IMPLEMENTATION OF KOGBETLIANTZ'S SVD ALGORITHM USING ORTHONORMAL μ–ROTATIONS

Jürgen Götze [†1] , *Peter Rieder* [‡] and *Josef A. Nossek* [‡]

[†]Electrical & Computer Engineering
Rice University
Houston, TX 77251–1892, U.S.A.

[‡]Network Theory & Circuit Design
Technical University of Munich
Arcisstr. 21, 80333 Munich, Germany

ABSTRACT

In this paper the implementation of Kogbetliantz's SVD algorithm using orthonormal μ–rotations is presented. An orthonormal μ–rotation is a rotation by an angle of a given set of μ–rotation angles (e.g. the angles $\Phi_i = \arctan 2^{-i}$) which are choosen such that the rotation can be implemented by a small amount of shift–add operations. All computations (evaluation and application of the rotations) can entirely be referred to orthonormal μ–rotations. Simulations show the reduced computational complexity of Kogbetliantz's SVD algorithm based on orthonormal μ–rotations comparded to the standard Kogbetliantz SVD algorithm.

1. INTRODUCTION

The singular value decomposition (SVD) of a $m \times n$ matrix A is given by

$$A = U\mathbf{\Sigma}V^T ,$$

where U, V are orthonormal matrices ($UU^T = I$, $VV^T = I$) and $\mathbf{\Sigma} = \text{diag}(\sigma_1, \ldots, \sigma_n)$ is a diagonal matrix containing the singular values ($\sigma_i = i$–th singular value of A).

The method of choise for the fast parallel computation of the SVD is Kogbetliantz's algorithm [1], since it offers a higher degree of parallelism than the respective QR–methods. These parallel implementations of the Kogbetliantz algorithm usually require the evaluation and application of plane rotations in the processor cells. Therefore, the complexity of the parallel (or sequential) implementation is mainly determined by the complexity of the rotation computations. Consequently, different strategies for simplifying the rotation computations have been presented (approximate rotations [3], factorized rotations [10], CORDIC [5]). Also, combinations of these modifications were recently proposed. In [6] the factorization was applied to approximate rotations yielding square root and division free algorithms. The idea of using approximate rotations was also combined with the CORDIC idea (computation in the rotation angle domain with basis "$\arctan 2^{-k}$") [8, 7].

For the Jacobi method for computing the symmetric EVD it was shown in [8] that using only one simple μ–rotation (one recursion) of the CORDIC sequence as an approximate rotation for each occuring similarity transformation reduces the computational complexity significantly. In [7] it was shown that the evaluation of the required μ–rotation (i.e. determing which CORDIC angle to use) can also be referred to the execution of μ–rotations. Now, having an algorithm completely based on the execution of μ–rotations an architecture (based on the floating point CORDIC architecture of [11]) was also proposed in [7].

In this paper the results for the Jacobi EVD algorithm are extended to the Kogbetliantz SVD algorithm. Usually, this extension to the SVD is straightforward [6]. However, the application of CORDIC–based approximate rotations to the elementary 2×2 SVD subproblem

$$\bar{A} = G(\Phi^U)^T A G(\Phi^V)$$

where

$$G(\Phi) = \begin{bmatrix} \cos\Phi & \sin\Phi \\ \Leftrightarrow\sin\Phi & \cos\Phi \end{bmatrix} ,$$

is not as straightforward. In the EVD case ($\Phi^U = \Phi^V$) determing the optimal CORDIC–based approximate rotation is a two–dimensional optimization problem which can be solved using μ–rotations itself [8, 7]. Simply extending these results to the SVD we would obtain an four–dimensional optimization problem.

It was shown in [2, 12], however, that the SVD subproblem can be decomposed into two independent rotation computations. Using these results it is possible to derive a Kogbetliantz SVD algorithm entirely based on orthonormal μ–rotations.

