Advances in Incremental PCA Algorithms*

Tal Halpern and Sivan Toledo

Tel-Aviv University

Abstract. We present a range of new incremental (single-pass streaming) algorithms for incremental principal components analysis (IPCA) and show that they are more effective than exiting ones. IPCA algorithms process the columns of a matrix A one at a time and attempt to build a basis for a low-dimensional subspace that spans the dominant subspace of A. We present a unified framework for IPCA algorithms, show that many existing ones are parameterizations of it, propose new sophisticated algorithms, and show that both the new algorithms and many existing ones can be implemented more efficiently than was previously known. We also show that many existing algorithms can fail even in easy cases and we show experimentally that our new algorithms outperform existing ones.

Keywords: Principal Components Analysis, Streaming Algorithms, Frequent Directions

1 Introduction

Incremental or streaming algorithms for *principal components analysis* have a wide range of big-data applications in machine learning and other applications [1,9,12,3,4,16]. We are interested in real matrices $A \in \mathbb{R}^{m\times n}$ in which columns represent m-dimensional data vectors. We assume that most of the vectors (columns) consist of a component that lies in some $\bar{k} \ll m$ dimensional subspace and of a noise component, and that noise components have lower norm than the data components. We also assume that some, but not too many, columns may be outright outliers (that is, that they are far from the unknown low-dimensional subspace).

Let $A = U_m S_m V_m^T = U_m \text{diag}(s_m) V_m^T$ be the singular value decomposition (SVD) of A, let s_ℓ be the ℓ dominant singular values and let U_ℓ and V_ℓ be the corresponding singular vectors. The columns of U_ℓ are called the *principal components* of A and algorithms that compute or approximate U_ℓ and s_ℓ are often referred to as *principlal components analysis* (PCA), and as stated, are useful in many data analyses; the right singular vectors V are far less useful.

Incremental (or streaming) PCA algorithms, which we refer to as IPCA, approximate $U_{\bar{k}}$ and $s_{\bar{k}}$ by processing the columns of A one at a time using a

* This research is supported by grants 965/15 and 863/15 from the Israel Science Foundation (founded by the Israel Academy of Sciences and Humanities) and by a grant from the United States-Israel Bi-national Science Foundation (BSF)

small data structure of size $\Theta(mk)$ for $k \geq \bar{k}$, often just an m-by-k matrix that we denote by B. The algorithms that we discuss are all *single-pass* algorithms that can discard a column once it has been processed. We describe a number of existing algorithms in the next section, where we show that they can all be viewed as instantiations of a unified framework.

There are several ways to measure the quality of an approximate PCA. In this paper, we focus on the reconstruction of the dominant left subspace of the matrix. That is, we measure the quality primarily by how well the left singular vectors of B, which we denote by U, span $U_{\bar{k}}$. We discuss existing ways to measure the quality of PCA algorithms, as well as a new metric, in Section 2. Unfortunately, as we show in Section 3, existing IPCA algorithms do not satisfy some useful error bounds, which leads us to define new algorithms in Section 4.

The computational cost of many existing (including very recent and highly regarded) IPCA algorithms is $\Theta(mk^2)$ operations per column. This is quite astonishing, since total $\Theta(mk^2n)$ cost of the algorithm is comparable and possibly higher than the cost of block Lanczos or subspace iterations, methods that can produce very accurate results if the spectral gap $\sigma_{\bar{k}}/\sigma_{k+1}$ is large. In other words, these single-pass IPCA methods are efficient mostly in the sense of memory usage, not in terms of computational effort. To address this issue, we show in Section 5 that a technique invented by Chahlaoui et al. [3] to reduce the per-iteration cost in one particular IPCA algorithm actually applies to all the algorithms in our unified framework. Our result shows that the cost of all of them can be reduced to $\Theta(mk)$ operations per column. We also describe in Section 5 another technique from the literature to reduce the total cost of IPCA algorithms.

