3.3.12 \( A \) is \( 2 \times 2 \) and orthogonal. Find the most general form of
\[
A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}.
\]

Compare with two-dimensional rotation.

3.3.12 \( 1 = A A^T = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} a & c \\ b & d \end{pmatrix} \), so

\[
1 = a^2 + b^2 \quad \quad 1 = c^2 + d^2
\]

\[
0 = ca + db
\]

We may satisfy \( 1 = a^2 + b^2 \) and \( 1 = c^2 + d^2 \) by setting \( a = \cos \theta \), \( b = \sin \theta \), \( c = \sin \phi \), and \( d = \cos \phi \). Now the \( \perp \) condition is

\[
0 = \cos \theta \sin \phi + \sin \theta \cos \phi = \sin (\theta + \phi).
\]

So we set \( \phi = -\theta + m \pi \), which give

\[
A(\theta) = \begin{pmatrix} \cos \theta & \sin \theta \\ -\sin \theta & \cos \theta \end{pmatrix}
\]

for \( m = 0 \)

and \( A(-\theta) \) for \( m = 1 \).
3.4.7 \( A \) and \( B \) are two noncommuting Hermitian matrices:
\[
AB - BA = iC.
\]

Prove that \( C \) is Hermitian.

\[
3.4.7 \quad C = -i(AB - BA)
\]
\[
C^+ = i(B^+A^+ - A^+B^+)
\]
\[
= i(BA - AB) = -i(A^+B^+)
\]
\[
= -i(AB - BA)
\]
\[
= C,
\]
Two matrices $A$ and $B$ are each Hermitian. Find a necessary and sufficient condition for their product $AB$ to be Hermitian.

ANS. $[A, B] = 0.$

$$(AB)^* = B^* A^* = BA = AB$$

$[A, B] = 0.$
Two equal masses are connected to each other and to walls by springs as shown in Fig. 3.8. The masses are constrained to stay on a horizontal line.

(a) Set up the Newtonian acceleration equation for each mass.
(b) Solve the secular equation for the eigenvectors.
(c) Determine the eigenvectors and thus the normal modes of motion.

\[ K \begin{pmatrix} m & k \\ k & m \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} k(y-x) - Kx \\ -Ky + k(x-y) \end{pmatrix} \]

(a) \[ m \ddot{x} = k(y-x) - Kx \]
\[ m \ddot{y} = -Ky + k(x-y) \]

\[ \begin{pmatrix} x(t) \\ y(t) \end{pmatrix} = \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} e^{i\omega t} \]

(b) \[ \begin{pmatrix} -K-k & k \\ k & -K-k \end{pmatrix} \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} = \begin{pmatrix} -m\omega^2 x_0 \\ -m\omega^2 y_0 \end{pmatrix} \]

\[ 0 = \begin{vmatrix} m\omega^2 - K-k & k \\ k & m\omega^2 - K-k \end{vmatrix} = (m\omega^2 - K-k)^2 - k^2 \]

\[ m\omega^2 - K-k = \pm \sqrt{k} \]
\[ m\omega^2 = K+2k \]
\[ m\omega^2 = K \]

If \( m\omega^2 = K+2k \), then \[ -(K+k)x_0 + ky_0 = -(K+2k)x_0 \] or \( \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} = x_0 \begin{pmatrix} 1 \\ i \end{pmatrix} \).

If \( m\omega^2 = K \), then \[ -(K+k)x_0 + ky_0 = -Kx_0 \] or \( \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} = x_0 \begin{pmatrix} 1 \\ -1 \end{pmatrix} \).