2. SEPARATION OF ANGLE COMPUTATION

Given

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$$

[1] The work of this author was supported by the Alexander von Humboldt Foundation and Texas Advanced Technology Program

one computes

$$\begin{array}{ll} x_1 = (a_{22} + a_{11})/2 & x_2 = (a_{22} \Leftrightarrow a_{11})/2 \\ y_1 = (a_{21} \Leftrightarrow a_{12})/2 & y_2 = (a_{21} + a_{12})/2 \end{array} \qquad (1)$$

Now, we can determine two angles Φ^R and Φ^S, independently, such that

$$\begin{bmatrix} x_1' \\ y_1' \end{bmatrix} = G(\Phi^R) \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} \qquad (2)$$

and

$$\begin{bmatrix} x_2' \\ y_2' \end{bmatrix} = G(\Phi^S) \begin{bmatrix} x_2 \\ y_2 \end{bmatrix}, \qquad (3)$$

where in the case of using exact rotations (i.e. using $\Phi^R = \arctan y_1/x_1$, $\Phi^S = \arctan y_2/x_2$) we obtain $y_1' = y_2' = 0$. Then,

$$\bar{A} = G(\Phi^S/2)^T G(\Leftrightarrow\Phi^R/2)^T A G(\Phi^R/2) G(\Phi^S/2) \qquad (4)$$

solves the SVD subproblem ($\bar{a}_{12} = \bar{a}_{21} = 0$) [2, 12]. Note also the interesting interpretation of this procedure in terms of Clifford algebra in [4].

3. ORTHONORMAL μ–ROTATIONS

All rotation computations in the above procedure are replaced by approximate rotations in the following, where orthonormal μ–rotations are used as the set of available approximate rotations.

3.1. Solving the QRD subproblem

While the execution of an exact rotation as described in (2) and (3) guaranees $y' = 0$ (we discard the indices for the time being), an approximate rotation $G(\tilde{\Phi})$ defined by an approximate rotation angle $\tilde{\Phi} \approx \Phi$ such that

$$\mid y' \mid = \mid d \mid \cdot \mid y \mid \qquad (5)$$

with $0 \leq \mid d \mid < 1$.

Using an approximate angle $\tilde{\Phi}$ in (2) or (3) yields:

$$y' = (\Leftrightarrow \sin\tilde{\Phi} \cdot \frac{x}{y} + \cos\tilde{\Phi}) \cdot y \,. \qquad (6)$$

Representing this equation (6) as $y' = d(\Phi, \tilde{\Phi}) \cdot y$ and using $\tan\Phi = y/x$ one obtains

$$d(\Phi, \tilde{\Phi}) = \Leftrightarrow \sin\tilde{\Phi} \cdot \frac{1}{\tan\Phi} + \cos\tilde{\Phi} \,. \qquad (7)$$

At this point having defined an approximate rotation we make use of the idea of CORDIC, i.e. with respect to a simple implementation of the rotation we restrict ourselves to the set of approximate angles

$$\tilde{\Phi} = \Phi_i = \arctan 2^{-i}, \qquad (8)$$

where $i \in \mathcal{I} = \{0, 1, 2, \ldots, w\}$. Therefore, we only allow rotations of the form

$$G(\Phi_i) = \frac{1}{K_i} G_u(\Phi_i) = \frac{1}{K_i} \begin{bmatrix} 1 & \tau_i 2^{-i} \\ \Leftrightarrow\tau_i 2^{-i} & 1 \end{bmatrix} \qquad (9)$$

where

$$\frac{1}{K_i} = \frac{1}{\sqrt{1 + 2^{-2i}}} \qquad (10)$$

is the scaling factor and $G_u(\Phi_i)$ is an (unscaled) μ–rotation. $G(\Phi_i)$ is called an orthonormal μ–rotation.

