SLATEC Common Mathematical Library

Version 4.1

Table of Contents

This table of contents of the SLATEC Common Mathematical Library (CML) has three sections.

Section I contains the names and purposes of all user-callable CML routines, arranged by GAMS category. Those unfamiliar with the GAMS scheme should consult the document "Guide to the SLATEC Common Mathematical Library". The current library has routines in the following GAMS major categories:

- A. Arithmetic, error analysis
- C. Elementary and special functions (search also class L5)
- D. Linear Algebra
- E. Interpolation
- F. Solution of nonlinear equations
- G. Optimization (search also classes K, L8)
- H. Differentiation, integration
- I. Differential and integral equations
- J. Integral transforms
- K. Approximation (search also class L8)
- L. Statistics, probability
- N. Data handling (search also class L2)
- R. Service routines
- Z. Other

The library contains routines which operate on different types of data but which are otherwise equivalent. The names of equivalent routines are listed vertically before the purpose. Immediately after each name is a hyphen (-) and one of the alphabetic characters S, D, C, I, H, L, or A, where S indicates a single precision routine, D double precision, C complex, I integer, H character, L logical, and A is a pseudo-type given to routines that could not reasonably be converted to some other type.

Section II contains the names and purposes of all subsidiary CML routines, arranged in alphabetical order. Usually these routines are not referenced directly by library users. They are listed here so that users will be able to avoid duplicating names that are used by the CML and for the benefit of programmers who may be able to use them in the construction of new routines for the library.

Section III is an alphabetical list of every routine in the CML and the categories to which the routine is assigned. Every user-callable routine has at least one category. An asterisk (*) immediately preceding a routine name indicates a subsidiary routine.

## SECTION I. User-callable Routines

### A. Arithmetic, error analysis
A3. Real
A3D. Extended range

```
        DXADD-D   with an extended exponent range.

        XADJ-S    To provide single-precision floating-point arithmetic
        DXADJ-D   with an extended exponent range.

        XC210-S   To provide single-precision floating-point arithmetic
        DXC210-D  with an extended exponent range.

        XCON-S    To provide single-precision floating-point arithmetic
        DXCON-D   with an extended exponent range.

        XRED-S    To provide single-precision floating-point arithmetic
        DXRED-D   with an extended exponent range.

        XSET-S    To provide single-precision floating-point arithmetic
        DXSET-D   with an extended exponent range.

A4.  Complex
A4A.  Single precision

        CARG-C    Compute the argument of a complex number.

A6.  Change of representation
A6B.  Base conversion

        R9PAK-S   Pack a base 2 exponent into a floating point number.
        D9PAK-D

        R9UPAK-S  Unpack a floating point number X so that X = Y*2**N.
        D9UPAK-D

C.  Elementary and special functions (search also class L5)

        FUNDOC-A  Documentation for FNLIB, a collection of routines for
                  evaluating elementary and special functions.

C1.  Integer-valued functions (e.g., floor, ceiling, factorial, binomial
     coefficient)

        BINOM-S   Compute the binomial coefficients.
        DBINOM-D

        FAC-S     Compute the factorial function.
        DFAC-D

        POCH-S    Evaluate a generalization of Pochhammer's symbol.
        DPOCH-D

        POCH1-S   Calculate a generalization of Pochhammer's symbol starting
        DPOCH1-D  from first order.

C2.  Powers, roots, reciprocals

        CBRT-S    Compute the cube root.
        DCBRT-D
        CCBRT-C

C3.  Polynomials
C3A.  Orthogonal
C3A2.  Chebyshev, Legendre
```

| | |
|---|---|
| CSEVL-S<br>DCSEVL-D | Evaluate a Chebyshev series. |
| INITS-S<br>INITDS-D | Determine the number of terms needed in an orthogonal polynomial series so that it meets a specified accuracy. |
| QMOMO-S<br>DQMOMO-D | This routine computes modified Chebyshev moments. The K-th modified Chebyshev moment is defined as the integral over (-1,1) of W(X)*T(K,X), where T(K,X) is the Chebyshev polynomial of degree K. |
| XLEGF-S<br>DXLEGF-D | Compute normalized Legendre polynomials and associated Legendre functions. |
| XNRMP-S<br>DXNRMP-D | Compute normalized Legendre polynomials. |

C4. Elementary transcendental functions
C4A. Trigonometric, inverse trigonometric

| | |
|---|---|
| CACOS-C | Compute the complex arc cosine. |
| CASIN-C | Compute the complex arc sine. |
| CATAN-C | Compute the complex arc tangent. |
| CATAN2-C | Compute the complex arc tangent in the proper quadrant. |
| COSDG-S<br>DCOSDG-D | Compute the cosine of an argument in degrees. |
| COT-S<br>DCOT-D<br>CCOT-C | Compute the cotangent. |
| CTAN-C | Compute the complex tangent. |
| SINDG-S<br>DSINDG-D | Compute the sine of an argument in degrees. |

C4B. Exponential, logarithmic

| | |
|---|---|
| ALNREL-S<br>DLNREL-D<br>CLNREL-C | Evaluate ln(1+X) accurate in the sense of relative error. |
| CLOG10-C | Compute the principal value of the complex base 10 logarithm. |
| EXPREL-S<br>DEXPRL-D<br>CEXPRL-C | Calculate the relative error exponential (EXP(X)-1)/X. |

C4C. Hyperbolic, inverse hyperbolic

| | |
|---|---|
| ACOSH-S<br>DACOSH-D<br>CACOSH-C | Compute the arc hyperbolic cosine. |
| ASINH-S<br>DASINH-D | Compute the arc hyperbolic sine. |

```
        CASINH-C

        ATANH-S   Compute the arc hyperbolic tangent.
        DATANH-D
        CATANH-C

        CCOSH-C   Compute the complex hyperbolic cosine.

        CSINH-C   Compute the complex hyperbolic sine.

        CTANH-C   Compute the complex hyperbolic tangent.

C5.  Exponential and logarithmic integrals

        ALI-S     Compute the logarithmic integral.
        DLI-D

        E1-S      Compute the exponential integral E1(X).
        DE1-D

        EI-S      Compute the exponential integral Ei(X).
        DEI-D

        EXINT-S   Compute an M member sequence of exponential integrals
        DEXINT-D  E(N+K,X), K=0,1,...,M-1 for N .GE. 1 and X .GE. 0.

        SPENC-S   Compute a form of Spence's integral due to K. Mitchell.
        DSPENC-D

C7.  Gamma
C7A.  Gamma, log gamma, reciprocal gamma

        ALGAMS-S  Compute the logarithm of the absolute value of the Gamma
        DLGAMS-D  function.

        ALNGAM-S  Compute the logarithm of the absolute value of the Gamma
        DLNGAM-D  function.
        CLNGAM-C

        C0LGMC-C  Evaluate (Z+0.5)*LOG((Z+1.)/Z) - 1.0 with relative
                  accuracy.

        GAMLIM-S  Compute the minimum and maximum bounds for the argument in
        DGAMLM-D  the Gamma function.

        GAMMA-S   Compute the complete Gamma function.
        DGAMMA-D
        CGAMMA-C

        GAMR-S    Compute the reciprocal of the Gamma function.
        DGAMR-D
        CGAMR-C

        POCH-S    Evaluate a generalization of Pochhammer's symbol.
        DPOCH-D

        POCH1-S   Calculate a generalization of Pochhammer's symbol starting
        DPOCH1-D  from first order.

C7B.  Beta, log beta
```

```
          ALBETA-S  Compute the natural logarithm of the complete Beta
          DLBETA-D  function.
          CLBETA-C

          BETA-S    Compute the complete Beta function.
          DBETA-D
          CBETA-C

C7C.  Psi function

          PSI-S     Compute the Psi (or Digamma) function.
          DPSI-D
          CPSI-C

          PSIFN-S   Compute derivatives of the Psi function.
          DPSIFN-D

C7E.  Incomplete gamma

          GAMI-S    Evaluate the incomplete Gamma function.
          DGAMI-D

          GAMIC-S   Calculate the complementary incomplete Gamma function.
          DGAMIC-D

          GAMIT-S   Calculate Tricomi's form of the incomplete Gamma function.
          DGAMIT-D

C7F.  Incomplete beta

          BETAI-S   Calculate the incomplete Beta function.
          DBETAI-D

C8.  Error functions
C8A.  Error functions, their inverses, integrals, including the normal
      distribution function

          ERF-S     Compute the error function.
          DERF-D

          ERFC-S    Compute the complementary error function.
          DERFC-D

C8C.  Dawson's integral

          DAWS-S    Compute Dawson's function.
          DDAWS-D

C9.  Legendre functions

          XLEGF-S   Compute normalized Legendre polynomials and associated
          DXLEGF-D  Legendre functions.

          XNRMP-S   Compute normalized Legendre polynomials.
          DXNRMP-D

C10.  Bessel functions
C10A.  J, Y, H-(1), H-(2)
C10A1.  Real argument, integer order

          BESJ0-S   Compute the Bessel function of the first kind of order
```

```
           DBESJ0-D  zero.

           BESJ1-S   Compute the Bessel function of the first kind of order one.
           DBESJ1-D

           BESY0-S   Compute the Bessel function of the second kind of order
           DBESY0-D  zero.

           BESY1-S   Compute the Bessel function of the second kind of order
           DBESY1-D  one.

C10A3.  Real argument, real order

           BESJ-S    Compute an N member sequence of J Bessel functions
           DBESJ-D   J/SUB(ALPHA+K-1)/(X), K=1,...,N for non-negative ALPHA
                     and X.

           BESY-S    Implement forward recursion on the three term recursion
           DBESY-D   relation for a sequence of non-negative order Bessel
                     functions Y/SUB(FNU+I-1)/(X), I=1,...,N for real, positive
                     X and non-negative orders FNU.

C10A4.  Complex argument, real order

           CBESH-C   Compute a sequence of the Hankel functions H(m,a,z)
           ZBESH-C   for superscript m=1 or 2, real nonnegative orders a=b,
                     b+1,... where b>0, and nonzero complex argument z.  A
                     scaling option is available to help avoid overflow.

           CBESJ-C   Compute a sequence of the Bessel functions J(a,z) for
           ZBESJ-C   complex argument z and real nonnegative orders a=b,b+1,
                     b+2,... where b>0.  A scaling option is available to
                     help avoid overflow.

           CBESY-C   Compute a sequence of the Bessel functions Y(a,z) for
           ZBESY-C   complex argument z and real nonnegative orders a=b,b+1,
                     b+2,... where b>0.  A scaling option is available to
                     help avoid overflow.

C10B.  I, K
C10B1.  Real argument, integer order

           BESI0-S   Compute the hyperbolic Bessel function of the first kind
           DBESI0-D  of order zero.

           BESI0E-S  Compute the exponentially scaled modified (hyperbolic)
           DBSI0E-D  Bessel function of the first kind of order zero.

           BESI1-S   Compute the modified (hyperbolic) Bessel function of the
           DBESI1-D  first kind of order one.

           BESI1E-S  Compute the exponentially scaled modified (hyperbolic)
           DBSI1E-D  Bessel function of the first kind of order one.

           BESK0-S   Compute the modified (hyperbolic) Bessel function of the
           DBESK0-D  third kind of order zero.

           BESK0E-S  Compute the exponentially scaled modified (hyperbolic)
           DBSK0E-D  Bessel function of the third kind of order zero.

           BESK1-S   Compute the modified (hyperbolic) Bessel function of the
```

```
         DBESK1-D  third kind of order one.

         BESK1E-S  Compute the exponentially scaled modified (hyperbolic)
         DBSK1E-D  Bessel function of the third kind of order one.

C10B3.  Real argument, real order

         BESI-S    Compute an N member sequence of I Bessel functions
         DBESI-D   I/SUB(ALPHA+K-1)/(X), K=1,...,N or scaled Bessel functions
                   EXP(-X)*I/SUB(ALPHA+K-1)/(X), K=1,...,N for non-negative
                   ALPHA and X.

         BESK-S    Implement forward recursion on the three term recursion
         DBESK-D   relation for a sequence of non-negative order Bessel
                   functions K/SUB(FNU+I-1)/(X), or scaled Bessel functions
                   EXP(X)*K/SUB(FNU+I-1)/(X), I=1,...,N for real, positive
                   X and non-negative orders FNU.

         BESKES-S  Compute a sequence of exponentially scaled modified Bessel
         DBSKES-D  functions of the third kind of fractional order.

         BESKS-S   Compute a sequence of modified Bessel functions of the
         DBESKS-D  third kind of fractional order.

C10B4.  Complex argument, real order

         CBESI-C   Compute a sequence of the Bessel functions I(a,z) for
         ZBESI-C   complex argument z and real nonnegative orders a=b,b+1,
                   b+2,... where b>0.  A scaling option is available to
                   help avoid overflow.

         CBESK-C   Compute a sequence of the Bessel functions K(a,z) for
         ZBESK-C   complex argument z and real nonnegative orders a=b,b+1,
                   b+2,... where b>0.  A scaling option is available to
                   help avoid overflow.

C10D.  Airy and Scorer functions

         AI-S      Evaluate the Airy function.
         DAI-D

         AIE-S     Calculate the Airy function for a negative argument and an
         DAIE-D    exponentially scaled Airy function for a non-negative
                   argument.

         BI-S      Evaluate the Bairy function (the Airy function of the
         DBI-D     second kind).

         BIE-S     Calculate the Bairy function for a negative argument and an
         DBIE-D    exponentially scaled Bairy function for a non-negative
                   argument.

         CAIRY-C   Compute the Airy function Ai(z) or its derivative dAi/dz
         ZAIRY-C   for complex argument z.  A scaling option is available
                   to help avoid underflow and overflow.

         CBIRY-C   Compute the Airy function Bi(z) or its derivative dBi/dz
         ZBIRY-C   for complex argument z.  A scaling option is available
                   to help avoid overflow.

C10F.  Integrals of Bessel functions
```

BSKIN-S — Compute repeated integrals of the K-zero Bessel function.
DBSKIN-D

C11. Confluent hypergeometric functions

CHU-S — Compute the logarithmic confluent hypergeometric function.
DCHU-D

C14. Elliptic integrals

RC-S — Calculate an approximation to
DRC-D
RC(X,Y) = Integral from zero to infinity of

$$(1/2)(t+X)^{-1/2}(t+Y)^{-1}\ dt,$$

where X is nonnegative and Y is positive.

RD-S — Compute the incomplete or complete elliptic integral of the
DRD-D — 2nd kind. For X and Y nonnegative, X+Y and Z positive,
RD(X,Y,Z) = Integral from zero to infinity of

$$(3/2)(t+X)^{-1/2}(t+Y)^{-1/2}(t+Z)^{-3/2}\ dt.$$

If X or Y is zero, the integral is complete.

RF-S — Compute the incomplete or complete elliptic integral of the
DRF-D — 1st kind. For X, Y, and Z non-negative and at most one of
them zero, RF(X,Y,Z) = Integral from zero to infinity of

$$(1/2)(t+X)^{-1/2}(t+Y)^{-1/2}(t+Z)^{-1/2}\ dt.$$

If X, Y or Z is zero, the integral is complete.

RJ-S — Compute the incomplete or complete (X or Y or Z is zero)
DRJ-D — elliptic integral of the 3rd kind. For X, Y, and Z non-
negative, at most one of them zero, and P positive,
RJ(X,Y,Z,P) = Integral from zero to infinity of

$$(3/2)(t+X)^{-1/2}(t+Y)^{-1/2}(t+Z)^{-1/2}(t+P)^{-1}\ dt.$$

C19. Other special functions

RC3JJ-S — Evaluate the 3j symbol $f(L1) = \begin{pmatrix} L1 & L2 & L3 \\ -M2-M3 & M2 & M3 \end{pmatrix}$
DRC3JJ-D
for all allowed values of L1, the other parameters
being held fixed.

RC3JM-S — Evaluate the 3j symbol $g(M2) = \begin{pmatrix} L1 & L2 & L3 \\ M1 & M2 & -M1-M2 \end{pmatrix}$
DRC3JM-D
for all allowed values of M2, the other parameters
being held fixed.

RC6J-S — Evaluate the 6j symbol $h(L1) = \begin{Bmatrix} L1 & L2 & L3 \\ L4 & L5 & L6 \end{Bmatrix}$
DRC6J-D
for all allowed values of L1, the other parameters
being held fixed.

D. Linear Algebra
D1. Elementary vector and matrix operations
D1A. Elementary vector operations
D1A2. Minimum and maximum components

ISAMAX-S — Find the smallest index of that component of a vector

```
          IDAMAX-D  having the maximum magnitude.
          ICAMAX-C

D1A3.  Norm
D1A3A.  L-1 (sum of magnitudes)

          SASUM-S   Compute the sum of the magnitudes of the elements of a
          DASUM-D   vector.
          SCASUM-C

D1A3B.  L-2 (Euclidean norm)

          SNRM2-S   Compute the Euclidean length (L2 norm) of a vector.
          DNRM2-D
          SCNRM2-C

D1A4.  Dot product (inner product)

          CDOTC-C   Dot product of two complex vectors using the complex
                    conjugate of the first vector.

          DQDOTA-D  Compute the inner product of two vectors with extended
                    precision accumulation and result.

          DQDOTI-D  Compute the inner product of two vectors with extended
                    precision accumulation and result.

          DSDOT-D   Compute the inner product of two vectors with extended
          DCDOT-C   precision accumulation and result.

          SDOT-S    Compute the inner product of two vectors.
          DDOT-D
          CDOTU-C

          SDSDOT-S  Compute the inner product of two vectors with extended
          CDCDOT-C  precision accumulation.

D1A5.  Copy or exchange (swap)

          ICOPY-S   Copy a vector.
          DCOPY-D
          CCOPY-C
          ICOPY-I

          SCOPY-S   Copy a vector.
          DCOPY-D
          CCOPY-C
          ICOPY-I

          SCOPYM-S  Copy the negative of a vector to a vector.
          DCOPYM-D

          SSWAP-S   Interchange two vectors.
          DSWAP-D
          CSWAP-C
          ISWAP-I

D1A6.  Multiplication by scalar

          CSSCAL-C  Scale a complex vector.
```

```
          SSCAL-S   Multiply a vector by a constant.
          DSCAL-D
          CSCAL-C

D1A7.  Triad (a*x+y for vectors x,y and scalar a)

          SAXPY-S   Compute a constant times a vector plus a vector.
          DAXPY-D
          CAXPY-C

D1A8.  Elementary rotation (Givens transformation)

          SROT-S    Apply a plane Givens rotation.
          DROT-D
          CSROT-C

          SROTM-S   Apply a modified Givens transformation.
          DROTM-D

D1B.  Elementary matrix operations
D1B4.  Multiplication by vector

          CHPR-C    Perform the hermitian rank 1 operation.

          DGER-D    Perform the rank 1 operation.

          DSPR-D    Perform the symmetric rank 1 operation.

          DSYR-D    Perform the symmetric rank 1 operation.

          SGBMV-S   Multiply a real vector by a real general band matrix.
          DGBMV-D
          CGBMV-C

          SGEMV-S   Multiply a real vector by a real general matrix.
          DGEMV-D
          CGEMV-C

          SGER-S    Perform rank 1 update of a real general matrix.

          CGERC-C   Perform conjugated rank 1 update of a complex general
          SGERC-S   matrix.
          DGERC-D

          CGERU-C   Perform unconjugated rank 1 update of a complex general
          SGERU-S   matrix.
          DGERU-D

          CHBMV-C   Multiply a complex vector by a complex Hermitian band
          SHBMV-S   matrix.
          DHBMV-D

          CHEMV-C   Multiply a complex vector by a complex Hermitian matrix.
          SHEMV-S
          DHEMV-D

          CHER-C    Perform Hermitian rank 1 update of a complex Hermitian
          SHER-S    matrix.
          DHER-D

          CHER2-C   Perform Hermitian rank 2 update of a complex Hermitian
```

| | |
|---|---|
| SHER2-S<br>DHER2-D | matrix. |
| CHPMV-C<br>SHPMV-S<br>DHPMV-D | Perform the matrix-vector operation. |
| CHPR2-C<br>SHPR2-S<br>DHPR2-D | Perform the hermitian rank 2 operation. |
| SSBMV-S<br>DSBMV-D<br>CSBMV-C | Multiply a real vector by a real symmetric band matrix. |
| SSDI-S<br>DSDI-D | Diagonal Matrix Vector Multiply.<br>Routine to calculate the product X = DIAG*B, where DIAG is a diagonal matrix. |
| SSMTV-S<br>DSMTV-D | SLAP Column Format Sparse Matrix Transpose Vector Product.<br>Routine to calculate the sparse matrix vector product: Y = A'*X, where ' denotes transpose. |
| SSMV-S<br>DSMV-D | SLAP Column Format Sparse Matrix Vector Product.<br>Routine to calculate the sparse matrix vector product: Y = A*X. |
| SSPMV-S<br>DSPMV-D<br>CSPMV-C | Perform the matrix-vector operation. |
| SSPR-S | Performs the symmetric rank 1 operation. |
| SSPR2-S<br>DSPR2-D<br>CSPR2-C | Perform the symmetric rank 2 operation. |
| SSYMV-S<br>DSYMV-D<br>CSYMV-C | Multiply a real vector by a real symmetric matrix. |
| SSYR-S | Perform symmetric rank 1 update of a real symmetric matrix. |
| SSYR2-S<br>DSYR2-D<br>CSYR2-C | Perform symmetric rank 2 update of a real symmetric matrix. |
| STBMV-S<br>DTBMV-D<br>CTBMV-C | Multiply a real vector by a real triangular band matrix. |
| STBSV-S<br>DTBSV-D<br>CTBSV-C | Solve a real triangular banded system of linear equations. |
| STPMV-S<br>DTPMV-D<br>CTPMV-C | Perform one of the matrix-vector operations. |
| STPSV-S<br>DTPSV-D | Solve one of the systems of equations. |

```
          CTPSV-C

          STRMV-S   Multiply a real vector by a real triangular matrix.
          DTRMV-D
          CTRMV-C

          STRSV-S   Solve a real triangular system of linear equations.
          DTRSV-D
          CTRSV-C

D1B6.  Multiplication

          SGEMM-S   Multiply a real general matrix by a real general matrix.
          DGEMM-D
          CGEMM-C

          CHEMM-C   Multiply a complex general matrix by a complex Hermitian
          SHEMM-S   matrix.
          DHEMM-D

          CHER2K-C  Perform Hermitian rank 2k update of a complex.
          SHER2-S
          DHER2-D
          CHER2-C

          CHERK-C   Perform Hermitian rank k update of a complex Hermitian
          SHERK-S   matrix.
          DHERK-D

          SSYMM-S   Multiply a real general matrix by a real symmetric matrix.
          DSYMM-D
          CSYMM-C

          DSYR2K-D  Perform one of the symmetric rank 2k operations.
          SSYR2-S
          DSYR2-D
          CSYR2-C

          SSYRK-S   Perform symmetric rank k update of a real symmetric matrix.
          DSYRK-D
          CSYRK-C

          STRMM-S   Multiply a real general matrix by a real triangular matrix.
          DTRMM-D
          CTRMM-C

          STRSM-S   Solve a real triangular system of equations with multiple
          DTRSM-D   right-hand sides.
          CTRSM-C

D1B9.  Storage mode conversion

          SS2Y-S    SLAP Triad to SLAP Column Format Converter.
          DS2Y-D    Routine to convert from the SLAP Triad to SLAP Column
                    format.

D1B10.  Elementary rotation (Givens transformation)

          CSROT-C   Apply a plane Givens rotation.
          SROT-S
          DROT-D
```

```
          SROTG-S   Construct a plane Givens rotation.
          DROTG-D
          CROTG-C

          SROTMG-S  Construct a modified Givens transformation.
          DROTMG-D
```

