NASA-CR-172,396

NASA Contractor Report 172396

ICASE REPORT NO. 84-30

NASA-CR-172396
19840022748

ICASE

SINGULAR VALUE DECOMPOSITION WITH SYSTOLIC ARRAYS

Ilse C. F. Ipsen

Contract No. NAS1-17070
July 1984

INSTITUTE FOR COMPUTER APPLICATIONS IN SCIENCE AND ENGINEERING
NASA Langley Research Center, Hampton, Virginia 23665

Operated by the Universities Space Research Association

LIBRARY COPY

AUG 2 1 1984

LANGLEY RESEARCH CENTER
LIBRARY, NASA
HAMPTON, VIRGINIA

NASA
National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23665

3 1176 00518 1723

SINGULAR VALUE DECOMPOSITION WITH SYSTOLIC ARRAYS

Ilse C. F. Ipsen
Yale University

Abstract

Systolic arrays for determining the Singular Value Decomposition of a $m \times n$, $m > n$, matrix A of bandwidth w are presented. After A has been reduced to bidiagonal form B by means of Givens plane rotations, the singular values of B are computed by the Golub-Reinsch iteration. The products of plane rotations form the matrices of left and right singular vectors.

Assuming each processor can compute or apply a plane rotation, $O(wn)$ processors accomplish the reduction to bidiagonal form in $O(np)$ steps, where p is the number of superdiagonals. A constant number of processors can then determine each singular value in about $6n$ steps. The singular vectors are computed by rerouting the rotations through the arrays used for the reduction to bidiagonal form, or else "along the way" by employing another rectangular array of $O(wm)$ processors.

Research was supported in part by the Office of Naval Research Contract N000014-82-K-0184 and in part by the National Aeronautics and Space Administration under NASA Contract No. NAS1-17070 while the author was in residence at the Institute for Computer Applications in Science and Engineering, NASA Langley Research Center, Hampton, VA 23665.

N84-30817#

Introduction

A singular value decomposition (SVD) of a $m \times n$ matrix A, $m > n$, consists of a factorization

$$A = U \Sigma V^T,$$

where U and V are orthogonal matrices and Σ is a nonnegative diagonal matrix.

The SVD is an indispensible tool for rank determination and handling of rank deficiencies, a more detailed account of its mathematical properties can be found in [6]. As for its applications in signal processing, Speiser and Whitehouse have presented its usefulness in adaptive beamforming and data compression [13].

With the advent of VLSI technology it seems now feasible to perform a SVD in real-time. A number of papers have recently dealt with algorithms for SVD of dense matrices amenable to implementation on systolic arrays. An $O(n^2)$-processor singular value array of [4] relies on a version of Hestenes' method where intermediate quantities are not completely updated; a formal convergence proof is not provided. Brent and Luk [1] construct an array for a one-sided orthogonalisation method due to Hestenes which uses a linearly connected mesh of $O(n)$ processors and $O(mn)$ steps to determine a singular value or else a two-dimensional $O(mn)$ processor array with a non-planar interconnection structure and time $O(n \log m)$. The method is quadratically convergent; experience suggests that 6 to 10 iterations per singular value provide for sufficient numerical accuracy. In [2] a similar architecture of $O(n^2)$ processors implements a cyclic Jacobi method for the SVD in $O(m + n \log n)$ steps. With the addition of QR and matrix multiplication arrays, the generalised SVD for matrices $A \varepsilon R^{m \times n}$ and $B \varepsilon R^{p \times n}$ is computed in $O(\ell \log n)$ steps on $O(\ell^2)$ processors, $\ell > m,n,p$ [3]. With reference to the previous works Schreiber [12] suggests a method to cope with problems that do not match the array size. In [11] he proposes a kn-processor design which reduces a dense matrix to upper triangular form of bandwidth $k+1$ in time $O(mn/k)$. A $k(k+1)$-processor array from [8] is used to implement a SVD iteration for $(k+1)$-diagonal matrices (analogous to the Golub-Reinsch iteration for bidiagonal matrices) in time $6n + O(k)$. For $k = O(n^{1/2})$ this amounts to processor and hardware requirements of $O(n^{3/2})$.

The systolic designs to be discussed are based on those in [8] and are thoroughly specified in [10]. They compute the singular values of a banded matrix A by first reducing A to bidiagonal form B and then computing the singular values of B. A VLSI implementation for the same problem was already proposed in [7]; this paper, however, substantially improves the bandwidth reduction array: it has become much simpler and is now also able to deal with problems not matching the hardware dimensions. The matrices of left and right singular vectors, U^T and V, are generated by accumulating the products of plane rotations.

After a description of the SVD algorithm in the next section, a review of the systolic designs in [8] will be given. It is followed by a discussion of systolic arrays for bandwidth reduction and singular value computation for bidiagonal matrices.

2. Singular Value Decomposition

The singular value decomposition (SVD) of a matrix $A \in R^{m\times n}$, $m > n$, is

$$U^T AV = \mathrm{diag}(\sigma_1,\ldots,\sigma_n),$$

where $U \in R^{m\times m}$ and $V \in R^{n\times n}$ are orthogonal matrices and $\mathrm{diag}(\sigma_1,\ldots,\sigma_n)$ is a nonnegative diagonal matrix. The columns of $U(V)$ constitute the left (right) singular vectors of A, and σ_i are the singular values.