To summarize, the main contributions of this paper are: (1) we show that a wide range of existing IPCA algorithms can be described as parameterized variants of a single unified framework, (2) we explain the weaknesses of existing error metrics for IPCA, propose a new one, and show that existing algorithms perform arbitrarily poorly in it, even in easy cases (huge spectral gaps), (3) we propose three new sophisticated IPCA algorithms, (4) we show how to implement them and many existing algorithms in $O(mk)$ operations per column, and (5) we show experimentally that our new algorithms outperform the best existing algorithms on both synthetic and real-world data sets.

2 A Unified Framework for IPCA

Our framework can express a wide range of IPCA algorithms using definitions of two functions, a *filter* $f : (\mathbb{R}^{m\times k}, \mathbb{R}^m) \longrightarrow \mathbb{R}^m$ and a *reweighter* $g : \mathbb{R}^{k+1} \longrightarrow \mathbb{R}^k$. The role of f is to filter or modify a new data vector (column) given the vector and a basis for a k-dimensional subspace that hopefully represents all the preceding columns of A. The role of g is to assign weights to the singular vectors of the new basis. Formally, the framework works as follows.

1. Initialization. Let U_t and s_t be the left singular vectors and the singular values of $A_{:\,,1:\,t}$ for some $t \geq k$. Set $B_t = (U_t)_{:\,,1:\,k}\,\mathrm{diag}\left((s_t)_{1:\,k}\right)$.

```
2. For $t+1$ to $n$,
    (a) Compute $w = f(B_t, A_{:\,,t+1})$.
    (b) Let $U_{t+1}$ and $\tilde{s}_{t+1}$ be the left singular vectors and the singular values of
        $\begin{bmatrix} B_t & w \end{bmatrix}$, and let $B_{t+1} = (U_{t+1})_{:\,,1:\,k}\operatorname{diag}(g(\tilde{s}_{t+1}))$.
```

It is tempting to think that in step 2(b) we need to compute the SVD or PCA of $\begin{bmatrix} B_t & w \end{bmatrix}$, but it turns out that it is often possible to carry out this step without computing the SVD/PCA explicitly.

We now give a few examples of how to instantiate algorithms from the literature using our framework. Setting $f_{\text{ID}}(B_t, A_{:\,,t+1}) = A_{:\,,t+1}$ and $g_{\text{ID}}(s) = s_{1:\,k}$ gives both Basic IPCA [2,16] and QR-IPCA [3]. Setting

$$f_{\text{Brand}}(B_t, A_{:\,,t+1}) = \begin{cases} U_t U_t^T A_{:\,,t+1} & \operatorname{rank}(U_t) = k \\ A_{:\,,t+1} & \text{otherwise} \end{cases}$$

(an orthonormal projection) and $g_{\text{ID}}(s) = s_{1:\,k}$ gives Brand's method [2]. Brand also proposed a variant in which the projection is used only if the projection error is below some threshold τ,

$$f_{\text{trancate}}(B_t, A_{:\,,t+1}) = \begin{cases} U_t U_t^T A_{:\,,t+1} & \|A_{:\,,t+1} - U_t U_t^T A_{:\,,t+1}\| < \tau \\ A_{:\,,t+1} & \text{otherwise ,} \end{cases}$$

still with the identity reweighter g_{ID}. Using the identity filter f_{ID} and

$$g_{\text{eva}}(s) = \sqrt{s_{1:k}^2 - s_{k+1}^2}$$

(the eigenvalues of the Gram matrix are shifted down so as to annihilate the smallest singular value) gives Liberty's Frequent Directions algorithm [10]. The identity filter and

$$g_{\text{decay}}(s) = \lambda s_{1:\,k}$$

(drop the smallest singular value and shrink the rest by a factor $0 < \lambda < 1$ gives a method suggested by Levey and Lindenbaum [9] (but their algorithm processes incoming columns in blocks, not one by one).

3 Error Metrics and Impossibility Results

Most of the early work on IPCA was heuristic and provided no provable guarantees on the quality of the approximation. One interesting but not particularly useful exception is the work of Chandrasekaran et el., [5,11]. They show how to keep track of the error and they propose to increment k whenever necessary to preserve the bound. However, their method results in very high a-priori bounds for k; in many interesting cases, their bound is equivalent to maintaining the full rank.