D2. Solution of systems of linear equations (including inversion, LU and related decompositions)

D2A. Real nonsymmetric matrices

D2A1. General

```
          SGECO-S   Factor a matrix using Gaussian elimination and estimate
          DGECO-D   the condition number of the matrix.
          CGECO-C

          SGEDI-S   Compute the determinant and inverse of a matrix using the
          DGEDI-D   factors computed by SGECO or SGEFA.
          CGEDI-C

          SGEFA-S   Factor a matrix using Gaussian elimination.
          DGEFA-D
          CGEFA-C

          SGEFS-S   Solve a general system of linear equations.
          DGEFS-D
          CGEFS-C

          SGEIR-S   Solve a general system of linear equations.  Iterative
          CGEIR-C   refinement is used to obtain an error estimate.

          SGESL-S   Solve the real system A*X=B or TRANS(A)*X=B using the
          DGESL-D   factors of SGECO or SGEFA.
          CGESL-C

          SQRSL-S   Apply the output of SQRDC to compute coordinate transfor-
          DQRSL-D   mations, projections, and least squares solutions.
          CQRSL-C
```

D2A2. Banded

```
          SGBCO-S   Factor a band matrix by Gaussian elimination and
          DGBCO-D   estimate the condition number of the matrix.
          CGBCO-C

          SGBFA-S   Factor a band matrix using Gaussian elimination.
          DGBFA-D
          CGBFA-C

          SGBSL-S   Solve the real band system A*X=B or TRANS(A)*X=B using
          DGBSL-D   the factors computed by SGBCO or SGBFA.
          CGBSL-C

          SNBCO-S   Factor a band matrix using Gaussian elimination and
          DNBCO-D   estimate the condition number.
          CNBCO-C

          SNBFA-S   Factor a real band matrix by elimination.
          DNBFA-D
          CNBFA-C
```

| | |
|---|---|
| SNBFS-S<br>DNBFS-D<br>CNBFS-C | Solve a general nonsymmetric banded system of linear equations. |
| SNBIR-S<br>CNBIR-C | Solve a general nonsymmetric banded system of linear equations. Iterative refinement is used to obtain an error estimate. |
| SNBSL-S<br>DNBSL-D<br>CNBSL-C | Solve a real band system using the factors computed by SNBCO or SNBFA. |

## D2A2A. Tridiagonal

| | |
|---|---|
| SGTSL-S<br>DGTSL-D<br>CGTSL-C | Solve a tridiagonal linear system. |

## D2A3. Triangular

| | |
|---|---|
| SSLI-S<br>DSLI-D | SLAP MSOLVE for Lower Triangle Matrix. This routine acts as an interface between the SLAP generic MSOLVE calling convention and the routine that actually computes $L^{-1} B = X$. |
| SSLI2-S<br>DSLI2-D | SLAP Lower Triangle Matrix Backsolve. Routine to solve a system of the form Lx = b , where L is a lower triangular matrix. |
| STRCO-S<br>DTRCO-D<br>CTRCO-C | Estimate the condition number of a triangular matrix. |
| STRDI-S<br>DTRDI-D<br>CTRDI-C | Compute the determinant and inverse of a triangular matrix. |
| STRSL-S<br>DTRSL-D<br>CTRSL-C | Solve a system of the form T*X=B or TRANS(T)*X=B, where T is a triangular matrix. |

## D2A4. Sparse

| | |
|---|---|
| SBCG-S<br>DBCG-D | Preconditioned BiConjugate Gradient Sparse Ax = b Solver. Routine to solve a Non-Symmetric linear system Ax = b using the Preconditioned BiConjugate Gradient method. |
| SCGN-S<br>DCGN-D | Preconditioned CG Sparse Ax=b Solver for Normal Equations. Routine to solve a general linear system Ax = b using the Preconditioned Conjugate Gradient method applied to the normal equations AA'y = b, x=A'y. |
| SCGS-S<br>DCGS-D | Preconditioned BiConjugate Gradient Squared Ax=b Solver. Routine to solve a Non-Symmetric linear system Ax = b using the Preconditioned BiConjugate Gradient Squared method. |
| SGMRES-S<br>DGMRES-D | Preconditioned GMRES Iterative Sparse Ax=b Solver. This routine uses the generalized minimum residual |

```
             (GMRES) method with preconditioning to solve
             non-symmetric linear systems of the form: Ax = b.

SIR-S        Preconditioned Iterative Refinement Sparse Ax = b Solver.
DIR-D        Routine to solve a general linear system  Ax = b  using
             iterative refinement with a matrix splitting.

SLPDOC-S     Sparse Linear Algebra Package Version 2.0.2 Documentation.
DLPDOC-D     Routines to solve large sparse symmetric and nonsymmetric
             positive definite linear systems, Ax = b, using precondi-
             tioned iterative methods.

SOMN-S       Preconditioned Orthomin Sparse Iterative Ax=b Solver.
DOMN-D       Routine to solve a general linear system  Ax = b  using
             the Preconditioned Orthomin method.

SSDBCG-S     Diagonally Scaled BiConjugate Gradient Sparse Ax=b Solver.
DSDBCG-D     Routine to solve a linear system  Ax = b  using the
             BiConjugate Gradient method with diagonal scaling.

SSDCGN-S     Diagonally Scaled CG Sparse Ax=b Solver for Normal Eqn's.
DSDCGN-D     Routine to solve a general linear system  Ax = b  using
             diagonal scaling with the Conjugate Gradient method
             applied to the the normal equations, viz.,  AA'y = b,
             where  x = A'y.

SSDCGS-S     Diagonally Scaled CGS Sparse Ax=b Solver.
DSDCGS-D     Routine to solve a linear system  Ax = b  using the
             BiConjugate Gradient Squared method with diagonal scaling.

SSDGMR-S     Diagonally Scaled GMRES Iterative Sparse Ax=b Solver.
DSDGMR-D     This routine uses the generalized minimum residual
             (GMRES) method with diagonal scaling to solve possibly
             non-symmetric linear systems of the form: Ax = b.

SSDOMN-S     Diagonally Scaled Orthomin Sparse Iterative Ax=b Solver.
DSDOMN-D     Routine to solve a general linear system  Ax = b  using
             the Orthomin method with diagonal scaling.

SSGS-S       Gauss-Seidel Method Iterative Sparse Ax = b Solver.
DSGS-D       Routine to solve a general linear system  Ax = b  using
             Gauss-Seidel iteration.

SSILUR-S     Incomplete LU Iterative Refinement Sparse Ax = b Solver.
DSILUR-D     Routine to solve a general linear system  Ax = b  using
             the incomplete LU decomposition with iterative refinement.

SSJAC-S      Jacobi's Method Iterative Sparse Ax = b Solver.
DSJAC-D      Routine to solve a general linear system  Ax = b  using
             Jacobi iteration.

SSLUBC-S     Incomplete LU BiConjugate Gradient Sparse Ax=b Solver.
DSLUBC-D     Routine to solve a linear system  Ax = b  using the
             BiConjugate Gradient method with Incomplete LU
             decomposition preconditioning.

SSLUCN-S     Incomplete LU CG Sparse Ax=b Solver for Normal Equations.
DSLUCN-D     Routine to solve a general linear system  Ax = b  using the
             incomplete LU decomposition with the Conjugate Gradient
             method applied to the normal equations, viz.,  AA'y = b,
             x = A'y.
```

```
          SSLUCS-S  Incomplete LU BiConjugate Gradient Squared Ax=b Solver.
          DSLUCS-D  Routine to solve a linear system  Ax = b  using the
                    BiConjugate Gradient Squared method with Incomplete LU
                    decomposition preconditioning.

          SSLUGM-S  Incomplete LU GMRES Iterative Sparse Ax=b Solver.
          DSLUGM-D  This routine uses the generalized minimum residual
                    (GMRES) method with incomplete LU factorization for
                    preconditioning to solve possibly non-symmetric linear
                    systems of the form: Ax = b.

          SSLUOM-S  Incomplete LU Orthomin Sparse Iterative Ax=b Solver.
          DSLUOM-D  Routine to solve a general linear system  Ax = b  using
                    the Orthomin method with Incomplete LU decomposition.

D2B.  Real symmetric matrices
D2B1.  General
D2B1A.  Indefinite

          SSICO-S   Factor a symmetric matrix by elimination with symmetric
          DSICO-D   pivoting and estimate the condition number of the matrix.
          CHICO-C
          CSICO-C

          SSIDI-S   Compute the determinant, inertia and inverse of a real
          DSIDI-D   symmetric matrix using the factors from SSIFA.
          CHIDI-C
          CSIDI-C

          SSIFA-S   Factor a real symmetric matrix by elimination with
          DSIFA-D   symmetric pivoting.
          CHIFA-C
          CSIFA-C

          SSISL-S   Solve a real symmetric system using the factors obtained
          DSISL-D   from SSIFA.
          CHISL-C
          CSISL-C

          SSPCO-S   Factor a real symmetric matrix stored in packed form
          DSPCO-D   by elimination with symmetric pivoting and estimate the
          CHPCO-C   condition number of the matrix.
          CSPCO-C

          SSPDI-S   Compute the determinant, inertia, inverse of a real
          DSPDI-D   symmetric matrix stored in packed form using the factors
          CHPDI-C   from SSPFA.
          CSPDI-C

          SSPFA-S   Factor a real symmetric matrix stored in packed form by
          DSPFA-D   elimination with symmetric pivoting.
          CHPFA-C
          CSPFA-C

          SSPSL-S   Solve a real symmetric system using the factors obtained
          DSPSL-D   from SSPFA.
          CHPSL-C
          CSPSL-C

D2B1B.  Positive definite
```

| | |
|---|---|
| SCHDC-S<br>DCHDC-D<br>CCHDC-C | Compute the Cholesky decomposition of a positive definite matrix. A pivoting option allows the user to estimate the condition number of a positive definite matrix or determine the rank of a positive semidefinite matrix. |
| SPOCO-S<br>DPOCO-D<br>CPOCO-C | Factor a real symmetric positive definite matrix and estimate the condition number of the matrix. |
| SPODI-S<br>DPODI-D<br>CPODI-C | Compute the determinant and inverse of a certain real symmetric positive definite matrix using the factors computed by SPOCO, SPOFA or SQRDC. |
| SPOFA-S<br>DPOFA-D<br>CPOFA-C | Factor a real symmetric positive definite matrix. |
| SPOFS-S<br>DPOFS-D<br>CPOFS-C | Solve a positive definite symmetric system of linear equations. |
| SPOIR-S<br>CPOIR-C | Solve a positive definite symmetric system of linear equations. Iterative refinement is used to obtain an error estimate. |
| SPOSL-S<br>DPOSL-D<br>CPOSL-C | Solve the real symmetric positive definite linear system using the factors computed by SPOCO or SPOFA. |
| SPPCO-S<br>DPPCO-D<br>CPPCO-C | Factor a symmetric positive definite matrix stored in packed form and estimate the condition number of the matrix. |
| SPPDI-S<br>DPPDI-D<br>CPPDI-C | Compute the determinant and inverse of a real symmetric positive definite matrix using factors from SPPCO or SPPFA. |
| SPPFA-S<br>DPPFA-D<br>CPPFA-C | Factor a real symmetric positive definite matrix stored in packed form. |
| SPPSL-S<br>DPPSL-D<br>CPPSL-C | Solve the real symmetric positive definite system using the factors computed by SPPCO or SPPFA. |

D2B2. Positive definite banded

| | |
|---|---|
| SPBCO-S<br>DPBCO-D<br>CPBCO-C | Factor a real symmetric positive definite matrix stored in band form and estimate the condition number of the matrix. |
| SPBFA-S<br>DPBFA-D<br>CPBFA-C | Factor a real symmetric positive definite matrix stored in band form. |
| SPBSL-S<br>DPBSL-D<br>CPBSL-C | Solve a real symmetric positive definite band system using the factors computed by SPBCO or SPBFA. |

D2B2A. Tridiagonal

SPTSL-S   Solve a positive definite tridiagonal linear system.
DPTSL-D
CPTSL-C

D2B4. Sparse

SBCG-S    Preconditioned BiConjugate Gradient Sparse Ax = b Solver.
DBCG-D    Routine to solve a Non-Symmetric linear system  Ax = b
          using the Preconditioned BiConjugate Gradient method.

SCG-S     Preconditioned Conjugate Gradient Sparse Ax=b Solver.
DCG-D     Routine to solve a symmetric positive definite linear
          system  Ax = b  using the Preconditioned Conjugate
          Gradient method.

SCGN-S    Preconditioned CG Sparse Ax=b Solver for Normal Equations.
DCGN-D    Routine to solve a general linear system  Ax = b  using the
          Preconditioned Conjugate Gradient method applied to the
          normal equations  AA'y = b, x=A'y.

SCGS-S    Preconditioned BiConjugate Gradient Squared Ax=b Solver.
DCGS-D    Routine to solve a Non-Symmetric linear system  Ax = b
          using the Preconditioned BiConjugate Gradient Squared
          method.

SGMRES-S  Preconditioned GMRES Iterative Sparse Ax=b Solver.
DGMRES-D  This routine uses the generalized minimum residual
          (GMRES) method with preconditioning to solve
          non-symmetric linear systems of the form: Ax = b.

SIR-S     Preconditioned Iterative Refinement Sparse Ax = b Solver.
DIR-D     Routine to solve a general linear system  Ax = b  using
          iterative refinement with a matrix splitting.

SLPDOC-S  Sparse Linear Algebra Package Version 2.0.2 Documentation.
DLPDOC-D  Routines to solve large sparse symmetric and nonsymmetric
          positive definite linear systems, Ax = b, using precondi-
          tioned iterative methods.

SOMN-S    Preconditioned Orthomin Sparse Iterative Ax=b Solver.
DOMN-D    Routine to solve a general linear system  Ax = b  using
          the Preconditioned Orthomin method.

SSDBCG-S  Diagonally Scaled BiConjugate Gradient Sparse Ax=b Solver.
DSDBCG-D  Routine to solve a linear system  Ax = b  using the
          BiConjugate Gradient method with diagonal scaling.

SSDCG-S   Diagonally Scaled Conjugate Gradient Sparse Ax=b Solver.
DSDCG-D   Routine to solve a symmetric positive definite linear
          system  Ax = b  using the Preconditioned Conjugate
          Gradient method.  The preconditioner is diagonal scaling.

SSDCGN-S  Diagonally Scaled CG Sparse Ax=b Solver for Normal Eqn's.
DSDCGN-D  Routine to solve a general linear system  Ax = b  using
          diagonal scaling with the Conjugate Gradient method
          applied to the the normal equations, viz.,  AA'y = b,
          where  x = A'y.

SSDCGS-S  Diagonally Scaled CGS Sparse Ax=b Solver.
DSDCGS-D  Routine to solve a linear system  Ax = b  using the

BiConjugate Gradient Squared method with diagonal scaling.

SSDGMR-S  Diagonally Scaled GMRES Iterative Sparse Ax=b Solver.
DSDGMR-D  This routine uses the generalized minimum residual (GMRES) method with diagonal scaling to solve possibly non-symmetric linear systems of the form: Ax = b.

SSDOMN-S  Diagonally Scaled Orthomin Sparse Iterative Ax=b Solver.
DSDOMN-D  Routine to solve a general linear system Ax = b using the Orthomin method with diagonal scaling.

SSGS-S  Gauss-Seidel Method Iterative Sparse Ax = b Solver.
DSGS-D  Routine to solve a general linear system Ax = b using Gauss-Seidel iteration.

SSICCG-S  Incomplete Cholesky Conjugate Gradient Sparse Ax=b Solver.
DSICCG-D  Routine to solve a symmetric positive definite linear system Ax = b using the incomplete Cholesky Preconditioned Conjugate Gradient method.

SSILUR-S  Incomplete LU Iterative Refinement Sparse Ax = b Solver.
DSILUR-D  Routine to solve a general linear system Ax = b using the incomplete LU decomposition with iterative refinement.

SSJAC-S  Jacobi's Method Iterative Sparse Ax = b Solver.
DSJAC-D  Routine to solve a general linear system Ax = b using Jacobi iteration.

SSLUBC-S  Incomplete LU BiConjugate Gradient Sparse Ax=b Solver.
DSLUBC-D  Routine to solve a linear system Ax = b using the BiConjugate Gradient method with Incomplete LU decomposition preconditioning.

SSLUCN-S  Incomplete LU CG Sparse Ax=b Solver for Normal Equations.
DSLUCN-D  Routine to solve a general linear system Ax = b using the incomplete LU decomposition with the Conjugate Gradient method applied to the normal equations, viz., AA'y = b, x = A'y.

SSLUCS-S  Incomplete LU BiConjugate Gradient Squared Ax=b Solver.
DSLUCS-D  Routine to solve a linear system Ax = b using the BiConjugate Gradient Squared method with Incomplete LU decomposition preconditioning.

SSLUGM-S  Incomplete LU GMRES Iterative Sparse Ax=b Solver.
DSLUGM-D  This routine uses the generalized minimum residual (GMRES) method with incomplete LU factorization for preconditioning to solve possibly non-symmetric linear systems of the form: Ax = b.

SSLUOM-S  Incomplete LU Orthomin Sparse Iterative Ax=b Solver.
DSLUOM-D  Routine to solve a general linear system Ax = b using the Orthomin method with Incomplete LU decomposition.

D2C. Complex non-Hermitian matrices
D2C1. General

CGECO-C  Factor a matrix using Gaussian elimination and estimate
SGECO-S  the condition number of the matrix.
DGECO-D

CGEDI-C   Compute the determinant and inverse of a matrix using the
SGEDI-S   factors computed by CGECO or CGEFA.
DGEDI-D

CGEFA-C   Factor a matrix using Gaussian elimination.
SGEFA-S
DGEFA-D

CGEFS-C   Solve a general system of linear equations.
SGEFS-S
DGEFS-D

CGEIR-C   Solve a general system of linear equations.  Iterative
SGEIR-S   refinement is used to obtain an error estimate.

CGESL-C   Solve the complex system A*X=B or CTRANS(A)*X=B using the
SGESL-S   factors computed by CGECO or CGEFA.
DGESL-D

CQRSL-C   Apply the output of CQRDC to compute coordinate transfor-
SQRSL-S   mations, projections, and least squares solutions.
DQRSL-D

CSICO-C   Factor a complex symmetric matrix by elimination with
SSICO-S   symmetric pivoting and estimate the condition number of the
DSICO-D   matrix.
CHICO-C

CSIDI-C   Compute the determinant and inverse of a complex symmetric
SSIDI-S   matrix using the factors from CSIFA.
DSIDI-D
CHIDI-C

CSIFA-C   Factor a complex symmetric matrix by elimination with
SSIFA-S   symmetric pivoting.
DSIFA-D
CHIFA-C

CSISL-C   Solve a complex symmetric system using the factors obtained
SSISL-S   from CSIFA.
DSISL-D
CHISL-C

CSPCO-C   Factor a complex symmetric matrix stored in packed form
SSPCO-S   by elimination with symmetric pivoting and estimate the
DSPCO-D   condition number of the matrix.
CHPCO-C

CSPDI-C   Compute the determinant and inverse of a complex symmetric
SSPDI-S   matrix stored in packed form using the factors from CSPFA.
DSPDI-D
CHPDI-C

CSPFA-C   Factor a complex symmetric matrix stored in packed form by
SSPFA-S   elimination with symmetric pivoting.
DSPFA-D
CHPFA-C

CSPSL-C   Solve a complex symmetric system using the factors obtained
SSPSL-S   from CSPFA.
DSPSL-D

```
          CHPSL-C

D2C2.  Banded

          CGBCO-C   Factor a band matrix by Gaussian elimination and
          SGBCO-S   estimate the condition number of the matrix.
          DGBCO-D

          CGBFA-C   Factor a band matrix using Gaussian elimination.
          SGBFA-S
          DGBFA-D

          CGBSL-C   Solve the complex band system A*X=B or CTRANS(A)*X=B using
          SGBSL-S   the factors computed by CGBCO or CGBFA.
          DGBSL-D

          CNBCO-C   Factor a band matrix using Gaussian elimination and
          SNBCO-S   estimate the condition number.
          DNBCO-D

          CNBFA-C   Factor a band matrix by elimination.
          SNBFA-S
          DNBFA-D

          CNBFS-C   Solve a general nonsymmetric banded system of linear
          SNBFS-S   equations.
          DNBFS-D

          CNBIR-C   Solve a general nonsymmetric banded system of linear
          SNBIR-S   equations.  Iterative refinement is used to obtain an error
                    estimate.

          CNBSL-C   Solve a complex band system using the factors computed by
          SNBSL-S   CNBCO or CNBFA.
          DNBSL-D

D2C2A.  Tridiagonal

          CGTSL-C   Solve a tridiagonal linear system.
          SGTSL-S
          DGTSL-D

D2C3.  Triangular

          CTRCO-C   Estimate the condition number of a triangular matrix.
          STRCO-S
          DTRCO-D

          CTRDI-C   Compute the determinant and inverse of a triangular matrix.
          STRDI-S
          DTRDI-D

          CTRSL-C   Solve a system of the form  T*X=B or CTRANS(T)*X=B, where
          STRSL-S   T is a triangular matrix.  Here CTRANS(T) is the conjugate
          DTRSL-D   transpose.

D2D.  Complex Hermitian matrices
D2D1.  General
D2D1A.  Indefinite

          CHICO-C   Factor a complex Hermitian matrix by elimination with sym-
```

```
SSICO-S   metric pivoting and estimate the condition of the matrix.
DSICO-D
CSICO-C

CHIDI-C   Compute the determinant, inertia and inverse of a complex
SSIDI-S   Hermitian matrix using the factors obtained from CHIFA.
DSISI-D
CSIDI-C

CHIFA-C   Factor a complex Hermitian matrix by elimination
SSIFA-S   (symmetric pivoting).
DSIFA-D
CSIFA-C

CHISL-C   Solve the complex Hermitian system using factors obtained
SSISL-S   from CHIFA.
DSISL-D
CSISL-C

CHPCO-C   Factor a complex Hermitian matrix stored in packed form by
SSPCO-S   elimination with symmetric pivoting and estimate the
DSPCO-D   condition number of the matrix.
CSPCO-C

CHPDI-C   Compute the determinant, inertia and inverse of a complex
SSPDI-S   Hermitian matrix stored in packed form using the factors
DSPDI-D   obtained from CHPFA.
DSPDI-C

CHPFA-C   Factor a complex Hermitian matrix stored in packed form by
SSPFA-S   elimination with symmetric pivoting.
DSPFA-D
DSPFA-C

CHPSL-C   Solve a complex Hermitian system using factors obtained
SSPSL-S   from CHPFA.
DSPSL-D
CSPSL-C
```

D2D1B.  Positive definite

```
CCHDC-C   Compute the Cholesky decomposition of a positive definite
SCHDC-S   matrix.  A pivoting option allows the user to estimate the
DCHDC-D   condition number of a positive definite matrix or determine
          the rank of a positive semidefinite matrix.

CPOCO-C   Factor a complex Hermitian positive definite matrix
SPOCO-S   and estimate the condition number of the matrix.
DPOCO-D

CPODI-C   Compute the determinant and inverse of a certain complex
SPODI-S   Hermitian positive definite matrix using the factors
DPODI-D   computed by CPOCO, CPOFA, or CQRDC.

CPOFA-C   Factor a complex Hermitian positive definite matrix.
SPOFA-S
DPOFA-D

CPOFS-C   Solve a positive definite symmetric complex system of
SPOFS-S   linear equations.
DPOFS-D
```

```
          CPOIR-C   Solve a positive definite Hermitian system of linear
          SPOIR-S   equations.  Iterative refinement is used to obtain an
                    error estimate.

          CPOSL-C   Solve the complex Hermitian positive definite linear system
          SPOSL-S   using the factors computed by CPOCO or CPOFA.
          DPOSL-D

          CPPCO-C   Factor a complex Hermitian positive definite matrix stored
          SPPCO-S   in packed form and estimate the condition number of the
          DPPCO-D   matrix.

          CPPDI-C   Compute the determinant and inverse of a complex Hermitian
          SPPDI-S   positive definite matrix using factors from CPPCO or CPPFA.
          DPPDI-D

          CPPFA-C   Factor a complex Hermitian positive definite matrix stored
          SPPFA-S   in packed form.
          DPPFA-D

          CPPSL-C   Solve the complex Hermitian positive definite system using
          SPPSL-S   the factors computed by CPPCO or CPPFA.
          DPPSL-D

D2D2.  Positive definite banded

          CPBCO-C   Factor a complex Hermitian positive definite matrix stored
          SPBCO-S   in band form and estimate the condition number of the
          DPBCO-D   matrix.

          CPBFA-C   Factor a complex Hermitian positive definite matrix stored
          SPBFA-S   in band form.
          DPBFA-D

          CPBSL-C   Solve the complex Hermitian positive definite band system
          SPBSL-S   using the factors computed by CPBCO or CPBFA.
          DPBSL-D

D2D2A.  Tridiagonal

          CPTSL-C   Solve a positive definite tridiagonal linear system.
          SPTSL-S
          DPTSL-D

D2E.  Associated operations (e.g., matrix reorderings)

          SLLTI2-S  SLAP Backsolve routine for LDL' Factorization.
          DLLTI2-D  Routine to solve a system of the form  L*D*L' X = B,
                    where L is a unit lower triangular matrix and D is a
                    diagonal matrix and ' means transpose.

          SS2LT-S   Lower Triangle Preconditioner SLAP Set Up.
          DS2LT-D   Routine to store the lower triangle of a matrix stored
                    in the SLAP Column format.

          SSD2S-S   Diagonal Scaling Preconditioner SLAP Normal Eqns Set Up.
          DSD2S-D   Routine to compute the inverse of the diagonal of the
                    matrix A*A', where A is stored in SLAP-Column format.

          SSDS-S    Diagonal Scaling Preconditioner SLAP Set Up.
```

DSDS-D Routine to compute the inverse of the diagonal of a matrix stored in the SLAP Column format.