The eigenvalues of A^TA are the squares of the singular values of A:

$$V^T(A^TA)V = \mathrm{diag}(\sigma_1^2,\ldots,\sigma_n^2).$$

Hence, computation of the singular values of A can be performed by computing the eigenvalues of A^TA.

The QR algorithm for computing the eigenvalues of a symmetric matrix $A \in R^{n\times n}$ is based on the decomposition of A into an orthogonal $(Q^{-1} = Q^T)$ matrix Q and an upper triangular matrix R,

$$A = QR.$$

It is most efficient when the original matrix A is first reduced to tridiagonal form T ($t_{ij} = 0$ for $i < j-1$ or $i > j+1$). With $T_0 = T$ an iteration of the QR method takes the form

$$T_i - s_{i+1}I = Q_{i+1}R_{i+1}, \quad T_{i+1} = R_{i+1}Q_{i+1} + s_{i+1}I$$

or,

$$T_{i+1} = Q_{i+1}^T T_i Q_{i+1}.$$

If the scalar s_{i+1} is chosen as the eigenvalue of the trailing 2×2 submatrix closest to $t_{nn}^{(i)}$ then this element will converge to an eigenvalue of A at least quadratically. Once $t_{n,n-1}^{(i)}$ is "close" to zero $t_{nn}^{(i)}$ is a good eigenvalue approximation. In that case, A_i is deflated (its trailing row and column are disregarded) and the procedure is repeated to find the next eigenvalue.

Having formulated the SVD as an eigenvalue problem a straightforward approach for its computation would be to first reduce the matrix A to bidiagonal form B ($b_{ij} = 0$ for $i > j$ or $i < j-1$), and then to compute the eigenvalues of $T = B^TB$ via

$$V^T TV = \text{diag}(\sigma_1^2,\ldots,\sigma_n^2).$$

Then V is the matrix of right singular vectors and the left singular vectors are obtained from the QR decomposition of AV. Yet, the explicit formation of B^TB squares the condition number of the problem and numerically results in loss of information.

Golub and Reinsch present an alternative method which avoids the explicit formation of the matrix product [5]. If T has nonzero offdiagonal elements, Q_1 is an orthogonal matrix and

$$T_1 = Q_1^T TQ_1,$$

it can be shown, e.g. [6], that Q_1 and T_1 are uniquely determined by the first column of Q_1. Furthermore, if P is an orthogonal matrix with the same first column as Q_1 and $P^T TP$ is reduced to tridiagonal form

$$T_2 = (PQ_2)^T T(PQ_2),$$

then $|T_1| = |T_2|$. Consequently, the singular values of B can be determined by iterations of the following form $(B_0 = B)$,

Let P_{i+1} be orthogonal with the same first column as Q_{i+1}

Compute $C_{i+1} = B_i P_{i+1}$

Reduce $B_i P_{i+1}$ to bidiagonal form B_{i+1}.

The effect of the shift s_{i+1} is now concentrated in the matrix P_{i+1} and the above procedure is known as the "implicitly shifted" version of the QR method. Deflation takes place as for eigenvalue computations.

Hence, the singular values of A are computed in two steps:

(1) Reduce A to bidiagonal form B

(2) Perform the above iterations on B until the required singular values are found.

3. Systolic Arrays for Givens Rotations

The purpose of an orthogonal matrix in this context is to stably reduce a matrix to a certain structured form by selectively introducing zero elements. It was shown in [8] that a VLSI implementation of Givens plane rotations would, unlike Householder transformations for example, require only nearest neighbour data communication. Thus, all orthogonal matrices will be products of plane rotations, where each rotation zeroes a single matrix element as follows:

$$P = \begin{pmatrix} c & s \\ -s & c \end{pmatrix}, \qquad c^2 + s^2 = 1,$$

$$P\begin{pmatrix} x_1 & x_2 & \cdots & x_k \\ y_1 & y_2 & \cdots & y_k \end{pmatrix} = \begin{pmatrix} x_1' & x_2' & \cdots & x_k' \\ 0 & y_2' & \cdots & y_k \end{pmatrix}, \qquad k > 1,$$

where

$$\text{if } \; y_1 = 0 \; \text{ then } \; x_1' = x_1, \; c = 1, \; s = 0,$$

$$\text{else } \; x_1' = \sqrt{x_1^2 + y_1^2}, \quad c = x_1/x_1', \quad s = y_1/x_1', \qquad \text{(P1)}$$

$$x_i' = cx_i + sy_i, \quad y_i' = -sx_i + cy_i, \qquad 2 \leq i \leq k, \qquad \text{(P2)}$$

It is assumed that each, equation (P1) and (P2), takes one "time step". Two kinds of processors are needed to realise rotations, one to execute (P1) and another for (P2), see Figure 1.

