ADAPTIVE JACOBI METHOD FOR PARALLEL SINGULAR VALUE DECOMPOSITIONS

Shen-Fu Hsiao

Institute of Computer and Information Engineering
National Sun Yat-Sen University
TAIWAN
sfhsiao@ie.nsysu.edu.tw

ABSTRACT

Jacobi method has been used on special-purpose multiprocessor VLSI systems for parallel singular value decomposition (SVD) of dense matrices, and CORDIC processors are often used as the basic processing elements to implement the two-sided rotations, the fundamental operations in the Jacobi method. Recently, generalizations of the original CORDIC algorithm to multi-dimensional spaces have been used in the SVD of complex matrices to achieve faster computation speed. A further speed-up of more than 2 can be gained by gradually refining the resolution of the CORDIC algorithms used in the Jacobi method.

1. INTRODUCTION

1.1. Real Plane Rotations via CORDIC

According to the CORDIC algorithm [7], a plane rotation of a 2-D real vector $[x\ y]^T$ can be decomposed into a sequence of simple elementary rotations:

$$\begin{aligned}\begin{bmatrix} x' \\ y' \end{bmatrix} &= \begin{pmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{pmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \\ &= \prod_{i=1}^{b} R_2(\alpha_i) \begin{bmatrix} x \\ y \end{bmatrix} \\ &= \prod_{i=1}^{b} \frac{1}{\sqrt{1+t_i^2}} \begin{pmatrix} 1 & \alpha_i t_i \\ -\alpha_i t_i & 1 \end{pmatrix} \begin{bmatrix} x \\ y \end{bmatrix}\end{aligned}$$

where the i-th CORDIC elementary rotation $R_2(\alpha_i)$ depends on the control sign $\alpha_i \in \{\pm 1\}$, and on the parameter t_i, which is selected to be equal to a non-positive power of 2. Hence, neglecting the scaling factor $1/\sqrt{1+t_i^2}$, the i-th elementary rotation operating on $[x_{i-1}\ y_{i-1}]^T$ is implemented by two concurrent shift-and-add operations, called a CORDIC iteration. ($[x_0\ y_0]^T = [x\ y]^T, [x_b\ y_b]^T = [x'\ y']^T$.) During the CORDIC evaluation mode, α_i is selected as $\alpha_i = sign(x_{i-1}y_{i-1})$ in order to force the second vector component to zero within approximately b-bit accuracy where b, the number of the decomposed elementary rotations, determines the resolution of the CORDIC algorithm. Recently, the CORDIC algorithm is generalized to multi-dimensional spaces [5], and this original CORDIC algorithm is also called the 2-D CORDIC algorithm.

1.2. Complex Plane Rotations via Quaternion CORDIC

A complex plane rotation on a 2-D complex vector $v_2 = [x_R + \jmath x_I\ y_R + \jmath y_I]^T$ can be written as

$$\begin{bmatrix} x'_R + \jmath x'_I \\ y'_R + \jmath y'_I \end{bmatrix} = \begin{pmatrix} c & s \\ -\bar{s} & \bar{c} \end{pmatrix} \begin{bmatrix} x_R + \jmath x_I \\ y_R + \jmath y_I \end{bmatrix}$$

where $\bar{c}, \bar{s}$ denote the complex conjugates of c, s respectively, and $c\bar{c} + s\bar{s} = 1$. A 4-D CORDIC algorithm, called quaternion CORDIC algorithm [5], was proposed to implement the above complex plane rotation by transforming the 2-D complex vector v_2 into a 4-D *real* vector $v_4 = [x_R\ x_I\ y_R\ y_I]^T$ and performing a sequence of quaternion CORDIC elementary rotations on v_4:

$$\begin{bmatrix} x'_R \\ x'_I \\ y'_R \\ y'_I \end{bmatrix} = \prod_{i=1}^{b} R_4(\begin{bmatrix} \alpha_i \\ \beta_i \\ \gamma_i \end{bmatrix}) \begin{bmatrix} x_R \\ x_I \\ y_R \\ y_I \end{bmatrix} = \prod_{i=1}^{b} \frac{1}{\sqrt{1+3t_i^2}} \times$$