SSDSCL-S Diagonal Scaling of system Ax = b.
DSDSCL-D This routine scales (and unscales) the system Ax = b by symmetric diagonal scaling.

SSICS-S Incompl. Cholesky Decomposition Preconditioner SLAP Set Up.
DSICS-D Routine to generate the Incomplete Cholesky decomposition, L*D*L-trans, of a symmetric positive definite matrix, A, which is stored in SLAP Column format. The unit lower triangular matrix L is stored by rows, and the inverse of the diagonal matrix D is stored.

SSILUS-S Incomplete LU Decomposition Preconditioner SLAP Set Up.
DSILUS-D Routine to generate the incomplete LDU decomposition of a matrix. The unit lower triangular factor L is stored by rows and the unit upper triangular factor U is stored by columns. The inverse of the diagonal matrix D is stored. No fill in is allowed.

SSLLTI-S SLAP MSOLVE for LDL' (IC) Factorization.
DSLLTI-D This routine acts as an interface between the SLAP generic MSOLVE calling convention and the routine that actually computes $(LDL')^{-1} B = X$.

SSLUI-S SLAP MSOLVE for LDU Factorization.
DSLUI-D This routine acts as an interface between the SLAP generic MSOLVE calling convention and the routine that actually computes $(LDU)^{-1} B = X$.

SSLUI2-S SLAP Backsolve for LDU Factorization.
DSLUI2-D Routine to solve a system of the form L*D*U X = B, where L is a unit lower triangular matrix, D is a diagonal matrix, and U is a unit upper triangular matrix.

SSLUI4-S SLAP Backsolve for LDU Factorization.
DSLUI4-D Routine to solve a system of the form (L*D*U)' X = B, where L is a unit lower triangular matrix, D is a diagonal matrix, and U is a unit upper triangular matrix and ' denotes transpose.

SSLUTI-S SLAP MTSOLV for LDU Factorization.
DSLUTI-D This routine acts as an interface between the SLAP generic MTSOLV calling convention and the routine that actually computes $(LDU)^{-T} B = X$.

SSMMI2-S SLAP Backsolve for LDU Factorization of Normal Equations.
DSMMI2-D To solve a system of the form (L*D*U)*(L*D*U)' X = B, where L is a unit lower triangular matrix, D is a diagonal matrix, and U is a unit upper triangular matrix and ' denotes transpose.

SSMMTI-S SLAP MSOLVE for LDU Factorization of Normal Equations.
DSMMTI-D This routine acts as an interface between the SLAP generic MMTSLV calling convention and the routine that actually computes $[(LDU)*(LDU)']^{-1} B = X$.

D3. Determinants
D3A. Real nonsymmetric matrices
D3A1. General

```
SGEDI-S   Compute the determinant and inverse of a matrix using the
DGEDI-D   factors computed by SGECO or SGEFA.
CGEDI-C
```

D3A2. Banded

```
SGBDI-S   Compute the determinant of a band matrix using the factors
DGBDI-D   computed by SGBCO or SGBFA.
CGBDI-C

SNBDI-S   Compute the determinant of a band matrix using the factors
DNBDI-D   computed by SNBCO or SNBFA.
CNBDI-C
```

D3A3. Triangular

```
STRDI-S   Compute the determinant and inverse of a triangular matrix.
DTRDI-D
CTRDI-C
```

D3B. Real symmetric matrices
D3B1. General
D3B1A. Indefinite

```
SSIDI-S   Compute the determinant, inertia and inverse of a real
DSIDI-D   symmetric matrix using the factors from SSIFA.
CHIDI-C
CSIDI-C

SSPDI-S   Compute the determinant, inertia, inverse of a real
DSPDI-D   symmetric matrix stored in packed form using the factors
CHPDI-C   from SSPFA.
CSPDI-C
```

D3B1B. Positive definite

```
SPODI-S   Compute the determinant and inverse of a certain real
DPODI-D   symmetric positive definite matrix using the factors
CPODI-C   computed by SPOCO, SPOFA or SQRDC.

SPPDI-S   Compute the determinant and inverse of a real symmetric
DPPDI-D   positive definite matrix using factors from SPPCO or SPPFA.
CPPDI-C
```

D3B2. Positive definite banded

```
SPBDI-S   Compute the determinant of a symmetric positive definite
DPBDI-D   band matrix using the factors computed by SPBCO or SPBFA.
CPBDI-C
```

D3C. Complex non-Hermitian matrices
D3C1. General

```
CGEDI-C   Compute the determinant and inverse of a matrix using the
SGEDI-S   factors computed by CGECO or CGEFA.
DGEDI-D
```

```
          CSIDI-C   Compute the determinant and inverse of a complex symmetric
          SSIDI-S   matrix using the factors from CSIFA.
          DSIDI-D
          CHIDI-C

          CSPDI-C   Compute the determinant and inverse of a complex symmetric
          SSPDI-S   matrix stored in packed form using the factors from CSPFA.
          DSPDI-D
          CHPDI-C
```

D3C2.  Banded

```
          CGBDI-C   Compute the determinant of a complex band matrix using the
          SGBDI-S   factors from CGBCO or CGBFA.
          DGBDI-D

          CNBDI-C   Compute the determinant of a band matrix using the factors
          SNBDI-S   computed by CNBCO or CNBFA.
          DNBDI-D
```

D3C3.  Triangular

```
          CTRDI-C   Compute the determinant and inverse of a triangular matrix.
          STRDI-S
          DTRDI-D
```

D3D.  Complex Hermitian matrices
D3D1.  General
D3D1A.  Indefinite

```
          CHIDI-C   Compute the determinant, inertia and inverse of a complex
          SSIDI-S   Hermitian matrix using the factors obtained from CHIFA.
          DSISI-D
          CSIDI-C

          CHPDI-C   Compute the determinant, inertia and inverse of a complex
          SSPDI-S   Hermitian matrix stored in packed form using the factors
          DSPDI-D   obtained from CHPFA.
          DSPDI-C
```

D3D1B.  Positive definite

```
          CPODI-C   Compute the determinant and inverse of a certain complex
          SPODI-S   Hermitian positive definite matrix using the factors
          DPODI-D   computed by CPOCO, CPOFA, or CQRDC.

          CPPDI-C   Compute the determinant and inverse of a complex Hermitian
          SPPDI-S   positive definite matrix using factors from CPPCO or CPPFA.
          DPPDI-D
```

D3D2.  Positive definite banded

```
          CPBDI-C   Compute the determinant of a complex Hermitian positive
          SPBDI-S   definite band matrix using the factors computed by CPBCO or
          DPBDI-D   CPBFA.
```

D4.  Eigenvalues, eigenvectors

```
          EISDOC-A  Documentation for EISPACK, a collection of subprograms for
                    solving matrix eigen-problems.
```

```
D4A.  Ordinary eigenvalue problems (Ax = (lambda) * x)
D4A1.  Real symmetric

          RS-S      Compute the eigenvalues and, optionally, the eigenvectors
          CH-C      of a real symmetric matrix.

          RSP-S     Compute the eigenvalues and, optionally, the eigenvectors
                    of a real symmetric matrix packed into a one dimensional
                    array.

          SSIEV-S   Compute the eigenvalues and, optionally, the eigenvectors
          CHIEV-C   of a real symmetric matrix.

          SSPEV-S   Compute the eigenvalues and, optionally, the eigenvectors
                    of a real symmetric matrix stored in packed form.

D4A2.  Real nonsymmetric

          RG-S      Compute the eigenvalues and, optionally, the eigenvectors
          CG-C      of a real general matrix.

          SGEEV-S   Compute the eigenvalues and, optionally, the eigenvectors
          CGEEV-C   of a real general matrix.

D4A3.  Complex Hermitian

          CH-C      Compute the eigenvalues and, optionally, the eigenvectors
          RS-S      of a complex Hermitian matrix.

          CHIEV-C   Compute the eigenvalues and, optionally, the eigenvectors
          SSIEV-S   of a complex Hermitian matrix.

D4A4.  Complex non-Hermitian

          CG-C      Compute the eigenvalues and, optionally, the eigenvectors
          RG-S      of a complex general matrix.

          CGEEV-C   Compute the eigenvalues and, optionally, the eigenvectors
          SGEEV-S   of a complex general matrix.

D4A5.  Tridiagonal

          BISECT-S  Compute the eigenvalues of a symmetric tridiagonal matrix
                    in a given interval using Sturm sequencing.

          IMTQL1-S  Compute the eigenvalues of a symmetric tridiagonal matrix
                    using the implicit QL method.

          IMTQL2-S  Compute the eigenvalues and eigenvectors of a symmetric
                    tridiagonal matrix using the implicit QL method.

          IMTQLV-S  Compute the eigenvalues of a symmetric tridiagonal matrix
                    using the implicit QL method.  Eigenvectors may be computed
                    later.

          RATQR-S   Compute the largest or smallest eigenvalues of a symmetric
                    tridiagonal matrix using the rational QR method with Newton
                    correction.

          RST-S     Compute the eigenvalues and, optionally, the eigenvectors
```

of a real symmetric tridiagonal matrix.

RT-S      Compute the eigenvalues and eigenvectors of a special real tridiagonal matrix.

TQL1-S    Compute the eigenvalues of symmetric tridiagonal matrix by the QL method.

TQL2-S    Compute the eigenvalues and eigenvectors of symmetric tridiagonal matrix.

TQLRAT-S  Compute the eigenvalues of symmetric tridiagonal matrix using a rational variant of the QL method.

TRIDIB-S  Compute the eigenvalues of a symmetric tridiagonal matrix in a given interval using Sturm sequencing.

TSTURM-S  Find those eigenvalues of a symmetric tridiagonal matrix in a given interval and their associated eigenvectors by Sturm sequencing.

D4A6. Banded

BQR-S     Compute some of the eigenvalues of a real symmetric matrix using the QR method with shifts of origin.

RSB-S     Compute the eigenvalues and, optionally, the eigenvectors of a symmetric band matrix.

D4B. Generalized eigenvalue problems (e.g., Ax = (lambda)*Bx)
D4B1. Real symmetric

RSG-S     Compute the eigenvalues and, optionally, the eigenvectors of a symmetric generalized eigenproblem.

RSGAB-S   Compute the eigenvalues and, optionally, the eigenvectors of a symmetric generalized eigenproblem.

RSGBA-S   Compute the eigenvalues and, optionally, the eigenvectors of a symmetric generalized eigenproblem.

D4B2. Real general

RGG-S     Compute the eigenvalues and eigenvectors for a real generalized eigenproblem.

D4C. Associated operations
D4C1. Transform problem
D4C1A. Balance matrix

BALANC-S  Balance a real general matrix and isolate eigenvalues
CBAL-C    whenever possible.

D4C1B. Reduce to compact form
D4C1B1. Tridiagonal

BANDR-S   Reduce a real symmetric band matrix to symmetric tridiagonal matrix and, optionally, accumulate orthogonal similarity transformations.

HTRID3-S  Reduce a complex Hermitian (packed) matrix to a real symmetric tridiagonal matrix by unitary similarity transformations.

HTRIDI-S  Reduce a complex Hermitian matrix to a real symmetric tridiagonal matrix using unitary similarity transformations.

TRED1-S  Reduce a real symmetric matrix to symmetric tridiagonal matrix using orthogonal similarity transformations.

TRED2-S  Reduce a real symmetric matrix to a symmetric tridiagonal matrix using and accumulating orthogonal transformations.

TRED3-S  Reduce a real symmetric matrix stored in packed form to symmetric tridiagonal matrix using orthogonal transformations.

D4C1B2.  Hessenberg

ELMHES-S
COMHES-C  Reduce a real general matrix to upper Hessenberg form using stabilized elementary similarity transformations.

ORTHES-S
CORTH-C  Reduce a real general matrix to upper Hessenberg form using orthogonal similarity transformations.

D4C1B3.  Other

QZHES-S  The first step of the QZ algorithm for solving generalized matrix eigenproblems.  Accepts a pair of real general matrices and reduces one of them to upper Hessenberg and the other to upper triangular form using orthogonal transformations. Usually followed by QZIT, QZVAL, QZVEC.

QZIT-S  The second step of the QZ algorithm for generalized eigenproblems.  Accepts an upper Hessenberg and an upper triangular matrix and reduces the former to quasi-triangular form while preserving the form of the latter.  Usually preceded by QZHES and followed by QZVAL and QZVEC.

D4C1C.  Standardize problem

FIGI-S  Transforms certain real non-symmetric tridiagonal matrix to symmetric tridiagonal matrix.

FIGI2-S  Transforms certain real non-symmetric tridiagonal matrix to symmetric tridiagonal matrix.

REDUC-S  Reduce a generalized symmetric eigenproblem to a standard symmetric eigenproblem using Cholesky factorization.

REDUC2-S  Reduce a certain generalized symmetric eigenproblem to a standard symmetric eigenproblem using Cholesky factorization.

D4C2.  Compute eigenvalues of matrix in compact form
D4C2A.  Tridiagonal

BISECT-S  Compute the eigenvalues of a symmetric tridiagonal matrix in a given interval using Sturm sequencing.

IMTQL1-S  Compute the eigenvalues of a symmetric tridiagonal matrix using the implicit QL method.

IMTQL2-S  Compute the eigenvalues and eigenvectors of a symmetric tridiagonal matrix using the implicit QL method.

IMTQLV-S  Compute the eigenvalues of a symmetric tridiagonal matrix using the implicit QL method.  Eigenvectors may be computed later.

RATQR-S  Compute the largest or smallest eigenvalues of a symmetric tridiagonal matrix using the rational QR method with Newton correction.

TQL1-S  Compute the eigenvalues of symmetric tridiagonal matrix by the QL method.

TQL2-S  Compute the eigenvalues and eigenvectors of symmetric tridiagonal matrix.

TQLRAT-S  Compute the eigenvalues of symmetric tridiagonal matrix using a rational variant of the QL method.

TRIDIB-S  Compute the eigenvalues of a symmetric tridiagonal matrix in a given interval using Sturm sequencing.

TSTURM-S  Find those eigenvalues of a symmetric tridiagonal matrix in a given interval and their associated eigenvectors by Sturm sequencing.

D4C2B.  Hessenberg

COMLR-C  Compute the eigenvalues of a complex upper Hessenberg matrix using the modified LR method.

COMLR2-C  Compute the eigenvalues and eigenvectors of a complex upper Hessenberg matrix using the modified LR method.

HQR-S
COMQR-C  Compute the eigenvalues of a real upper Hessenberg matrix using the QR method.

HQR2-S
COMQR2-C  Compute the eigenvalues and eigenvectors of a real upper Hessenberg matrix using QR method.

INVIT-S
CINVIT-C  Compute the eigenvectors of a real upper Hessenberg matrix associated with specified eigenvalues by inverse iteration.

D4C2C.  Other

QZVAL-S  The third step of the QZ algorithm for generalized eigenproblems.  Accepts a pair of real matrices, one in quasi-triangular form and the other in upper triangular form and computes the eigenvalues of the associated eigenproblem.  Usually preceded by QZHES, QZIT, and followed by QZVEC.

D4C3.  Form eigenvectors from eigenvalues

BANDV-S  Form the eigenvectors of a real symmetric band matrix associated with a set of ordered approximate eigenvalues by inverse iteration.

QZVEC-S The optional fourth step of the QZ algorithm for generalized eigenproblems. Accepts a matrix in quasi-triangular form and another in upper triangular and computes the eigenvectors of the triangular problem and transforms them back to the original coordinates Usually preceded by QZHES, QZIT, and QZVAL.

TINVIT-S Compute the eigenvectors of symmetric tridiagonal matrix corresponding to specified eigenvalues, using inverse iteration.

D4C4. Back transform eigenvectors

BAKVEC-S Form the eigenvectors of a certain real non-symmetric tridiagonal matrix from a symmetric tridiagonal matrix output from FIGI.

BALBAK-S
CBABK2-C Form the eigenvectors of a real general matrix from the eigenvectors of matrix output from BALANC.

ELMBAK-S
COMBAK-C Form the eigenvectors of a real general matrix from the eigenvectors of the upper Hessenberg matrix output from ELMHES.

ELTRAN-S Accumulates the stabilized elementary similarity transformations used in the reduction of a real general matrix to upper Hessenberg form by ELMHES.

HTRIB3-S Compute the eigenvectors of a complex Hermitian matrix from the eigenvectors of a real symmetric tridiagonal matrix output from HTRID3.

HTRIBK-S Form the eigenvectors of a complex Hermitian matrix from the eigenvectors of a real symmetric tridiagonal matrix output from HTRIDI.

ORTBAK-S
CORTB-C Form the eigenvectors of a general real matrix from the eigenvectors of the upper Hessenberg matrix output from ORTHES.

ORTRAN-S Accumulate orthogonal similarity transformations in the reduction of real general matrix by ORTHES.

REBAK-S Form the eigenvectors of a generalized symmetric eigensystem from the eigenvectors of derived matrix output from REDUC or REDUC2.

REBAKB-S Form the eigenvectors of a generalized symmetric eigensystem from the eigenvectors of derived matrix output from REDUC2.

TRBAK1-S Form the eigenvectors of real symmetric matrix from the eigenvectors of a symmetric tridiagonal matrix formed by TRED1.

TRBAK3-S Form the eigenvectors of a real symmetric matrix from the eigenvectors of a symmetric tridiagonal matrix formed by TRED3.

D5. QR decomposition, Gram-Schmidt orthogonalization

LLSIA-S
DLLSIA-D
Solve a linear least squares problems by performing a QR factorization of the matrix using Householder transformations. Emphasis is put on detecting possible rank deficiency.

SGLSS-S
DGLSS-D
Solve a linear least squares problems by performing a QR factorization of the matrix using Householder transformations. Emphasis is put on detecting possible rank deficiency.

SQRDC-S
DQRDC-D
CQRDC-C
Use Householder transformations to compute the QR factorization of an N by P matrix. Column pivoting is a users option.

D6. Singular value decomposition

SSVDC-S
DSVDC-D
CSVDC-C
Perform the singular value decomposition of a rectangular matrix.

D7. Update matrix decompositions
D7B. Cholesky

SCHDD-S
DCHDD-D
CCHDD-C
Downdate an augmented Cholesky decomposition or the triangular factor of an augmented QR decomposition.

SCHEX-S
DCHEX-D
CCHEX-C
Update the Cholesky factorization A=TRANS(R)*R of A positive definite matrix A of order P under diagonal permutations of the form TRANS(E)*A*E, where E is a permutation matrix.

SCHUD-S
DCHUD-D
CCHUD-C
Update an augmented Cholesky decomposition of the triangular part of an augmented QR decomposition.

D9. Overdetermined or underdetermined systems of equations, singular systems, pseudo-inverses (search also classes D5, D6, K1a, L8a)

BNDACC-S
DBNDAC-D
Compute the LU factorization of a banded matrices using sequential accumulation of rows of the data matrix. Exactly one right-hand side vector is permitted.

BNDSOL-S
DBNDSL-D
Solve the least squares problem for a banded matrix using sequential accumulation of rows of the data matrix. Exactly one right-hand side vector is permitted.

HFTI-S
DHFTI-D
Solve a linear least squares problems by performing a QR factorization of the matrix using Householder transformations.

LLSIA-S
DLLSIA-D
Solve a linear least squares problems by performing a QR factorization of the matrix using Householder transformations. Emphasis is put on detecting possible rank deficiency.

LSEI-S
DLSEI-D
Solve a linearly constrained least squares problem with equality and inequality constraints, and optionally compute a covariance matrix.

MINFIT-S Compute the singular value decomposition of a rectangular matrix and solve the related linear least squares problem.

SGLSS-S
DGLSS-D
Solve a linear least squares problems by performing a QR factorization of the matrix using Householder transformations. Emphasis is put on detecting possible rank deficiency.

SQRSL-S
DQRSL-D
CQRSL-C
Apply the output of SQRDC to compute coordinate transformations, projections, and least squares solutions.

ULSIA-S
DULSIA-D
Solve an underdetermined linear system of equations by performing an LQ factorization of the matrix using Householder transformations. Emphasis is put on detecting possible rank deficiency.

E. Interpolation

BSPDOC-A Documentation for BSPLINE, a package of subprograms for working with piecewise polynomial functions in B-representation.

E1. Univariate data (curve fitting)
E1A. Polynomial splines (piecewise polynomials)

BINT4-S
DBINT4-D
Compute the B-representation of a cubic spline which interpolates given data.

BINTK-S
DBINTK-D
Compute the B-representation of a spline which interpolates given data.

BSPDOC-A Documentation for BSPLINE, a package of subprograms for working with piecewise polynomial functions in B-representation.

PCHDOC-A Documentation for PCHIP, a Fortran package for piecewise cubic Hermite interpolation of data.

PCHIC-S
DPCHIC-D
Set derivatives needed to determine a piecewise monotone piecewise cubic Hermite interpolant to given data. User control is available over boundary conditions and/or treatment of points where monotonicity switches direction.

PCHIM-S
DPCHIM-D
Set derivatives needed to determine a monotone piecewise cubic Hermite interpolant to given data. Boundary values are provided which are compatible with monotonicity. The interpolant will have an extremum at each point where monotonicity switches direction. (See PCHIC if user control is desired over boundary or switch conditions.)

PCHSP-S
DPCHSP-D
Set derivatives needed to determine the Hermite representation of the cubic spline interpolant to given data, with specified boundary conditions.

