



CONVEX SCILIB

Quick Reference

Document No. 710-014330-002

First Edition

August 1991

CONVEX Computer Corporation
Richardson, Texas USA

CONVEX SCILIB Quick Reference
Order No. DSW-361
First Edition

© 1991 CONVEX Computer Corporation
All rights reserved.

This document is copyrighted. This document may not, in whole or part, be copied, duplicated, reproduced, translated, stored electronically, or reduced to machine-readable form without prior written consent from CONVEX Computer Corporation.

Although the material contained herein has been carefully reviewed, CONVEX Computer Corporation (CONVEX) does not warrant it to be free of errors or omissions. CONVEX reserves the right to make corrections, updates, revisions or changes to the information contained herein. CONVEX does not warrant the material described herein to be free of patent infringement.

CONVEX, the CONVEX logo ("C"), and C200 Series architecture are registered trademarks of CONVEX Computer Corporation.

Printed in the United States of America

Contents

1 Quick Reference Explanation	1
2 Basic Vector Operations	3
3 Basic Matrix Operations	8
4 Linear Equations	13
5 Eigenvalues and Eigenvectors	16
6 Fast Fourier Transforms	17
7 Correlation and Convolution	19
8 Linear Recurrences	20
9 Miscellaneous Routines	22

The *SCILIB Quick Reference* provides a quick and efficient method for obtaining information on SCILIB. This reference lists subprogram names, purposes, and usage examples for all the subprograms in SCILIB.

This quick reference has the following format:

```

Subprogram Names
Function
Usage Example

```

Consider the following example:

```

MINV
Invert Matrix or Solve Linear Equations
INTEGER*8 n,ldab,m,job
REAL*8 ab(ldab,n+m),work(2*n),det,tol
CALL MINV (ab, n, ldab, work, det, tol, m, job)

```

Please note:

- The quick reference lists subprograms in the order that they appear in the *CONVEX SCILIB User's Guide*.
- The quick reference usage examples include only one subroutine type.
- The corresponding subprogram types in the usage examples must be substituted when using other types.

SCILIB PRODUCT

The SCILIB software library is a collection of FORTRAN-callable mathematical subprograms that provide a look-alike implementation of Cray Research Incorporated's UNICOS Math and Scientific Library. SCILIB contains all of the public domain routines found in the UNICOS library and all of the CRAY-specific routines.

Only SCILIB routines are enumerated in this quick reference. For information on VECLIB routines, see the *CONVEX VECLIB User's Guide*.

ACCESSING SCILIB

To incorporate SCILIB subprograms into your programs, simply include the appropriate declarations and **CALL** statements in your FORTRAN source program and specify that SCILIB be used as an object library at link time. To link SCILIB with your program, use the **-l** option on the *fc* command line:

```
fc [options] file [linker-options]
```

where [*linker-options*] includes the string:

```
-lscilib
```

If you use any VECLIB routines you'll need to include VECLIB8 also. In this case **-lveclib8** must *follow* SCILIB in the list of linker options. A typical *fc* command line might look like:

```
fc -cfc progr1.f -lscilib -lveclib8
```

The *-cfc* specifies Cray FORTRAN mode. For more information, see the *CONVEX SCILIB User's Guide*.

DEFAULT DATA TYPES

In Cray FORTRAN, the single precision REAL and INTEGER data types are 64 bits in length; single precision COMPLEX and double precision REAL types are 128 bits in length. In CONVEX SCILIB, it is important to note that *only default single-precision Cray data type lengths are used*. Any REAL or INTEGER variables passed to SCILIB subprograms must be of length 8 bytes; any COMPLEX variables passed must be of length 16 bytes. For more information, see Table 1-1 in the *CONVEX SCILIB User's Guide*.

This section lists subprograms for performing dense and sparse vector operations. These subprograms include some BLAS and BLAS extensions.

