset and a improper small k, when the random projection is employed on it, it is almost sure that a large part of original data will be poorly approximated by the projected low-dimensional vectors (Zhang, Mahdavi, Jin, & Yang, 2012). On the other hand, when the current basis cannot approximate $\boldsymbol{x}_t$ well (the input vector $\boldsymbol{x}$ is strongly independent to the learned basis), IOCA extracts a new OC $\boldsymbol{b}_{k+1}$ and updates feature subspace dimension. After that has been done, it is obvious that this $\boldsymbol{x}_t$ can be perfectly represented. This ensures that the number of components finally learned by IOCA will never be improperly small and the approximation error for each data vector is bounded.

Theorem 4.2. *Assuming that the final feature subspace dimension is $k = k^{(N)}$, we have $L^{(N)}_{\max} = \max\{\|\boldsymbol{x}_1\|_2, \|\boldsymbol{x}_2\|_2, \ldots, \|\boldsymbol{x}_N\|_2\}$. For each data $\boldsymbol{x}_t$, there is a determined upper bound for its approximation error:*

$$\|\boldsymbol{r}_t\|_2 < f\left(\frac{k^{(N)}}{d}\right)L^{(N)}_{\max}. \tag{8}$$

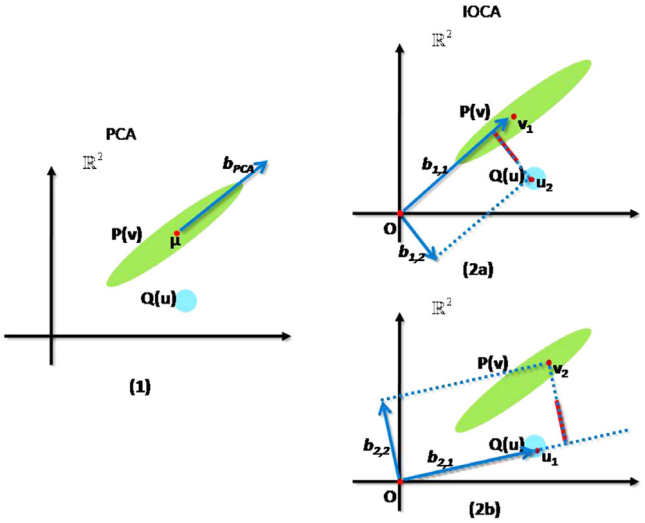

Fig. 4. The differences between PCA and IOCA. We assume that high-dimensional data lie in a 2-dimensional subspace, and the whole dataset contains two distributions: $\boldsymbol{P}(\boldsymbol{v})$ (green area) and $\boldsymbol{Q}(\boldsymbol{u})$ (blue area); N_1 and N_2 are the number of data that follow distribution $\boldsymbol{P}(\boldsymbol{v})$ and the number of data that follow distribution $\boldsymbol{Q}(\boldsymbol{u})$, respectively, and we have $N_1 \gg N_2$. For convenience, in the figures, the components extracted by PCA and IOCA have not been normalized. (1) For PCA, the mean of the whole dataset is close to the mean of distribution $\boldsymbol{P}(\boldsymbol{v})$, the direction of obtained PC also mainly depends on distribution $\boldsymbol{P}(\boldsymbol{v})$, and the first PC ($\boldsymbol{b}_{PCA}$) has much higher variance than the other components. Therefore, the information of the data following $\boldsymbol{Q}(\boldsymbol{u})$ may be discarded. (2a) For IOCA, if $\boldsymbol{v}_1 \in \boldsymbol{P}(\boldsymbol{v})$ is the first input data, IOCA learns OC $\boldsymbol{b}_{1,1}$ from it. The red line is the threshold. When the other data belong to $\boldsymbol{P}(\boldsymbol{v})$ are input, as they can be well approximated by $\boldsymbol{b}_{1,1}$, no new OCA is extracted. (2b) Similarly, if $\boldsymbol{u}_1 \in \boldsymbol{Q}(\boldsymbol{u})$ is the first input data, IOCA learns OC $\boldsymbol{b}_{2,1}$ from it. Then due to the threshold policy, when $\boldsymbol{v}_2$ the first data belongs to $\boldsymbol{P}(\boldsymbol{v})$ is input, the second OC $\boldsymbol{b}_{2,2}$ is extracted. Note that in this case the learned feature subspace $S_1 = span\{\boldsymbol{b}_{1,1}, \boldsymbol{b}_{1,2}\}$ and $S_2 = span\{\boldsymbol{b}_{2,1}, \boldsymbol{b}_{2,2}\}$ are the same. Usually when the dataset is put in different orders, IOCA learns similar feature subspaces. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

The proof of Theorem 4.2 is given in Appendix B. (8) is a theoretical upper bound, we admit that sometimes it may be very loose. Note that in the learning process, the real upper bound for $\boldsymbol{x}_t$'s approximation error is $f\left(\frac{k^{(t)}}{d}\right)L_{\max}^{(t)}$, and $k^{(t)}$ is the dimension of the obtained feature subspace when $\boldsymbol{x}_t$ is processed. When the ℓ_2-norms of the input data do not differ greatly, this upper bound is relatively tight. The significance of (8) is that we may have a general upper bound for each $\boldsymbol{x}_t$'s approximation error.

Note that the OCs extracted by IOCA are different from the principal components (PCs) obtained by PCA or the independent components (ICs) obtained by ICA. IOCA selects data vectors that are fairly linear independent with each other from the online input data sequence, and the OCs are learned from these selected samples by GS process. In IOCA, the solutions are not only closed-form, but also they are obtained in a finite number of steps. On the aspect of basis extraction, we take PCA and IOCA for example and compare them in Fig. 4.

IOCA is data-dependent, and it learns data's properties through the input data stream. Naturally, we believe the data stream reflects the statistical properties of the data. Based on the proposed threshold policy, IOCA dynamically learns new components and expand the feature subspace when it is necessary. If IOCA extract components from outliers or the data has great noise, when IOCA finds that newly input date are poorly approximated by these components, new components will be learned. In this way, feature subspace is updated.

In summary, given a dataset of size N, if these N vectors are input into IOCA with different orders, for each specific input sequence, IOCA may output a corresponding orthogonal basis. Accordingly, the final k also fluctuates. However, as described above, IOCA avoids blindly expanding the feature subspace and tries to achieve a stable state. Meanwhile, for each data, IOCA is able to give a definite upper bound of the distance from it to the obtained feature subspace. This conclusion does not mean that as an online learning algorithm, IOCA will never output a non-ideal component set whatever the input sequence is, but it theoretically demonstrates the effectiveness of this algorithm to some extent.

5. Experiments

To demonstrate the effectiveness and efficiency of IOCA, we conduct experiments on synthetic dataset and some widely used real-world datasets. All the experiments in this paper are run in MATLAB R2013b on Win7, RAM 16G, CPU 3.6 GHz.

5.1. Experiments on synthetic data

5.1.1. Dimension estimation and threshold function choosing

In this section, we take experiments on synthetic datasets to evaluate IOCA's ability on dimension estimation with different threshold functions. The datasets are generated as follows. Firstly, we randomly generate D-dimensional standard orthonormal basis $\{\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_{d_0}\}$. Then, we obtain $N = 200$ data $\tilde{\boldsymbol{x}}_1, \tilde{\boldsymbol{x}}_2, \ldots, \tilde{\boldsymbol{x}}_N$ by $\tilde{\boldsymbol{x}}_i = \sum_{j=1}^{d_0} \alpha_{i,j}\boldsymbol{w}_j$, where $\alpha_{i,j}$ is generated with the standard normal distribution. Obviously, these data lay in d_0-dimensional subspace $S_0 = span\{\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_{d_0}\}$. Assume $\tilde{x}_{i,j}$ is the jth entry of $\tilde{\boldsymbol{x}}_i$, we calculate $x_m = \frac{1}{dN}\sum_{i,j}|\tilde{x}_{i,j}|$. After that, we compute the data with noise by adding Gaussian noise proportional to x_m: $\boldsymbol{x}_i = \tilde{\boldsymbol{x}}_i + \delta x_m \times randn(d, 1)$, where $\delta = 0.02$ is the noise level. Then, we employ IOCA on the dataset with threshold function $f(\omega) = \sqrt{\omega}$, $f(\omega) = \omega$ and $f(\omega) = \omega^2$, respectively, to evaluate the influence of $f(\omega)$'s choosing.

Suppose IOCA is employed on the synthetic dataset and obtains feature subspace $S = span\{\boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_k\}$ where k is the estimated intrinsic dimension of this dataset. To evaluate the quality of the obtained OCs, we measure the directional distance from d_0-dimensional subspace S_0 to d-dimensional subspace S. Suppose $\{\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_{d_0}\}$ and $\{\boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_k\}$ are subspaces S_0 and S's standard orthogonal basis, respectively, the distance from basis vector $\boldsymbol{w}_i$ to S is

$$Dist(\boldsymbol{w}_i, S) = \min_{\boldsymbol{b}\in S} \|\boldsymbol{w}_i - \boldsymbol{b}\|_2 = \|\boldsymbol{w}_i - \sum_{j=1}^{k} \boldsymbol{b}_j\boldsymbol{b}_j^\top \boldsymbol{w}_i\|_2. \tag{9}$$

As $\|\boldsymbol{w}_i\|_2 = 1$, we obtain

$$Dist(\boldsymbol{w}_i, S) = \sqrt{1 - \cos^2\beta_i} = \sqrt{1 - \sum_{j=1}^{k}(\boldsymbol{w}_i^\top \boldsymbol{b}_j)^2} \tag{10}$$

β_i denotes the angle between $\boldsymbol{w}_i$ and S. Based on (10), the distance from S_0 to S can be defined as (Wang, Wang, & Feng, 2006):

$$\begin{aligned} Dist(S_0, S) &= \sqrt{\sum_{i=1}^{d_0} Dist^2(\boldsymbol{w}_i, S)} = \sqrt{d_0 - \sum_{i=1}^{\min(d_0,k)} \cos^2\theta_i} \\ &= \sqrt{d_0 - \sum_{i=1}^{d_0}\sum_{j=1}^{k}(\boldsymbol{w}_i^\top \boldsymbol{b}_j)^2} \end{aligned} \tag{11}$$

$0 \le \theta_1 \le \theta_2 \le \cdots \le \theta_{\min(d_0,k)} \le \frac{\pi}{2}$ are the principal angles between subspaces S_0 and S and they are defined as (Hotelling, 1935)

$$\cos\theta_i = \max_{\boldsymbol{w}_i\in S_0,\ \boldsymbol{b}_i\in S} \boldsymbol{w}_i^\top \boldsymbol{b}_i, \quad i = 1, 2, \ldots, \min(d_0, k) \tag{12}$$

Table 1
The results of experiments on synthetic data $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N$. $f(\omega)$ is the employed threshold function. The results are obtained after 10 repeats.

	$f(\omega)=\sqrt{\omega}$		$f(\omega)=\omega$		$f(\omega)=\omega^2$	
	k	$Dist^2(S_0, S)$	k	$Dist^2(S_0, S)$	k	$Dist^2(S_0, S)$
$d_0 = 10\ d = 30$	9.7	0.4057	10.0	0.0153	10.5	0.0176
$d_0 = 10\ d = 100$	10.0	0.0025	10.7	0.0203	16.7	0.0025

Table 2
The results of experiments on synthetic data $\boldsymbol{x}_0, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N$. $\boldsymbol{x}_0$ is the outlier and it is firstly input. λ is a parameter in outlier generating. $f(\omega)$ is the employed threshold function. The results are obtained after 10 repeats.

