NOTICE

PORTIONS OF THIS REPORT ARE ILLEGIBLE. It has been reproduced from the best available copy to permit the broadest possible availability.

Distribution Category:
Mathematics and Computers
(UC-32)

ANL-83-16

ANL--83-16

DE84 008852

ARGONNE NATIONAL LABORATORY
9700 South Cass Avenue
Argonne, IL 60439

THE TOEPLITZ PACKAGE USERS' GUIDE*

by

O. B. Arushanian, M. K. Samarin, V. V. Voevodin,** E. E. Tyrtyshnikov

Science Research Computing Center
Moscow State University
USSR

B. S. Garbow, J. M. Boyle, W. R. Cowell, K. W. Dritz

Mathematics and Computer Science Division
Argonne National Laboratory
USA

October 1983

*The work of the American participants was supported in part by the National Science Foundation and in part by the Applied Mathematical Sciences Research Program (KC-04-02) of the Office of Energy Research of the U.S. Department of Energy under Contract W-31-109-Eng-38. The work of the Soviet participants was supported by the State Committee for Science and Technology (U.S.S.R.).

**Permanent address: Academy of Sciences, State Committee for Science and Technology, U.S.S.R.

TABLE OF CONTENTS

CHAPTER 2: TOEPLITZ- AND CIRCULANT-TYPE MATRICES OF THE SECOND LEVEL

CHAPTER 3: TOEPLITZ- AND CIRCULANT-TYPE MATRICES OF THE THIRD LEVEL

CHAPTER 3 (cont'd)

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

THE TOEPLITZ PACKAGE USERS' GUIDE

O. B. Arushanian, M. K. Samarin, V. V. Voevodin, E. E. Tyrtyshnikov (USSR)
B. S. Garbow, J. M. Boyle, W. R. Cowell, K. W. Dritz (USA)

ABSTRACT

The TOEPLITZ package is a collection of Fortran subroutines for the numerical solution of systems of linear equations with coefficient matrices of Toeplitz or circulant form. This report provides a description of the algorithms and software in the package and includes program listings.

INTRODUCTION

1. Overview of the TOEPLITZ Package

The TOEPLITZ package is a collection of Fortran subroutines for solving linear systems

$$Ax = b ,$$

where A is a Toeplitz matrix (see subsection 1.1 of Chapter 1), a circulant matrix (see subsection 1.2 of Chapter 1), or has one of several block structures based on Toeplitz or circulant matrices. Included also is capability for orthogonal factorization of a column-circulant matrix (see subsection 1.4 of Chapter 1).

Such systems arise in problems of electrodynamics, acoustics, mathematical statistics, algebra, in the numerical solution of integral equations with a difference kernel, and in the theory of stationary time series and signals (see, e.g., [5,7,9,17,20,25,26]). Circulant matrices play an important role in the theory of circular convolutions [13]. Block-Toeplitz matrices have recently begun to play a significant role as the applicability of multichannel time series increases [22,30].

Although the theoretical and practical significance of Toeplitz matrices was recognized early in this century [23,28,31], computational aspects were not studied until more recently. The most influential and fundamental paper on algorithmic aspects was Levinson's extension to the discrete case of

Wiener's basic work on filtering [19,29]. It was here that the technique of bordering and recursion on the order of the system was first shown to be an effective way to produce efficient algorithms for Toeplitz systems. Levinson's algorithm is an $O(M^2)$ method for solving an order M positive-definite symmetric Toeplitz system of equations. Trench later used the same ideas to show how bordering could be exploited for general Toeplitz systems [24]. Trench's work was made more explicit and generalized by Zohar [32,33].

These $O(M^2)$ algorithms for Toeplitz systems are currently the most practical methods for such problems. They have simple descriptions as programs, they use simple storage and control structures, and error analyses are available for some of them [8,10,11].

The algorithms in this package for circulant matrices appear to have been known classically (see [13]). Toeplitz matrices of the second level are discussed in [4,21,22,27]; the algorithms are essentially the same as those in this package.

Toeplitz matrices arising in time series and signal processing are quite often covariance matrices that occur in normal equations for linear least-squares problems. The coefficient matrices in these problems often have column-circulant structures that lend themselves to efficient methods for problem solution by orthogonalization. These methods are usually called "lattice methods" in the signal processing literature [12,14,18]; one such method [12] is implemented in the TOEPLITZ package.

The TOEPLITZ package has an intentional similarity to LINPACK [15] in the format of the Fortran source, in the comments, and in the subroutine naming conventions. All names consist of four, five, or six letters (depending on the level of block structure of the matrix A) in the forms XSL#, XYSL#, or XYZSL# for the system solving subroutines and CQR# for the orthogonal factorization subroutines.* When A has no special block structure (see Chapter 1), the letter in the X position specifies the type of the matrix:

T Toeplitz
C Circulant.

---

*The one member not governed by the naming convention is the service subroutine SALWC (SALWZ in double precision), called by most of the two-level and all of the three-level system solving subroutines.

When A has a two-level block structure (see Chapter 2), the letters in the XY positions specify the type of the matrix:

TG Block-Toeplitz where the blocks are general matrices

CT Block-circulant where the blocks are Toeplitz matrices

CC Block-circulant where the blocks themselves are circulant matrices

CG Block-circulant where the blocks are general matrices.

When A has a three-level block structure (see Chapter 3), the letters in the XYZ positions specify the type of the matrix:

CTG Block-circulant where the blocks are two-level TG-type matrices

CCT Block-circulant where the blocks are two-level CT-type matrices

CCC Block-circulant where the blocks are two-level CC-type matrices

CCG Block-circulant where the blocks are two-level CG-type matrices.

By permuting corresponding rows and columns, one can transform any two-level XY-type matrix to YX-type (see Tyrtyshnikov [25]). Similarly, one can interchange any two levels of a three-level XYZ-type matrix. These circumstances effectively extend the capability of the TOEPLITZ package to additional matrix types.

The fixed letters SL indicate that the routine solves a linear system, while the letters QR indicate that the routine performs an orthogonal factorization.

The last letter in the # position specifies the matrix data type. Standard Fortran allows the use of three such types:

S REAL
D DOUBLE PRECISION
C COMPLEX.

In addition, some Fortran systems allow a double precision complex type:

Z DOUBLE COMPLEX.

2. The Leading Array Dimension Parameter

Those members of the TOEPLITZ package that process a two-dimensional array include in their calling sequences the parameter LDA (or LDQ,LDS) to communicate the leading dimension of the array. "Leading dimension" refers to the DIMENSION statement storage allocation for the array and should be distinguished from the order of the linear system. The inclusion of this parameter enables flexibility in processing systems of varying order without the bother of changing the DIMENSION statement for the coefficient matrix.

For example, if the array A has been declared "A(50,20)" in the DIMENSION statement, then simply enter the statement "LDA = 50" into the body of the program before the call to the TOEPLITZ package subroutine.

3. Development of the TOEPLITZ Package

In offering the TOEPLITZ package to the international computing community, it is appropriate to note that this software is the result of collaboration among scientists in the United States and the Soviet Union. Hence, in addition to the intrinsic usefulness of the package, the software in its present form demonstrates the possibilities inherent in Soviet-American collaboration in the development of scientific software. The work was carried out under the auspices of the agreement between the U.S.A. and the U.S.S.R. on Scientific and Technological Cooperation in the Field of Application of Computers to Economics and Management, subtopic Mathematical Software.

This collaborative effort was initiated at the Numerical Software Workshop which took place at the National Science Foundation (NSF) in Washington, D.C. in December of 1975. The general framework of joint efforts was discussed during that workshop by D. Aufenkamp of NSF, W. Cody of the Applied Mathematics Division, Argonne National Laboratory (AMD-ANL), and O. Arushanian of the Science Research Computing Center, Moscow State University (SRCC-MSU), then visiting Pennsylvania State University for the year. Further steps were discussed during a meeting which took place at Penn State in February of 1976 involving D. Aufenkamp (NSF), J. Boyle (AMD-ANL), W. Cowell (AMD-ANL), and O. Arushanian (SRCC-MSU), and during a short visit by O. Arushanian to J. Bunch, University of California at San Diego (UCSD). In accordance with plans agreed upon during these meetings and approved in the meeting of coordinators and experts on the topic "Theoretical Foundations of Software for Application in Economics and Management" which took place in Moscow in June of 1976, long-term visits of American scientists to the U.S.S.R. in 1976 and 1978 and of Soviet scientists to the U.S.A. in 1978 and 1979 were arranged to exchange information and to carry out joint work on numerical software development. These joint efforts came to be known as the SALAR (Soviet-American Libraries and Algorithms Research) project. Results of accomplished works have appeared in 25 papers (see [1] and [2]) and were presented at the IFIP Congress in August of 1977 in Toronto, Canada (see [3]).

The contributions from the U.S.A. side were made by J. Boyle, K. Dritz, W. Cowell, and B. Garbow of AMD-ANL (now redesignated MCS-ANL), J. Bunch of UCSD, D. Sorensen (now of MCS-ANL), W. Miller (now of the University of Arizona), and C. Moler of the University of New Mexico. The contributions from the U.S.S.R. side were made by V. Voevodin (now of the Academy of Sciences, State Committee for Science and Technology), O. Arushanian, M. Samarin, E. Nikolaev, V. Morozov, Y. Kuchevskiy, E. Tyrtyshnikov, N. Bogomolov, and V. Borisov of SRCC-MSU.

The SALAR project had a number of objectives. First of all, it represented joint research into the methodology and practical aspects of producing mathematical software, namely, numerical libraries and packages. This main objective dictated the necessity of also investigating systems aspects of mathematical software development, which include the study of transportability problems, tailoring of programs to user requests, abstract formulation of numerical algorithms, and program transformation and generation systems. Methodological questions associated with the joint systematization, testing, and certification of mathematical software packages were also of great importance in the SALAR project. Research in numerical algorithms development was conducted mostly in linear algebra on problems such as updating algorithms for matrix decomposition and solving special types of linear systems.

The TOEPLITZ package was produced as a part of the SALAR project and can be considered as a practical result of previous investigations. The routines were originally written in 1978 at Moscow State University by E. Tyrtyshnikov [25] on the basis of the theoretical results of W. Trench [24] and S. Voevodina [27], and on his own research. A preliminary version of the users' guide was written by Soviet and American scientists during a visit to Argonne National Laboratory (U.S.A.) made by Soviet scientists O. Arushanian and M. Samarin (of SRCC-MSU) in 1979. Multiple versions of TOEPLITZ subroutines and formatting of codes were obtained with the help of the TAMPR-system [3], produced by J. Boyle and K. Dritz of AMD-ANL. Modifications, commenting,

and test driver design were also accomplished during this Argonne visit. Scientific supervision over the development of the TOEPLITZ package at SRCC-MSU was provided by V. Voevodin.

Further developmental work on the codes and preparation of this users' guide were accomplished at Argonne in 1982. The added capability for orthogonalization of column-circulant matrices derives from a new algorithm of G. Cybenko [12] (of Tufts University). Cybenko also suggested an improved formulation of another of the algorithms, supplied background information included in the "Overview" section of this guide, and pointed us to many of the references.

In conclusion, we wish to acknowledge the support of the National Science Foundation (U.S.A.) and the State Committee for Science and Technology (U.S.S.R.), executors of the Science and Technology Agreement. Special thanks are due to D. Aufenkamp (U.S.A.), B. Rameev (U.S.S.R.), and Y. Baraboshkin (U.S.S.R.) who created conditions in which our joint work could flourish. We also express our great gratitude to Judy Beumer (of MCS-ANL) who carefully typed the manuscript for this users' guide.

4. Availability of the TOEPLITZ Package

The TOEPLITZ package is available on tape from the following sources.

National Energy Software Center		IMSL, Inc.
Argonne National Laboratory	or	Sixth Floor, NBC Bldg.
9700 South Cass Avenue		7500 Bellaire Blvd.
Argonne, IL 60439		Houston, TX 77036-5085
Phone: (312) 972-7250		Phone: (713) 772-1927

The package includes both single precision and double precision versions of the programs, and testing aids are also provided on the tape (see The TOEPLITZ Package Implementation Guide, ANL-83-17).

Comments and questions regarding the TOEPLITZ package should be directed to

Burton S. Garbow
Mathematics and Computer Science Division
Argonne National Laboratory
9700 South Cass Avenue
Argonne, IL 60439
Phone: (312) 972-7184

CHAPTER 1: TOEPLITZ AND CIRCULANT MATRICES

1. Structure and Representation

1.1. Toeplitz matrices (T-matrices)

A Toeplitz matrix, or T-matrix, A is a real or complex square matrix whose elements along the main diagonal and along each co-diagonal are equal; thus A has the representation

$$A = \begin{pmatrix} a_0 & a_1 & a_2 & \cdots & a_{M-1} \\ a_{-1} & a_0 & a_1 & \cdots & a_{M-2} \\ a_{-2} & a_{-1} & a_0 & \cdots & a_{M-3} \\ \cdots & \cdots & \cdots & \cdots & \\ a_{-M+1} & a_{-M+2} & a_{-M+3} & \cdots & a_0 \end{pmatrix} .$$

A T-matrix is completely specified by its first row and column.

In the TOEPLITZ package a T-matrix of order M is represented by a singly subscripted array of 2*M-1 elements which contains the first row of the matrix followed by its first column beginning with the second element:

$$a_0, a_1, a_2, \ldots, a_{M-1}, a_{-1}, a_{-2}, \ldots, a_{-M+1} .$$

1.2. Circulant matrices (C-matrices)

A circulant matrix, or C-matrix, A is a T-matrix, limited here to complex mode, with the further property that

$$a_{-i} = a_{M-i} , \quad i = 1,2,\ldots,M-1 ;$$

thus A has the representation

$$A = \begin{pmatrix} a_0 & a_1 & a_2 & \cdots & a_{M-1} \\ a_{M-1} & a_0 & a_1 & \cdots & a_{M-2} \\ a_{M-2} & a_{M-1} & a_0 & \cdots & a_{M-3} \\ \cdot & \cdot & \cdot & \cdots & \cdot \\ a_1 & a_2 & a_3 & \cdots & a_0 \end{pmatrix} .$$

A C-matrix is completely specified by its first row; each further row may be obtained from the previous one by a right cyclic shift.

In the TOEPLITZ package a C-matrix of order M is represented by a singly subscripted array of M elements which contains the first row of the matrix:

$$a_0, a_1, a_2, \ldots, a_{M-1} .$$

1.3. General matrices (G-matrices)

A general real or complex square matrix

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1M} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2M} \\ a_{31} & a_{32} & a_{33} & \cdots & a_{3M} \\ \cdot & \cdot & \cdot & \cdots & \cdot \\ a_{M1} & a_{M2} & a_{M3} & \cdots & a_{MM} \end{pmatrix}$$

will be called a G-matrix.

In the TOEPLITZ package a G-matrix of order M is represented by a singly subscripted array of M**2 elements which contains the successive columns of the matrix:

$$a_{11}, a_{21}, a_{31}, \ldots, a_{M1}, a_{12}, a_{22}, a_{32}, \ldots, a_{M2}, \ldots, a_{1M}, a_{2M}, a_{3M}, \ldots, a_{MM} .$$

1.4. Column-circulant matrices

The designation "column-circulant" will be given to a real or complex rectangular matrix A, with row order M at least equal to its column order L, whose first column is specified and each further column obtained from its predecessor by a downward cyclic shift; thus A has the representation

$$A = \begin{pmatrix} a_0 & a_{M-1} & a_{M-2} & \cdots & a_{M-L+1} \\ a_1 & a_0 & a_{M-1} & \cdots & a_{M-L+2} \\ a_2 & a_1 & a_0 & \cdots & a_{M-L+3} \\ \vdots & \vdots & \vdots & & \vdots \\ a_{M-1} & a_{M-2} & a_{M-3} & \cdots & a_{M-L} \end{pmatrix} .$$

In the TOEPLITZ package a column-circulant matrix with M rows is represented by a singly subscripted array of M elements which contains the first column of the matrix:

$$a_0, a_1, a_2, \ldots, a_{M-1} .$$

2. Solution of Linear Equations with T-Matrices

2.1. Purpose

The TOEPLITZ subroutines in this section are designed to solve linear algebraic equations with T-matrices. Usage will be described for the single precision real version. Double precision, complex, and double precision complex versions are also available. Indeed, the complex version is called in solving two-level CT-matrix systems (see subsection 3.5 of Chapter 2).

2.2. Usage

Single precision real T-matrices. TSLS solves a linear system with a real Toeplitz matrix. The calling sequence is

```
CALL TSLS(A,X,R,M) .
```

On entry,

A   is a singly subscripted array of 2*M-1 elements which contains the first row of the T-matrix followed by its first column beginning with the second element. A is unaltered by TSLS.

X is a singly subscripted array of M elements which contains the right hand side of the system.

R is a singly subscripted array of 2*M-2 elements used for work space.

M is the order of A and the number of elements in X.

On return,

X contains the solution of the system.

Double precision real T-matrices. The calling sequence of the double precision real T-matrix subroutine TSLD is the same as that of TSLS with A, X, and R DOUBLE PRECISION variables.

Single precision complex T-matrices. The calling sequence of the single precision complex T-matrix subroutine TSLC is the same as that of TSLS with A, X, and R COMPLEX variables.

Double precision complex T-matrices. In those computing systems where it is available, the calling sequence of the double precision complex T-matrix subroutine TSLZ is the same as that of TSLS with A, X, and R DOUBLE COMPLEX variables.

2.3. Example

The following program segment illustrates the use of the single precision subroutine TSLS for real T-matrices. Examples of the use of TSLD, TSLC, and TSLZ could be obtained by changing the subroutine name and type declaration. The system is of order 4 with coefficients as follows.

$$A = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 5 & 1 & 2 & 3 \\ 6 & 5 & 1 & 2 \\ 7 & 6 & 5 & 1 \end{pmatrix} \qquad X = \begin{pmatrix} 10 \\ 11 \\ 14 \\ 19 \end{pmatrix}$$

```
      REAL A(7),X(4),R(6)
      INTEGER M,I
      DATA A(1)/1.0/,A(2)/2.0/,A(3)/3.0/,A(4)/4.0/,
     *     A(5)/5.0/,A(6)/6.0/,A(7)/7.0/
      DATA X(1)/10.0/,X(2)/11.0/,X(3)/14.0/,X(4)/19.0/
      M = 4
      CALL TSLS(A,X,R,M)
      DO 10 I = 1, M
         WRITE(...,...) X(I)
   10 CONTINUE
      STOP
      END
```

The solution of the system is

$$X = (1.0,1.0,1.0,1.0) .$$

2.4. Algorithm

The algorithm for the solution of a system of linear algebraic equations

$$Ax = b \tag{1}$$

with a T-matrix A of order M comprises a sequence of M steps. At the (k+1)-st step the solution of the system

$$A_k y_k = d_k \tag{2}$$

is determined. Here

$$A = \begin{pmatrix} a_0 & a_1 & \cdots & a_k \\ a_{-1} & a_0 & \cdots & a_{k-1} \\ a_{-2} & a_{-1} & \cdots & a_{k-2} \\ \cdots & \cdots & \cdots & \cdots \\ a_{-k} & a_{-k+1} & \cdots & a_0 \end{pmatrix}, \quad y_k = \begin{pmatrix} y_{0,k} \\ y_{1,k} \\ y_{2,k} \\ \vdots \\ y_{k,k} \end{pmatrix}, \quad d_k = \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ \vdots \\ b_k \end{pmatrix} .$$

The vector y_k is calculated by recurrence from y_{k-1}. The final result of the recurrent process is the solution of system (1), namely, $x = y_{M-1}$.