Liberty et al. [7,10] made a huge step forward. They showed that their Frequent Directions algorithms achieves two useful a-priori bounds, a so-called *Gram reconstruction bound*

$$\|AA^T - BB^T\|_2 \leq \frac{1}{k - \bar{k}} \|A - A_k\|_F^2$$

and the so-called *projection bound*

$$\|A - \bar{U}\bar{U}^T A\|_F^2 \leq \left(1 + \frac{1}{k - \bar{k}}\right) \|A - A_k\|_F^2 \ , \tag{1}$$

where $\bar{U}$ consists of the k dominant left singular vectors of B. We are interested in projection bounds and variants of it, which measure the quality of the approximation of the dominant subspace, not the approximation of the Gram matrix of A.

In extensive experiments, we found that (1) does not provide useful bounds in high-dimensional noisy problems. The reason is simple: $\|A - A_k\|_F^2 = \sum_{k+1}^m \sigma_i^2$, so if m is large and if the singular values do not decay quickly to insignificant values, $\|A - A_k\|_F^2$ may be large even for good approximations, say $\text{span}(B) = \text{span}(A_k)$. This means that this bound cannot distinguish between good approximations and bad ones.

One way to address this issue is to replace the Frobenius norm by the 2-norm in the projection error. No rigorous bounds of this form are known for IPCA algorithms, but it may still be a good way to assess the quality of approximations. We advocate and use a slightly different bound, which we call the *subspace reconstruction bound* (or just *reconstruction bound*),

$$E_{\text{recon}} = E_{\text{recon}}\left(\bar{k}, A, U\right) = \frac{\|A_{\bar{k}} - UU^T A_{\bar{k}}\|_F}{\|A_{\bar{k}}\|_F} \ .$$

This bound measures how well U spans the dominant subspace of A. Note that U has rank k and that $A_{\bar{k}}$ has rank $\bar{k} \leq k$. We note that if the gap between $\sigma_{\bar{k}}$ and σ_k is small, the problem of finding a U with a small reconstruction error is highly ill conditioned, because small perturbations in A can cause dramatic changes in $A_{\bar{k}}$; we feel that this is acceptable when the sought-after object is the dominant subspace.

Unfortunately, it turns out that guaranteeing a small reconstruction error is impossible for all of the existing algorithms, including Frequent Directions, even in easy cases.

Theorem 1. *For any positive real number M, and for any ranks $\bar{k} \leq k$, there exist a matrix $A \in \mathbb{R}^{m \times n}$ with $\sigma_{\bar{k}}/\sigma_{\bar{k}+1} > M$ such that if U is the rank-k basis found by IPCA with f_{ID} and g_{ID}, then $E_{\text{recon}}\left(\bar{k}, A, U\right) = 1$. The same is true (with different counter example matrices) for (f_{ID}, g_{eva}) (Frequent Directions), for (f_{Brand}, g_{ID}), $(f_{truncate}, g_{ID})$, and (f_{ID}, g_{decay}).*

We omit the proof, which is available at [8], due to lack of space. This result is quite dramatic. The counter examples are all easy, in the sense that the spectral gap can be large, and that the rank of U is allowed to be much larger than $\bar{k}$. The fact that this wide range of simple algorithms fails to guarantee a good approximation leads us to define more sophisticated algorithms in the next section. Currently, they are all heuristic; we do not have strong reconstruction bounds for them, but we do have experimental evidence that they work well.

4 New Heuristics

The first heuristic that we propose is *Tunable Shrinkage*, which is a parameterization of Frequent Directions [7,10]. A different parameterized shrinkage strategy was proposed by [6]. Tunable Shrinkage uses a modified reweighter

$$g_{\text{r-eva}}(s) = \sqrt{s_{1:k}^2 - s_{k+1}^2/r}$$

for some $1 \leq r < \infty$. The essence of this reweighter is to drop the smallest singular value, like g_{eva}, but to shrink the other singular values by a smaller amount. Setting $r = 1$ gives g_{eva} and setting $r = \infty$ gives g_{ID}. We can show (proof is omitted due to lack of space; see [8]) that the degradation in the approximation bound relative to Frequent Direction depends on r,