CLUSEQ/CLUSNE/CLUSFxx/CLUSLxx

(xx = GE, GT, LE, or LT)

Find Clusters of Selected Vector Elements

INTEGER*8 n, incx, a, indx, nindx

REAL*8 x(lenx), a

CALL CLUSEQ(n, x, incx, a, indx, nindx)

GATHER

Gather Sparse Vector

INTEGER*8 m, indx(m)

REAL*8 y(n), x(m)

CALL GATHER(m, x, y, indx)

ILLZ

Count Initial Zero Elements

INTEGER*8 i, ILLZ, n, x(lenx), incx

i = ILLZ(n, x, incx)

ILLZ

Count Initial Positive Elements

INTEGER*8 i, ILLZ, n, x(lenx), incx

i = ILLZ(n, x, incx)

ILSUM

Count TRUE Vector Elements

INTEGER*8 i, ILSUM, n, incx

LOGICAL*8 x(lenx)

i = ILSUM(n, x, incx)

INFLMAX

Index of Maximum Element of Vector

INTEGER*8 i, INFLMAX, n, x(lenx), incx,

mask, rshift

i = INFLMAX(n, x, incx, mask, rshift)

INFLMIN

Index of Minimum Element of Vector

INTEGER*8 i, INFLMIN, n, x(lenx), incx,
mask, rshift

i = INFLMIN(n, x, incx, mask, rshift)

ISAMAX/ICAMAX

Index of Maximum of Magnitudes

INTEGER*8 i, ISAMAX, n, incx
REAL*8 x(lenx)

i = ISAMAX(n, x, incx)

ISAMIN

Index of Minimum of Magnitudes

INTEGER*8 i, ISAMIN, n, incx
REAL*8 x(lenx)

i = ISAMIN(n, x, incx)

ISMAX/INTMAX

Index of Maximum Element of Vector

INTEGER*8 i, ISMAX, n, incx
REAL*8 x(lenx)

i = ISMAX(n, x, incx)

ISMIN/INTMIN

Index of Minimum Element of Vector

INTEGER*8 i, ISMIN, n, incx
REAL*8 x(lenx)

i = ISMIN(n, x, incx)

ISRCHEQ/ISRCHNE/ISEARCH/ISRCHF_{xx}/ISRCHL_{xx}

(xx = GE, GT, LE, or LT)

Search Vector for Element

INTEGER*8 i, ISRCHEQ, n, incx
REAL*8 x(lenx), a

i = ISRCHEQ(n, x, incx, a)

Basic Vector Operations

ISRCHMxx

(xx = EQ, GE, GT, LE, LT, or NE)

Search Vector for Element

```
INTEGER*8 i, ISRCHMxx, n, x(lenx), incx, a,  
          mask, rshift
```

```
i = ISRCHMxx(n, x, incx, a, mask, rshift)
```

OSRCHF/OSRCHI

Search Ordered Vector for Element

```
INTEGER*8 n, incx, indx, ifound, iwould, icount
```

```
REAL*8 x(lenx), a
```

```
CALL OSRCHF(n, x, incx, a ifound, iwould, icount)
```

OSRCHM

Search Ordered Vector for Element

```
INTEGER*8 n, x(lenx), incx, a, mask, rshift,  
          ifound, iwould, icount
```

```
CALL OSRCHM(n, x, incx, a, mask, rshift,  
            ifound, iwould, icount)
```

SASUM/SCASUM

Sum of Magnitudes of the Elements of a Vector

```
INTEGER*8 n, incx
```

```
REAL*8 s, SASUM, x(lenx)
```

```
s = SASUM (n, x, incx)
```

SAXPY/CAXPY

Elementary Vector Operation

```
INTEGER*8 n, incx, incy
```

```
REAL*8 a, x(lenx), y(leny)
```

```
CALL SAXPY (n, a, x, incx, y, incy)
```

SCATTER

Scatter Sparse Vector

```
INTEGER*8 m, indx(m)
```

```
REAL*8 y(n), x(m)
```

```
CALL SCATTER(m, y, indx, x)
```

SCOPY/CCOPY

Copy Vector

```

INTEGER*8 n,incx,incy
REAL*8 x(lenx),y(leny)
CALL SCOPY (n, x, incx, y, incy)

```

SDOT/CDOTC/CDOTU

Dot Product of Two Vectors

```

INTEGER*8 n,incx,incy
REAL*8 s,SDOT,x(lenx),y(leny)
s = SDOT (n, x, incx, y, incy)

```

SNRM2/SCNM2

Euclidean Norm of a Vector

```

INTEGER*8 n,incx
REAL*8 s,SNRM2,x(lenx)
s = SNRM2 (n, x, incx)

```

SPAXPY

Sparse Elementary Vector Operation

```

INTEGER*8 m, indx(m)
REAL*8 a, x(m), y(n)
CALL SPAXPY(m, a, x, y, indx)