At step 1, $y_0 = b_0/a_0$. At step k+1, let us consider the unknown vector y_k to be the sum of two vectors, one of which, augmented by a zero, was determined at the k-th step:

$$\begin{pmatrix} y_{0,k} \\ y_{1,k} \\ \vdots \\ y_{k-1,k} \\ y_{k,k} \end{pmatrix} = \begin{pmatrix} y_{0,k-1} \\ y_{1,k-1} \\ \vdots \\ y_{k-1,k-1} \\ 0 \end{pmatrix} + \begin{pmatrix} z_{0,k} \\ z_{1,k} \\ \vdots \\ z_{k-1,k} \\ z_{k,k} \end{pmatrix} . \tag{3}$$

Substituting this sum into equation (2) and taking into account that the vector y_{k-1} satisfies the equation

$$A_{k-1}y_{k-1} = d_{k-1} ,$$

we see that the unknown vector z_k from (3) with elements

$$z_{0,k}, z_{1,k}, \ldots, z_{k,k}$$

is the solution of the system

$$A_k z_k = f_k ,$$

where

$$f_k = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ f_{k,k} \end{pmatrix} , \quad f_{k,k} = b_k - \sum_{\ell=1}^{k} a_{-\ell} y_{k-\ell,k-1} .$$

Thus, the vector z_k is the same as the last column of the matrix A_k^{-1} multiplied by $f_{k,k}$. Hence, for recurrent calculation of the vectors y_k it is sufficient to evaluate recurrently the last column of the matrix A_k^{-1}, or as done here for further economy an appropriately chosen multiple of this column. It is here that advantage is taken of the Toeplitz structure of A.

Let us denote by g_k and h_k the first and last columns, respectively, each scaled by the as yet unspecified factor q_k, of the matrix A_k^{-1}:

$$g_k = \begin{pmatrix} g_{0,k} \\ g_{1,k} \\ \vdots \\ g_{k-1,k} \\ g_{k,k} \end{pmatrix} , \quad h_k = \begin{pmatrix} h_{0,k} \\ h_{1,k} \\ \vdots \\ h_{k-1,k} \\ h_{k,k} \end{pmatrix} , \quad \text{and} \quad A_k g_k = \begin{pmatrix} q_k \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix} , \quad A_k h_k = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ q_k \end{pmatrix} .$$

It is clear that when $k = 0$ the unscaled vectors coincide and contain the single element $1/a_0$; we choose $q_0 = a_0$ so that $g_0 = h_0 = 1$. We will determine g_k, h_k, and q_k from g_{k-1}, h_{k-1}, and q_{k-1} using the following two sums:

$$g_k = \begin{pmatrix} g_{0,k-1} \\ g_{1,k-1} \\ \vdots \\ g_{k-1,k-1} \\ 0 \end{pmatrix} + v \begin{pmatrix} 0 \\ h_{0,k-1} \\ \vdots \\ h_{k-2,k-1} \\ h_{k-1,k-1} \end{pmatrix} ,$$

$$h_k = r \begin{pmatrix} g_{0,k-1} \\ g_{1,k-1} \\ \vdots \\ g_{k-1,k-1} \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ h_{0,k-1} \\ \vdots \\ h_{k-2,k-1} \\ h_{k-1,k-1} \end{pmatrix} ,$$

where v and r are unknown scalars which we are going to derive.

Since g_k and h_k are columns of the matrix A_k^{-1} scaled by q_k, then

$$A_k g_k = A_k \begin{pmatrix} g_{0,k-1} \\ g_{1,k-1} \\ \vdots \\ g_{k-1,k-1} \\ 0 \end{pmatrix} + vA_k \begin{pmatrix} 0 \\ h_{0,k-1} \\ \vdots \\ h_{k-2,k-1} \\ h_{k-1,k-1} \end{pmatrix} = \begin{pmatrix} q_k \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix} ,$$

$$A_k h_k = rA_k \begin{pmatrix} g_{0,k-1} \\ g_{1,k-1} \\ \vdots \\ g_{k-1,k-1} \\ 0 \end{pmatrix} + A_k \begin{pmatrix} 0 \\ h_{0,k-1} \\ \vdots \\ h_{k-2,k-1} \\ h_{k-1,k-1} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ q_k \end{pmatrix} .$$

These relationships reduce to the following equations for determining the unknown scalars:

$$\begin{cases} q_{k-1} + f_2 v = q_k \\ f_1 + q_{k-1} v = 0 \end{cases} \qquad \begin{cases} q_{k-1} r + f_2 = 0 \\ f_1 r + q_{k-1} = q_k \end{cases} , \tag{4}$$

where

$$f_1 = \sum_{\ell=1}^{k} a_{-\ell} g_{k-\ell,k-1} , \quad f_2 = \sum_{\ell=1}^{k} a_\ell h_{\ell-1,k-1} .$$

Solving equations (4) we find

$$v = -f_1/q_{k-1}, \quad r = -f_2/q_{k-1}, \quad q_k = q_{k-1} - f_1 f_2/q_{k-1} .$$

Note that this algorithm for solving linear systems with T-matrices requires that A_k be non-singular for all k.

2.5. Programming details — subroutine TSLS1

The calling sequence of subroutine TSLS is consistent with those of the other TOEPLITZ subroutines. However, it proves convenient in the implementation to consider the input matrix as two arrays and to partition the work space. Therefore, subroutine TSLS1 was produced to directly implement the algorithm, and subroutine TSLS merely acts as a user interface that calls TSLS1. TSLS1 may be called directly by the user, if desired.

The calling sequence of subroutine TSLS1 is

```
CALL TSLS1(A1,A2,B,X,C1,C2,M) .
```

On entry,

A1 is a singly subscripted array of M elements which contains the first row of the T-matrix. A1 is unaltered by TSLS1.

A2 is a singly subscripted array of M-1 elements which contains the first column of the T-matrix beginning with the second element. A2 is unaltered by TSLS1.

B is a singly subscripted array of M elements which contains the right hand side of the system. B is unaltered by TSLS1.

C1,C2 are singly subscripted arrays of M-1 elements used for work space.

M is the order of the T-matrix and the number of elements in B and X.

On return,

X is a singly subscripted array of M elements which contains the solution of the system. X may coincide with B.

Subroutine TSLS1 has double precision, complex, and double precision complex versions with names TSLD1, TSLC1, and TSLZ1, respectively, whose calling sequences are the same as that of TSLS1 with A1, A2, B, C1, C2, and X variables of the corresponding type.

Towards timing estimation, note that the algorithm for solving linear systems with T-matrices requires approximately $3M^2$ multiplications.

2.6. Additional information

The calling sequences of subroutines TSLS and TSLS1 for the solution of linear systems with T-matrices limit the right hand sides to single column vectors. There may be situations where the solutions of two or more such systems with the same coefficient matrix are desired. In these situations, modifications of the subroutines that would permit all solutions to be obtained in a single step could markedly improve efficiency. Fortunately, the algorithm organization for T-matrices enables such modifications to be made with little effort.

Three changes need to be made: 1) The parameter list must be extended to include the column order of X and B, and the leading dimension for these newly created two-dimensional arrays; 2) References to X and B must be rendered two-dimensional; and 3) DO loops must be introduced for cycling over the columns of X and B. Resulting forms of TSLS and TSLS1 are given below and can be compared with the official versions listed in Appendix B; to facilitate the comparison, the changes are indicated in lower case. The identical changes could be made to the double precision, complex, and double precision complex versions of these subroutines.

```
      SUBROUTINE TSLS(A,X,R,M,mcol,ldx)
      INTEGER M,mcol,ldx
      REAL A(1),X(ldx,mcol),R(1)
C
C     TSLS CALLS TSLS1 TO SOLVE THE REAL LINEAR SYSTEM
C     A * X = B
C     WITH THE T - MATRIX A .
C
C     ON ENTRY
C
C        A       REAL(2*M - 1)
C                THE FIRST ROW OF THE T - MATRIX FOLLOWED BY ITS
C                FIRST COLUMN BEGINNING WITH THE SECOND ELEMENT .
C                ON RETURN A IS UNALTERED .
C
C        X       REAL(M,mcol)
C                THE RIGHT HAND SIDE matrix B .
C
C        R       REAL(2*M - 2)
C                A WORK VECTOR .
C
C        M       INTEGER
C                THE ORDER OF THE MATRIX A .
C
c        mcol    integer
c                the number of columns of the matrices x and b .
c
c        ldx     integer
c                the leading dimension of the array x .
c
C     ON RETURN
C
C        X       THE SOLUTION matrix .
C
C     SUBROUTINES AND FUNCTIONS
C
C        TOEPLITZ PACKAGE ... TSLS1
C
C     CALL SUBROUTINE TSLS1
C
      CALL TSLS1(A,A(M+1),X,X,R,R(M),M,mcol,ldx)
C
      RETURN
      END
      SUBROUTINE TSLS1(A1,A2,B,X,C1,C2,M,mcol,ldx)
      INTEGER M,mcol,ldx
      REAL A1(M),A2(1),B(ldx,mcol),X(ldx,mcol),C1(1),C2(1)
C
C     TSLS1 SOLVES THE REAL LINEAR SYSTEM
C     A * X = B
C     WITH THE T - MATRIX A .
C
C     ON ENTRY
C
C        A1      REAL(M)
```

```
C                THE FIRST ROW OF THE T - MATRIX A .
C                ON RETURN A1 IS UNALTERED .
C
C        A2      REAL(M - 1)
C                THE FIRST COLUMN OF THE T - MATRIX A
C                BEGINNING WITH THE SECOND ELEMENT .
C                ON RETURN A2 IS UNALTERED .
C
C        B       REAL(M,mcol)
C                THE RIGHT HAND SIDE matrix .
C                ON RETURN B IS UNALTERED .
C
C        C1      REAL(M - 1)
C                A WORK VECTOR .
C
C        C2      REAL(M - 1)
C                A WORK VECTOR .
C
C        M       INTEGER
C                THE ORDER OF THE MATRIX A .
C
c        mcol    integer
c                the number of columns of the matrices x and b .
c
c        ldx     integer
c                the leading dimension of the arrays x and b .
c
C     ON RETURN
C
C        X       REAL(M,mcol)
C                THE SOLUTION matrix. X MAY COINCIDE WITH B .
C
C     INTERNAL VARIABLES
C
      INTEGER I1,I2,j,N,N1,N2
      REAL R1,R2,R3,R5,R6
C
C     SOLVE THE SYSTEM WITH THE PRINCIPAL MINOR OF ORDER 1 .
C
      R1 = A1(1)
      do 5 j = 1, mcol
         X(1,j) = B(1,j)/R1
    5 continue
      IF (M .EQ. 1) GO TO 80
C
C     RECURRENT PROCESS FOR SOLVING THE SYSTEM
C     WITH THE T - MATRIX FOR N = 2, M .
C
      DO 70 N = 2, M
C
C        COMPUTE MULTIPLES OF THE FIRST AND LAST COLUMNS OF
C        THE INVERSE OF THE PRINCIPAL MINOR OF ORDER N .
C
         N1 = N - 1
```

```
      N2 = N - 2
      R5 = A2(N1)
      R6 = A1(N)
      IF (N .EQ. 2) GO TO 20
         C1(N1) = R2
         DO 10 I1 = 1, N2
            I2 = N - I1
            R5 = R5 + A2(I1)*C1(I2)
            R6 = R6 + A1(I1+1)*C2(I1)
   10    CONTINUE
   20 CONTINUE
      R2 = -R5/R1
      R3 = -R6/R1
      R1 = R1 + R5*R3
      IF (N .EQ. 2) GO TO 40
         R6 = C2(1)
         C2(N1) = 0.0E0
         DO 30 I1 = 2, N1
            R5 = C2(I1)
            C2(I1) = C1(I1)*R3 + R6
            C1(I1) = C1(I1) + R6*R2
            R6 = R5
   30    CONTINUE
   40 CONTINUE
      C2(1) = R3
C
C     COMPUTE THE SOLUTION OF THE SYSTEM WITH THE
C     PRINCIPAL MINOR OF ORDER N .
C
      do 65 j = 1, mcol
         R5 = 0.0E0
         DO 50 I1 = 1, N1
            I2 = N - I1
            R5 = R5 + A2(I1)*X(I2,j)
   50    CONTINUE
         R6 = (B(N,j) - R5)/R1
         DO 60 I1 = 1, N1
            X(I1,j) = X(I1,j) + C2(I1)*R6
   60    CONTINUE
         X(N,j) = R6
   65 continue
   70 CONTINUE
   80 CONTINUE
      RETURN
      END
```

3. Solution of Linear Equations with C-Matrices

3.1. Purpose

The TOEPLITZ subroutine in this section is designed to solve linear algebraic equations with C-matrices; it is limited to complex matrices because the algorithm employs complex arithmetic. Users with real circulant matrices can either declare them complex or consider them simply T-matrices and employ the subroutines of section 2. Running times as real T-matrices are shorter, but the unitary transformations employed in the algorithm described below for C-matrices offer greater stability. A double precision version of the subroutine is also available.

3.2. Usage

Single precision C-matrices. CSLC solves a linear system with a complex circulant matrix. The calling sequence is

```
CALL CSLC(A,X,R,M) .
```

On entry,

A is a singly subscripted array of M elements which contains the first row of the C-matrix. A is unaltered by CSLC.

X is a singly subscripted array of M elements which contains the right hand side of the system.

R is a singly subscripted array of M elements used for work space.

M is the order of A and the number of elements in X.

On return,

X contains the solution of the system.

Double precision C-matrices. In those computing systems where it is available, the calling sequence of the double precision C-matrix subroutine CSLZ is the same as that of CSLC with A, X, and R DOUBLE COMPLEX variables.

3.3. Example

The following program segment illustrates the use of the single precision subroutine CSLC for C-matrices. An example of the use of CSLZ could be obtained by changing the subroutine name and type declaration. The system is of order 4 with coefficients as follows.

$$A = \begin{pmatrix} 1+i & 2+2i & 3+3i & 4+4i \\ 4+4i & 1+i & 2+2i & 3+3i \\ 3+3i & 4+4i & 1+i & 2+2i \\ 2+2i & 3+3i & 4+4i & 1+i \end{pmatrix} \qquad X = \begin{pmatrix} 10+10i \\ 10+10i \\ 10+10i \\ 10+10i \end{pmatrix}$$

```
      COMPLEX A(4),X(4),R(4)
      INTEGER M,I
      DATA A(1)/(1.0,1.0)/,A(2)/(2.0,2.0)/,A(3)/(3.0,3.0)/,
     *     A(4)/(4.0,4.0)/
      DATA X(1)/(10.0,10.0)/,X(2)/(10.0,10.0)/,X(3)/(10.0,10.0)/,
     *     X(4)/(10.0,10.0)/
      M = 4
      CALL CSLC(A,X,R,M)
      DO 10 I = 1, M
         WRITE(...,...) X(I)
   10 CONTINUE
      STOP
      END
```

The solution of the system is

$$X = ((1.0,0.0),(1.0,0.0),(1.0,0.0),(1.0,0.0)) .$$

3.4. Algorithm

The algorithm for solving a system of linear algebraic equations

$$Ax = b \tag{1}$$

with a C-matrix A of order M proceeds from a similarity transformation of A to a diagonal matrix

$$D = Q^{*}AQ ,$$

where Q is unitary. (The symbol * denotes conjugate transpose.) The elements of Q are inverse discrete Fourier transformations defined as

$$q_{ij} = E^{(i-1)\cdot(j-1)}/\sqrt{M} ,$$

where $E = \exp(2\pi\sqrt{-1}/M)$. The solution x of the system (1) is then determined as

$$x = QD^{-1}Q^{*}b \ . \tag{2}$$

The diagonal elements d_{ii} of D can be calculated as simply

$$d_{ii} = \sqrt{M} \sum_{j=1}^{M} q_{ij} a_{j-1} \ , \qquad i = 1,2,\ldots,M \ .$$

In other words, if d is a column vector composed of the diagonal elements $d_{11}, d_{22}, \ldots, d_{MM}$ of D, and a is a column vector composed of the elements $a_0, a_1, \ldots, a_{M-1}$ of the first row of A, then these vectors are related by

$$d = \sqrt{M}\ Qa \ .$$

3.5. Programming details

In the implementation of subroutine CSLC, instead of Q the matrix $\overline{Q} = \sqrt{M}\ Q$ is used, and formula (2) of subsection 3.4 becomes

$$x = \overline{Q}D^{-1}\overline{Q}^{*}b/M \ .$$

The vector d composed of the diagonal elements d_{ii} of D is then calculated more simply as

$$d = \overline{Q}\ a \ .$$

Towards timing estimation, note that the algorithm for solving linear systems with C-matrices requires approximately $3M^2$ multiplications.

3.6. Additional information

The calling sequence of subroutine CSLC for the solution of linear systems with C-matrices limits the right hand side to a single column vector. There may be situations where the solutions of two or more such systems with the same coefficient matrix are desired. In these situations, modifications of the subroutine that would permit all solutions to be obtained in a single step could markedly improve efficiency. Unlike TSLS and TSLS1 discussed in subsection 2.6, CSLC admits no simple modification for this purpose; however, subroutine SALWC could be used instead.

Subroutine SALWC is discussed in subsection 3.5 of Chapter 2 -- it is called as a service subroutine in the solution of second- and third-level matrix systems. SALWC is similar to CSLC; its different organization, however, enables it to be separately useful, although somewhat awkward, for the solution of C-matrix systems with multiple right hand sides. Its use requires three calls with some arithmetic in-between, the presentation of the transpose of the right hand side matrix, and additional work space; also, unlike CSLC, it overwrites the coefficient array.

The following program segment illustrates the use of SALWC for C-matrix systems of order M with MROW right hand sides (refer to subsection 3.5 of Chapter 2 for a description of the SALWC calling sequence).

```
      COMPLEX A(M),X(LDX,M),R1(M),R2(M)
      .
      .
      .
      RM = FLOAT(M)
      CALL SALWC(A,R1,R2,1,M,1,-1)
      CALL SALWC(X,R1,R2,MROW,M,LDX,1)
      DO 10 J = 1, M
         DO 5 I = 1, MROW
            X(I,J) = X(I,J)/A(J)/RM
    5    CONTINUE
   10 CONTINUE
      CALL SALWC(X,R1,R2,MROW,M,LDX,-1)
```

The dominant term in the multiplication count for the above segment is $M^2 \cdot (2 \cdot MROW+1)$, while for MROW calls of CSLC it is $3M^2 \cdot MROW$. Comparing these quantities leads to the expectation that when MROW is 1 the two algorithms should be about equally fast, and as MROW increases a savings of up to 1/3 should be possible with the above segment. For double precision, substitute SALWZ.

4. Solution of Linear Equations with G-Matrices

4.1. Purpose

Capability to solve linear algebraic equations with G-matrices is required for processing second- and third-level Toeplitz- and circulant-type matrices described in Chapters 2 and 3. The availability of the LINPACK package makes it unnecessary to duplicate effort to provide this capability; the TOEPLITZ package simply invokes that subset of LINPACK which treats general square matrices. Usage will be briefly described for the single precision real version; double precision, complex, and double precision complex versions are also available. Referral to the LINPACK Users' Guide [15] is recommended for fuller discussion than will be given here, including algorithm descriptions and programming details.