E1B. Polynomials

POLCOF-S
DPOLCF-D
Compute the coefficients of the polynomial fit (including Hermite polynomial fits) produced by a previous call to POLINT.

```
          POLINT-S  Produce the polynomial which interpolates a set of discrete
          DPLINT-D  data points.

E3.  Service routines (e.g., grid generation, evaluation of fitted functions)
     (search also class N5)

          BFQAD-S   Compute the integral of a product of a function and a
          DBFQAD-D  derivative of a B-spline.

          BSPDR-S   Use the B-representation to construct a divided difference
          DBSPDR-D  table preparatory to a (right) derivative calculation.

          BSPEV-S   Calculate the value of the spline and its derivatives from
          DBSPEV-D  the B-representation.

          BSPPP-S   Convert the B-representation of a B-spline to the piecewise
          DBSPPP-D  polynomial (PP) form.

          BSPVD-S   Calculate the value and all derivatives of order less than
          DBSPVD-D  NDERIV of all basis functions which do not vanish at X.

          BSPVN-S   Calculate the value of all (possibly) nonzero basis
          DBSPVN-D  functions at X.

          BSQAD-S   Compute the integral of a K-th order B-spline using the
          DBSQAD-D  B-representation.

          BVALU-S   Evaluate the B-representation of a B-spline at X for the
          DBVALU-D  function value or any of its derivatives.

          CHFDV-S   Evaluate a cubic polynomial given in Hermite form and its
          DCHFDV-D  first derivative at an array of points.  While designed for
                    use by PCHFD, it may be useful directly as an evaluator
                    for a piecewise cubic Hermite function in applications,
                    such as graphing, where the interval is known in advance.
                    If only function values are required, use CHFEV instead.

          CHFEV-S   Evaluate a cubic polynomial given in Hermite form at an
          DCHFEV-D  array of points.  While designed for use by PCHFE, it may
                    be useful directly as an evaluator for a piecewise cubic
                    Hermite function in applications, such as graphing, where
                    the interval is known in advance.

          INTRV-S   Compute the largest integer ILEFT in 1 .LE. ILEFT .LE. LXT
          DINTRV-D  such that XT(ILEFT) .LE. X where XT(*) is a subdivision
                    of the X interval.

          PCHBS-S   Piecewise Cubic Hermite to B-Spline converter.
          DPCHBS-D

          PCHCM-S   Check a cubic Hermite function for monotonicity.
          DPCHCM-D

          PCHFD-S   Evaluate a piecewise cubic Hermite function and its first
          DPCHFD-D  derivative at an array of points.  May be used by itself
                    for Hermite interpolation, or as an evaluator for PCHIM
                    or PCHIC.  If only function values are required, use
                    PCHFE instead.

          PCHFE-S   Evaluate a piecewise cubic Hermite function at an array of
```

```
          DPCHFE-D  points.  May be used by itself for Hermite interpolation,
                    or as an evaluator for PCHIM or PCHIC.

          PCHIA-S   Evaluate the definite integral of a piecewise cubic
          DPCHIA-D  Hermite function over an arbitrary interval.

          PCHID-S   Evaluate the definite integral of a piecewise cubic
          DPCHID-D  Hermite function over an interval whose endpoints are data
                    points.

          PFQAD-S   Compute the integral on (X1,X2) of a product of a function
          DPFQAD-D  F and the ID-th derivative of a B-spline,
                    (PP-representation).

          POLYVL-S  Calculate the value of a polynomial and its first NDER
          DPOLVL-D  derivatives where the polynomial was produced by a previous
                    call to POLINT.

          PPQAD-S   Compute the integral on (X1,X2) of a K-th order B-spline
          DPPQAD-D  using the piecewise polynomial (PP) representation.

          PPVAL-S   Calculate the value of the IDERIV-th derivative of the
          DPPVAL-D  B-spline from the PP-representation.

F.  Solution of nonlinear equations
F1.  Single equation
F1A.  Smooth
F1A1.  Polynomial
F1A1A.  Real coefficients

          RPQR79-S  Find the zeros of a polynomial with real coefficients.
          CPQR79-C

          RPZERO-S  Find the zeros of a polynomial with real coefficients.
          CPZERO-C

F1A1B.  Complex coefficients

          CPQR79-C  Find the zeros of a polynomial with complex coefficients.
          RPQR79-S

          CPZERO-C  Find the zeros of a polynomial with complex coefficients.
          RPZERO-S

F1B.  General (no smoothness assumed)

          FZERO-S   Search for a zero of a function F(X) in a given interval
          DFZERO-D  (B,C).  It is designed primarily for problems where F(B)
                    and F(C) have opposite signs.

F2.  System of equations
F2A.  Smooth

          SNSQ-S    Find a zero of a system of a N nonlinear functions in N
          DNSQ-D    variables by a modification of the Powell hybrid method.

          SNSQE-S   An easy-to-use code to find a zero of a system of N
          DNSQE-D   nonlinear functions in N variables by a modification of
                    the Powell hybrid method.

          SOS-S     Solve a square system of nonlinear equations.
```

```
          DSOS-D

F3.  Service routines (e.g., check user-supplied derivatives)

          CHKDER-S  Check the gradients of M nonlinear functions in N
          DCKDER-D  variables, evaluated at a point X, for consistency
                    with the functions themselves.

G.  Optimization (search also classes K, L8)
G2.  Constrained
G2A.  Linear programming
G2A2.  Sparse matrix of constraints

          SPLP-S    Solve linear programming problems involving at
          DSPLP-D   most a few thousand constraints and variables.
                    Takes advantage of sparsity in the constraint matrix.

G2E.  Quadratic programming

          SBOCLS-S  Solve the bounded and constrained least squares
          DBOCLS-D  problem consisting of solving the equation
                              E*X = F  (in the least squares sense)
                     subject to the linear constraints
                                    C*X = Y.

          SBOLS-S   Solve the problem
          DBOLS-D        E*X = F (in the least  squares  sense)
                    with bounds on selected X values.

G2H.  General nonlinear programming
G2H1.  Simple bounds

          SBOCLS-S  Solve the bounded and constrained least squares
          DBOCLS-D  problem consisting of solving the equation
                              E*X = F  (in the least squares sense)
                     subject to the linear constraints
                                    C*X = Y.

          SBOLS-S   Solve the problem
          DBOLS-D        E*X = F (in the least  squares  sense)
                    with bounds on selected X values.

G2H2.  Linear equality or inequality constraints

          SBOCLS-S  Solve the bounded and constrained least squares
          DBOCLS-D  problem consisting of solving the equation
                              E*X = F  (in the least squares sense)
                     subject to the linear constraints
                                    C*X = Y.

          SBOLS-S   Solve the problem
          DBOLS-D        E*X = F (in the least  squares  sense)
                    with bounds on selected X values.

G4.  Service routines
G4C.  Check user-supplied derivatives

          CHKDER-S  Check the gradients of M nonlinear functions in N
          DCKDER-D  variables, evaluated at a point X, for consistency
                    with the functions themselves.
```

H. Differentiation, integration

H1. Numerical differentiation

| | |
|---|---|
| CHFDV-S<br>DCHFDV-D | Evaluate a cubic polynomial given in Hermite form and its first derivative at an array of points. While designed for use by PCHFD, it may be useful directly as an evaluator for a piecewise cubic Hermite function in applications, such as graphing, where the interval is known in advance. If only function values are required, use CHFEV instead. |
| PCHFD-S<br>DPCHFD-D | Evaluate a piecewise cubic Hermite function and its first derivative at an array of points. May be used by itself for Hermite interpolation, or as an evaluator for PCHIM or PCHIC. If only function values are required, use PCHFE instead. |

H2. Quadrature (numerical evaluation of definite integrals)

| | |
|---|---|
| QPDOC-A | Documentation for QUADPACK, a package of subprograms for automatic evaluation of one-dimensional definite integrals. |

H2A. One-dimensional integrals

H2A1. Finite interval (general integrand)

H2A1A. Integrand available via user-defined procedure

H2A1A1. Automatic (user need only specify required accuracy)

| | |
|---|---|
| GAUS8-S<br>DGAUS8-D | Integrate a real function of one variable over a finite interval using an adaptive 8-point Legendre-Gauss algorithm. Intended primarily for high accuracy integration or integration of smooth functions. |
| QAG-S<br>DQAG-D | The routine calculates an approximation result to a given definite integral I = integral of F over (A,B), hopefully satisfying following claim for accuracy ABS(I-RESULT)LE.MAX(EPSABS,EPSREL*ABS(I)). |
| QAGE-S<br>DQAGE-D | The routine calculates an approximation result to a given definite integral   I = Integral of F over (A,B), hopefully satisfying following claim for accuracy ABS(I-RESLT).LE.MAX(EPSABS,EPSREL*ABS(I)). |
| QAGS-S<br>DQAGS-D | The routine calculates an approximation result to a given Definite integral  I = Integral of F over (A,B), Hopefully satisfying following claim for accuracy ABS(I-RESULT).LE.MAX(EPSABS,EPSREL*ABS(I)). |
| QAGSE-S<br>DQAGSE-D | The routine calculates an approximation result to a given definite integral I = Integral of F over (A,B), hopefully satisfying following claim for accuracy ABS(I-RESULT).LE.MAX(EPSABS,EPSREL*ABS(I)). |
| QNC79-S<br>DQNC79-D | Integrate a function using a 7-point adaptive Newton-Cotes quadrature rule. |
| QNG-S<br>DQNG-D | The routine calculates an approximation result to a given definite integral I = integral of F over (A,B), hopefully satisfying following claim for accuracy ABS(I-RESULT).LE.MAX(EPSABS,EPSREL*ABS(I)). |

H2A1A2. Nonautomatic

```
          QK15-S    To compute I = Integral of F over (A,B), with error
          DQK15-D                      estimate
                                 J = integral of ABS(F) over (A,B)

          QK21-S    To compute I = Integral of F over (A,B), with error
          DQK21-D                      estimate
                                 J = Integral of ABS(F) over (A,B)

          QK31-S    To compute I = Integral of F over (A,B) with error
          DQK31-D                      estimate
                                 J = Integral of ABS(F) over (A,B)

          QK41-S    To compute I = Integral of F over (A,B), with error
          DQK41-D                      estimate
                                 J = Integral of ABS(F) over (A,B)

          QK51-S    To compute I = Integral of F over (A,B) with error
          DQK51-D                      estimate
                                 J = Integral of ABS(F) over (A,B)

          QK61-S    To compute I = Integral of F over (A,B) with error
          DQK61-D                      estimate
                                 J = Integral of ABS(F) over (A,B)

H2A1B.  Integrand available only on grid
H2A1B2.  Nonautomatic

          AVINT-S   Integrate a function tabulated at arbitrarily spaced
          DAVINT-D  abscissas using overlapping parabolas.

          PCHIA-S   Evaluate the definite integral of a piecewise cubic
          DPCHIA-D  Hermite function over an arbitrary interval.

          PCHID-S   Evaluate the definite integral of a piecewise cubic
          DPCHID-D  Hermite function over an interval whose endpoints are data
                    points.

H2A2.  Finite interval (specific or special type integrand including weight
       functions, oscillating and singular integrands, principal value
       integrals, splines, etc.)
H2A2A.  Integrand available via user-defined procedure
H2A2A1.  Automatic (user need only specify required accuracy)

          BFQAD-S   Compute the integral of a product of a function and a
          DBFQAD-D  derivative of a B-spline.

          BSQAD-S   Compute the integral of a K-th order B-spline using the
          DBSQAD-D  B-representation.

          PFQAD-S   Compute the integral on (X1,X2) of a product of a function
          DPFQAD-D  F and the ID-th derivative of a B-spline,
                    (PP-representation).

          PPQAD-S   Compute the integral on (X1,X2) of a K-th order B-spline
          DPPQAD-D  using the piecewise polynomial (PP) representation.

          QAGP-S    The routine calculates an approximation result to a given
          DQAGP-D   definite integral I = Integral of F over (A,B),
                    hopefully satisfying following claim for accuracy
                    break points of the integration interval, where local
                    difficulties of the integrand may occur(e.g. SINGULARITIES,
```

```
              DISCONTINUITIES), are provided by the user.

    QAGPE-S   Approximate a given definite integral I = Integral of F
    DQAGPE-D  over (A,B), hopefully satisfying the accuracy claim:
                    ABS(I-RESULT).LE.MAX(EPSABS,EPSREL*ABS(I)).
              Break points of the integration interval, where local
              difficulties of the integrand may occur (e.g. singularities
              or discontinuities) are provided by the user.

    QAWC-S    The routine calculates an approximation result to a
    DQAWC-D   Cauchy principal value I = INTEGRAL of F*W over (A,B)
              (W(X) = 1/((X-C), C.NE.A, C.NE.B), hopefully satisfying
              following claim for accuracy
              ABS(I-RESULT).LE.MAX(EPSABE,EPSREL*ABS(I)).

    QAWCE-S   The routine calculates an approximation result to a
    DQAWCE-D  CAUCHY PRINCIPAL VALUE I = Integral of F*W over (A,B)
              (W(X) = 1/(X-C), (C.NE.A, C.NE.B), hopefully satisfying
              following claim for accuracy
              ABS(I-RESULT).LE.MAX(EPSABS,EPSREL*ABS(I))

    QAWO-S    Calculate an approximation to a given definite integral
    DQAWO-D    I = Integral of F(X)*W(X) over (A,B), where
                     W(X) = COS(OMEGA*X)
                  or W(X) = SIN(OMEGA*X),
              hopefully satisfying the following claim for accuracy
                  ABS(I-RESULT).LE.MAX(EPSABS,EPSREL*ABS(I)).

    QAWOE-S   Calculate an approximation to a given definite integral
    DQAWOE-D     I = Integral of F(X)*W(X) over (A,B), where
                    W(X) = COS(OMEGA*X)
                 or W(X) = SIN(OMEGA*X),
              hopefully satisfying the following claim for accuracy
                 ABS(I-RESULT).LE.MAX(EPSABS,EPSREL*ABS(I)).

    QAWS-S    The routine calculates an approximation result to a given
    DQAWS-D   definite integral I = Integral of F*W over (A,B),
              (where W shows a singular behaviour at the end points
              see parameter INTEGR).
              Hopefully satisfying following claim for accuracy
              ABS(I-RESULT).LE.MAX(EPSABS,EPSREL*ABS(I)).

    QAWSE-S   The routine calculates an approximation result to a given
    DQAWSE-D  definite integral I = Integral of F*W over (A,B),
              (where W shows a singular behaviour at the end points,
              see parameter INTEGR).
              Hopefully satisfying following claim for accuracy
              ABS(I-RESULT).LE.MAX(EPSABS,EPSREL*ABS(I)).

    QMOMO-S   This routine computes modified Chebyshev moments.  The K-th
    DQMOMO-D  modified Chebyshev moment is defined as the integral over
              (-1,1) of W(X)*T(K,X), where T(K,X) is the Chebyshev
              polynomial of degree K.

H2A2A2.  Nonautomatic

    QC25C-S   To compute I = Integral of F*W over (A,B) with
    DQC25C-D  error estimate, where W(X) = 1/(X-C)

    QC25F-S   To compute the integral I=Integral of F(X) over (A,B)
    DQC25F-D  Where W(X) = COS(OMEGA*X) Or (WX)=SIN(OMEGA*X)
```

```
                and to compute J=Integral of ABS(F) over (A,B). For small
                value of OMEGA or small intervals (A,B) 15-point GAUSS-
                KRONROD Rule used. Otherwise generalized CLENSHAW-CURTIS us

      QC25S-S   To compute I = Integral of F*W over (BL,BR), with error
      DQC25S-D  estimate, where the weight function W has a singular
                behaviour of ALGEBRAICO-LOGARITHMIC type at the points
                A and/or B. (BL,BR) is a part of (A,B).

      QK15W-S   To compute I = Integral of F*W over (A,B), with error
      DQK15W-D                 estimate
                           J = Integral of ABS(F*W) over (A,B)

H2A3.  Semi-infinite interval (including e**(-x) weight function)
H2A3A.  Integrand available via user-defined procedure
H2A3A1.  Automatic (user need only specify required accuracy)

      QAGI-S    The routine calculates an approximation result to a given
      DQAGI-D   INTEGRAL   I = Integral of F over (BOUND,+INFINITY)
                        OR I = Integral of F over (-INFINITY,BOUND)
                        OR I = Integral of F over (-INFINITY,+INFINITY)
                Hopefully satisfying following claim for accuracy
                ABS(I-RESULT).LE.MAX(EPSABS,EPSREL*ABS(I)).

      QAGIE-S   The routine calculates an approximation result to a given
      DQAGIE-D  integral   I = Integral of F over (BOUND,+INFINITY)
                        or I = Integral of F over (-INFINITY,BOUND)
                        or I = Integral of F over (-INFINITY,+INFINITY),
                        hopefully satisfying following claim for accuracy
                        ABS(I-RESULT).LE.MAX(EPSABS,EPSREL*ABS(I))

      QAWF-S    The routine calculates an approximation result to a given
      DQAWF-D   Fourier integral
                I = Integral of F(X)*W(X) over (A,INFINITY)
                where W(X) = COS(OMEGA*X) or W(X) = SIN(OMEGA*X).
                Hopefully satisfying following claim for accuracy
                ABS(I-RESULT).LE.EPSABS.

      QAWFE-S   The routine calculates an approximation result to a
      DQAWFE-D  given Fourier integral
                I = Integral of F(X)*W(X) over (A,INFINITY)
                 where W(X) = COS(OMEGA*X) or W(X) = SIN(OMEGA*X),
                hopefully satisfying following claim for accuracy
                ABS(I-RESULT).LE.EPSABS.

H2A3A2.  Nonautomatic

      QK15I-S   The original (infinite integration range is mapped
      DQK15I-D  onto the interval (0,1) and (A,B) is a part of (0,1).
                it is the purpose to compute
                I = Integral of transformed integrand over (A,B),
                J = Integral of ABS(Transformed Integrand) over (A,B).

H2A4.  Infinite interval (including e**(-x**2)) weight function)
H2A4A.  Integrand available via user-defined procedure
H2A4A1.  Automatic (user need only specify required accuracy)

      QAGI-S    The routine calculates an approximation result to a given
      DQAGI-D   INTEGRAL   I = Integral of F over (BOUND,+INFINITY)
                        OR I = Integral of F over (-INFINITY,BOUND)
                        OR I = Integral of F over (-INFINITY,+INFINITY)
```

```
                    Hopefully satisfying following claim for accuracy
                    ABS(I-RESULT).LE.MAX(EPSABS,EPSREL*ABS(I)).

          QAGIE-S   The routine calculates an approximation result to a given
          DQAGIE-D  integral   I = Integral of F over (BOUND,+INFINITY)
                            or I = Integral of F over (-INFINITY,BOUND)
                            or I = Integral of F over (-INFINITY,+INFINITY),
                            hopefully satisfying following claim for accuracy
                            ABS(I-RESULT).LE.MAX(EPSABS,EPSREL*ABS(I))
```

H2A4A2.  Nonautomatic

```
          QK15I-S   The original (infinite integration range is mapped
          DQK15I-D  onto the interval (0,1) and (A,B) is a part of (0,1).
                    it is the purpose to compute
                    I = Integral of transformed integrand over (A,B),
                    J = Integral of ABS(Transformed Integrand) over (A,B).
```

I.  Differential and integral equations
I1.  Ordinary differential equations
I1A.  Initial value problems
I1A1.  General, nonstiff or mildly stiff
I1A1A.  One-step methods (e.g., Runge-Kutta)

```
          DERKF-S   Solve an initial value problem in ordinary differential
          DDERKF-D  equations using a Runge-Kutta-Fehlberg scheme.
```

I1A1B.  Multistep methods (e.g., Adams’ predictor-corrector)

```
          DEABM-S   Solve an initial value problem in ordinary differential
          DDEABM-D  equations using an Adams-Bashforth method.

          SDRIV1-S  The function of SDRIV1 is to solve N (200 or fewer)
          DDRIV1-D  ordinary differential equations of the form
          CDRIV1-C  dY(I)/dT = F(Y(I),T), given the initial conditions
                    Y(I) = YI.  SDRIV1 uses single precision arithmetic.

          SDRIV2-S  The function of SDRIV2 is to solve N ordinary differential
          DDRIV2-D  equations of the form dY(I)/dT = F(Y(I),T), given the
          CDRIV2-C  initial conditions Y(I) = YI.  The program has options to
                    allow the solution of both stiff and non-stiff differential
                    equations.  SDRIV2 uses single precision arithmetic.

          SDRIV3-S  The function of SDRIV3 is to solve N ordinary differential
          DDRIV3-D  equations of the form dY(I)/dT = F(Y(I),T), given the
          CDRIV3-C  initial conditions Y(I) = YI.  The program has options to
                    allow the solution of both stiff and non-stiff differential
                    equations.  Other important options are available.  SDRIV3
                    uses single precision arithmetic.

          SINTRP-S  Approximate the solution at XOUT by evaluating the
          DINTP-D   polynomial computed in STEPS at XOUT.  Must be used in
                    conjunction with STEPS.

          STEPS-S   Integrate a system of first order ordinary differential
          DSTEPS-D  equations one step.
```

I1A2.  Stiff and mixed algebraic-differential equations

```
          DEBDF-S   Solve an initial value problem in ordinary differential
          DDEBDF-D  equations using backward differentiation formulas.  It is
```

intended primarily for stiff problems.

SDASSL-S DDASSL-D  This code solves a system of differential/algebraic equations of the form G(T,Y,YPRIME) = 0.

SDRIV1-S DDRIV1-D CDRIV1-C  The function of SDRIV1 is to solve N (200 or fewer) ordinary differential equations of the form dY(I)/dT = F(Y(I),T), given the initial conditions Y(I) = YI.  SDRIV1 uses single precision arithmetic.

SDRIV2-S DDRIV2-D CDRIV2-C  The function of SDRIV2 is to solve N ordinary differential equations of the form dY(I)/dT = F(Y(I),T), given the initial conditions Y(I) = YI.  The program has options to allow the solution of both stiff and non-stiff differential equations.  SDRIV2 uses single precision arithmetic.

SDRIV3-S DDRIV3-D CDRIV3-C  The function of SDRIV3 is to solve N ordinary differential equations of the form dY(I)/dT = F(Y(I),T), given the initial conditions Y(I) = YI.  The program has options to allow the solution of both stiff and non-stiff differential equations.  Other important options are available.  SDRIV3 uses single precision arithmetic.

I1B.  Multipoint boundary value problems
I1B1.  Linear

BVSUP-S DBVSUP-D  Solve a linear two-point boundary value problem using superposition coupled with an orthonormalization procedure and a variable-step integration scheme.

I2.  Partial differential equations
I2B.  Elliptic boundary value problems
I2B1.  Linear
I2B1A.  Second order
I2B1A1.  Poisson (Laplace) or Helmholz equation
I2B1A1A.  Rectangular domain (or topologically rectangular in the coordinate system)

HSTCRT-S  Solve the standard five-point finite difference approximation on a staggered grid to the Helmholtz equation in Cartesian coordinates.

HSTCSP-S  Solve the standard five-point finite difference approximation on a staggered grid to the modified Helmholtz equation in spherical coordinates assuming axisymmetry (no dependence on longitude).

HSTCYL-S  Solve the standard five-point finite difference approximation on a staggered grid to the modified Helmholtz equation in cylindrical coordinates.

HSTPLR-S  Solve the standard five-point finite difference approximation on a staggered grid to the Helmholtz equation in polar coordinates.

HSTSSP-S  Solve the standard five-point finite difference approximation on a staggered grid to the Helmholtz equation in spherical coordinates and on the surface of the unit sphere (radius of 1).

HW3CRT-S  Solve the standard seven-point finite difference approximation to the Helmholtz equation in Cartesian coordinates.

HWSCRT-S Solves the standard five-point finite difference approximation to the Helmholtz equation in Cartesian coordinates.

HWSCSP-S Solve a finite difference approximation to the modified Helmholtz equation in spherical coordinates assuming axisymmetry (no dependence on longitude).