		$f(\omega)=\sqrt{\omega}$		$f(\omega)=\omega$		$f(\omega)=\omega^2$	
		k	$Dist^2(S_0, S)$	k	$Dist^2(S_0, S)$	k	$Dist^2(S_0, S)$
$d_0 = 10\ d = 30$	$\lambda = 0.5$	10.6	0.4045	11.0	0.0115	11.3	0.0211
	$\lambda = 1$	10.6	0.4044	11.0	0.0118	11.3	0.0211
	$\lambda = 2$	9.0	2.0021	11.0	0.0089	11.0	0.0274
	$\lambda = 3$	6.0	5.0008	9.9	1.1032	11.0	0.0214
	$\lambda = 4$	4.8	6.2005	8.7	2.3021	11.0	0.0178
	$\lambda = 5$	3.6	7.4003	7.8	3.2018	11.0	0.0108
$d_0 = 10\ d = 100$	$\lambda = 0.5$	11.0	0.0194	11.6	0.0218	16.8	0.0029
	$\lambda = 1$	11.0	0.0194	11.6	0.0218	16.8	0.0029
	$\lambda = 2$	10.9	0.0105	11.5	0.0263	15.7	0.0033
	$\lambda = 3$	10.5	0.5063	11.0	0.0282	14.8	0.0041
	$\lambda = 4$	8.5	2.5025	11.0	0.0235	13.5	0.0042
	$\lambda = 5$	6.6	4.5015	11.0	0.0139	13.3	0.0073

Table 3
The results of experiments on synthetic data $\boldsymbol{x}_0, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N$. The $N+1$ data are input in random order. λ is a parameter in outlier generating. $f(\omega)$ is the employed threshold function. The results are obtained after 100 repeats.

		$f(\omega)=\sqrt{\omega}$		$f(\omega)=\omega$		$f(\omega)=\omega^2$	
		k	$Dist^2(S_0, S)$	k	$Dist^2(S_0, S)$	k	$Dist^2(S_0, S)$
$d_0 = 10\ d = 30$	$\lambda = 1$	10.50	0.3555	11.00	0.0150	11.37	0.0192
	$\lambda = 2$	10.28	0.7252	11.00	0.0150	11.44	0.0211
	$\lambda = 5$	10.23	0.7849	10.78	0.0258	11.37	0.0214
	$\lambda = 10$	9.92	1.0848	10.72	0.0295	11.37	0.0390
$d_0 = 10\ d = 100$	$\lambda = 1$	11.00	0.0193	11.62	0.0174	17.09	0.0027
	$\lambda = 2$	11.00	0.0198	11.59	0.0193	17.09	0.0028
	$\lambda = 5$	11.92	0.1003	11.55	0.0210	16.99	0.0029
	$\lambda = 10$	10.87	0.1497	11.49	0.1085	16.61	0.0038

subject to

$$\boldsymbol{w}_i^\top \boldsymbol{w}_j = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases} \quad \text{and} \quad \boldsymbol{b}_i^\top \boldsymbol{b}_j = \begin{cases} 1, & i = j \\ 0, & i \neq j. \end{cases} \tag{13}$$

Note that given subspace S_0 and S, the principal angles between them are uniquely defined: assume $\{\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_{d_0}\}$ and $\{\tilde{\boldsymbol{w}}_1, \tilde{\boldsymbol{w}}_2, \ldots, \tilde{\boldsymbol{w}}_{d_0}\}$ are two arbitrary standard orthogonal basis of S_0, $\{\boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_k\}$ and $\{\tilde{\boldsymbol{b}}_1, \tilde{\boldsymbol{b}}_2, \ldots, \tilde{\boldsymbol{b}}_k\}$ are two arbitrary standard orthogonal basis of S_0, we have $\sum_{i=1}^{d_0}\sum_{j=1}^{k}(\boldsymbol{w}_i^\top \boldsymbol{b}_j)^2 = \sum_{i=1}^{d_0}\sum_{j=1}^{k}(\tilde{\boldsymbol{w}}_i^\top \tilde{\boldsymbol{b}}_j)^2 = \sum_{i=1}^{\min(d_0,k)} \cos^2\theta_i$. However, in most cases, $\cos^2\theta_i \neq \sum_{j=1}^{k}(\boldsymbol{b}_j^\top \boldsymbol{w}_i)^2$.

In this section, we employ $Dist^2(S_0, S)$ (the square of the distance) to measure the similarity between S_0 and S. Obviously, when d is fixed, the smaller $Dist^2(S_0, S)$ is, the better the quality of the learned S is. When $Dist^2(S_0, S) = 0$, we have $S_0 \subseteq S$.

The experimental results are reported in Table 1. They are the average of 10 replicates. These results are consistent with the conclusion that the stricter the employed threshold function $f(\omega)$ is, the smaller d the dimension of the subspace obtained by OCA is. No matter which $f(\omega)$ is employed for threshold function, IOCA estimates a larger dimension in the case $d = 100$ than in the case $d = 30$. When $f(\omega) = \sqrt{\omega}$ or $f(\omega) = \omega$ is employed, the estimated k is similar with the true intrinsic dimension $d_0 = 10$. However, when $d = 100$ and $f(\omega) = \omega^2$ is employed, a too large k is obtained. It implies that $f(\omega) = \omega^2$ may be a too loose threshold function when D is greatly larger than d_0. In all these cases, the obtained $Dist^2(S_0, S)$ is very close to 0, this means the synthetic data distribute on or near the obtained S.

To evaluate IOCA's performance when outlier exists, we add an extra vector $\boldsymbol{x_0}$ to the previously generated dataset: $\boldsymbol{x}_0 = \lambda \|randn(d, 1)\|_2 \boldsymbol{w}_0$, $\boldsymbol{w}_0$ is a vector that is orthogonal to S_0, λ is a parameter manually set and it varies in the following experiments. Then we employ IOCA on the $N+1$ data in two conditions: the outlier $\boldsymbol{x}_0$ is firstly input; the $N+1$ data are input in random order. The results are listed in Tables 2 and 3, respectively.

When outlier $\boldsymbol{x}_0$ is firstly input, as the first OC is directly extracted from $\boldsymbol{x}_0$, $k = d_0 + 1 = 11$ is the optimal estimation result IOCA can obtain. In Table 2 when length λ is equal or less than 1, $\boldsymbol{x}_0$ has little influence on IOCA's performance. When λ is large ($\lambda = 5$), the threshold may become too strict and IOCA may estimate an improper small k. Especially, at this time, if $f(\omega) = \sqrt{\omega}$ is employed, the result is poor. However, as reported in Table 2, when $\lambda = 2$, IOCA performance well with $f(\omega) = \sqrt{\omega}$ and $f(\omega) = \omega$. We can also find that with the fixed d_0, the smaller $\frac{d_0}{d}$ is, the easier for IOCA to find all the basis generated the dataset.

Note that inputting outlier $\boldsymbol{x}_0$ firstly is the worst case for IOCA. In online environment, on one knows the input order of data. Thus, we take experiments on the situation that the $N+1$ data are input in random order. The results in Table 3 demonstrate that in these cases, outlier $\boldsymbol{x}_0$ does not cause great trouble, although to perfectly represent $\boldsymbol{x}_0$, an additional OC should be extracted.

The learning process of IOCA is a trade-off: on one hand, the extracted feature subspace tries to contain all the original data; on the other hand, the dimension of the feature subspace should be

Table 4
The results obtained on synthetic dataset in which each data is a d-dimensional Gaussian vector with independent standard normal entries.

	Time (s)	k	k/d
$d = 2000$	194.8	1259.1	0.6259
$d = 5000$	1211.6	3124.9	0.6250

the smaller the better. According to the performance on dimension estimation, we believe selecting the above discussed $f(\omega) = \omega$ as the default threshold function is a not bad choice and it is employed in the following experiments. Note that for different applications, users can also set $f(\omega)$ by themselves. In real world applications, for convenience, users may set the threshold function as $f(\omega) = \omega^c$, c is a positive parameter. If they want the dimension of the obtained feature space to be higher, they may set c more than 1; if they want the dimension to be lower, they may set c less than 1.

5.1.2. Verification of Corollary 4.1

In Section 4, theoretical analysis is given to the upper bound of the possibility for IOCA to extract new OC under some assumptions. To verify the conclusion about "Golden Ratio", we designed the following experiments.

At first, we generate dataset X in which each data is a d-dimensional Gaussian vector with independent standard normal entries and take experiment on it. Here N is the size of X, the entries of each $\boldsymbol{x_i}$ follow the normal distribution $\mathcal{N}(0, 1)$. Then experiments are taken on X.

Setting $N = 100\,000$, we execute IOCA on two cases: $d = 2000$ and $d = 5000$. Each case is executed 10 times. The average results are listed in Table 4, where "time" is IOCA's average running time, k is the final target dimension IOCA obtained and k/d is the corresponding compression ratio.

The results in Table 4 show that in both cases the final compression ratio is a little larger than the "Golden Ratio". In addition, with the same N, the final compression ratio is obtained when $d = 5000$ is closer to the "Golden Ratio" than that obtained when $d = 2000$. As time complexity of IOCA is $O(Ndk)$, we can come to the conclusion that in these cases the running time of IOCA is proportional to d^2. The practical results support this conclusion: the time cost of the latter case is about 6.2 times as that of the former case. Therefore, these results are consistent with the conclusions given in Section 4.

5.2. Experiments on real-world data

Next, to evaluate the performance of IOCA in real-world applications, we conduct experiments under 12 real-world datasets. They are widely-used standard datasets and in particular, six of them are large-scale: their sample sizes are larger than 5000. In the following experiments, each dataset is partitioned into two disjoint subsets: the training set and the testing set. The data from the training set are employed for OC extraction and dimensionality reduction. The details of these datasets are shown in Table 5.

For the data vectors in Hill dataset, normalization operation is taken on them before they are input. The data of the other datasets are input directly without any preprocessing operations. In all the experiments, IOCA and the other methods are employed on the same data.

To validate the effectiveness and efficiency of IOCA, in this section, we compare IOCA with several typical dimensionality reduction methods: PCA (Jolliffe, 1986), random orthogonal projection (ROP), FastICA (Hyvarinen, 1999) (a fast version of ICA), locality preserving projections (LPP) (He & Niyogi, 2005), incremental PCA (IPCA) (Artač et al., 2002) and candid covariance-free incremental PCA (CCIPCA) (Weng et al., 2003). Except IPCA and CCIPCA, the other four methods are non-incremental methods. As described above, IOCA is designed specially to meet the need of online component extraction. It is able to keep learning continually and output the result of dimensionality reduction at any required time. Therefore, IOCA can be employed in the applications that batch methods are not suitable for. In the experiments, for the purpose of fair comparison, the non-incremental methods calculate their output after the entire training sets have been input. On the other hand, to simulate the online environment, for IPCA and CCIPCA, only two samples are available in the beginning, and then the other samples in the training set are input one by one. IPCA and CCIPCA update their learned components each time a single sample is input. The final reported results of these incremental methods are their average results obtained after 10 executions. In each execution, the input order of the data sequence is randomly generated.

Firstly, we compare the recognition rate (RR) of different methods. For each method, after all the original data are transformed into low-dimensional feature subspace learned from the training set, 1-nearest neighbor classifier (NNC) is employed to classify the testing data.

