Chapter 9

Singular Value Decomposition

This chapter describes the CMSSL singular value decomposition routines listed below.

Bidiagonalization, corresponding basis transformation, and singular value decomposition of bidiagonal systems (Sections 9.1, 9.2, and 9.3):

gen_bidiag — Given a real CM array ***A*** containing one or more dense matrices A of dimensions m x n with $m \geq n$, finds bidiagonal matrices B and matrices U and V such that, for each instance, $A = UBV^{T}$.

gen_bidiag_apply_u — Given the results of **gen_bidiag** and a real CM array ***C*** containing arbitrary matrices or vectors C, computes $C = UC$ for each instance.

gen_bidiag_apply_v_tra — Computes $C = V^{T}C$ for each instance.

gen_bidiag_apply_u_tra — Computes $C = U^{T}C$ for each instance.

gen_bidiag_apply_v — Computes $C = VC$ for each instance.

bidiag_to_gen_left — Computes $C = UP_1C$ for each instance, where P_1 is the permutation giving the correspondence between standard and block cyclic row order.

bidiag_to_gen_right — Computes $C = CP_2{}^{-1}V^{T}$ for each instance, where P_2 is the permutation giving the correspondence between standard and block cyclic column order.

Copyright © 1994 Thinking Machines Corporation

gen_to_bidiag_left Computes $C = CP_1^{-1}U^T$ for each instance.

gen_to_bidiag_right Computes $C = VP_2C$ for each instance.

bidiag_svd_singular_values Given one or more real bidiagonal matrices B, performs the singular value decomposition $B = U\Sigma V^T$, where U and V are orthogonal and Σ is diagonal and contains the singular values of B. Returns the singular values of B.

bidiag_svd_singular_vectors Given one or more real bidiagonal matrices B, and the singular values for each B, computes the left and right singular vectors corresponding to each singular value. That is, for each instance, **bidiag_svd_singular_vectors** returns the requested portions of U and V^T, where $B = U\Sigma V^T$ is the singular value decomposition of B.

Singular value decomposition of dense real matrices (Section 9.4):

gen_bidiag_singular_system Computes the singular values and, if desired, the singular vectors of one or more dense real matrices. Combines the functionality of the **gen_bidiag**, **bidiag_svd_singular_values**, **bidiag_svd_singular_vectors**, **bidiag_to_gen_left**, and **bidiag_to_gen_right** routines.

Section 9.5 lists references.

Version 3.2, April 1994
Copyright © 1994 Thinking Machines Corporation

9.1 Bidiagonalization and Corresponding Basis Transformation

The **gen_bidiag** routine transforms dense real matrices to bidiagonal form. That is, given one or more real dense matrices A of dimensions m x n with $m \geq n$, **gen_bidiag** finds, for each A, a bidiagonal matrix B and matrices U and V such $A = UBV^T$. Auxiliary routines apply the transformations U, V, U^T, and V^T to arbitrary matrices or vectors, and allow you to transform an arbitrary set of vectors between the basis of the original dense matrix A and the basis of the corresponding bidiagonal matrix. The man page that follows provides details.

9.1.1 Load Balancing

The bidiagonalization and basis transformation routines use load balancing to enhance performance. (Blocking is planned for future releases. A block size is included in the **gen_bidiag** calling sequence, but in the current release it must be set to 1.) Blocking and load balancing are described in Chapter 15. Details specific to the bidiagonalization routines are provided in the man page that follows.

9.1.2 Numerical Stability

The routines described in this section are stable.

Copyright © 1994 Thinking Machines Corporation

Bidiagonalization and Corresponding Basis Transformation

Given one or more real dense matrices A of dimensions m x n with $m \geq n$, the **gen_bidiag** routine finds, for each A, a bidiagonal matrix B and matrices U and V such $A = UBV^{T}$. The **gen_bidiag_apply_** routines apply the transformations U, V, U^{T}, and V^{T} to arbitrary matrices or vectors. The **bidiag_to_gen_** and **gen_to_bidiag_** routines allow you to transform an arbitrary set of vectors between the basis of the original dense matrix A and the basis of the corresponding bidiagonal matrix.

SYNTAX

setup = **gen_bidiag**	(***d***, ***e***, ***A***, *m*, *n*, *row_axis*, *col_axis*, *nblock*, *ier*)
gen_bidiag_apply_u	(***C***, ***A***, *setup*, *nrhs*, *ier*)
gen_bidiag_apply_v_tra	(***C***, ***A***, *setup*, *nrhs*, *ier*)
gen_bidiag_apply_u_tra	(***C***, ***A***, *setup*, *nrhs*, *ier*)
gen_bidiag_apply_v	(***C***, ***A***, *setup*, *nrhs*, *ier*)
bidiag_to_gen_left	(***C***, ***A***, *setup*, *nrhs*, *ier*)
bidiag_to_gen_right	(***C***, ***A***, *setup*, *nrhs*, *ier*)
gen_to_bidiag_left	(***C***, ***A***, *setup*, *nrhs*, *ier*)
gen_to_bidiag_right	(***C***, ***A***, *setup*, *nrhs*, *ier*)
deallocate_gen_bidiag	(*setup*)

ARGUMENTS

setup	Scalar integer variable. Setup ID. You must supply the value returned by **gen_bidiag** when calling any of the other routines.
d	Real CM array with the same precision as ***A***, containing a vector for each matrix A in ***A***. The array ***d*** must have either the same rank as ***A*** or rank one less than that of ***A***, as follows:

Version 3.2, April 1994
Copyright © 1994 Thinking Machines Corporation

- If ***d*** has the same rank as ***A***, then the vector elements are counted by axis *row_axis*, which must have extent greater than or equal to *n*; axis *col_axis* must have extent 1.
- If ***d*** has rank one less than that of ***A***, then axis *col_axis* is omitted. The vector elements are counted by axis *row_axis* (if *row_axis* < *col_axis*) or by axis *row_axis*−1 (if *col_axis* < *row_axis*).