Since an orthonormal μ–rotation $G(\Phi_i)$ is defined by the angle index i computing the optimal orthonormal μ–rotation $G(\Phi_{i_o})$ corresponds to

$$\mid \Phi_{i_o} \Leftrightarrow \Phi \mid = \min_{i \in \mathcal{I}} \mid \Phi_i \Leftrightarrow \Phi \mid \quad \Rightarrow \quad \tilde{\Phi} = \Phi_{i_o} \,. \qquad (11)$$

The direction of the rotation τ_i is determined by $\tau_i = \text{sign}(x)\text{sign}(y)$. Therefore, having determined τ_i we can work with $x > 0$ ($x \leftarrow |x|$) and $y > 0$ ($y \leftarrow |y|$) to evaluate Φ_{i_o}.

Choosing the optimal approximate angle according to (11) is equivalent to

$$\min_{i \in \mathcal{I}} |d(\Phi, \Phi_i)| \,. \qquad (12)$$

This minimization can be done by determing the angle intervals $[\Phi_{gi}, \Phi_{gi+1}]$ such that $\tilde{\Phi} = \Phi_i$ is the optimal angle whenever $\Phi \in [\Phi_{gi}, \Phi_{gi+1}[$. The limits of the intervals Φ_{gi} follow from the solution of

$$d(\Phi_{gi}, \Phi_i) = \Leftrightarrow d(\Phi_{gi}, \Phi_{i+1}) \,, \qquad (13)$$

i.e. the angle Φ_{gi} where choosing Φ_i leads to the same reduction factor $|d|$ as choosing Φ_{i+1}. Solving (13) yields:

$$\tan\Phi_{gi} = \frac{\sin\Phi_i + \sin\Phi_{i+1}}{\cos\Phi_i + \cos\Phi_{i+1}} = \tan\left(\frac{\Phi_i + \Phi_{i+1}}{2}\right) \,. \qquad (14)$$

As the orthonormal μ–rotation $G(\Phi_i)$ in (9) is given by one specific recursion step of the original CORDIC sequence we define an orthonormal double μ–rotation $G_d(\bar{\Phi}_i)$ by the product of two orthonormal μ–rotations (9), i.e.

$$\begin{aligned} G_d(\bar{\Phi}_i) &= G(\Phi_i)G(\Phi_i) = \frac{1}{K_i^2} G_{du}(\bar{\Phi}_i) = \\ &= \frac{1}{K_i^2} \begin{bmatrix} 1 \Leftrightarrow 2^{-2i} & \sigma 2^{-i+1} \\ \Leftrightarrow\sigma 2^{-i+1} & 1 \Leftrightarrow 2^{-2i} \end{bmatrix} \end{aligned} \qquad (15)$$

where

$$\frac{1}{K_i^2} = \frac{1}{1 + 2^{-2i}} \qquad (16)$$

is the scaling factor and $G_{du}(i)$ is the (unscaled) double μ–rotation. Therefore, instead of working with the set of approximate angels Φ_i, the set of availabel approximate angles is now given by

$$\bar{\Phi}_i = \arctan \frac{2^{-i+1}}{1 \Leftrightarrow 2^{-2i}} = 2 \cdot \Phi_{i+1} \,.$$

The scaling factor of the orthonormal double μ–rotation can be recursively computed by shift–and–add operations [8]:

$$1/K_i^2 = (1 \Leftrightarrow 2^{-2i}) \prod_{s=1}^{b} (1 + 2^{-2^{s+1}i}) \qquad (17)$$

with $b = \log_2 \lceil \frac{w}{2i} \rceil$. Since the orthonormal double μ–rotations enable an easy scaling factor compensation (17) and since the limits of the intervals can be easily determined for the set of orthonormal double μ–rotations we restrict our set of approximate rotations to the orthonormal double μ–rotations from now on. The limits of the intervals $\left[\bar{\Phi}_{gi}, \bar{\Phi}_{gi+1}\right[$ for the choice of the optimal double μ–rotation angle are now given by

$$\tan \bar{\Phi}_{gi} = \tan\left(\frac{\bar{\Phi}_i + \bar{\Phi}_{i+1}}{2}\right) . \tag{18}$$