HWSCYL-S Solve a standard finite difference approximation to the Helmholtz equation in cylindrical coordinates.

HWSPLR-S Solve a finite difference approximation to the Helmholtz equation in polar coordinates.

HWSSSP-S Solve a finite difference approximation to the Helmholtz equation in spherical coordinates and on the surface of the unit sphere (radius of 1).

I2B1A2. Other separable problems

SEPELI-S Discretize and solve a second and, optionally, a fourth order finite difference approximation on a uniform grid to the general separable elliptic partial differential equation on a rectangle with any combination of periodic or mixed boundary conditions.

SEPX4-S Solve for either the second or fourth order finite difference approximation to the solution of a separable elliptic partial differential equation on a rectangle. Any combination of periodic or mixed boundary conditions is allowed.

I2B4. Service routines
I2B4B. Solution of discretized elliptic equations

BLKTRI-S
CBLKTR-C Solve a block tridiagonal system of linear equations (usually resulting from the discretization of separable two-dimensional elliptic equations).

GENBUN-S
CMGNBN-C Solve by a cyclic reduction algorithm the linear system of equations that results from a finite difference approximation to certain 2-d elliptic PDE's on a centered grid .

POIS3D-S Solve a three-dimensional block tridiagonal linear system which arises from a finite difference approximation to a three-dimensional Poisson equation using the Fourier transform package FFTPAK written by Paul Swarztrauber.

POISTG-S Solve a block tridiagonal system of linear equations that results from a staggered grid finite difference approximation to 2-D elliptic PDE's.

J. Integral transforms
J1. Fast Fourier transforms (search class L10 for time series analysis)

FFTDOC-A Documentation for FFTPACK, a collection of Fast Fourier Transform routines.

J1A. One-dimensional
J1A1. Real

| | |
|---|---|
| EZFFTB-S | A simplified real, periodic, backward fast Fourier transform. |
| EZFFTF-S | Compute a simplified real, periodic, fast Fourier forward transform. |
| EZFFTI-S | Initialize a work array for EZFFTF and EZFFTB. |
| RFFTB1-S<br>CFFTB1-C | Compute the backward fast Fourier transform of a real coefficient array. |
| RFFTF1-S<br>CFFTF1-C | Compute the forward transform of a real, periodic sequence. |
| RFFTI1-S<br>CFFTI1-C | Initialize a real and an integer work array for RFFTF1 and RFFTB1. |

J1A2. Complex

| | |
|---|---|
| CFFTB1-C<br>RFFTB1-S | Compute the unnormalized inverse of CFFTF1. |
| CFFTF1-C<br>RFFTF1-S | Compute the forward transform of a complex, periodic sequence. |
| CFFTI1-C<br>RFFTI1-S | Initialize a real and an integer work array for CFFTF1 and CFFTB1. |

J1A3. Trigonometric (sine, cosine)

| | |
|---|---|
| COSQB-S | Compute the unnormalized inverse cosine transform. |
| COSQF-S | Compute the forward cosine transform with odd wave numbers. |
| COSQI-S | Initialize a work array for COSQF and COSQB. |
| COST-S | Compute the cosine transform of a real, even sequence. |
| COSTI-S | Initialize a work array for COST. |
| SINQB-S | Compute the unnormalized inverse of SINQF. |
| SINQF-S | Compute the forward sine transform with odd wave numbers. |
| SINQI-S | Initialize a work array for SINQF and SINQB. |
| SINT-S | Compute the sine transform of a real, odd sequence. |
| SINTI-S | Initialize a work array for SINT. |

J4. Hilbert transforms

| | |
|---|---|
| QAWC-S<br>DQAWC-D | The routine calculates an approximation result to a Cauchy principal value I = INTEGRAL of F*W over (A,B) (W(X) = 1/((X-C), C.NE.A, C.NE.B), hopefully satisfying following claim for accuracy ABS(I-RESULT).LE.MAX(EPSABE,EPSREL*ABS(I)). |

```
          QAWCE-S   The routine calculates an approximation result to a
          DQAWCE-D  CAUCHY PRINCIPAL VALUE I = Integral of F*W over (A,B)
                    (W(X) = 1/(X-C), (C.NE.A, C.NE.B), hopefully satisfying
                    following claim for accuracy
                    ABS(I-RESULT).LE.MAX(EPSABS,EPSREL*ABS(I))

          QC25C-S   To compute I = Integral of F*W over (A,B) with
          DQC25C-D  error estimate, where W(X) = 1/(X-C)

K.  Approximation (search also class L8)

          BSPDOC-A  Documentation for BSPLINE, a package of subprograms for
                    working with piecewise polynomial functions
                    in B-representation.

K1.  Least squares (L-2) approximation
K1A.  Linear least squares (search also classes D5, D6, D9)
K1A1.  Unconstrained
K1A1A.  Univariate data (curve fitting)
K1A1A1.  Polynomial splines (piecewise polynomials)

          EFC-S     Fit a piecewise polynomial curve to discrete data.
          DEFC-D    The piecewise polynomials are represented as B-splines.
                    The fitting is done in a weighted least squares sense.

          FC-S      Fit a piecewise polynomial curve to discrete data.
          DFC-D     The piecewise polynomials are represented as B-splines.
                    The fitting is done in a weighted least squares sense.
                    Equality and inequality constraints can be imposed on the
                    fitted curve.

K1A1A2.  Polynomials

          PCOEF-S   Convert the POLFIT coefficients to Taylor series form.
          DPCOEF-D

          POLFIT-S  Fit discrete data in a least squares sense by polynomials
          DPOLFT-D  in one variable.

K1A2.  Constrained
K1A2A.  Linear constraints

          EFC-S     Fit a piecewise polynomial curve to discrete data.
          DEFC-D    The piecewise polynomials are represented as B-splines.
                    The fitting is done in a weighted least squares sense.

          FC-S      Fit a piecewise polynomial curve to discrete data.
          DFC-D     The piecewise polynomials are represented as B-splines.
                    The fitting is done in a weighted least squares sense.
                    Equality and inequality constraints can be imposed on the
                    fitted curve.

          LSEI-S    Solve a linearly constrained least squares problem with
          DLSEI-D   equality and inequality constraints, and optionally compute
                    a covariance matrix.

          SBOCLS-S  Solve the bounded and constrained least squares
          DBOCLS-D  problem consisting of solving the equation
                              E*X = F  (in the least squares sense)
                     subject to the linear constraints
```

```
                                   C*X = Y.

          SBOLS-S   Solve the problem
          DBOLS-D        E*X = F (in the least  squares  sense)
                    with bounds on selected X values.

          WNNLS-S   Solve a linearly constrained least squares problem with
          DWNNLS-D  equality constraints and nonnegativity constraints on
                    selected variables.

K1B.  Nonlinear least squares
K1B1.  Unconstrained

          SCOV-S    Calculate the covariance matrix for a nonlinear data
          DCOV-D    fitting problem.  It is intended to be used after a
                    successful return from either SNLS1 or SNLS1E.

K1B1A.  Smooth functions
K1B1A1.  User provides no derivatives

          SNLS1-S   Minimize the sum of the squares of M nonlinear functions
          DNLS1-D   in N variables by a modification of the Levenberg-Marquardt
                    algorithm.

          SNLS1E-S  An easy-to-use code which minimizes the sum of the squares
          DNLS1E-D  of M nonlinear functions in N variables by a modification
                    of the Levenberg-Marquardt algorithm.

K1B1A2.  User provides first derivatives

          SNLS1-S   Minimize the sum of the squares of M nonlinear functions
          DNLS1-D   in N variables by a modification of the Levenberg-Marquardt
                    algorithm.

          SNLS1E-S  An easy-to-use code which minimizes the sum of the squares
          DNLS1E-D  of M nonlinear functions in N variables by a modification
                    of the Levenberg-Marquardt algorithm.

K6.  Service routines (e.g., mesh generation, evaluation of fitted functions)
     (search also class N5)

          BFQAD-S   Compute the integral of a product of a function and a
          DBFQAD-D  derivative of a B-spline.

          DBSPDR-D  Use the B-representation to construct a divided difference
          BSPDR-S   table preparatory to a (right) derivative calculation.

          BSPEV-S   Calculate the value of the spline and its derivatives from
          DBSPEV-D  the B-representation.

          BSPPP-S   Convert the B-representation of a B-spline to the piecewise
          DBSPPP-D  polynomial (PP) form.

          BSPVD-S   Calculate the value and all derivatives of order less than
          DBSPVD-D  NDERIV of all basis functions which do not vanish at X.

          BSPVN-S   Calculate the value of all (possibly) nonzero basis
          DBSPVN-D  functions at X.

          BSQAD-S   Compute the integral of a K-th order B-spline using the
          DBSQAD-D  B-representation.
```

```
          BVALU-S   Evaluate the B-representation of a B-spline at X for the
          DBVALU-D  function value or any of its derivatives.

          INTRV-S   Compute the largest integer ILEFT in 1 .LE. ILEFT .LE. LXT
          DINTRV-D  such that XT(ILEFT) .LE. X where XT(*) is a subdivision
                    of the X interval.

          PFQAD-S   Compute the integral on (X1,X2) of a product of a function
          DPFQAD-D  F and the ID-th derivative of a B-spline,
                    (PP-representation).

          PPQAD-S   Compute the integral on (X1,X2) of a K-th order B-spline
          DPPQAD-D  using the piecewise polynomial (PP) representation.

          PPVAL-S   Calculate the value of the IDERIV-th derivative of the
          DPPVAL-D  B-spline from the PP-representation.

          PVALUE-S  Use the coefficients generated by POLFIT to evaluate the
          DP1VLU-D  polynomial fit of degree L, along with the first NDER of
                    its derivatives, at a specified point.

L.  Statistics, probability
L5.  Function evaluation (search also class C)
L5A.  Univariate
L5A1.  Cumulative distribution functions, probability density functions
L5A1E.  Error function, exponential, extreme value

          ERF-S     Compute the error function.
          DERF-D

          ERFC-S    Compute the complementary error function.
          DERFC-D

L6.  Pseudo-random number generation
L6A.  Univariate
L6A14.  Negative binomial, normal

          RGAUSS-S  Generate a normally distributed (Gaussian) random number.

L6A21.  Uniform

          RAND-S    Generate a uniformly distributed random number.

          RUNIF-S   Generate a uniformly distributed random number.

L7.  Experimental design, including analysis of variance
L7A.  Univariate
L7A3.  Analysis of covariance

          CV-S      Evaluate the variance function of the curve obtained
          DCV-D     by the constrained B-spline fitting subprogram FC.

L8.  Regression (search also classes G, K)
L8A.  Linear least squares (L-2) (search also classes D5, D6, D9)
L8A3.  Piecewise polynomial (i.e. multiphase or spline)

          EFC-S     Fit a piecewise polynomial curve to discrete data.
          DEFC-D    The piecewise polynomials are represented as B-splines.
                    The fitting is done in a weighted least squares sense.
```

```
          FC-S      Fit a piecewise polynomial curve to discrete data.
          DFC-D     The piecewise polynomials are represented as B-splines.
                    The fitting is done in a weighted least squares sense.
                    Equality and inequality constraints can be imposed on the
                    fitted curve.

N.  Data handling (search also class L2)
N1.  Input, output

          SBHIN-S   Read a Sparse Linear System in the Boeing/Harwell Format.
          DBHIN-D   The matrix is read in and if the right hand side is also
                    present in the input file then it too is read in.  The
                    matrix is then modified to be in the SLAP Column format.

          SCPPLT-S  Printer Plot of SLAP Column Format Matrix.
          DCPPLT-D  Routine to print out a SLAP Column format matrix in a
                    "printer plot" graphical representation.

          STIN-S    Read in SLAP Triad Format Linear System.
          DTIN-D    Routine to read in a SLAP Triad format matrix and right
                    hand side and solution to the system, if known.

          STOUT-S   Write out SLAP Triad Format Linear System.
          DTOUT-D   Routine to write out a SLAP Triad format matrix and right
                    hand side and solution to the system, if known.

N6.  Sorting
N6A.  Internal
N6A1.  Passive (i.e. construct pointer array, rank)
N6A1A.  Integer

          IPSORT-I  Return the permutation vector generated by sorting a given
          SPSORT-S  array and, optionally, rearrange the elements of the array.
          DPSORT-D  The array may be sorted in increasing or decreasing order.
          HPSORT-H  A slightly modified quicksort algorithm is used.

N6A1B.  Real

          SPSORT-S  Return the permutation vector generated by sorting a given
          DPSORT-D  array and, optionally, rearrange the elements of the array.
          IPSORT-I  The array may be sorted in increasing or decreasing order.
          HPSORT-H  A slightly modified quicksort algorithm is used.

N6A1C.  Character

          HPSORT-H  Return the permutation vector generated by sorting a
          SPSORT-S  substring within a character array and, optionally,
          DPSORT-D  rearrange the elements of the array.  The array may be
          IPSORT-I  sorted in forward or reverse lexicographical order.  A
                    slightly modified quicksort algorithm is used.

N6A2.  Active
N6A2A.  Integer

          IPSORT-I  Return the permutation vector generated by sorting a given
          SPSORT-S  array and, optionally, rearrange the elements of the array.
          DPSORT-D  The array may be sorted in increasing or decreasing order.
          HPSORT-H  A slightly modified quicksort algorithm is used.

          ISORT-I   Sort an array and optionally make the same interchanges in
          SSORT-S   an auxiliary array.  The array may be sorted in increasing
```

```
          DSORT-D   or decreasing order.  A slightly modified QUICKSORT
                    algorithm is used.

N6A2B.  Real

          SPSORT-S  Return the permutation vector generated by sorting a given
          DPSORT-D  array and, optionally, rearrange the elements of the array.
          IPSORT-I  The array may be sorted in increasing or decreasing order.
          HPSORT-H  A slightly modified quicksort algorithm is used.

          SSORT-S   Sort an array and optionally make the same interchanges in
          DSORT-D   an auxiliary array.  The array may be sorted in increasing
          ISORT-I   or decreasing order.  A slightly modified QUICKSORT
                    algorithm is used.

N6A2C.  Character

          HPSORT-H  Return the permutation vector generated by sorting a
          SPSORT-S  substring within a character array and, optionally,
          DPSORT-D  rearrange the elements of the array.  The array may be
          IPSORT-I  sorted in forward or reverse lexicographical order.  A
                    slightly modified quicksort algorithm is used.

N8.  Permuting

          SPPERM-S  Rearrange a given array according to a prescribed
          DPPERM-D  permutation vector.
          IPPERM-I
          HPPERM-H

R.  Service routines
R1.  Machine-dependent constants

          I1MACH-I  Return integer machine dependent constants.

          R1MACH-S  Return floating point machine dependent constants.
          D1MACH-D

R2.  Error checking (e.g., check monotonicity)

          GAMLIM-S  Compute the minimum and maximum bounds for the argument in
          DGAMLM-D  the Gamma function.

R3.  Error handling

          FDUMP-A   Symbolic dump (should be locally written).

R3A.  Set criteria for fatal errors

          XSETF-A   Set the error control flag.

R3B.  Set unit number for error messages

          XSETUA-A  Set logical unit numbers (up to 5) to which error
                    messages are sent.

          XSETUN-A  Set output file to which error messages are to be sent.

R3C.  Other utility programs

          NUMXER-I  Return the most recent error number.
```

XERCLR-A  Reset current error number to zero.

XERDMP-A  Print the error tables and then clear them.

XERMAX-A  Set maximum number of times any error message is to be printed.

XERMSG-A  Process error messages for SLATEC and other libraries.

XGETF-A  Return the current value of the error control flag.

XGETUA-A  Return unit number(s) to which error messages are being sent.

XGETUN-A  Return the (first) output file to which error messages are being sent.

Z. Other

AAAAAA-A  SLATEC Common Mathematical Library disclaimer and version.

BSPDOC-A  Documentation for BSPLINE, a package of subprograms for working with piecewise polynomial functions in B-representation.

EISDOC-A  Documentation for EISPACK, a collection of subprograms for solving matrix eigen-problems.

FFTDOC-A  Documentation for FFTPACK, a collection of Fast Fourier Transform routines.

FUNDOC-A  Documentation for FNLIB, a collection of routines for evaluating elementary and special functions.

PCHDOC-A  Documentation for PCHIP, a Fortran package for piecewise cubic Hermite interpolation of data.

QPDOC-A  Documentation for QUADPACK, a package of subprograms for automatic evaluation of one-dimensional definite integrals.

SLPDOC-S  Sparse Linear Algebra Package Version 2.0.2 Documentation.
DLPDOC-D  Routines to solve large sparse symmetric and nonsymmetric positive definite linear systems, Ax = b, using preconditioned iterative methods.

SECTION II. Subsidiary Routines

ASYIK  Subsidiary to BESI and BESK

ASYJY  Subsidiary to BESJ and BESY

BCRH  Subsidiary to CBLKTR

BDIFF  Subsidiary to BSKIN

BESKNU  Subsidiary to BESK

BESYNU  Subsidiary to BESY

| | |
|---|---|
| BKIAS | Subsidiary to BSKIN |
| BKISR | Subsidiary to BSKIN |
| BKSOL | Subsidiary to BVSUP |
| BLKTR1 | Subsidiary to BLKTRI |
| BNFAC | Subsidiary to BINT4 and BINTK |
| BNSLV | Subsidiary to BINT4 and BINTK |
| BSGQ8 | Subsidiary to BFQAD |
| BSPLVD | Subsidiary to FC |
| BSPLVN | Subsidiary to FC |
| BSRH | Subsidiary to BLKTRI |
| BVDER | Subsidiary to BVSUP |
| BVPOR | Subsidiary to BVSUP |
| C1MERG | Merge two strings of complex numbers. Each string is ascending by the real part. |
| C9LGMC | Compute the log gamma correction factor so that LOG(CGAMMA(Z)) = 0.5*LOG(2.*PI) + (Z-0.5)*LOG(Z) - Z + C9LGMC(Z). |
| C9LN2R | Evaluate LOG(1+Z) from second order relative accuracy so that LOG(1+Z) = Z - Z**2/2 + Z**3*C9LN2R(Z). |
| CACAI | Subsidiary to CAIRY |
| CACON | Subsidiary to CBESH and CBESK |
| CASYI | Subsidiary to CBESI and CBESK |
| CBINU | Subsidiary to CAIRY, CBESH, CBESI, CBESJ, CBESK and CBIRY |
| CBKNU | Subsidiary to CAIRY, CBESH, CBESI and CBESK |
| CBLKT1 | Subsidiary to CBLKTR |
| CBUNI | Subsidiary to CBESI and CBESK |
| CBUNK | Subsidiary to CBESH and CBESK |
| CCMPB | Subsidiary to CBLKTR |
| CDCOR | Subroutine CDCOR computes corrections to the Y array. |
| CDCST | CDCST sets coefficients used by the core integrator CDSTP. |
| CDIV | Compute the complex quotient of two complex numbers. |
| CDNTL | Subroutine CDNTL is called to set parameters on the first call to CDSTP, on an internal restart, or when the user has altered MINT, MITER, and/or H. |

| | |
|---|---|
| CDNTP | Subroutine CDNTP interpolates the K-th derivative of Y at TOUT, using the data in the YH array. If K has a value greater than NQ, the NQ-th derivative is calculated. |
| CDPSC | Subroutine CDPSC computes the predicted YH values by effectively multiplying the YH array by the Pascal triangle matrix when KSGN is +1, and performs the inverse function when KSGN is -1. |
| CDPST | Subroutine CDPST evaluates the Jacobian matrix of the right hand side of the differential equations. |
| CDSCL | Subroutine CDSCL rescales the YH array whenever the step size is changed. |
| CDSTP | CDSTP performs one step of the integration of an initial value problem for a system of ordinary differential equations. |
| CDZRO | CDZRO searches for a zero of a function F(N, T, Y, IROOT) between the given values B and C until the width of the interval (B, C) has collapsed to within a tolerance specified by the stopping criterion, ABS(B - C) .LE. 2.*(RW*ABS(B) + AE). |
| CFFTB | Compute the unnormalized inverse of CFFTF. |
| CFFTF | Compute the forward transform of a complex, periodic sequence. |
| CFFTI | Initialize a work array for CFFTF and CFFTB. |
| CFOD | Subsidiary to DEBDF |
| CHFCM | Check a single cubic for monotonicity. |
| CHFIE | Evaluates integral of a single cubic for PCHIA |
| CHKPR4 | Subsidiary to SEPX4 |
| CHKPRM | Subsidiary to SEPELI |
| CHKSN4 | Subsidiary to SEPX4 |
| CHKSNG | Subsidiary to SEPELI |
| CKSCL | Subsidiary to CBKNU, CUNK1 and CUNK2 |
| CMLRI | Subsidiary to CBESI and CBESK |
| CMPCSG | Subsidiary to CMGNBN |
| CMPOSD | Subsidiary to CMGNBN |
| CMPOSN | Subsidiary to CMGNBN |
| CMPOSP | Subsidiary to CMGNBN |
| CMPTR3 | Subsidiary to CMGNBN |

| | |
|---|---|
| CMPTRX | Subsidiary to CMGNBN |
| COMPB | Subsidiary to BLKTRI |
| COSGEN | Subsidiary to GENBUN |
| COSQB1 | Compute the unnormalized inverse of COSQF1. |
| COSQF1 | Compute the forward cosine transform with odd wave numbers. |
| CPADD | Subsidiary to CBLKTR |
| CPEVL | Subsidiary to CPZERO |
| CPEVLR | Subsidiary to CPZERO |
| CPROC | Subsidiary to CBLKTR |
| CPROCP | Subsidiary to CBLKTR |
| CPROD | Subsidiary to BLKTRI |
| CPRODP | Subsidiary to BLKTRI |
| CRATI | Subsidiary to CBESH, CBESI and CBESK |
| CS1S2 | Subsidiary to CAIRY and CBESK |
| CSCALE | Subsidiary to BVSUP |
| CSERI | Subsidiary to CBESI and CBESK |
| CSHCH | Subsidiary to CBESH and CBESK |
| CSROOT | Compute the complex square root of a complex number. |
| CUCHK | Subsidiary to SERI, CUOIK, CUNK1, CUNK2, CUNI1, CUNI2 and CKSCL |
| CUNHJ | Subsidiary to CBESI and CBESK |
| CUNI1 | Subsidiary to CBESI and CBESK |
| CUNI2 | Subsidiary to CBESI and CBESK |
| CUNIK | Subsidiary to CBESI and CBESK |
| CUNK1 | Subsidiary to CBESK |
| CUNK2 | Subsidiary to CBESK |
| CUOIK | Subsidiary to CBESH, CBESI and CBESK |
| CWRSK | Subsidiary to CBESI and CBESK |
| D1MERG | Merge two strings of ascending double precision numbers. |
| D1MPYQ | Subsidiary to DNSQ and DNSQE |
| D1UPDT | Subsidiary to DNSQ and DNSQE |

D9AIMP Evaluate the Airy modulus and phase.

D9ATN1 Evaluate DATAN(X) from first order relative accuracy so that DATAN(X) = X + X**3*D9ATN1(X).

D9B0MP Evaluate the modulus and phase for the J0 and Y0 Bessel functions.

D9B1MP Evaluate the modulus and phase for the J1 and Y1 Bessel functions.

D9CHU Evaluate for large Z Z**A * U(A,B,Z) where U is the logarithmic confluent hypergeometric function.

D9GMIC Compute the complementary incomplete Gamma function for A near a negative integer and X small.

D9GMIT Compute Tricomi's incomplete Gamma function for small arguments.

D9KNUS Compute Bessel functions EXP(X)*K-SUB-XNU(X) and EXP(X)* K-SUB-XNU+1(X) for 0.0 .LE. XNU .LT. 1.0.