(c) So the normal modes are
\[ \begin{pmatrix} x(t) \\ y(t) \end{pmatrix} = x_0 \begin{pmatrix} 1 \\ i \end{pmatrix} e^{\pm \sqrt{K/m} t} \quad \text{and} \quad \begin{pmatrix} x(t) \\ y(t) \end{pmatrix} = x_0 \begin{pmatrix} 1 \\ -1 \end{pmatrix} e^{\pm \sqrt{(K+2k)/m} t} \]
3.6.20 (c)(2) There are really four solutions here since the $w$'s can be $\pm$ the square roots. So one mode is

$$
\begin{pmatrix}
    x \\
    y
\end{pmatrix} = \begin{pmatrix}
    1 \\
    1
\end{pmatrix} \left[ a \cos t \sqrt{\frac{K}{m}} + b \sin t \sqrt{\frac{K}{m}} \right]
$$

and the other is

$$
\begin{pmatrix}
    x \\
    y
\end{pmatrix} = \begin{pmatrix}
    1 \\
    -1
\end{pmatrix} \left[ c \cos t \sqrt{(K+2k)/m} + d \sin t \sqrt{(K+2k)/m} \right].
$$
4.1.1 Show that an $n \times n$ orthogonal matrix has $n(n - 1)/2$ independent parameters. 

Hint. The orthogonality condition, Eq. (3.71), provides constraints.

$$1 = AA^T$$ implies the $n^2$ constraints

$$a_{ij} = a_{ik}a_{jk}$$

of which only the $n$ for $i = j$ and the $n(n-1)/2$ for $i \neq j$ are independent. So the number of constraints is

$$N_c = n + n \left( \frac{n-1}{2} \right) = n \left( 1 + \frac{n-1}{2} \right) = \frac{n(n+1)}{2}.$$ 

So the number of independent parameters is

$$n^2 - \frac{n(n+1)}{2} = \frac{n^2}{2} - n = \frac{n(n-1)}{2}.$$
4.1.3 The special linear group \( \text{SL}(2) \) consists of all \( 2 \times 2 \) matrices (with complex elements) having a determinant of \(+1\). Show that such matrices form a group.

*Note.* The \( \text{SL}(2) \) group can be related to the full Lorentz group in Section 4.4, much as the \( \text{SU}(2) \) group is related to \( \text{SO}(3) \).

4.1.3 If \( D \) and \( D' \) are in \( \text{SL}(2, \mathbb{C}) \), then

\[
\det(DD') = \det D \det D' = 1 \cdot 1 = 1,
\]

so multiplication is closed.

\[
\det D^{-1} = 1/\det D = 1/1 = 1.
\]

Each matrix has an inverse in \( \text{SL}(2, \mathbb{C}) \).

\[
\det I = 1
\]

The identity matrix is in \( \text{SL}(2, \mathbb{C}) \).

Matrix multiplication is associative.
4.3.1 a) \([J_4, J^2] = [J_4, J_1^2, J^2]\)

\[= [J_1, J^2] \mathbf{1} + [J_2, J^2] 0 + 0 = 0.\]

b) \([J_-, J^2] = [J_-, J^2] - i[J_-, J^2] = 0.\)
4.3.2 First $SU(2)$, $H_1 = \frac{\sigma_3}{2}$, $E_{\pm 1} = \frac{\sigma_1 \pm i \sigma_2}{2}$

\[
[H_1, E_{\pm 1}] = \left[ \frac{\sigma_3}{2}, \frac{\sigma_1 \pm i \sigma_2}{2} \right] = \left( \epsilon_{312} \frac{\sigma_2}{2} + \epsilon_{321} \frac{\sigma_1}{2} \right)
\]

\[
= \pm \frac{\sigma_1}{2} \pm i \frac{\sigma_2}{2} = \pm \left( \frac{\sigma_1 \pm i \sigma_2}{2} \right)
\]

So $\alpha_\pm = \pm 1$.

Now $SU(3)$: $H_1 = \frac{\lambda_3}{2}$, $H_2 = \frac{\lambda_8}{2}$, Let's first find the ev's and weights of the defining rep.

\[
\frac{\lambda_3}{2} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \frac{1}{2} \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad \frac{\lambda_8}{2} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \frac{\sqrt{3}}{6} \begin{pmatrix} 1 \\ 0 \end{pmatrix}
\]

So $m_1 = (\frac{1}{2}, \frac{\sqrt{3}}{6})$. Next

\[
\frac{\lambda_3}{2} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = -\frac{1}{2} \begin{pmatrix} 0 \\ 1 \end{pmatrix} \quad \frac{\lambda_8}{2} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \frac{\sqrt{3}}{6} \begin{pmatrix} 0 \\ 1 \end{pmatrix}
\]