Theorem 2. *Let U be the basis of the the sketch $B \in \mathbb{R}^{m\times k}$be the sketch produced by Tunable Shrinkage, for any $\bar{k} < k/r$ it holds that*

$$\|A - UU^T A\|_F^2 \leq \left(1 + \frac{\bar{k}r}{k - \bar{k}r}\right) \|A - A_{\bar{k}}\|_F^2 \ .$$

Next, we propose *Boosted IPCA (BIPCA)*. This method uses the identity reweighter but with a sophisticated statefull randomized filter. The state that the filter maintains is the average mass of columns $\alpha_t = \|A_{:\ ,1:\ t}\|_F^2/t$, the smallest singular σ_t value of B_t, and a counter c. We initialize $c = 2$. The filter starts by tossing a biased coin with success probability $1/c$. If the coin toss is successful, the filter simply sets w to the projection $p_t = U_t U_t^T A_{:\ ,t+1}$ and it increments c. Otherwise, the filter sets $c = 2$ and computes the projection-residual $r_t = A_{:\ ,t+1} - U_t U_t^T A_{:\ ,t+1}$ and its 2-norm ρ. If $\rho > \sigma_t$, we set $w = A_{:\ ,t+1}$ and continue. If the residual is small $\rho \leq \sigma_t$, we toss another coin with success probability $1 - \min(1, \rho^2/\alpha_t)$. If the toss is successful, we again set $w = A_{:\ ,t+1}$. If the coin toss is unsuccessful, we *boost* the residual and set $w = p_t + \beta_t r_t$ where

$$\beta_t = \begin{cases} \sigma_t/\rho + \epsilon & p_t = 0 \\ \min\left(\sigma_t/\rho, \sqrt{(\|A_{:\ ,t+1}\|^2 + \sigma_t^2)/\|A_{:\ ,t+1}\|^2}\right) & \text{otherwise,} \end{cases}$$

where ϵ is infinitesimal (not a significant numeric value). The test $p_t = 0$ is done in a numerically-robust way (small p_ts are admitted). The ϵ term forces w to be retained in B_{t+1} when $p_t = 0$.

We omit the detailed rationale for these heuristic rules due to lack of space, but the essence is to use an inexpensive update rule when a more expensive update is not likely to significantly improve the approximation. they are justified experimentally below.

Our third heuristic, *JIT-PCA*, is closely related to BIPCA and uses the same notation, but is a little simpler, often more efficient, but sometimes a little less accurate. It also uses an identity reweighter and a sophisticated filter. The filter tosses one coin with probability $(1/c)(1 - \min(1, \rho^2/\alpha_t))$ (the product of the probabilities in BIPCA). If the coin toss is successful, we set $w = p_t$ and increment c, otherwise we set $w = p_t + \gamma_t r_t$ and set $c = 2$, with γ_t defined as

$$\gamma_t = \begin{cases} 1 & \rho > \sigma_t \\ \beta_t & \text{otherwise} . \end{cases}$$

5 Efficient QR-Based Implementations

Naive implementations of step 2(b) of our framework compute the SVD of $\begin{bmatrix} B_t \; w \end{bmatrix}$. This is expensive, costing $\Theta(mk^2)$ operations per incoming data vector (column). Given U_t, s_t and w, we can *update* the SVD, but this is still expensive. The singular values can be updated in $\Theta(k^3) \ll \Theta(mk^2)$, but updating the singular vectors requires multiplying an m-by-k matrix by a k-by-k matrix, costing $\Theta(mk^2)$ operations. Researchers proposed three main mechanisms to reduce this cost. One, aggressively used by Brand [2] and somewhat less aggressively by our new heuristics, it to set $w = p_t$. This implies that we do not need to explicitly update U_t; instead, we represent it as a product of an m-by-k matrix by a k-by-k matrix and we update only the k-by-k matrix. Steps of this form only cost $2mk + \Theta(k^3)$. The second mechanism is to batch columns and to update the basis only every ℓ columns. By setting $\ell \approx k$ the amortized per-column cost drops to $\Theta(mk)$ operations. This idea is used in Frequent Directions [7,10], by Levey, Lindenbaum [9], etc.