The remaining axes of ***d*** must be instance axes that match the instance axes of ***A*** in extent and order of declaration. The arrays ***d*** and ***e*** must be distinct.

Upon successful completion of **gen_bidiag**, each vector in ***d*** contains the diagonal entries of the *n* x *n* bidiagonal matrix associated with the corresponding matrix *A* in ***A***. This bidiagonal matrix is in block cyclic form; details are provided in the Description section.

e Real CM array with the same precision, rank, and shape as ***d***. Thus, each vector in ***e*** corresponds to a matrix *A* within ***A***. The arrays ***d*** and ***e*** must be distinct.

Upon successful completion of **gen_bidiag**, elements 2 through *n* of each vector in ***e*** contain the superdiagonal entries of the bidiagonal matrix associated with the corresponding matrix *A*. This bidiagonal matrix is in block cyclic form; details are provided in the Description section. The first element of each vector in ***e*** is arbitrary.

A Real CM array of rank greater than or equal to 2. When you call **gen_bidiag**, ***A*** must contain one or more dense matrices *A* that you want to bidiagonalize. Each *A* must have dimensions *m* x *n* where $m \geq n$. Axis *row_axis* counts the rows of the matrices *A*; axis *col_axis* counts the columns. If axes *row_axis* and *col_axis* have extents greater than *m* and *n*, then each instance of *A* consists of the upper left-hand *m* x *n* elements of a larger matrix within the array ***A***. ***A*** must be distinct from ***C***.

Upon successful return from **gen_bidiag**, ***A*** contains information needed by the other routines. When you call any of the other routines, supply in ***A*** the values **gen_bidiag** returned in ***A***.

m Scalar integer variable. The number of rows in each matrix *A* within ***A***. Must be greater than or equal to *n*.

Copyright © 1994 Thinking Machines Corporation

n Scalar integer variable. The number of columns in each matrix *A* within ***A***. Must be less than or equal to *m*.

row_axis Scalar integer variable. Identifies the axis of ***A*** that counts the rows of each matrix *A*. (For definitions of row axes and column axes, see Chapter 1.)

col_axis Scalar integer variable. Identifies the axis of ***A*** that counts the columns of each matrix *A*. (For definitions of row axes and column axes, see Chapter 1.)

nblock Scalar integer variable. Block size. In the current release, you must set *nblock* to 1.

C Real CM array that is distinct from ***A*** and has the same precision as ***A***. Must have either the same rank as ***A*** or rank one less than that of ***A***. On input, ***C*** must contain one or more matrices or vectors *C*. The instance axes of ***C*** must match those of ***A*** in extent and order of declaration. Thus, each matrix or vector *C* within ***C*** corresponds to a matrix *A* within ***A***.

Each instance *C* may consist of the upper left-hand elements of a larger matrix or vector within ***C***. The following restrictions hold:

- When you call **gen_bidiag_apply_u**, **gen_bidiag_apply_u_tra**, or **bidiag_to_gen_left**, the number of rows in each *C* must be greater than or equal to *m*.
- When you call **gen_to_bidiag_left**, the number of columns in each *C* must be greater than or equal to *m*.
- When you call **gen_bidiag_apply_v**, **gen_bidiag_apply_v_tra**, or **gen_to_bidiag_right**, the number of rows in each *C* must be greater than or equal to *n*.
- When you call **bidiag_to_gen_right**, the number of columns in each *C* must be greater than or equal to *n*.

The following restrictions also hold:

- If each instance *C* within ***C*** consists of only one vector (*nrhs* = 1), you may represent *C* in either of the following ways:

Version 3.2, April 1994
Copyright © 1994 Thinking Machines Corporation

- It may have rank 2 with number of columns = 1. The rows of each *C* must be counted by axis *row_axis*; the single column must be counted by axis *col_axis*.

- It may have rank 1. The elements of each *C* must be counted by axis *row_axis* (if *row_axis* < *col_axis*) or by axis *row_axis*−1 (if *col_axis* < *row_axis*). For an example, see the Notes section.

- If each instance *C* within ***C*** consists of multiple right-hand-side vectors (*nrhs* > 1), the rows and columns of *C* must be counted by axes *row_axis* and *col_axis*, respectively.

Upon return from **gen_bidiag_apply_u**, **gen_bidiag_apply_v_tra**, **gen_bidiag_apply_u_tra**, or **gen_bidiag_apply_v**, each *C* is overwritten with the product UC, $V^{T}C$, $U^{T}C$, or VC, respectively.

Upon return from **bidiag_to_gen_left**, **bidiag_to_gen_right**, **gen_to_bidiag_left**, or **gen_to_bidiag_right**, each *C* is overwritten with the product UP_1C, $CP_2{}^{-1}V^{T}$, $CP_1{}^{-1}U^{T}$, or VP_2C, respectively, where P_1 is the permutation giving the correspondence between standard and block cyclic row order and P_2 is the permutation giving the correspondence between standard and block cyclic column order. See the Description section for details.

nrhs Scalar integer variable. The number of columns of each instance *C* within ***C***. Must be greater than or equal to 1.

ier Scalar integer variable. Return code; set to 0 upon successful return.

Values between −1 and −9, inclusive, indicate problems with ***A*** or ***C***:

−1 Invalid array home. The array must be a CM array.

−2 Invalid rank (must be $\geq$ 2 for ***A*** or $\geq$ 1 for ***C***).

−3 Invalid column extent.

−4 Invalid row extent.