So $m_2 = (-\frac{1}{2}, \frac{\sqrt{3}}{6})$.

\[
\frac{\lambda_3}{2} \begin{pmatrix} 0 \\ 0 \end{pmatrix} = 0 \quad \frac{\lambda_8}{2} \begin{pmatrix} 0 \\ 0 \end{pmatrix} = -\frac{\sqrt{3}}{3} \begin{pmatrix} 0 \\ 1 \end{pmatrix}
\]

So $m_3 = (0, -\sqrt{3}/3)$. 
4.3.2 (2) Now for any rep. \( D \)

\[ H_i E_{\pm \alpha} |m, D \rangle = [H_i, E_{\pm \alpha}] |m, D \rangle + E_{\pm \alpha} H_i |m, D \rangle \]

\[ = \{ m_i, E_{\pm \alpha} |m, D \rangle \} + m_i E_{\pm \alpha} |m, D \rangle \]

\[ = (m_i \pm \alpha i) E_{\pm \alpha} |m, D \rangle. \]

So the root vectors point from one \( \vec{m} \) to another, e.g., \( \vec{m}' = \vec{m} - \alpha \vec{n} \rightarrow \vec{m}' = \vec{m} - \alpha \vec{n} \).

So,

\[ \alpha_{12} = m_1 - m_2 = (\frac{1}{2}, \frac{\sqrt{3}}{6}) - (-\frac{1}{2}, \frac{\sqrt{3}}{6}) \]

\[ = (1, 0) \]

\[ \alpha_{21} = -\alpha_{12}. \]

\[ \alpha_{13} = m_1 - m_3 = (\frac{1}{2}, \frac{\sqrt{3}}{6}) - (0, -\frac{\sqrt{3}}{3}) = (\frac{1}{2}, \frac{\sqrt{3}}{6}) = (\frac{1}{2}, \frac{\sqrt{3}}{2}) \]

\[ \alpha_{31} = -\alpha_{13}. \]

\[ \alpha_{23} = m_2 - m_3 = (-\frac{1}{2}, \frac{\sqrt{3}}{6}) - (0, -\frac{\sqrt{3}}{3}) = (-\frac{1}{2}, \frac{\sqrt{3}}{2}) \]

\[ \alpha_{32} = -\alpha_{23}. \]

Also, \( \alpha_1 = (0, 0) \) and \( \alpha_2 = (0, 0) \)

are the root vectors of \( H_1 \) & \( H_2 \). So

\[ (-\frac{1}{2}, \frac{\sqrt{3}}{2}) = \alpha_{23} \]

\[ (1, 0) = \alpha_{12} \]

\[ (\frac{1}{2}, \frac{\sqrt{3}}{2}) = \alpha_{13} \]

\[ (-\frac{1}{2}, -\frac{\sqrt{3}}{2}) = \alpha_{31} \]

\[ (\frac{1}{2}, -\frac{\sqrt{3}}{2}) = \alpha_{32} \]
Special problem one!

1. Find the multiplication table for a group with three elements and prove that it is unique.

Call the three elements $e$, $a$, and $b$, with $e$ the identity. Then $ea = a$; $eb = b$; and $ee = e$. We must figure out what $a^2$, $b^2$, $ab$, and $ba$ are.

If $ab = a$, then $a' (ab) = a' a = e = (a' a) b = eb = b$ so that we'd have $b = e$. Then the group would have two elements, not three. So $ab \neq a$.

Similarly, $ab = b \Rightarrow a = e$. So $ab \neq b$.

Thus $ab = e$.

Similarly, $ba = e$. So $b = a'$.

If $a^2 = a$, then $a = e$; so $a^{-1} = a$.

If $a^2 = e$, then $a = a^{-1} = b$. So $a^2 = e$.

So $a^2 = b$.

If $b^2 = e$, then $abb = a \Rightarrow b = a$.

If $b^2 = b$, then $b = e$; so $b^2 \neq b$.

So $b^2 = a$.