A more interesting mechanism was discovered by Chahlaoui, Gallivan, and Van Dooren [3]. They proposed to represent B_t using its QR factorization, $B_t = Q_t R_t$. The singular values of B_t are those of R_t and the smallest singular value, when needed, can be extracted from R_t. Their method is based on two clever observations. The first is that the smallest singular pair/triplet of R_t can be computed inexpensively using Lanczos. The second, and perhaps the more surprising, is that Q_t can be updated using $4mk$ operations by applying a single Householder reflection to it. This reduces the total cost of our framework to $8mk$ operations per incoming column.

Our key observation is that the QR-based representation can be applied not only to the simple filter and reweighter choices of Chahlaoui et al., but also to those of Frequent Directions and our new heuristics (Section 4). This implies that the cost of all of these heuristics is bounded by $8mk + \Theta(k^3)$ operations per column. More specifically, we perform the QR-IPCA update as originally proposed and keep σ_{k+1} (Chahlaoui et al. discard it). We then compute the SVD of $R_t = U_R S_R V_R^T$. We now apply the reweighter to the diagonal of S_R reconstruct $\tilde{R}_t = U_R \tilde{S}_R V_R^T$, where $\tilde{S}_R$ is the reweighted diagonal matrix. The last step is to perform an RQ decomposition on $\tilde{R}_t$ to restore its upper triangular structure. The Q factor is not used.

We note that the paper of Chahlaoui et al. preceded the discovery of Frequent Directions, and that the researchers who discovered Frequent Directions were not aware of this but were very much interested in reducing the per-column cost to $O(mk)$; this led them to the batching technique, which is really not necessary.

6 Experimental Evaluation

We demonstrate the effectiveness of our new methods as well as the weakness of Frequent Directions with both synthetic data sets and real-world data. The results also demonstrate that Frequent Directions is not always superior to older methods, and in particular that Basic IPCA sometimes beats it.

The first family of synthetic matrices were produced as $BD + BN$ where B is a square random orthonormal matrix of dimension m, D is an m-by-n matrix with zeros in rows $3, \ldots, m$ and uniform random entries between -0.5 and 0.5 in the first two rows, and N is an m-by-n matrix with Gaussian entries with zero mean and standard deviation 0.05. The columns of D are sorted by norm; this facilitates evaluation of approximations, as we shall see below. The BD term represents data; its rank is 2, and its column space is spanned by the first columns of B. The BN term represents noise. We use $m = 50$ or $m = 200$ and $n = 5000$. We then ran IPCA algorithms that each produced an m-by-2 orthonormal approximation U of the dominant left singular vectors of A.

Figure 1 presents the output of four algorithms on a problem with $m = 50$. We projected the columns of A onto the basis U and plotted the coordinates that we received. Points are colored sequentially using a color map that spans red to blue. In this problem, all the approximations are good. The coordinates roughly span a square, and the color correlates well with the norm, which implies that the

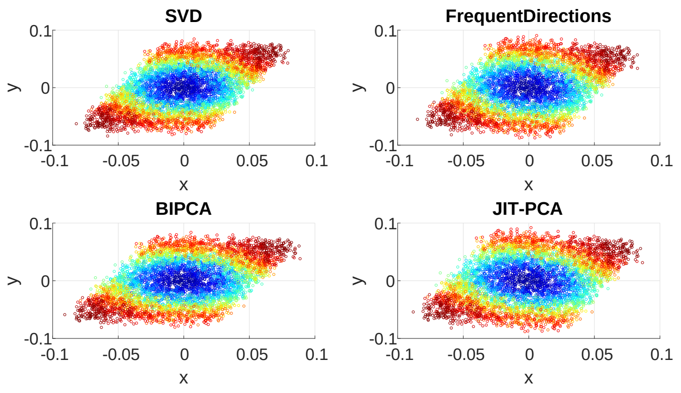

Fig. 1. Reconstructions of noisy 2-dimensional data embedded in vectors of dimension 50; all the algorithms perform well.