−9 Invalid data type; must be real (single- or double-precision).

Copyright © 1994 Thinking Machines Corporation

Values that are multiples of −10 indicate problems with non-array arguments:

−10 System failed to allocate the setup object, *setup*.

−20 *m*, *n*, or *nrhs* is invalid; all must be greater than 0 and *m* must be greater than or equal to *n*.

−30 *row_axis* or *col_axis* is invalid. $1 \leq row_axis, col_axis \leq$ rank (***A***) must be true, and *row_axis* and *col_axis* must not be equal.

−40 *nblock* is invalid. It must be 1 in the current release.

−60 *nrhs* is invalid (must be greater than or equal to 1).

−100 *setup* is invalid. (You did not supply the value returned by **gen_bidiag**.)

Values between −102 and −108, inclusive, indicate problems with the consistency of ***A*** or ***C*** in one of the routines following a bidiagonalization call:

−102 The rank of ***A*** or ***C*** is invalid (must be ≥ 2 for ***A*** or ≥ 1 for ***C***), or is inconsistent with the rank of ***A*** in the factorization call.

−105 The extents of the instance axes of ***A*** or ***C*** are inconsistent with those of ***A*** in the factorization call.

−108 The data type of ***A*** or ***C*** is inconsistent with that of ***A*** in the factorization call.

DESCRIPTION

Given a real CM array ***A*** containing one or more dense matrices A of dimensions $m \times n$ with $m \geq n$, and a real CM array ***C*** containing arbitrary matrices or vectors C, the routines described in this man page perform the following operations for each instance:

gen_bidiag $A = UBV^{T}$ (B bidiagonal)

gen_bidiag_apply_u $C = UC$

gen_bidiag_apply_v_tra $C = V^{T}C$

Version 3.2, April 1994
Copyright © 1994 Thinking Machines Corporation

gen_bidiag_apply_u_tra	$C = U^{\mathrm{T}}C$
gen_bidiag_apply_v	$C = VC$
bidiag_to_gen_left	$C = UP_1C$
bidiag_to_gen_right	$C = CP_2{}^{-1}V^{\mathrm{T}}$
gen_to_bidiag_left	$C = CP_1{}^{-1}U^{\mathrm{T}}$
gen_to_bidiag_right	$C = VP_2C$
deallocate_gen_bidiag	Deallocates the processing element memory allocated by **gen_bidiag**.

P_1 is the permutation giving the correspondence between standard and block cyclic row order and P_2 is the permutation giving the correspondence between standard and block cyclic column order. These permutations depend on the array size and layout, the partition size, and the blocking factor you supply (the blocking factor must be 1 in the current release).

Setup and Deallocation. The **gen_bidiag** routine allocates processing element storage and returns a setup ID, *setup*. You must supply this setup ID in any of the other subroutines described in this man page, as long as you are working with the same bidiagonaliztion. You can follow one call to **gen_bidiag** with multiple calls to the other routines, thus avoiding the overhead of bidiagonalizing the same matrix or matrices repeatedly.

The **deallocate_gen_bidiag** routine deallocates the processing element storage allocated by **gen_bidiag**, and invalidates the associated setup ID. When you have finished working with a given bidiagonalization, be sure to call **deallocate_gen_bidiag** to deallocate the associated memory.

Block Cyclic Ordering. Effectively, **gen_bidiag** factors a block cyclic permutation, A_c, of A into

$$A_c = U_c B_c V_c{}^{\mathrm{T}}$$

where B_c is block cyclic bidiagonal. By definition,

$$A_c = P_1{}^{-1}AP_2.$$

We therefore have

Copyright © 1994 Thinking Machines Corporation

$$\begin{aligned} A = UBV^{\mathrm{T}} &= P_1 A_c {P_2}^{-1} \\ &= P_1 \left(U_c B_c {V_c}^{\mathrm{T}}\right) {P_2}^{-1} \\ &= (P_1 U_c)({P_1}^{-1} P_1)\, B_c\, ({P_2}^{-1} P_2)({V_c}^{\mathrm{T}} {P_2}^{-1}) \\ &= (P_1 U_c {P_1}^{-1})\, (P_1 B_c {P_2}^{-1})\, (P_2 {V_c}^{\mathrm{T}} {P_2}^{-1}) \end{aligned}$$

from which we choose to define

$$\begin{aligned} U &= P_1 U_c {P_1}^{-1} \\ B &= P_1 B_c {P_2}^{-1} \\ V^{\mathrm{T}} &= P_2 {V_c}^{\mathrm{T}} {P_2}^{-1} \end{aligned}$$

Use of the Routines. The U and V that are applied by the **gen_bidiag_apply_** routines are the non-block-cyclic U and V, not U_c and V_c. By supplying the identity matrix to the **gen_bidiag_apply_** routines, you can obtain U, V, U^{T}, and V^{T} for use in any computation. However, the matrix whose diagonal and superdiagonal are returned in ***d*** and ***e*** is the block cyclic bidiagonal matrix B_c. The four **bidiag_to_gen_** and **gen_to_bidiag_** routines are provided for convenience in cases where you want to transform a set of vectors between the basis of the original dense matrix A and the basis of B_c.