D9LGIC Compute the log complementary incomplete Gamma function for large X and for A .LE. X.

D9LGIT Compute the logarithm of Tricomi's incomplete Gamma function with Perron's continued fraction for large X and A .GE. X.

D9LGMC Compute the log Gamma correction factor so that LOG(DGAMMA(X)) = LOG(SQRT(2*PI)) + (X-5.)*LOG(X) - X + D9LGMC(X).

D9LN2R Evaluate LOG(1+X) from second order relative accuracy so that LOG(1+X) = X - X**2/2 + X**3*D9LN2R(X)

DASYIK Subsidiary to DBESI and DBESK

DASYJY Subsidiary to DBESJ and DBESY

DBDIFF Subsidiary to DBSKIN

DBKIAS Subsidiary to DBSKIN

DBKISR Subsidiary to DBSKIN

DBKSOL Subsidiary to DBVSUP

DBNFAC Subsidiary to DBINT4 and DBINTK

DBNSLV Subsidiary to DBINT4 and DBINTK

DBOLSM Subsidiary to DBOCLS and DBOLS

DBSGQ8 Subsidiary to DBFQAD

DBSKNU Subsidiary to DBESK

DBSYNU Subsidiary to DBESY

DBVDER    Subsidiary to DBVSUP

DBVPOR    Subsidiary to DBVSUP

DCFOD    Subsidiary to DDEBDF

DCHFCM    Check a single cubic for monotonicity.

DCHFIE    Evaluates integral of a single cubic for DPCHIA

DCHKW    SLAP WORK/IWORK Array Bounds Checker.
This routine checks the work array lengths and interfaces to the SLATEC error handler if a problem is found.

DCOEF    Subsidiary to DBVSUP

DCSCAL    Subsidiary to DBVSUP and DSUDS

DDAINI    Initialization routine for DDASSL.

DDAJAC    Compute the iteration matrix for DDASSL and form the LU-decomposition.

DDANRM    Compute vector norm for DDASSL.

DDASLV    Linear system solver for DDASSL.

DDASTP    Perform one step of the DDASSL integration.

DDATRP    Interpolation routine for DDASSL.

DDAWTS    Set error weight vector for DDASSL.

DDCOR    Subroutine DDCOR computes corrections to the Y array.

DDCST    DDCST sets coefficients used by the core integrator DDSTP.

DDES    Subsidiary to DDEABM

DDNTL    Subroutine DDNTL is called to set parameters on the first call to DDSTP, on an internal restart, or when the user has altered MINT, MITER, and/or H.

DDNTP    Subroutine DDNTP interpolates the K-th derivative of Y at TOUT, using the data in the YH array.  If K has a value greater than NQ, the NQ-th derivative is calculated.

DDOGLG    Subsidiary to DNSQ and DNSQE

DDPSC    Subroutine DDPSC computes the predicted YH values by effectively multiplying the YH array by the Pascal triangle matrix when KSGN is +1, and performs the inverse function when KSGN is -1.

DDPST    Subroutine DDPST evaluates the Jacobian matrix of the right hand side of the differential equations.

DDSCL    Subroutine DDSCL rescales the YH array whenever the step size is changed.

DDSTP    DDSTP performs one step of the integration of an initial value problem for a system of ordinary differential equations.

| | |
|---|---|
| DDZRO | DDZRO searches for a zero of a function F(N, T, Y, IROOT) between the given values B and C until the width of the interval (B, C) has collapsed to within a tolerance specified by the stopping criterion, ABS(B - C) .LE. 2.*(RW*ABS(B) + AE). |
| DEFCMN | Subsidiary to DEFC |
| DEFE4 | Subsidiary to SEPX4 |
| DEFEHL | Subsidiary to DERKF |
| DEFER | Subsidiary to SEPELI |
| DENORM | Subsidiary to DNSQ and DNSQE |
| DERKFS | Subsidiary to DERKF |
| DES | Subsidiary to DEABM |
| DEXBVP | Subsidiary to DBVSUP |
| DFCMN | Subsidiary to FC |
| DFDJC1 | Subsidiary to DNSQ and DNSQE |
| DFDJC3 | Subsidiary to DNLS1 and DNLS1E |
| DFEHL | Subsidiary to DDERKF |
| DFSPVD | Subsidiary to DFC |
| DFSPVN | Subsidiary to DFC |
| DFULMT | Subsidiary to DSPLP |
| DGAMLN | Compute the logarithm of the Gamma function |
| DGAMRN | Subsidiary to DBSKIN |
| DH12 | Subsidiary to DHFTI, DLSEI and DWNNLS |
| DHELS | Internal routine for DGMRES. |
| DHEQR | Internal routine for DGMRES. |
| DHKSEQ | Subsidiary to DBSKIN |
| DHSTRT | Subsidiary to DDEABM, DDEBDF and DDERKF |
| DHVNRM | Subsidiary to DDEABM, DDEBDF and DDERKF |
| DINTYD | Subsidiary to DDEBDF |
| DJAIRY | Subsidiary to DBESJ and DBESY |
| DLPDP | Subsidiary to DLSEI |

| | |
|---|---|
| DLSI | Subsidiary to DLSEI |
| DLSOD | Subsidiary to DDEBDF |
| DLSSUD | Subsidiary to DBVSUP and DSUDS |
| DMACON | Subsidiary to DBVSUP |
| DMGSBV | Subsidiary to DBVSUP |
| DMOUT | Subsidiary to DBOCLS and DFC |
| DMPAR | Subsidiary to DNLS1 and DNLS1E |
| DOGLEG | Subsidiary to SNSQ and SNSQE |
| DOHTRL | Subsidiary to DBVSUP and DSUDS |
| DORTH | Internal routine for DGMRES. |
| DORTHR | Subsidiary to DBVSUP and DSUDS |
| DPCHCE | Set boundary conditions for DPCHIC |
| DPCHCI | Set interior derivatives for DPCHIC |
| DPCHCS | Adjusts derivative values for DPCHIC |
| DPCHDF | Computes divided differences for DPCHCE and DPCHSP |
| DPCHKT | Compute B-spline knot sequence for DPCHBS. |
| DPCHNG | Subsidiary to DSPLP |
| DPCHST | DPCHIP Sign-Testing Routine |
| DPCHSW | Limits excursion from data for DPCHCS |
| DPIGMR | Internal routine for DGMRES. |
| DPINCW | Subsidiary to DSPLP |
| DPINIT | Subsidiary to DSPLP |
| DPINTM | Subsidiary to DSPLP |
| DPJAC | Subsidiary to DDEBDF |
| DPLPCE | Subsidiary to DSPLP |
| DPLPDM | Subsidiary to DSPLP |
| DPLPFE | Subsidiary to DSPLP |
| DPLPFL | Subsidiary to DSPLP |
| DPLPMN | Subsidiary to DSPLP |
| DPLPMU | Subsidiary to DSPLP |
| DPLPUP | Subsidiary to DSPLP |

DPNNZR   Subsidiary to DSPLP

DPOPT   Subsidiary to DSPLP

DPPGQ8   Subsidiary to DPFQAD

DPRVEC   Subsidiary to DBVSUP

DPRWPG   Subsidiary to DSPLP

DPRWVR   Subsidiary to DSPLP

DPSIXN   Subsidiary to DEXINT

DQCHEB   This routine computes the CHEBYSHEV series expansion of degrees 12 and 24 of a function using A FAST FOURIER TRANSFORM METHOD
F(X) = SUM(K=1,..,13) (CHEB12(K)*T(K-1,X)),
F(X) = SUM(K=1,..,25) (CHEB24(K)*T(K-1,X)),
Where T(K,X) is the CHEBYSHEV POLYNOMIAL OF DEGREE K.

DQELG   The routine determines the limit of a given sequence of approximations, by means of the Epsilon algorithm of P.Wynn. An estimate of the absolute error is also given. The condensed Epsilon table is computed. Only those elements needed for the computation of the next diagonal are preserved.

DQFORM   Subsidiary to DNSQ and DNSQE

DQPSRT   This routine maintains the descending ordering in the list of the local error estimated resulting from the interval subdivision process. At each call two error estimates are inserted using the sequential search method, top-down for the largest error estimate and bottom-up for the smallest error estimate.

DQRFAC   Subsidiary to DNLS1, DNLS1E, DNSQ and DNSQE

DQRSLV   Subsidiary to DNLS1 and DNLS1E

DQWGTC   This function subprogram is used together with the routine DQAWC and defines the WEIGHT function.

DQWGTF   This function subprogram is used together with the routine DQAWF and defines the WEIGHT function.

DQWGTS   This function subprogram is used together with the routine DQAWS and defines the WEIGHT function.

DREADP   Subsidiary to DSPLP

DREORT   Subsidiary to DBVSUP

DRKFAB   Subsidiary to DBVSUP

DRKFS   Subsidiary to DDERKF

DRLCAL   Internal routine for DGMRES.

| | |
|---|---|
| DRSCO | Subsidiary to DDEBDF |
| DSLVS | Subsidiary to DDEBDF |
| DSOSEQ | Subsidiary to DSOS |
| DSOSSL | Subsidiary to DSOS |
| DSTOD | Subsidiary to DDEBDF |
| DSTOR1 | Subsidiary to DBVSUP |
| DSTWAY | Subsidiary to DBVSUP |
| DSUDS | Subsidiary to DBVSUP |
| DSVCO | Subsidiary to DDEBDF |
| DU11LS | Subsidiary to DLLSIA |
| DU11US | Subsidiary to DULSIA |
| DU12LS | Subsidiary to DLLSIA |
| DU12US | Subsidiary to DULSIA |
| DUSRMT | Subsidiary to DSPLP |
| DVECS | Subsidiary to DBVSUP |
| DVNRMS | Subsidiary to DDEBDF |
| DVOUT | Subsidiary to DSPLP |
| DWNLIT | Subsidiary to DWNNLS |
| DWNLSM | Subsidiary to DWNNLS |
| DWNLT1 | Subsidiary to WNLIT |
| DWNLT2 | Subsidiary to WNLIT |
| DWNLT3 | Subsidiary to WNLIT |
| DWRITP | Subsidiary to DSPLP |
| DWUPDT | Subsidiary to DNLS1 and DNLS1E |
| DX | Subsidiary to SEPELI |
| DX4 | Subsidiary to SEPX4 |
| DXLCAL | Internal routine for DGMRES. |
| DXPMU | To compute the values of Legendre functions for DXLEGF. Method: backward mu-wise recurrence for P(-MU,NU,X) for fixed nu to obtain P(-MU2,NU1,X), P(-(MU2-1),NU1,X), ..., P(-MU1,NU1,X) and store in ascending mu order. |
| DXPMUP | To compute the values of Legendre functions for DXLEGF. This subroutine transforms an array of Legendre functions |

of the first kind of negative order stored in array PQA into Legendre functions of the first kind of positive order stored in array PQA. The original array is destroyed.

DXPNRM — To compute the values of Legendre functions for DXLEGF. This subroutine transforms an array of Legendre functions of the first kind of negative order stored in array PQA into normalized Legendre polynomials stored in array PQA. The original array is destroyed.

DXPQNU — To compute the values of Legendre functions for DXLEGF. This subroutine calculates initial values of P or Q using power series, then performs forward nu-wise recurrence to obtain P(-MU,NU,X), Q(0,NU,X), or Q(1,NU,X). The nu-wise recurrence is stable for P for all mu and for Q for mu=0,1.

DXPSI — To compute values of the Psi function for DXLEGF.

DXQMU — To compute the values of Legendre functions for DXLEGF. Method: forward mu-wise recurrence for Q(MU,NU,X) for fixed nu to obtain Q(MU1,NU,X), Q(MU1+1,NU,X), ..., Q(MU2,NU,X).

DXQNU — To compute the values of Legendre functions for DXLEGF. Method: backward nu-wise recurrence for Q(MU,NU,X) for fixed mu to obtain Q(MU1,NU1,X), Q(MU1,NU1+1,X), ..., Q(MU1,NU2,X).

DY — Subsidiary to SEPELI

DY4 — Subsidiary to SEPX4

DYAIRY — Subsidiary to DBESJ and DBESY

EFCMN — Subsidiary to EFC

ENORM — Subsidiary to SNLS1, SNLS1E, SNSQ and SNSQE

EXBVP — Subsidiary to BVSUP

EZFFT1 — EZFFTI calls EZFFT1 with appropriate work array partitioning.

FCMN — Subsidiary to FC

FDJAC1 — Subsidiary to SNSQ and SNSQE

FDJAC3 — Subsidiary to SNLS1 and SNLS1E

FULMAT — Subsidiary to SPLP

GAMLN — Compute the logarithm of the Gamma function

GAMRN — Subsidiary to BSKIN

H12 — Subsidiary to HFTI, LSEI and WNNLS

HKSEQ — Subsidiary to BSKIN

HSTART — Subsidiary to DEABM, DEBDF and DERKF

HSTCS1 — Subsidiary to HSTCSP

HVNRM    Subsidiary to DEABM, DEBDF and DERKF

HWSCS1   Subsidiary to HWSCSP

HWSSS1   Subsidiary to HWSSSP

I1MERG   Merge two strings of ascending integers.

IDLOC    Subsidiary to DSPLP

INDXA    Subsidiary to BLKTRI

INDXB    Subsidiary to BLKTRI

INDXC    Subsidiary to BLKTRI

INTYD    Subsidiary to DEBDF

INXCA    Subsidiary to CBLKTR

INXCB    Subsidiary to CBLKTR

INXCC    Subsidiary to CBLKTR

IPLOC    Subsidiary to SPLP

ISDBCG   Preconditioned BiConjugate Gradient Stop Test.
This routine calculates the stop test for the BiConjugate
Gradient iteration scheme.  It returns a non-zero if the
error estimate (the type of which is determined by ITOL)
is less than the user specified tolerance TOL.

ISDCG    Preconditioned Conjugate Gradient Stop Test.
This routine calculates the stop test for the Conjugate
Gradient iteration scheme.  It returns a non-zero if the
error estimate (the type of which is determined by ITOL)
is less than the user specified tolerance TOL.

ISDCGN   Preconditioned CG on Normal Equations Stop Test.
This routine calculates the stop test for the Conjugate
Gradient iteration scheme applied to the normal equations.
It returns a non-zero if the error estimate (the type of
which is determined by ITOL) is less than the user
specified tolerance TOL.

ISDCGS   Preconditioned BiConjugate Gradient Squared Stop Test.
This routine calculates the stop test for the BiConjugate
Gradient Squared iteration scheme.  It returns a non-zero
if the error estimate (the type of which is determined by
ITOL) is less than the user specified tolerance TOL.

ISDGMR   Generalized Minimum Residual Stop Test.
This routine calculates the stop test for the Generalized
Minimum RESidual (GMRES) iteration scheme.  It returns a
non-zero if the error estimate (the type of which is
determined by ITOL) is less than the user specified
tolerance TOL.

ISDIR    Preconditioned Iterative Refinement Stop Test.
This routine calculates the stop test for the iterative

refinement iteration scheme.  It returns a non-zero if the error estimate (the type of which is determined by ITOL) is less than the user specified tolerance TOL.

ISDOMN    Preconditioned Orthomin Stop Test.
This routine calculates the stop test for the Orthomin iteration scheme.  It returns a non-zero if the error estimate (the type of which is determined by ITOL) is less than the user specified tolerance TOL.

ISSBCG    Preconditioned BiConjugate Gradient Stop Test.
This routine calculates the stop test for the BiConjugate Gradient iteration scheme.  It returns a non-zero if the error estimate (the type of which is determined by ITOL) is less than the user specified tolerance TOL.

ISSCG    Preconditioned Conjugate Gradient Stop Test.
This routine calculates the stop test for the Conjugate Gradient iteration scheme.  It returns a non-zero if the error estimate (the type of which is determined by ITOL) is less than the user specified tolerance TOL.

ISSCGN    Preconditioned CG on Normal Equations Stop Test.
This routine calculates the stop test for the Conjugate Gradient iteration scheme applied to the normal equations. It returns a non-zero if the error estimate (the type of which is determined by ITOL) is less than the user specified tolerance TOL.

ISSCGS    Preconditioned BiConjugate Gradient Squared Stop Test.
This routine calculates the stop test for the BiConjugate Gradient Squared iteration scheme.  It returns a non-zero if the error estimate (the type of which is determined by ITOL) is less than the user specified tolerance TOL.

ISSGMR    Generalized Minimum Residual Stop Test.
This routine calculates the stop test for the Generalized Minimum RESidual (GMRES) iteration scheme.  It returns a non-zero if the error estimate (the type of which is determined by ITOL) is less than the user specified tolerance TOL.

ISSIR    Preconditioned Iterative Refinement Stop Test.
This routine calculates the stop test for the iterative refinement iteration scheme.  It returns a non-zero if the error estimate (the type of which is determined by ITOL) is less than the user specified tolerance TOL.

ISSOMN    Preconditioned Orthomin Stop Test.
This routine calculates the stop test for the Orthomin iteration scheme.  It returns a non-zero if the error estimate (the type of which is determined by ITOL) is less than the user specified tolerance TOL.

IVOUT    Subsidiary to SPLP

J4SAVE    Save or recall global variables needed by error handling routines.

JAIRY    Subsidiary to BESJ and BESY

| | |
|---|---|
| LA05AD | Subsidiary to DSPLP |
| LA05AS | Subsidiary to SPLP |
| LA05BD | Subsidiary to DSPLP |
| LA05BS | Subsidiary to SPLP |
| LA05CD | Subsidiary to DSPLP |
| LA05CS | Subsidiary to SPLP |
| LA05ED | Subsidiary to DSPLP |
| LA05ES | Subsidiary to SPLP |
| LMPAR | Subsidiary to SNLS1 and SNLS1E |
| LPDP | Subsidiary to LSEI |
| LSAME | Test two characters to determine if they are the same letter, except for case. |
| LSI | Subsidiary to LSEI |
| LSOD | Subsidiary to DEBDF |
| LSSODS | Subsidiary to BVSUP |
| LSSUDS | Subsidiary to BVSUP |
| MACON | Subsidiary to BVSUP |
| MC20AD | Subsidiary to DSPLP |
| MC20AS | Subsidiary to SPLP |
| MGSBV | Subsidiary to BVSUP |
| MINSO4 | Subsidiary to SEPX4 |
| MINSOL | Subsidiary to SEPELI |
| MPADD | Subsidiary to DQDOTA and DQDOTI |
| MPADD2 | Subsidiary to DQDOTA and DQDOTI |
| MPADD3 | Subsidiary to DQDOTA and DQDOTI |
| MPBLAS | Subsidiary to DQDOTA and DQDOTI |
| MPCDM | Subsidiary to DQDOTA and DQDOTI |
| MPCHK | Subsidiary to DQDOTA and DQDOTI |
| MPCMD | Subsidiary to DQDOTA and DQDOTI |
| MPDIVI | Subsidiary to DQDOTA and DQDOTI |
| MPERR | Subsidiary to DQDOTA and DQDOTI |

| | |
|---|---|
| MPMAXR | Subsidiary to DQDOTA and DQDOTI |
| MPMLP | Subsidiary to DQDOTA and DQDOTI |
| MPMUL | Subsidiary to DQDOTA and DQDOTI |
| MPMUL2 | Subsidiary to DQDOTA and DQDOTI |
| MPMULI | Subsidiary to DQDOTA and DQDOTI |
| MPNZR | Subsidiary to DQDOTA and DQDOTI |
| MPOVFL | Subsidiary to DQDOTA and DQDOTI |
| MPSTR | Subsidiary to DQDOTA and DQDOTI |
| MPUNFL | Subsidiary to DQDOTA and DQDOTI |
| OHTROL | Subsidiary to BVSUP |
| OHTROR | Subsidiary to BVSUP |
| ORTHO4 | Subsidiary to SEPX4 |
| ORTHOG | Subsidiary to SEPELI |
| ORTHOL | Subsidiary to BVSUP |
| ORTHOR | Subsidiary to BVSUP |
| PASSB | Calculate the fast Fourier transform of subvectors of arbitrary length. |
| PASSB2 | Calculate the fast Fourier transform of subvectors of length two. |
| PASSB3 | Calculate the fast Fourier transform of subvectors of length three. |
| PASSB4 | Calculate the fast Fourier transform of subvectors of length four. |
| PASSB5 | Calculate the fast Fourier transform of subvectors of length five. |
| PASSF | Calculate the fast Fourier transform of subvectors of arbitrary length. |
| PASSF2 | Calculate the fast Fourier transform of subvectors of length two. |
| PASSF3 | Calculate the fast Fourier transform of subvectors of length three. |
| PASSF4 | Calculate the fast Fourier transform of subvectors of length four. |
| PASSF5 | Calculate the fast Fourier transform of subvectors of length five. |
| PCHCE | Set boundary conditions for PCHIC |

| | |
|---|---|
| PCHCI | Set interior derivatives for PCHIC |
| PCHCS | Adjusts derivative values for PCHIC |
| PCHDF | Computes divided differences for PCHCE and PCHSP |
| PCHKT | Compute B-spline knot sequence for PCHBS. |
| PCHNGS | Subsidiary to SPLP |
| PCHST | PCHIP Sign-Testing Routine |
| PCHSW | Limits excursion from data for PCHCS |
| PGSF | Subsidiary to CBLKTR |
| PIMACH | Subsidiary to HSTCSP, HSTSSP and HWSCSP |
| PINITM | Subsidiary to SPLP |
| PJAC | Subsidiary to DEBDF |
| PNNZRS | Subsidiary to SPLP |
| POISD2 | Subsidiary to GENBUN |
| POISN2 | Subsidiary to GENBUN |
| POISP2 | Subsidiary to GENBUN |
| POS3D1 | Subsidiary to POIS3D |
| POSTG2 | Subsidiary to POISTG |
| PPADD | Subsidiary to BLKTRI |
| PPGQ8 | Subsidiary to PFQAD |
| PPGSF | Subsidiary to CBLKTR |
| PPPSF | Subsidiary to CBLKTR |
| PPSGF | Subsidiary to BLKTRI |
| PPSPF | Subsidiary to BLKTRI |
| PROC | Subsidiary to CBLKTR |
| PROCP | Subsidiary to CBLKTR |
| PROD | Subsidiary to BLKTRI |
| PRODP | Subsidiary to BLKTRI |
| PRVEC | Subsidiary to BVSUP |
| PRWPGE | Subsidiary to SPLP |
| PRWVIR | Subsidiary to SPLP |

PSGF Subsidiary to BLKTRI

PSIXN Subsidiary to EXINT

PYTHAG Compute the complex square root of a complex number without destructive overflow or underflow.

QCHEB This routine computes the CHEBYSHEV series expansion of degrees 12 and 24 of a function using A FAST FOURIER TRANSFORM METHOD
F(X) = SUM(K=1,..,13) (CHEB12(K)*T(K-1,X)),
F(X) = SUM(K=1,..,25) (CHEB24(K)*T(K-1,X)),
Where T(K,X) is the CHEBYSHEV POLYNOMIAL OF DEGREE K.

QELG The routine determines the limit of a given sequence of approximations, by means of the Epsilon algorithm of P. Wynn. An estimate of the absolute error is also given. The condensed Epsilon table is computed. Only those elements needed for the computation of the next diagonal are preserved.

QFORM Subsidiary to SNSQ and SNSQE

QPSRT Subsidiary to QAGE, QAGIE, QAGPE, QAGSE, QAWCE, QAWOE and QAWSE

QRFAC Subsidiary to SNLS1, SNLS1E, SNSQ and SNSQE

QRSOLV Subsidiary to SNLS1 and SNLS1E

QS2I1D Sort an integer array, moving an integer and DP array. This routine sorts the integer array IA and makes the same interchanges in the integer array JA and the double precision array A. The array IA may be sorted in increasing order or decreasing order. A slightly modified QUICKSORT algorithm is used.

