Available online at www.sciencedirect.com
SCIENCE DIRECT®
ELSEVIER
Linear Algebra and its Applications 415 (2006) 20–30
LINEAR ALGEBRA AND ITS APPLICATIONS
www.elsevier.com/locate/laa

Fast low-rank modifications of the thin singular value decomposition

Matthew Brand

MERL, 201 Broadway, Cambridge, MA 02139, USA

Received 28 May 2003; accepted 27 July 2005
Available online 27 September 2005
Submitted by Kugazov

Abstract

This paper develops an identity for additive modifications of a singular value decomposition (SVD) to reflect updates, downdates, shifts, and edits of the data matrix. This sets the stage for fast and memory-efficient sequential algorithms for tracking singular values and subspaces. In conjunction with a fast solution for the pseudo-inverse of a submatrix of an orthogonal matrix, we develop a scheme for computing a thin SVD of streaming data in a single pass with linear time complexity: A rank-r thin SVD of a $p \times q$ matrix can be computed in $O(pqr)$ time for $r \leqslant \sqrt{\min(p, q)}$.

AMS classification: 49M27; 15A18; 15A23; 65F20

Keywords: Singular value decomposition; Sequential updating; Subspace tracking

1. The singular value decomposition

The singular value decomposition (SVD) diagonalizes a real matrix $\mathbf{X} \in \mathbb{R}^{p\times q}$ via left and right rotations by orthonormal matrices $\mathbf{U} \in \mathbb{R}^{p\times p}$ and $\mathbf{V} \in \mathbb{R}^{q\times q}$, e.g.,

E-mail address: brand@merl.com
URL: http://www.merl.com/people/brand/

0024-3795/$ - see front matter
doi:10.1016/j.laa.2005.07.021

$\mathbf{U}^\top\mathbf{X}\mathbf{V} = \mathbf{S}$ is diagonal and nonnegative. Equivalently, it decomposes $\mathbf{X}$ into a sum of rank-1 matrices generated by singular value *triplets*: $\mathbf{X} = \mathbf{U}\,\mathrm{diag}(\mathbf{s})\mathbf{V}^\top = \sum_i \mathbf{u}_i s_i \mathbf{v}_i^\top$ for singular values s_i on the diagonal of $\mathbf{S}$ and singular vectors $\mathbf{u}_i$ and $\mathbf{v}_i$ drawn from the columns of $\mathbf{U}$ and $\mathbf{V}$.

The rank-r *thin* SVD restricts this sum to the r triplets having the largest-magnitude singular values. We will write this $\mathbf{U}\,\mathrm{diag}(\mathbf{s})\mathbf{V}^\top \overset{r}{\leftarrow} \mathbf{X}$ with orthonormal subspace matrices $\mathbf{U} \in \mathbb{R}^{p\times r}$, $\mathbf{V} \in \mathbb{R}^{q\times r}$ and singular value vector $\mathbf{s} \in \mathbb{R}^r \geqslant 0$. In signal processing, this reduction of $\mathbf{X}$ to a product of thin matrices is interpreted as a form of lossy compression, with the subspace matrices acting as encoding and decoding operators. By the Schmidt (later Eckart–Young–Mirsky) theorem, the thin SVD is the optimal rank-r approximation of $\mathbf{X}$ under any unitarily invariant norm, including the Frobenius norm [1]. This licenses the additional interpretation of the thin SVD as a form of noise suppression, where $\mathbf{X}$ is presumed to be a low-rank data matrix containing measurements contaminated with additive Gaussian noise.

Computing a full SVD is fundamentally an $O(pq \cdot \min(p, q))$-time problem, making decompositions of extremely large matrices infeasible. Shortly after the introduction of a practical algorithm for computing the SVD on digital computers in the 1960s [2], research turned to problems of faster methods for computing approximations such as the thin SVD, as well as updating an SVD to incorporate new data (e.g., [3,4]). In recent years the practical need for such methods has become acute and the literature has grown accordingly. Section 5 reviews the recent literature in light of the results presented below:

(1) A general identity for additive modifications of an SVD (Section 2).
(2) Specializations of this identity to give SVD updates, downdates, and rank-1 modifications with reduced computational complexity (Section 3).
(3) An expanded thin SVD and sequential updating scheme that offers a strictly linear-time thin SVD in a single pass through a data matrix (Section 4).

The last result has practical value in online settings where data must be incorporated into the SVD as it arrives, typically because the data is too large to be stored or even buffered. For example, many computer vision algorithms call for a "running" thin SVD of a video stream—effectively a data matrix with $\approx 10^5$ rows and an inexhaustible supply of columns. Financial transaction streams and network activity streams are even more demanding.