Moreover, to testify whether the learned components can accurately preserve the structure of the original data, we employ the mean relative reconstruction cost E to measure the performance of these methods. Here, we define it as

$$E = \frac{1}{N}\sum_{t=1}^{N}\frac{\|\boldsymbol{r}_t\|_2}{\|\boldsymbol{x}_t\|_2} = \frac{1}{N}\sum_{t=1}^{N}\frac{\|\boldsymbol{x}_t - \tilde{\boldsymbol{x}}_t\|_2}{\|\boldsymbol{x}_t\|_2}. \tag{14}$$

Here, $\boldsymbol{x}_t$ is the original data and $\|\boldsymbol{r}_t\|_2$ is the reconstruction cost, $\tilde{\boldsymbol{x}}_t$ is its approximation that is reconstructed by the learned components. For IOCA, as the dimension of the learned feature subspace increases, $\|\boldsymbol{r}_t\|_2$ decreases monotonically.

While the mutual orthogonality of the obtained basis $\{\boldsymbol{b_i}\}_{i=1}^{k}$ is ensured, E can be rewritten as

$$E = \frac{1}{N}\sum_{t=1}^{N}\frac{\left\|\boldsymbol{x}_t - \sum_{j=1}^{k}\boldsymbol{b}_j\boldsymbol{b}_j^{\top}\boldsymbol{x}_t\right\|_2}{\|\boldsymbol{x}_t\|_2}. \tag{15}$$

Table 5
The employed real-world datasets.

Dataset	Features	Training samples	Testing samples	Classes
Hill (Blake & Merz, 1996)	100	606	606	2
Ionosphere (Blake & Merz, 1996)	34	100	251	2
Musk (Blake & Merz, 1996)	166	476	6 598	2
OptDigit (Blake & Merz, 1996)	64	3 823	1 797	10
Sonar (Blake & Merz, 1996)	60	104	104	2
Waveform (Blake & Merz, 1996)	21	300	5 000	3
IJCNN1 (Prokhorov, 2001)	22	49 990	91 701	2
ISOLET (Blake & Merz, 1996)	617	6 238	1 559	26
MNIST (Yann et al., 1998)	784	60 000	10 000	10
Protein (Wang, 2002)	357	17 766	6 621	3
SensIT vehicle (Duarte & Hu, 2004)	50	78 823	19 705	3
USPS (Hull, 1994)	256	7 291	2 007	10

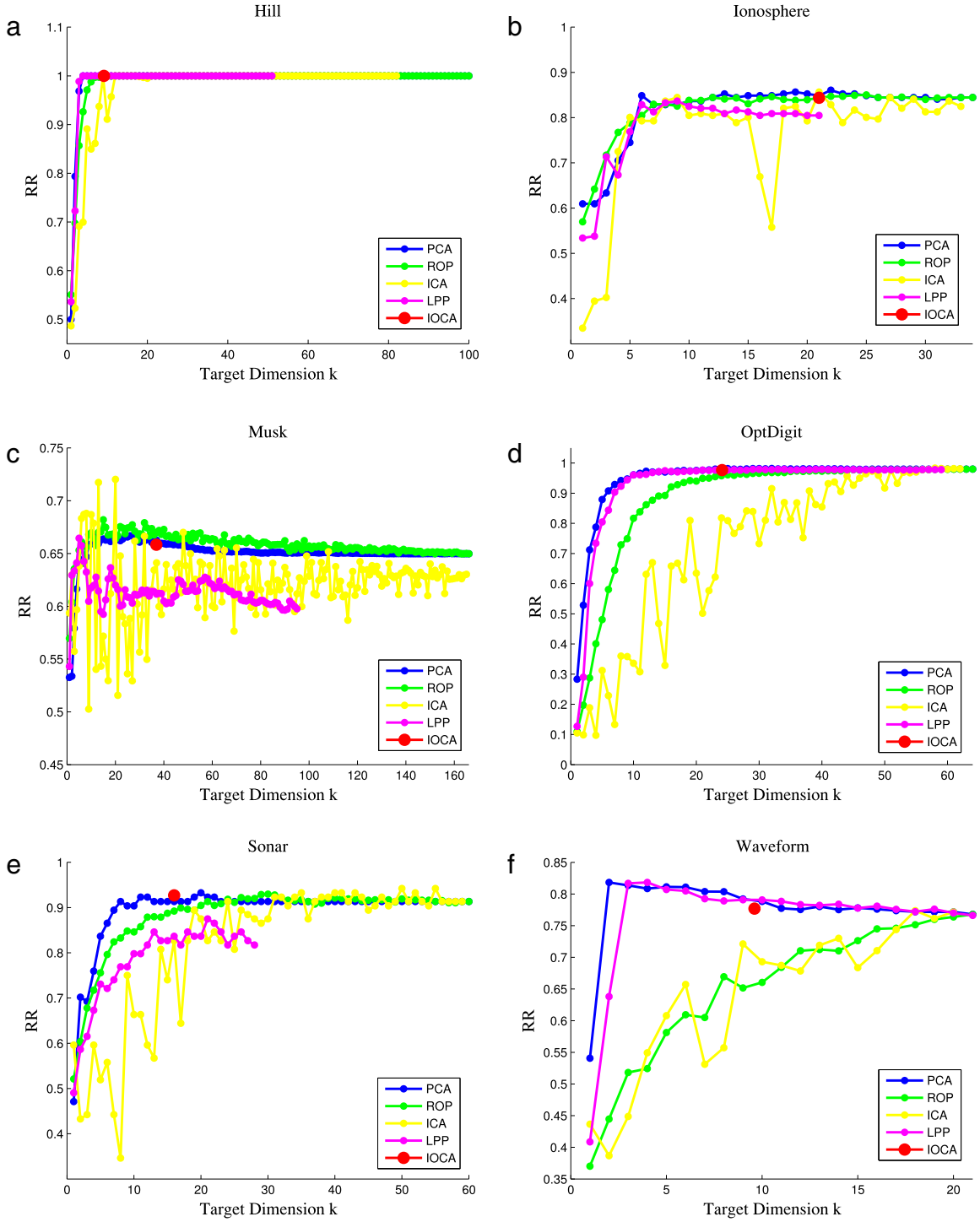

Fig. 5. The comparison between IOCA and non-incremental learning methods in recognition rate (RR) on (a) Hill, (b) Ionosphere, (c) Musk, (d) OptDigit, (e) Sonar, (f) Waveform datasets.

For IOCA and ROP, we directly employ (15) in their mean relative reconstruction cost computation. For PCA and IPCA, (15) is employed after the data are made to be zero-mean. Though the basis obtained by CCIPCA usually is not ensured to be standard orthogonal in practice, the basic assumption taken by CCIPCA is that the basis vectors can converge approximately to PCs during incremental learning. Theoretically speaking, PCs are standard orthogonal and as the number of extracted PCs increases, (15) will monotonically decrease and its range is [0, 1]. As a result of that, after the data are made to be zero-mean, we also employ (15) as CCIPCA's mean relative reconstruction cost function.

Furthermore, the execute time of the incremental methods IOCA, IPCA and CCIPCA is reported. Obviously, low time cost is an excellent property for online learning.

5.2.1. Comparisons with non-incremental methods

Figs. 5–8 report the result comparisons between IOCA and the typical non-incremental methods in all these 12 datasets: the abscissa stands for the value of target dimension k and the vertical coordinate stands for the corresponding RR and E, respectively. As IOCA is able to automatically determine the target dimension, we only portray a point to show its performance. For

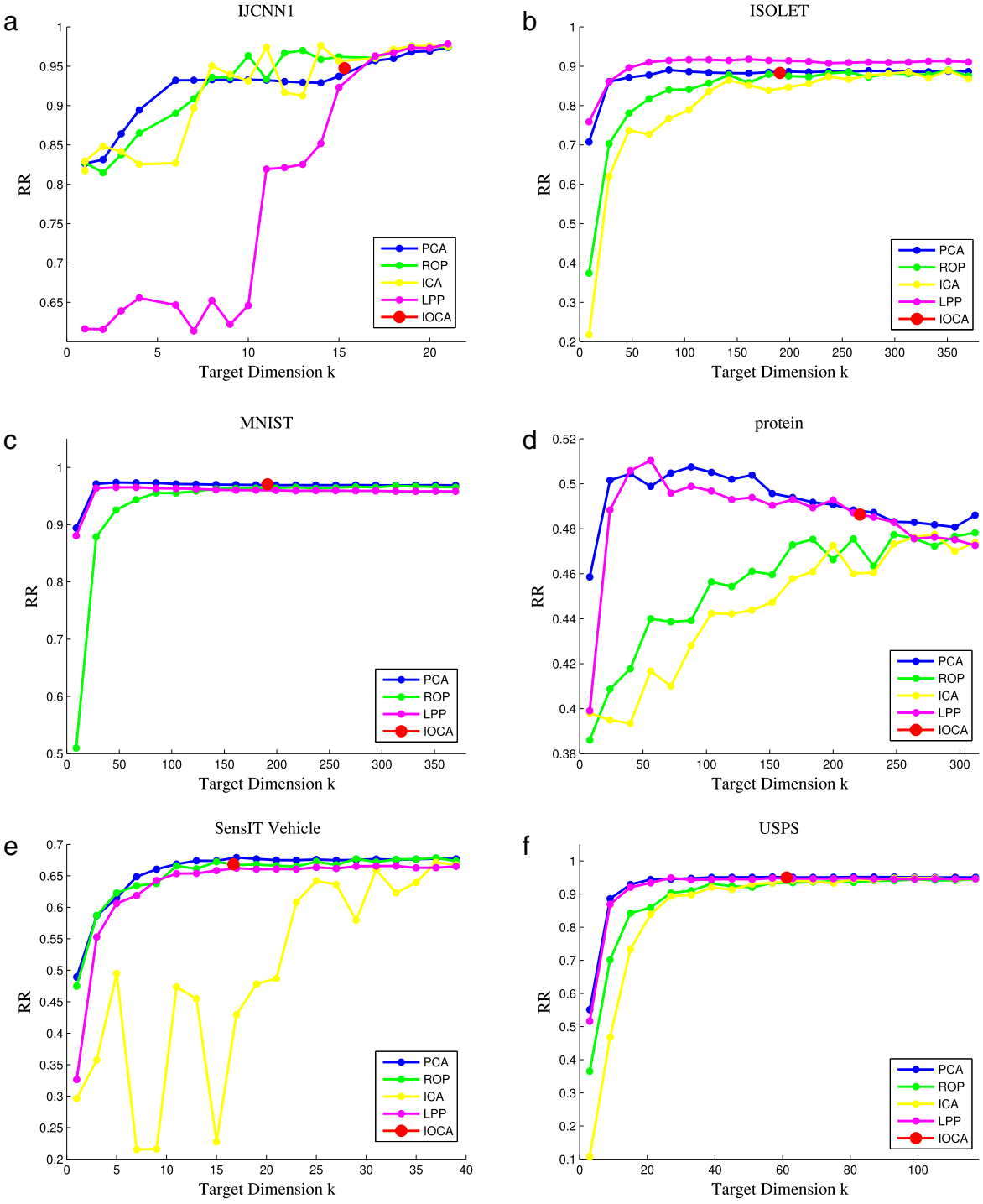

Fig. 6. The comparison between IOCA and non-incremental learning methods in recognition rate (RR) on (a) IJCNN1, (b) ISOLET, (c) MNIST, (d) protein, (e) SensIT vehicle, (f) USPS datasets. We only report the available results.

the other methods, their results are calculated with different predetermined k and represented by curved lines. Note that FastICA estimates the maximum number of ICs before IC calculation; LPP has to employ a series of operations to discard unreliable results while solving the generalized eigenproblem. As a result of that, sometimes the dimension of the available feature subspace obtained by these two methods is much lower than the original dimension d. In MNIST, FastICA fails to converge. Therefore, we only report the available results.