$$\begin{pmatrix} 1 & \alpha_i t_i & \beta_i t_i & \gamma_i t_i \\ -\alpha_i t_i & 1 & -\gamma_i t_i & \beta_i t_i \\ -\beta_i t_i & \gamma_i t_i & 1 & -\alpha_i t_i \\ -\gamma_i t_i & -\beta_i t_i & \alpha_i t_i & 1 \end{pmatrix} \begin{bmatrix} x_R \\ x_I \\ y_R \\ y_I \end{bmatrix}$$

The i-th quaternion CORDIC iteration depends on three control signs $\alpha_i, \beta_i, \gamma_i \in \{1, -1\}$ and can be implemented by four concurrent multiple-operand shift-and-add operations. During the CORDIC evaluation mode,

0-7803-2431-5/95 $4.00

the control signs are selected according to

$$\alpha_i = sign(x_{R,i-1} \cdot x_{I,i-1}), \ \ \beta_i = sign(x_{R,i-1} \cdot y_{R,i-1}),$$
$$\gamma_i = sign(x_{R,i-1} \cdot y_{I,i-1})$$

in order to annihilate the last three vector components, i.e., to zero out the second component of the 2-D complex vector and at the same time force the first one real.

1.3. Jacobi Method

The Jacobi-like method for the SVD of an $m \times m$ matrix with complex entries iteratively applies suitable two-sided complex plane rotations to the left-handed and the right-handed sides of the 2×2 complex submatrices in the input matrix A as

$$A_0 = A, \ \ A_{j+1} = U_j A_j V_j, \ \ j = 0, 1, 2, \cdots$$

where the unitary matrices U_j and V_j are identity matrices embedded with 2×2 complex plane rotation matrices along their diagonals. Each step of the Jacobi method consists of two parts. First, U_j, V_j are selected such that every 2×2 submatrix along the diagonal of A_j is diagonalized. Then, each submatrix exchanges entries with its neighboring submatrices according to some ordering, say the parallel ordering proposed in [1]. In $(m-1)$ Jacobi steps, called a *sweep*, every off-diagonal entry is brought into a diagonal submatrix for annihilation exactly once. An array architecture for the Jacobi-SVD of 10×10 matrices is shown in Figure 1 where each processor P_{ij} performs a sequence of two-sided rotations on a 2×2 submatrix stored within it.

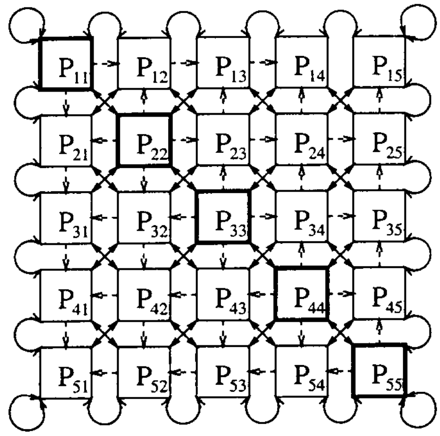

Figure 1: Systolic array architecture for the Jacobi-SVD method.

2. JACOBI-SVD WITH CORDIC

It has been shown in [4][5] that the two-sided rotations may be parallelized through an additive decomposition of a 2×2 complex matrix M into the sum of an even matrix E and an odd matrix O,

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} e_1 & e_2 \\ -\bar{e}_2 & \bar{e}_1 \end{pmatrix} + \begin{pmatrix} o_1 & o_2 \\ \bar{o}_2 & -\bar{o}_1 \end{pmatrix}$$

where

$$e_1 = \frac{a+\bar{d}}{2}, \ \ e_2 = \frac{b-\bar{c}}{2},$$
$$o_1 = \frac{a-\bar{d}}{2}, \ \ o_2 = \frac{b+\bar{c}}{2}$$