2. Additive modifications

Let real matrix $\mathbf{X} \in \mathbb{R}^{p\times q}$ have rank r and economy SVD $\mathbf{U}\mathbf{S}\mathbf{V}^\top = \mathbf{X}$ with $\mathbf{S} \in \mathbb{R}^{r\times r}$. Let $\mathbf{A} \in \mathbb{R}^{p\times c}$, $\mathbf{B} \in \mathbb{R}^{q\times c}$ be arbitrary matrices of rank c. We are interested in the SVD of the sum

$$\mathbf{X} + \mathbf{A}\mathbf{B}^\top = [\mathbf{U} \quad \mathbf{A}] \begin{bmatrix} \mathbf{S} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} [\mathbf{V} \quad \mathbf{B}]^\top \tag{1}$$

expressed as modifications to $\mathbf{U}, \mathbf{S}, \mathbf{V}$. We are most interested in the case where $\operatorname{rank}(\mathbf{X} + \mathbf{A}\mathbf{B}^\top) \leqslant r + c < \min(p, q)$, so that $\mathbf{U}$, $\mathbf{V}$, $\mathbf{A}$, $\mathbf{B}$ are tall thin matrices. However, what follows is completely general.

Let $\mathbf{P}$ be an orthogonal basis of the column space of $(\mathbf{I} - \mathbf{U}\mathbf{U}^\top)\mathbf{A}$—the component of $\mathbf{A}$ that is orthogonal to $\mathbf{U}$—and set $\mathbf{R}_\mathbf{A} \doteq \mathbf{P}^\top(\mathbf{I} - \mathbf{U}\mathbf{U}^\top)\mathbf{A}$. Note that $\operatorname{cols}(\mathbf{P}) = \operatorname{rows}(\mathbf{R}_\mathbf{A}) = \operatorname{rank}((\mathbf{I} - \mathbf{U}\mathbf{U}^\top)\mathbf{A}) \leqslant c$, and may be zero. The relationships between these matrices is summarized as

$$[\mathbf{U} \quad \mathbf{A}] = [\mathbf{U} \quad \mathbf{P}] \begin{bmatrix} \mathbf{I} & \mathbf{U}^\top\mathbf{A} \\ \mathbf{0} & \mathbf{R}_\mathbf{A} \end{bmatrix}. \tag{2}$$

Though similar to a QR decomposition, $\mathbf{R}_\mathbf{A}$ need not be upper-triangular or square. Similarly, let $\mathbf{Q}\mathbf{R}_\mathbf{B} = (\mathbf{I} - \mathbf{V}\mathbf{V}^\top)\mathbf{B}$. Substituting Eq. (2) into Eq. (1), we have

$$\mathbf{X} + \mathbf{A}\mathbf{B}^\top = [\mathbf{U} \quad \mathbf{P}]\mathbf{K}[\mathbf{V} \quad \mathbf{Q}]^\top \tag{3}$$

a product of two orthonormal matrices and

$$\mathbf{K} \doteq \begin{bmatrix} \mathbf{I} & \mathbf{U}^\top\mathbf{A} \\ \mathbf{0} & \mathbf{R}_\mathbf{A} \end{bmatrix} \begin{bmatrix} \mathbf{S} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{V}^\top\mathbf{B} \\ \mathbf{0} & \mathbf{R}_\mathbf{B} \end{bmatrix}^\top = \begin{bmatrix} \mathbf{S} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} + \begin{bmatrix} \mathbf{U}^\top\mathbf{A} \\ \mathbf{R}_\mathbf{A} \end{bmatrix} \begin{bmatrix} \mathbf{V}^\top\mathbf{B} \\ \mathbf{R}_\mathbf{B} \end{bmatrix}^\top, \tag{4}$$

which is usually small, highly structured, and sparse. It follows immediately that diagonalizing $\mathbf{K}$ as $\mathbf{U}'^\top\mathbf{K}\mathbf{V}' = \mathbf{S}'$ gives rotations $\mathbf{U}'$ and $\mathbf{V}'$ of the extended subspaces $[\mathbf{U} \quad \mathbf{P}]$ and $[\mathbf{V} \quad \mathbf{Q}]$ such that

$$\mathbf{X} + \mathbf{A}\mathbf{B}^\top = ([\mathbf{U} \quad \mathbf{P}]\mathbf{U}')\mathbf{S}'([\mathbf{V} \quad \mathbf{Q}]\mathbf{V}')^\top \tag{5}$$

is the desired SVD.

The rest of this paper develops scenarios where Eq. (5) provides a computationally attractive route to low-rank modifications of a thin SVD. For column updates and downdates of $\mathbf{X}$, the $\mathbf{K}$ matrix is sparse and easily diagonalized. Indeed, for low rank matrices and those having good low-rank approximations, one can compute a thin SVD through sequential column updates in linear time.

3. Rank-1 modifications

Here we develop some special efficiencies offered by rank-1 modifications. For the SVD of $\mathbf{U}\mathbf{S}\mathbf{V}^\top + \mathbf{a}\mathbf{b}^\top$ with vectors $\mathbf{a} \in \mathbb{R}^p$ and $\mathbf{b} \in \mathbb{R}^q$, Eq. (2) can be effected in a partial step of the modified Gram–Schmidt algorithm:

$$\mathbf{m} \doteq \mathbf{U}^\top\mathbf{a}; \quad \mathbf{p} \doteq \mathbf{a} - \mathbf{U}\mathbf{m}; \quad R_a = \|\mathbf{p}\|; \quad \mathbf{P} = R_a^{-1} \cdot \mathbf{p} \tag{6}$$

and similarly

$$\mathbf{n} \doteq \mathbf{V}^\top\mathbf{b}; \quad \mathbf{q} \doteq \mathbf{b} - \mathbf{V}\mathbf{n}; \quad R_b = \|\mathbf{q}\|; \quad \mathbf{Q} = R_b^{-1} \cdot \mathbf{q}. \tag{7}$$