From Figs. 5 and 6, we find that for these datasets, though the RR of IOCA is a little worse than the optimal RR of the other methods, in most cases, the gap between them is not obvious. Here, we have to emphasize that IOCA is not designed for classification tasks, its main purpose is to achieve high-speed incremental dimensionality reduction in an online environment while automatically determining the target dimension k. The figures show that the result points of IOCA never appears with an improper small k.

The results of E reported in Figs. 7 and 8 indicate that compared with the listed non-incremental methods, with the same setting of k, IOCA achieves the smallest E in five datasets. In most cases, E of PCA and IOCA is much smaller than that of ROP. The results

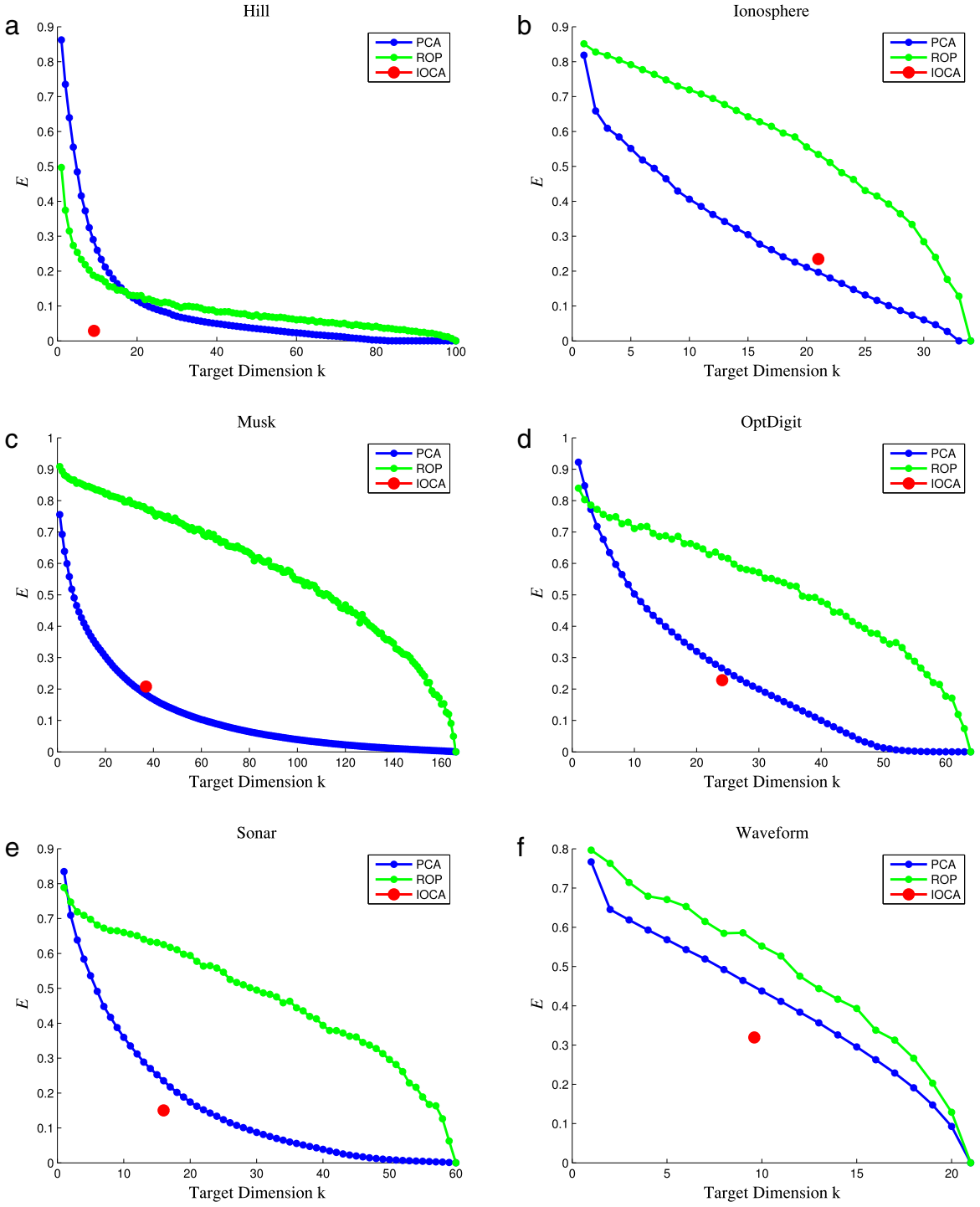

Fig. 7. The comparison between IOCA and non-incremental learning methods in their mean relative reconstruction error (E) on (a) Hill, (b) Ionosphere, (c) Musk, (d) OptDigit, (e) Sonar, (f) Waveform datasets.

demonstrate that though the strategy IOCA employed in target dimension estimation is very simple, it is indeed effective in approximating the original data. The orthogonality of the learned basis also ensures the low reconstruction cost of IOCA. In the large-scale datasets, PCA has the smallest E, that may due to the strategy PCA employed that only reserve the k directions along which the data variations are maximum.

Furthermore, from Figs. 5–8, we can also find the following facts.

Firstly, in most cases, PCA performances better than ROP when k is not large. The reason may be that PCA is a method highly data-dependent; its objective function ensures the low approximation error. On the other hand, as discussed in Section 4, ROP is data-independent and that makes it almost impossible to achieve high performance when the target dimensional k is set to be too small. The performance of FastICA fluctuates greatly in some datasets, that may be because of the random initialization of the algorithm.

Secondly, in classification tasks, PCA is not always able to achieve the optimal accuracy: in Musk dataset, ICA has the highest accuracy; in ISOLET and protein, the best result is achieved by LPP; in some dataset, with certain k, ROP performs better than PCA. FastICA is designed for blind source separation, and it is not

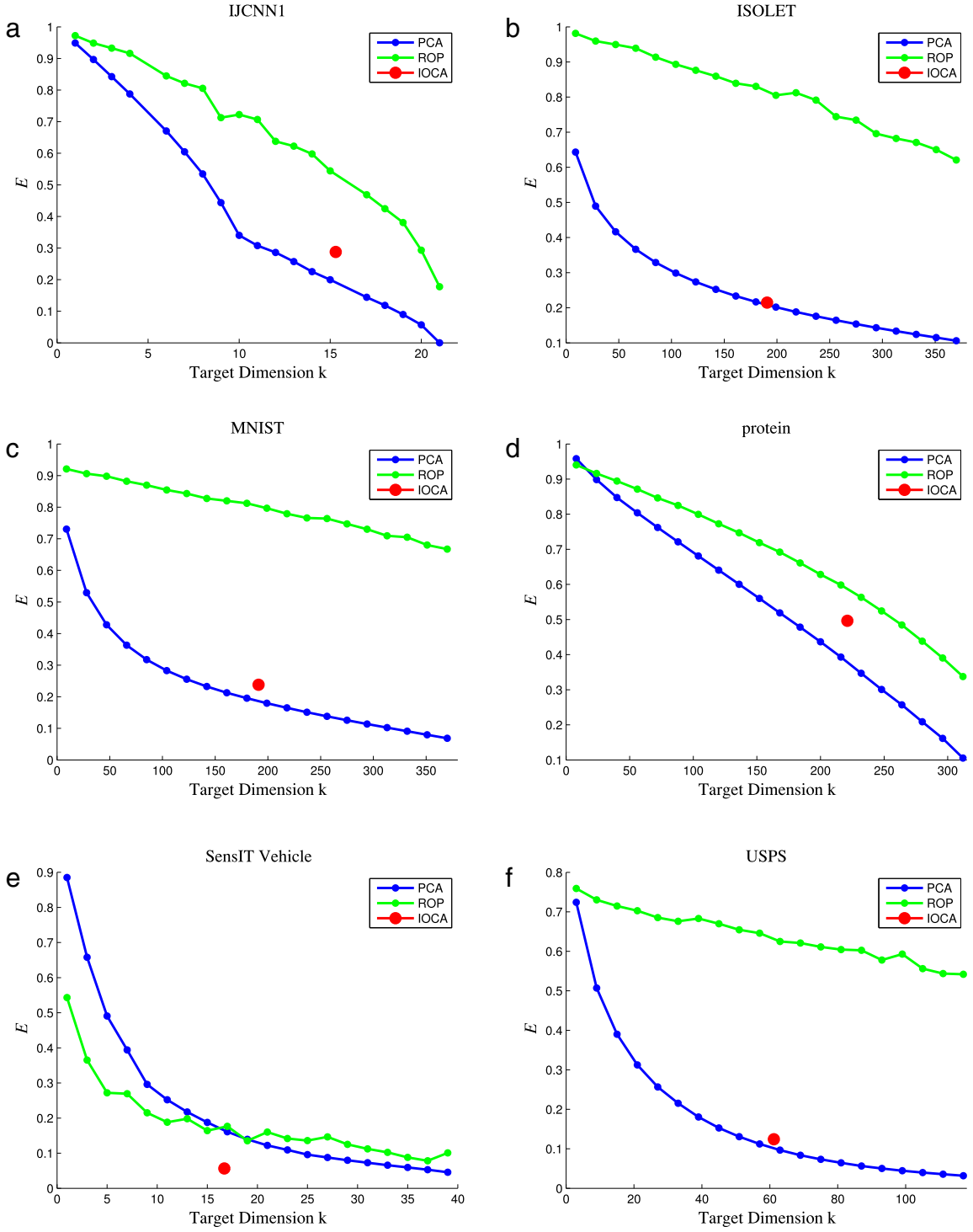

Fig. 8. The comparison between IOCA and non-incremental learning methods in their mean relative reconstruction error (E) on (a) IJCNN1, (b) ISOLET, (c) MNIST, (d) protein, (e) SensIT vehicle, (f) USPS datasets.

always suitable for dimensionality reduction tasks. We admit that the other methods have advantages over the IOCA in some specific applications. However, as a universal method, in all these 12 datasets, with the same k, the performance of IOCA is competitive.

Thirdly, the results confirm that without a proper setting of k, it is difficult for the dimensionality reduction methods to preserve the sketch of the actual feature subspace properly. How to estimate the intrinsic dimension of an arbitrarily given dataset is still an open problem. IOCA tries to solve this problem with a simple strategy based on linear independence measure. We directly employ IOCA on these datasets that may have different characteristics. The results prove that the k automatically determined by IOCA is not improper.

5.2.2. Comparisons with incremental methods

Then, two incremental methods IPCA and CCIPCA are employed to compare with IOCA. As described above, all these methods start with only two available samples and they keep learning from the online input data. Each experiment has run 10 times with random order input sequences.