So the multiplication table is:

<table>
<thead>
<tr>
<th></th>
<th>e</th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>e</td>
<td>e</td>
<td>a</td>
<td>b</td>
</tr>
<tr>
<td>a</td>
<td>a</td>
<td>b</td>
<td>e</td>
</tr>
<tr>
<td>b</td>
<td>b</td>
<td>e</td>
<td>a</td>
</tr>
</tbody>
</table>
2. Consider the $2 \times 3$ matrix $A$ with elements

$$
(1 \ 2 \ 3) \\
(-3 \ 0 \ 1),
$$

$(A(1,1) = A(2,3) = 1$, etc.$$)

Perform the singular value decomposition

$$
A = U S V^T
$$

where $V^T$ is the transpose of $V$. Find the singular values and the real matrices $U$ and $V$.

We use the LAPACK driver dgesvd: LAPACK version 3.0

dgesvd(3)

PURPOSE

DGESVD - compute the singular value decomposition (SVD) of a real M-by-N
matrix $A$, optionally computing the left and/or right singular vectors

SYNTAX

SUBROUTINE DGESVD( JOBU, JOBVT, M, N, A, LDA, S, U, LDU, VT, LDVT, WORK,
                  LWORK, INFO )

CHARACTER JOBU, JOBVT

INTEGER INFO, LDA, LDU, LDVT, LWORK, M, N

DOUBLE PRECISION A( LDA, * ), S( * ), U( LDU, * ), VT( LDVT, * ), WORK( * )

DESCRIPTION

DGESVD computes the singular value decomposition (SVD) of a real M-by-N
matrix $A$, optionally computing the left and/or right singular vectors. The
SVD is written

$$
A = U \cdot \text{SIGMA} \cdot \text{transpose}(V)
$$

where SIGMA is an M-by-N matrix which is zero except for its $\min(m, n)$
diagonal elements, $U$ is an M-by-M orthogonal matrix, and $V$ is an N-by-N
orthogonal matrix. The diagonal elements of SIGMA are the singular values
of $A$; they are real and non-negative, and are returned in descending order.
The first $\min(m, n)$ columns of $U$ and $V$ are the left and right singular
vectors of $A$.

Note that the routine returns $V^{**T}$, not $V$.

ARGUMENTS

JOBU  (input) CHARACTER*1
Specifies options for computing all or part of the matrix $U$:

- 'A': all $M$ columns of $U$ are returned in array $U$;
- 'S': the first $\min(m, n)$ columns of $U$ (the left singular vectors)
  are returned in the array $U$; = 'O': the first $\min(m, n)$ columns of
  $U$ (the left singular vectors) are overwritten on the array $A$; =
  'N': no columns of $U$ (no left singular vectors) are computed.

JOBVT (input) CHARACTER*1
Specifies options for computing all or part of the matrix $V^{**T}$:

- 'A': all $N$ rows of $V^{**T}$ are returned in the array $VT$;
- 'S': the first $\min(m, n)$ rows of $V^{**T}$ (the right singular

vectors) are returned in the array VT; = 'O': the first min(m,n)
rows of V**T (the right singular vectors) are overwritten on the
array A; = 'N': no rows of V**T (no right singular vectors) are
computed.

JOBVT and JOBU cannot both be 'O'.

M
(input) INTEGER
The number of rows of the input matrix A. M >= 0.

N
(input) INTEGER
The number of columns of the input matrix A. N >= 0.

A
(input/output) DOUBLE PRECISION array, dimension (LDA,N)
On entry, the M-by-N matrix A. On exit, if JOBU = 'O', A is
overwritten with the first min(m,n) columns of U (the left singular
vectors, stored columnwise); if JOBVT = 'O', A is overwritten with
the first min(m,n) rows of V**T (the right singular vectors, stored
rowwise); if JOBU .ne. 'O' and JOBVT .ne. 'O', the contents of A
are destroyed.

LDA
(input) INTEGER
The leading dimension of the array A. LDA >= max(1,M).

S
(output) DOUBLE PRECISION array, dimension (min(M,N))
The singular values of A, sorted so that S(i) >= S(i+1).