```

SPDOT

Sparse Dot Product

```

INTEGER*8 m, indx(m)
REAL*8 s, SPDOT, y(n), x(m)
s = SPDOT(m, y, indx, x)

```

SROT/CROT/CSROT

Apply a Givens Rotation to Two Vectors

```

INTEGER*8 n,incx,incy
REAL*8 x(lenx),y(leny),c,s
CALL SROT (n, x, incx, y, incy, c, s)

```

SROTG/CROTG

Construct a Givens Rotation

```

REAL*8
CALL SROTG (a, b, c, s)

```

Basic Vector Operations

SROTM

Apply a Modified Givens Rotation

INTEGER*8 n,incx,incy

REAL*8 x(lenx),y(leny),param(5)

CALL SROTM (n, x, incx, y, incy, param)

SROTMG

Construct a Modified Givens Rotation

REAL*8

CALL SROTMG (d1, d2, x1, y1, param)

SSCAL/CSCAL/CSSCAL

Scale a Vector

INTEGER*8 n,incx

REAL*8 a,x(lenx)

CALL SSCAL (n, a, x, incx)

SSUM/CSUM

Sum of the Elements of a Vector

INTEGER*8 n,incx

REAL*8 s,SSUM,x(lenx)

s = SSUM (n, x, incx)

SSWAP/CSWAP

Swap Two Vectors

INTEGER*8 n,incx,incy

REAL*8 x(lenx),y(leny)

CALL SSWAP (n, x, incx, y, incy)

WHENEQ/WHENNE/WHENFxx/WHENLxx

(xx = GE, GT, LE, or LT)

Find Selected Vector Elements

INTEGER*8 n, incx, indx(n), nindx

REAL*8 x(lenx), a

CALL WHENEQ(n, x, incx, a, indx, nindx)

WHENMxx

(xx = EQ, GE, GT, LE, LT, or NE)

Find Selected Vector Elements

INTEGER*8 n, x(lenx), incx, a, indx(n), nind,
mask, rshift

CALL WHENMxx(n, x, incx, a, indx, nindx,
mask, rshift)

3**Basic Matrix Operations**

This section lists the subprograms perform basic matrix operations. They are functionally equivalent to the Extended BLAS subprograms found in Chapter 3 of the *VECLIB User's Guide* in that they produce equivalent answers for the same problems, but with different usage.

MXM

Specialized Matrix-Matrix Multiply

```
INTEGER*8 m, k, n
REAL*8 a(m, k), b(k,n), c(m,n)
CALL MXM(a, m, b, k, c, n)
```

MXMA

Generalized Matrix-Matrix Multiply

```
INTEGER*8 ia, ja, ib, jb, ic, jc, m, k, n
REAL*8 a(lena), b(lenb), c(lenc)
CALL MXMA(a, ia, ja, b, ib, jb, c, ic, jc, m, k, n)
```

MXV

Specialized Matrix-Vector Multiply

```
INTEGER*8 m, n
REAL*8 a(m,n), x(n), y(m)
CALL MXV(a, m, x, n, y)
```

MXVA

Generalized Matrix-Vector Multiply

```
INTEGER*8 ia, ja, incx, m, n
REAL*8 a(lena), x(lenx), y(leny)
CALL MXVA(a, ia, ja, x, incx, y, incy, m, n)
```

SGBMV/CGBMV

General Band Matrix-Vector Multiply

```
CHARACTER*1 trans
INTEGER*8 m,n,ml,mu,lda,incx,incy
REAL*8 alpha,beta,a(lda,n),x(lenx),y(leny)
CALL SGBMV (trans,m,n,ml,mu,alpha,a,lda,x,
            incx,beta,y,incy)
```

Basic Matrix Operations

SGEMMS/CGEMMS

Strassen Matrix-Matrix Multiply

CHARACTER*1 transa, transb

INTEGER*8 m, n, k, lda, ldb, ldc

REAL*8 alpha, beta, a(lda, *), b(ldb, *),
c(ldc, n), work(lwork)

CALL SGEMMS(transa, transb, m, n, k, alpha,
a, lda, b, ldb, beta, c, ldc, work)

SGEMV/CGEMV

General Matrix-Vector Multiply

CHARACTER*1 trans

INTEGER*8 m,n,lda,incx,incy

REAL*8 alpha,beta,a(lda,n),x(lenx),y(leny)

CALL SGEMV (trans, m, n, alpha, a, lda, x,
incx, beta, y, incy)