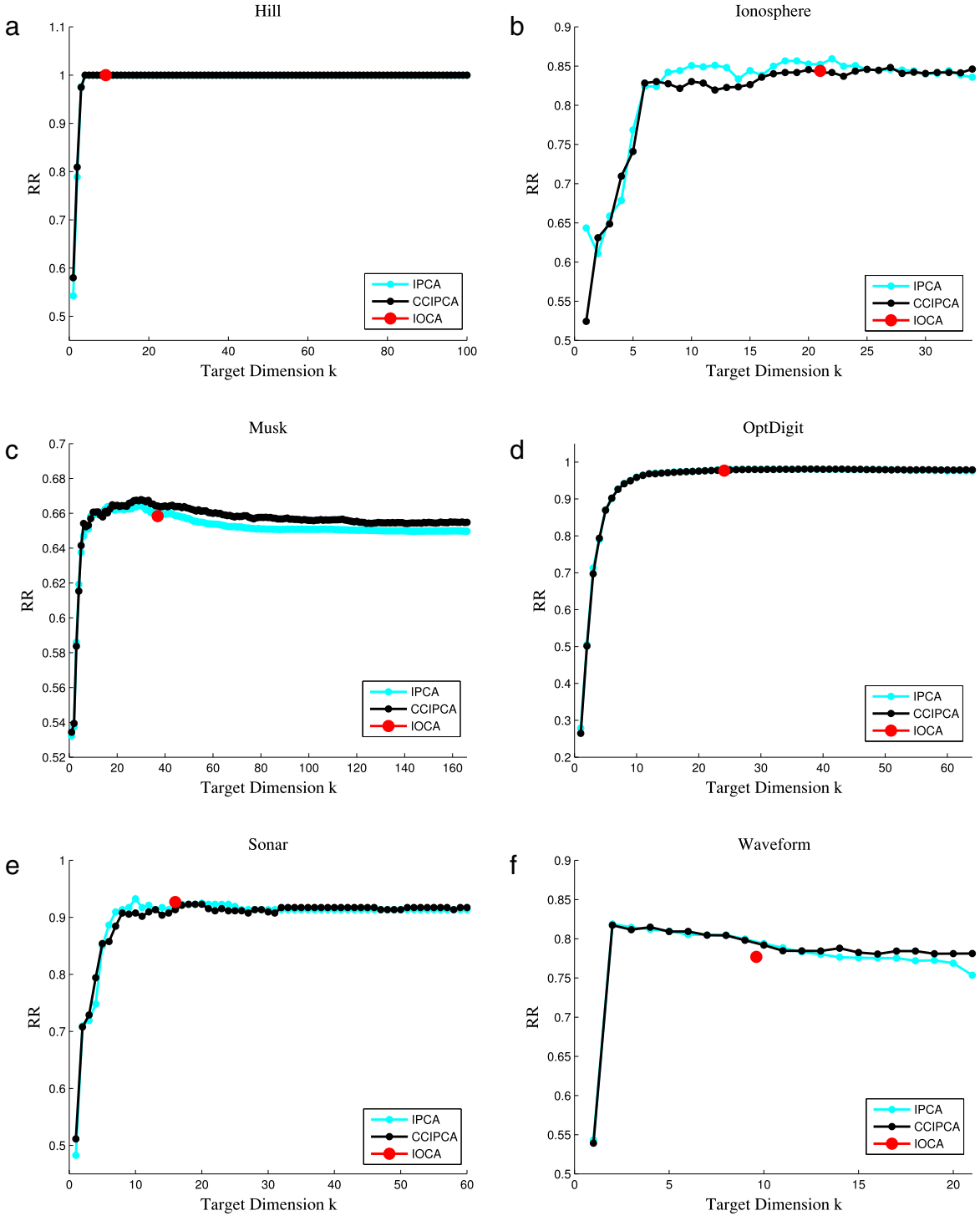

Fig. 9. The comparison between IOCA, IPCA and CCIPCA in recognition rate (RR) on (a) Hill, (b) Ionosphere, (c) Musk, (d) OptDigit, (e) Sonar, (f) Waveform datasets.

For the first six datasets (their scale is not large), the recognition rate (RR) and mean relative reconstruction cost E are represented in Figs. 9 and 10, respectively.

The results of RR in Fig. 9 indicate that IOCA is a competitive algorithm compared with IPCA and CCIPCA in classification tasks. Though the accuracy of IOCA is a little worse than the optimal performance IPCA and CCIPCA can achieve, when IPCA and CCIPCA employ the same k that learned by IOCA, they achieve almost the same RR. Automatically determining the target dimension k is the obvious advantage of IOCA. For IPCA and CCIPCA, though they may update and adjust the learned components during incremental learning process, it is still impossible for them to know how to set k properly.

As shown in Fig. 10, the performance of IOCA in data reconstruction is competitive: IOCA has smaller E than IPCA and CCIPCA in four datasets when their target dimension k is the same.

In Fig. 10, we can also find an interesting phenomenon. For IPCA and CCIPCA, with k increases, at first their reconstruction costs keep decreasing; however, when an inflection point is reached, their reconstruction costs start to increase. CCIPCA reaches its inflection point much earlier than IPCA. On the other hand, IPCA

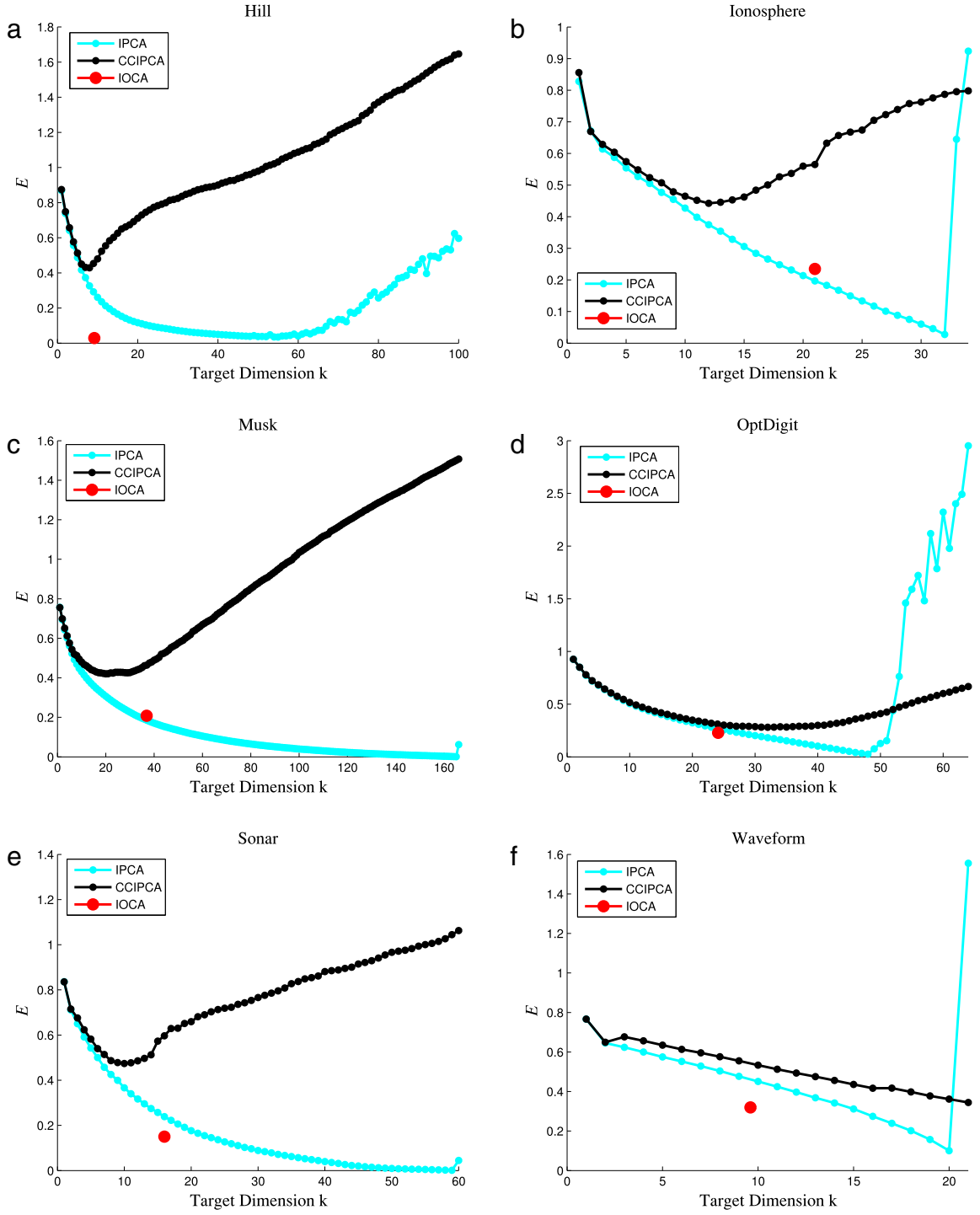

Fig. 10. The comparison between IOCA, IPCA and CCIPCA in their mean relative reconstruction error (E) on (a) Hill, (b) Ionosphere, (c) Musk, (d) OptDigit, (e) Sonar, (f) Waveform datasets.

does not reach its inflection point until the value of k is close to d. Then, IPCA's reconstruction error increases rapidly and greatly.

Theoretically, as k increases, E will decrease monotonically (just like the results of PCA and ROP shown in Figs. 7 and 8). However, the experimental results of IPCA and CCIPCA appear to contradict this conclusion. There is one reason for this contradiction: in these cases, the mutual orthogonality of the extracted components is destroyed.

For CCIPCA, it adopts the assumption that the mutual orthogonality of the learned components is maintained after these principal components have been adjusted. Therefore, CCIPCA only focuses on the orthogonality between the newly input data vector and the k learned components. This strategy helps CCIPCA reduce the number of dot product from $\frac{k(k+1)}{2}$ to k during each principal components update operation. Though in Zhang and Weng (2011), the researchers have theoretically proved that when $N \to \infty$, the vectors obtained by CCIPCA converge to the PCs obtained by PCA, with probability 1. However, experiment results show that when N is finite and k is large, the algorithm may not reach the convergence and CCIPCA may fail to obtain standard orthogonal components.

For IPCA, the problem of orthogonality destruction is due to the round-off error. Assume the columns of $\boldsymbol{B}$ are the k PCs learned

Table 6
The comparison between IOCA and incremental algorithms in recognition rate (RR), mean relative reconstruction error (E) and execution time on six large-scale datasets. The best performance is emphasized with figures in bold typeface.