Table 1
Common operations on the last column or on all columns expressed as rank-1 modifications of an SVD $\mathbf{USV}^\top = \mathbf{X}$ to give $\mathbf{U}'\mathbf{S}'\mathbf{V}'^\top = \mathbf{X} + \mathbf{ab}^\top$

Operation	Known	Desired	$\mathbf{a}$	$\mathbf{b}^\top$
Update	$\mathbf{US}[\mathbf{V}^\top \quad \mathbf{0}] = [\mathbf{X} \quad \mathbf{0}]$	$\mathbf{U}'\mathbf{S}'\mathbf{V}'^\top = [\mathbf{X} \quad \mathbf{c}]$	$\mathbf{c}$	$[0, \ldots, 0, 1]$
Downdate	$\mathbf{USV}^\top = [\mathbf{X} \quad \mathbf{c}]$	$\mathbf{U}'\mathbf{S}'\mathbf{V}'^\top = \mathbf{X}$	$-\mathbf{c}$	$[0, \ldots, 0, 1]$
Revise	$\mathbf{USV}^\top = [\mathbf{X} \quad \mathbf{c}]$	$\mathbf{U}'\mathbf{S}'\mathbf{V}'^\top = [\mathbf{X} \quad \mathbf{d}]$	$\mathbf{d} - \mathbf{c}$	$[0, \ldots, 0, 1]$
Recenter	$\mathbf{USV}^\top = \mathbf{X}$	$\mathbf{U}'\mathbf{S}'\mathbf{V}'^\top = \mathbf{X}(\mathbf{I} - \frac{1}{q}\mathbf{1}\mathbf{1}^\top)$	$-\frac{1}{q}\mathbf{X1}$	$\mathbf{1}^\top \doteq [1, \ldots, 1]$

The rediagonalization problem of Eq. (4) simplifies to

$$\mathbf{K} = \begin{bmatrix} \mathbf{S} & \mathbf{0} \\ \mathbf{0} & 0 \end{bmatrix} + \begin{bmatrix} \mathbf{m} \\ \|\mathbf{p}\| \end{bmatrix} \begin{bmatrix} \mathbf{n} \\ \|\mathbf{q}\| \end{bmatrix}^\top \tag{8}$$

a diagonal + rank-1 matrix, amenable to special treatment.

Table 1 shows how updating, downdating, and revising individual columns of the SVD are expressed as specializations of this scheme. Each offers further opportunities for reducing computation.

For example, to update $\mathbf{X}$ with a new column $\mathbf{c} \in \mathbb{R}^p$, one appends a row of zeros to $\mathbf{V}$ and then computes the rank-1 modification $\mathbf{U}'\mathbf{S}'\mathbf{V}'^\top = [\mathbf{X} \quad \mathbf{0}] + \mathbf{c}[0, \ldots, 0, 1]$. In this case, $\mathbf{n} = \mathbf{0}$, so Eq. (4) asks us only to rediagonalize the broken-arrow matrix

$$\mathbf{K} = \begin{bmatrix} \mathbf{S} & \mathbf{m} \\ \mathbf{0} & \|\mathbf{p}\| \end{bmatrix}, \tag{9}$$

which can be done in $O(r^2)$ time [5].

Similarly, one downdates the SVD by zeroing a column. In this case Eq. (4) simplifies to

$$\mathbf{K} = \begin{bmatrix} \mathbf{S} & \mathbf{0} \\ \mathbf{0} & 0 \end{bmatrix} \left(\mathbf{I} - \begin{bmatrix} \mathbf{Sn} \\ 0 \end{bmatrix} \begin{bmatrix} \mathbf{n} \\ \sqrt{1 - \mathbf{n}^\top \mathbf{n}} \end{bmatrix}^\top \right), \tag{10}$$

$\mathbf{P}$ is unused, and $\mathbf{Q} = (\mathbf{b} - \mathbf{Vn})/\sqrt{1 - \mathbf{n}^\top \mathbf{n}}$ is used only if updating $\mathbf{V}$. Note that downdating the ith column only requires knowing the ith row of $\mathbf{V}$.

4. An extended decomposition for reduced complexity

In a naïve implementation of the update, the QR-like decomposition of Eq. (2) takes $O(p(r + c)^2)$ time, the rediagonalization of Eq. (4) takes $O((r + c)^3)$ time, and the rotations of the subspaces takes $O((p + q)(c + r)^2)$ time. In the setting of a rank-1 update of a fixed-rank SVD, these times can be reduced to $O(pr)$, $O(r^2)$, and $O(r^3)$, respectively, by expanding the MGS as shown above, performing sparse diagonalizations, and deferring the costly subspace rotations as follows.