In the Jacobi method, the complex plane rotations are applied to both sides of a 2×2 complex submatrix $M = E + O$. As shown in Section 1.2, the complex rotations in both sides of E (or O) may be realized by two quaternion CORDIC operations applied from the left and from the right to the 4-D real vector corresponding to E (or O). In this case, a quaternion CORDIC elementary rotation on the 4-D vector may be moved from one side to the other and the two quaternion CORDIC elementary rotations may be merged into another new elementary rotation, called the 4-D Householder CORDIC rotation [6], with matrix representation

$$R_4'(\begin{bmatrix} \alpha_i \\ \beta_i \\ \gamma_i \end{bmatrix}) = \frac{1}{1+3t_i^2} \times$$

$$\begin{pmatrix} 1-3t_i^2 & 2\alpha_i t_i & 2\beta_i t_i & 2\gamma_i t_i \\ -2\alpha_i t_i & 1+t_i^2 & -2\alpha_i\beta_i t_i^2 & -2\alpha_i\gamma_i t_i^2 \\ -2\beta_i t_i & -2\beta_i\alpha_i t_i^2 & 1+t_i^2 & -2\beta_i\gamma_i t_i^2 \\ -2\gamma_i t_i & -2\gamma_i\alpha_i t_i^2 & -2\gamma_i\beta_i t_i^2 & 1+t_i^2 \end{pmatrix}.$$

Thus the two-sided rotations to diagonalize a 2-D complex matrix may be realized by a *single* 4-D Householder CORDIC operation, instead of several 2-D CORDIC operations as required in early work. Compared with the previously proposed Jacobi-SVD based on the original 2-D CORDIC algorithm [2][8], the above approach achieves a speed-up rate of 3 to 6 [4].

3. ADAPTIVE JACOBI-SVD

In the CORDIC-based Jacobi-SVD method, the zeros created by the two-sided rotations at each Jacobi step are smeared by the subsequent rotations. Thus, the exact annihilation of matrix entries is not necessary during the early phase of the computation. In [3], an approximation rotation scheme for parallel eigenvalue decomposition (EVD) of *real symmetric* matrices has been proposed, in which each 2-D CORDIC operation contains only one single iteration. The method calls for several magnitude comparisons at each Jacobi step and the associated scaling operation creates extra computation overhead. Besides, this Jacobi-EVD method cannot be easily generalized to the EVD of Hermitian matrices or the SVD of general complex matrices.

In order to overcome the problems mentioned above, an *adaptive* multi-dimensional CORDIC-based Jacobi-SVD method is proposed which employs CORDIC operations with smaller bit-accuracy (smaller b) to approximate the two-sided rotations during the first several sweeps, and gradually refine the CORDIC resolution. The bit accuracy of the CORDIC operations is identical for all 2×2 submatrices and remains constant throughout each sweep. To keep the overall hardware of the VLSI system as simple as possible, the criterion to determine the bit accuracy of the CORDIC operations in each sweep should be simple enough so that no other complicated arithmetic operations such as multiplications, divisions or square-roots are needed.

Let $b^{(k)}$ denote the bit resolution of the 4-D Householder CORDIC algorithm in the k-th sweep and b is the desired accuracy in the final results when the singular values are found. The adaptive Jacobi-SVD method increases the CORDIC resolution $b^{(k)}$ as follows:

- Start with the initial resolution $b^{(1)} = b_0$.
- After each Jacobi sweep, the resolution is increase by $\triangle b$ bits until it reaches the resolution limit b, i.e., $b^{(k+1)} = \min(b^{(k)} + \triangle b, b)$.

In other words, the Jacobi-SVD begins with CORDIC algorithms of smaller resolution (smaller number of iterations) and gradually increases CORDIC resolution from sweep to sweep. The selections of the initial CORDIC resolution b_0 and the resolution increase $\triangle b$ per sweep have slight effects on the convergence rate of the SVD computations. It is found out experimentally that $b_0 = \triangle b = 4$ is a good choice for complex matrices of size up to 100×100.