Given a matrix $A \varepsilon R^{m \times n}$, $m > n$, its diagonal consists of the elements a_{ii}, $1 < i < n$, the kth subdiagonal of $a_{k+i,i}$, $1 < i < m-k$, and the kth superdiagonal of $a_{i,k+i}$, $1 < i < n-k$. In the sequel, only band matrices of bandwidth $w = p + q + 1$ will be considered where subdiagonal $q > 0$ is the leftmost nonzero subdiagonal and superdiagonal $p > 1$ is the rightmost nonzero superdiagonal. Consequently, either $m = n$ or $m = n + q$.

A QR decomposition of a matrix A does not increase the bandwidth; for each eliminated subdiagonal a new nonzero superdiagonal is filled in, so R has bandwidth w with p+q superdiagonals.

For an overview on systolic arrays for the QR factorization the reader is referred to [8]. There a linearly connected mesh of w processors, call it L_Q, removes the qth, outermost, subdiagonal of A: each of its elements is removed by an appropriate rotation, (P1), which is then applied to the corresponding pair of rows, (P2). The product of these rotations forms an orthogonal matrix Q_1^T and

$$R_1 = Q_1^T A,$$

where R_1 has $p + 1$ superdiagonals. The first, leftmost, processor of the linear mesh repeatedly computes (P1) while the $w - 1$ succeeding processors to its right each compute (P2), see Figure 2. The matrix A is input by diagonals, the qth subdiagonal enters processor 1, the diagonal processor $q + 1$ and the pth superdiagonal enters the rightmost, wth, processor. On output these diagonals are shifted one place to the left, the (q-1)st subdiagonal exits from processor 1, the diagonal from processor q and the (p+1)st superdiagonal from processor w. The rotations flow to the right until they leave processor w. The time from the first input to the last output is 2n steps if $m = n$ and $2(n+q-1)$ if $m > n$.

The QR decomposition of a band matrix with q subdiagonals, accomplished by routing the matrix through q successive L_Q-meshes, takes $2(n+q-1)$ steps. The ith mesh encountered removes the (q-i+1)st subdiagonal.

Elimination of a superdiagonal is achieved by multiplying from the right by an orthogonal matrix, which is the product of rotations to be applied to the columns of A, so

$$L_2 = AQ_2^T,$$

where L_2 has $q + 1$ subdiagonals. The data lines of the processors are reversed and the corresponding linear mesh, call it L_H, is a "mirror image" of the one for subdiagonal elimination. The processor computing (P1) is now the rightmost, wth processor, and it sends the rotations travelling towards the left. The reduction to lower Hessenberg form (in the upper triangular part only the first superdiagonal is nonzero) of a square matrix A is accomplished by sending A through $p - 1$ successive L_H-meshes; this takes $2(n+p-2)$ time steps.

In general, k subdiagonals (superdiagonals) of A are eliminated in $2(n+k-1)$ steps by sending A through k successive L_Q- (L_H-) meshes.

In the sections to come, the systolic arrays will be described in terms of the linear meshes, L_H and L_Q, eliminating superdiagonals and subdiagonals, respectively, - rather than processors eliminating individual matrix elements. The subscript Q (from QR) will be connected with removal of subdiagonals, while H (from lower Hessenberg form) is associated with elimination of superdiagonals.

4. Systolic Arrays for Reduction to Bidiagonal Form

To keep hardware to a minimmum, an algorithm for bandwidth reduction will be chosen that does not increase the number of nonzeroes per row (even temporarily).

The algorithm for a reduction to bidiagonal form of a banded matrix A to be presented here basically proceeds in two stages, removal of q subdiagonals followed by removal of p-1 superdiagonals (since $m > n$, subdiagonal removal decreases the order of the matrix, as well as the computation time and should be performed first). It will be assumed that the bandwidth of A does not exceed the size of the L_Q- or L_H-meshes, otherwise partitioning strategies have to be applied, see [9].

As for the first stage,

Compute the QR decomposition of $A_0 = A$, $R_1 = U_1^T A_0$

Remove the filled-in superdiagonals, $H_1 = R_1 V_1$

Let A_1 consist of the first n rows of H_1

```
For $i = 1 \ldots [n/p]$
    Compute the QR decomposition $R_{i+1} = U^T_{i+1}A_i$
    Remove the filled-in superdiagonals, $H_{i+1} = R_{i+1}V_i$
    Let $A_{i+1}$ be $H_{i+1}$ without its leading $p$ rows and
      columns.
```

The first two steps essentially reduce the problem size from $m\times n$ to $n\times n$, i.e., the QR decomposition causes the last q rows of R_1 to become zero. During the loop, the QR decomposition, which eliminates the zeroes in the lower triangular part, is followed by removal of the filled-in superdiagonals. This will restore the previously eliminated subdiagonals - save their p leading elements. Thus, disregarding the leading p rows and columns the whole process is repeated on the remaining matrix until the subdiagonal part has totally disappeared. After $[n/p]$ such iterations,

$$A_{[n/p]+1} = U^T_{[n/p]+1} \cdots U^T_1 \; A \; V_1 \cdots V_{[n/p]+1}$$

is an upper triangular matrix ($[x]$ denotes the smallest integer equal to or greater than x). On an array with q L_Q-meshes followed by q L_H-meshes, all meshes being of size w, stage 1 takes time proportional to $2n^2/p$. If on the order of n L_Q-meshes are available so that uninterrupted pipelining is possible, the computation time comes to about $4nq/p$ steps.

