Contents lists available at ScienceDirect

Computers and Mathematics with Applications

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

A fast and stable algorithm for downdating the singular value decomposition

Jieyuan Zhang[a,c], Shengguo Li[b,*], Lizhi Cheng[a,c], Xiangke Liao[b], Guangquan Cheng[d]

[a] College of Science, National University of Defense Technology, Changsha 410073, China
[b] College of Computer Science, National University of Defense Technology, Changsha 410073, China
[c] The State Key Laboratory for High Performance Computation, National University of Defense Technology, Changsha 410073, China
[d] Science and Technology on Information Systems Engineering Laboratory, National University of Defense Technology, Changsha 410073, China

ARTICLE INFO

Article history:
Received 31 January 2014
Received in revised form 18 August 2014
Accepted 13 September 2014
Available online 7 October 2014

Keywords:
Downdating SVD
HSS matrices
Cauchy-like matrices
Numerical stability

ABSTRACT

In this paper, we modify a classical downdating SVD algorithm and reduce its complexity significantly. We use a structured low-rank approximation algorithm to compute an hierarchically semiseparable (HSS) matrix approximation to the eigenvector matrix of a diagonal matrix plus rank-one modification. The complexity of our downdating algorithm is analyzed. We further show that the structured low-rank approximation algorithm is backward stable. Numerous experiments have been done to show the efficiency of our algorithm. For some matrices with large dimensions, our algorithm can be much faster than that using plain matrix–matrix multiplication routine in Intel MKL in both sequential and parallel cases.

1. Introduction

Let A be an $M \times N$ real matrix, $M > N$, and assume that its SVD is defined as

$$A = U\Sigma V^T = \begin{pmatrix} U_1 & U_2 \end{pmatrix} \begin{pmatrix} \Sigma_1 \\ 0 \end{pmatrix} V^T = U_1 \Sigma_1 V^T, \tag{1}$$

where $U = \begin{pmatrix} U_1 & U_2 \end{pmatrix} \in R^{M\times M}$ and $V \in R^{N\times N}$ are orthogonal matrices, $U_1 \in R^{M\times N}$, and $\Sigma_1 \in R^{N\times N}$ is a diagonal matrix, whose diagonal entries are the singular values of A. In this paper, we propose a fast algorithm to compute the SVD of A' satisfying

$$A = \begin{pmatrix} A' \\ a^T \end{pmatrix}, \tag{2}$$

where a^T is the bottom row of A. Note that the case of deleting a column of A can be considered similarly.

* Corresponding author. Tel.: +86 13687340094.
E-mail address: nudtlsg@nudt.edu.cn (S. Li).

http://dx.doi.org/10.1016/j.camwa.2014.09.008
0898-1221/© 2014 Elsevier Ltd. All rights reserved.

Computing the SVD of A' is well-known as the *downdating SVD* problem [1], which has been widely applied in signal processing, image analysis and least square problems (see [2–5] for more details). When using Latent Semantic Indexing (LSI) for information retrieval, some outdated terms and/or documents can be removed from the term-by-document matrix [6], which is also reduced to the downdating SVD problem.

Most algorithms for downdating SVD (see [1,3]) are expensive, costing $O(N^3)$ flops. For simplicity of analysis, we assume that M and N are in the same order, $M = O(N)$. In this paper, we show that the complexity of downdating SVD problem (2) can be reduced to $O(N^2r)$ flops,

$$A' = U'\Sigma'V'^T = \begin{pmatrix} U_1' & U_2' \end{pmatrix} \begin{pmatrix} \Sigma_1' \\ 0 \end{pmatrix} V'^T = U_1'\Sigma_1'V'^T, \tag{3}$$

where r is a moderate constant (see Section 3.2 for details), and $U_1' \in R^{(M-1)\times N}$. The technique used in this paper is similar to that in [7,8]. The main observation is that some intermediate matrices appeared in the algorithm in [1] are Cauchy-like matrices and have off-diagonal low-rank property, see Eqs. (7) and (8). We use hierarchically semiseparable (HSS) matrices [9–11] to approximate these matrices, and then use fast HSS matrix–matrix multiplication algorithm [12] to update the singular vectors. Note that the updating SVD problem in [3,13] can be accelerated similarly.