For example, if you supply the values returned in ***d*** and ***e*** to the **bidiag_svd_singular_vectors** routine (described in Section 9.3), that routine returns the singular vectors of B_c. If $B_c = U_{Bc} \Sigma {V_{Bc}}^{\mathrm{T}}$ is the singular value decomposition of B_c, then we have (from above):

$$\begin{aligned} A &= UBV^{\mathrm{T}} \\ &= U\, (P_1 B_c {P_2}^{-1})\, V^{\mathrm{T}} \\ &= UP_1\, (U_{Bc} \Sigma {V_{Bc}}^{\mathrm{T}})\, {P_2}^{-1} V^{\mathrm{T}} \\ &= (UP_1 U_{Bc})\, \Sigma\, (V_{Bc}^{\mathrm{T}} {P_2}^{-1} V^{\mathrm{T}}) \\ &= (UP_1 U_{Bc})\, \Sigma\, (VP_2 V_{Bc})^{\mathrm{T}} \end{aligned}$$

which is the singular value decomposition of A. This analysis shows that if U_{Bc} contains the left singular vectors of B_c, then to obtain the left singular values of A — that is, UP_1U_{Bc} — you must apply **bidiag_to_gen_left** to U_{Bc}. Similarly, if ${V_{Bc}}^{\mathrm{T}}$ contains the right singular vectors of B_c, then to obtain the right singular vectors of A — that is, $(VP_2V_{Bc})^{\mathrm{T}}$ — you must apply **bidiag_to_gen_right** to ${V_{Bc}}^{\mathrm{T}}$. See the Notes section, below, for a further detail.

Matrix Shapes. For each matrix A in $\boldsymbol{A}$, **gen_bidiag** finds a bidiagonal matrix B_1 and orthogonal matrices U and V such that $A = UBV^{\mathrm{T}}$, where B is defined by

Version 3.2, April 1994
Copyright © 1994 Thinking Machines Corporation

$$B \equiv \begin{bmatrix} B_1 \\ 0 \end{bmatrix}.$$

The following diagram shows the dimensions of these matrices:

$$\underset{m \times n}{\begin{bmatrix} A \end{bmatrix}} = \underset{m \times m}{\begin{bmatrix} U \end{bmatrix}} \underset{m \times n}{\begin{bmatrix} B_1 \\ \hline 0 \end{bmatrix}} \underset{n \times n}{\begin{bmatrix} V^{\mathrm{T}} \end{bmatrix}}$$

If you supply **gen_bidiag_apply_u** with the $m \times m$ identity matrix, the U that overwrites the identity matrix is $m \times m$ (as shown above) and orthogonal. If you supply **gen_bidiag_apply_u** with the $m \times n$ matrix

$$\underset{m \times n}{\begin{bmatrix} 1 & & & \\ & 1 & & \\ & & \ddots & \\ & & & 1 \\ & & 0 & \end{bmatrix}}$$

then this matrix is overwritten by the $m \times n$ matrix U from the equation

$$\underset{m \times n}{\begin{bmatrix} A \end{bmatrix}} = \underset{m \times n}{\begin{bmatrix} U \end{bmatrix}} \underset{n \times n}{\begin{bmatrix} B \end{bmatrix}} \underset{n \times n}{\begin{bmatrix} V^{\mathrm{T}} \end{bmatrix}}.$$

In this formulation, $B = B_1$.

Copyright © 1994 Thinking Machines Corporation

NOTES

Distinct Variables. The input CM arrays ***A*** and ***C*** must be distinct variables. The arrays ***d*** and ***e*** must also be distinct.

Singular Value Decomposition. As mentioned earlier, if you use these routines in conjunction with the **bidiag_svd_singular_vectors** routine, you will probably want to apply **bidiag_to_gen_left** to U_{Bc} to obtain UP_1U_{Bc}, the left singular vectors of A. U_{Bc} is n x n, but **bidiag_to_gen_left** requires the input matrix to have at least m rows. Supply U_{Bc} to **bidiag_to_gen_left** in the following form:

$$\begin{bmatrix} U_{Bc} \\ 0 \end{bmatrix}$$

m x n

This causes **bidiag_svd_singular_vectors** to return UP_1U_{Bc} as an m x n matrix, which is standard format for the left singular vectors of A.

Include the CMSSL Header File. Because the **gen_bidiag** routine is a function, you must include the line

```
INCLUDE '/usr/include/cm/cmssl-cmf.h'
```

at the top of any program module that calls these routines. This file declares the types of the CMSSL functions and symbolic constants.

Preservation of Argument Values. The internal variable *setup* is required for communicating information between **gen_bidiag** and the other routines. Do not modify the contents of this variable.

Numerical Stability. These routines are stable.

Numerical Complexity. If the matrices A have dimensions m x n, the matrices C have *nrhs* right-hand sides, and I is the number of instances, the routines described in this man page use the following numbers of floating-point operations:

Version 3.2, April 1994
Copyright © 1994 Thinking Machines Corporation

Routine(s)	Number of Floating-Point Operations
gen_bidiag	$4n^2(m-n/3)I$
gen_bidiag_apply_u	$(2nrhs)(2m-n)nI$
gen_bidiag_apply_u_tra	$(2nrhs)(2m-n)nI$
bidiag_to_gen_left	$(2nrhs)(2m-n)nI$
gen_to_bidiag_left	$(2nrhs)(2m-n)nI$
gen_bidiag_apply_v	$(2nrhs)n^2I$
gen_bidiag_apply_v_tra	$(2nrhs)n^2I$
bidiag_to_gen_right	$(2nrhs)n^2I$
gen_to_bidiag_right	$(2nrhs)n^2I$

Rank of *C*. The following example illustrates the options for defining the rank of ***C***. Assume that each *C* in ***C*** is a single vector of length *m*. Suppose ***A***, *m*, *n, row_axis*, and *col_axis* are defined as follows:

A (8, 10, 4)
$m = 8$
$n = 4$
$row_axis = 1$
$col_axis = 3$

You may define ***C*** in either of the two following (equivalent) ways:

C (8, 10, 1)
C (8, 10)

On the other hand, if you define

A (4, 10, 8)
$m = 8$
$n = 4$
$row_axis = 3$
$col_axis = 1$

then the possibilities for ***C*** are as follows:

C (1, 10, 8)
C (10, 8)

Acknowledgments. The routines described in this section are the result of collaborative development between Thinking Machines Corporation and the Danish

Copyright © 1994 Thinking Machines Corporation

Computing Center for Research and Education (UNI−C) (contributor: Susanne Balle).