During the second stage, $n-2$ iterations are necessary to reduce the upper triangular matrix $A_{[n/p]+1}$ to bidiagonal form B. The iterations are similar to the ones above.

```
For $i = [n/p]+1 \dots [n/p]+n-1$
    Reduce $A_i$ to lower Hessenberg form $H_{i+1} = A_i V_i$
    Remove the filled-in subdiagonals, $R_{i+1} = U_{i+1}^T H_{i+1}$
    Let $A_{i+1}$ be $R_{i+1}$ without its leading row and column.
```

Eventually,

$$B = U^T_{[n/p]+n} \cdots U^T_{[n/p]+2} \, A_{[n/p]+1} \, V_{[n/p]+2} \cdots V_{[n/p]+n}.$$

The corresponding array comprising $p-1$ L_H-meshes succeeded by $p-1$ L_Q-meshes, each of size $p+1$, completes stage 2 in $2n^2+O(np)$ steps.

The example in Figure 3 illustrates several steps in the reduction to bidiagonal form of a matrix with three subdiagonals and two superdiagonals. Given an array for stage 1 with $2wq$ processors and one for stage 2 with $2(p^2-1)$ processors the reduction to bidiagonal form takes $2n^2+O(n^2/p)$ steps. If $O(n)$ such arrays are available the time reduces to $4np + O(nq/p)$.

For the computation of the singular vectors, the rotations forming the U_i^T are rerouted through the L_Q-meshes and applied to a $m\times m$ identity matrix, while rotations forming the V_i are input into L_H meshes to be applied to an $n\times n$ identity matrix. Since U and V are generally full matrices, they have to be determined by inputting submatrices of order $w/2$ [7] that "fit into" the L_Q- and L_H-meshes. Alternatively, another rectangular $2(m-1)w$ processor array as in [11] may be employed to which rotations are input as soon as they have left the L_Q- and L_H-meshes; the computation of the singular values and vectors can thus proceed concurrently.

5. A More Flexible Array for Reduction to Bidiagonal Form

Instead of having different arrays for stages 1 and 2, essentially one array can be shared by both of them. It consists of two separate parts, one succession of $\max(p,q)$ L_Q-meshes of size w and another one containing the same number of L_H-meshes of the same size. During an iteration of stage 1 the matrix is first entered into the L_Q-part and thereafter into the L_H-part. This order is reversed in stage 2.

But now the size of the meshes may be wider than the actual bandwidth of the matrix. Yet, the matrix must be entered "leftbound" into the L_Q-part and "rightbound" into the L_H-part, so that the doomed sub- or superdiagonal enters the processor computing (P1). Hence, before entering the L_H-part the matrix may have to be aligned to the right, and possibly to the left before input to the L_Q-part. Three different cases can occur.

If $p - 1 = q$, no alignment is necessary in stage 1, since the bandwidth is equal to the size of the meshes, while the number of meshes corresponds to the number of subdiagonals to be removed. During stage

2, the matrix has to be shifted by q places during each transition between L_Q- and L_H-parts and vice versa.

If $p - 1 < q$, no alignment occurs during stage 1. During stage 2, though, $q - (p-1)$ meshes are idle, i.e., they generate only identity rotations. Consider the reduction to Hessenberg form. After having traversed the first $p - 1$ L_H-meshes, the first superdiagonal is in the wth processor, the reduction is completed. However, the remaining $q - (p-1)$ meshes shift the matrix further to the right, each by one place, so it is "squeezed" out of the array to the right. To properly enter the L_Q-part it therefore has to be shifted to the left - by the distance it was squeezed out, which is $q - (p-1)$, plus the distance between the outermost, (p-1)st, subdiagonal and the first processor, i.e., $w - (p+1)$. Thus, after leaving the L_H-part the matrix must be shifted $w + q - 2p$ places to the left before entering the L_Q-part. It must be shifted the same distance to the right after output from the L_Q-part. Figure 4 illustrates this situation.

If $q < p - 1$, consider the QR decomposition in the first stage. After the first q L_Q-meshes have been traversed, the remaining $p - 1 - q$ meshes will squeeze the matrix out to the left, one place per mesh. To enter the L_H-part, the matrix is shifted to the right - by $p - q - 1$ places, equalling the distance by which it was squeezed out. During the second stage shifting occurs by q places.

Moreover, in [10] it is shown that, by slight reprogramming of the processors, one type of mesh can fulfill the functions of both L_Q- and L_R-meshes. A reduction to bidiagonal form of a matrix with bandwidth w

is then performed on k meshes of size at least w. The alignment can be limited to k places if every kth processor in a mesh is a (P1) processor.

6. A Systolic Array for Computation of the Singular Values

The processor which implements the singular value computation for bidiagonal matrices is a special case of the one for eigenvalue computation [7]. It executes one step of the Golub-Reinsch iteration [5].

Let P_{i+1}^T be a rotation removing the (2,1) element of
$(B_i^T B_i - s_{i+1} I)$

Compute $C_{i+1} = B_i P_{i+1}$

Compute B_{i+1} by reducing C_{i+1} to bidiagonal form.