QS2I1R Sort an integer array, moving an integer and real array. This routine sorts the integer array IA and makes the same interchanges in the integer array JA and the real array A. The array IA may be sorted in increasing order or decreasing order. A slightly modified QUICKSORT algorithm is used.

QWGTC This function subprogram is used together with the routine QAWC and defines the WEIGHT function.

QWGTF This function subprogram is used together with the routine QAWF and defines the WEIGHT function.

QWGTS This function subprogram is used together with the routine QAWS and defines the WEIGHT function.

R1MPYQ Subsidiary to SNSQ and SNSQE

R1UPDT Subsidiary to SNSQ and SNSQE

R9AIMP Evaluate the Airy modulus and phase.

R9ATN1 Evaluate ATAN(X) from first order relative accuracy so that ATAN(X) = X + X**3*R9ATN1(X).

| | |
|---|---|
| R9CHU | Evaluate for large Z  Z**A * U(A,B,Z) where U is the logarithmic confluent hypergeometric function. |
| R9GMIC | Compute the complementary incomplete Gamma function for A near a negative integer and for small X. |
| R9GMIT | Compute Tricomi's incomplete Gamma function for small arguments. |
| R9KNUS | Compute Bessel functions EXP(X)*K-SUB-XNU(X) and EXP(X)* K-SUB-XNU+1(X) for 0.0 .LE. XNU .LT. 1.0. |
| R9LGIC | Compute the log complementary incomplete Gamma function for large X and for A .LE. X. |
| R9LGIT | Compute the logarithm of Tricomi's incomplete Gamma function with Perron's continued fraction for large X and A .GE. X. |
| R9LGMC | Compute the log Gamma correction factor so that LOG(GAMMA(X)) = LOG(SQRT(2*PI)) + (X-.5)*LOG(X) - X + R9LGMC(X). |
| R9LN2R | Evaluate LOG(1+X) from second order relative accuracy so that LOG(1+X) = X - X**2/2 + X**3*R9LN2R(X). |
| RADB2 | Calculate the fast Fourier transform of subvectors of length two. |
| RADB3 | Calculate the fast Fourier transform of subvectors of length three. |
| RADB4 | Calculate the fast Fourier transform of subvectors of length four. |
| RADB5 | Calculate the fast Fourier transform of subvectors of length five. |
| RADBG | Calculate the fast Fourier transform of subvectors of arbitrary length. |
| RADF2 | Calculate the fast Fourier transform of subvectors of length two. |
| RADF3 | Calculate the fast Fourier transform of subvectors of length three. |
| RADF4 | Calculate the fast Fourier transform of subvectors of length four. |
| RADF5 | Calculate the fast Fourier transform of subvectors of length five. |
| RADFG | Calculate the fast Fourier transform of subvectors of arbitrary length. |
| REORT | Subsidiary to BVSUP |
| RFFTB | Compute the backward fast Fourier transform of a real coefficient array. |

RFFTF Compute the forward transform of a real, periodic sequence.

RFFTI Initialize a work array for RFFTF and RFFTB.

RKFAB Subsidiary to BVSUP

RSCO Subsidiary to DEBDF

RWUPDT Subsidiary to SNLS1 and SNLS1E

S1MERG Merge two strings of ascending real numbers.

SBOLSM Subsidiary to SBOCLS and SBOLS

SCHKW SLAP WORK/IWORK Array Bounds Checker. This routine checks the work array lengths and interfaces to the SLATEC error handler if a problem is found.

SCLOSM Subsidiary to SPLP

SCOEF Subsidiary to BVSUP

SDAINI Initialization routine for SDASSL.

SDAJAC Compute the iteration matrix for SDASSL and form the LU-decomposition.

SDANRM Compute vector norm for SDASSL.

SDASLV Linear system solver for SDASSL.

SDASTP Perform one step of the SDASSL integration.

SDATRP Interpolation routine for SDASSL.

SDAWTS Set error weight vector for SDASSL.

SDCOR Subroutine SDCOR computes corrections to the Y array.

SDCST SDCST sets coefficients used by the core integrator SDSTP.

SDNTL Subroutine SDNTL is called to set parameters on the first call to SDSTP, on an internal restart, or when the user has altered MINT, MITER, and/or H.

SDNTP Subroutine SDNTP interpolates the K-th derivative of Y at TOUT, using the data in the YH array. If K has a value greater than NQ, the NQ-th derivative is calculated.

SDPSC Subroutine SDPSC computes the predicted YH values by effectively multiplying the YH array by the Pascal triangle matrix when KSGN is +1, and performs the inverse function when KSGN is -1.

SDPST Subroutine SDPST evaluates the Jacobian matrix of the right hand side of the differential equations.

SDSCL Subroutine SDSCL rescales the YH array whenever the step size is changed.

| | |
|---|---|
| SDSTP | SDSTP performs one step of the integration of an initial value problem for a system of ordinary differential equations. |
| SDZRO | SDZRO searches for a zero of a function F(N, T, Y, IROOT) between the given values B and C until the width of the interval (B, C) has collapsed to within a tolerance specified by the stopping criterion, ABS(B - C) .LE. 2.*(RW*ABS(B) + AE). |
| SHELS | Internal routine for SGMRES. |
| SHEQR | Internal routine for SGMRES. |
| SLVS | Subsidiary to DEBDF |
| SMOUT | Subsidiary to FC and SBOCLS |
| SODS | Subsidiary to BVSUP |
| SOPENM | Subsidiary to SPLP |
| SORTH | Internal routine for SGMRES. |
| SOSEQS | Subsidiary to SOS |
| SOSSOL | Subsidiary to SOS |
| SPELI4 | Subsidiary to SEPX4 |
| SPELIP | Subsidiary to SEPELI |
| SPIGMR | Internal routine for SGMRES. |
| SPINCW | Subsidiary to SPLP |
| SPINIT | Subsidiary to SPLP |
| SPLPCE | Subsidiary to SPLP |
| SPLPDM | Subsidiary to SPLP |
| SPLPFE | Subsidiary to SPLP |
| SPLPFL | Subsidiary to SPLP |
| SPLPMN | Subsidiary to SPLP |
| SPLPMU | Subsidiary to SPLP |
| SPLPUP | Subsidiary to SPLP |
| SPOPT | Subsidiary to SPLP |
| SREADP | Subsidiary to SPLP |
| SRLCAL | Internal routine for SGMRES. |
| STOD | Subsidiary to DEBDF |
| STOR1 | Subsidiary to BVSUP |

| | |
|---|---|
| STWAY | Subsidiary to BVSUP |
| SUDS | Subsidiary to BVSUP |
| SVCO | Subsidiary to DEBDF |
| SVD | Perform the singular value decomposition of a rectangular matrix. |
| SVECS | Subsidiary to BVSUP |
| SVOUT | Subsidiary to SPLP |
| SWRITP | Subsidiary to SPLP |
| SXLCAL | Internal routine for SGMRES. |
| TEVLC | Subsidiary to CBLKTR |
| TEVLS | Subsidiary to BLKTRI |
| TRI3 | Subsidiary to GENBUN |
| TRIDQ | Subsidiary to POIS3D |
| TRIS4 | Subsidiary to SEPX4 |
| TRISP | Subsidiary to SEPELI |
| TRIX | Subsidiary to GENBUN |
| U11LS | Subsidiary to LLSIA |
| U11US | Subsidiary to ULSIA |
| U12LS | Subsidiary to LLSIA |
| U12US | Subsidiary to ULSIA |
| USRMAT | Subsidiary to SPLP |
| VNWRMS | Subsidiary to DEBDF |
| WNLIT | Subsidiary to WNNLS |
| WNLSM | Subsidiary to WNNLS |
| WNLT1 | Subsidiary to WNLIT |
| WNLT2 | Subsidiary to WNLIT |
| WNLT3 | Subsidiary to WNLIT |
| XERBLA | Error handler for the Level 2 and Level 3 BLAS Routines. |
| XERCNT | Allow user control over handling of errors. |
| XERHLT | Abort program execution and print error message. |
| XERPRN | Print error messages processed by XERMSG. |

XERSVE Record that an error has occurred.

XPMU To compute the values of Legendre functions for XLEGF. Method: backward mu-wise recurrence for P(-MU,NU,X) for fixed nu to obtain P(-MU2,NU1,X), P(-(MU2-1),NU1,X), ..., P(-MU1,NU1,X) and store in ascending mu order.

XPMUP To compute the values of Legendre functions for XLEGF. This subroutine transforms an array of Legendre functions of the first kind of negative order stored in array PQA into Legendre functions of the first kind of positive order stored in array PQA. The original array is destroyed.

XPNRM To compute the values of Legendre functions for XLEGF. This subroutine transforms an array of Legendre functions of the first kind of negative order stored in array PQA into normalized Legendre polynomials stored in array PQA. The original array is destroyed.

XPQNU To compute the values of Legendre functions for XLEGF. This subroutine calculates initial values of P or Q using power series, then performs forward nu-wise recurrence to obtain P(-MU,NU,X), Q(0,NU,X), or Q(1,NU,X). The nu-wise recurrence is stable for P for all mu and for Q for mu=0,1.

XPSI To compute values of the Psi function for XLEGF.

XQMU To compute the values of Legendre functions for XLEGF. Method: forward mu-wise recurrence for Q(MU,NU,X) for fixed nu to obtain Q(MU1,NU,X), Q(MU1+1,NU,X), ..., Q(MU2,NU,X).

XQNU To compute the values of Legendre functions for XLEGF. Method: backward nu-wise recurrence for Q(MU,NU,X) for fixed mu to obtain Q(MU1,NU1,X), Q(MU1,NU1+1,X), ..., Q(MU1,NU2,X).

YAIRY Subsidiary to BESJ and BESY

ZABS Subsidiary to ZBESH, ZBESI, ZBESJ, ZBESK, ZBESY, ZAIRY and ZBIRY

ZACAI Subsidiary to ZAIRY

ZACON Subsidiary to ZBESH and ZBESK

ZASYI Subsidiary to ZBESI and ZBESK

ZBINU Subsidiary to ZAIRY, ZBESH, ZBESI, ZBESJ, ZBESK and ZBIRY

ZBKNU Subsidiary to ZAIRY, ZBESH, ZBESI and ZBESK

ZBUNI Subsidiary to ZBESI and ZBESK

ZBUNK Subsidiary to ZBESH and ZBESK

ZDIV Subsidiary to ZBESH, ZBESI, ZBESJ, ZBESK, ZBESY, ZAIRY and ZBIRY

ZEXP Subsidiary to ZBESH, ZBESI, ZBESJ, ZBESK, ZBESY, ZAIRY and ZBIRY

| | |
|---|---|
| ZKSCL | Subsidiary to ZBESK |
| ZLOG | Subsidiary to ZBESH, ZBESI, ZBESJ, ZBESK, ZBESY, ZAIRY and ZBIRY |
| ZMLRI | Subsidiary to ZBESI and ZBESK |
| ZMLT | Subsidiary to ZBESH, ZBESI, ZBESJ, ZBESK, ZBESY, ZAIRY and ZBIRY |
| ZRATI | Subsidiary to ZBESH, ZBESI and ZBESK |
| ZS1S2 | Subsidiary to ZAIRY and ZBESK |
| ZSERI | Subsidiary to ZBESI and ZBESK |
| ZSHCH | Subsidiary to ZBESH and ZBESK |
| ZSQRT | Subsidiary to ZBESH, ZBESI, ZBESJ, ZBESK, ZBESY, ZAIRY and ZBIRY |
| ZUCHK | Subsidiary to SERI, ZUOIK, ZUNK1, ZUNK2, ZUNI1, ZUNI2 and ZKSCL |
| ZUNHJ | Subsidiary to ZBESI and ZBESK |
| ZUNI1 | Subsidiary to ZBESI and ZBESK |
| ZUNI2 | Subsidiary to ZBESI and ZBESK |
| ZUNIK | Subsidiary to ZBESI and ZBESK |
| ZUNK1 | Subsidiary to ZBESK |
| ZUNK2 | Subsidiary to ZBESK |
| ZUOIK | Subsidiary to ZBESH, ZBESI and ZBESK |
| ZWRSK | Subsidiary to ZBESI and ZBESK |

SECTION III. Alphabetic List of Routines and Categories
As stated in the introduction, an asterisk (*) immediately preceeding a routine name indicates a subsidiary routine.

| | | | |
|---|---|---|---|
| AAAAAA | Z | ACOSH | C4C |
| AI | C10D | AIE | C10D |
| ALBETA | C7B | ALGAMS | C7A |
| ALI | C5 | ALNGAM | C7A |
| ALNREL | C4B | ASINH | C4C |
| *ASYIK | | *ASYJY | |
| ATANH | C4C | AVINT | H2A1B2 |
| BAKVEC | D4C4 | BALANC | D4C1A |
| BALBAK | D4C4 | BANDR | D4C1B1 |
| BANDV | D4C3 | *BCRH | |
| *BDIFF | | BESI | C10B3 |
| BESI0 | C10B1 | BESI0E | C10B1 |
| BESI1 | C10B1 | BESI1E | C10B1 |
| BESJ | C10A3 | BESJ0 | C10A1 |
| BESJ1 | C10A1 | BESK | C10B3 |

| | | | |
|---|---|---|---|
| BESK0 | C10B1 | BESK0E | C10B1 |
| BESK1 | C10B1 | BESK1E | C10B1 |
| BESKES | C10B3 | *BESKNU | |
| BESKS | C10B3 | BESY | C10A3 |
| BESY0 | C10A1 | BESY1 | C10A1 |
| *BESYNU | | BETA | C7B |
| BETAI | C7F | BFQAD | H2A2A1, E3, K6 |
| BI | C10D | BIE | C10D |
| BINOM | C1 | BINT4 | E1A |
| BINTK | E1A | BISECT | D4A5, D4C2A |
| *BKIAS | | *BKISR | |
| *BKSOL | | *BLKTR1 | |
| BLKTRI | I2B4B | BNDACC | D9 |
| BNDSOL | D9 | *BNFAC | |
| *BNSLV | | BQR | D4A6 |
| *BSGQ8 | | BSKIN | C10F |
| BSPDOC | E, E1A, K, Z | BSPDR | E3 |
| BSPEV | E3, K6 | *BSPLVD | |
| *BSPLVN | | BSPPP | E3, K6 |
| BSPVD | E3, K6 | BSPVN | E3, K6 |
| BSQAD | H2A2A1, E3, K6 | *BSRH | |
| BVALU | E3, K6 | *BVDER | |
| *BVPOR | | BVSUP | I1B1 |
| C0LGMC | C7A | *C1MERG | |
| *C9LGMC | C7A | *C9LN2R | C4B |
| *CACAI | | *CACON | |
| CACOS | C4A | CACOSH | C4C |
| CAIRY | C10D | CARG | A4A |
| CASIN | C4A | CASINH | C4C |
| *CASYI | | CATAN | C4A |
| CATAN2 | C4A | CATANH | C4C |
| CAXPY | D1A7 | CBABK2 | D4C4 |
| CBAL | D4C1A | CBESH | C10A4 |
| CBESI | C10B4 | CBESJ | C10A4 |
| CBESK | C10B4 | CBESY | C10A4 |
| CBETA | C7B | *CBINU | |
| CBIRY | C10D | *CBKNU | |
| *CBLKT1 | | CBLKTR | I2B4B |
| CBRT | C2 | *CBUNI | |
| *CBUNK | | CCBRT | C2 |
| CCHDC | D2D1B | CCHDD | D7B |
| CCHEX | D7B | CCHUD | D7B |
| *CCMPB | | CCOPY | D1A5 |
| CCOSH | C4C | CCOT | C4A |
| CDCDOT | D1A4 | *CDCOR | |
| *CDCST | | *CDIV | |
| *CDNTL | | *CDNTP | |
| CDOTC | D1A4 | CDOTU | D1A4 |
| *CDPSC | | *CDPST | |
| CDRIV1 | I1A2, I1A1B | CDRIV2 | I1A2, I1A1B |
| CDRIV3 | I1A2, I1A1B | *CDSCL | |
| *CDSTP | | *CDZRO | |
| CEXPRL | C4B | *CFFTB | J1A2 |
| CFFTB1 | J1A2 | *CFFTF | J1A2 |
| CFFTF1 | J1A2 | *CFFTI | J1A2 |
| CFFTI1 | J1A2 | *CFOD | |
| CG | D4A4 | CGAMMA | C7A |
| CGAMR | C7A | CGBCO | D2C2 |
| CGBDI | D3C2 | CGBFA | D2C2 |
| CGBMV | D1B4 | CGBSL | D2C2 |
| CGECO | D2C1 | CGEDI | D2C1, D3C1 |

| | | | |
|---|---|---|---|
| CGEEV | D4A4 | CGEFA | D2C1 |
| CGEFS | D2C1 | CGEIR | D2C1 |
| CGEMM | D1B6 | CGEMV | D1B4 |
| CGERC | D1B4 | CGERU | D1B4 |
| CGESL | D2C1 | CGTSL | D2C2A |
| CH | D4A3 | CHBMV | D1B4 |
| CHEMM | D1B6 | CHEMV | D1B4 |
| CHER | D1B4 | CHER2 | D1B4 |
| CHER2K | D1B6 | CHERK | D1B6 |
| *CHFCM | | CHFDV | E3, H1 |
| CHFEV | E3 | *CHFIE | |
| CHICO | D2D1A | CHIDI | D2D1A, D3D1A |
| CHIEV | D4A3 | CHIFA | D2D1A |
| CHISL | D2D1A | CHKDER | F3, G4C |
| *CHKPR4 | | *CHKPRM | |
| *CHKSN4 | | *CHKSNG | |
| CHPCO | D2D1A | CHPDI | D2D1A, D3D1A |
| CHPFA | D2D1A | CHPMV | D1B4 |
| CHPR | D1B4 | CHPR2 | D1B4 |
| CHPSL | D2D1A | CHU | C11 |
| CINVIT | D4C2B | *CKSCL | |
| CLBETA | C7B | CLNGAM | C7A |
| CLNREL | C4B | CLOG10 | C4B |
| CMGNBN | I2B4B | *CMLRI | |
| *CMPCSG | | *CMPOSD | |
| *CMPOSN | | *CMPOSP | |
| *CMPTR3 | | *CMPTRX | |
| CNBCO | D2C2 | CNBDI | D3C2 |
| CNBFA | D2C2 | CNBFS | D2C2 |
| CNBIR | D2C2 | CNBSL | D2C2 |
| COMBAK | D4C4 | COMHES | D4C1B2 |
| COMLR | D4C2B | COMLR2 | D4C2B |
| *COMPB | | COMQR | D4C2B |
| COMQR2 | D4C2B | CORTB | D4C4 |
| CORTH | D4C1B2 | COSDG | C4A |
| *COSGEN | | COSQB | J1A3 |
| *COSQB1 | J1A3 | COSQF | J1A3 |
| *COSQF1 | J1A3 | COSQI | J1A3 |
| COST | J1A3 | COSTI | J1A3 |
| COT | C4A | *CPADD | |
| CPBCO | D2D2 | CPBDI | D3D2 |
| CPBFA | D2D2 | CPBSL | D2D2 |
| *CPEVL | | *CPEVLR | |
| CPOCO | D2D1B | CPODI | D2D1B, D3D1B |
| CPOFA | D2D1B | CPOFS | D2D1B |
| CPOIR | D2D1B | CPOSL | D2D1B |
| CPPCO | D2D1B | CPPDI | D2D1B, D3D1B |
| CPPFA | D2D1B | CPPSL | D2D1B |
| CPQR79 | F1A1B | *CPROC | |
| *CPROCP | | *CPROD | |
| *CPRODP | | CPSI | C7C |
| CPTSL | D2D2A | CPZERO | F1A1B |
| CQRDC | D5 | CQRSL | D9, D2C1 |
| *CRATI | | CROTG | D1B10 |
| *CS1S2 | | CSCAL | D1A6 |
| *CSCALE | | *CSERI | |
| CSEVL | C3A2 | *CSHCH | |
| CSICO | D2C1 | CSIDI | D2C1, D3C1 |
| CSIFA | D2C1 | CSINH | C4C |
| CSISL | D2C1 | CSPCO | D2C1 |
| CSPDI | D2C1, D3C1 | CSPFA | D2C1 |

| | | | |
|---|---|---|---|
| CSPSL | D2C1 | *CSROOT | |
| CSROT | D1B10 | CSSCAL | D1A6 |
| CSVDC | D6 | CSWAP | D1A5 |
| CSYMM | D1B6 | CSYR2K | D1B6 |
| CSYRK | D1B6 | CTAN | C4A |
| CTANH | C4C | CTBMV | D1B4 |
| CTBSV | D1B4 | CTPMV | D1B4 |
| CTPSV | D1B4 | CTRCO | D2C3 |
| CTRDI | D2C3, D3C3 | CTRMM | D1B6 |
| CTRMV | D1B4 | CTRSL | D2C3 |
| CTRSM | D1B6 | CTRSV | D1B4 |
| *CUCHK | | *CUNHJ | |
| *CUNI1 | | *CUNI2 | |
| *CUNIK | | *CUNK1 | |
| *CUNK2 | | *CUOIK | |
| CV | L7A3 | *CWRSK | |
| D1MACH | R1 | *D1MERG | |
| *D1MPYQ | | *D1UPDT | |
| *D9AIMP | C10D | *D9ATN1 | C4A |
| *D9B0MP | C10A1 | *D9B1MP | C10A1 |
| *D9CHU | C11 | *D9GMIC | C7E |
| *D9GMIT | C7E | *D9KNUS | C10B3 |
| *D9LGIC | C7E | *D9LGIT | C7E |
| *D9LGMC | C7E | *D9LN2R | C4B |
| D9PAK | A6B | D9UPAK | A6B |
| DACOSH | C4C | DAI | C10D |
| DAIE | C10D | DASINH | C4C |
| DASUM | D1A3A | *DASYIK | |
| *DASYJY | | DATANH | C4C |
| DAVINT | H2A1B2 | DAWS | C8C |
| DAXPY | D1A7 | DBCG | D2A4, D2B4 |
| *DBDIFF | | DBESI | C10B3 |
| DBESI0 | C10B1 | DBESI1 | C10B1 |
| DBESJ | C10A3 | DBESJ0 | C10A1 |
| DBESJ1 | C10A1 | DBESK | C10B3 |
| DBESK0 | C10B1 | DBESK1 | C10B1 |
| DBESKS | C10B3 | DBESY | C10A3 |
| DBESY0 | C10A1 | DBESY1 | C10A1 |
| DBETA | C7B | DBETAI | C7F |
| DBFQAD | H2A2A1, E3, K6 | DBHIN | N1 |
| DBI | C10D | DBIE | C10D |
| DBINOM | C1 | DBINT4 | E1A |
| DBINTK | E1A | *DBKIAS | |
| *DBKISR | | *DBKSOL | |
| DBNDAC | D9 | DBNDSL | D9 |
| *DBNFAC | | *DBNSLV | |
| DBOCLS | K1A2A, G2E, G2H1, G2H2 | DBOLS | K1A2A, G2E, G2H1, G2H2 |
| *DBOLSM | | *DBSGQ8 | |
| DBSI0E | C10B1 | DBSI1E | C10B1 |
| DBSK0E | C10B1 | DBSK1E | C10B1 |
| DBSKES | C10B3 | DBSKIN | C10F |
| *DBSKNU | | DBSPDR | E3, K6 |
| DBSPEV | E3, K6 | DBSPPP | E3, K6 |
| DBSPVD | E3, K6 | DBSPVN | E3, K6 |
| DBSQAD | H2A2A1, E3, K6 | *DBSYNU | |
| DBVALU | E3, K6 | *DBVDER | |
| *DBVPOR | | DBVSUP | I1B1 |
| DCBRT | C2 | DCDOT | D1A4 |
| *DCFOD | | DCG | D2B4 |
| DCGN | D2A4, D2B4 | DCGS | D2A4, D2B4 |
| DCHDC | D2B1B | DCHDD | D7B |