	Dataset	IJCNN1	ISOLET	MNIST
	k	15.30 ± 0.67	190.60 ± 1.78	191.10 ± 2.13
IOCA	RR	$94.75 \pm 1.32\%$	$88.29 \pm 0.28\%$	$\mathbf{97.05 \pm 0.00}\%$
	E	0.28 ± 0.02	$\mathbf{0.21 \pm 0.00}$	0.24 ± 0.00
	Time (s)	**0.29**	**1.02**	**11.25**
IPCA	RR	$93.84 \pm 1.00\%$	$\mathbf{88.42 \pm 0.18}\%$	$96.93 \pm 0.00\%$
	E	$\mathbf{0.19 \pm 0.02}$	$\mathbf{0.21 \pm 0.00}$	$\mathbf{0.19 \pm 0.00}$
	Time (s)	103.38	178.14	1768.65
CCIPCA	RR	$\mathbf{95.05 \pm 0.93}\%$	$88.32 \pm 0.52\%$	$96.95 \pm 0.00\%$
	E	0.21 ± 0.02	0.59 ± 0.06	0.34 ± 0.05
	time	4.55s	21.20s	205.83s
	Dataset	Protein	SensIT vehicle	USPS
	k	221.10 ± 0.74	16.70 ± 0.48	61.20 ± 0.92
IOCA	RR	$48.62 \pm 0.55\%$	$66.83 \pm 0.29\%$	$94.96 \pm 0.00\%$
	E	0.50 ± 0.00	$\mathbf{0.06 \pm 0.01}$	0.12 ± 0.00
	Time (s)	**1.90**	**0.47**	**0.18**
IPCA	RR	$48.85 \pm 0.27\%$	$\mathbf{67.57 \pm 0.12}\%$	$\mathbf{95.06 \pm 0.00}\%$
	E	$\mathbf{0.39 \pm 0.00}$	0.17 ± 0.01	$\mathbf{0.10 \pm 0.00}$
	Time (s)	675.59	172.64	45.23
CCIPCA	RR	$\mathbf{49.70 \pm 0.51}\%$	$67.52 \pm 0.22\%$	$94.75 \pm 0.20\%$
	E	0.54 ± 0.00	0.24 ± 0.07	0.42 ± 0.09
	Time (s)	46.94	8.38	4.24

by IPCA. When a new vector $\boldsymbol{x}$ is input and average-removed, the orthogonal residual vector $\boldsymbol{r} = \boldsymbol{x} - \boldsymbol{B}\boldsymbol{B}^\top \boldsymbol{x}$ is calculated. If $\|\boldsymbol{r}\|_2 > 0$, $\boldsymbol{r}' = \frac{\boldsymbol{r}}{\|\boldsymbol{r}\|_2}$ is obtained. Then, the new PCs are calculated by $\boldsymbol{B}' = [\boldsymbol{B}, \boldsymbol{r}']\boldsymbol{R}$, $\boldsymbol{R}$ is a $(k+1) \times (k+1)$ size rotation matrix obtained by solving eigenproblem. Theoretically, the algorithm successfully maintains the mutual orthogonality of the extracted components in updating process. However, in practice, when $\boldsymbol{r}$ is a $\boldsymbol{0}$ vector or its ℓ_2-norm is very small, sometimes due to the round-off error, the normalized $\boldsymbol{r}'$ is not orthogonal with the existing components. As the matrix rotation operation is executed each time a new sample is input, the orthogonality of the learned components is gradually destroyed. After a period of time, the orthogonality has been completely dominated. This problem occurs with high possibility when k is set to be close to d. Once this problem occurs, the reconstruction error of IPCA will greatly increase and sometimes the RR of IPCA drops accordingly. Nonetheless, this problem can be easily addressed. To prevent round-off error, a threshold may be employed to discard the residual vectors whose ℓ_2-norms are too small. This is the very strategy IOCA adopted. It is simple, but effective.

To evaluate the performance of these incremental learning methods in an online environment, we also compare their running time in Fig. 11. The running time of IPCA and CCIPCA is nearly proportional to the target dimension k. IOCA is much faster than IPCA and CCIPCA.

Then, we compare IOCA with IPCA and CCIPCA on six large-scale datasets. Each experiment has been implemented 10 times. The target dimension k determined by IOCA is employed by IPCA and CCIPCA. The results (k, RR, E and execution time) are summarized in Table 6. For k, RR, E, we give their average values and standard deviations.

Different input sequences of the same dataset may cause the fluctuation of the values of k that are automatically obtained by IOCA. However, the results show that, even the size of input samples is large, the k obtained by IOCA only fluctuates in a small range. The practical results in RR and E also illustrate that the stability of IOCA is acceptable.

According to the results in Table 6, IOCA has huge advantage in time cost. It is nearly 20 times faster than CCIPCA and in most cases, CCIPCA is almost 10 times faster than IPCA. Meanwhile, the performance of IOCA in RR and E is only a little worse than that of IPCA or CCIPCA. The results also implies that when the predetermined k is large, even the size of samples is large, it is still difficult for CCIPCA to obtain orthogonal PCs.

In sum, by comparing IOCA with several typical linear dimensionality reduction methods, we can declare that IOCA is a competitive method. It employs a simple but effective adaptive threshold policy to estimate the target dimension and determine the feature subspace at the same time. The compression ratio of IOCA ($\frac{k}{d}$) is acceptable. Note that even with the same k, IOCA is much faster than IPCA and CCIPCA. This is a decisive advantage for IOCA in online large-scale data processing. However, it is unfair to only mention the advantages of IOCA over the other methods. PCA obtain PCs that are ranked according to the data variance; ICA learns ICs that are statistically independent from each other under non-Gaussian assumption; LPP tries to preserve the neighborhood structure of the dataset. Each typical method has important characteristics for many applications. IOCA is not intended to replace the existing methods. However, as a convenient and high-speed OC extraction method, we believe IOCA is a good choice for online data processing, especially when little prior knowledge is available.

6. Conclusion

In this paper, we propose an incremental method named IOCA for online dimensionality reduction tasks. With low computational cost, IOCA achieves high-speed learning. By proposing an adaptive threshold policy, the target dimension is automatically estimated and the numerical orthogonality of the learned OCs is ensured. In the experiments, we compare IOCA with several typical linear dimensionality reduction methods. Results show that IOCA is a competitive method. Although the performance of the other methods sometimes is slightly better in certain aspect, IOCA fulfills good compromise among component quality, classification power, and storage efficiency. When considering the properties needed in online learning, we declare that IOCA's performance is outstanding.

Note that IOCA is not exclusively designed for some specific applications. Our principal aim is to propose a universal approach which can be employed efficiently and effectively in various fields. We believe that the concise algorithm, automatic threshold policy and low computational cost will help IOCA to be successful in online learning.

In the future, IOCA method can be improved in the following two aspects. In this paper, once the OCs have been extracted by IOCA, they cannot be adjusted any more. In some applications, we can make IOCA to have the ability to adjust the extracted OCs during learning. To cope with the nonlinear dimensionality reduction problem, we can also combine the power of kernel technique with IOCA.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant No. 61373001 and 61375064), Foundation of Jiangsu NSF (Grant No. BK20131279).

Appendix A. Proof of Theorem 4.1

Proof. We assume that before $\boldsymbol{x}_t$ is input, the k OCs extracted by IOCA are $\boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_k$. IOCA accepts the candidate OC extracted from $\boldsymbol{x}_t$ when the threshold is satisfied, i.e. $\frac{\|\boldsymbol{r}_t\|}{L_{\max}^{(t)}} \geq \sqrt{\alpha}$, $\boldsymbol{r}_t$ is $\boldsymbol{x}_t$'s residual vector.

Given standard orthogonal vectors $\boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_k$, we may find the other $d - k$ vectors $\boldsymbol{v}_{k+1}, \boldsymbol{v}_{k+2}, \ldots, \boldsymbol{v}_d$ that make $\boldsymbol{U} =$

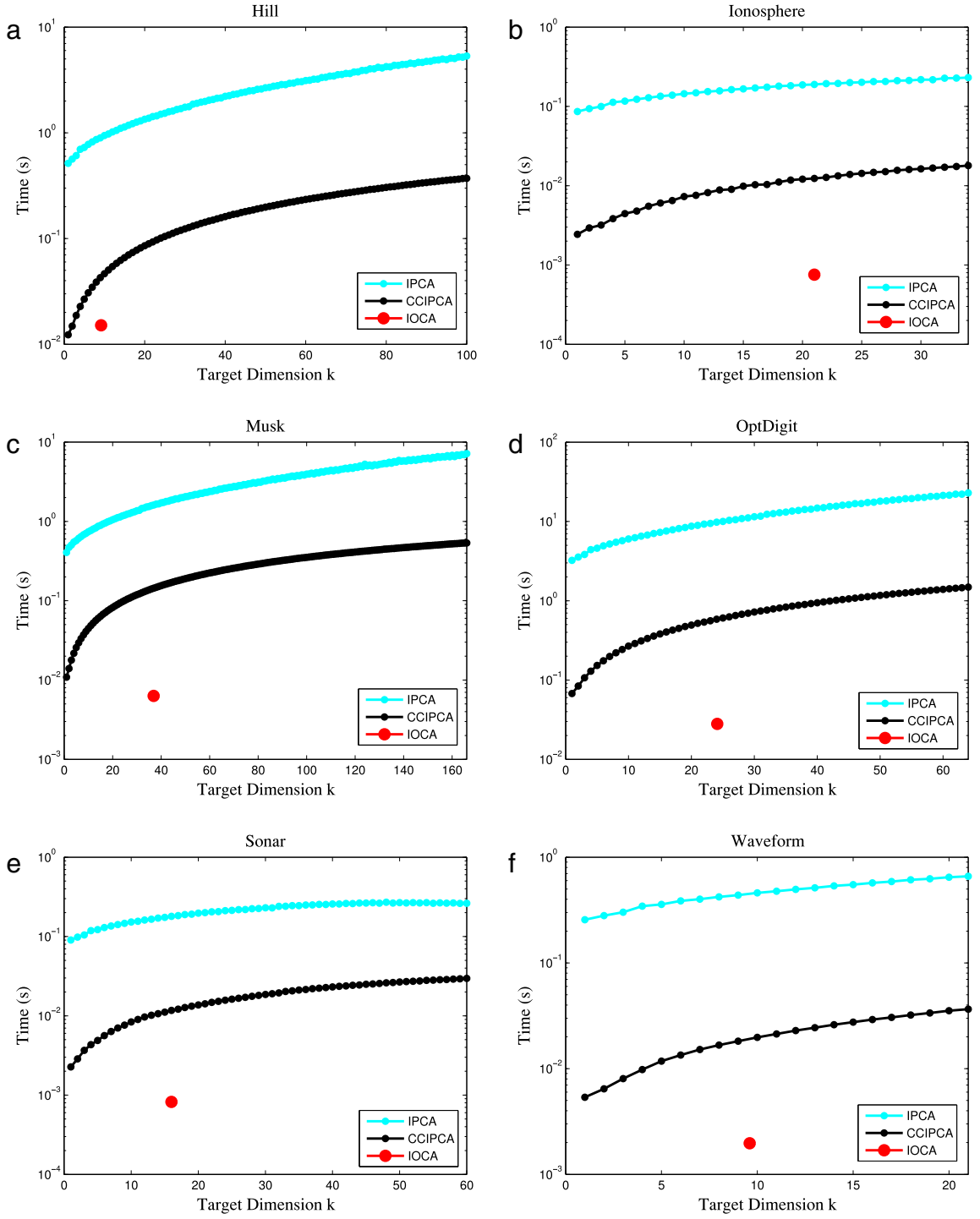

Fig. 11. The comparison between IOCA, IPCA and CCIPCA in execution time (logarithmic scale) on (a) Hill, (b) Ionosphere, (c) Musk, (d) OptDigit, (e) Sonar, (f) Waveform datasets.

$[\boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_k, \boldsymbol{v}_{k+1}, \boldsymbol{v}_{k+2}, \ldots, \boldsymbol{v}_d]$ a $d \times d$ orthogonal matrix. For random vector $\boldsymbol{x} = (x_1, x_2, \ldots, x_d)^\top \in \mathbb{R}^d$ that is independent of the k obtained OCs, we have $\boldsymbol{x}_t = \sum_{i=1}^{k} a_i \boldsymbol{b}_i + \sum_{i=k+1}^{d} a_i \boldsymbol{v}_i$. Note that $\|\boldsymbol{x}_t\|_2^2 = \sum_{i=1}^{d} a_i^2 = \sum_{i=1}^{d} x_i^2$. Then, we obtain $\boldsymbol{r} = \sum_{i=k+1}^{d} a_i \boldsymbol{v}_i$ and $\|\boldsymbol{r}_t\|_2^2 = \sum_{i=k+1}^{d} a_i^2$.