EXAMPLES

Sample CM Fortran code that uses the routines described above can be found on-line in the subdirectory

```
/bidiag/cmf
```

of a CMSSL examples directory whose location is site-specific.

Version 3.2, April 1994
Copyright © 1994 Thinking Machines Corporation

9.2 Singular Values of Bidiagonal Matrices

The **bidiag_svd_singular_values** routine computes all the singular values of one or more real bidiagonal matrices. A detailed description of this routine is provided in the man page at the end of this section.

9.2.1 Implementation

In the current release, **bidiag_svd_singular_values** transforms a bidiagonal matrix to a tridiagonal matrix. Given a bidiagonal matrix B of dimensions n x n, consider the $2n$ x $2n$ matrix

$$M \equiv \begin{bmatrix} 0 & B^{\mathrm{T}} \\ B & 0 \end{bmatrix}.$$

Suppose B has diagonal entries a_1, a_2, ..., a_n and superdiagonal entries b_1, b_2, ..., b_{n-1}, and that we permute the rows and columns of M to order them as follows: 1, $n+1$, 2, $n+2$, ..., n, $2n$. Then M is orthogonally similar to the symmetric tridiagonal matrix M' with zeros on the diagonal and off-diagonal elements a_1, b_1, a_2, b_2, ..., b_{n-1}, a_n (the rows and columns of M' are also permuted to the new order just mentioned). The singular values of B are the absolute values of the eigenvalues of M'. For further details, see reference 2 in Section 9.5.

In the current release, **bidiag_svd_singular_values** calls the **sym_tridiag_eigenvalues** routine to find the eigenvalues of M' using parallel bisection. The **sym_tridiag_eigenvalues** routine is described in Chapter 8.

Copyright © 1994 Thinking Machines Corporation

Singular Values of Bidiagonal Matrices

The **bidiag_svd_singular_values** routine computes the singular values of one or more real bidiagonal matrices of the same order. The diagonal and superdiagonal matrix elements are stored in two CM vectors or arrays.

SYNTAX

bidiag_svd_singular_values (***d, e,*** *axis, tolerance, ier*)

ARGUMENTS

d	Real CM array consisting of one or more vectors lying along axis *axis*; all other axes are instance axes. On input, each vector must contain the diagonal elements of a bidiagonal matrix. On return, each vector is overwritten with the singular values of the bidiagonal matrix, sorted with the algebraically largest value in the first element. If you are supplying **bidiag_svd_singular_values** with the values that the **gen_bidiag** routine returned in ***d***, see the Notes section, below.
e	Real CM array with the same precision, shape, and layout as ***d***, containing the superdiagonal elements of the bidiagonal matrix or matrices. The first element in each instance can have any value. The **bidiag_svd_singular_values** routine does not modify ***e***. If you are supplying **bidiag_svd_singular_values** with the values that the **gen_bidiag** routine returned in ***e***, see the Notes section, below.
axis	Scalar integer variable. The axis of ***d*** and ***e*** along which the diagonal and superdiagonal (respectively) elements of each matrix lie (the non-instance axis).

Copyright © 1994 Thinking Machines Corporation

tolerance Scalar real variable with the same precision as ***d*** and ***e***. Absolute error tolerance for the computed singular values. This is the same argument that occurs in the **sym_tridiag_eigenvalues** routine. If you supply a non-positive value, *tolerance* is changed internally to the default value $\varepsilon||B||$, where ε is the machine precision and $||B||$ is the 1-norm of the bidiagonal matrix. In the multiple-instance case, the default tolerance is the smallest tolerance over all the instances. For the accurate extraction of tiny singular values, you may need to set *tolerance* to a smaller value than the default. (For details, see Section 8.3.2). If the routine uses the default tolerance, then *tolerance* is set to the default value on return. Because *tolerance* may be returned with a new value, it must be a variable, not a constant expression.

ier Scalar integer variable. Set to 0 on successful completion, or to -9 to indicate an invalid data type; the type must be real (single- or double-precision).

DESCRIPTION

For each bidiagonal matrix B stored in ***d*** and ***e***, the **bidiag_svd_singular_values** routine computes the singular values given by the singular value decomposition

$$B = U\Sigma V^{\mathrm{T}}$$

U and V are orthogonal; Σ is diagonal and contains the singular values of B. The singular values are returned, in descending order, in ***d***.

NOTES

Supplying Values Returned by the Bidiagonalization Routine. If you supply **bidiag_svd_singular_values** with the values that the **gen_bidiag** routine returned in ***d*** and ***e***, and if the ***d*** and ***e*** returned by **gen_bidiag** had an extra axis of length 1 (axis *col_axis* in the **gen_bidiag** call), you may retain that extra axis; it becomes an extra instance axis in the **bidiag_svd_singular_values** call, and has no effect. The axis identified by *row_axis* in the **gen_bidiag** call becomes axis *axis* in the **bidiag_svd_singular_values** call.

The bidiagonal matrix whose entries **gen_bidiag** returns in ***d*** and ***e*** is in block cyclic form. However, the permutations that transform a matrix between block cyclic and

Copyright © 1994 Thinking Machines Corporation

consecutive elementwise order do not affect the singular values. For details about the block cyclic computations, see the man page for **gen_bidiag**.

Note that in a **bidiag_svd_singular_vectors** call, you may *not* retain the extra axis of length 1 in ***d*** and ***e***; see the man pages for **bidiag_svd_singular_vectors**.