Notice that C_{i+1} differs from B_i only in the first two rows and columns. One can assume, that it is computed separately and then passed through a network, built around the processor [7,10] depicted in Figure 5, which might be viewed as a conglomerate of processors computing (P1) and (P2):

<u>Step 1</u>: values already in cell are r_1, s_1, r_2, s_2

a) input s_3

b) generate P so that $P\begin{pmatrix} r_1 \\ s_1 \end{pmatrix} = \begin{pmatrix} r_1' \\ 0 \end{pmatrix}$

c) compute $\begin{pmatrix} r_1' & r_2' & r_3' \\ 0 & s_2' & s_3' \end{pmatrix} = P \begin{pmatrix} r_1 & r_2 & 0 \\ s_1 & s_2 & s_3 \end{pmatrix}$

d) output r_1', P and retain r_2', s_2', r_3', s_3'

<u>Step 2</u>:

e) input t_2, t_3

f) generate Q so that $(r_2', r_3')Q = (r_2'', 0)$

g) compute $\begin{pmatrix} r_2'' & 0 \\ s_2'' & s_3'' \\ t_2' & t_3' \end{pmatrix} = \begin{pmatrix} r_2' & r_3' \\ s_2' & s_3' \\ t_2 & t_3 \end{pmatrix} Q$

h) output t_2', Q and retain $(s_2'', t_2', s_3'', t_3')$
as (r_1, s_1, r_2, s_2) for the next operation
of the cell.

For the singular value computation this means that in every two steps the processor computes one pass through the following loop, which generates B_{i+1} $(C_{i+1,2} = C_{i+1})$,

```
For j = 2 ... n-1
    Generate a rotation $P^T_{j,j-1}$ to annihilate element (j,j-1)
      of $C_{i+1,j}$
    Apply it, generating a fill-in at position (j-1,j+1) of
      $P^T_{j,j-1}C_{i+1,j}$
    Generate a rotation $P_{j-1,j+1}$ to annihilate position
      (j-1,j+1)
    Apply it, generating a fill-in at position (j+1,j) of
```

$$C_{i+1,j+1} = (P^T_{j,j-1} C_{i+1,j})P_{j-1,j+1}.$$

The total time to generate $B_{i+1} = C_{i+1,n}$ is $2n + 2$; the corresponding network is shown in Figure 6. On the average 2 to 3 iterations are required per singular value, bringing its computation time to 6n steps. Note that the input t_2 is zero except in the first step.

To complete the computation of the singular values, the rotations generated above are applied to the partially computed singular vector matrices from the bandwidth reduction step.

The incorporation of the preceeding designs into a system computing all the singular values is described in [7]. One remaining problem is the efficient computation of the shift values for convergence acceleration. Presently, values taken from the trailing end of the matrix have to be incorporated into an orthogonal rotation which is applied to the leading rows. Therefore it is not obvious how to pipeline several singular value iterations while at the same time maintaining quadratic (and in practice cubic) convergence.

References

1. Brent, R. P. and Luk, F. T., "A Systolic Architecture for the Singular Value Decomposition", Report TR-CS-82-09, Department of Computer Science, The Australian National University, 1982.

2. Brent, R. P., Luk, F. T. and Van Loan, C. F., "Computation of the Singular Value Decomposition Using Mesh Connected Processors", Report TR-82-528, Department of Computer Science, Cornell University, 1983.

3. Brent, R. P., Luk, F. T. and Van Loan, C. F., "Computation of the Generalised Singular Value Decomposition Using Mesh-Connected Processors", Report TR 83-563, Department of Computer Science, Cornell University, 1983.

4. Finn, A. M., Luk, F. T. and Pottle, C., "A Systolic Array Computation of the Singular Value Decomposition", Proc. SPIE Symposium, 341 (Real Time Signal Processing V), pp. 35-43, 1982.

5. Golub, G. H. and Reinsch, C., "Singular Value Decomposition and Least Squares Solutions", Numer. Math., Vol. 14, pp. 403-420, 1970.

6. Golub, G. H. and Van Loan, C. F., Matrix Computations, The Johns Hopkins University Press, Baltimore, MD, 1983.

7. Heller, D. E. and Ipsen, I. C. F., "Systolic Networks for Orthogonal Equivalence Transformations and their Applications", Proc. 1982 Conf. on Advanced Research in VLSI, MIT, Cambridge, MA, pp. 113-122, 1982.

8. Heller, D. E. and Ipsen, I. C. F., "Systolic Networks for Orthogonal Decompositions", SIAM J. Sci. Statist. Comput., Vol. 4, pp. 261-269, 1983.

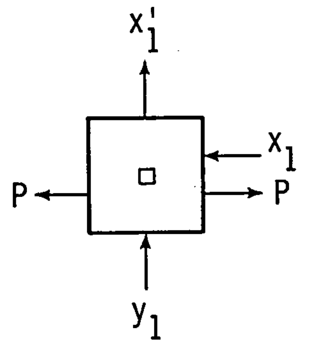
9. Ipsen, I. C. F., "Stable Matrix Computations in VLSI", Ph.D. Thesis, The Pennsylvania State University, 1983.

10. Ipsen, I. C. F., "Singular Value Computations with Systolic Arrays", Report RR-291, Department of Computer Science, Yale University, 1983.