Therefore, given $v = [x, y]^T$ (i.e. $\Phi = \arctan y/x$) we use

$$\begin{gathered} \bar{\Phi}_i \text{if} \Phi > \bar{\Phi}_{gi} \\ \bar{\Phi}_{i+1} \text{if} \Phi \leq \bar{\Phi}_{gi} \, . \end{gathered}$$

Since $\bar{\Phi}_{gi} = \bar{\Phi}_i/2 + \bar{\Phi}_{i+1}/2 = \Phi_{i+1} + \Phi_{i+2}$ this decision can be made by using two unscaled μ–rotations $G_u(\Phi_{i+1})$ and $G_u(\Phi_{i+2})$, i.e. compute

$$[x_r, y_r]^T = G_u(\Phi_{i+1}) G_u(\Phi_{i+2}) [x, y]^T \, . \tag{19}$$

Thereby, we obtain

$$\begin{aligned} y_r > 0 \Leftrightarrow \Phi > \bar{\Phi}_{gi} \quad &\Rightarrow \quad \text{use } \bar{\Phi}_i \\ y_r \leq 0 \Leftrightarrow \Phi \leq \bar{\Phi}_{gi} \quad &\Rightarrow \quad \text{use } \bar{\Phi}_{i+1} \, . \end{aligned}$$

In order to find the optimal i_o one more unscaled μ–rotation is required. Let $\text{man}(a)$ and $\exp(a)$, respectively, denote the mantissa and the exponent of a binary floating point number a. We can obtain an estimate for the optimal i_o by computing

$$i_e = \exp(y) \Leftrightarrow \exp(x) \, . \tag{20}$$

Since $\text{man}(y)/\text{man}(x) \in [0.25, 1[$ one obtains

$$i_o \in \mathcal{J} = \{i_e, i_e + 1, i_e + 2\} \, . \tag{21}$$

Therefore, it is possible to determine the optimal $\bar{\Phi}_{i_o}$ ($i_o \in \mathcal{J}$) as follows. Compute:

$$\begin{aligned} v_0 &= G_u(\Phi_{i+2}) v \\ v_1 &= G_u(\Phi_{i+1}) v_0 \\ v_2 &= G_u(\Phi_{i+3}) v_0 \, . \end{aligned}$$

Then

$$\bar{\Phi}_{i_o} = \begin{cases} \bar{\Phi}_{i_e} & \text{if } v_1(2) > 0 \\ \bar{\Phi}_{i_e+1} & \text{if } v_2(2) > 0 \\ \bar{\Phi}_{i_e+2} & \text{otherwise} \end{cases}$$

This procedure yields the optimal orthonormal double μ–rotation $G_d(\bar{\Phi}_{i_o})$ such that $y' = d(\Phi, \bar{\Phi}_{i_o}) \cdot y$ with $|d(\Phi, \bar{\Phi}_{i_o})| \leq 1/3$.

3.2. Solving the SVD subproblem

The optimal orthonormal double μ–rotations for the two QRD subproblems (2) and (3) can be determined by applying the procedure described in section 3.1 twice. Thereby one obtains $G_d(\bar{\Phi}^R_{i_o})$ and $G_d(\bar{\Phi}^S_{i_o})$. Given these rotations the SVD rotations are obtained according to (4). Since $\bar{\Phi}_i \approx 2\bar{\Phi}_{i+1}$ we obtain the approximate rotations for the SVD subproblem as follows:

$$\begin{aligned} G(\tilde{\Phi}^U) &= G_d(\Leftrightarrow \bar{\Phi}^R_{i_o+1}) G_d(\bar{\Phi}^S_{i_o+1}) \qquad (22) \\ G(\tilde{\Phi}^V) &= G_d(\bar{\Phi}^R_{i_o+1}) G_d(\bar{\Phi}^S_{i_o+1}) \qquad (23) \end{aligned}$$