SGER/CGERC/CGERU

General Rank-1 Update

INTEGER*8 m,n,lda,incx,incy

REAL*8 alpha,a(lda,n),x(lenx),y(leny)

CALL SGER (m, n, alpha, x, incx, y, incy, a, lda)

SMXPY

Matrix-Vector Multiply and Add

INTEGER*8 m, n, lda

REAL*8 a(lda, n), x(n), y(m)

CALL SMXPY(m, y, n, lda, x, a)

SSBMV/CHBMV

Symmetric or Hermitian Band Matrix-Vector Multiply

CHARACTER*1 uplo

INTEGER*8 n,m,lda,incx,incy

REAL*8 alpha,beta,a(lda,n),x(lenx),y(leny)

CALL SSBMV (uplo, n, m, alpha, a, lda, x,
incx, beta, y, incy)

SSPMV/CHPMV

Symmetric or Hermitian Matrix-Vector Multiply

CHARACTER*1 uplo

INTEGER*8 n,incx,incy

REAL*8 alpha,beta,ap(lenap),x(lenx),y(leny)

CALL SSPMV (uplo, n, alpha, ap, x, incx,
beta, y, incy)

SSPR/CHPR

Symmetric Rank-1 Update

CHARACTER*1 uplo

INTEGER*8 n,incx

REAL*8 alpha,ap(lenap),x(lenx)

CALL SSPR (uplo, n, alpha, x, incx, ap)

SSPR2/CHPR2

Symmetric or Hermitian Rank-2 Update

CHARACTER*1 uplo

INTEGER*8 n,incx,incy

REAL*8 alpha,ap(lenap),x(lenx),y(leny)

CALL SSPR2 (uplo, n, alpha, x, incx, y, incy, ap)

SSYMM/CHEMM/CSYMM/ZHEMM,

Symmetric or Hermitian Matrix-Matrix Multiply

CHARACTER*1 side,uplo

INTEGER*8 m,n,lda,ldb,ldc

REAL*8 alpha,beta,a(lda,*),b(ldb,*),c(ldc,*)

CALL SSYMM (side, uplo, m, n, alpha, a, lda, b,
ldb, beta, c, ldc)**SSYMV/CHEMV**

Symmetric or Hermitian Matrix-Vector Multiply

CHARACTER*1 uplo

INTEGER*8 n,lda,incx

REAL*8 alpha,a(lda,n),x(lenx)

CALL SSYR (uplo, n, alpha, x, incx, a, lda)

SSYR2/CHER2

Symmetric or Hermitian Rank-2 Update

CHARACTER*1 uplo

INTEGER*8 n,lda,incx,incy

REAL*8 alpha,a(lda,n),x(lenx),y(leny)

CALL SSYR2 (uplo, n, alpha, x, incx, y,
incy, a, lda)**SSYR2K/CSYR2K**

Symmetric or Hermitian Rank-2k Update

CHARACTER*1 uplo,trans

INTEGER*8 n,k,lda,ldb,ldc

REAL*8 alpha,beta,a(lda,*),b(ldb,*),c(ldc,*)

CALL SSYR2K (uplo, trans, n, k, alpha, a, lda,
b, ldb, beta, c, ldc)

Basic Matrix Operations

SSYRK/CSYRK

Symmetric or Hermitian Rank-k Update

CHARACTER*1 uplo,trans

INTEGER*8 n,k,lda,ldc

REAL*8 alpha,beta,a(lda,*),c(ldc,*)

CALL SSYRK (uplo, trans, n, k, alpha, a, lda, beta, c, ldc)

STBMV/CTBMV

Triangular Band Matrix-Vector Multiply

CHARACTER*1 uplo,trans,diag

INTEGER*8 n,m,ldt,incx

REAL*8 t(ldt,n),x(lenx)

CALL STBMV (uplo, trans, diag, n, m, t, ldt, x, incx)

STBSV/CTBSV

Solve Triangular Band System

CHARACTER*1 uplo,trans,diag

INTEGER*8 n,m,ldt,incx

REAL*8 t(ldt,n),x(lenx)

CALL STBSV (uplo, trans, diag, n, m, t, ldt, x, incx)

STPMV/CTPMV

Triangular Matrix-Vector Multiply

CHARACTER*1 uplo,trans,diag

INTEGER*8 n,incx

REAL*8 tp(lentp),x(lenx)

CALL STPMV (uplo, trans, diag, n, tp, x, incx)