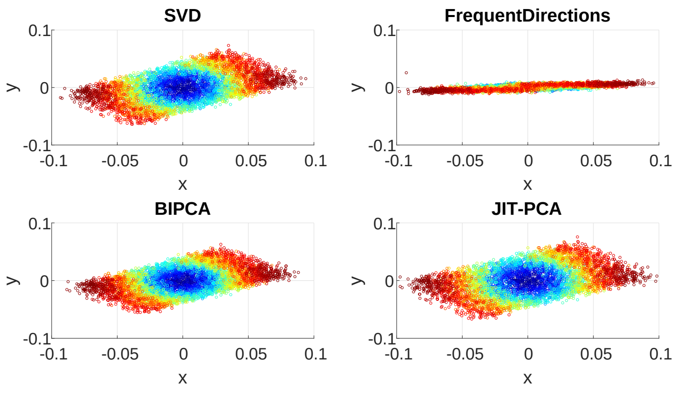

Fig. 2. Reconstructions of noisy 2-dimensional data embedded in vectors of dimension 200; Frequent Directions performs poorly.

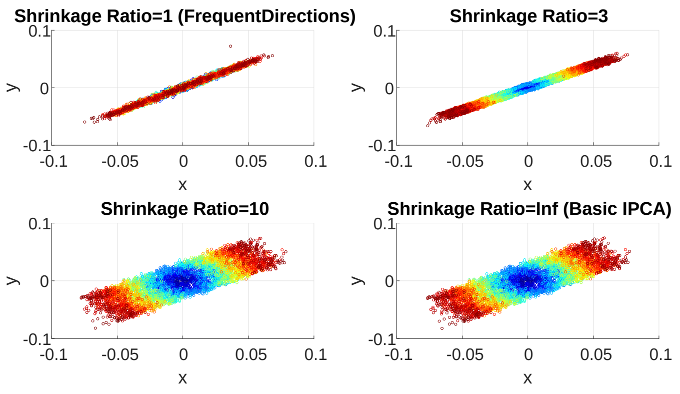

Fig. 3. Reconstructions of noisy 2-dimensional data with $m = 200$ using variants of Frenquent Direction with several levels of shirnakage (decay).

reconstruction is good (recall that the columns of D are ordered by norm). This visualization mechanism is common in the dimension-reduction literature [13,14].

Figure 2 presents the results of a similar experiment but with $m = 200$. The higher dimension causes Frequent Directions to fail; The one-dimensional point spread implies one column in U_{FD} is in the span of the first two columns of B (in the subspace that the algorithms are trying to recover) but the other is almost orthogonal to that subspace. The results of JIT-PCA are also not perfect, but also not nearly as bad.

Figure 3 shows that the shrikage (decay) is the cause of the failure in this case. The results show that as we reduce the shrikage, the reconstruction improves.

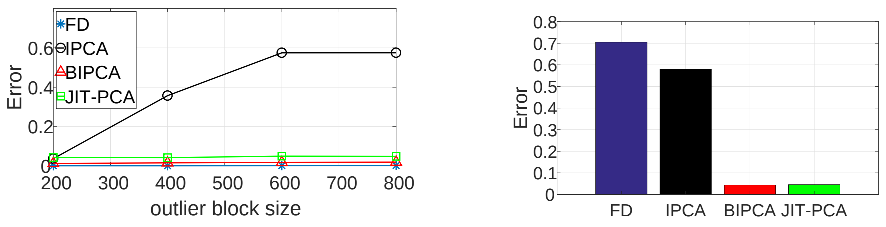

Fig. 4. Coping with a block of outliers; see text for the details of the experiments. The graph on the left is for $m = 50$ and that on the right for $m = 350$.

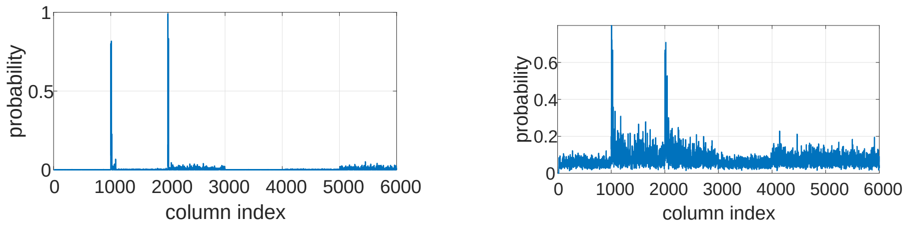

Fig. 5. The probability of a full update when a column is processed by JIT-PCA. On the left the noise ratio is 100 and the amortized coefficient of the mk term in the number of operations is 2.03. On the right the noise ratio is 10 and the coefficient is 2.45.