U
(output) DOUBLE PRECISION array, dimension (LDU,UCOL)
(LDU,M) if JOBU = 'A' or (LDU,min(M,N)) if JOBU = 'S'. If JOBU =
'A', U contains the M-by-N orthogonal matrix U; if JOBU = 'S', U
contains the first min(m,n) columns of U (the left singular
vectors, stored columnwise); if JOBU = 'N' or 'O', U is not
referenced.

LDU
(input) INTEGER
The leading dimension of the array U. LDU >= 1; if JOBU = 'S' or
'A', LDU >= M.

VT
(output) DOUBLE PRECISION array, dimension (LDVT,N)
If JOBVT = 'A', VT contains the N-by-N orthogonal matrix V**T; if
JOBVT = 'S', VT contains the first min(m,n) rows of V**T (the right
singular vectors, stored rowwise); if JOBVT = 'N' or 'O', VT is not
referenced.

LDVT
(input) INTEGER
The leading dimension of the array VT. LDVT >= 1; if JOBVT = 'A',
LDVT >= N; if JOBVT = 'S', LDVT >= min(M,N).

WORK
workspace/output) DOUBLE PRECISION array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK; if INFO >
0, WORK(2:MIN(M,N)) contains the unconverted superdiagonal elements
of an upper bidiagonal matrix B whose diagonal is in S (not
necessarily sorted). B satisfies A = U * B * VT so it has the same
singular values as A, and singular vectors related by U and VT.

LWORK
(input) INTEGER
The dimension of the array WORK. LWORK >= 1. LWORK >=
MAX(3*MIN(M,N)+MAX(M,N),5*MIN(M,N)). For good performance, LWORK
should generally be larger.

If LWORK = -1, then a workspace query is assumed; the routine only
calculates the optimal size of the WORK array, returns this value as
the first entry of the WORK array, and no error message related
to LWORK is issued by XERBLA.

INFO
(output) INTEGER
= 0: successful exit.
< 0: if INFO = -i, the i-th argument had an illegal value.
> 0: if DBDSQR did not converge, INFO specifies how many
superdiagonals of an intermediate bidiagonal form B did not
converge to zero. See the description of WORK above for details.

CXML Home Page
module defs
  integer, parameter :: i4b = selected_int_kind(9)
  integer, parameter :: i2b = selected_int_kind(4)
  integer, parameter :: ip = kind(1)
  integer, parameter :: sp = kind(1.0)
  integer, parameter :: dp = kind(1.0d0)
end module defs
program svd
    use defs, implicit none
    character(len=1), parameter :: jobu='a', jobvt='a'
    integer(4b):: info, i
    integer(4b), parameter :: m=2, n=3, lda=m, ldu=m, ldvt=n, &
    lwork=100
    real(dp), dimension(lda):: S
    real(dp), dimension(lwork):: work
    real(dp), dimension(lda, n):: A
    real(dp), dimension(m, m):: U
    real(dp), dimension(n, n):: VT
    do i = 1, 3
        A(1, i) = real(i, dp)
    end do
    A(2, 1) = -3.0; A(2, 2) = 0.0; A(2, 3) = 1.0
    call dgesvd(jobu, jobvt, m, n, A, lda, S, U, ldu, VT, ldvt, &
    work, lwork, info)
    open(7, file='problem-2')
    write(7,*) 'The', lda, ' singular values are:
    do i = 1, lda
        write(7,*) 'S(', i, ',') = ', S(i)
    end do
    write(7,*) 'The', lda, ' left singular vectors are:
    write(7,*)
    do i = 1, lda
        write(7,*) 'L(', i, ',') = ', U(:, i)
    end do
    write(7,*) 'The', lda, ' right singular vectors are:
    do i = 1, lda
        write(7,*) 'R(', i, ',') = ', VT(i, :)
    end do
end program svd
The 2 singular values are:
S( 1 ) = 3.741657386773942
S( 2 ) = 3.162277660168379
The 2 left singular vectors are:
  L( 1 ) = 1.00000000000000E+00  0.00000000000000E+00
  L( 2 ) = 0.00000000000000E+00  1.00000000000000E+00
The 2 right singular vectors are:
  R( 1 ) = 0.2672612419124243  0.5345224838248487  0.8017837257372731
  R( 2 ) = -0.9486832980505137  0.00000000000000E+00  0.3162277660168381
3. Consider the 6 x 9 matrix A with elements
   \[ A(j,k) = x + x^{**}j + i*(y - y^{**}k) \]
   where \( x = 1.1 \) and \( y = 1.02 \) and * means multiplication while ** means exponentiation.
   Find the singular values, and the first left and right singular vectors. **Lapack** is one way
   to do this problem. This assignment is due on the Tuesday after Thanksgiving.