Acknowledgments. The **bidiag_svd_singular_values** routine is the result of collaborative development between Thinking Machines Corporation and the Danish Computing Center for Research and Education (UNI−C) (contributor: Susanne Balle).

EXAMPLES

Sample CM Fortran code that uses the **bidiag_svd_singular_values** routine can be found on-line in the subdirectory

```
/svd/cmf
```

of a CMSSL examples directory whose location is site-specific.

Copyright © 1994 Thinking Machines Corporation

9.3 Singular Vectors of Bidiagonal Matrices

The **bidiag_svd_singular_vectors** routine computes the singular vectors corresponding to a given set of singular values of one or more real bidiagonal matrices. A detailed description of this routine is provided in the man page at the end of this section.

9.3.1 Implementation

Section 9.2.1 describes the relationship between a bidiagonal matrix B and a corresponding symmetric tridiagonal matrix M'. In the current release, **bidiag_svd_singular_vectors** finds the singular vectors of B by calling the **sym_tridiag_eigenvectors** routine to apply inverse iteration on M' for a given set of singular values. The **sym_tridiag_eigenvectors** routine is described in Chapter 8.

Copyright © 1994 Thinking Machines Corporation

Singular Vectors of Bidiagonal Matrices

The **bidiag_svd_singular_vectors** routine computes the singular vectors corresponding to a set of singular values of one or more real bidiagonal matrices of the same order. The diagonal and superdiagonal matrix elements and the singular values are stored in CM vectors or arrays. The singular vectors are stored in a multidimensional CM array.

SYNTAX

bidiag_svd_singular_vectors (***d, e,*** *axis,* ***f, U, V_tra,*** *singular_vector_axis, group, ier*)

ARGUMENTS

d	Real CM array consisting of one or more vectors lying along axis *axis*; all other axes are instance axes. On input, each vector must contain the diagonal elements of a bidiagonal matrix. This array is unchanged on return. If you are supplying **bidiag_svd_singular_vectors** with the values that the **gen_bidiag** routine returned in ***d***, see the Notes section, below.
e	Real CM array with the same precision, shape, and layout as ***d***, containing the superdiagonal elements of the bidiagonal matrix or matrices. The first element in each instance can have any value. This array is unchanged on return. If you are supplying **bidiag_svd_singular_vectors** with the values that the **gen_bidiag** routine returned in ***e***, see the Notes section, below.
axis	Scalar integer variable. The axis of ***d*** and ***e*** along which the diagonal and superdiagonal (respectively) elements of each matrix lie (the non-instance axis).

Copyright © 1994 Thinking Machines Corporation

f	Real CM array containing the singular values for which the singular vectors are sought. Must have the same rank and precision as ***d*** and ***e***. The instance axes must match those of ***d*** and ***e*** in extent and order of declaration. Within each vector lying along axis *axis*, the singular values belonging to the same spectrum must be sorted in decreasing order (with the algebraically largest singular value stored in the first array element). The extent of axis *axis* can be smaller in ***f*** than it is in ***d*** and ***e***, as described for the **sym_tridiag_eigenvectors** routine in Section 8.4.3.
U	Real CM array containing the left singular vectors on return. Must have the same precision as ***d***, ***e***, and ***f***, and rank one greater than that of ***d***, ***e***, and ***f***. You must specify the index of the extra dimension in the *singular_vector_axis* argument. The array section obtained by collapsing this extra dimension must be of the same shape as ***f***, as discussed for **sym_tridiag_eigenvectors** in Section 8.4.3. Thus, for each singular value supplied in ***f***, there is an associated vector lying along axis *singular_vector_axis* in ***U***. On return from **bidiag_svd_singular_vectors**, this vector contains the left singular vector corresponding to the singular value.
V_tra	Real CM array containing the right singular vectors on return. Must have the same precision as ***d***, ***e***, and ***f***, and the same rank and shape as ***U***. For each singular value supplied in ***f***, there is an associated vector lying along axis *singular_vector_axis* in ***V_tra***. On return from **bidiag_svd_singular_vectors**, this vector contains the right singular vector corresponding to the singular value.
singular_vector_axis	Scalar integer variable. The axis of ***U*** and ***V_tra*** along which the returned singular vectors lie. The extent of axis *singular_vector_axis* is the order of the bidiagonal matrices.

Copyright © 1994 Thinking Machines Corporation

group Scalar real variable with the same precision as ***d*** and ***e***. Grouping criterion. This is the same argument that occurs in the **sym_tridiag_eigenvectors** routine. Singular values that differ by less than $group||B||_\infty$ (where $||B||_\infty$ is the infinity norm of the bidiagonal matrix) are grouped together, and their corresponding singular vectors are orthogonalized. If you supply a non-positive value, *group* is changed internally to the default value 10^{-4}. (For details, see Section 8.4.2.) If the routine uses the default grouping criterion, then *group* is set to the default value on return. Because *group* may be returned with a new value, it must be a variable, not a constant expression.

ier Scalar integer variable. Set to 0 on successful completion. On error, contains one of the following codes:

1	The rank of ***f*** is not the same as the rank of ***d***.
2	The rank of ***U*** or ***V_tra*** is not the same as (rank of ***d***) + 1.
3	The array section corresponding to a fixed index of ***U*** or ***V_tra*** along dimension *singular_vector_axis* does not have the same shape as ***f***.
−9	Invalid data type; must be real (single- or double-precision).
1000+*i*	*i* singular vectors are not determined after 5 inverse iterations. The non-converged singular vectors are set to 0 on return.