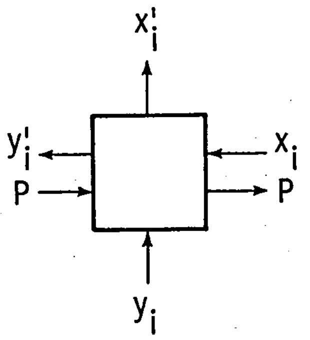
11. Schreiber, R., "A Systolic Architecture for the Singular Value Decomposition", Proc. 1er Colloque International sur les Méthodes Vectorielles et Paralèles en Calcul Scientifique, Electricité de France, Bulletin de la Direction des Etudes et Recherches, Série C No. 1, 1983.

12. Schreiber, R., "On the Systolic Arrays of Brent, Luk and Van Loan for the Symmetric Eigenvalue and Singular Value Problems", Report TRITA-NA-8311, NADA, KTH Stockholm, 1983.

13. Speiser, J. M. and Whitehouse, H. J., "Architectures for Real Time Matrix Operations", Proc. Government Microcircuit Application Conf., Houston, Texas, 1980.

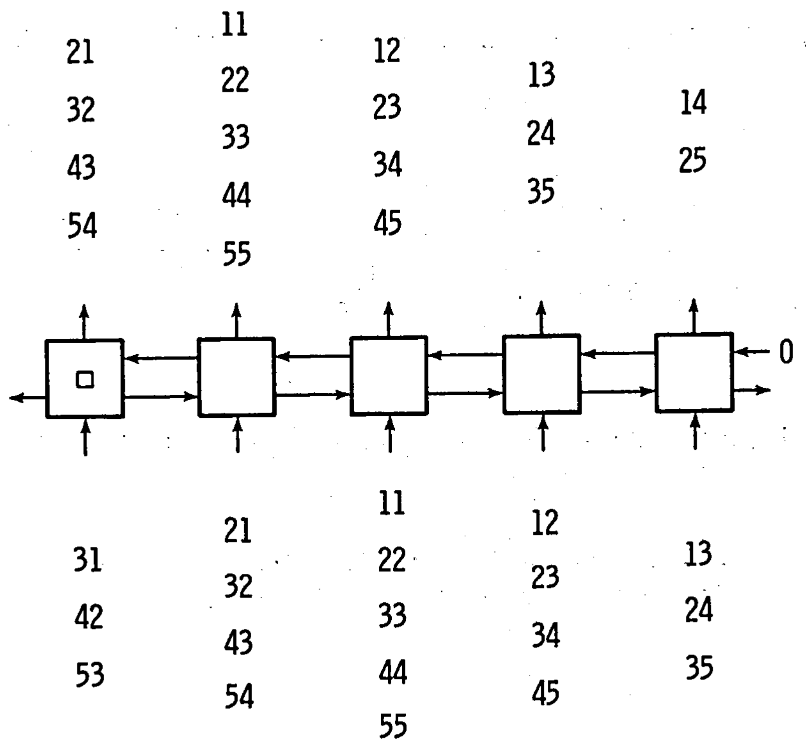


Processor for (P1)



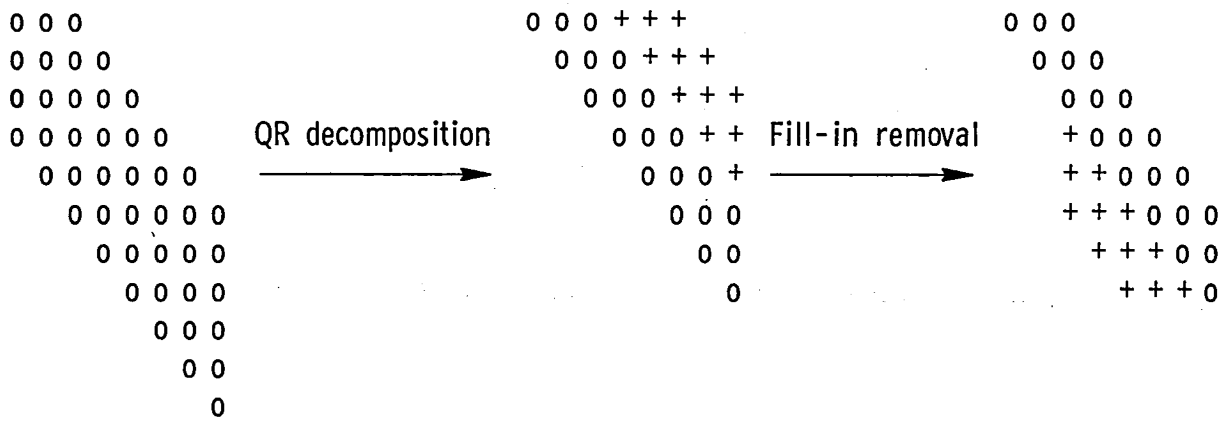
Processor for (P2)