4.2. Usage

Single precision real G-matrices. SGEFA and SGESL together solve a linear system Ax = b with a real general matrix A; SGEFA computes the LU factorization of A and SGESL uses the factorization to solve the linear system.

The calling sequence for SGEFA is

```
CALL SGEFA(A,LDA,M,PVT,INFO) .
```

On entry,

A   is a doubly subscripted M by M array which contains the G-matrix.

LDA is the leading dimension of the array A.

M   is the order of A and the number of elements in PVT.

On return,

A   contains information from the LU factorization.

PVT is a singly subscripted array of M elements which contains information to be transmitted to SGESL about the pivoting strategy used in the factorization. Note: In the LINPACK package PVT is specified as an integer array. For use in the TOEPLITZ package, PVT has the variable type of A; this simplifies the partition of the work space.

INFO is an integer which if nonzero warns of singularity of A. Note: Nonsingularity of A and indeed all its principal minors is fundamental for use of the TOEPLITZ package; no interrogation of INFO is made anywhere.

The calling sequence for SGESL is

CALL SGESL(A,LDA,M,PVT,X,JOB) .

On entry,

A is a doubly subscripted M by M array which contains the information from the factorization stored by SGEFA.

LDA is the leading dimension of the array A.

M is the order of A and the number of elements in X and PVT.

PVT is a singly subscripted array of M elements which contains the pivot information stored by SGEFA.

X is a singly subscripted array of M elements which contains the right hand side of the system.

JOB is an integer which specifies the system to be solved. If JOB is zero, the system $Ax = b$ is solved. If JOB is nonzero, the system $A^T x = b$ is solved. Note: In its use with the TOEPLITZ package, JOB is always zero.

On return,

X contains the solution of the system.

Double precision real G-matrices. The calling sequences of the double precision real G-matrix subroutines DGEFA and DGESL are the same as those of SGEFA and SGESL with A, X, and PVT DOUBLE PRECISION variables.

Single precision complex G-matrices. The calling sequences of the single precision complex G-matrix subroutines CGEFA and CGESL are the same as those of SGEFA and SGESL with A, X, and PVT COMPLEX variables.

Double precision complex G-matrices. In those computing systems where they are available, the calling sequences of the double precision complex G-matrix subroutines ZGEFA and ZGESL are the same as those of SGEFA and SGESL with A, X, and PVT DOUBLE COMPLEX variables.

5. Orthogonal Factorization of Column-Circulant Matrices

5.1. Purpose

Given an M by L column-circulant matrix A, the TOEPLITZ subroutines in this section determine an M by L matrix Q with orthonormal columns and an upper triangular matrix S of order L such that AS = Q. The AS = Q factorization can be transformed to the more familiar A = QR factorization by inverting S, i.e., $R = S^{-1}$. Usage will be described here for the single precision real version. Double precision, complex, and double precision complex versions are also available.

5.2. Usage

Single precision real column-circulant matrices. CQRS performs the orthogonal factorization AS = Q of a real column-circulant matrix A. The calling sequence is

```
CALL CQRS(A,Q,S,M,L,LDQ,LDS) .
```

On entry,

A is a singly subscripted array of M elements which contains the first column of the column-circulant matrix. A is unaltered by CQRS.

M is the number of rows of the matrices A and Q. M must be at least equal to L.

L is the number of columns of the matrices A and Q and the order of the upper triangular matrix S.

LDQ is the leading dimension of the array Q.

LDS is the leading dimension of the array S.

On return,

Q is a doubly subscripted M by L array which contains the factor with orthonormal columns.

S is a doubly subscripted L by L array which contains the upper triangular factor. Elements below the main diagonal of S are not accessed.

Double precision real column-circulant matrices. The calling sequence of the double precision real column-circulant orthogonal factorization subroutine CQRD is the same as that of CQRS with A, Q, and S DOUBLE PRECISION variables.

Single precision complex column-circulant matrices. The calling sequence of the single precision complex column-circulant orthogonal factorization subroutine CQRC is the same as that of CQRS with A, Q, and S COMPLEX variables.

Double precision complex column-circulant matrices. In those computing systems where it is available, the calling sequence of the double precision complex column-circulant orthogonal factorization subroutine CQRZ is the same as that of CQRS with A, Q, and S DOUBLE COMPLEX variables.

5.3. Example

The following program segment illustrates the use of the single precision subroutine for orthogonal factorization of real column-circulant matrices; factors Q and S are returned satisfying AS = Q. Examples of the use of CQRD, CQRC, and CQRZ could be obtained by changing the subroutine name and type declaration. The matrix is 4 by 3 with coefficients as follows.

$$A = \begin{pmatrix} 1 & 4 & 3 \\ 2 & 1 & 4 \\ 3 & 2 & 1 \\ 4 & 3 & 2 \end{pmatrix}$$

```
REAL A(4),Q(4,3),S(3,3)
INTEGER M,L,LDQ,LDS,I,J
DATA A(1)/1.0/,A(2)/2.0/,A(3)/3.0/,A(4)/4.0/
M = 4
L = 3
LDQ = 4
LDS = 3
CALL CQRS(A,Q,S,M,L,LDQ,LDS)
```

```
      DO 10 I = 1, M
         WRITE(...,...) (Q(I,J),J=1,L)
   10 CONTINUE
      DO 20 I = 1, L
         WRITE(...,...) (S(I,J),J=I,L)
   20 CONTINUE
      STOP
      END
```

The factors Q and S are

$$Q = \begin{pmatrix} 1/\sqrt{30} & 16/\sqrt{270} & 10/\sqrt{7344} \\ 2/\sqrt{30} & -3/\sqrt{270} & 78/\sqrt{7344} \\ 3/\sqrt{30} & -2/\sqrt{270} & -26/\sqrt{7344} \\ 4/\sqrt{30} & -1/\sqrt{270} & -22/\sqrt{7344} \end{pmatrix} \qquad S = \begin{pmatrix} 1/\sqrt{30} & -4/\sqrt{270} & -7/\sqrt{7344} \\ 0 & 5/\sqrt{270} & -16/\sqrt{7344} \\ 0 & 0 & 27/\sqrt{7344} \end{pmatrix}$$

5.4. Algorithm

The algorithm description can be found in [12]. Note that usage of this algorithm for orthogonal factorization of column-circulant matrices requires that the matrix have full rank L.

5.5. Programming details

The algorithm for the orthogonal factorization of an M by L column-circulant matrix requires approximately $6ML+L^2$ multiplications.

CHAPTER 2: TOEPLITZ- AND CIRCULANT-TYPE MATRICES OF THE SECOND LEVEL

1. Structure and Representation

1.1. Overview

A matrix

$$A = \begin{pmatrix} A_{11} & A_{12} & A_{13} & \cdots & A_{1L} \\ A_{21} & A_{22} & A_{23} & \cdots & A_{2L} \\ A_{31} & A_{32} & A_{33} & \cdots & A_{3L} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ A_{L1} & A_{L2} & A_{L3} & \cdots & A_{LL} \end{pmatrix} \qquad (1)$$

with L elements in a row (or column) where the elements A_{ij} are blocks of order M is called a two-level matrix. L is called the first-level order and M becomes the second-level order of the matrix A. The order N of A is then the product of the orders of its levels: N = L*M.

We will call the two-level matrix (1) an XY-type if A considered as a block matrix is an X-type and each of its blocks A_{ij} is a Y-type. As X- and Y-types in the TOEPLITZ package we consider T-, C-, and G-matrices defined in section 1 of Chapter 1. Examples of two-level matrices can be found below and in subsections 2.3, 3.3, 4.3, and 5.3 of this chapter.

By permuting corresponding rows and columns, we can transform any XY-type to YX-type (see Tyrtyshnikov [25]). For example, the TC-matrix

$$\left(\begin{array}{ccc|ccc} a & b & c & d & e & f \\ c & a & b & f & d & e \\ b & c & a & e & f & d \\ \hline g & h & i & a & b & c \\ i & g & h & c & a & b \\ h & i & g & b & c & a \end{array}\right)$$

with L=2, M=3 can be permuted to the CT-matrix

$$\left(\begin{array}{cc|cc|cc} a & i & b & g & c & h \\ e & a & f & b & d & c \\ \hline c & h & a & i & b & g \\ d & c & e & a & f & b \\ \hline b & g & c & h & a & i \\ f & b & d & c & e & a \end{array}\right)$$

with L=3, M=2 by interchanging row and column pairs (1,6) and (3,4). This circumstance allows us to limit consideration to one of each XY- YX-type pair.

The scheme for compact representation of two-level matrices is the following. Let A be of XY-type with first-level order L and second-level order M. Furthermore, let $\tilde{L}$ be the number of elements required in the compact representation of X and $\tilde{M}$ be the number of elements required in the compact representation of Y. Recall that for T-, C- and G-matrices of order M as described in section 1 of Chapter 1 the values of $\tilde{M}$ are, respectively, 2*M-1, M, and M**2. In the TOEPLITZ package such a two-level matrix is represented by a doubly subscripted $\tilde{M}$ by $\tilde{L}$ array. The blocks in the array are indexed by the second subscript and ordered in accordance with the X-type compact representation. In turn, the elements in a block are indexed by the first subscript and ordered in accordance with the block's Y-type compact representation.

1.2. TG-matrices

A matrix

$$A = \begin{pmatrix} A_0 & A_1 & A_2 & \cdots & A_{L-1} \\ A_{-1} & A_0 & A_1 & \cdots & A_{L-2} \\ A_{-2} & A_{-1} & A_0 & \cdots & A_{L-3} \\ \cdot & \cdot & \cdot & \cdots & \cdot \\ A_{-L+1} & A_{-L+2} & A_{-L+3} & \cdots & A_0 \end{pmatrix}$$

is called a TG-matrix if A_i and A_{-i}, $i=0,1,2,\ldots,L-1$, are G-matrices of order M (see subsection 1.3 of Chapter 1).

In the TOEPLITZ package this TG-matrix is represented by a doubly subscripted M**2 by 2*L-1 array in which the blocks are ordered in the following way:

$$A_0, A_1, A_2, \ldots, A_{L-1}, A_{-1}, A_{-2}, \ldots, A_{-L+1} \ .$$

1.3. CT-matrices

A complex matrix

$$A = \begin{pmatrix} A_0 & A_1 & A_2 & \cdots & A_{L-1} \\ A_{L-1} & A_0 & A_1 & \cdots & A_{L-2} \\ A_{L-2} & A_{L-1} & A_0 & \cdots & A_{L-3} \\ \cdot & \cdot & \cdot & \cdots & \cdot \\ A_1 & A_2 & A_3 & \cdots & A_0 \end{pmatrix} . \tag{2}$$

is called a CT-matrix if A_i, $i=0,1,2,\ldots,L-1$, are T-matrices of order M (see subsection 1.1 of Chapter 1).

In the TOEPLITZ package this CT-matrix is represented by a doubly subscripted 2*M-1 by L array in which the blocks are ordered in the following way:

$$A_0, A_1, A_2, \ldots, A_{L-1} \ .$$

1.4. CC-matrices

A matrix of form (2) is called a CC-matrix if A_i, $i=0,1,2,\ldots,L-1$, are C-matrices of order M (see subsection 1.2 of Chapter 1).

In the TOEPLITZ package this CC-matrix is represented by a doubly subscripted M by L array in which the blocks are ordered in the following way:

$$A_0, A_1, A_2, \ldots, A_{L-1} \ .$$

1.5. CG-matrices

A matrix of form (2) is called a CG-matrix if A_i, $i=0,1,2,\ldots,L-1$, are G-matrices of order M (see subsection 1.3 of Chapter 1).

In the TOEPLITZ package this CG-matrix is represented by a doubly subscripted M**2 by L array in which the blocks are ordered in the following way:

$$A_0, A_1, A_2, \ldots, A_{L-1} .$$

1.6. Other types of two-level matrices

GT-, TC-, and GC-matrices, defined in analogous ways, can be permuted, respectively, to TG-, CT-, and CG-matrices (see example in subsection 1.1). Therefore, the TOEPLITZ package does not include subroutines for solving linear systems with two-level matrices of these types. At the present time no algorithm is known that capitalizes effectively on the structure of TT-matrices, so TT-matrices should be treated as TG-matrices.

2. Solution of Linear Equations with TG-matrices

2.1. Purpose

The TOEPLITZ subroutines in this section are designed to solve linear algebraic equations with TG-matrices, that is, block-Toeplitz matrices whose blocks are G-matrices. Usage will be described for the single precision real version. Double precision, complex, and double precision complex versions are also available. Indeed, the complex version is called in solving three-level CTG-matrix systems (see subsection 2.5 of Chapter 3).

2.2. Usage

Single precision real TG-matrices. TGSLS solves a linear system with a real block-Toeplitz matrix whose blocks are G-matrices. The calling sequence is

```
CALL TGSLS(A,X,R,M,L,LDA) .
```

On entry,

A is a doubly subscripted M**2 by 2*L-1 array which contains the TG-matrix in the form described in subsection 1.2. A is unaltered by TGSLS.

X is a singly subscripted array of M*L elements which contains the right hand side of the system.

R is a singly subscripted array of 2*M**2*L+3*M**2+M elements used for work space.

M is the order of each G-matrix block of A.

L is the number of blocks in each row or column of A.

LDA is the leading dimension of the array A.

On return,

X contains the solution of the system.

Double precision real TG-matrices. The calling sequence of the double precision real TG-matrix subroutine TGSLD is the same as that of TGSLS with A, X, and R DOUBLE PRECISION variables.

Single precision complex TG-matrices. The calling sequence of the single precision complex TG-matrix subroutine TGSLC is the same as that of TGSLS with A, X, and R COMPLEX variables.

Double precision complex TG-matrices. In those computing systems where it is available, the calling sequence of the double precision complex TG-matrix subroutine TGSLZ is the same as that of TGSLS with A, X, and R DOUBLE COMPLEX variables.

2.3. Example

The following program segment illustrates the use of the single precision subroutine TGSLS for real TG-matrices. Examples of the use of TGSLD, TGSLC, and TGSLZ could be obtained by changing the subroutine name and type declaration. The system is of order 4 with coefficients as follows.

$$A = \left(\begin{array}{cc|cc} 1 & 3 & 5 & 7 \\ 2 & 4 & 6 & 8 \\ \hline 9 & 11 & 1 & 3 \\ 10 & 12 & 2 & 4 \end{array}\right) \qquad X = \left(\begin{array}{c} 16 \\ 20 \\ 24 \\ 28 \end{array}\right)$$

```
      REAL A(4,3),X(4),R(30)
      INTEGER M,L,LDA,I,J
      DATA A(1,1)/1.0/,A(2,1)/2.0/.A(3,1)/3.0/,A(4,1)/4.0/,
     *     A(1,2)/5.0/,A(2,2)/6.0/,A(3,2)/7.0/,A(4,2)/8.0/,
     *     A(1,3)/9.0/,A(2,3)/10.0/,A(3,3)/11.0/,A(4,3)/12.0/
      DATA X(1)/16.0/,X(2)/20.0/,X(3)/24.0/,X(4)/28.0/
      M = 2
      L = 2
      LDA = 4
      CALL TGSLS(A,X,R,M,L,LDA)
      J = M*L
      DO 10 I = 1, J
         WRITE(...,...) X(I)
   10 CONTINUE
      STOP
      END
```

The solution of the system is

$$X = (1.0, 1.0, 1.0, 1.0) \ .$$

2.4. Algorithm

The algorithm for solving a linear system

$$Ax = b \tag{1}$$

with the TG-matrix

$$A = \begin{pmatrix} A_0 & A_1 & A_2 & \cdots & A_{L-1} \\ A_{-1} & A_0 & A_1 & \cdots & A_{L-2} \\ A_{-2} & A_{-1} & A_0 & \cdots & A_{L-3} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ A_{-L+1} & A_{-L+2} & A_{-L+3} & \cdots & A_0 \end{pmatrix},$$

where A_i and A_{-i}, $i=0,1,2,\ldots,L-1$, are G-matrices of order M, is the block analogue of the algorithm for solving linear systems with T-matrices (see subsection 2.4 of Chapter 1).

Let us introduce the following notation:

$$C_k = \begin{pmatrix} A_0 & A_1 & \cdots & A_k \\ A_{-1} & A_0 & \cdots & A_{k-1} \\ \cdots & \cdots & \cdots & \cdots \\ A_{-k} & A_{-k+1} & \cdots & A_0 \end{pmatrix}, \quad y_k = \begin{pmatrix} y_{0,k} \\ y_{1,k} \\ \vdots \\ y_{k,k} \end{pmatrix}, \quad d_k = \begin{pmatrix} b_0 \\ b_1 \\ \vdots \\ b_{k*M+M-1} \end{pmatrix},$$

where $y_{i,k}$, $i=0,1,\ldots,k$, are vectors of M elements. The algorithm consists of step-by-step recurrent solution of systems

$$C_k y_k = d_k \tag{2}$$

for $k=0,1,2,\ldots,L-1$. The final result of the recurrent process is the solution of the given system (1):

$$x = y_{L-1}.$$

At step 1, $y_0 = A_0^{-1} d_0$. At step k+1, the vector y_k is calculated from y_{k-1} as follows. Let us consider the vector y_k to be the sum of two vectors, one of which, augmented by a zero vector of M elements, was determined at the k-th step:

$$\begin{pmatrix} y_{0,k} \\ y_{1,k} \\ \vdots \\ y_{k-1,k} \\ y_{k,k} \end{pmatrix} = \begin{pmatrix} y_{0,k-1} \\ y_{1,k-1} \\ \vdots \\ y_{k-1,k-1} \\ 0 \end{pmatrix} + \begin{pmatrix} z_{0,k} \\ z_{1,k} \\ \vdots \\ z_{k-1,k} \\ z_{k,k} \end{pmatrix}. \tag{3}$$

Substituting this sum into equation (2) and taking into account that the vector y_{k-1} satisfies the equation

$$C_{k-1}y_{k-1} = d_{k-1} ,$$

we see that the unknown vector z_k from (3) consisting of component vectors $z_{0,k}, z_{1,k}, \ldots, z_{k,k}$ each of M elements is the solution of the system

$$C_k z_k = f_k ,$$

where

$$f_k = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ f_{k,k} \end{pmatrix} , \quad f_{k,k} = \begin{pmatrix} b_{k*M} \\ b_{k*M+1} \\ \vdots \\ b_{k*M+M-2} \\ b_{k*M+M-1} \end{pmatrix} - \sum_{\ell=1}^{k} a_{-\ell} y_{k-\ell,k-1} .$$

Thus, the vector z_k is a linear combination of the last M columns of the matrix C_k^{-1}, and the elements of the vector $f_{k,k}$ are the coefficients of that linear combination. Hence, for recurrent calculation of the vectors y_k it is sufficient to evaluate recurrently the last block column of the matrix C_k^{-1}, or as done here for further economy an appropriately chosen block multiple of this block column. It is here that advantage is taken of the block-Toeplitz structure of A.