DESCRIPTION

Given one or more real bidiagonal matrices represented by the CM arrays ***d*** and ***e***, and a CM array ***f*** containing a set of singular values for each matrix, the **bidiag_svd_singular_vectors** routine computes the left and right singular vectors corresponding to each singular value. That is, for each instance, **bidiag_svd_singular_vectors** returns in ***U*** and ***V_tra*** the requested portions of U and V^T, where

Version 3.2, April 1994
Copyright © 1994 Thinking Machines Corporation

$$B = U\Sigma V^{T}$$

is the singular value decomposition of the bidiagonal matrix B.

NOTES

Supplying Values Returned by the Bidiagonalization Routine. If you supply **bidiag_svd_singular_vectors** with the values that the **gen_bidiag** routine returned in ***d*** and ***e***, and if the ***d*** and ***e*** returned by **gen_bidiag** had an extra axis of length 1 (axis *col_axis* in the **gen_bidiag** call), you must remove this extra axis before supplying ***d*** and ***e*** to **bidiag_svd_singular_vectors**. The axis identified by *row_axis* in the **gen_bidiag** call becomes axis *axis* in the **bidiag_svd_singular_vectors** call.

The bidiagonal matrix whose entries **gen_bidiag** returns in ***d*** and ***e*** is in block cyclic form; therefore, the singular vectors that **bidiag_svd_singular_values** returns in ***U*** and ***V_tra*** are the singular vectors of the block cyclic bidiagonal matrix. For details about the block cyclic computations, including information about how to obtain the singular vectors of the original dense matrices A, see the man page for **gen_bidiag**.

Performance. For best performance, axis *singular_vector_axis* of ***U*** and ***V_tra*** should be local.

Acknowledgments. The **bidiag_svd_singular_vectors** routine is the result of collaborative development between Thinking Machines Corporation and the Danish Computing Center for Research and Education (UNI−C) (contributor: Susanne Balle).

EXAMPLES

Sample CM Fortran code that uses the **bidiag_svd_singular_vectors** routine can be found on-line in the subdirectory

```
/svd/cmf
```

of a CMSSL examples directory whose location is site-specific.

Copyright © 1994 Thinking Machines Corporation

9.4 Singular Value Decomposition of Dense Real Matrices

The **gen_bidiag_singular_system** routine combines the functionality of the following routines:

- **gen_bidiag**
- **bidiag_svd_singular_values**
- **bidiag_svd_singular_vectors**
- **bidiag_to_gen_left**
- **bidiag_to_gen_right**

Given a CM array A containing one or more real dense matrices A of dimensions m x n with $m \geq n$, **gen_bidiag_singular_system** computes the singular values and, if desired, the singular vectors of each A.

The **gen_bidiag_singular_system** routine offers a convenient packaging of the five routines listed above. On the other hand, the **bidiag_svd_singular_vectors** routine allows you the flexibility of computing only selected singular vectors of each matrix, whereas **gen_bidiag_singular_system** computes either all or none of the singular vectors.

For a detailed description of **gen_bidiag_singular_system**, see the man page that follows.

Version 3.2, April 1994
Copyright © 1994 Thinking Machines Corporation

Singular Value Decomposition of Dense Real Matrices

Given a CM array containing one or more real dense matrices of dimensions m x n with $m \geq n$, the **gen_bidiag_singular_system** routine computes the singular values and, if desired, the singular vectors of each matrix.

SYNTAX

gen_bidiag_singular_system (***d, U, V_tra, A***, *m, n, row_axis, col_axis, nblock, svects_flag, tolerance, group, ier*)

ARGUMENTS

d Real CM array with the same precision as ***A*** and rank one less than that of ***A***. The axis of ***d*** corresponding to axis *col_axis* of ***A*** must be omitted. Each matrix A in ***A*** corresponds to a vector in ***d***. The vectors lie along axis *row_axis* (if *row_axis* < *col_axis*) or axis *row_axis* − 1 (if *col_axis* < *row_axis*); this axis must have extent greater than or equal to n. The remaining axes of ***d*** must be instance axes that match the instance axes of ***A*** in extent and order of declaration.

Upon successful completion of **gen_bidiag_singular_system**, elements 1 through n of each vector in ***d*** contain the singular values of the corresponding matrix A in ***A***. The singular values are sorted with the algebraically largest value in the first element.

Copyright © 1994 Thinking Machines Corporation

U Real CM array with the same rank and precision as ***A***. Axes *row_axis* and *col_axis* define instances of two-dimensional arrays with dimensions $m \times n$ or $m \times m$. The remaining axes are instance axes that must match the instance axes of ***A*** in extent and order of declaration. Thus, for each matrix A within ***A*** there is a corresponding two-dimensional array within ***U***. If *svects_flag* is set to 1 on input, then on return, the left singular vectors of each A within ***A*** are placed in the corresponding array within ***U***. The vectors lie along axis *col_axis* and are sorted so that they occur in the same order as the corresponding singular values returned in ***d***.

V_tra Real CM array with the same rank and precision as ***A***. Axes *row_axis* and *col_axis* define instances of two-dimensional arrays with dimensions $n \times n$. The remaining axes are instance axes that must match the instance axes of ***A*** in extent and order of declaration. Thus, for each matrix A within ***A*** there is a corresponding two-dimensional array within ***V_tra***. If *svects_flag* is set to 1 on input, then on return, the right singular vectors of each A within ***A*** are placed in the corresponding array within ***V_tra***. The vectors lie along axis *col_axis* and are sorted so that they occur in the same order as the corresponding singular values returned in ***d***.

A Real CM array of rank greater than or equal to 2, containing one or more dense matrices A. Each A must have dimensions $m \times n$ where $m \geq n$. Axis *row_axis* counts the rows of the matrices A; axis *col_axis* counts the columns. If axes *row_axis* and *col_axis* have extents greater than m and n, then each instance of A consists of the upper left-hand $m \times n$ elements of a larger matrix within the array ***A***.