Figure 1. Processors for Elimination of Subdiagonals

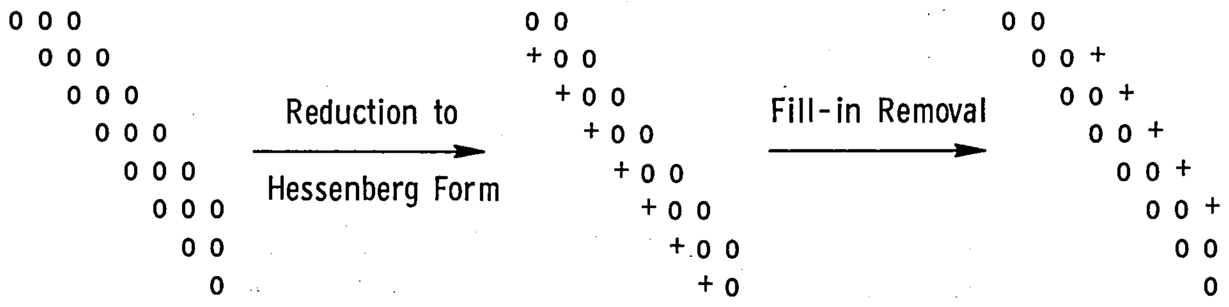


Matrix elements are represented by their subscripts.

Figure 2. L_Q-Mesh for Subdiagonal Elimination of Size w = 5 and Input Matrix with p = q = 2



(a) Stage 1: QR Decomposition Followed by Removal of Superdiagonals.



(b) Stage 2: Reduction to Lower Hessenberg Form Followed by Removal of Subdiagonals.

Figure 3. Several Steps in the Reduction to Bidiagonal Form of a Matrix with m = 11, n = 8, p = 2, q = 3

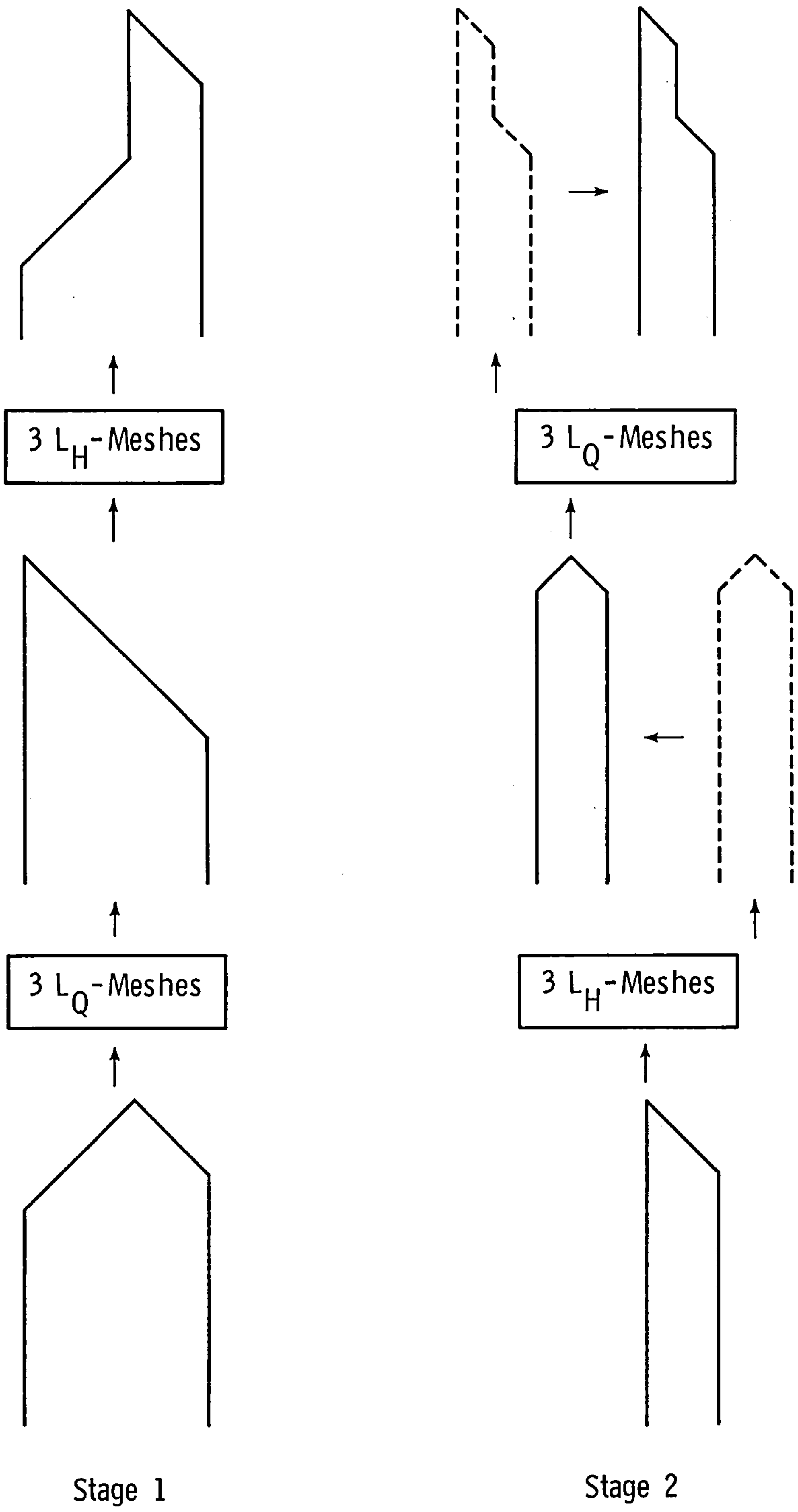


Figure 4. Input/Output Format and Alignment for Systolic Arrays Performing a Reduction to Bidiagonal Form in Case $p-1 < q$