Let us denote by G_k and H_k the first and last block columns, respectively scaled by M-order matrices P_k and Q_k, of the matrix C_k^{-1}:

$$G_k = \begin{pmatrix} G_{0,k} \\ G_{1,k} \\ \vdots \\ G_{k-1,k} \\ G_{k,k} \end{pmatrix} , \qquad H_k = \begin{pmatrix} H_{0,k} \\ H_{1,k} \\ \vdots \\ H_{k-1,k} \\ H_{k,k} \end{pmatrix} ,$$

and

$$C_k G_k = \begin{pmatrix} P_k \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix} , \quad C_k H_k = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ Q_k \end{pmatrix} .$$

It is clear that when $k = 0$ the unscaled block columns coincide and contain the single block A_0^{-1}; we choose $P_0 = Q_0 = A_0$ so that $G_0 = H_0 = I$. We will determine G_k, H_k, P_k, and Q_k from G_{k-1}, H_{k-1}, P_{k-1}, and Q_{k-1} using the following two sums:

$$G_k = \begin{pmatrix} G_{0,k-1} \\ G_{1,k-1} \\ \vdots \\ G_{k-1,k-1} \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ H_{0,k-1} \\ \vdots \\ H_{k-2,k-1} \\ H_{k-1,k-1} \end{pmatrix} V ,$$

$$H_k = \begin{pmatrix} G_{0,k-1} \\ G_{1,k-1} \\ \vdots \\ G_{k-1,k-1} \\ 0 \end{pmatrix} R + \begin{pmatrix} 0 \\ H_{0,k-1} \\ \vdots \\ H_{k-2,k-1} \\ H_{k-1,k-1} \end{pmatrix} ,$$

where V and R are unknown M by M matrices which we are going to derive.

Since G_k and H_k are block columns of the matrix C_k^{-1} scaled by P_k and Q_k, respectively, then

$$C_k G_k = C_k \begin{pmatrix} G_{0,k-1} \\ G_{1,k-1} \\ \vdots \\ G_{k-1,k-1} \\ 0 \end{pmatrix} + C_k \begin{pmatrix} 0 \\ H_{0,k-1} \\ \vdots \\ H_{k-2,k-1} \\ H_{k-1,k-1} \end{pmatrix} V = \begin{pmatrix} P_k \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix} ,$$

$$C_kH_k = C_k\begin{pmatrix} G_{0,k-1} \\ G_{1,k-1} \\ \vdots \\ G_{k-1,k-1} \\ 0 \end{pmatrix} R + C_k\begin{pmatrix} 0 \\ H_{0,k-1} \\ \vdots \\ H_{k-2,k-1} \\ H_{k-1,k-1} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ Q_k \end{pmatrix} .$$

These relationships reduce to the following equations for determining the unknown matrices:

$$\begin{cases} P_{k-1} + F_2V = P_k \\ F_1 + Q_{k-1}V = 0 \end{cases} \qquad \begin{cases} P_{k-1}R + F_2 = 0 \\ F_1R + Q_{k-1} = Q_k \ , \end{cases} \tag{4}$$

where

$$F_1 = \sum_{\ell=1}^{k} A_{-\ell}G_{k-\ell,k-1} \ , \qquad F_2 = \sum_{\ell=1}^{k} A_{\ell}H_{\ell-1,k-1} \ .$$

Solving systems (4) we find

$$V = -(Q_{k-1})^{-1}F_1 \ , \qquad R = -(P_{k-1})^{-1}F_2 \ ,$$
$$P_k = P_{k-1} - F_2(Q_{k-1})^{-1}F_1 \ ,$$
$$Q_k = Q_{k-1} - F_1(P_{k-1})^{-1}F_2 \ .$$

Note that this algorithm for solving linear systems with TG-matrices requires that C_k be non-singular for all k.

2.5. Programming details — subroutine TGSLS1

Subroutine TGSLS merely acts as an interface to subroutine TGSLS1, in the manner of TSLS and TSLS1 for T-matrices as explained in subsection 2.5 of Chapter 1.

The calling sequence of subroutine TGSLS1 is

CALL TGSLS1(A1,A2,B,X,C1,C2,R1,R2,R3,R5,R6,R,M,L,LDA) .

On entry,

A1 is a doubly subscripted M**2 by L array which contains the first row of blocks of the TG-matrix. A1 is unaltered by TGSLS1.

A2 is a doubly subscripted M**2 by L-1 array which contains the first column of blocks of the TG-matrix beginning with the second block. A2 is unaltered by TGSLS1.

B is a singly subscripted array of M*L elements which contains the right hand side of the system. B is unaltered by TGSLS1.

C1,C2 are triply subscripted arrays with dimension (M,M,L-1) used for work space.

R1,R2,R3,R5,R6 are doubly subscripted arrays with dimension (M,M) used for work space.

R is a singly subscripted array of M elements used for work space.

M is the order of each G-matrix block of the TG-matrix.

L is the number of blocks in each row or column of the TG-matrix.

LDA is the leading dimension of the arrays A1 and A2.

On return,

X is a singly subscripted array of M*L elements which contains the solution of the system. X may coincide with B.

For solving G-matrix systems in accordance with the algorithm described in subsection 2.4, TGSLS1 calls the LINPACK subroutines SGEFA and SGESL (see section 4 of Chapter 1).

Vector operations are facilitated by calls to the LINPACK BLA subroutine SAXPY. This subroutine is coded efficiently but there is a cost associated with communication to it; this cost can become relatively large when computation within SAXPY itself is small and the further computations of TGSLS1 are highly optimized by the compiler. Therefore, when the number of vector components (M for two-level TG-matrices) is small and the compiler is capable of a high level of optimization, it may be more efficient to perform the vector computations in-line instead of repeatedly calling SAXPY. (It is of interest to note that in TGSLC1, overhead associated with the use of the corresponding LINPACK BLA subroutine CAXPY is much less significant in the presence of the slower complex arithmetic.) To facilitate a possible change to in-line computation, directions are provided through code comments in subroutine TGSLS1 (and also TGSLD1, TGSLC1, and TGSLZ1).

For solving systems with double precision, complex, and double precision complex TG-matrices, versions corresponding to TGSLS1 are available with names TGSLD1, TGSLC1, and TGSLZ1, respectively. These in turn call the corresponding versions of the LINPACK subroutines.

The algorithm implemented in subroutine TGSLS1 requires approximately $2M^3L^2$ multiplications.

3. Solution of Linear Equations with CT-matrices

3.1. Purpose

The TOEPLITZ subroutine in this section is designed to solve linear algebraic equations with CT-matrices, that is, complex block-circulant matrices whose blocks are T-matrices. A double precision version of the subroutine is also available.

3.2. Usage

Single precision CT-matrices. CTSLC solves a linear system with a complex block-circulant matrix whose blocks are T-matrices. The calling sequence is

```
CALL CTSLC(A,X,R,M,L,LDA) .
```

On entry,

A   is a doubly subscripted 2*M-1 by L array which contains the CT-matrix in the form described in subsection 1.3. A is destroyed by CTSLC.

X   is a singly subscripted array of M*L elements which contains the right hand side of the system.

R   is a singly subscripted array of max(2*M-2,2*L) elements used for work space.

M   is the order of each T-matrix block of A.

L   is the number of blocks in each row or column of A.

LDA   is the leading dimension of the array A.

On return,

X   contains the solution of the system.

Double precision CT-matrices. In those computing systems where it is available, the calling sequence of the double precision CT-matrix subroutine CTSLZ is the same as that of CTSLC with A, X, and R DOUBLE COMPLEX variables.

3.3. Example

The following program segment illustrates the use of the single precision subroutine CTSLC for CT-matrices. An example of the use of CTSLZ could be obtained by changing the subroutine name and type declaration. The system is of order 4 with coefficients as follows.

$$A = \left[\begin{array}{cc|cc} 1+i & 2+2i & 2+2i & 3+3i \\ 3+3i & 1+i & 4+4i & 2+2i \\ \hline 2+2i & 3+3i & 1+i & 2+2i \\ 4+4i & 2+2i & 3+3i & 1+i \end{array}\right] \qquad X = \left[\begin{array}{c} 8+8i \\ 10+10i \\ 8+8i \\ 10+10i \end{array}\right]$$

```
      COMPLEX A(3,2),X(4),R(4)
      INTEGER M,L,LDA,I,J
      DATA A(1,1)/(1.0,1.0)/,A(2,1)/(2.0,2.0)/,A(3,1)/(3.0.3.0)/,
     *     A(1,2)/(2.0,2.0)/,A(2,2)/(3.0,3.0)/,A(3,2)/(4.0,4.0)/
      DATA X(1)/(8.0,8.0)/,X(2)/(10.0,10.0)/,X(3)/(8.0,8.0)/,
     *     X(4)/(10.0,10.0)/
      M = 2
      L = 2
      LDA = 3
      CALL CTSLC(A,X,R,M,L,LDA)
      J = M*L
      DO 10 I = 1, J
         WRITE(...,...) X(I)
   10 CONTINUE
      STOP
      END
```

The solution of the system is

$$X = ((1.0,0.0),(1.0,0.0),(1.0,0.0),(1.0,0.0)) .$$

3.4. Algorithm

The algorithm for solving a linear system

$$Ax = b \tag{1}$$

with the CT-matrix

$$A = \begin{pmatrix} A_0 & A_1 & A_2 & \cdots & A_{L-1} \\ A_{L-1} & A_0 & A_1 & \cdots & A_{L-2} \\ A_{L-2} & A_{L-1} & A_0 & \cdots & A_{L-3} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ A_1 & A_2 & A_3 & \cdots & A_0 \end{pmatrix} ,$$

where A_i, $i=0,1,2,\ldots,L-1$, are T-matrices of order M, proceeds from a similarity transformation of A to a block-diagonal matrix

$$D = Q^*AQ$$

in which each diagonal block is a T-matrix. (The symbol * denotes conjugate transpose.) Q is a two-level matrix with first-level order L and second-level order M whose blocks are scalar matrices; the matrix of the scalars themselves is unitary.

Block Q_{ij} of Q is defined as

$$Q_{ij} = E^{(i-1)\cdot(j-1)} * I/\sqrt{L} ,$$

where $E = \exp(2\pi\sqrt{-1}/L)$ and I is the identity matrix of order M. However, as for C-matrices (see section 3 of Chapter 1), it is more efficient to use instead the matrix

$$\overline{Q} = \sqrt{L}\, Q .$$

Thus the solution x of the system (1) can be found by the following steps:

a) Transform the matrix A to the block-diagonal matrix

$$D = \overline{Q}^{*} A \overline{Q}/L .$$

b) Transform the right hand side

$$y = \overline{Q}^{*} b .$$

c) Solve the system

$$Dz = y .$$

d) Transform the vector z back to

$$x = \overline{Q} z/L .$$

Note that since A is a CT-matrix, its transformation to D becomes simply

$$D_{ii} = \sum_{j=1}^{L} \overline{Q}_{ij} A_{j-1} ,$$

where D_{ii} is the i-th diagonal block of D and A_{j-1} is the block with index j-1 at the top of A. Furthermore, since D is block-diagonal each block of which is a T-matrix, the system (1) reduces to L systems with T-matrices.

3.5. <u>Programming details — subroutine SALWC</u>

The implementation of subroutine CTSLC corresponds to the algorithm described in subsection 3.4. All needed operations with matrices $\overline{Q}$ and $\overline{Q}^{*}$ are implemented by the service subroutine SALWC. The structure of these matrices and the compact form of input representation are such that, from the point of view of programming, these operations (or more properly Q and Q^*) can be considered respectively as inverse and direct discrete Fourier transformations upon a set of row vectors in a certain rectangular matrix.

The calling sequence of subroutine SALWC is

CALL SALWC(A,R1,R2,M,L,LDA,JOB) .

On entry,

A is a doubly subscripted M by L array which contains the matrix upon whose rows the Fourier transformation will be performed.

R1,R2 are singly subscripted arrays of L elements used for work space.

M is the number of rows of A.

L is the number of columns of A.

LDA is the leading dimension of the array A.

JOB indicates what is to be computed. If JOB is 1, the direct Fourier transformation will be performed and if JOB is -1, the inverse Fourier transformation will be performed.

On return,

A contains the transformed rows of the matrix.

For solving the L systems with T-matrices, first-level subroutines TSLC and TSLC1 are called. For solving systems with double precision CT-matrices (using CTSLZ), the double precision subroutine SALWZ is called, as well as TSLZ and TSLZ1.

The overall algorithm implemented in subroutine CTSLC requires approximately $4ML^2+3M^2L$ multiplications -- $4ML^2$ in SALWC and $3M^2L$ in TSLC1.

4. Solution of Linear Equations with CC-matrices

4.1. Purpose

The TOEPLITZ subroutine in this section is designed to solve linear algebraic equations with CC-matrices, that is, complex block-circulant matrices whose blocks themselves are circulant matrices. A double precision version of the subroutine is also available.

4.2. Usage

Single precision CC-matrices. CCSLC solves a linear system with a complex block-circulant matrix whose blocks are C-matrices. The calling sequence is

```
CALL CCSLC(A,X,R,M,L,LDA) .
```

On entry,

A is a doubly subscripted M by L array which contains the CC-matrix of the system in the form described in subsection 1.4. A is destroyed by CCSLC.

X is a singly subscripted array of M*L elements which contains the right hand side of the system.

R is a singly subscripted array of max(M,2*L) elements used for work space.

M is the order of each C-matrix block of A.

L is the number of blocks in each row or column of A.

LDA is the leading dimension of the array A.

On return,

X contains the solution of the system.

Double precision CC-matrices. In those computing systems where it is available, the calling sequence of the double precision CC-matrix subroutine CCSLZ is the same as that of CCSLC with A, X, and R DOUBLE COMPLEX variables.

4.3. Example

The following program segment illustrates the use of the single precision subroutine CCSLC for CC-matrices. An example of the use of CCSLZ could be obtained by changing the subroutine name and type declaration. The system is of order 4 with coefficients as follows.

$$A = \left|\begin{array}{cc|cc} 1+i & 2+2i & 2+2i & 4+4i \\ 2+2i & 1+i & 4+4i & 2+2i \\ \hline 2+2i & 4+4i & 1+i & 2+2i \\ 4+4i & 2+2i & 2+2i & 1+i \end{array}\right| \qquad X = \left|\begin{array}{c} 9+9i \\ 9+9i \\ 9+9i \\ 9+9i \end{array}\right|$$

```
      COMPLEX A(2,2),X(4),R(4)
      INTEGER M,L,LDA,I,J
      DATA A(1,1)/(1.0,1.0)/,A(2,1)/(2.0,2.0)/,
     *     A(1,2)/(2.0,2.0)/,A(2,2)/(4.0,4.0)/
      DATA X(1)/(9.0,9.0)/,X(2)/(9.0,9.0)/,
     *     X(3)/(9.0,9.0)/,X(4)/(9.0,9.0)/
      M = 2
      L = 2
      LDA = 2
      CALL CCSLC(A,X,R,M,L,LDA)
      J = M*L
      DO 10 I = 1, J
         WRITE(...,...) X(I)
   10 CONTINUE
      STOP
      END
```

The solution of the system is

X = ((1.0,0.0),(1.0,0.0),(1.0,0.0),(1.0,0.0)) .

4.4 Algorithm

The algorithm used in subroutine CCSLC is the same as that described in subsection 3.4 for CT-matrices except that the solution of C-matrix rather than T-matrix systems is involved.

4.5. Programming details

Programming details of subroutine CCSLC are as for CTSLC (see subsection 3.5) except that subroutine CSLC is called instead of subroutines TSLC and TSLC1; the number of multiplications is approximately $3ML^2+3M^2L$ -- $3ML^2$ in SALWC and $3M^2L$ in CSLC.

5. Solution of Linear Equations with CG-matrices

5.1. Purpose

The TOEPLITZ subroutine in this section is designed to solve linear algebraic equations with CG-matrices, that is, complex block-circulant matrices whose blocks are general matrices. A double precision version of the subroutine is also available.

5.2. Usage

Single precision CG-matrices. CGSLC solves a linear system with a complex block-circulant matrix whose blocks are G-matrices. The calling sequence is

```
CALL CGSLC(A,X,R,M,L,LDA) .
```

On entry,

- A is a doubly subscripted M**2 by L array which contains the CG matrix of the system in the form described in subsection 1.5. A is destroyed by CGSLC.

- X is a singly subscripted array of M*L elements which contains the right hand side of the system.

- R is a singly subscripted array of max(M,2*L) elements used for work space.

- M is the order of each G-matrix block of A.

- L is the number of blocks in each row or column of A.

- LDA is the leading dimension of the array A.

On return,

X contains the solution of the system.

Double precision CG-matrices. In those computing systems where it is available, the calling sequence of the double precision CG-matrix subroutine CGSLZ is the same as that of CGSLC with A, X, and R DOUBLE COMPLEX variables.

5.3. Example

The following program segment illustrates the use of the single precision subroutine CGSLC for CG-matrices. An example of the use of CGSLZ could be obtained by changing the subroutine name and type declaration. The system is of order 4 with coefficients as follows.

$$A = \left|\begin{array}{cc|cc} 1+i & 3+3i & 5+5i & 7+7i \\ 2+2i & 4+4i & 6+6i & 8+8i \\ \hline 5+5i & 7+7i & 1+i & 3+3i \\ 6+6i & 8+8i & 2+2i & 4+4i \end{array}\right| \qquad X = \left|\begin{array}{c} 16 + 16i \\ 20 + 20i \\ 16 + 16i \\ 20 + 20i \end{array}\right|$$

```
      COMPLEX A(4,2),X(4),R(4)
      INTEGER M,L,LDA,I,J
      DATA A(1,1)/(1.0,1.0)/,A(2,1)/(2.0,2.0)/,A(3,1)/(3.0,3.0)/,
     *     A(4,1)/(4.0,4.0)/,A(1,2)/(5.0,5.0)/,A(2,2)/(6.0,6.0)/,
     *     A(3,2)/(7.0,7.0)/,A(4,2)/(8.0,8.0)/
      DATA X(1)/(16.0,16.0)/,X(2)/(20.0,20.0)/,X(3)/(16.0,16.0)/,
     *     X(4)/(20.0,20.0)/
      M = 2
      L = 2
      LDA = 4
      CALL CGSLC(A,X,R,M,L,LDA)
      J = M*L
      DO 10 I = 1, J
         WRITE(...,...) X(I)
   10 CONTINUE
      STOP
      END
```

The solution of the system is

X = ((1.0,0.0),(1.0,0.0),(1.0,0.0),(1.0,0.0)) .

5.4. Algorithm

The algorithm used in subroutine CGSLC is the same as that described in subsection 3.4 for CT-matrices except that the solution of G-matrix rather than T-matrix systems is involved.

5.5. Programming details

Programming details of subroutine CGSLC are as for CTSLC (see subsection 3.5) except that LINPACK subroutines CGEFA and CGESL (see section 4 of Chapter 1) are called instead of subroutines TSLC and TSLC1; the number of multiplications is $M^2L^2+M^3L/3$ plus terms of lesser degree -- M^2L^2 in SALWC (first call), $M^3L/3$ in CGEFA, and lesser amounts in CGESL and further calls of SALWC.

CHAPTER 3: TOEPLITZ- AND CIRCULANT-TYPE MATRICES OF THE THIRD LEVEL

1. Structure and Representation

1.1. Overview

A matrix

$$A = \begin{vmatrix} A_{11} & A_{12} & A_{13} & \cdots & A_{1K} \\ A_{21} & A_{22} & A_{23} & \cdots & A_{2K} \\ A_{31} & A_{32} & A_{33} & \cdots & A_{3K} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ A_{K1} & A_{K2} & A_{K3} & \cdots & A_{KK} \end{vmatrix} \tag{1}$$

with K elements in a row (or column) where the elements A_{ij} are two-level matrices (see Chapter 2) with first-level order L and second-level order M is called a three-level matrix. K is called the first-level order, L becomes the second-level order, and M becomes the third-level order of the matrix A. The order N of A is then the product of the orders of its levels: $N = K*L*M$.