On return, ***A*** contains the same values it contains on return from **gen_bidiag**. These values are used internally by **gen_bidiag_singular_system**.

m Scalar integer variable. The number of rows in each matrix A within ***A***.

Version 3.2, April 1994
Copyright © 1994 Thinking Machines Corporation

n	Scalar integer variable. The number of columns in each matrix *A* within ***A***.
row_axis	Scalar integer variable. The axis of ***A*** that counts the rows of each matrix *A*. (For definitions of row axes and column axes, see Chapter 1.)
col_axis	Scalar integer variable. The axis of ***A*** that counts the columns of each matrix *A*. Also, the axis of ***U*** and ***V_tra*** along which the returned singular vectors lie. (For definitions of row axes and column axes, see Chapter 1.)
nblock	Scalar integer variable. Block size. In the current release, you must set *nblock* to 1.
svects_flag	Scalar integer variable. If you set *svects_flag* to 0, only the singular values are computed. If you set *svects_flag* to 1, both singular values and singular vectors are computed.
tolerance	Scalar real variable with the same precision as ***d***. Controls the absolute accuracy of the singular values after reduction to bidiagonal form. This is the same argument that occurs in the **bidiag_svd_singular_values** and **sym_tridiag_eigenvalues** routines. If you supply a non-positive value, *tolerance* is changed internally to the default value $\varepsilon\|\|B\|\|$, where ε is the machine precision and $\|\|B\|\|$ is the 1-norm of the bidiagonal matrix corresponding to *A*. In the multiple-instance case, the default tolerance is the smallest tolerance over all the instances. For the accurate extraction of tiny singular values, you may need to set *tolerance* to a smaller value than the default. (For details, see Section 8.3.2). If the routine uses the default tolerance, then *tolerance* is set to the default value on return. Because *tolerance* may be returned with a new value, it must be a variable, not a constant expression.

Copyright © 1994 Thinking Machines Corporation

group Scalar real variable with the same precision as ***d***. Grouping criterion for the singular values after reduction to bidiagonal form. This is the same argument that occurs in the **bidiag_svd_singular_vectors** and **sym_tridiag_eigenvectors** routines. Singular values that differ by less than $group||B||\infty$ (where $||B||\infty$ is the infinity norm of the bidiagonal matrix corresponding to A) are grouped together, and their corresponding singular vectors are orthogonalized. If you supply a non-positive value, *group* is changed internally to the default value 10^{-4}. (For details, see Section 8.4.2.) If the routine uses the default grouping criterion, then *group* is set to the default value on return. Because *group* may be returned with a new value, it must be a variable, not a constant expression.

ier Scalar integer variable. Set to 0 on successful completion. On error, contains one of the following codes:

1	The rank of ***d*** is not one less than the rank of ***A***.
2	The rank of ***U*** or ***V_tra*** is not the same as the rank of ***A***.
3	The axes of ***U*** or ***V_tra*** other than *row_axis* and *col_axis* do not match the instance axes of ***A*** in extent and order of declaration.
$1000+i$	i singular vectors are not determined after 5 inverse iterations. The non-converged singular vectors are set to 0 on return.
-1	***A*** is not a valid CM array.
-2	***A*** has an invalid rank (must be ≥ 2).
-3	The length of axis *row_axis* of ***A*** is less than m; it must be greater than or equal to m.

Copyright © 1994 Thinking Machines Corporation

−4 The length of axis *col_axis* of ***A*** is less than *n*; it must be greater than or equal to *n*.

−9 An array has an invalid data type; the data type must be real (single- or double-precision).

−20 *m* or *n* is invalid; both must be greater than 0 and *m* must be greater than or equal to *n*.

−30 *row_axis* or *col_axis* is invalid. 1 $\leq$ *row_axis, col_axis* $\leq$ rank (***A***) must be true, and *row_axis* and *col_axis* must not be equal.

−40 *nblock* is invalid. It must be 1 in the current release.

DESCRIPTION

Given a CM array ***A*** containing one or more real dense matrices *A* of dimensions $m \times n$ with $m \geq n$, the **gen_bidiag_singular_system** routine computes the singular values of each matrix *A*. If *svects_flag* is set to 1, then **gen_bidiag_singular_system** also computes the left and right singular vectors associated with each singular value. That is, it computes

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

where *U* is column-orthogonal and contains the left singular vectors of *A*, V^{T} is orthogonal and contains the right singular vectors of *A*, and Σ is a diagonal matrix containing the singular values of *A*.

NOTES

Performance. For best performance, axis *col_axis* of ***U*** and ***V_tra*** should be local.

Acknowledgments. The **gen_bidiag_singular_system** routine is the result of collaborative development between Thinking Machines Corporation and the Danish

Copyright © 1994 Thinking Machines Corporation

Computing Center for Research and Education (UNI−C) (contributor: Susanne Balle).

EXAMPLES

Sample CM Fortran code that uses the **gen_bidiag_singular_system** routine can be found on-line in the subdirectory

```
/svd/cmf
```

of a CMSSL examples directory whose location is site-specific.

Copyright © 1994 Thinking Machines Corporation

9.5 References

For more information about singular value decomposition, see the following references:

1. Golub, G. H., and C. F. Van Loan. *Matrix Computations*. 2d ed. Baltimore: Johns Hopkins University Press, 1989. Pp. 427−437.

2. Demmel, J. and W. Kahan. Accurate Singular Values of Bidiagonal Matrices. *SIAM J. Sci. Stat. Comput.* **11**, no. 5 (1990): 873−912.

3. Watkins, D. *Fundamentals of Matrix Computations*. New York: John Wiley and Sons, 1991. Pp. 390−404.

Copyright © 1994 Thinking Machines Corporation