STPSV/CTPSV

Solve One Triangular System

CHARACTER*1 uplo,trans,diag

INTEGER*8 n,incx

REAL*8 tp(lentp),x(lenx)

CALL STPSV (uplo, trans, diag, n, tp, x, incx)

STRMM/CTRMM

Triangular Matrix-Matrix Multiply

CHARACTER*1 side,uplo,transt,diag

INTEGER*8 m,n,ldt,ldb

REAL*8 alpha,t(ldt,*),b(ldb,*)

CALL STRMM (side, uplo, transt, diag, m, n,
alpha, t, ldt, b, ldb)

STRMV/CTRMV

Triangular Matrix-Vector Multiply

CHARACTER*1 uplo,trans,diag

INTEGER*8 n,ldt,incx

REAL*8 t(ldt,n),x(lenx)

CALL STRMV (uplo, trans, diag, n, t, ldt, x, incx)

STRSM/CTRSM

Solve Simultaneous Triangular Systems

CHARACTER*1 side,uplo,transt,diag

INTEGER*8 m,n,ldt,ldb

REAL*8 alpha,t(ldt,*),b(ldb,*)

CALL STRSM (side, uplo, transt, diag, m, n, alpha,
t, ldt, b, ldb)

STRSV/CTRSV

Solve One Triangular System

CHARACTER*1 uplo,trans,diag

INTEGER*8 n,ldt,incx

REAL*8 t(ldt,n),x(lenx)

CALL STRSV (uplo, trans, diag, n, t, ldt, x, incx)

SXMPY

Matrix-Vector Multiply and Add

INTEGER*8 n,m,lda,incx,incy

REAL*8 a(lda,n),x(lenx),y(leny)

CALL SXMPY (n,incy,y,m,incx,x,lda,a)

XERBLA

BLAS Error Handler

CHARACTER*6 name

INTEGER*8 larg

CALL XERBLA (name, larg)

This section lists the subprograms for solving linear equations contained in the *SCILIB User's Guide*.

MINV

Invert Matrix or Solve Linear Equations

```
INTEGER*8 n, ldab, m, job
REAL*8 ab(ldab, n+m), work(2*n), det, tol
CALL MINV(ab, n, ldab, work, det, tol, m, job)
```

OPFILT

Solve Symmetric Toeplitz Linear Equations

```
INTEGER*8 n
REAL*8 x(n), b(n), work(2*n), q(n)
CALL OPFILT(n, x, b, work, q)
```

SGBCO/CGBCO

Factor a General Band Matrix and Estimate its Condition Number

```
INTEGER*8 lda, n, ml, mu, ipvt(n)
REAL*8 a(lda, n), rcond, work(n)
CALL SGBCO (a, lda, n, ml, mu, ipvt, rcond, work)
```

SGBDI/CGBDI

Determinant of a General Band Matrix

```
INTEGER*8 lda, n, ml, mu, ipvt(n)
REAL*8 a(lda, n), det(2)
CALL SGBDI (a, lda, n, ml, mu, ipvt, det)
```

SGBFA/CGBFA

Factor a General Band Matrix

```
INTEGER*8 lda, n, ml, mu, ipvt(n), ier
REAL*8 a(lda, n)
CALL SGBFA (a, lda, n, ml, mu, ipvt, ier)
```

SGBSL/CGBSL

Solve Linear Equations with a General Band Matrix

INTEGER*8 lda,n,ml,mu,ipvt(n),job

REAL*8 a(lda,n),b(n)

CALL SGBSL (a, lda, n, ml, mu, ipvt, b, job)

SGECO/CGECO

Factor a General Matrix and Estimate its Condition Number

INTEGER*8 lda,n,ipvt(n)

REAL*8 a(lda,n),rcond,work(n)

CALL SGEKO (a, lda, n, ipvt, rcond, work)

SGEDI/CGEDI

Determinant and Inverse of a General Matrix

INTEGER*8 lda,n,ipvt(n),job

REAL*8 a(lda,n),det(2),work(n)

CALL SGEDI (a, lda, n, ipvt, det, work, job)

SGEFA/CGEFA

Factor a General Matrix

INTEGER*8 lda,n,ipvt(n),ier

REAL*8 a(lda,n)

CALL SGEFA (a, lda, n, ipvt, ier)

SGESL/CGESL

Solve Linear Equations with a General Matrix

INTEGER*8 lda,n,ipvt(n),job

REAL*8 a(lda,n),b(n)