The second family of synthetic problems again used matrices of the form $BD+BN$ with the same structure for B, same structure for N but with standard deviation 0.1 for its entries. D is now a block matrix, $D = \begin{bmatrix} D_1 & D_2 & D_3 \end{bmatrix}$. The first and last blocks D_1 and D_3 have only 3 nonzero rows each with Gaussian entries with standard deviation 1, and D_2 has only 6 nonzero rows with a larger standard deviation 3. The nonzero rows in each block are different. The row dimension is $m = 50$ or $m = 350$, D_1 and D_3 have 10000 columns each, and D_2 has 200 to 800 columns. We ran the algorithms to produce a basis U with $k = 10$ columns and measured E_{recon} for $\bar{k} = 6$. Because D_2 has relatively few columns, $A_{\bar{6}}$ is spanned by three vectors close to the column basis of BD_1 and three more close to those of BD_3.

The graph on the left in Figure 4 shows that for $m = 50$, all the algorithms except for the basic IPCA were able to reconstruct the dominant left singular vectors of A. When we increase m to 350, Frequent Directions also fails, as shown in the bar chart on the right in Figure 4.

Figure 5 shows how effective JIT-PCA is in avoiding full updates. The input matrices are 50-by-6000 with the same $B \begin{bmatrix} D_1 & D_2 & D_3 & D_{\tilde{1}} & D_{\tilde{2}} & D_{\tilde{3}} \end{bmatrix} + BN$. The D_i and $D_{\tilde{i}}$ blocks have only 5 nonzero rows, different for different is but the same for D_i and $D_{\tilde{i}}$. Each block of D has 1000 columns. The noise ratio between the standard deviation in nonzero entries of D and entries of N is 100 or 10. We set $\bar{k} = 15$ and $k = 20$. We can see that with relatively low noise, the algorithm

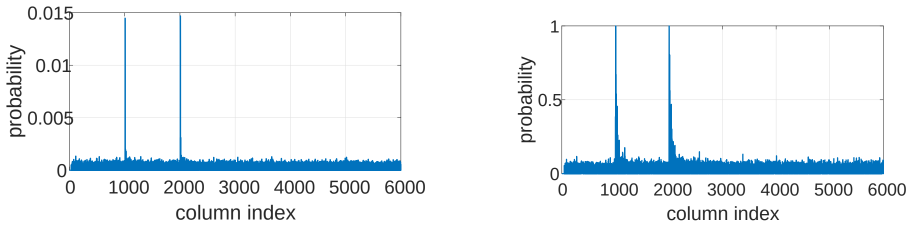

Fig. 6. The probability that BIPCA boost a column. On the left the noise ratio is 100 and on the right it is 10. The coefficient of the mk term in both cases is 5.45.

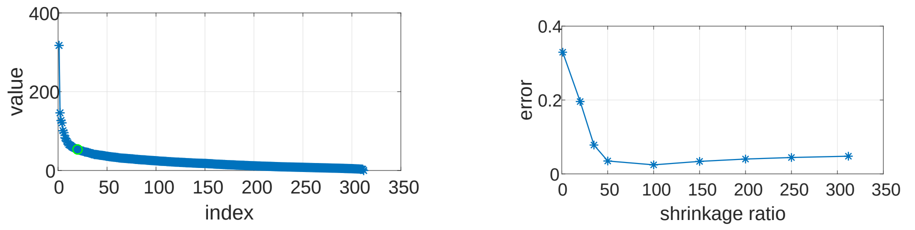

Fig. 7. The singular values of the BIRDS data set (left), with the 20th marked in green, and the reconstruction error of the Tunable Shrinkage algorithm with $k = 30$.