Instead of rotating the large singular vector matrices as prescribed in Eq. (5), we leave the SVD decomposed into the five matrices

$$\mathbf{U}_{p\times r}\cdot\mathbf{U}'_{r\times r}\cdot\mathbf{S}_{r\times r}\cdot\mathbf{V}'^{\top}_{r\times r}\cdot\mathbf{V}^{\top}_{q\times r} \tag{11}$$

with orthonormal $\mathbf{U}\cdot\mathbf{U}'$, $\mathbf{V}\cdot\mathbf{V}'$, $\mathbf{U}$, and $\mathbf{U}'$ (but not $\mathbf{V}'$ or $\mathbf{V}$). The large outer matrices only record the span of the left and right subspaces and are built by appending columns to $\mathbf{U}$ and rows to $\mathbf{V}$. The transforms of these subspace bases that make $\mathbf{S}$ diagonal are maintained in the much smaller $\mathbf{U}'$, $\mathbf{V}'$ matrices. This makes the update much faster and eliminates the numerical error that would accumulate if the bases specified by the tall $\mathbf{U}$, $\mathbf{V}$ matrices were rotated on each update. The rest of this section details the updates of the left and right subspaces.

4.1. Updating the left subspace

Let $\mathbf{K}$ and $\mathbf{p}$ be defined as above, and let orthogonal $\mathbf{C}$, $\mathbf{D}\in\mathbb{R}^{(r+1)\times(r+1)}$ diagonalize $\mathbf{K}$ as $\mathbf{CS}'\mathbf{D}^{\top}=\mathbf{K}$. From Eq. (5), the left-side update must satisfy $\mathbf{U}_{\text{new}}\mathbf{U}'_{\text{new}}=[\mathbf{U}_{\text{old}}\quad\mathbf{p}]\mathbf{U}'_{old}\mathbf{C}$. If $\mathbf{K}$ has rank r (i.e. the update is not rank-increasing), then $\mathbf{C}$ has the form $\mathbf{C}=\begin{bmatrix}\mathbf{C}_{1:r,1:r} & \mathbf{0}\\ \mathbf{0} & 1\end{bmatrix}$ and the left of side Eq. (11) is simply updated

$$\mathbf{U}'\leftarrow\mathbf{U}'\mathbf{C}_{1:r,1:r}. \tag{12}$$

Otherwise the rank-increasing update is

$$\mathbf{U}'\leftarrow\begin{bmatrix}\mathbf{U}' & \mathbf{0}\\ \mathbf{0} & 1\end{bmatrix}\mathbf{C};\quad \mathbf{U}\leftarrow[\mathbf{U}\quad\mathbf{p}]. \tag{13}$$

The appends preserve orthogonality of $\mathbf{U}$ because $\mathbf{U}^{\top}\mathbf{p}=\mathbf{0}$ by construction.

Over thousands or millions of updates, the multiplications may erode the orthogonality of $\mathbf{U}'$ through numerical error, albeit slowly because these matrices remain small. Loss of orthogonality can be contained and corrected by occasionally forming the small product $\mathbf{U}'\mathbf{S}'\in\mathbb{R}^{r\times r}$, refactoring via SVD, and premultiplying the resulting right subspace into $\mathbf{V}'$. It is an open question how often this is necessary to guarantee a certain overall level of numerical precision; it does not change the overall complexity.

4.2. Updating the right subspace

The right-side updates are somewhat more complicated because updates add *rows* to $\mathbf{V}$ while guaranteeing that the *columns* of the product $\mathbf{VV}'$ are orthogonal. From Eq. (5), the right-side update must satisfy

$$\mathbf{V}_{\text{new}}\mathbf{V}'_{\text{new}}=\begin{bmatrix}\mathbf{V}_{\text{old}}\mathbf{V}'_{\text{old}} & \mathbf{0}\\ \mathbf{0} & 1\end{bmatrix}\mathbf{D}. \tag{14}$$

To do so, it is convenient to calculate and update a small pseudo-inverse matrix $\mathbf{V}'^{+}$. When the update is rank-increasing, the right-hand side update is simply

$$\mathbf{V}'_{\text{new}} \leftarrow \begin{bmatrix} \mathbf{V}'_{\text{old}} & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix} \mathbf{D}; \quad (\mathbf{V}'^{+})_{\text{new}} \leftarrow \mathbf{D}^{\top} \begin{bmatrix} (\mathbf{V}'^{+})_{\text{old}} & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix};$$
$$\mathbf{V}_{\text{new}} \leftarrow \begin{bmatrix} \mathbf{V}_{\text{old}} & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix} \tag{15}$$

because $\begin{bmatrix} \mathbf{V}\mathbf{V}' & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix} \mathbf{D} = \begin{bmatrix} \mathbf{V} & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{V}' & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix} \mathbf{D}$ and $\mathbf{D}$ is orthogonal.