   Since A is complex, we use
   \[ \text{zgesvd} \]
   instead of \[ \text{gesvd} \].
ZGESVD - compute the singular value decomposition (SVD) of a complex M-by-N matrix A, optionally computing the left and/or right singular vectors.

SUBROUTINE ZGESVD( JOBU, JOBV, M, N, A, LDA, S, U, LDU, VT, LDVT, WORK, LWORK, RWORK, INFO )

CHARACTER JOBV, JOBU
INTEGER INFO, LDA, LDU, LDVT, LWORK, M, N
DOUBLE PRECISION RWORK(*)
COMPLEX*16 A(LDA,*), U(LDU,*), VT(LDVT,*), WORK(*)

ZGESVD computes the singular value decomposition (SVD) of a complex M-by-N matrix A, optionally computing the left and/or right singular vectors. The SVD is written

\[ A = U \times \Sigma \times V^H \]

where \( \Sigma \) is an M-by-N matrix which is zero except for its \( \min(m,n) \) diagonal elements, U is an M-by-M unitary matrix, and V is an N-by-N unitary matrix. The diagonal elements of \( \Sigma \) are the singular values of A; they are real and non-negative, and are returned in descending order. The first \( \min(m,n) \) columns of U and V are the left and right singular vectors of A.

Note that the routine returns \( V^H \), not V.

ARGUMENTS

JOBU (input) CHARACTER*1
Specifies options for computing all or part of the matrix U:
- 'A': all M columns of U are returned in array U;
- 'S': the first \( \min(m,n) \) columns of U (the left singular vectors) are returned in the array U;
- 'O': the first \( \min(m,n) \) columns of U (the left singular vectors) are overwritten on the array U; 'N': no columns of U (no left singular vectors) are computed.

JOBV (input) CHARACTER*1
Specifies options for computing all or part of the matrix V^H:
- 'A': all N rows of V^H are returned in the array VT;
- 'S': the first \( \min(m,n) \) rows of V^H (the right singular vectors) are returned in the array VT;
- 'O': the first \( \min(m,n) \) rows of V^H (the right singular vectors) are overwritten on the array A;
- 'N': no rows of V^H (no right singular vectors) are computed.

JOBU and JOBV cannot both be 'O'.

M (input) INTEGER
The number of rows of the input matrix A. \( M \geq 0 \).

N (input) INTEGER
The number of columns of the input matrix A. \( N \geq 0 \).

A (input/output) COMPLEX*16 array, dimension (LDA,N)
On entry, the M-by-N matrix A. On exit, if JOBU = 'O', A is overwritten with the first \( \min(m,n) \) columns of U (the left singular vectors, stored columnwise); if JOBV = 'O', A is overwritten with the first \( \min(m,n) \) rows of V^H (the right singular vectors, stored rowwise); if JOBU .ne. 'O' and JOBV .ne. 'O', the contents of A are destroyed.

LDA (input) INTEGER
The leading dimension of the array A. LDA >= max(1,M).

S (output) DOUBLE PRECISION array, dimension (\( \min(M,N) \))
The singular values of A, sorted so that \( S(i) \geq S(i+1) \).
U (output) COMPLEX*16 array, dimension (LDU, UCOL)
(LDU,M) if JOB = 'A' or (LDU,min(M,N)) if JOB = 'S'. If JOB = 'A', U contains the M-by-M unitary matrix U; if JOB = 'S', U contains the first min(m,n) columns of U (the left singular vectors, stored columnwise); if JOB = 'N' or 'O', U is not referenced.

LDU (input) INTEGER
The leading dimension of the array U. LDU >= 1; if JOB = 'S' or 'A', LDU >= M.