| | | | |
|---|---|---|---|
| DCHEX | D7B | *DCHFCM | |
| DCHFDV | E3, H1 | DCHFEV | E3 |
| *DCHFIE | | *DCHKW | R2 |
| DCHU | C11 | DCHUD | D7B |
| DCKDER | F3, G4C | *DCOEF | |
| DCOPY | D1A5 | DCOPYM | D1A5 |
| DCOSDG | C4A | DCOT | C4A |
| DCOV | K1B1 | DCPPLT | N1 |
| *DCSCAL | | DCSEVL | C3A2 |
| DCV | L7A3 | *DDAINI | |
| *DDAJAC | | *DDANRM | |
| *DDASLV | | DDASSL | I1A2 |
| *DDASTP | | *DDATRP | |
| DDAWS | C8C | *DDAWTS | |
| *DDCOR | | *DDCST | |
| DDEABM | I1A1B | DDEBDF | I1A2 |
| DDERKF | I1A1A | *DDES | |
| *DDNTL | | *DDNTP | |
| *DDOGLG | | DDOT | D1A4 |
| *DDPSC | | *DDPST | |
| DDRIV1 | I1A2, I1A1B | DDRIV2 | I1A2, I1A1B |
| DDRIV3 | I1A2, I1A1B | *DDSCL | |
| *DDSTP | | *DDZRO | |
| DE1 | C5 | DEABM | I1A1B |
| DEBDF | I1A2 | DEFC | K1A1A1, K1A2A, L8A3 |
| *DEFCMN | | *DEFE4 | |
| *DEFEHL | | *DEFER | |
| DEI | C5 | *DENORM | |
| DERF | C8A, L5A1E | DERFC | C8A, L5A1E |
| DERKF | I1A1A | *DERKFS | |
| *DES | | *DEXBVP | |
| DEXINT | C5 | DEXPRL | C4B |
| DFAC | C1 | DFC | K1A1A1, K1A2A, L8A3 |
| *DFCMN | | *DFDJC1 | |
| *DFDJC3 | | *DFEHL | |
| *DFSPVD | | *DFSPVN | |
| *DFULMT | | DFZERO | F1B |
| DGAMI | C7E | DGAMIC | C7E |
| DGAMIT | C7E | DGAMLM | C7A, R2 |
| *DGAMLN | C7A | DGAMMA | C7A |
| DGAMR | C7A | *DGAMRN | |
| DGAUS8 | H2A1A1 | DGBCO | D2A2 |
| DGBDI | D3A2 | DGBFA | D2A2 |
| DGBMV | D1B4 | DGBSL | D2A2 |
| DGECO | D2A1 | DGEDI | D3A1, D2A1 |
| DGEFA | D2A1 | DGEFS | D2A1 |
| DGEMM | D1B6 | DGEMV | D1B4 |
| DGER | D1B4 | DGESL | D2A1 |
| DGLSS | D9, D5 | DGMRES | D2A4, D2B4 |
| DGTSL | D2A2A | *DH12 | |
| *DHELS | D2A4, D2B4 | *DHEQR | D2A4, D2B4 |
| DHFTI | D9 | *DHKSEQ | |
| *DHSTRT | | *DHVNRM | |
| DINTP | I1A1B | DINTRV | E3, K6 |
| *DINTYD | | DIR | D2A4, D2B4 |
| *DJAIRY | | DLBETA | C7B |
| DLGAMS | C7A | DLI | C5 |
| DLLSIA | D9, D5 | DLLTI2 | D2E |
| DLNGAM | C7A | DLNREL | C4B |
| DLPDOC | D2A4, D2B4, Z | *DLPDP | |
| DLSEI | K1A2A, D9 | *DLSI | |

| | | | |
|---|---|---|---|
| DQRDC | D5 | *DQRFAC | |
| DQRSL | D9, D2A1 | *DQRSLV | |
| *DQWGTC | | *DQWGTF | |
| *DQWGTS | | DRC | C14 |
| DRC3JJ | C19 | DRC3JM | C19 |
| DRC6J | C19 | DRD | C14 |
| *DREADP | | *DREORT | |
| DRF | C14 | DRJ | C14 |
| *DRKFAB | | *DRKFS | |
| *DRLCAL | D2A4, D2B4 | DROT | D1A8 |
| DROTG | D1B10 | DROTM | D1A8 |
| DROTMG | D1B10 | *DRSCO | |
| DS2LT | D2E | DS2Y | D1B9 |
| DSBMV | D1B4 | DSCAL | D1A6 |
| DSD2S | D2E | DSDBCG | D2A4, D2B4 |
| DSDCG | D2B4 | DSDCGN | D2A4, D2B4 |
| DSDCGS | D2A4, D2B4 | DSDGMR | D2A4, D2B4 |
| DSDI | D1B4 | DSDOMN | D2A4, D2B4 |
| DSDOT | D1A4 | DSDS | D2E |
| DSDSCL | D2E | DSGS | D2A4, D2B4 |
| DSICCG | D2B4 | DSICO | D2B1A |
| DSICS | D2E | DSIDI | D2B1A, D3B1A |
| DSIFA | D2B1A | DSILUR | D2A4, D2B4 |
| DSILUS | D2E | DSINDG | C4A |
| DSISL | D2B1A | DSJAC | D2A4, D2B4 |
| DSLI | D2A3 | DSLI2 | D2A3 |
| DSLLTI | D2E | DSLUBC | D2A4, D2B4 |
| DSLUCN | D2A4, D2B4 | DSLUCS | D2A4, D2B4 |
| DSLUGM | D2A4, D2B4 | DSLUI | D2E |
| DSLUI2 | D2E | DSLUI4 | D2E |
| DSLUOM | D2A4, D2B4 | DSLUTI | D2E |
| *DSLVS | | DSMMI2 | D2E |
| DSMMTI | D2E | DSMTV | D1B4 |
| DSMV | D1B4 | DSORT | N6A2B |
| DSOS | F2A | *DSOSEQ | |
| *DSOSSL | | DSPCO | D2B1A |
| DSPDI | D2B1A, D3B1A | DSPENC | C5 |
| DSPFA | D2B1A | DSPLP | G2A2 |
| DSPMV | D1B4 | DSPR | D1B4 |
| DSPR2 | D1B4 | DSPSL | D2B1A |
| DSTEPS | I1A1B | *DSTOD | |
| *DSTOR1 | | *DSTWAY | |
| *DSUDS | | *DSVCO | |
| DSVDC | D6 | DSWAP | D1A5 |
| DSYMM | D1B6 | DSYMV | D1B4 |
| DSYR | D1B4 | DSYR2 | D1B4 |
| DSYR2K | D1B6 | DSYRK | D1B6 |
| DTBMV | D1B4 | DTBSV | D1B4 |
| DTIN | N1 | DTOUT | N1 |
| DTPMV | D1B4 | DTPSV | D1B4 |
| DTRCO | D2A3 | DTRDI | D2A3, D3A3 |
| DTRMM | D1B6 | DTRMV | D1B4 |
| DTRSL | D2A3 | DTRSM | D1B6 |
| DTRSV | D1B4 | *DU11LS | |
| *DU11US | | *DU12LS | |
| *DU12US | | DULSIA | D9 |
| *DUSRMT | | *DVECS | |
| *DVNRMS | | *DVOUT | |
| *DWNLIT | | *DWNLSM | |
| *DWNLT1 | | *DWNLT2 | |
| *DWNLT3 | | DWNNLS | K1A2A |

| | | | |
|---|---|---|---|
| *DWRITP | | *DWUPDT | |
| *DX | | *DX4 | |
| DXADD | A3D | DXADJ | A3D |
| DXC210 | A3D | DXCON | A3D |
| *DXLCAL | D2A4, D2B4 | DXLEGF | C3A2, C9 |
| DXNRMP | C3A2, C9 | *DXPMU | C3A2, C9 |
| *DXPMUP | C3A2, C9 | *DXPNRM | C3A2, C9 |
| *DXPQNU | C3A2, C9 | *DXPSI | C7C |
| *DXQMU | C3A2, C9 | *DXQNU | C3A2, C9 |
| DXRED | A3D | DXSET | A3D |
| *DY | | *DY4 | |
| *DYAIRY | | E1 | C5 |
| EFC | K1A1A1, K1A2A, L8A3 | *EFCMN | |
| EI | C5 | EISDOC | D4, Z |
| ELMBAK | D4C4 | ELMHES | D4C1B2 |
| ELTRAN | D4C4 | *ENORM | |
| ERF | C8A, L5A1E | ERFC | C8A, L5A1E |
| *EXBVP | | EXINT | C5 |
| EXPREL | C4B | *EZFFT1 | |
| EZFFTB | J1A1 | EZFFTF | J1A1 |
| EZFFTI | J1A1 | FAC | C1 |
| FC | K1A1A1, K1A2A, L8A3 | *FCMN | |
| *FDJAC1 | | *FDJAC3 | |
| FDUMP | R3 | FFTDOC | J1, Z |
| FIGI | D4C1C | FIGI2 | D4C1C |
| *FULMAT | | FUNDOC | C, Z |
| FZERO | F1B | GAMI | C7E |
| GAMIC | C7E | GAMIT | C7E |
| GAMLIM | C7A, R2 | *GAMLN | C7A |
| GAMMA | C7A | GAMR | C7A |
| *GAMRN | | GAUS8 | H2A1A1 |
| GENBUN | I2B4B | *H12 | |
| HFTI | D9 | *HKSEQ | |
| HPPERM | N8 | HPSORT | N6A1C, N6A2C |
| HQR | D4C2B | HQR2 | D4C2B |
| *HSTART | | HSTCRT | I2B1A1A |
| *HSTCS1 | | HSTCSP | I2B1A1A |
| HSTCYL | I2B1A1A | HSTPLR | I2B1A1A |
| HSTSSP | I2B1A1A | HTRIB3 | D4C4 |
| HTRIBK | D4C4 | HTRID3 | D4C1B1 |
| HTRIDI | D4C1B1 | *HVNRM | |
| HW3CRT | I2B1A1A | HWSCRT | I2B1A1A |
| *HWSCS1 | | HWSCSP | I2B1A1A |
| HWSCYL | I2B1A1A | HWSPLR | I2B1A1A |
| *HWSSS1 | | HWSSSP | I2B1A1A |
| I1MACH | R1 | *I1MERG | |
| ICAMAX | D1A2 | ICOPY | D1A5 |
| IDAMAX | D1A2 | *IDLOC | |
| IMTQL1 | D4A5, D4C2A | IMTQL2 | D4A5, D4C2A |
| IMTQLV | D4A5, D4C2A | *INDXA | |
| *INDXB | | *INDXC | |
| INITDS | C3A2 | INITS | C3A2 |
| INTRV | E3, K6 | *INTYD | |
| INVIT | D4C2B | *INXCA | |
| *INXCB | | *INXCC | |
| *IPLOC | | IPPERM | N8 |
| IPSORT | N6A1A, N6A2A | ISAMAX | D1A2 |
| *ISDBCG | D2A4, D2B4 | *ISDCG | D2B4 |
| *ISDCGN | D2A4, D2B4 | *ISDCGS | D2A4, D2B4 |
| *ISDGMR | D2A4, D2B4 | *ISDIR | D2A4, D2B4 |
| *ISDOMN | D2A4, D2B4 | ISORT | N6A2A |

| | | | |
|---|---|---|---|
| *ISSBCG | D2A4, D2B4 | *ISSCG | D2B4 |
| *ISSCGN | D2A4, D2B4 | *ISSCGS | D2A4, D2B4 |
| *ISSGMR | D2A4, D2B4 | *ISSIR | D2A4, D2B4 |
| *ISSOMN | D2A4, D2B4 | ISWAP | D1A5 |
| *IVOUT | | *J4SAVE | |
| *JAIRY | | *LA05AD | |
| *LA05AS | | *LA05BD | |
| *LA05BS | | *LA05CD | |
| *LA05CS | | *LA05ED | |
| *LA05ES | | LLSIA | D9, D5 |
| *LMPAR | | *LPDP | |
| *LSAME | R, N3 | LSEI | K1A2A, D9 |
| *LSI | | *LSOD | |
| *LSSODS | | *LSSUDS | |
| *MACON | | *MC20AD | |
| *MC20AS | | *MGSBV | |
| MINFIT | D9 | *MINSO4 | |
| *MINSOL | | *MPADD | |
| *MPADD2 | | *MPADD3 | |
| *MPBLAS | | *MPCDM | |
| *MPCHK | | *MPCMD | |
| *MPDIVI | | *MPERR | |
| *MPMAXR | | *MPMLP | |
| *MPMUL | | *MPMUL2 | |
| *MPMULI | | *MPNZR | |
| *MPOVFL | | *MPSTR | |
| *MPUNFL | | NUMXER | R3C |
| *OHTROL | | *OHTROR | |
| ORTBAK | D4C4 | ORTHES | D4C1B2 |
| *ORTHO4 | | *ORTHOG | |
| *ORTHOL | | *ORTHOR | |
| ORTRAN | D4C4 | *PASSB | |
| *PASSB2 | | *PASSB3 | |
| *PASSB4 | | *PASSB5 | |
| *PASSF | | *PASSF2 | |
| *PASSF3 | | *PASSF4 | |
| *PASSF5 | | PCHBS | E3 |
| *PCHCE | | *PCHCI | |
| PCHCM | E3 | *PCHCS | |
| *PCHDF | | PCHDOC | E1A, Z |
| PCHFD | E3, H1 | PCHFE | E3 |
| PCHIA | E3, H2A1B2 | PCHIC | E1A |
| PCHID | E3, H2A1B2 | PCHIM | E1A |
| *PCHKT | E3 | *PCHNGS | |
| PCHSP | E1A | *PCHST | |
| *PCHSW | | PCOEF | K1A1A2 |
| PFQAD | H2A2A1, E3, K6 | *PGSF | |
| *PIMACH | | *PINITM | |
| *PJAC | | *PNNZRS | |
| POCH | C1, C7A | POCH1 | C1, C7A |
| POIS3D | I2B4B | *POISD2 | |
| *POISN2 | | *POISP2 | |
| POISTG | I2B4B | POLCOF | E1B |
| POLFIT | K1A1A2 | POLINT | E1B |
| POLYVL | E3 | *POS3D1 | |
| *POSTG2 | | *PPADD | |
| *PPGQ8 | | *PPGSF | |
| *PPPSF | | PPQAD | H2A2A1, E3, K6 |
| *PPSGF | | *PPSPF | |
| PPVAL | E3, K6 | *PROC | |
| *PROCP | | *PROD | |

| | | | |
|---|---|---|---|
| *PRODP | | *PRVEC | |
| *PRWPGE | | *PRWVIR | |
| *PSGF | | PSI | C7C |
| PSIFN | C7C | *PSIXN | |
| PVALUE | K6 | *PYTHAG | |
| QAG | H2A1A1 | QAGE | H2A1A1 |
| QAGI | H2A3A1, H2A4A1 | QAGIE | H2A3A1, H2A4A1 |
| QAGP | H2A2A1 | QAGPE | H2A2A1 |
| QAGS | H2A1A1 | QAGSE | H2A1A1 |
| QAWC | H2A2A1, J4 | QAWCE | H2A2A1, J4 |
| QAWF | H2A3A1 | QAWFE | H2A3A1 |
| QAWO | H2A2A1 | QAWOE | H2A2A1 |
| QAWS | H2A2A1 | QAWSE | H2A2A1 |
| QC25C | H2A2A2, J4 | QC25F | H2A2A2 |
| QC25S | H2A2A2 | *QCHEB | |
| *QELG | | *QFORM | |
| QK15 | H2A1A2 | QK15I | H2A3A2, H2A4A2 |
| QK15W | H2A2A2 | QK21 | H2A1A2 |
| QK31 | H2A1A2 | QK41 | H2A1A2 |
| QK51 | H2A1A2 | QK61 | H2A1A2 |
| QMOMO | H2A2A1, C3A2 | QNC79 | H2A1A1 |
| QNG | H2A1A1 | QPDOC | H2, Z |
| *QPSRT | | *QRFAC | |
| *QRSOLV | | *QS2I1D | N6A2A |
| *QS2I1R | N6A2A | *QWGTC | |
| *QWGTF | | *QWGTS | |
| QZHES | D4C1B3 | QZIT | D4C1B3 |
| QZVAL | D4C2C | QZVEC | D4C3 |
| R1MACH | R1 | *R1MPYQ | |
| *R1UPDT | | *R9AIMP | C10D |
| *R9ATN1 | C4A | *R9CHU | C11 |
| *R9GMIC | C7E | *R9GMIT | C7E |
| *R9KNUS | C10B3 | *R9LGIC | C7E |
| *R9LGIT | C7E | *R9LGMC | C7E |
| *R9LN2R | C4B | R9PAK | A6B |
| R9UPAK | A6B | *RADB2 | |
| *RADB3 | | *RADB4 | |
| *RADB5 | | *RADBG | |
| *RADF2 | | *RADF3 | |
| *RADF4 | | *RADF5 | |
| *RADFG | | RAND | L6A21 |
| RATQR | D4A5, D4C2A | RC | C14 |
| RC3JJ | C19 | RC3JM | C19 |
| RC6J | C19 | RD | C14 |
| REBAK | D4C4 | REBAKB | D4C4 |
| REDUC | D4C1C | REDUC2 | D4C1C |
| *REORT | | RF | C14 |
| *RFFTB | J1A1 | RFFTB1 | J1A1 |
| *RFFTF | J1A1 | RFFTF1 | J1A1 |
| *RFFTI | J1A1 | RFFTI1 | J1A1 |
| RG | D4A2 | RGAUSS | L6A14 |
| RGG | D4B2 | RJ | C14 |
| *RKFAB | | RPQR79 | F1A1A |
| RPZERO | F1A1A | RS | D4A1 |
| RSB | D4A6 | *RSCO | |
| RSG | D4B1 | RSGAB | D4B1 |
| RSGBA | D4B1 | RSP | D4A1 |
| RST | D4A5 | RT | D4A5 |
| RUNIF | L6A21 | *RWUPDT | |
| *S1MERG | | SASUM | D1A3A |
| SAXPY | D1A7 | SBCG | D2A4, D2B4 |

| | | | |
|---|---|---|---|
| SBHIN | N1 | SBOCLS | K1A2A, G2E, G2H1, G2H2 |
| SBOLS | K1A2A, G2E, G2H1, G2H2 | *SBOLSM | |
| SCASUM | D1A3A | SCG | D2B4 |
| SCGN | D2A4, D2B4 | SCGS | D2A4, D2B4 |
| SCHDC | D2B1B | SCHDD | D7B |
| SCHEX | D7B | *SCHKW | R2 |
| SCHUD | D7B | *SCLOSM | |
| SCNRM2 | D1A3B | *SCOEF | |
| SCOPY | D1A5 | SCOPYM | D1A5 |
| SCOV | K1B1 | SCPPLT | N1 |
| *SDAINI | | *SDAJAC | |
| *SDANRM | | *SDASLV | |
| SDASSL | I1A2 | *SDASTP | |
| *SDATRP | | *SDAWTS | |
| *SDCOR | | *SDCST | |
| *SDNTL | | *SDNTP | |
| SDOT | D1A4 | *SDPSC | |
| *SDPST | | SDRIV1 | I1A2, I1A1B |
| SDRIV2 | I1A2, I1A1B | SDRIV3 | I1A2, I1A1B |
| *SDSCL | | SDSDOT | D1A4 |
| *SDSTP | | *SDZRO | |
| SEPELI | I2B1A2 | SEPX4 | I2B1A2 |
| SGBCO | D2A2 | SGBDI | D3A2 |
| SGBFA | D2A2 | SGBMV | D1B4 |
| SGBSL | D2A2 | SGECO | D2A1 |
| SGEDI | D2A1, D3A1 | SGEEV | D4A2 |
| SGEFA | D2A1 | SGEFS | D2A1 |
| SGEIR | D2A1 | SGEMM | D1B6 |
| SGEMV | D1B4 | SGER | D1B4 |
| SGESL | D2A1 | SGLSS | D9, D5 |
| SGMRES | D2A4, D2B4 | SGTSL | D2A2A |
| *SHELS | D2A4, D2B4 | *SHEQR | D2A4, D2B4 |
| SINDG | C4A | SINQB | J1A3 |
| SINQF | J1A3 | SINQI | J1A3 |
| SINT | J1A3 | SINTI | J1A3 |
| SINTRP | I1A1B | SIR | D2A4, D2B4 |
| SLLTI2 | D2E | SLPDOC | D2A4, D2B4, Z |
| *SLVS | | *SMOUT | |
| SNBCO | D2A2 | SNBDI | D3A2 |
| SNBFA | D2A2 | SNBFS | D2A2 |
| SNBIR | D2A2 | SNBSL | D2A2 |
| SNLS1 | K1B1A1, K1B1A2 | SNLS1E | K1B1A1, K1B1A2 |
| SNRM2 | D1A3B | SNSQ | F2A |
| SNSQE | F2A | *SODS | |
| SOMN | D2A4, D2B4 | *SOPENM | |
| *SORTH | D2A4, D2B4 | SOS | F2A |
| *SOSEQS | | *SOSSOL | |
| SPBCO | D2B2 | SPBDI | D3B2 |
| SPBFA | D2B2 | SPBSL | D2B2 |
| *SPELI4 | | *SPELIP | |
| SPENC | C5 | *SPIGMR | D2A4, D2B4 |
| *SPINCW | | *SPINIT | |
| SPLP | G2A2 | *SPLPCE | |
| *SPLPDM | | *SPLPFE | |
| *SPLPFL | | *SPLPMN | |
| *SPLPMU | | *SPLPUP | |
| SPOCO | D2B1B | SPODI | D2B1B, D3B1B |
| SPOFA | D2B1B | SPOFS | D2B1B |
| SPOIR | D2B1B | *SPOPT | |
| SPOSL | D2B1B | SPPCO | D2B1B |
| SPPDI | D2B1B, D3B1B | SPPERM | N8 |

```
*WNLT2
 WNNLS        K1A2A
 XADJ         A3D
 XCON         A3D
 XERCLR       R3C
 XERDMP       R3C
 XERMAX       R3C
*XERPRN       R3C
 XGETF        R3C
 XGETUN       R3C
 XNRMP        C3A2, C9
*XPMUP        C3A2, C9
*XPQNU        C3A2, C9
*XQMU         C3A2, C9
 XRED         A3D
 XSETF        R3A
 XSETUN       R3B
*ZABS
*ZACON
*ZASYI
 ZBESI        C10B4
 ZBESK        C10B4
*ZBINU
*ZBKNU
*ZBUNK
*ZEXP
*ZLOG
*ZMLT
*ZS1S2
*ZSHCH
*ZUCHK
*ZUNI1
*ZUNIK
*ZUNK2
*ZWRSK
*WNLT3
 XADD         A3D
 XC210        A3D
*XERBLA       R3
*XERCNT       R3C
*XERHLT       R3C
 XERMSG       R3C
*XERSVE       R3
 XGETUA       R3C
 XLEGF        C3A2, C9
*XPMU         C3A2, C9
*XPNRM        C3A2, C9
*XPSI         C7C
*XQNU         C3A2, C9
 XSET         A3D
 XSETUA       R3B
*YAIRY
*ZACAI
 ZAIRY        C10D
 ZBESH        C10A4
 ZBESJ        C10A4
 ZBESY        C10A4
 ZBIRY        C10D
*ZBUNI
*ZDIV
*ZKSCL
*ZMLRI
*ZRATI
*ZSERI
*ZSQRT
*ZUNHJ
*ZUNI2
*ZUNK1
*ZUOIK
```