By using $i_o + 1$ instead of i_o for the SVD rotations we actually use $\tilde{\Phi}^R = 2\bar{\Phi}^R_{i_o+1}$ and $\tilde{\Phi}^S = 2\bar{\Phi}^S_{i_o+1}$ as the approximate angles for the QRD subproblem (instead of $\tilde{\Phi}^R = \bar{\Phi}^R_{i_o}$ and $\tilde{\Phi}^S = \bar{\Phi}^S_{i_o}$). Therefore, the appximation of the QRD subproblem changes correspondingly and we obtain $y' = d(\Phi, \tilde{\Phi}) \cdot y$ with $|d(\Phi, \tilde{\Phi})| < d_{max} = 0.42$.

3.3. Convergence

It remains to show that $a'^2_{12} + a'^2_{21} \leq d^2_{SVD}(a^2_{12} + a^2_{21})$ with $0 \leq d^2_{SVD} < 1$ is guaranteed for each subproblem, in order to meet the requirements for the convergence of the Kogbetliantz algorithm [6]. For the two QRD subproblems (2) and (3) one obtains

$$\begin{aligned} y'_1 = d_1 y_1 \qquad & \text{where } 0 \leq |d_1| < d_{max} \qquad (24) \\ y'_2 = d_2 y_2 \qquad & \text{where } 0 \leq |d_2| < d_{max} \qquad (25) \end{aligned}$$

such that [12]

$$a'_{12} = \Leftrightarrow y'_1 + y'_2 ; \qquad a'_{21} = y'_1 + y'_2 . \tag{26}$$

Therefore, with (1) one obtains

$$\begin{aligned} \bar{a}^2_{12} + \bar{a}^2_{21} &= 2(y'^2_1 + y'^2_2) \\ &< 2d^2_{max}(y^2_1 + y^2_2) \\ &= d^2_{max}(a^2_{12} + a^2_{21}) \end{aligned}$$

such that

$$d^2_{SVD} \leq d^2_{max} = 0.17 \, . \tag{27}$$

Furthermore, instead of $\bar{\Phi}_1 = 53.1301°$ we use $90° \Leftrightarrow \bar{\Phi}_1 = 36.8699°$ that guarantees $|\tilde{\Phi}^U|, |\tilde{\Phi}^V| < 90°$ and also leads to an improved approximation of the exact angles. Note that this method improves the reduction factors (and therefore the performance of the entire algorithms) given in [8].

4. SIMULATIONS

In Figure 1 we compare the performance of the presented Kogbetliantz SVD algorithm using orthonormal μ–rotations (solid lines) to the standard Kogbetliantz SVD algorithm using exact rotations (dotted lines). A

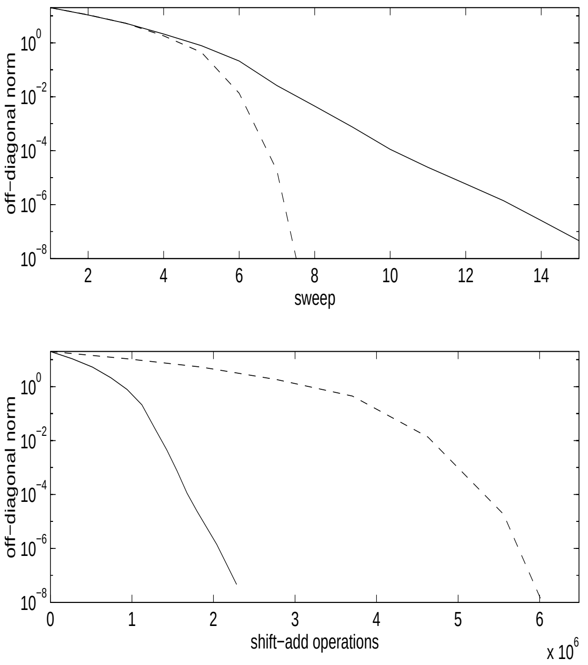

Figure 1: Off-diagonal norm vs. (a) sweeps and (b) number of shift&add operations.