We will call the three-level matrix (1) an XYZ-type if A considered as a block matrix is an X-type and each of its blocks A_{ij} is a YZ-type (see section 1 of Chapter 2). As X-, Y-, and Z-types in the TOEPLITZ package we consider T-, C-, and G-matrices defined in section 1 of Chapter 1. Examples of three-level matrices can be found in subsections 2.3, 3.3, 4.3, and 5.3 of this chapter.

By permuting corresponding rows and columns, we can transform an XYZ-type to any of types XZY, YXZ, YZX, ZXY, or ZYX (see Tyrtyshnikov [25]). This circumstance allows us to limit consideration to a few among the possible three-level types.

The scheme for compact representation of three-level matrices is the following. Let A be of XYZ-type with level orders K, L, and M, respectively. Furthermore, let $\widetilde{K}$ be the number of elements required in the compact representation of X, and $\widetilde{M}*\widetilde{L}$ be the number of elements required in the compact representation of a YZ-type with level orders L and M. Recall that for TG-, CT-, CC-, and CG-matrices described in section 1 of Chapter 2 the values of $\tilde{M}*\tilde{L}$ are, respectively, M**2*(2*L-1), (2*M-1)*L, M*L, and M**2*L. In the TOEPLITZ package such a three-level matrix is represented by a doubly subscripted $\tilde{M}*\tilde{L}$ by $\tilde{K}$ array. The blocks in the array are indexed by the second subscript and ordered in accordance with the X-type compact representation. In turn, the elements in a block are indexed by the first subscript and ordered in accordance with the block's YZ-type compact representation packed linearly by columns.

1.2. CTG-matrices

A complex matrix

$$A = \begin{pmatrix} A_0 & A_1 & A_2 & \cdots & A_{K-1} \\ A_{K-1} & A_0 & A_1 & \cdots & A_{K-2} \\ A_{K-2} & A_{K-1} & A_0 & \cdots & A_{K-3} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ A_1 & A_2 & A_3 & \cdots & A_0 \end{pmatrix} \tag{2}$$

is called a CTG-matrix if A_i, i=0,1,2,...,K-1, are TG-matrices of first-level order L and second-level order M (see subsection 1.2 of Chapter 2).

In the TOEPLITZ package this CTG-matrix is represented by a doubly subscripted M**2*(2*L-1) by K array in which the first-level blocks are ordered in the following way:

$$A_0, A_1, A_2, \ldots, A_{K-1} .$$

Each block A_i is a TG-matrix packed linearly by columns.

1.3. CCT-matrices

A matrix of form (2) is called a CCT-matrix if A_i, i=0,1,2,...,K-1, are CT-matrices of first-level order L and second-level order M (see subsection 1.3 of Chapter 2).

In the TOEPLITZ package this CCT-matrix is represented by a doubly subscripted (2*M-1)*L by K array. The storage arrangement for the CCT-matrix is as for the CTG-matrix except that each block A_i is a CT-matrix.

1.4. CCC-matrices

A matrix of form (2) is called a CCC-matrix if A_i, $i=0,1,2,\ldots,K-1$, are CC-matrices of first-level order L and second-level order M (see subsection 1.4 of Chapter 2).

In the TOEPLITZ package this CCC-matrix is represented by a doubly subscripted M*L by K array. The storage arrangement for the CCC-matrix is as for the CTG-matrix except that each block A_j is a CC-matrix.

1.5. CCG-matrices

A matrix of form (2) is called a CCG-matrix if A_i, $i=0,1,2,\ldots,K-1$, are CG-matrices of first-level order L and second-level order M (see subsection 1.5 of Chapter 2).

In the TOEPLITZ package this CCG-matrix is represented by a doubly subscripted M**2*L by K array. The storage arrangement for the CCG-matrix is as for the CTG-matrix except that each block A_i is a CG-matrix.

1.6. Other types of three-level matrices

CGT-, TCG-, TGC-, GCT-, GTC-, CTC-, TCC-, CGC-, and GCC-matrices, defined in analogous ways, can be transformed to the types discussed in subsections 1.2-1.5 by permuting corresponding levels. Therefore, the TOEPLITZ package does not include subroutines for solving linear systems with three-level matrices of these types. At the present time no algorithm is known that capitalizes effectively on the structure of linear systems with three-level matrices more than one of whose levels is of T- or G-type.

2. Solution of Linear Equations with CTG-matrices

2.1. Purpose

The TOEPLITZ subroutine in this section is designed to solve linear algebraic equations with CTG-matrices, that is, complex block-circulant matrices whose blocks are TG-matrices. A double precision version is also available.

2.2. Usage

Single precision CTG-matrices. CTGSLC solves a linear system with a CTG-matrix. The calling sequence is

CALL CTGSLC(A,X,R,M,L,K,LDA) .

On entry,

A is a doubly subscripted M**2*(2*L-1) by K array which contains the CTG-matrix in the form described in subsection 1.2. A is destroyed by CTGSLC.

X is a singly subscripted array of M*L*K elements which contains the right hand side of the system.

R is a singly subscripted array of max(2*M**2*L+3*M**2+M,2*K) elements used for work space.

M is the order of each inner G-matrix block of A.

L is the number of inner blocks in each row or column of the TG-matrices which comprise the outer blocks of A.

K is the number of outer blocks in each row or column of A.

LDA is the leading dimension of the array A.

On return,

X contains the solution of the system.

Double precision CTG-matrices. In those computing systems where it is available, the calling sequence of the double precision CTG-matrix subroutine CTGSLZ is the same as that of CTGSLC with A, X, and R DOUBLE COMPLEX variables.

2.3. Example

The following program segment illustrates the use of the single precision subroutine CTGSLC for CTG-matrices. An example of the use of CTGSLZ could be obtained by changing the subroutine name and type declaration. The system is of order 8 with coefficients as follows.

$$A = \left(\begin{array}{cc|cc|cc|cc}
1+i & 3+3i & 5+5i & 7+7i & 13+13i & 15+15i & 17+17i & 19+19i \\
2+2i & 4+4i & 6+6i & 8+8i & 14+14i & 16+16i & 18+18i & 20+20i \\
\hline
9+9i & 11+11i & 1+i & 3+3i & 21+21i & 23+23i & 13+13i & 15+15i \\
10+10i & 12+12i & 2+2i & 4+4i & 22+22i & 24+24i & 14+14i & 16+16i \\
\hline
13+13i & 15+15i & 17+17i & 19+19i & 1+i & 3+3i & 5+5i & 7+7i \\
14+14i & 16+16i & 18+18i & 20+20i & 2+2i & 4+4i & 6+6i & 8+8i \\
\hline
21+21i & 23+23i & 13+13i & 15+15i & 9+9i & 11+11i & 1+i & 3+3i \\
22+22i & 24+24i & 14+14i & 16+16i & 10+10i & 12+12i & 2+2i & 4+4i
\end{array}\right) \qquad X = \begin{pmatrix} 80+80i \\ 88+88i \\ 96+96i \\ 104+104i \\ 80+80i \\ 88+88i \\ 96+96i \\ 104+104i \end{pmatrix}$$

```
      COMPLEX A(12,2),X(8),R(30)
      INTEGER M,L,K,LDA,I,J
      DATA A(1,1)/(1.0,1.0)/,A(2,1)/(2.0,2.0)/,A(3,1)/(3.0,3.0)/,
     *     A(4,1)/(4.0,4.0)/,A(5,1)/(5.0,5.0)/,A(6,1)/(6.0,6.0)/,
     *     A(7,1)/(7.0,7.0)/,A(8,1)/(8.0,8.0)/,A(9,1)/(9.0,9.0)/,
     *     A(10,1)/(10.0,10.0)/,A(11,1)/(11.0,11.0)/,A(12,1)/(12.0,12.0)/,
     *     A(1,2)/(13.0,13.0)/,A(2,2)/(14.0,14.0)/,A(3,2)/(15.0,15.0)/,
     *     A(4,2)/(16.0,16.0)/,A(5,2)/(17.0,17.0)/,A(6,2)/(18.0,18.0)/,
     *     A(7,2)/(19.0,19.0)/,A(8,2)/(20.0,20.0)/,A(9,2)/(21.0,21.0)/,
     *     A(10,2)/(22.0,22.0)/,A(11,2)/(23.0,23.0)/,A(12,2)/(24.0,24.0)/
      DATA X(1)/(80.0,80.0)/,X(2)/(88.0,88.0)/,X(3)/(96.0,96.0)/,
     *     X(4)/(104.0,104.0)/,X(5)/(80.0,80.0)/,X(6)/(88.0,88.0)/,
     *     X(7)/(96.0,96.0)/,X(8)/(104.0,104.0)/
      M = 2
      L = 2
      K = 2
      LDA = 12
      CALL CTGSLC(A,X,R,M,L,K,LDA)
      J = M*L*K
      DO 10 I = 1, J
         WRITE(...,...) X(I)
   10 CONTINUE
      STOP
      END
```

The solution of the system is

$$X = ((1.0,0.0),(1.0,0.0),(1.0,0.0),(1.0,0.0)\\ (1.0,0.0),(1.0,0.0),(1.0,0.0),(1.0,0.0)) \ .$$

2.4. Algorithm

The algorithm for solving a linear system

$$Ax = b \tag{1}$$

with the CTG-matrix

$$A = \begin{pmatrix} A_0 & A_1 & A_2 & \cdots & A_{K-1} \\ A_{K-1} & A_0 & A_1 & \cdots & A_{K-2} \\ A_{K-2} & A_{K-1} & A_0 & \cdots & A_{K-3} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ A_1 & A_2 & A_3 & \cdots & A_0 \end{pmatrix},$$

where A_i, $i=0,1,2,\ldots,K-1$, are TG-matrices of first-level order L and second-level order M, is analogous to that described in subsection 3.4 of Chapter 2 for CT-matrices. It proceeds from a similarity transformation of A to a block-diagonal matrix

$$D = Q^*AQ$$

in which each diagonal block is a TG-matrix. (The symbol * denotes conjugate transpose.) Q is a two-level matrix with first-level order K and second-level order L*M whose blocks are scalar matrices; the matrix of the scalars themselves is unitary.

Block Q_{ij} of Q is defined as

$$Q_{ij} = E^{(i-1)\cdot(j-1)}*I/\sqrt{K} \ ,$$

where $E = \exp(2\pi\sqrt{-1}/K)$ and I is the identity matrix of order L*M. However, as for C-matrices (see section 3 of Chapter 1), it is more efficient to use instead the matrix

$$\overline{Q} = \sqrt{K}\ Q \ .$$

Thus the solution x of the system (1) can be found by the following steps:

a) Transform the matrix A to the block-diagonal matrix

$$D = \overline{Q}^{*} A \overline{Q}/K .$$

b) Transform the right hand side

$$y = \overline{Q}^{*} b .$$

c) Solve the system

$$Dz = y .$$

d) Transform the vector z back to

$$x = \overline{Q}z/K .$$

Note that since A is a CTG-matrix, its transformation to D becomes simply

$$D_{ii} = \sum_{j=1}^{K} \overline{Q}_{ij} A_{j-1} ,$$

where D_{ii} is the i-th diagonal block of D and A_{j-1} is the outer block with index j-1 at the top of A. Furthermore, since D is block-diagonal each block of which is a TG-matrix, the system (1) reduces to K systems with TG-matrices.

2.5. Programming details

The implementation of subroutine CTGSLC corresponds to the algorithm described in subsection 2.4. All needed operations with matrices $\overline{Q}$ and $\overline{Q}^{*}$ are implemented by the service subroutine SALWC described in subsection 3.5 of Chapter 2. For solving the K systems with TG-matrices, second-level subroutines TGSLC and TGSLC1 are called.

For solving systems with double precision CTG-matrices (using CTGSLZ), corresponding versions of subroutines TGSLC, TGSLC1, and SALWC are called, namely, TGSLZ, TGSLZ1, and SALWZ.

The number of multiplications in executing subroutine CTGSLC is $2M^3L^2K + 2M^2LK^2$ plus terms of lesser degree.

3. Solution of Linear Equations with CCT-matrices

3.1. Purpose

The TOEPLITZ subroutine in this section is designed to solve linear algebraic equations with CCT-matrices, that is, complex block-circulant matrices whose blocks are CT-matrices. A double precision version is also available.

3.2. Usage

Single precision CCT-matrices. CCTSLC solves a linear system with a CCT-matrix. The calling sequence is

```
CALL CCTSLC(A,X,R,M,L,K,LDA) .
```

On entry,

A   is a doubly subscripted (2*M-1)*L by K array which contains the CCT-matrix in the form described in subsection 1.3. A is destroyed by CCTSLC.

X   is a singly subscripted array of M*L*K elements which contains the right hand side of the system.

R   is a singly subscripted array of max(2*M-2,2*L,2*K) elements used for work space.

M   is the order of each inner T-matrix block of A.

L   is the number of inner blocks in each row or column of the CT-matrices which comprise the outer blocks of A.

K   is the number of outer blocks in each row or column of A.

LDA is the leading dimension of the array A.

On return,

X   contains the solution of the system.

Double precision CCT-matrices. In those computing systems where it is available, the calling sequence of the double precision CCT-matrix subroutine CCTSLZ is the same as that of CCTSLC with A, X, and R DOUBLE COMPLEX variables.

3.3. Example

The following program segment illustrates the use of the single precision subroutine CCTSLC for CCT-matrices. An example of the use of CCTSLZ could be obtained by changing the subroutine name and type declaration. The system is of order 8 with coefficients as follows.

$$A = \left(\begin{array}{cc|cc|cc|cc}
1+i & 2+2i & 4+4i & 5+5i & 7+7i & 8+8i & 10+10i & 11+11i \\
3+3i & 1+i & 6+6i & 4+4i & 9+9i & 7+7i & 12+12i & 10+10i \\
\hline
4+4i & 5+5i & 1+i & 2+2i & 10+10i & 11+11i & 7+7i & 8+8i \\
6+6i & 4+4i & 3+3i & 1+i & 12+12i & 10+10i & 9+9i & 7+7i \\
\hline
7+7i & 8+8i & 10+10i & 11+11i & 1+i & 2+2i & 4+4i & 5+5i \\
9+9i & 7+7i & 12+12i & 10+10i & 3+3i & 1+i & 6+6i & 4+4i \\
\hline
10+10i & 11+11i & 7+7i & 8+8i & 4+4i & 5+5i & 1+i & 2+2i \\
12+12i & 10+10i & 9+9i & 7+7i & 6+6i & 4+4i & 3+3i & 1+i
\end{array}\right) \qquad X = \begin{pmatrix} 48+48i \\ 52+52i \\ 48+48i \\ 52+52i \\ 48+48i \\ 52+52i \\ 48+48i \\ 52+52i \end{pmatrix}$$

```
      COMPLEX A(6,2),X(8),R(4)
      INTEGER M,L,K,LDA,I,J
      DATA A(1,1)/(1.0,1.0)/,A(2,1)/(2.0,2.0)/,A(3,1)/(3.0,3.0)/,
     *     A(4,1)/(4.0,4.0)/,A(5,1)/(5.0,5.0)/,A(6,1)/(6.0,6.0)/,
     *     A(1,2)/(7.0,7.0)/,A(2,2)/(8.0,8.0)/,A(3,2)/(9.0,9.0)/,
     *     A(4,2)/(10.0,10.0)/,A(5,2)/(11.0,11.0)/,A(6,2)/(12.0,12.0)/
      DATA X(1)/(48.0,48.0)/,X(2)/(52.0,52.0)/,
     *     X(3)/(48.0,48.0)/,X(4)/(52.0,52.0)/,
     *     X(5)/(48.0,48.0)/,X(6)/(52.0,52.0)/,
     *     X(7)/(48.0,48.0)/,X(8)/(52.0,52.0)/
      M = 2
      L = 2
      K = 2
      LDA = 6
      CALL CCTSLC(A,X,R,M,L,K,LDA)
      J = M*L*K
      DO 10 I = 1, J
         WRITE(...,...) X(I)
   10 CONTINUE
      STOP
      END
```

The solution of the system is

$$X = ((1.0,0.0),(1.0,0.0),(1.0,0.0),(1.0,0.0),\\ (1.0,0.0),(1.0,0.0),(1.0,0.0),(1.0,0.0)) .$$

3.4. Algorithm

The algorithm used in subroutine CCTSLC is the same as that described in subsection 2.4 for CTG-matrices except that the solution of CT-matrix rather than TG-matrix systems is involved.

3.5. Programming details

Programming details of subroutine CCTSLC are as for CTGSLC (see subsection 2.5) except that subroutine CTSLC is called instead of subroutines TGSLC and TGSLC1; the number of multiplications is approximately $4MLK^2+4ML^2K+3M^2LK$.

4. Solution of Linear Equations with CCC-matrices

4.1. Purpose

The TOEPLITZ subroutine in this section is designed to solve linear algebraic equations with CCC-matrices, that is, complex block-circulant matrices whose blocks are CC-matrices. A double precision version is also available.

4.2. Usage

Single precision CCC-matrices. CCSLC solves a linear system with a CCC-matrix. The calling sequence is

```
CALL CCCSLC(A,X,R,M,L,K,LDA) .
```

On entry,

A is a doubly subscripted M*L by K array which contains the CCC-matrix in the form described in subsection 1.4. A is destroyed by CCCSLC.

X is a singly subscripted array of M*L*K elements which contains the right hand side of the system.

R is a singly subscripted array of max(M,2*L,2*K) elements used for work space.

M is the order of each inner C-matrix block of A.

L is the number of inner blocks in each row or column of the CC-matrices which comprise the outer blocks of A.

K is the number of outer blocks in each row or column of A.

LDA is the leading dimension of the array A.

On return,

X contains the solution of the system.

Double precision CCC-matrices. In those computing systems where it is available, the calling sequence of the double precision CCC-matrix subroutine CCCSLZ is the same as that of CCCSLC with A, X, and R DOUBLE COMPLEX variables.