4. EXPERIMENTAL RESULTS

Since the convergence rate of the Jacobi-SVD may depend on the distribution of the singular values of the matrices under test, two types of complex random matrices are considered here. The first type of random matrices have the the real and imaginary parts of the matrix entries uniformly distributed between -1, 1. This type of random matrices usually have separated singular values. The second type of random matrices are generated so that the matrices tend to have multiple singular values.

Both the nonadaptive and adaptive Jacobi methods based on the 4-D Householder CORDIC algorithms are used to compute the SVD of the above two types of random matrices. The stop criterion of the Jacobi-SVD algorithms in the experiment was that the sum of the squares of all the off-diagonal entries was reduced to 10^{-12} times the sum of the squares of all the entries. Figure 2(a)(b) respectively compares the number of sweeps and CORDIC iterations required for the SVD of complex matrices using the nonadaptive Jacobi-SVD approach (denoted by solid lines) in [4] and the new adaptive Jacobi-SVD approach (denoted by dashed lines). The unboldfaced lines denote the experiments for the first type of random matrices while the boldfaced lines denote the results for the second type of random matrices.

Although the adaptive Jacobi method calls for slightly more sweeps, the total number of the CORDIC iterations required is smaller than that using nonadaptive Jacobi method. The speed performance of the adaptive Jacobi-SVD method is at least twice faster than our previous approach in [4] which has already been three to four times faster than the approaches in [2][8]. It is an interesting observation that the second type of random matrices (matrices tending to have multiple singular values) call for more sweeps than the first type of random matrices, and thus require more CORDIC iterations for convergence.

In a systolic-like implementation such as the one shown in Figure 1, classical matrix dependent termination criteria for the Jacobi-SVD algorithm are very expensive. Thus a good strategy is to stop after a predetermined number of sweeps, possibly dependent on the dimension of the matrix. In order for the SVD algorithm to converge for most matrices, the worse case situation should be considered. From our experiment, the number of sweeps is selected to be 13 for any random matrix of size up to 100. In [1], Brent *et al.* suggested to select 10 as the number of sweeps for matrices of practical size (say, up to 1000) because they considered only random matrices of the first type. But as been shown in Figure 2, the second type of random matrices require more sweeps for convergence. Larger matrices require larger numbers of sweeps

5. CONCLUSIONS

The recently proposed multi-dimensional CORDIC algorithms have been applied to finding the singular val-