When the rank does not increase, the last column of $\mathbf{D}$ represents an unused subspace dimension and should be suppressed. This licenses another optimization: Split $\mathbf{D} \in \mathbb{R}^{(r+1)\times r} \rightarrow \begin{bmatrix} \mathbf{W} \in \mathbb{R}^{r\times r} \\ \mathbf{w} \in \mathbb{R}^{1\times r} \end{bmatrix}$ where submatrix $\mathbf{W}$ is a linear transform that will be applied to $\mathbf{V}'$, and row-vector $\mathbf{w}$ is the subspace projection of the new data vector. The resulting right-side update

$$\mathbf{V}'_{\text{new}} \leftarrow \mathbf{V}'_{\text{old}}\mathbf{W}; \quad (\mathbf{V}'^{+})_{\text{new}} \leftarrow \mathbf{W}^{+}(\mathbf{V}'^{+})_{\text{old}}; \quad \mathbf{V}_{\text{new}} \leftarrow \begin{bmatrix} \mathbf{V}_{\text{old}} \\ \mathbf{w}(\mathbf{V}'^{+})_{\text{new}} \end{bmatrix} \tag{16}$$

can be verified by substitution into Eq. (14).

Conveniently, the pseudo-inverse $\mathbf{W}^{+}$ can be computed in $O(r^2)$-time using only matrix–vector and vector–vector products via the identity

$$\mathbf{W}^{+} = \mathbf{W}^{\top} + \frac{\mathbf{w}^{\top}}{1 - \|\mathbf{w}\|^2}(\mathbf{w}\mathbf{W}^{\top}), \tag{17}$$

which is a special case of the following result for submatrices of an orthonormal matrix:

Proposition 1. *Let tall matrix* $\mathbf{D} = \begin{bmatrix} \mathbf{W} \\ \mathbf{Y} \end{bmatrix}$ *have orthonormal columns, such that* $\mathbf{D}^{\top}\mathbf{D} = \mathbf{W}^{\top}\mathbf{W} + \mathbf{Y}^{\top}\mathbf{Y} = \mathbf{I}$. *Let* $\mathbf{U}^{\top}\mathbf{Y}\mathbf{V} = \mathbf{S}$ *be a diagonalization of* $\mathbf{Y}$ *with* $\mathbf{U}^{\top}\mathbf{U} = \mathbf{V}^{\top}\mathbf{V} = \mathbf{I}$ *and* $\mathbf{S}$ *having positive values on its diagonal. Then*

$$\mathbf{W}^{+} = \mathbf{W}^{\top} + \mathbf{V}\mathbf{S}^2(\mathbf{I} - \mathbf{S}^2)^{+}\mathbf{V}^{\top}\mathbf{W}^{\top} \tag{18}$$

with

$$\mathbf{W}^{+} = \mathbf{W}^{\top} + \mathbf{Y}^{\top}(\mathbf{I} - \mathbf{Y}\mathbf{Y}^{\top})^{+}\mathbf{Y}\mathbf{W}^{\top} \tag{19}$$

when $\mathbf{Y}$ *is a square or wide matrix and*

$$\mathbf{W}^{+} = \mathbf{W}^{\top} + \frac{\mathbf{y}^{\top}}{1 - \|\mathbf{y}\|^2}(\mathbf{y}\mathbf{W}) \tag{20}$$

when $\mathbf{y} = \mathbf{Y}$ *is a row vector.*

This is essentially the Sherman–Woodbury–Morrison formula applied to the cosine–sine decomposition. Readers desiring more detail can find a proof in Appendix A.

4.3. Complexity and speed

The expanded update eliminates the costliest and numerically most vulnerable steps of the update—the rotation and re-orthogonalization of $\mathbf{U}$, $\mathbf{V}$. The time complexity falls to $O(pr + r^3)$ for each update, with an overall complexity of $O(pqr + qr^3) = O(pqr)$ for the entire thin SVD, assuming that the desired rank is small relative to the matrix, specifically $r = O(\sqrt{p})$. For a high-dimensional low-rank matrices, we effectively have a *linear-time* SVD algorithm. If, furthermore, the data is streamed through the CPU, the update requires only $O((p + q)r)$ space to store the current SVD and data vector, i.e., it is sublinear in the number of data matrix elements.

The predicted linear scaling behavior is borne out empirically in trials with large dense random matrices: The proposed incremental SVD exhibits linear scaling behavior with size and rank (see Fig. 1 for details) and is orders of magnitude faster than Lanczos methods (as implemented in Matlab 5 and 6). Part of the dramatic speed-up may be attributed to the fact that the working storage of the proposed method can be kept entirely in the CPU's onboard cache, while the Lanczos methods make heavy use of the computer's bus. The two algorithms illustrate a trade-off between speed and accuracy: The multi-pass Lanczos method achieves slightly better numerical accuracy but the single-pass incremental method is orders of magnitude faster.

5. Related work

Modern thin SVD and SVD updating methods are generally based on symmetric eigenvalue perturbations or projections of the problem into subspaces. Many of these methods can be derived as special cases of the framework given in Section 2. Businger [3] proposed an update adapted from QR-decomposition updates based on annihilations via Givens rotations. Bunch and Nielsen [4] and Gu and Eisenstat [5] developed updates based on symmetric eigenvalue updates of the data's gram matrix. As remarked in a Gu and Eisenstat [8] downdating paper, all such methods can be as expensive as computing a new SVD from scratch. Noting that updating a thin SVD offers more attractive economies, Berry et al. [9] proposed to project the updating problem into a previously estimated low-rank subspace, but the resulting updates ignore any component of new data that lies outside that subspace. This was remediated by Zha and Simon [10]; their solution requires a full SVD of a dense matrix on each update. Witter and Berry [11] also introduced a related downdate. Chandrasekaran et al. [6] and Levy and Lindenbaum [7] proposed sequential eigenspace (left subspace) updates based on analyses that, with some algebra, can be rendered as special cases of Eq. (5) (with binary-valued $\mathbf{B}$). The Levy and Lindenbaum [7] method takes linear time for $r \ll \min(p, q)$ but appears to exhibit quadratic scaling well before $r = \sqrt{p}$. All of these methods require expensive multiplications of large subspace matrices, making them considerably slower than our proposed method, and also making loss of orthogonality an important numerical issue. In addition, the eigenspace updates