CALL SGESL (a, lda, n, ipvt, b, job)

SGTSL/CGTSL

Solve Linear Equations with a Tridiagonal Matrix

INTEGER*8 n,ier

REAL*8 c(n),d(n),e(n),b(n)

CALL SGTSL (n, c, d, e, b, ier)

SPBCO/CPBCO

Factor a Positive Definite Band Matrix and Estimate its Condition Number

INTEGER*8 lda,n,m,ier

REAL*8 a(lda,n),rcond,work(n)

CALL SPBCO (a, lda, n, m, rcond, work, ier)

Linear Equations

SPBDI/CPBDI

Determinant of a Positive Definite Band Matrix

```
INTEGER*8 lda,n,m  
REAL*8 a(lda,n),det(2)  
CALL SPBDI (a, lda, n, m, det)
```

SPBFA/CPBFA

Cholesky Factorization of a Positive Definite Band Matrix

```
INTEGER*8 lda,n,m,ier  
REAL*8 a(lda,n)  
CALL SPBCO (a, lda, n, m, ier)
```

SPBSL/CPBSL

Solve Linear Equations with a Positive Definite Band Matrix

```
INTEGER*8 lda,n,m  
REAL*8 a(lda,n),b(n)  
CALL SPBSL (a, lda, n, m, b)
```

SPOCO/CPOCO

Factor a Positive Definite Matrix and Estimate
its Condition Number

```
INTEGER*8 lda,n,ier  
REAL*8 a(lda,n),rcond,work(n)  
CALL SPOCO (a, lda, n, rcond, work, ier)
```

SPODI/CPODI

Determinant and Inverse of a Positive Definite Matrix

```
INTEGER*8 lda,n,job  
REAL*8 a(lda,n),det(2)  
CALL SPODI (a, lda, n, det, job)
```

SPOFA/CPOFA

Cholesky Factorization of a Positive Definite Matrix

```
INTEGER*8 lda,n,ier  
REAL*8 a(lda,n)  
CALL SPOFA (a, lda, n, ier)
```

SPOSL/CPOSL

Solve Linear Equations with a Positive Definite Matrix

```
INTEGER*8 lda,n  
REAL*8 a(lda,n),b(n)  
CALL SPOSL (a, lda, n, b)
```

SPTSL/CPTSL

Solve Linear Equations with a Positive Definite
Tridiagonal Matrix

INTEGER*8 n

REAL*8 d(n),e(n-1),b(n)

CALL SPTSL (n, d, e, b)

This section lists the SCILIB subprograms to compute eigenvalues or eigenvalues and eigenvectors of matrices.

RS

Eigenvalues and Eigenvectors of a Real Symmetric Matrix

INTEGER*8 ldx,n,job,ier

REAL*8 a(ldax,n),w(n),x(ldax,n),work1(n),work2(n)

CALL RS (ldx, n, a, w, job, x, work1, work2, ier)

TQL2

Eigenvalues and Eigenvectors of a Real Symmetric Tridiagonal Matrix

INTEGER*8 ldx,n,ier

REAL*8 d(n),e(n),x(ldx,n)

CALL TQL2 (ldx, n, d, e, x, ier)

TQLRAT

Eigenvalues of a Real Symmetric Tridiagonal Matrix

INTEGER*8 n,ier

REAL*8 d(n),e2(n)

CALL TQLRAT (n, d, e2, ier)

TRED1

Reduce a Real Symmetric Matrix to Real Symmetric Tridiagonal Form

INTEGER*8 lda,n

REAL*8 a(lda,n),d(n),e(n),e2(n)

CALL TRED1 (lda, n, a, d, e, e2)

TRED2

Reduce a Real Symmetric Matrix to Real Symmetric Tridiagonal Form

INTEGER*8 ldx,n

REAL*8 a(ldax,n),d(n),e(n),x(ldax,n)

CALL TRED2 (ldax, n, a, d, e, x)

This section lists the SCILIB Fast Fourier transform (FFT) subprograms.