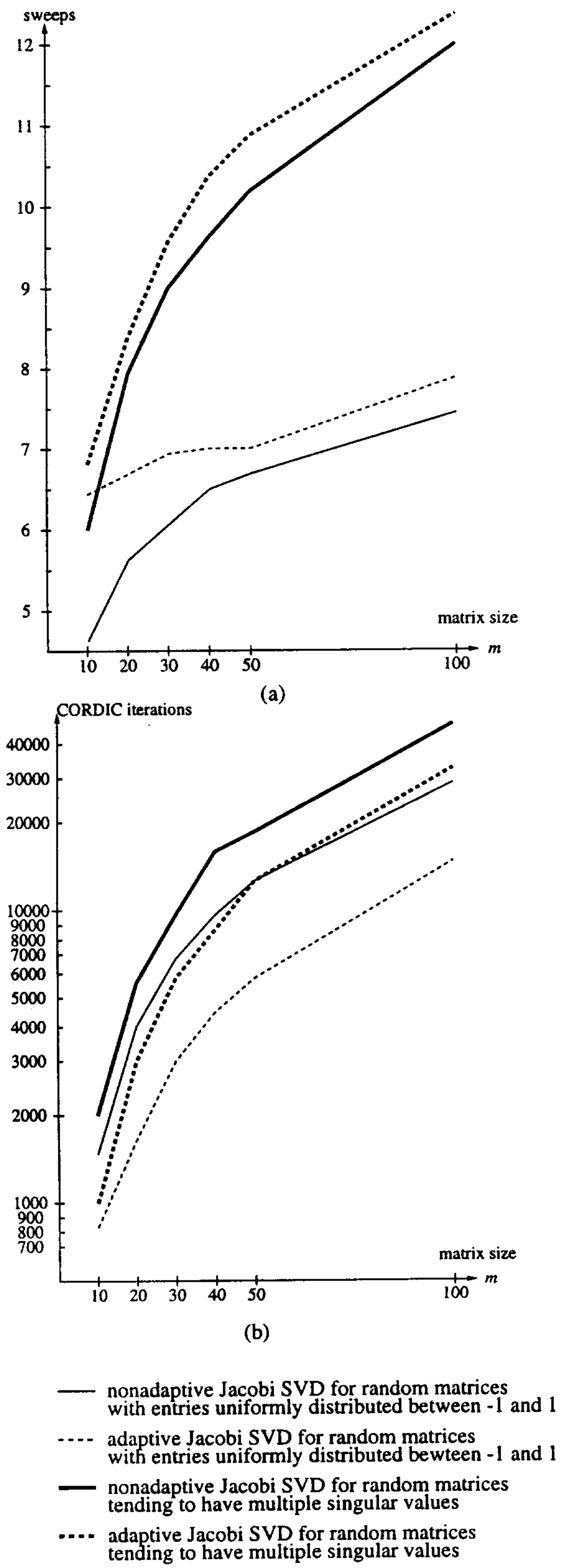

Figure 2: (a) Number of sweeps and (b) number of CORDIC iterations required for Jacobi-SVD.

ues of complex matrices based on a Jacobi-like method in which some sequentiality of two-sided complex rotations is removed by decomposing additively a 2×2 complex matrix into the sum of an even and an odd matrices. The parallelized two-sided rotations, realized by the 4-D Householder CORDIC algorithm, is at least 3 times faster compared to earlier work based on the original 2-D CORDIC algorithms. In this paper, the Jacobi-SVD computation speed is further increased by a factor of 2 by adaptively refining the accuracy of the CORDIC algorithms employed in the Jacobi-SVD. Some experimental results are included to show the superiority of the new adaptive Jacobi-SVD method.

6. REFERENCES

[1] R.P. Brent and F.T. Luk, *The Solution of Singular-Value and Symmetric Eigenvalue Problems on Multiprocessor Arrays*, SIAM J. Sci. Stat. Comput., Vol. 6, No. 1, pp. 69-84, Jan. 1985.

[2] J.R. Cavallaro and F.T. Luk, *CORDIC Arithmetic for an SVD Processor*, Journal of Parallel and Distributed Computing, Vol. 5, pp. 271-290, 1988.

[3] J. Götze, S. Paul, and M. Sauer, *An Efficient Jacobi-like Algorithm for Parallel Eigenvalue Computation*, IEEE Trans. Computer, pp. 1058-1065, Sept. 1993.

[4] S.-F. Hsiao and J.-M. Delosme, *Parallel Complex Singular Value Decomposition Using Multidimensional CORDIC Algorithms*, Int. Conf. Parallel and Distributed Systems, pp. 487-494, Dec. 1993.

[5] S.-F. Hsiao, *Multi-dimensional CORDIC Algorithms*, Ph.D. Dissertation, Dept. of Electrical Engineering, Yale Univ., Dec. 1993.

[6] S.-F. Hsiao and J.-M. Delosme, *The CORDIC Householder Algorithm*, 10-th IEEE Intl. Symposium on Computer Arithmetic, pp. 256-263, June 1991.

[7] J.E. Volder, *The CORDIC Trigonometric Computing Technique*, IRE Trans. on Electronic Computers, Vol. EC-8, No. 3, pp. 330-334, Sept. 1959.

[8] B. Yang and J.F. Böhme, *Reducing the Computations of the Singular Value Decomposition Array Given By Brent and Luk*, SIAM J. Matrix Anal. Appl., Vol. 12, No. 4, pp. 713-725, Oct. 1991.