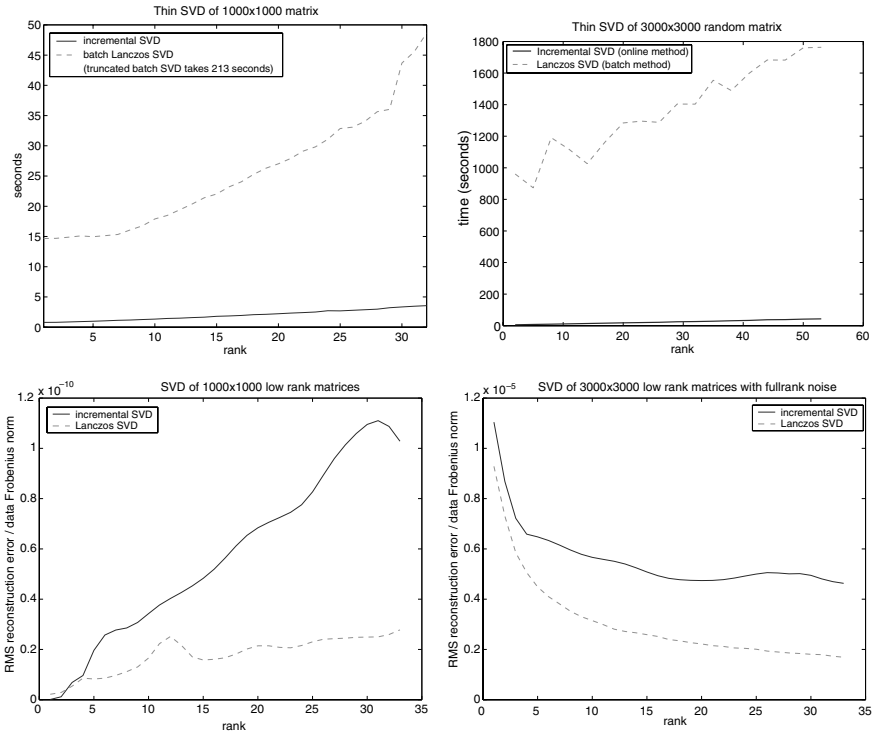

Fig. 1 Run-time (top) and matrix reconstruction error (bottom) of sequential SVD updating (solid line) versus batch Lanczos (dashed line), as a function of the number of singular vector/value triplets computed from a random matrix. Each datapoint represents the average of 100 trials. The sequential update shows clear linear scaling and speed advantages. The experiment graphed at left employed low-rank matrices; at right, full-rank matrices having reasonable low-rank approximations. The proposed method exhibits similar speed/scaling advantages over other updating algorithms (e.g., [6,7]), but produces more accurate results. Experiments were performed in Matlab 5 and 6 on an AlphaServer with a 600 MHz CPU, 10 G RAM, and a fast crossbar bus.

(e.g., [6,7]) presume centered (zero-mean) data; if the data stream does not comply one would need to use the recentering operator given in Table of this paper in conjunction with an estimator of the data mean.

6. Discussion: Approximation, sublinear, and tracking algorithms

The $O(pqr)$ time complexity result of section 4 rests on the assumption that the rank of the thin SVD holds at $r \leqslant O(\sqrt{p})$. If the data stream has higher numerical rank, it may eventually be desirable to suppress rank-increasing updates by some form of truncation. One possibility is to ignore the component of an update the lies outside the current subspace by setting $\|\mathbf{p}\| \to 0$ in Eq. (9) and using the optimizations that

follow. The optimal greedy tactic is to allow a rank-increasing update but immediately discard the new low-order singular value triplet. Clearly if a matrix has rank $> r$ then either truncating tactic will eventually discard variance that would be retained in an optimal low-rank approximation of the data matrix.

It is worth asking how suboptimal a greedy truncation scheme can be. In principle, an adversary could order a data stream such that some significant data trends only appear late in the stream, where they may be partly lost to truncation. One obvious strategy for such poorly behaved data streams is to exploit the low complexity of the update to compute a thin SVD of higher rank than ultimately desired, thereby allocating more memory to accommodate novel directions of variance in the data stream. The excess singular value triplets can be discarded later. On the other hand, if the vectors are random samples from a stationary data source, then on average the thin SVD will orient to approximate the true singular vectors and values, even if r is set too small. This can be established and quantified using the same central-limit-theorem arguments that justify approximate SVDs made by randomly sampling a subset of rows and/or columns from a large matrix (see [12]). Indeed, the fast update can be combined with random sampling to give sublinear algorithms for massive data streams.