To fully take advantage of these two properties, a structured low-rank approximation method is proposed for Cauchy-like matrices in [7,8], which is called *SRRSC* (structured rank-revealing Schur complement factorization). SRRSC only works on several vectors, therefore its memory cost is linear $O(N)$. The complexity of using SRRSC to construct HSS approximations is analyzed in Section 4.2, which is shown to be $O(N^2r)$, where N is the dimension of the matrix and r is a modest constant as above. In Section 4.3, the rounding error analysis of SRRSC is included, which shows that SRRSC is backward stable.

Numerous experiments have been done in Section 5 to show the efficiency of our algorithm. Since both the HSS construction algorithm and HSS matrix–matrix multiplication algorithm are good for parallel, we further implement the proposed downdating SVD algorithm by using *OpenMP* on the shared memory multicore platform. For matrices with large dimensions, the proposed downdating algorithm can be 3x–5x faster than that using plain matrix–matrix multiplication routine in Intel MKL in the serial and parallel cases.

2. Preliminaries

The HSS matrix is an important kind of rank-structured matrices. It explores the low-rank property of off-diagonal blocks, which was first discussed in [10,11]. In this section, we briefly summarize some key concepts of the HSS structure following the notation in [14,15,9].

2.1. Tree structure

Suppose that H is a general $N \times N$ matrix, and $I = \{1, 2, \dots, N\}$ is the set of its row and column indices. Let $\mathscr{T}$ be a binary tree with $2n-1$ nodes labeled as $i = 1, 2, \dots, 2n-1$, in which the root node is numbered $2n-1$ and the number of leaf nodes is n. Here $\mathscr{T}$ is assumed to be in post order, which means that the ordering of non-leaf node i satisfies $i_1 < i_2 < i$, where i_1 and i_2 are its left and right child respectively. Each node i is associated with a subset of indices $t_i \subseteq I$. Thus t_i has the following properties:

1. *Each non-leaf node satisfies* $t_{i_1} \cup t_{i_2} = t_i$ *and* $t_{i_1} \cap t_{i_2} = \varnothing$, *and the leaf nodes satisfy* $\cup_{\text{all leaves } i}\ t_i = I$.
2. $t_i = I$, *when node i is the root of* $\mathscr{T}$.

Following the notation in [14], let $H_{t_it_j}$ denote the submatrix of H corresponding to row index set t_i and column index set t_j.

2.2. Generators

If matrix H of order N is represented in HSS form and its corresponding HSS tree is $\mathscr{T}$, there exist matrices D_i, U_i, V_i, R_i, W_i and B_i (called *HSS generators*) associated with each node i of $\mathscr{T}$ satisfying

$$\begin{aligned} &D_{2n-1} = H, \qquad U_{2n-1} = \varnothing, \qquad V_{2n-1} = \varnothing, \\ &D_i = H_{t_it_i} = \begin{pmatrix} D_{i_1} & U_{i_1}B_{i_1}V_{i_2}^T \\ U_{i_2}B_{i_2}V_{i_1}^T & D_{i_2} \end{pmatrix}, \\ &U_i = \begin{pmatrix} U_{i_1}R_{i_1} \\ U_{i_2}R_{i_2} \end{pmatrix}, \qquad V_i = \begin{pmatrix} V_{i_1}W_{i_1} \\ V_{i_2}W_{i_2} \end{pmatrix}. \end{aligned} \tag{4}$$

For each node i of $\mathscr{T}$, its corresponding *HSS block row and column* are defined as follows:

$$H_i^{row} = H_{t_i\times(I\setminus t_i)} \quad \text{and} \quad H_i^{column} = H_{(I\setminus t_i)\times t_i}, \tag{5}$$