Let $\boldsymbol{a}_t = (a_1, a_2, \ldots, a_d)^\top \in \mathbb{R}^d$, we have $\boldsymbol{a}_t = \boldsymbol{U}^\top \boldsymbol{x}_t$. As d-dimensional random vector $\boldsymbol{x}_t$ follows zero-mean isotropic distribution, $\boldsymbol{U}$ is $d \times d$ orthogonal matrix, and $\boldsymbol{x}_t$ and $\boldsymbol{a}_t = \boldsymbol{U}^\top \boldsymbol{x}_t$ have the same distribution. Thus, random vector $\boldsymbol{a}_t$ is isotropic and zero-mean, and $\frac{\boldsymbol{a}_t}{\|\boldsymbol{a}_t\|_2}$ is a uniform random unit vector.

Let $z_1, z_2, \ldots, z_d$ be i.i.d. random variables, each drawn from the normal distribution $\mathcal{N}(0, 1)$. Let $\boldsymbol{z}_t = (z_1, z_2, \ldots, z_d)^\top$. $\boldsymbol{z}_t$ is isotropic and zero-mean, then $\frac{\boldsymbol{z}_t}{\|\boldsymbol{z}_t\|_2}$ is a uniformly random unit vector. Therefore, though $\boldsymbol{a}_t$ and $\boldsymbol{z}_t$ may follow different distributions, $\frac{\boldsymbol{a}_t}{\|\boldsymbol{a}_t\|_2}$ and $\frac{\boldsymbol{z}_t}{\|\boldsymbol{z}_t\|_2}$ follow the same distribution. Furthermore, given arbitrary fixed matrix $\boldsymbol{L} \in \mathbb{R}^{m \times d}$, $\frac{\|\boldsymbol{L}\boldsymbol{a}_t^2\|_2}{\|\boldsymbol{a}_t\|_2^2}$ and $\frac{\|\boldsymbol{L}\boldsymbol{z}_t\|_2^2}{\|\boldsymbol{z}_t\|_2^2}$ have the same distribution.

Let $\boldsymbol{M}$ be such a fixed matrix which extracts the last $d - k$ coordinates of the vectors in $\mathbb{R}^d$, i.e. for any $\boldsymbol{v} = (v_1, v_2, \ldots, v_d)^\top$,

$\boldsymbol{Mv} = (v_{k+1}, v_{k+2}, \ldots, v_d)^\top$, we have $\frac{\|\boldsymbol{r}_t\|_2^2}{\|\boldsymbol{x}_t\|_2^2} = \frac{\sum_{i=k+1}^d a_i^2}{\sum_{i=1}^d a_i^2} = \frac{\|\boldsymbol{Ma}_t\|_2^2}{\|\boldsymbol{a}_t\|_2^2}$, $\frac{\sum_{i=k+1}^d z_i^2}{\sum_{i=1}^d z_i^2} = \frac{\|\boldsymbol{Mz}_t\|_2^2}{\|\boldsymbol{z}_t\|_2^2}$. Then, we may obtain the conclusion that $\frac{\|\boldsymbol{r}_t\|_2^2}{\|\boldsymbol{x}_t\|_2^2}$ and $\frac{\sum_{i=k+1}^d z_i^2}{\sum_{i=1}^d z_i^2}$ follow the same distribution.

Accordingly, we obtain the following inequality:

$$\begin{aligned}
&Pr\Big[\text{Accepted } \boldsymbol{b}_{k+1}\Big] = Pr\Big[\frac{\|\boldsymbol{r}_t\|_2}{L_{\max}^{(t)}} \geq \sqrt{\alpha}\Big] \\
&\leq Pr\Big[\frac{\|\boldsymbol{r}_t\|_2^2}{\|\boldsymbol{x}_t\|_2^2} \geq \alpha\Big] \\
&= Pr\Big[\frac{z_{k+1}^2 + z_{k+2}^2 + \cdots + z_d^2}{z_1^2 + z_2^2 + \cdots + z_d^2} \geq \alpha\Big] \\
&= Pr\Big[\sum_{i=k+1}^d z_i^2 \geq \alpha \sum_{i=1}^d z_i^2\Big] \\
&= Pr\Big[(1-\alpha)\sum_{i=k+1}^d z_i^2 - \alpha\sum_{i=1}^k z_i^2 \geq 0\Big]. \qquad \text{(A.1)}
\end{aligned}$$

The probability is a tail probability of the sum of d independent variables. Due to the assumptions above, z_i^2s are independent. The moment generating functions for z_i^2s can be computed as follows: if z follows the normal distribution $N(0, 1)$, then $t_1 = z^2$ follows chi-squared distribution whose probability density function is

$$g(t_1) = \begin{cases} \dfrac{t_1^{-\frac{1}{2}} e^{-\frac{t_1}{2}}}{\sqrt{2\pi}}, & t_1 \geq 0 \\ 0, & \text{otherwise.} \end{cases}$$

When $\lambda < \frac{1}{2}$, we obtain

$$\begin{aligned}
E\big[e^{\lambda z^2}\big] &= E\big[e^{\lambda t_1}\big] \\
&= \int_{-\infty}^{+\infty} g(t_1) e^{\lambda t_1} dt_1 \\
&= \frac{1}{\sqrt{2\pi}} \int_0^{+\infty} t_1^{-\frac{1}{2}} e^{-\left(\frac{1}{2}-\lambda\right)t_1} dt_1 \\
&\quad \Big(\text{let } a = \frac{1}{2} - \lambda > 0,\ t_2 = a t_1\Big) \\
&= \frac{1}{\sqrt{2\pi a}} \int_0^{+\infty} t_2^{-\frac{1}{2}} e^{-t_2} dt_2 \quad (\text{let } t_3 = \sqrt{t_2}) \\
&= \frac{1}{\sqrt{2\pi a}} \int_0^{+\infty} 2e^{-t_3^2} dt_3 \Big(\int_0^{+\infty} e^{-t_3^2} dt_3 = \frac{\sqrt{\pi}}{2}\Big) \\
&= (1-2\lambda)^{-\frac{1}{2}}. \qquad \text{(A.2)}
\end{aligned}$$

Therefore, we firstly apply Markov's inequality to the moment generating function by introducing a parameter λ and then optimize this λ to bound the probability.

$$\begin{aligned}
&Pr\Big[(1-\alpha)\sum_{i=k+1}^d z_i^2 - \alpha\sum_{i=1}^k z_i^2 \geq 0\Big] \quad (\text{for } \lambda > 0) \\
&= Pr\Big[\exp\Big\{\lambda\Big\{(1-\alpha)\sum_{i=k+1}^d z_i^2 - \alpha\sum_{i=1}^k z_i^2\Big\}\Big\} \geq 1\Big] \\
&\quad \Big(\text{by Markov's inequality: for nonnegative } X, Pr[X \geq c] \leq \frac{E[X]}{c}\Big) \\
&\leq E\Big[\exp\Big\{\lambda\Big\{(1-\alpha)\sum_{i=k+1}^d z_i^2 - \alpha\sum_{i=1}^k z_i^2\Big\}\Big\}\Big] \\
&\quad (z_i\text{s are independent with each other}) \\
&= \prod_{i=k+1}^d E\Big[e^{\lambda(1-\alpha)z_i^2}\Big] \cdot \prod_{i=1}^k E\Big[e^{-\lambda\alpha z_i^2}\Big] \\
&\quad (z_i\text{s follow the same distribution}) \\
&= E\Big[e^{\lambda(1-\alpha)z^2}\Big]^{d-k} \cdot E\Big[e^{-\lambda\alpha z^2}\Big]^k \\
&\quad \Big(\text{for } \lambda(1-\alpha) < \frac{1}{2} \text{ and } -\lambda\alpha < \frac{1}{2}\Big) \\
&= \Big(1-2\lambda(1-\alpha)\Big)^{-\frac{d-k}{2}} \cdot \Big(1+2\lambda\alpha\Big)^{-\frac{k}{2}}. \qquad \text{(A.3)}
\end{aligned}$$

Let $\mathcal{L} = \Big(1-2\lambda(1-\alpha)\Big)^{-\frac{d-k}{2}} \Big(1+2\lambda\alpha\Big)^{-\frac{k}{2}}$, $\mathcal{L}$ is minimized when

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \lambda} &= \Big(d - k - \alpha d - (2\alpha d(1-\alpha))\lambda\Big) \\
&\quad \times \Big(1-2\lambda(1-\alpha)\Big)^{-\frac{d-k}{2}-1} \Big(1+2\lambda\alpha\Big)^{-\frac{k}{2}-1} = 0.
\end{aligned}$$

Thus, we obtain

$$\lambda = -\frac{d-k-\alpha d}{2\alpha d(1-\alpha)}. \qquad \text{(A.4)}$$

According to the constrains in (A.3): $\lambda > 0, \lambda(1-\alpha) < \frac{1}{2}$ and $-\lambda\alpha < \frac{1}{2}$, we have $k \geq (1-\alpha)d$.

So that

$$\begin{aligned}
&Pr\Big[\text{Accepted } \boldsymbol{b}_{k+1}\Big] \\
&\leq \Big(\frac{d-k}{\alpha d}\Big)^{-\frac{d-k}{2}} \cdot \Big(\frac{k}{(1-\alpha)d}\Big)^{-\frac{k}{2}} \\
&= \left(\left(1 + \frac{\alpha d + k - d}{d-k}\right)^{\frac{d-k}{\alpha d+k-d}}\right)^{\frac{\alpha d+k-d}{d-k}\cdot\frac{d-k}{2}} \cdot \Big(\frac{(1-\alpha)d}{k}\Big)^{\frac{k}{2}} \\
&\quad \Big(\text{for } n > 0,\ (1+\frac{1}{n})^n \leq e\Big) \\
&\leq \exp\left(\frac{k}{2}\left(1 - \frac{(1-\alpha)d}{k} + \ln\frac{(1-\alpha)d}{k}\right)\right) \\
&\quad \Big(\text{let } \epsilon = 1 - \frac{(1-\alpha)d}{k}\Big) \\
&= \exp\Big(\frac{k}{2}\big(\epsilon + \ln(1-\epsilon)\big)\Big) \\
&\quad \Big(\text{by Taylor expansion } \ln(1-\epsilon) \leq -\epsilon - \frac{\epsilon^2}{2}\Big) \\
&\leq \exp\Big(-\frac{k\epsilon^2}{4}\Big) \\
&= \exp\Big(-\frac{(k-d+\alpha d)^2}{4k}\Big). \quad \square \qquad \text{(A.5)}
\end{aligned}$$

Appendix B. Proof of Theorem 4.2

Proof. The low-dimensional representation of $\boldsymbol{x}_t$ is wrote as a $k^{(N)}$-dimensional vector: $\boldsymbol{y}_t = (y_{t,1}, y_{t,2}, \ldots, y_{t,k^{(t)}}, 0, \ldots, 0)^\top$.