If the data source is nonstationary (like the adversary above), the update is trivially adapted to subspace tracking by causing the singular values to decay between updates: $\mathbf{s} \to \lambda\mathbf{s}$ for $0 < \lambda < 1$. This allows the subspace to rotate toward new directions of variance in the data stream, which would otherwise by dwarfed by the "weight of experience" already recorded by the singular values. Similarly, the rank capacity r can be adjusted online in response to data vectors having a significant component outside the estimated subspace.

Acknowledgment

I am grateful to Andrew Knyazev and to anonymous reviewers for constructive comments on earlier drafts of this article.

Appendix

A. Proof of partitioned pseudo-inverse update

Without loss of generality, suppress any all-zero rows in $\mathbf{W}$ and $\mathbf{Y}$ so that the singular value decompositions $\mathbf{U}_1\mathbf{S}_1\mathbf{V}_1^\top = \mathbf{W} \in \mathbb{R}^{r_1 \times c}$ and $\mathbf{U}_2\mathbf{S}_2\mathbf{V}_2^\top = \mathbf{Y} \in \mathbb{R}^{r_2 \times c}$ have no zero-valued singular values. Restating the orthogonal sum in terms of the SVDs, we have $\mathbf{V}_1\mathbf{S}_1^2\mathbf{V}_1^\top = \mathbf{W}^\top\mathbf{W} = \mathbf{I} - \mathbf{Y}^\top\mathbf{Y} = \mathbf{I} - \mathbf{V}_2\mathbf{S}_2^2\mathbf{V}_2^\top$. Pre- and post-multiplying by $\mathbf{V}_1^\top, \mathbf{V}_1$ gives $\mathbf{S}_1^2 = \mathbf{I} - \mathbf{V}_1^\top\mathbf{V}_2\mathbf{S}_2^2\mathbf{V}_2^\top\mathbf{V}_1$ and similarly $\mathbf{S}_2^2 = \mathbf{I} - \mathbf{V}_2^\top\mathbf{V}_1\mathbf{S}_1^2\mathbf{V}_1^\top\mathbf{V}_2$. The fact that diagonality is preserved by $\mathbf{V}_1^\top\mathbf{V}_2 \in \mathbb{R}^{r_1 \times r_2}$ and its transpose implies

that $\mathbf{V}_1^\top \mathbf{V}_2$ has at most one nonzero element in each row and column. This in turn implies that every column in $\mathbf{V}_1$ is orthogonal to at least $r_2 - 1$ columns in $\mathbf{V}_2$, and vice versa, making $r_1 r_2 - \min(r_1, r_2)$ orthogonality relationships between $\mathbf{V}_1$ and $\mathbf{V}_2$ and $r_1(r_1-1)/2 + r_2(r_2-1)/2$ orthogonality relationships within $\mathbf{V}_1$ and $\mathbf{V}_2$. Since $r_1 + r_2 \geqslant c$, this exhausts all the $c(c-1)/2$ possible orthogonalities in a c-dimensional basis, therefore any pair of columns $\{\mathbf{v}_i \in \mathbf{V}_1,\ \mathbf{v}_j \in \mathbf{V}_2\}$ that is not orthogonal must be identical (up to a flippable sign). I.e. $\mathbf{v}_1^\top \mathbf{v}_2 \neq 0 \Rightarrow \mathbf{v}_1 = \pm \mathbf{v}_2$. After making appropriate sign flips, the concatenation $[\mathbf{V}_1, \mathbf{V}_2] \in \mathbb{R}^{c \times (r_1 + r_2)}$ is an orthogonal basis of $\mathbb{R}^c$ with duplicate columns, and the product $\mathbf{V}_1^\top \mathbf{V}_2 \in \{0, 1\}^{r_1 \times r_2}$ can be viewed as a submatrix of a permutation matrix, having some all-zero rows or columns along its shorter axis. It follows that the equality $\mathbf{S}_1^2 = \mathbf{I} - \mathbf{V}_1^\top \mathbf{V}_2 \mathbf{S}_2^2 \mathbf{V}_2^\top \mathbf{V}_1$ can be separated into independent equations $s_i^2 = 1 - s_j^2$ for singular values $s_i \in \mathrm{diag}(\mathbf{S}_1)$ and $s_j \in \mathrm{diag}(\mathbf{S}_2)$ where $(\mathbf{V}_1^\top \mathbf{V}_2)_{ij} = \mathbf{v}_i^\top \mathbf{v}_j = 1$, while those singular values not put into correspondence by $\mathbf{V}_1^\top \mathbf{V}_2$ are unitary, i.e. if $(\mathbf{V}_1^\top \mathbf{V}_2)_{ij} = 0$ for all j, then $s_i = 1$. Pseudo-inverting both sides of each independent scalar equation we obtain $(s_i^2)^+ = (1 - s_j^2)^+ = s_j^2 (1 - s_j^2)^+ + 1$ for all s_j within the spectral radius of an orthogonal matrix, including $s_j = 1 \Rightarrow s_i = 0$. and $s_j \neq 1$. The correspondences between the singular vectors and singular values is summarized by the following chain of equalities for the symmetric pseudo-inverse:

$$\mathbf{V}_1 (\mathbf{S}_1^+)^2 \mathbf{V}_1^\top = (\mathbf{W}^\top \mathbf{W})^+ = (\mathbf{I} - \mathbf{Y}^\top \mathbf{Y})^+ \tag{A.1}$$

$$= (\mathbf{I} - \mathbf{V}_2 \mathbf{S}_2^2 \mathbf{V}_2^\top)^+ \tag{A.2}$$

$$= \mathbf{V}_2 \mathbf{S}_2^2 (\mathbf{I} - \mathbf{S}_2^2)^+ \mathbf{V}_2^\top + \mathbf{I}. \tag{A.3}$$