4.3. Example

The following program segment illustrates the use of the single precision subroutine CCCSLC for CCC-matrices. An example of the use of CCCSLZ could be obtained by changing the subroutine name and type declaration. The system is of order 8 with coefficients as follows.

$$A = \left(\begin{array}{cc|cc|cc|cc}
1+i & 2+2i & 3+3i & 4+4i & 5+5i & 6+6i & 7+7i & 8+8i \\
2+2i & 1+i & 4+4i & 3+3i & 6+6i & 5+5i & 8+8i & 7+7i \\
\hline
3+3i & 4+4i & 1+i & 2+2i & 7+7i & 8+8i & 5+5i & 6+6i \\
4+4i & 3+3i & 2+2i & 1+i & 8+8i & 7+7i & 6+6i & 5+5i \\
\hline
5+5i & 6+6i & 7+7i & 8+8i & 1+i & 2+2i & 3+3i & 4+4i \\
6+6i & 5+5i & 8+8i & 7+7i & 2+2i & 1+i & 4+4i & 3+3i \\
\hline
7+7i & 8+8i & 5+5i & 6+6i & 3+3i & 4+4i & 1+i & 2+2i \\
8+8i & 7+7i & 6+6i & 5+5i & 4+4i & 3+3i & 2+2i & 1+i
\end{array}\right)
\qquad
X = \begin{pmatrix} 36+36i \\ 36+36i \\ 36+36i \\ 36+36i \\ 36+36i \\ 36+36i \\ 36+36i \\ 36+36i \end{pmatrix}$$

```
      COMPLEX A(4,2),X(8),R(4)
      INTEGER M,L,K,LDA,I,J
      DATA A(1,1)/(1.0,1.0)/,A(2,1)/(2.0,2.0)/,A(3,1)/(3.0,3.0)/,
     *     A(4,1)/(4.0,4.0)/,A(1,2)/(5.0,5.0)/,A(2,2)/(6.0,6.0)/,
     *     A(3,2)/(7.0,7.0)/,A(4,2)/(8.0,8.0)/
      DATA X(1)/(36.0,36.0)/,X(2)/(36.0,36.0),X(3)/(36.0,36.0)/,
     *     X(4)/(36.0,36.0)/,X(5)/(36.0,36.0),X(6)/(36.0,36.0)/,
     *     X(7)/(36.0,36.0)/,X(8)/(36.0,36.0)/
      M = 2
      L = 2
      K = 2
      LDA = 4
      CALL CCCSLC(A,X,R,M,L,K,LDA)
      J = M*L*K
      DO 10 I = 1, J
         WRITE(...,...) X(I)
   10 CONTINUE
      STOP
      END
```

The solution of the system is

```
X = ((1.0,0.0),(1.0,0.0),(1.0,0.0),(1.0,0.0),
     (1.0,0.0),(1.0,0.0),(1.0,0.0),(1.0,0.0)) .
```

4.4. Algorithm

The algorithm used in subroutine CCCSLC is the same as that described in subsection 2.4 for CTG-matrices except that the solution of CC-matrix rather than TG-matrix systems is involved.

4.5. Programming details

Programming details of subroutine CCCSLC are as for CTGSLC (see subsection 2.5) except that subroutine CCSLC is called instead of subroutines TGSLC and TGSLC1; the number of multiplications is approximately $3MLK^2+3ML^2K+3M^2LK$.

5. Solution of Linear Equations with CCG-matrices

5.1. Purpose

The TOEPLITZ subroutine in this section is designed to solve linear algebraic equations with CCG-matrices, that is, complex block-circulant matrices whose blocks are CG-matrices. A double precision version is also available.

5.2. Usage

Single precision CCG-matrices. CCGSLC solves a linear system with a CCG-matrix. The calling sequence is

```
CALL CCGSLC(A,X,R,M,L,K,LDA) .
```

On entry,

A   is a doubly subscripted M**2*L by K array which contains the CCG-matrix in the form described in subsection 1.5. A is destroyed by CCGSLC.

X   is a singly subscripted array of M*L*K elements which contains the right hand side of the system.

R   is a singly subscripted array of max(M,2*L,2*K) elements used for work space.

M   is the order of each inner G-matrix block of A.

L   is the number of inner blocks in each row or column of the CG-matrices which comprise the outer blocks of A.

K   is the number of outer blocks in each row or column of A.

LDA   is the leading dimension of the array A.

On return,

X   contains the solution of the system.

Double precision CCG-matrices. In those computing systems where it is available, the calling sequence of the double precision CCG-matrix subroutine CCGSLZ is the same as that of CCGSLC with A, X, and R DOUBLE COMPLEX variables.

5.3. Example

The following program segment illustrates the use of the single precision subroutine CCGSLC for CCG-matrices. An example of the use of CCGSLZ could be obtained by changing the subroutine name and type declaration. The system is of order 8 with coefficients as follows.

$$A = \left(\begin{array}{cc|cc|cc|cc}
1+i & 3+3i & 5+5i & 7+7i & 9+9i & 11+11i & 13+13i & 15+15i \\
2+2i & 4+4i & 6+6i & 8+8i & 10+10i & 12+12i & 14+14i & 16+16i \\
\hline
5+5i & 7+7i & 1+i & 3+3i & 13+13i & 15+15i & 9+9i & 11+11i \\
6+6i & 8+8i & 2+2i & 4+4i & 14+14i & 16+16i & 10+10i & 12+12i \\
\hline
9+9i & 11+11i & 13+13i & 15+15i & 1+i & 3+3i & 5+5i & 7+7i \\
10+10i & 12+12i & 14+14i & 16+16i & 2+2i & 4+4i & 6+6i & 8+8i \\
\hline
13+13i & 15+15i & 9+9i & 11+11i & 5+5i & 7+7i & 1+i & 3+3i \\
14+14i & 16+16i & 10+10i & 12+12i & 6+6i & 8+8i & 2+2i & 4+4i
\end{array}\right) \qquad X = \begin{pmatrix} 64+64i \\ 72+72i \\ 64+64i \\ 72+72i \\ 64+64i \\ 72+72i \\ 64+64i \\ 72+72i \end{pmatrix}$$

```
      COMPLEX A(8,2),X(8),R(4)
      INTEGER M,L,K,LDA,I,J
      DATA A(1,1)/(1.0,1.0)/,A(2,1)/(2.0,2.0)/,A(3,1)/(3.0,3.0)/,
     *     A(4,1)/(4.0,4.0)/,A(5,1)/(5.0,5.0)/,A(6,1)/(6.0,6.0)/,
     *     A(7,1)/(7.0,7.0)/,A(8,1)/(8.0,8.0)/,A(1,2)/(9.0,9.0)/,
     *     A(2,2)/(10.0,10.0)/,A(3,2)/(11.0,11.0)/,A(4,2)/(12.0,12.0)/,
     *     A(5,2)/(13.0,13.0)/,A(6,2)/(14.0,14.0)/,A(7,2)/(15.0,15.0)/,
     *     A(8,2)/(16.0,16.0)/
      DATA X(1)/(64.0,64.0)/,X(2)/(72.0,72.0)/,
     *     X(3)/(64.0,64.0)/,X(4)/(72.0,72.0)/,
     *     X(5)/(64.0,64.0)/,X(6)/(72.0,72.0)/,
     *     X(7)/(64.0,64.0)/,X(8)/(72.0,72.0)/
      M = 2
      L = 2
      K = 2
      LDA = 8
      CALL CCGSLC(A,X,R,M,L,K,LDA)
      J = M*L*K
      DO 10 I = 1, J
         WRITE(...,...) X(I)
   10 CONTINUE
      STOP
      END
```

The solution of the system is

X = ((1.0,0.0),(1.0,0.0),(1.0,0.0),(1.0,0.0),
(1.0,0.0),(1.0,0.0),(1.0,0.0),(1.0,0.0)) .

5.4 Algorithm

The algorithm used in subroutine CCGSLC is the same as that described in subsection 2.4 for CTG-matrices except that the solution of CG-matrix rather than TG-matrix systems is involved.

5.5. Programming details

Programming details of subroutine CCGSLC are as for CTGSLC (see subsection 2.5) except that subroutine CGSLC is called instead of subroutines TGSLC and TGSLC1; the number of multiplications is $M^2LK^2+M^2L^2K+M^3LK/3$ plus terms of lesser degree.

REFERENCES

[1] Numerical Analysis in Fortran, v. 17, published in Russian by Moscow University Press in 1976 and in English at the University of California, San Diego, in 1977, edited by James R. Bunch with the title Cooperative Development of Mathematical Software.

[2] Numerical Analysis in Fortran, Numerical Methods and Software Tools, published in Russian by Moscow University Press in 1979 and in English at the University of California, San Diego, in 1980, edited by James R. Bunch with the title Cooperative Development of Mathematical Software, Vol. 2.

[3] J. M. Boyle, K. W. Dritz, O. B. Arushanian, and Y. V. Kuchevskiy, Program Generation and Transformation - Tools for Mathematical Software Development, Information Processing 77, North Holland Publishing Co., 1977.

[4] H. Akaike, Block Toeplitz matrix inversion, SIAM J. Appl. Math., 24 (1973), pp. 234-241.

[5] B. D. O. Anderson and J. B. Moore, Optimal Filtering, Prentice-Hall, Englewood Cliffs, 1979.

[6] E. H. Bareiss, Numerical solution of linear equations with Toeplitz and vector Toeplitz matrices, Numer. Math., 13 (1969), pp. 404-424.

[7] G. E. P. Box and G. M. Jenkins, Time Series Analysis; Forecasting and Control, Holden-Day, San Francisco, 1970.

[8] A. Bultheel, Error analysis of incoming and outgoing schemes for the trigonometric moment problem, in Lecture Notes in Mathematics 888, van Rossum and de Bruin eds., Springer, Berlin, 1981, pp. 100-109.

[9] L. Collatz, The Numerical Treatment of Differential Equations, 3rd edition, Springer, Berlin, 1960.

[10] G. Cybenko, Error analyses of Levinson's, Durbin's, and Trench's algorithms, Proc. IEEE Int. Conf. on Acoust., Speech and Signal Proc., Washington, D.C., 1979.

[11] G. Cybenko, The numerical stability of the Levinson-Durbin algorithm for Toeplitz systems of equations, SIAM J. Sci. Stat. Comput., 1 (1980), pp. 303-319.

[12] G. Cybenko, A general orthogonalization technique with applications to time series analysis and signal processing, Math. Comp., 40 (1983), pp. 323-336.

[13] P. J. Davis, Circulant Matrices, John Wiley & Sons, New York, 1979.

[14] R. De Meersman, A method for least squares solution of systems with a cyclic rectangular coefficient matrix, J. of Comp. and Appl. Math., 1 (1975), pp. 51-54.

[15] J. J. Dongarra, J. R. Bunch, C. B. Moler, and G. W. Stewart, LINPACK Users' Guide, SIAM, Philadelphia, 1979.

[16] D. C. Farden, The solution of a special set of Hermitian Toeplitz linear equations, ACM Trans. Math. Software, 3 (1977), pp. 159-163.

[17] U. Grenander and G. Szego, Toeplitz Forms and Their Applications, University of California Press, Berkeley, 1958.

[18] F. Itakura and S. Saito, Digital filtering techniques for speech analysis and synthesis, Conference Record, 7th Int. Cong. on Acoust., Budapest, 1971, v. 3, paper 25 C 1, pp. 261-264.

[19] N. Levinson, The Wiener RMS error criterion in filter design and prediction, J. Math. Phys., 25 (1947), pp. 261-278.

[20] L. B. Rall, Computational Solution of Nonlinear Operator Equations, John Wiley & Sons, New York, 1969.

[21] J. Rissanen, Algorithms for triangular decomposition of block Hankel and Toeplitz matrices..., Math. Comp., 27 (1973), pp. 147-154.

[22] E. A. Robinson, Multichannel Time Series Analysis with Digital Computer Programs, Holden-Day, San Francisco, 1967.

[23] O. Toeplitz, Zur Theorie der quadrischen und bilinearen Formen von unendlichvielen Veranderlichen, Math. Ann., 70 (1911), pp. 351-376.

[24] W. F. Trench, An algorithm for the inversion of finite Toeplitz matrices, J. SIAM, 12 (1964), pp. 515-522.

[25] E. E. Tyrtyshnikov, On solving systems with Toeplitz-type matrices, Numerical Analysis in Fortran, Numerical Methods and Software Tools (in Russian), Moscow University Press, Moscow, 1979 (see reference 2 above).

[26] V. V. Voevodin, Foundations of Numerical Linear Algebra (in Russian), Nauka-Press, Moscow, 1977.

[27] S. N. Voevodina, The solution of systems with block-Toeplitz matrices, Numerical Methods and Programming (in Russian), Moscow University Press, Moscow, 1975.

[28] G. Walker, On periodicity in series of related terms, Proc. Royal Soc. London Ser. A, 131A (1931), pp. 518-532.

[29] N. Wiener, Extrapolation, Interpolation, and Smoothing of Stationary Time Series..., MIT Press, Cambridge, 1949.

[30] A. S. Willsky, Digital Signal Processing and Control and Estimation Theory..., MIT Press, Cambridge, 1979.

[31] G. U. Yule, On a method of investigating periodicities in disturbed series..., Philos. Trans. Roy. Soc. London Ser. A, 226A (1927), pp. 267-298.

[32] S. Zohar, Toeplitz matrix inversion: The algorithm of W. F. Trench, J. ACM, 16 (1969), pp. 592-601.

[33] S. Zohar, The solution of a Toeplitz set of linear equations, J. ACM, 21 (1974), pp. 272-276.

APPENDIX A. TABLES OF EXECUTION TIMES

We provide here three tables of sample execution times for the TOEPLITZ package subroutines. The first two tables report times for the single precision and double precision versions, respectively, on the VAX 11/780; the third table reports times for the single precision version on the IBM 3033. The VAX compilations were made with the Fortran 77 compiler running under UNIX; the IBM compilations were made with the Fortran H Extended (Enhanced) compiler running under MVS. Using these tables and the approximate multiplication counts given in the discussions of the algorithms in the previous chapters, it should be possible to extrapolate execution times for problems of different dimensions.

SUMMARY OF EXECUTION TIMES FOR THE
SINGLE PRECISION TOEPLITZ SUBROUTINES ON THE VAX 11/780

SUBROUTINE	3rd LEVEL	2nd LEVEL	1st LEVEL	TIME (sec.)
TSLS (TSLS1)			100	0.35
TSLC (TSLC1)			100	1.6
CSLC			100	1.2
CQRS		100(rows)	20(columns)	0.25
CQRC		100(rows)	20(columns)	0.83
TGSLS (TGSLS1)		10	10	5.9
TGSLS (IN-LINE SAXPY)		10	10	6.7
TGSLC (TGSLC1)		10	10	14.5
TGSLC (IN-LINE CAXPY)		10	10	16.6
CTSLC		20	20	2.7
CCSLC		20	20	2.2
CGSLC		20	20	13.5
CTGSLC	6	6	6	9.2
CCTSLC	8	8	8	2.5
CCCSLC	8	8	8	2.0
CCGSLC	8	8	8	5.7

SUMMARY OF EXECUTION TIMES FOR THE
DOUBLE PRECISION TOEPLITZ SUBROUTINES ON THE VAX 11/780

SUBROUTINE	3^{rd} LEVEL	2^{nd} LEVEL	1^{st} LEVEL	TIME (sec.)
TSLD (TSLD1)			100	0.52
TSLZ (TSLZ1)			100	2.5
CSLZ			100	1.9
CQRD		100(rows)	20(columns)	0.38
CQRZ		100(rows)	20(columns)	1.6
TGSLD (TGSLD1)		10	10	8.5
TGSLZ (TGSLZ1)		10	10	27.3
CTSLZ		20	20	4.0
CCSLZ		20	20	3.3
CGSLZ		20	20	22.2
CTGSLZ	6	6	6	14.8
CCTSLZ	8	8	8	3.7
CCCSLZ	8	8	8	3.2
CCGSLZ	8	8	8	9.4

SUMMARY OF EXECUTION TIMES FOR THE
SINGLE PRECISION TOEPLITZ SUBROUTINES ON THE IBM 3033

SUBROUTINE	2^{nd} LEVEL	1^{st} LEVEL	TIME (sec.)
TSLS (TSLS1)		100	.028
		200	.11
		300	.25
		400	.44
		500	.69
TSLC (TSLC1)		100	.19
CSLC		100	.18
		200	.70
		300	1.6
		400	2.8
		500	4.3
TGSLS (TGSLS1)	10	10	.47
TGSLS (IN-LINE SAXPY)	10	10	.27
TGSLC (TGSLC1)	10	10	1.5
TGSLC (IN-LINE CAXPY)	10	10	1.4
CTSLC	20	20	.35
CCSLC	20	20	.31
CGSLC	20	20	1.5

APPENDIX B. PROGRAM LISTINGS

There follows the single precision version of the TOEPLITZ package program listings; both single precision and double precision versions of the subprograms are available with the TOEPLITZ package. The listings appear in the following order:

TSLS, TSLS1, TSLC, TSLC1, CSLC, CQRS, CQRC, TGSLS, TGSLS1, TGSLC, TGSLC1, CTSLC, CCSLC, CGSLC, SALWC, CTGSLC, CCTSLC, CCCSLC, CCGSLC.

```
      SUBROUTINE TSLS(A,X,R,M)
      INTEGER M
      REAL A(1),X(M),R(1)
C
C     TSLS CALLS TSLS1 TO SOLVE THE REAL LINEAR SYSTEM
C     A * X = B
C     WITH THE T - MATRIX A .
C
C     ON ENTRY
C
C        A       REAL(2*M - 1)
C                THE FIRST ROW OF THE T - MATRIX FOLLOWED BY ITS
C                FIRST COLUMN BEGINNING WITH THE SECOND ELEMENT .
C                ON RETURN A IS UNALTERED .
C
C        X       REAL(M)
C                THE RIGHT HAND SIDE VECTOR B .
C
C        R       REAL(2*M - 2)
C                A WORK VECTOR .
C
C        M       INTEGER
C                THE ORDER OF THE MATRIX A .
C
C     ON RETURN
C
C        X       THE SOLUTION VECTOR .
C
C     TOEPLITZ PACKAGE. THIS VERSION DATED 07/23/82 .
C
C     SUBROUTINES AND FUNCTIONS
C
C        TOEPLITZ PACKAGE ... TSLS1
C
C     CALL SUBROUTINE TSLS1
C
      CALL TSLS1(A,A(M+1),X,X,R,R(M),M)
C
      RETURN
      END
```

```
      SUBROUTINE TSLS1(A1,A2,B,X,C1,C2,M)
      INTEGER M
      REAL A1(M),A2(1),B(M),X(M),C1(1),C2(1)
C
C     TSLS1 SOLVES THE REAL LINEAR SYSTEM
C     A * X = B
C     WITH THE T - MATRIX A .
C
C     ON ENTRY
C
C        A1      REAL(M)
C                THE FIRST ROW OF THE T - MATRIX A .
C                ON RETURN A1 IS UNALTERED .
C
C        A2      REAL(M - 1)
C                THE FIRST COLUMN OF THE T - MATRIX A
C                BEGINNING WITH THE SECOND ELEMENT .
C                ON RETURN A2 IS UNALTERED .
C
C        B       REAL(M)
C                THE RIGHT HAND SIDE VECTOR .
C                ON RETURN B IS UNALTERED .
C
C        C1      REAL(M - 1)
C                A WORK VECTOR .
C
C        C2      REAL(M - 1)
C                A WORK VECTOR .
C
C        M       INTEGER
C                THE ORDER OF THE MATRIX A .
C
C     ON RETURN
C
C        X       REAL(M)
C                THE SOLUTION VECTOR. X MAY COINCIDE WITH B .
C
C     TOEPLITZ PACKAGE. THIS VERSION DATED 07/23/82 .
C
C     INTERNAL VARIABLES
C
      INTEGER I1,I2,N,N1,N2
      REAL R1,R2,R3,R5,R6
C
C     SOLVE THE SYSTEM WITH THE PRINCIPAL MINOR OF ORDER 1 .
C
      R1 = A1(1)
      X(1) = B(1)/R1
      IF (M .EQ. 1) GO TO 80
C
C     RECURRENT PROCESS FOR SOLVING THE SYSTEM
C     WITH THE T - MATRIX FOR N = 2, M .
C
      DO 70 N = 2, M
C
```

```
C        COMPUTE MULTIPLES OF THE FIRST AND LAST COLUMNS OF
C        THE INVERSE OF THE PRINCIPAL MINOR OF ORDER N .
C
         N1 = N - 1
         N2 = N - 2
         R5 = A2(N1)
         R6 = A1(N)
         IF (N .EQ. 2) GO TO 20
            C1(N1) = R2
            DO 10 I1 = 1, N2
               I2 = N - I1
               R5 = R5 + A2(I1)*C1(I2)
               R6 = R6 + A1(I1+1)*C2(I1)
   10       CONTINUE
   20    CONTINUE
         R2 = -R5/R1
         R3 = -R6/R1
         R1 = R1 + R5*R3
         IF (N .EQ. 2) GO TO 40
            R6 = C2(1)
            C2(N1) = 0.0E0
            DO 30 I1 = 2, N1
               R5 = C2(I1)
               C2(I1) = C1(I1)*R3 + R6
               C1(I1) = C1(I1) + R6*R2
               R6 = R5
   30       CONTINUE
   40    CONTINUE
         C2(1) = R3
C
C        COMPUTE THE SOLUTION OF THE SYSTEM WITH THE
C        PRINCIPAL MINOR OF ORDER N .
C
         R5 = 0.0E0
         DO 50 I1 = 1, N1
            I2 = N - I1
            R5 = R5 + A2(I1)*X(I2)
   50    CONTINUE
         R6 = (B(N) - R5)/R1
         DO 60 I1 = 1, N1
            X(I1) = X(I1) + C2(I1)*R6
   60    CONTINUE
         X(N) = R6
   70 CONTINUE
   80 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE TSLC(A,X,R,M)
      INTEGER M
      COMPLEX A(1),X(M),R(1)
C
C     TSLC CALLS TSLC1 TO SOLVE THE COMPLEX LINEAR SYSTEM
C     A * X = B
C     WITH THE T - MATRIX A .
C
C     ON ENTRY
C
C        A       COMPLEX(2*M - 1)
C                THE FIRST ROW OF THE T - MATRIX FOLLOWED BY ITS
C                FIRST COLUMN BEGINNING WITH THE SECOND ELEMENT .
C                ON RETURN A IS UNALTERED .
C
C        X       COMPLEX(M)
C                THE RIGHT HAND SIDE VECTOR B .
C
C        R       COMPLEX(2*M - 2)
C                A WORK VECTOR .
C
C        M       INTEGER
C                THE ORDER OF THE MATRIX A .
C
C     ON RETURN
C
C        X       THE SOLUTION VECTOR .
C
C     TOEPLITZ PACKAGE. THIS VERSION DATED 07/23/82 .
C
C     SUBROUTINES AND FUNCTIONS
C
C        TOEPLITZ PACKAGE ... TSLC1
C
C     CALL SUBROUTINE TSLC1
C
      CALL TSLC1(A,A(M+1),X,X,R,R(M),M)
C
      RETURN
      END
```

```
      SUBROUTINE TSLC1(A1,A2,B,X,C1,C2,M)
      INTEGER M
      COMPLEX A1(M),A2(1),B(M),X(M),C1(1),C2(1)
C
C     TSLC1 SOLVES THE COMPLEX LINEAR SYSTEM
C     A * X = B
C     WITH THE T - MATRIX A .
C
C     ON ENTRY
C
C        A1       COMPLEX(M)
C                 THE FIRST ROW OF THE T - MATRIX A .
C                 ON RETURN A1 IS UNALTERED .
C
C        A2       COMPLEX(M - 1)
C                 THE FIRST COLUMN OF THE T - MATRIX A
C                 BEGINNING WITH THE SECOND ELEMENT .
C                 ON RETURN A2 IS UNALTERED .
C
C        B        COMPLEX(M)
C                 THE RIGHT HAND SIDE VECTOR .
C                 ON RETURN B IS UNALTERED .
C
C        C1       COMPLEX(M - 1)
C                 A WORK VECTOR .
C
C        C2       COMPLEX(M - 1)
C                 A WORK VECTOR .
C
C        M        INTEGER
C                 THE ORDER OF THE MATRIX A .
C
C     ON RETURN
C
C        X        COMPLEX(M)
C                 THE SOLUTION VECTOR. X MAY COINCIDE WITH B .
C
C     TOEPLITZ PACKAGE. THIS VERSION DATED 07/23/82 .
C
C     INTERNAL VARIABLES
C
      INTEGER I1,I2,N,N1,N2
      COMPLEX R1,R2,R3,R5,R6
C
C     SOLVE THE SYSTEM WITH THE PRINCIPAL MINOR OF ORDER 1 .
C
      R1 = A1(1)
      X(1) = B(1)/R1
      IF (M .EQ. 1) GO TO 80
C
C     RECURRENT PROCESS FOR SOLVING THE SYSTEM
C     WITH THE T - MATRIX FOR N = 2, M .
C
      DO 70 N = 2, M
C
```

```
C        COMPUTE MULTIPLES OF THE FIRST AND LAST COLUMNS OF
C        THE INVERSE OF THE PRINCIPAL MINOR OF ORDER N .
C
         N1 = N - 1
         N2 = N - 2
         R5 = A2(N1)
         R6 = A1(N)
         IF (N .EQ. 2) GO TO 20
            C1(N1) = R2
            DO 10 I1 = 1, N2
               I2 = N - I1
               R5 = R5 + A2(I1)*C1(I2)
               R6 = R6 + A1(I1+1)*C2(I1)
   10       CONTINUE
   20    CONTINUE
         R2 = -R5/R1
         R3 = -R6/R1
         R1 = R1 + R5*R3
         IF (N .EQ. 2) GO TO 40
            R6 = C2(1)
            C2(N1) = (0.0E0,0.0E0)
            DO 30 I1 = 2, N1
               R5 = C2(I1)
               C2(I1) = C1(I1)*R3 + R6
               C1(I1) = C1(I1) + R6*R2
               R6 = R5
   30       CONTINUE
   40    CONTINUE
         C2(1) = R3
C
C        COMPUTE THE SOLUTION OF THE SYSTEM WITH THE
C        PRINCIPAL MINOR OF ORDER N .
C
         R5 = (0.0E0,0.0E0)
         DO 50 I1 = 1, N1
            I2 = N - I1
            R5 = R5 + A2(I1)*X(I2)
   50    CONTINUE
         R6 = (B(N) - R5)/R1
         DO 60 I1 = 1, N1
            X(I1) = X(I1) + C2(I1)*R6
   60    CONTINUE
         X(N) = R6
   70 CONTINUE
   80 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE CSLC(A,X,R,M)
      INTEGER M
      COMPLEX A(M),X(M),R(M)
C
C     CSLC SOLVES THE COMPLEX LINEAR SYSTEM
C     A * X = B
C     WITH THE C - MATRIX A .
C
C     ON ENTRY
C
C        A       COMPLEX(M)
C                THE FIRST ROW OF THE C - MATRIX .
C                ON RETURN A IS UNALTERED .
C
C        X       COMPLEX(M)
C                THE RIGHT HAND SIDE VECTOR B .
C
C        R       COMPLEX(M)
C                A WORK VECTOR .
C
C        M       INTEGER
C                THE ORDER OF THE MATRIX A .
C
C     ON RETURN
C
C        X       THE SOLUTION VECTOR .
C
C     TOEPLITZ PACKAGE. THIS VERSION DATED 07/23/82 .
C
C     SUBROUTINES AND FUNCTIONS
C
C        FORTRAN ... CMPLX,COS,FLOAT,SIN
C
C     INTERNAL VARIABLES
C
      INTEGER I1,I2
      REAL P,RI,RM,V1,V2
      COMPLEX E,E1,F,F1,T,T1
C
      T1 = X(1)
      X(1) = T1/A(1)
      IF (M .EQ. 1) GO TO 50
      RM = FLOAT(M)
C
C     COMPUTE THE INVERSE DISCRETE FOURIER TRANSFORMATION
C     OF THE FIRST ROW OF THE MATRIX AND THE DISCRETE
C     FOURIER TRANSFORMATION OF THE RIGHT HAND SIDE VECTOR .
C
      T = (0.0E0,0.0E0)
      RI = -1.0E0
      DO 20 I1 = 1, M
         RI = RI + 1.0E0
C
C        MINIMIZE ERROR IN FORMING MULTIPLES OF 2*PI
C
```

```
           P = ((201.E0/32.E0)*RI + 1.93530717958647692528E-3*RI)/RM
C
           V1 = COS(P)
           V2 = SIN(P)
           E = CMPLX(V1,-V2)
           E1 = CMPLX(V1,V2)
           F = A(1)
           F1 = T1
           DO 10 I2 = 2, M
              F = E*F + A(I2)
              F1 = E1*F1 + X(I2)
   10      CONTINUE
           R(I1) = (E1*F1)/(E*F)
           T = T + R(I1)
   20 CONTINUE
C
C     COMPUTE THE SOLUTION OF THE GIVEN SYSTEM BY
C     THE INVERSE DISCRETE FOURIER TRANSFORMATION .
C
      X(1) = T/RM
      RI = 0.0E0
      DO 40 I1 = 2, M
         RI = RI + 1.0E0
C
C        MINIMIZE ERROR IN FORMING MULTIPLES OF 2*PI .
C
         P = ((201.E0/32.E0)*RI + 1.93530717958647692528E-3*RI)/RM
C
         V1 = COS(P)
         V2 = SIN(P)
         E = CMPLX(V1,-V2)
         F = R(1)
         DO 30 I2 = 2, M
            F = E*F + R(I2)
   30    CONTINUE
         X(I1) = E*F/RM
   40 CONTINUE
   50 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE CQRS(A,Q,S,M,L,LDQ,LDS)
      INTEGER M,L,LDQ,LDS
      REAL A(M),Q(LDQ,L),S(LDS,L)
C
C     CQRS COMPUTES THE QR FACTORIZATION IN THE FORM
C     A * R(INVERSE) = Q
C     OF THE REAL COLUMN-CIRCULANT MATRIX A .
C
C     ON ENTRY
C
C        A       REAL(M)
C                THE FIRST COLUMN OF THE COLUMN-CIRCULANT MATRIX .
C                ON RETURN A IS UNALTERED .
C
C        M       INTEGER
C                THE NUMBER OF ROWS OF THE MATRICES A AND Q .
C                M MUST BE AT LEAST AS LARGE AS L .
C
C        L       INTEGER
C                THE NUMBER OF COLUMNS OF THE MATRICES A AND Q
C                AND THE ORDER OF THE UPPER TRIANGULAR MATRIX S .
C
C        LDQ     INTEGER
C                THE LEADING DIMENSION OF THE ARRAY Q .
C
C        LDS     INTEGER
C                THE LEADING DIMENSION OF THE ARRAY S .
C
C     ON RETURN
C
C        Q       REAL(M,L)
C                THE Q MATRIX OF THE FACTORIZATION .
C                THE COLUMNS OF Q ARE ORTHONORMAL .
C
C        S       REAL(L,L)
C                THE INVERSE OF THE R MATRIX OF THE FACTORIZATION .
C                ELEMENTS BELOW THE MAIN DIAGONAL ARE NOT ACCESSED .
C
C     TOEPLITZ PACKAGE. THIS VERSION DATED 07/23/82 .
C
C     SUBROUTINES AND FUNCTIONS
C
C        LINPACK ... SAXPY,SDOT,SSCAL,SNRM2
C
C     INTERNAL VARIABLES
C
      INTEGER I,J,J1,JI
      REAL SCALE,SNRM2
      REAL C,SDOT
C
C     INITIALIZATION (LAST COLUMN OF Q USED AS WORK VECTOR) .
C
      DO 10 I = 1, M
         Q(I,1) = A(I)
```

```
         Q(I,L) = A(I)
   10 CONTINUE
C
C     RECURRENT PROCESS FOR THE LATTICE ALGORITHM WITH NORMALIZATION .
C
      DO 70 J1 = 1, L
         J = J1 + 1
         SCALE = 1.0E0/SNRM2(M,Q(1,J1),1)
         IF (J1 .EQ. L) GO TO 60
            C = -SCALE*(Q(M,J1)*Q(1,L) +
     *           SDOT(M-1,Q(1,J1),1,Q(2,L),1))/SNRM2(M,Q(1,L),1)
            Q(1,J) = Q(M,J1) + C*Q(1,L)
            DO 20 I = 2, M
               Q(I,J) = Q(I-1,J1) + C*Q(I,L)
   20       CONTINUE
            IF (J .EQ. L) GO TO 30
               Q(1,L) = Q(1,L) + C*Q(M,J1)
               CALL SAXPY(M-1,C,Q(1,J1),1,Q(2,L),1)
   30       CONTINUE
            S(1,J) = C
            IF (J .EQ. 2) GO TO 50
               DO 40 I = 2, J1
                  JI = J - I
                  S(I,J) = S(I-1,J1) + C*S(JI,J1)
   40          CONTINUE
   50       CONTINUE
   60    CONTINUE
         CALL SSCAL(M,SCALE,Q(1,J1),1)
         S(J1,J1) = 1.0E0
         CALL SSCAL(J1,SCALE,S(1,J1),1)
   70 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE CQRC(A,Q,S,M,L,LDQ,LDS)
      INTEGER M,L,LDQ,LDS
      COMPLEX A(M),Q(LDQ,L),S(LDS,L)
C
C     CQRC COMPUTES THE QR FACTORIZATION IN THE FORM
C     A * R(INVERSE) = Q
C     OF THE COMPLEX COLUMN-CIRCULANT MATRIX A .
C
C     ON ENTRY
C
C        A       COMPLEX(M)
C                THE FIRST COLUMN OF THE COLUMN-CIRCULANT MATRIX .
C                ON RETURN A IS UNALTERED .
C
C        M       INTEGER
C                THE NUMBER OF ROWS OF THE MATRICES A AND Q .
C                M MUST BE AT LEAST AS LARGE AS L .
C
C        L       INTEGER
C                THE NUMBER OF COLUMNS OF THE MATRICES A AND Q
C                AND THE ORDER OF THE UPPER TRIANGULAR MATRIX S .
C
C        LDQ     INTEGER
C                THE LEADING DIMENSION OF THE ARRAY Q .
C
C        LDS     INTEGER
C                THE LEADING DIMENSION OF THE ARRAY S .
C
C     ON RETURN
C
C        Q       COMPLEX(M,L)
C                THE Q MATRIX OF THE FACTORIZATION .
C                THE COLUMNS OF Q ARE ORTHONORMAL .
C
C        S       COMPLEX(L,L)
C                THE INVERSE OF THE R MATRIX OF THE FACTORIZATION .
C                ELEMENTS BELOW THE MAIN DIAGONAL ARE NOT ACCESSED .
C
C     TOEPLITZ PACKAGE. THIS VERSION DATED 07/23/82 .
C
C     SUBROUTINES AND FUNCTIONS
C
C        LINPACK ... CAXPY,CDOTC,CSSCAL,SCNRM2
C        FORTRAN ... CONJG
C
C     INTERNAL VARIABLES
C
      INTEGER I,J,J1,JI
      REAL SCALE,SCNRM2
      COMPLEX C,CDOTC
C
C     INITIALIZATION (LAST COLUMN OF Q USED AS WORK VECTOR) .
C
      DO 10 I = 1, M
```

```
         Q(I,1) = A(I)
         Q(I,L) = A(I)
   10 CONTINUE
C
C     RECURRENT PROCESS FOR THE LATTICE ALGORITHM WITH NORMALIZATION .
C
      DO 70 J1 = 1, L
         J = J1 + 1
         SCALE = 1.0E0/SCNRM2(M,Q(1,J1),1)
         IF (J1 .EQ. L) GO TO 60
            C = -SCALE*(CONJG(Q(M,J1))*Q(1,L) +
     *           CDOTC(M-1,Q(1,J1),1,Q(2,L),1))/SCNRM2(M,Q(1,L),1)
            Q(1,J) = Q(M,J1) + C*Q(1,L)
            DO 20 I = 2, M
               Q(I,J) = Q(I-1,J1) + C*Q(I,L)
   20       CONTINUE
            IF (J .EQ. L) GO TO 30
               Q(1,L) = Q(1,L) + C*Q(M,J1)
               CALL CAXPY(M-1,C,Q(1,J1),1,Q(2,L),1)
   30       CONTINUE
            S(1,J) = C
            IF (J .EQ. 2) GO TO 50
               DO 40 I = 2, J1
                  JI = J - I
                  S(I,J) = S(I-1,J1) + C*S(JI,J1)
   40          CONTINUE
   50       CONTINUE
   60    CONTINUE
         CALL CSSCAL(M,SCALE,Q(1,J1),1)
         S(J1,J1) = (1.0E0,0.0E0)
         CALL CSSCAL(J1,SCALE,S(1,J1),1)
   70 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE TGSLS(A,X,R,M,L,LDA)
      INTEGER M,L,LDA
      REAL A(LDA,1),X(M,L),R(1)
C
C     TGSLS CALLS TGSLS1 TO SOLVE THE REAL LINEAR SYSTEM
C     A * X = B
C     WITH THE TG - MATRIX A .
C
C     ON ENTRY
C
C        A       REAL(M**2,2*L - 1)
C                THE FIRST ROW OF BLOCKS OF THE TG - MATRIX
C                FOLLOWED BY ITS FIRST COLUMN OF BLOCKS BEGINNING
C                WITH THE SECOND BLOCK. EACH BLOCK IS REPRESENTED
C                BY COLUMNS. ON RETURN A IS UNALTERED .
C
C        X       REAL(M*L)
C                THE RIGHT HAND SIDE VECTOR B .
C
C        R       REAL(M**2*(2*L + 3) + M)
C                A WORK VECTOR .
C
C        M       INTEGER
C                THE ORDER OF THE BLOCKS OF THE MATRIX A .