Assume when $\boldsymbol{x}_t$ is input, $k^{(t-1)}$ is the dimension of obtained feature subspace, if $\|\boldsymbol{x}_t - \sum_{i=1}^{k^{(t-1)}} y_{t,i}\boldsymbol{b}_i\| \geq f(\frac{k^{(t-1)}}{d})$, a new component will be extracted from $\boldsymbol{x}_t$, then we have $k^{(t)} = k^{(t-1)}+1$ and $\|\boldsymbol{x}_t - \sum_{i=1}^{k^{(t)}} y_{t,i}\boldsymbol{b}_i\| = 0$; otherwise, the basis set remains unchanged. As $f(\omega)$ is a strictly monotonic increasing function

whose range is [0, 1], we obtain that $f(0) \geq 0$ and when $\omega > 0$, $f(\omega) > 0$. Therefore,

$$\|\boldsymbol{r}_t\|_2 = \left\|\boldsymbol{x_t} - \sum_{i=1}^{k^{(t)}} y_{t,i}\boldsymbol{b_i}\right\|_2 < f\left(\frac{k^{(t)}}{d}\right)L_{\max}^{(t)}. \tag{B.1}$$

Obviously, we also have $f(\frac{k^{(t)}}{d}) \leq f(\frac{k^{(N)}}{d})$ and $L_{\max}^{(t)} \leq L_{\max}^{(N)}$. Thus,

$$\|\boldsymbol{r}_t\|_2 < f\left(\frac{k^{(N)}}{d}\right)L_{\max}^{(N)}. \quad \square \tag{B.2}$$

References

Araújo, U., Saldanha, B., Galvao, R., Yoneyama, T., Chame, H., & Visani, H. (2001). The successive projections algorithm for variable selection in spectroscopic multicomponent analysis. *Chemometrics and Intelligent Laboratory Systems*, *57*(2), 65–73.

Artač, M., Jogan, M., & Leonardis, A. (2002). Incremental PCA for on-line visual learning and recognition, In *16th International conference on pattern recognition*, Vol. 3, (pp. 781–784).

Bingham, E., & Mannila, H. (2001). Random projection in dimensionality reduction: Applications to image and text data, In *KDD-2001: proceedings of the seventh ACM SIGKDD international conference on knowledge discovery and data mining, New York*, (pp. 245–250).

Bishop, C. M. (1996). *Neural networks for pattern recognition*. Oxford University.

Bishop, C. M. (1999). Bayesian pca. *Advances in Neural Information Processing Systems*, *11*, 382–388.

Bishop, C. M. (2006). *Pattern recognition and machine learning*. New York: Springer.

Blake, C.L., & Merz, C.J. (1996). UCI repository of machine learning databases, Irvine, CA: University of California Department of Information.

Cai, D., & He, X. (2005). Orthogonal locality preserving indexing, In *Proc. ACM SIGIR conf. research and development in information retrieval*, (pp. 3–10).

Chartier, T. (2006). Devastating roundoff error. *Math Horizons*, *13*(4), 11-11.

Choi, M. D. (1983). Tricks or treats with the Hilbert matrix. *American Mathematical Monthly*, *90*(5), 301–312.

Duarte, M., & Hu, Y. H. (2004). Vehicle classification in distributed sensor networks. *Journal of Parallel and Distributed Computing*, *64*(7), 826–838.

Fan, M. Y., Qiao, H., & Zhang, B. (2009). Intrinsic dimension estimation of manifolds by incising balls. *Pattern Recognition*, *42*(5), 780–787.

Fan, M. Y., Zhang, X. Q., Chen, S. Y., Bao, H. J., & Maybank, S. (2013). Dimension estimation of image manifolds by minimal cover approximation. *Neurocomputing*, *105*(3), 19–29.

Fodor, I. K. (2002). A survey of dimension reduction techniques. *Neoplasia*, *7*(5), 475–485.

Fukunaga, K. (1990). *Introduction to statistical pattern recognition* (2nd ed.). New York: Academic Press.

Fukunaga, K. (1982). Intrinsic dimensionality extraction. In *Handbook of statistics*: *Vol. 2. Classification, pattern recognition and reduction of dimensionality* (pp. 347–362).

Gheyas, I. A., & Smith, L. S. (2010). Feature subset selection in large dimensionality domains. *Pattern Recognition*, *43*(1), 5–13.

Gillis, N., & Vavasis, S. A. (2014). Fast and robust recursive algorithms for separable nonnegative matrix factorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *36*(4), 698–714.

Golub, G. H., & Loan, C. F. V. (1996). *Matrix computations* (3rd ed.). Baltimore: Johns Hopkins University Press.

Guan, N. Y., Tao, D. C., Luo, Z. G., & Yuan, B. (2012). Online nonnegative matrix factorization with robust stochastic approximation. *IEEE Transactions on Neural Networks and Learning Systems*, *23*(7), 1087–1099.

Hall, P., Marshall, D., & Manin, R. (1998). Incremental eigenalysis for classification, In *British machine vision conference*, Vol. 1, (pp. 286–295).

Hall, P., Marshall, D., & Martin, R. (2000). Merging and splitting eigenspace models. *Transactions on Pattern Analysis and Machine Intelligence*, *22*(9), 1042–1049.

Hazewinkel, Michiel (Ed.) (2001). *Encyclopedia of mathematics*. Springer.

He, X. F., & Niyogi, P. (2005). Locality preserving projections. *Advances in Neural Information Processing Systems*, *45*(1), 186–197.

Hotelling, H. (1935). Relations between two sets of variants. *Biometrika*, *28*(3–4), 312–377.

Hua, J., Tembe, W., & Dougherty, E.R. (2008). Feature selection in the classification of high-dimension data, In *IEEE international workshop on genomic signal processing and statistics*, (pp. 667–671).

Hull, J. J. (1994). A database for handwritten text recognition research. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *16*(5), 550–554.

Huo, X. M., & Smith, A. K. (2008). A survey of manifold-based learning methods. In *Mining of Enterprise Data*.

Hyvarinen, A. (1999). Fast and robust fixed-point algorithms for independent component analysis. *IEEE Transactions on Neural Networks*, *10*(3), 626–634.

Jin, X., Xu, A., Bie, R., & Guo, P. (2006). Machine learning techniques and chi-square feature selection for cancer classification using SAGE gene expression profiles. In *Data Mining for Biomedical Applications* (pp. 106–115). Berlin, Heidelberg: Springer.

Jolliffe, I. (1986). *Principal component analysis*. New York: Springer-Verlag.

Kim, T. K., Stenger, B., Kittler, J., & Cipolla, R. (2011). Incremental linear discriminant analysis using sufficient spanning sets and its applications. *International Journal of Computer Vision*, *91*(2), 216–232.

Law, M. H. C., & Jain, A. K. (2006). Incremental nonlinear dimensionality reduction by manifold learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *28*(3), 377–391.

Lee, K., Ho, J., & Kriegman, D. J. (2005). Acquiring linear subspaces for face recognition under variable lighting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *27*(5), 684–698.

Lee, D., & Seung, H. (1999). Learning the parts of objects by non-negative matrix factorization. *Nature*, *401*(6755), 788–791.

Lee, J., & Zhang, C. S. (2006). Classification of gene-expression data: the manifold-based metric learning way. *Pattern Recognition*, *39*(1), 2450–2463.

Leon, S. J., Björck, Å., & Gander, W. (2013). Gram-schmidt orthogonalization: 100 years and more. *Numerical Linear Algebra with Applications*, *20*(3), 492–532.

Liu, H., & Yu, L. (2005). Toward integrating feature selection algorithms for classification and clustering. *IEEE Transactions on Knowledge and Data Engineering*, *17*(4), 491–502.

Lohr, S. (2008). The age of big data. *New York Times*, *16*(4), 10–15.

Lu, J. W., Tan, Y. P., & Wang, G. (2011). Discriminative multimanifold analysis for face recognition from a single training sample per person. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*(1), 39–51.

Mairal, J., & Bach, F. (2010). Online learning for matrix factorization and sparse coding. *Journal of Machine Learning Research*, *11*(1), 19–60.

Pang, S., Ozawa, S., & Kasabov, N. (2005). Incremental linear discriminant analysis for classification of data streams. *IEEE Transactions on Systems, Man and Cybernetics B*, *35*(5), 905–914.

Peng, H., Long, F., & Ding, C. (2005). Feature selection based on mutual information criteria of max-dependency, max-relevance, andminredundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *27*(8), 1226–1238.

Prokhorov, D. (2001). IJCNN 2001 neural network competition, Ford Research Laboratory.

Ren, H., & Chang, C. I. (2003). Automatic spectral target recognition in hyperspectral imagery. *IEEE Transactions on Aerospace and Electronic Systems*, *39*(4), 1232–1249.

Ren, C. X., & Dai, D. Q. (2010). Incremental learning of bidirectional principal components for face recognition. *Pattern Recognition*, *43*(1), 318–330.

Roweis, S., & Saul, L. (2000). Nonlinear dimensionality reduction by locally linear embedding. *Science*, *290*(5500), 2323–2326.

Sarveniazi, A. (2014). An actual survey of dimensionality reduction. *American Journal of Computational Mathematics*, *4*(2), 55–72.

Tao, D., Li, X., Wu, X., & Maybank, S. J. (2007). General tensor discriminant analysis and Gabor features for gait recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *29*(10), 1700–1715.

Tenenbaum, J., de Silva, V., & Langford, J. (2000). A global geometric framework for nonlinear dimensionality reduction. *Science*, *290*(5500), 2319–2323.

Torkkola, K. (2001). Linear discriminant analysis in document classification, In *IEEE ICDM workshop text mining*, 77(20), 998-1002.

Wang, J. Y. (2002). *Application of support vector machines in bioinformatics*. (Master's thesis), Department of Computer Science and Information Engineering, National Taiwan University.

Wang, D., & Lu, H. C. (2013). On-line learning parts-based representation via incremental orthogonal projective non-negative matrix factorization. *Signal Processing*, *93*(6), 1608–1623.

Wang, L. W., Wang, X., & Feng, J. F. (2006). Subspace distance analysis with application to adaptive Bayesian algorithm for face recognition. *Pattern Recognition*, *39*(3), 456–464.

Wang, Q., Xu, G.Y., & Ai, H.Z. (2003). Learning object intrinsic structure for robust visual tracking, In *Proceedings of international conference on computer vision and pattern recognition, vol. 2, Madison, Wisconsin*, (pp. 227–233).

Weinberger, K.Q., & Saul, L.K. (2004). Unsupervised learning of image manifolds by semi-definite programming, In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR-04), Vol. 2, Washington, DC*, (pp. 988–995).

Weng, J. Y., Zhang, Y. L., & Hwang, W. S. (2003). Candid covariance-free incremental principal component analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *25*(8), 1034–1040.

Yann, L. C., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324.

Ye, J. P., Janardan, R., & Li, Q. (2004). GPCA: an efficient dimension reduction scheme for image compression and retrieval. In *Proceedings of the tenth ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 354–363). ACM.

Ye, J., Li, Q., Xiong, H., & Park, H. (2005). IDR/QR: an incremental dimension reduction algorithm via qr decomposition. *IEEE Transactions on Knowledge and Data Engineering*, *17*(9), 1208–1222.

Zhang, T., Li, X., Tao, D., & Yang, J. (2008). Multimodal biometrics using geometry preserving projections. *Pattern Recognition*, *41*(3), 805–813.

Zhang, L., Mahdavi, M., Jin, R., & Yang, T. (2012). Recovering optimal solution by dual random projection. *Journal of Machine Learning Research*, *30*, 135–157.

Zhang, Y., & Weng, J. (2011). Convergence analysis of complementary candid incremental principal component analysis, Dept. of Computer Science and Eng., Michigan State Univ., East Lansing.

Zhao, H., & Yuen, P. C. (2008). Incremental linear discriminant analysis for face recognition. *IEEE Transactions on Systems, Man and Cybernetics B*, *38*(1), 210–221.