CFFT2

One-Dimensional Complex-to-Complex FFT

```

INTEGER*8 init, sign, l
COMPLEX*16 zin(1), zout(1)
REAL*8 work(5*1)
CALL CFFT2 (init, isgn, l, zin, work, zout)

```

CFTFAX/CFFTMLT

Simultaneous One-Dimensional FFT

```

INTEGER*8 work1(19), incl, incn, l, n, isgn
REAL*8 x(lenxy), y(lenxy), work1(4*1*n), work2(2*1)

```

Initialization call:

```
CALL CFTFAX(1, work3, work2)
```

Transform call:

```
CALL CFFTMLT(x, y, work1, work2, work3,
             incl, incn, l, n, isgn)
```

CRFFT2

Complex-to-Real One-Dimensional FFT

```

INTEGER*8 init, isgn, l
COMPLEX*16 zin(1/2+1)
REAL*8 work(3*1+4), xout(1)
CALL CRFFT2(init, isgn, l, zin, work, xout)

```

RCFFT2

Real-to Complex One-Dimensional FFT

```

INTEGER*8 init, isgn, l
REAL*8 zin(1) work(3*1+4)
COMPLEX*16 zout(1/2+1)
CALL RCFFT2(init, isgn, l, xin, work, zout)

```

Fast Fourier Transforms

FFTFAX/RFFTMLT

Simultaneous One-Dimensional FFT

INTEGER*8 work3(19), incl, incn, l, n, isgn

REAL*8 x(lenx), work1(2*1*n), work2(2*1)

Initialization call:

CALL FFTFAX(l, work3, work2)

Transform call:

CALL RFFTMLT(x, work1, work2, work3,
 incl, incn, l, n, isgn)

This section lists the SCILIB subprograms available for correlations, convolutions, and related operations, such as filtering by means of convolutions.

FILTERG

Discrete Correlation

```
INTEGER*8 m, n  
REAL*8 f(m), x(n), y(n-m+1)  
CALL FILTERG(f, m, x, n, y)
```

FILTERS

Discrete Correlation

```
INTEGER*8 m, n  
REAL*8 f((m+1)/2), x(n), y(n-m+1)  
CALL FILTERS(f, m, x, n, y)
```

This section lists the SCILIB subprograms available for solving a variety of linear recurrence operations.

FOLR/FOLRP

First Order Linear Recurrence

```
INTEGER*8 n, inca, incx
REAL*8 a(lena), x(lenx)
CALL FOLR(n, a, inca, x, incx)
```

FOLR2/FOLR2P

First Order Linear Recurrence

```
INTEGER*8 n, inca, incx
REAL*8 a(lena), x(lenx), y(leny)
CALL FOLR2(n, a, inca, x, incx, y, incy)
```

FOLRC

First Order Linear Recurrence

```
INTEGER*8 n, incx, inca
REAL*8 x(lenx), a(lenx), alpha
CALL FOLRC(n, x, incx, a, inca, alpha)
```

FOLRN/FOLRNP

Last Term of First Order Linear Recurrence

```
INTEGER*8 n, inca, incx
REAL*8 yn, FOLRN, a(lena), x(lenx)
yn = FOLRN(n, a, inca, x, incx)
```

RECPP

Compute Vector of Partial Products

```
INTEGER*8 n, incx, inca
REAL*8 x(lenx), a(lena)
CALL RECPP(n, x, incx, a, inca)
```

RECPS

Compute Vector of Partial Sums

```
INTEGER*8 n, incx, inca
REAL*8 x(lenx), a(lena)
CALL RECPS(n, x, incx, a, inca)
```

SOLR

Second Order Linear Recurrence

INTEGER*8 n, inca, incb, incx

REAL*8 a(lena), b(lenb), x(lenx)

CALL SOLR(n, a, inca, b, incb, x, incx)

SOLR3

Second Order Linear Recurrence

INTEGER*8 n, inca, incb, incx

REAL*8 a(lena), b(lenb), x(lenx)

CALL SOLR3(n, a, inca, b, incb, x, incx)

SOLRN

Last Term of Second Order Linear Recurrence

INTEGER*8 n, inca, incb, incx

REAL*8 xn, SOLRN, a(lena), b(lenb), x(lenx)

xn = SOLRN(n, a, inca, b, incb, x, incx)

This section lists SCILIB subprograms that perform a variety of operations.

ORDERS

Sort Array into Ascending Order

```
INTEGER*8 mode, work(lwork), indx(n), n, ldx,  
          keylen, iradix  
REAL*8 x(ldx, n)  
CALL ORDERS(mode, work, x, indx, n, ldx,  
            keylen, iradix)
```

XERSCI

SCILIB Error Handler

```
CHARACTER*1 name, message  
INTEGER*8 iarg  
CALL XERSCI(name, iarg, messag)
```