To verify, pre- and post-multiply by $\mathbf{V}_1^\top$, $\mathbf{V}_1$ or $\mathbf{V}_2^\top$, $\mathbf{V}_2$ to recover the separate pseudo-inverse equalities. This is then substituted into an expansion of the asymmetric pseudo-inverse,

$$\mathbf{W}^+ = (\mathbf{W}^\top \mathbf{W})^+ \mathbf{W}^\top \tag{A.4}$$

$$= (\mathbf{I} + (\mathbf{W}^\top \mathbf{W})^+ - \mathbf{I}) \mathbf{W}^\top \tag{A.5}$$

$$= \mathbf{W}^\top + ((\mathbf{W}^\top \mathbf{W})^+ - \mathbf{I}) \mathbf{W}^\top \tag{A.6}$$

$$= \mathbf{W}^\top + (\mathbf{V}_2 \mathbf{S}_2^2 (\mathbf{I} - \mathbf{S}_2^2)^+ \mathbf{V}_2^\top + \mathbf{I} - \mathbf{I}) \mathbf{W}^\top \tag{A.7}$$

to prove the proposition. For the special case of $r_2 \leqslant c$,

$$\mathbf{V}_2 \mathbf{S}_2 (\mathbf{I} - \mathbf{S}_2^2)^+ \mathbf{S}_2 \mathbf{V}_2^\top = \mathbf{V}_2 \mathbf{S}_2 \mathbf{U}_2^\top \mathbf{U}_2 (\mathbf{I} - \mathbf{S}_2^2)^+ \mathbf{U}_2^\top \mathbf{U}_2 \mathbf{S}_2 \mathbf{V}_2^\top \tag{A.8}$$

$$= \mathbf{Y}^\top \mathbf{U}_2 (\mathbf{I} - \mathbf{S}_2 \mathbf{V}_2^\top \mathbf{V}_2 \mathbf{S}_2)^+ \mathbf{U}_2^\top \mathbf{Y} \tag{A.9}$$

$$= \mathbf{Y}^\top (\mathbf{I} - \mathbf{U}_2 \mathbf{S}_2 \mathbf{V}_2^\top \mathbf{V}_2 \mathbf{S}_2 \mathbf{U}_2^\top)^+ \mathbf{Y} \tag{A.10}$$

$$= \mathbf{Y}^\top (\mathbf{I} - \mathbf{Y}\mathbf{Y}^\top) \mathbf{Y}, \tag{A.11}$$

where $\mathbf{U}_2$ enters the pseudo-inverse without generating extra terms because it is square.

References

[1] L. Mirsky, Symmetric gauge functions and unitarily invariant norms, Quart. J. Math. Oxford 11 (1960) 50–59.
[2] Gene Golub, Arthur van Loan, Matrix Computations, Johns Hopkins U. Press, 1996.
[3] P. Businger, Updating a singular value decomposition, BIT 10 (1970) 376–385.
[4] J.R. Bunch, C.P. Nielsen, Updating the singular value decomposition, Numer. Math. 31 (1978) 111–129.
[5] M. Gu, S.C. Eisenstat, A stable and fast algorithm for updating the singular value decomposition, Tech. Report YALEU/DCS/RR-966, Department of Computer Science, Yale University, New Haven, CT, 1993.
[6] S. Chandrasekaran, B.S. Manjunath, Y.F. Wang, J. Winkeler, H. Zhang, An eigenspace update algorithm for image analysis, Graphical Models Image Process.: GMIP 59 (5) (1997) 321–332.
[7] A. Levy, M. Lindenbaum, Sequential Karhunen–loeve basis extraction and its application to images, Technical Report CIS9809, Technion, 1998.
[8] M. Gu, S. Eisenstat, Downdating the singular value decomposition, SIAM J. Matrix Anal. Appl. 16 (1995) 793–810.
[9] M. Berry, S. Dumais, T. Letsche, Computational methods for intelligent information access, in: Proc. Supercomputing'95, 1995.
[10] H. Zha, H.D. Simon, On updating problems in latent semantic indexing, SIAM J. Sci. Comput. 21 (2) (1999) 782–791.
[11] D.I. Witter, M.W. Berry, Downdating the latent semantic indexing model for conceptual information retrieval, The Computer J. 410 (1998) 1998.
[12] A.M. Frieze, R. Kannan, S. Vempala, Fast Monte-Carlo algorithms for finding low-rank approximations, in: IEEE Symposium on Foundations of Computer Science, 1998, pp. 370–378.