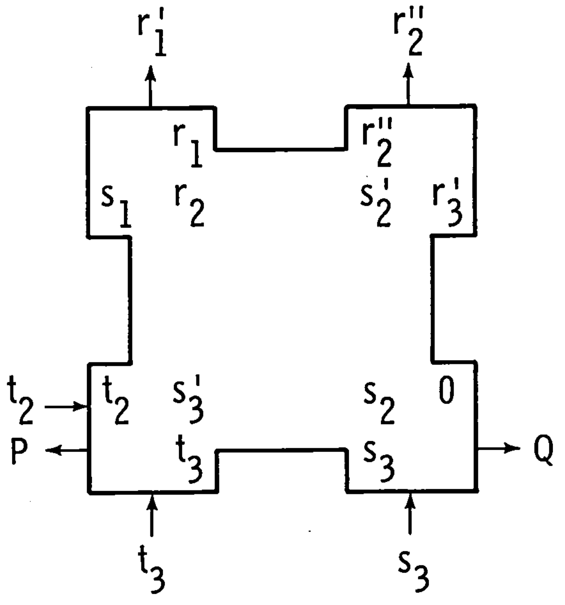
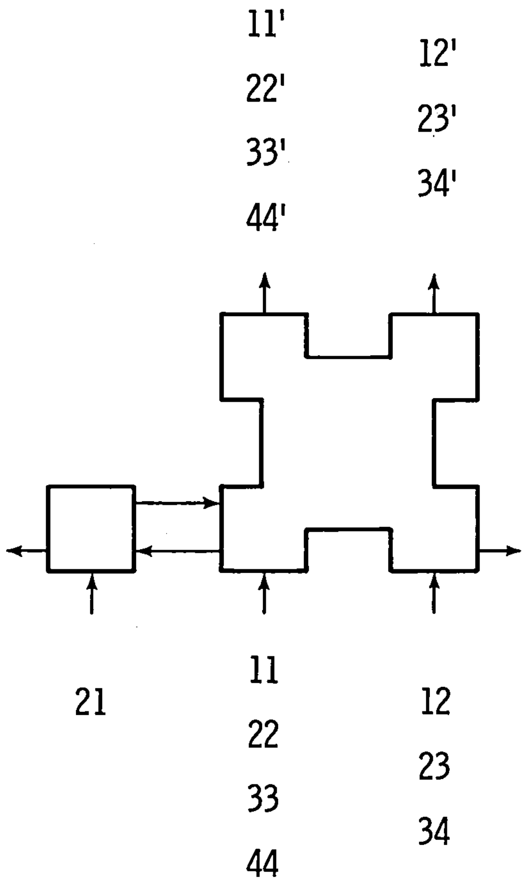


Figure 5. Processor for the Computation of Singular Values



Matrix elements are represented by their subscripts.

Figure 6. Systolic Array for the Computation of Singular Values with Input/Output Format

1. Report No. NASA CR-172396 ICASE Report No. 84-30	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle Singular Value Decomposition with Systolic Arrays		5. Report Date July 1984 6. Performing Organization Code
7. Author(s) Ilse C. F. Ipsen		8. Performing Organization Report No. 84-30
9. Performing Organization Name and Address Institute for Computer Applications in Science and Engineering Mail Stop 132C, NASA Langley Research Center Hampton, VA 23665		10. Work Unit No. 11. Contract or Grant No. NAS1-17070
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, D.C. 20546		13. Type of Report and Period Covered Contractor Report 14. Sponsoring Agency Code 505-31-83-01

15. Supplementary Notes

Langley Technical Monitor: Robert H. Tolson
Final Report

16. Abstract

Systolic arrays for determining the Singular Value Decomposition of a $m \times n$, $m > n$, matrix A of bandwidth w are presented. After A has been reduced to bidiagonal form B by means of Givens plane rotations, the singular values of B are computed by the Golub-Reinsch iteration. The products of plane rotations form the matrices of left and right singular vectors.

Assuming each processor can compute or apply a plane rotation, O(wn) processors accomplish the reduction to bidiagonal form in O(np) steps, where p is the number of superdiagonals. A constant number of processors can then determine each singular value in about 6n steps. The singular vectors are computed by rerouting the rotations through the arrays used for the reduction to bidiagonal form, or else "along the way" by employing another rectangular array of O(wm) processors.

17. Key Words (Suggested by Author(s)) singular value decomposition, VLSI, systolic arrays, parallel computation	18. Distribution Statement 59 - Mathematics & Computer Science General 61 - Computer Programming & Software 64 - Numerical Analysis Unclassified - Unlimited

19. Security Classif. (of this report) Unclassified	20. Security Classif. (of this page) Unclassified	21. No. of Pages 24	22. Price A02

LANGLEY RESEARCH CENTER
3 1176 00518 1723