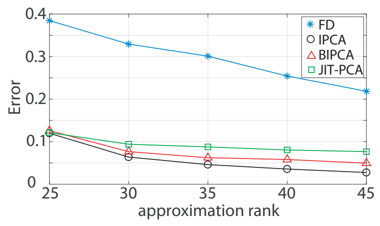

Fig. 8. Reconstruction error on the BIRDS data set for *FrequentDirections, Basic IPCA*, *BIPCA* and *JIT-PCA* as a function of the approximation rank k.

performs full updates mostly when the subspace of the columns changes and is new (not a change back to columns in a subspace seen before). With a high level of noise, the probability of full updates hovers around 10% most of the time, but is really high only when a new subspace is encountered. Figure 6 shows that the probability of boosting in BIPCA follows a similar pattern.

Figures 7 and 8 explore the behavior of the algorithms on a real-world data set called BIRDS, which was also used to demonstrate the effectiveness of Frequent Directions [7,15]. This data set has 11788 columns of dimension 312. The singular values flatten out after about 20, so we set $\bar{k} = 20$. We can see that Frequent Direction performs poorly, even with k as high as 45. The other algorithms perform better and show only little improvement after $k = 30$. We also see that shrinkage helps the performance of Frequent Directions.

References

1. M. Brand. Incremental singular value decomposition of uncertain data with missing values. In A. Heyden, G. Sparr, M. Nielsen, and P. Johansen, editors, *ECCV (1)*, volume 2350 of *Lecture Notes in Computer Science*, pages 707–720. Springer, 2002.
2. M. Brand. Fast low-rank modifications of the thin singular value decomposition. *Linear Algebra and Its Applications*, 415(1):20–30, 2006.
3. Y. Chahlaoui, K. A. Gallivan, and P. V. Dooren. An incremental method for computing dominant singular spaces. In *In Computational Information Retrieval*, pages 53–62, 2001.
4. Y. Chahlaoui, K. A. Gallivan, and P. V. Dooren. Recursive calculation of dominant singular subspaces. *SIAM J. Matrix Analysis Applications*, 25(2):445–463, 2003.
5. S. Chandrasekaran, B. Manjunath, Y.-F. Wang, J. Winkeler, and H. Zhang. An eigenspace update algorithm for image analysis. *Graphical Models and Image Processing*, 59(5):321–332, 1997.
6. A. Desai, M. Ghashami, and J. M. Phillips. Improved practical matrix sketching with guarantees. *IEEE Transactions on Knowledge and Data Engineering*, 28(7):1678–1690, 2016.
7. M. Ghashami, E. Liberty, J. M. Phillips, and D. P. Woodruff. Frequent directions: Simple and deterministic matrix sketching. *SIAM Journal on Computing*, 45(5):1762–1792, 2016.
8. T. Halpern. Fast and robust algorithms for large-scale streaming pca. Master's thesis, Tel Aviv University, Jul 2017. Available online at `http://www.tau.ac.il/~stoledo/Pubs/MSc_Tal_Halpern.pdf`.
9. A. Levey and M. Lindenbaum. Sequential karhunen-loeve basis extraction and its application to images. *IEEE Transactions on Image processing*, 9(8):1371–1374, 2000.
10. E. Liberty. Simple and deterministic matrix sketching. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 581–588. ACM, 2013.
11. B. Manjunath, S. Chandrasekaran, and Y.-F. Wang. An eigenspace update algorithm for image analysis. In *Computer Vision, 1995. Proceedings., International Symposium on*, pages 551–556. IEEE, 1995.
12. G. W. O'Brien. Information management tools for updating an svd-encoded indexing scheme. Master's thesis, University of Tennessee, Knoxville, 1994.
13. S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *science*, 290(5500):2323–2326, 2000.
14. J. B. Tenenbaum, V. De Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323, 2000.
15. C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011. Data available online at `http://www.vision.caltech.edu/visipedia/CUB-200-2011.html`.
16. H. Zha and H. D. Simon. On updating problems in latent semantic indexing. *SIAM Journal on Scientific Computing*, 21(2):782–791, 1999.