random $n \times n$ matrix A ($n = 20$) is diagonalized using both methods. The off–diagonal norm S[1]

$$S = \left[||A||_F^2 \Leftrightarrow \sum_{i=1}^{n} a_{ii}^2 \right]^{1/2}$$

is shown vs. the sweeps (Fig. 1(a)) and vs. the required number of shift&add operations (Fig. 1(b)). Each sweep consists of $n(n \Leftrightarrow 1)/2$ solutions of 2×2 SVD subproblems and the application of the computed rotations (exact, approximate) to the entire matrix in the respective plane. The cyclic–by–row ordering scheme [1] is used for choosing the respective 2×2 SVD subproblems. The quadratic convergence of the standard Kogbetliantz algorithm vs. the sweeps is lost for the μ–rotation based algorithm (Fig. 1(a)). However, the reduction of the off–diagonal norm vs. the shift&add operations (actual computational complexity) is significantly better for Kogbetliantz's SVD algorithm using orthonormal μ–rotations. (Fig. 1(b)).

5. CONCLUSIONS

The presented Kogbetliantz SVD algorithm using orthonormal μ–rotations reduces the overall amount of shift&add operations compared to the use of the standard CORDIC significantly. The use of orthonormal μ–rotations is also highly suited for subspace tracking using the SVD–updating algorithm [9]

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.*, 6:69–84, 1985.

[2] J.R. Cavallaro and F.T. Luk. CORDIC Arithmetic for an SVD Processor. *J. Parallel & Distributed Computing*, 5:271–290, 1988.

[3] J.P. Charlier, M. Vanbegin, and P. van Dooren. On Efficient Implementations of Kogbetliantz's Algorithm for Computing the Singular Value Decomposition. *Numer. Math.*, 52:279–300, 1988.

[4] J.-M. Delosme. Bit-Level Systolic Algorithm for the Symmetric Eigenvalue Problem. In *Proc. Int. Conf. on Application Specific Array Processors*, pages 771–781, Princeton (USA), 1990.

[5] J.M. Delosme. Parallel Implementation of the SVD Using Implicit CORDIC Arithmetic. In R.J. Vaccaro, editor, *SVD and Signal Processing II*, pages 33–56. Elsevier Science Publishers, 1991.

[6] J. Götze. On the Parallel Implementation of Jacobi and Kogbetliantz Algorithms. *SIAM J. on Sci. Comput.*, 15:1331–1348, 1994.

[7] J. Götze and G.J. Hekstra. An Algorithm and Architecture based on Orthonormal μ–Rotations for Computing the Symmetric EVD. *INTEGRATION – The VLSI Journal. Special issue on algorithms and parallel VLSI architectures*, pages 21–40, 1995.

[8] J. Götze, S. Paul, and M. Sauer. An Efficient Jacobi–Like Algorithm for Parallel Eigenvalue Computation. *IEEE Trans. on Computers*, 42:1058–1065, 1993.

[9] J. Götze, P. Rieder, and G.J. Hekstra. SVD–Updating Using Orthonormal μ–Rotations. *J. of VLSI Signal Processing (Special Issue on Array Optimization and Adaptive Tracking Algorithms)*, 1996.

[10] J. Götze and U. Schwiegelshohn. A Square Root and Division Free Givens Rotation for Solving Least Squares Problems on Systolic Arrays. *SIAM J. Sci Stat. Comput.*, 12:800–807, 1991.

[11] G.J. Hekstra and E.F. Deprettere. Floating Point CORDIC. In *11th Symp. on Computer Arithmetic*, Winsor (Canada), 1993.

[12] 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.*, 12:713–725, 1991.

[1] $S \rightarrow 0$ implies that A converges to the diagonal matrix Σ.