VT (output) COMPLEX*16 array, dimension (LDVT, N)
If JOBV = 'A', VT contains the N-by-N unitary matrix V**H; if JOBV = 'S', VT contains the first min(m,n) rows of V**H (the right singular vectors, stored rowwise); if JOBV = 'N' or 'O', VT is not referenced.

LDVT (input) INTEGER
The leading dimension of the array VT. LDVT >= 1; if JOBV = 'A', LDVT >= N; if JOBV = 'S', LDVT >= min(M, N).

WORK (workspace/output) COMPLEX*16 array, dimension (LWORK)
On exit, if INFO = 0, WORK(1) returns the optimal LWORK.

LWORK (input) INTEGER
The dimension of the array WORK. LWORK >= 1. LWORK >= 2*MIN(M,N)+MAX(M,N). For good performance, LWORK should generally be larger.

RWORK (workspace) DOUBLE PRECISION array, dimension (5*MAX(M,N))
On exit, if INFO > 0, RWORK(1:MIN(M,N)-1) contains the unconverged superdiagonal elements of an upper bidiagonal matrix B whose diagonal is in S (not necessarily sorted). B satisfies A = U * B * VT, so it has the same singular values as A, and singular vectors related by U and VT.

INFO (output) INTEGER
= 0: successful exit.
< 0: if INFO = -i, the i-th argument had an illegal value.
> 0: if ZGESQR did not converge, INFO specifies how many superdiagonals of an intermediate bidiagonal form B did not converge to zero. See the description of RWORK above for details.

Back to the listing of simple driver routines
program svd3
  usedefs; implicit none
  character(len=1), parameter :: jobu='a', jobvt='a'
  integer(4b), :: info, i, j
  integer(4b), parameter :: m=6, n=9, lda=m, ldu=m, ldvt=n, 
                          lwork=200
  real(dp), dimension(lda):: S
  real(dp), dimension(m*n) :: rwork
  complex(dp), dimension(lwork):: work
  complex(dp), dimension(lda,n) :: A
  complex(dp), dimension(m,m) :: U
  complex(dp), dimension(n,n) :: VT
  real(dp), parameter :: x=1.1, y=1.02
  do j = 1, n; do i = 1, m
    A(i,j) = cmplx(x + x**i, y - y**x)
  end do; end do
  call zgesvd(jobu, jobvt, m, n, A, lda, S, U, ldu, VT, ldvt, 
              work, lwork, rwork, info)
  open(7,file='problem-3')
  write(7,*),'The singular values are:
  do i = 1, lda
    write(7,*)' S(',i,') = ',S(i)
  end do
  write(7,*)
  write(7,*)'The first left singular vector is:'
  write(7,*)'U(',i,') = ',U(:,1)
  write(7,*)
  write(7,*)'The first right singular vector is:'
  write(7,*)'V(',i,') = ',conjg(VT(1,:))
end program svd3
The 6 singular values are:
S( 1 ) = 18.56999776537926
S( 2 ) = 3.771581343789893E-02
S( 3 ) = 2.146946191637828E-15
S( 4 ) = 4.725174820839520E-16
S( 5 ) = 3.158251745532310E-16
S( 6 ) = 1.959748226820287E-16

The first left singular vector is:
L( 1 ) = (-0.3557955535806104,2.025284796465866E-23 (-0.3735483605589319,-6.900867214146602E-04 (-0.3930763943657091,-1.449180020961566E-03 (-0.4145573085094160,-2.284185641904907E-03 (-0.4381862640459296,-3.202689880505448E-03 (-0.4641781228317201,-4.213044842110218E-03

The first right singular vector is:
R( 1 ) = (-0.3330537031449926,-1.709902107912780E-03 (-0.3330406972638836,-4.399629339108539E-03 (-0.3330274312664829,-7.122950840891350E-03 (-0.3330138999467118,-9.910939271848393E-03 (-0.3330000980034902,-1.2754668686462866E-02 (-0.3329860200217367,-1.56553934075532E-02 (-0.3329716604794932,-1.861394444257107E-02 (-0.3329570137479247,-2.163175193323814E-02 (-0.3329420740794449,-2.470991604349603E-02)