C
C        L       INTEGER
C                THE NUMBER OF BLOCKS IN A ROW OR COLUMN
C                OF THE MATRIX A .
C
C        LDA     INTEGER
C                THE LEADING DIMENSION OF THE ARRAY A .
C
C     ON RETURN
C
C        X       THE SOLUTION VECTOR .
C
C     TOEPLITZ PACKAGE. THIS VERSION DATED 07/23/82 .
C
C     SUBROUTINES AND FUNCTIONS
C
C        TOEPLITZ PACKAGE ... TGSLS1
C
C     INTERNAL VARIABLES
C
      INTEGER MM,MML,MML1,MML2,MML3,MML4,MML5,MML6
C
C     CALL SUBROUTINE TGSLS1
C
      MM = M**2
      MML = MM*(L - 1) + 1
      MML1 = 2*MML - 1
      MML2 = MML1 + MM
      MML3 = MML2 + MM
      MML4 = MML3 + MM
```

```
      MML5 = MML4 + MM
      MML6 = MML5 + MM
C
      CALL TGSLS1(A,A(1,L+1),X,X,R,R(MML),R(MML1),R(MML2),
     *            R(MML3),R(MML4),R(MML5),R(MML6),M,L,LDA)
C
      RETURN
      END
```

```
      SUBROUTINE TGSLS1(A1,A2,B,X,C1,C2,R1,R2,R3,R5,R6,R,M,L,LDA)
      INTEGER M,L,LDA
      REAL A1(LDA,L),A2(LDA,1),B(M,L),X(M,L),C1(M,M,1),
     *     C2(M,M,1),R1(M,M),R2(M,M),R3(M,M),R5(M,M),R6(M,M),R(M)
C
C     TGSLS1 SOLVES THE REAL LINEAR SYSTEM
C     A * X = B
C     WITH THE TG - MATRIX A .
C
C     ON ENTRY
C
C        A1      REAL(M**2,L)
C                THE FIRST ROW OF BLOCKS OF THE TG - MATRIX A .
C                EACH BLOCK IS REPRESENTED BY COLUMNS .
C                ON RETURN A1 IS UNALTERED .
C
C        A2      REAL(M**2,L - 1)
C                THE FIRST COLUMN OF BLOCKS OF THE TG - MATRIX A
C                BEGINNING WITH THE SECOND BLOCK. EACH BLOCK IS
C                REPRESENTED BY COLUMNS. ON RETURN A2 IS UNALTERED .
C
C        B       REAL(M*L)
C                THE RIGHT HAND SIDE VECTOR .
C                ON RETURN B IS UNALTERED .
C
C        C1      REAL(M,M,L - 1)
C                A WORK ARRAY .
C
C        C2      REAL(M,M,L - 1)
C                A WORK ARRAY .
C
C        R1      REAL(M,M)
C                A WORK ARRAY .
C
C        R2      REAL(M,M)
C                A WORK ARRAY .
C
C        R3      REAL(M,M)
C                A WORK ARRAY .
C
C        R5      REAL(M,M)
C                A WORK ARRAY .
C
C        R6      REAL(M,M)
C                A WORK ARRAY .
C
C        R       REAL(M)
C                A WORK VECTOR .
C
C        M       INTEGER
C                THE ORDER OF THE BLOCKS OF THE MATRIX A .
C
C        L       INTEGER
C                THE NUMBER OF BLOCKS IN A ROW OR COLUMN
C                OF THE MATRIX A .
```

```
C
C        LDA     INTEGER
C                THE LEADING DIMENSION OF THE ARRAY A .
C
C     ON RETURN
C
C        X       REAL(M*L)
C                THE SOLUTION VECTOR. X MAY COINCIDE WITH B .
C
C     TOEPLITZ PACKAGE. THIS VERSION DATED 07/23/82 .
C
C     SUBROUTINES AND FUNCTIONS
C
C        LINPACK ... SAXPY,SGEFA,SGESL
C                ... (FOR IN-LINE SAXPY, SEE DIRECTIONS IN COMMENTS)
C
C     INTERNAL VARIABLES
C
      INTEGER I,I1,I2,I3,II,J,N,N1,N2
C
C     SOLVE THE SYSTEM WITH THE PRINCIPAL MINOR OF ORDER M .
C
      I3 = 1
      DO 20 J = 1, M
         DO 10 I = 1, M
            C1(I,J,1) = A1(I3,1)
            R1(I,J) = A1(I3,1)
            R3(I,J) = R1(I,J)
            I3 = I3 + 1
   10    CONTINUE
         X(J,1) = B(J,1)
   20 CONTINUE
      CALL SGEFA(R3,M,M,R,II)
      CALL SGESL(R3,M,M,R,X(1,1),0)
      IF (L .EQ. 1) GO TO 420
C
C     RECURRENT PROCESS FOR SOLVING THE SYSTEM
C     WITH THE TG - MATRIX FOR N = 2, L .
C
      DO 410 N = 2, L
C
C        COMPUTE MULTIPLES OF THE FIRST AND LAST BLOCK COLUMNS OF
C        THE INVERSE OF THE PRINCIPAL MINOR OF ORDER M*N .
C
         N1 = N - 1
         N2 = N - 2
         I3 = 1
         DO 40 J = 1, M
            DO 30 I = 1, M
               R5(I,J) = A2(I3,N1)
               R6(I,J) = A1(I3,N)
               I3 = I3 + 1
   30       CONTINUE
   40    CONTINUE
```

```
      IF (N .EQ. 2) GO TO 100
         DO 60 J = 1, M
            DO 50 I = 1, M
               C1(I,J,N1) = R2(I,J)
   50       CONTINUE
   60    CONTINUE
         DO 90 I1 = 1, N2
            I2 = N - I1
            DO 80 J = 1, M
               I3 = 1
               DO 70 I = 1, M
C FOR IN-LINE SAXPY, ACTIVATE NEXT 5 LINES AND DEACTIVATE FOLLOWING 3 .
C                 DO 65 II = 1, M
C                    R5(II,J) = R5(II,J) + C1(I,J,I2)*A2(I3,I1)
C                    R6(II,J) = R6(II,J) + C2(I,J,I1)*A1(I3,I1+1)
C                    I3 = I3 + 1
C  65             CONTINUE
                  CALL SAXPY(M,C1(I,J,I2),A2(I3,I1),1,R5(1,J),1)
                  CALL SAXPY(M,C2(I,J,I1),A1(I3,I1+1),1,R6(1,J),1)
                  I3 = I3 + M
   70          CONTINUE
   80       CONTINUE
   90    CONTINUE
  100 CONTINUE
      DO 120 J = 1, M
         DO 110 I = 1, M
            R2(I,J) = -R5(I,J)
  110    CONTINUE
         CALL SGESL(R3,M,M,R,R2(1,J),0)
  120 CONTINUE
      DO 140 J = 1, M
         DO 130 I = 1, M
            R3(I,J) = R6(I,J)
            R6(I,J) = -C1(I,J,1)
  130    CONTINUE
  140 CONTINUE
      DO 160 J = 1, M
         DO 150 I = 1, M
C FOR IN-LINE SAXPY, ACTIVATE NEXT 3 LINES AND DEACTIVATE FOLLOWING 1 .
C           DO 145 II = 1, M
C              C1(II,J,1) = C1(II,J,1) + R2(I,J)*R3(II,I)
C 145       CONTINUE
            CALL SAXPY(M,R2(I,J),R3(1,I),1,C1(1,J,1),1)
  150    CONTINUE
  160 CONTINUE
      CALL SGEFA(R6,M,M,R,II)
      DO 180 J = 1, M
         CALL SGESL(R6,M,M,R,R3(1,J),0)
         DO 170 I = 1, M
C FOR IN-LINE SAXPY, ACTIVATE NEXT 3 LINES AND DEACTIVATE FOLLOWING 1 .
C           DO 165 II = 1, M
C              R1(II,J) = R1(II,J) + R3(I,J)*R5(II,I)
C 165       CONTINUE
            CALL SAXPY(M,R3(I,J),R5(1,I),1,R1(1,J),1)
```

```
  170       CONTINUE
  180    CONTINUE
         IF (N .EQ. 2) GO TO 320
            DO 200 J = 1, M
               DO 190 I = 1, M
                  R6(I,J) = C2(I,J,1)
  190          CONTINUE
  200       CONTINUE
            DO 310 I1 = 2, N1
               IF (I1 .EQ. N1) GO TO 230
                  DO 220 J = 1, M
                     DO 210 I = 1, M
                        R5(I,J) = C2(I,J,I1)
  210                CONTINUE
  220             CONTINUE
  230          CONTINUE
               DO 260 J = 1, M
                  DO 240 I = 1, M
                     C2(I,J,I1) = R6(I,J)
  240             CONTINUE
                  DO 250 I = 1, M
C FOR IN-LINE SAXPY, ACTIVATE NEXT 3 LINES AND DEACTIVATE FOLLOWING 1 .
C                    DO 245 II = 1, M
C                       C2(II,J,I1) = C2(II,J,I1) + R3(I,J)*C1(II,I,I1)
C 245                CONTINUE
                     CALL SAXPY(M,R3(I,J),C1(1,I,I1),1,C2(1,J,I1),1)
  250             CONTINUE
  260          CONTINUE
               DO 280 J = 1, M
                  DO 270 I = 1, M
C FOR IN-LINE SAXPY, ACTIVATE NEXT 3 LINES AND DEACTIVATE FOLLOWING 1 .
C                    DO 265 II = 1, M
C                       C1(II,J,I1) = C1(II,J,I1) + R2(I,J)*R6(II,I)
C 265                CONTINUE
                     CALL SAXPY(M,R2(I,J),R6(1,I),1,C1(1,J,I1),1)
  270             CONTINUE
  280          CONTINUE
               DO 300 J = 1, M
                  DO 290 I = 1, M
                     R6(I,J) = R5(I,J)
  290             CONTINUE
  300          CONTINUE
  310       CONTINUE
  320    CONTINUE
         DO 340 J = 1, M
            DO 330 I = 1, M
               C2(I,J,1) = R3(I,J)
  330       CONTINUE
  340    CONTINUE
C
C        COMPUTE THE SOLUTION OF THE SYSTEM WITH THE
C        PRINCIPAL MINOR OF ORDER M*N .
C
         DO 360 J = 1, M
```

```
          DO 350 I = 1, M
             R3(I,J) = R1(I,J)
  350     CONTINUE
          X(J,N) = B(J,N)
  360   CONTINUE
        DO 380 I1 = 1, N1
           I2 = N - I1
           I3 = 1
           DO 370 I = 1, M
C FOR IN-LINE SAXPY, ACTIVATE NEXT 4 LINES AND DEACTIVATE FOLLOWING 2 .
C             DO 365 II = 1, M
C                X(II,N) = X(II,N) - X(I,I2)*A2(I3,I1)
C                I3 = I3 + 1
C 365         CONTINUE
              CALL SAXPY(M,-X(I,I2),A2(I3,I1),1,X(1,N),1)
              I3 = I3 + M
  370      CONTINUE
  380   CONTINUE
        CALL SGEFA(R3,M,M,R,II)
        CALL SGESL(R3,M,M,R,X(1,N),0)
        DO 400 I1 = 1, N1
           DO 390 I = 1, M
C FOR IN-LINE SAXPY, ACTIVATE NEXT 3 LINES AND DEACTIVATE FOLLOWING 1 .
C             DO 385 II = 1, M
C                X(II,I1) = X(II,I1) + X(I,N)*C2(II,I,I1)
C 385         CONTINUE
              CALL SAXPY(M,X(I,N),C2(1,I,I1),1,X(1,I1),1)
  390      CONTINUE
  400   CONTINUE
  410 CONTINUE
  420 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE TGSLC(A,X,R,M,L,LDA)
      INTEGER M,L,LDA
      COMPLEX A(LDA,1),X(M,L),R(1)
C
C     TGSLC CALLS TGSLC1 TO SOLVE THE COMPLEX LINEAR SYSTEM
C     A * X = B
C     WITH THE TG - MATRIX A .
C
C     ON ENTRY
C
C        A       COMPLEX(M**2,2*L - 1)
C                THE FIRST ROW OF BLOCKS OF THE TG - MATRIX
C                FOLLOWED BY ITS FIRST COLUMN OF BLOCKS BEGINNING
C                WITH THE SECOND BLOCK. EACH BLOCK IS REPRESENTED
C                BY COLUMNS. ON RETURN A IS UNALTERED .
C
C        X       COMPLEX(M*L)
C                THE RIGHT HAND SIDE VECTOR B .
C
C        R       COMPLEX(M**2*(2*L + 3) + M)
C                A WORK VECTOR .
C
C        M       INTEGER
C                THE ORDER OF THE BLOCKS OF THE MATRIX A .
C
C        L       INTEGER
C                THE NUMBER OF BLOCKS IN A ROW OR COLUMN
C                OF THE MATRIX A .
C
C        LDA     INTEGER
C                THE LEADING DIMENSION OF THE ARRAY A .
C
C     ON RETURN
C
C        X       THE SOLUTION VECTOR .
C
C     TOEPLITZ PACKAGE. THIS VERSION DATED 07/23/82 .
C
C     SUBROUTINES AND FUNCTIONS
C
C        TOEPLITZ PACKAGE ... TGSLC1
C
C     INTERNAL VARIABLES
C
      INTEGER MM,MML,MML1,MML2,MML3,MML4,MML5,MML6
C
C     CALL SUBROUTINE TGSLC1
C
      MM = M**2
      MML = MM*(L - 1) + 1
      MML1 = 2*MML - 1
      MML2 = MML1 + MM
      MML3 = MML2 + MM
      MML4 = MML3 + MM
```

```
      MML5 = MML4 + MM
      MML6 = MML5 + MM
C
      CALL TGSLC1(A,A(1,L+1),X,X,R,R(MML),R(MML1),R(MML2),
     *            R(MML3),R(MML4),R(MML5),R(MML6),M,L,LDA)
C
      RETURN
      END
```

```
      SUBROUTINE TGSLC1(A1,A2,B,X,C1,C2,R1,R2,R3,R5,R6,R,M,L,LDA)
      INTEGER M,L,LDA
      COMPLEX A1(LDA,L),A2(LDA,1),B(M,L),X(M,L),C1(M,M,1),
     *     C2(M,M,1),R1(M,M),R2(M,M),R3(M,M),R5(M,M),R6(M,M),R(M)
C
C     TGSLC1 SOLVES THE COMPLEX LINEAR SYSTEM
C     A * X = B
C     WITH THE TG - MATRIX A .
C
C     ON ENTRY
C
C        A1       COMPLEX(M**2,L)
C                 THE FIRST ROW OF BLOCKS OF THE TG - MATRIX A .
C                 EACH BLOCK IS REPRESENTED BY COLUMNS .
C                 ON RETURN A1 IS UNALTERED .
C
C        A2       COMPLEX(M**2,L - 1)
C                 THE FIRST COLUMN OF BLOCKS OF THE TG - MATRIX A
C                 BEGINNING WITH THE SECOND BLOCK. EACH BLOCK IS
C                 REPRESENTED BY COLUMNS. ON RETURN A2 IS UNALTERED .
C
C        B        COMPLEX(M*L)
C                 THE RIGHT HAND SIDE VECTOR .
C                 ON RETURN B IS UNALTERED .
C
C        C1       COMPLEX(M,M,L - 1)
C                 A WORK ARRAY .
C
C        C2       COMPLEX(M,M,L - 1)
C                 A WORK ARRAY .
C
C        R1       COMPLEX(M,M)
C                 A WORK ARRAY .
C
C        R2       COMPLEX(M,M)
C                 A WORK ARRAY .
C
C        R3       COMPLEX(M,M)
C                 A WORK ARRAY .
C
C        R5       COMPLEX(M,M)
C                 A WORK ARRAY .
C
C        R6       COMPLEX(M,M)
C                 A WORK ARRAY .
C
C        R        COMPLEX(M)
C                 A WORK VECTOR .
C
C        M        INTEGER
C                 THE ORDER OF THE BLOCKS OF THE MATRIX A .
C
C        L        INTEGER
C                 THE NUMBER OF BLOCKS IN A ROW OR COLUMN
C                 OF THE MATRIX A .
```

```
C
C          LDA      INTEGER
C                   THE LEADING DIMENSION OF THE ARRAY A .
C
C       ON RETURN
C
C          X        COMPLEX(M*L)
C                   THE SOLUTION VECTOR. X MAY COINCIDE WITH B .
C
C       TOEPLITZ PACKAGE. THIS VERSION DATED 07/23/82 .
C
C       SUBROUTINES AND FUNCTIONS
C
C          LINPACK ... CAXPY,CGEFA,CGESL
C                  ... (FOR IN-LINE CAXPY, SEE DIRECTIONS IN COMMENTS)
C
C       INTERNAL VARIABLES
C
        INTEGER I,I1,I2,I3,II,J,N,N1,N2
C
C       SOLVE THE SYSTEM WITH THE PRINCIPAL MINOR OF ORDER M .
C
        I3 = 1
        DO 20 J = 1, M
           DO 10 I = 1, M
              C1(I,J,1) = A1(I3,1)
              R1(I,J) = A1(I3,1)
              R3(I,J) = R1(I,J)
              I3 = I3 + 1
   10      CONTINUE
           X(J,1) = B(J,1)
   20   CONTINUE
        CALL CGEFA(R3,M,M,R,II)
        CALL CGESL(R3,M,M,R,X(1,1),0)
        IF (L .EQ. 1) GO TO 420
C
C       RECURRENT PROCESS FOR SOLVING THE SYSTEM
C       WITH THE TG - MATRIX FOR N = 2, L .
C
        DO 410 N = 2, L
C
C          COMPUTE MULTIPLES OF THE FIRST AND LAST BLOCK COLUMNS OF
C          THE INVERSE OF THE PRINCIPAL MINOR OF ORDER M*N .
C
           N1 = N - 1
           N2 = N - 2
           I3 = 1
           DO 40 J = 1, M
              DO 30 I = 1, M
                 R5(I,J) = A2(I3,N1)
                 R6(I,J) = A1(I3,N)
                 I3 = I3 + 1
   30         CONTINUE
   40      CONTINUE
```

```
      IF (N .EQ. 2) GO TO 100
         DO 60 J = 1, M
            DO 50 I = 1, M
               C1(I,J,N1) = R2(I,J)
   50       CONTINUE
   60    CONTINUE
         DO 90 I1 = 1, N2
            I2 = N - I1
            DO 80 J = 1, M
               I3 = 1
               DO 70 I = 1, M
C FOR IN-LINE CAXPY, ACTIVATE NEXT 5 LINES AND DEACTIVATE FOLLOWING 3 .
C                 DO 65 II = 1, M
C                    R5(II,J) = R5(II,J) + C1(I,J,I2)*A2(I3,I1)
C                    R6(II,J) = R6(II,J) + C2(I,J,I1)*A1(I3,I1+1)
C                    I3 = I3 + 1
C  65             CONTINUE
                  CALL CAXPY(M,C1(I,J,I2),A2(I3,I1),1,R5(1,J),1)
                  CALL CAXPY(M,C2(I,J,I1),A1(I3,I1+1),1,R6(1,J),1)
                  I3 = I3 + M
   70          CONTINUE
   80       CONTINUE
   90    CONTINUE
  100 CONTINUE
      DO 120 J = 1, M
         DO 110 I = 1, M
            R2(I,J) = -R5(I,J)
  110    CONTINUE
         CALL CGESL(R3,M,M,R,R2(1,J),0)
  120 CONTINUE
      DO 140 J = 1, M
         DO 130 I = 1, M
            R3(I,J) = R6(I,J)
            R6(I,J) = -C1(I,J,1)
  130    CONTINUE
  140 CONTINUE
      DO 160 J = 1, M
         DO 150 I = 1, M
C FOR IN-LINE CAXPY, ACTIVATE NEXT 3 LINES AND DEACTIVATE FOLLOWING 1 .
C           DO 145 II = 1, M
C              C1(II,J,1) = C1(II,J,1) + R2(I,J)*R3(II,I)
C 145       CONTINUE
            CALL CAXPY(M,R2(I,J),R3(1,I),1,C1(1,J,1),1)
  150    CONTINUE
  160 CONTINUE
      CALL CGEFA(R6,M,M,R,II)
      DO 180 J = 1, M
         CALL CGESL(R6,M,M,R,R3(1,J),0)
         DO 170 I = 1, M
C FOR IN-LINE CAXPY, ACTIVATE NEXT 3 LINES AND DEACTIVATE FOLLOWING 1 .
C           DO 165 II = 1, M
C              R1(II,J) = R1(II,J) + R3(I,J)*R5(II,I)
C 165       CONTINUE
            CALL CAXPY(M,R3(I,J),R5(1,I),1,R1(1,J),1)
```

```
  170          CONTINUE
  180       CONTINUE
            IF (N .EQ. 2) GO TO 320
               DO 200 J = 1, M
                  DO 190 I = 1, M
                     R6(I,J) = C2(I,J,1)
  190             CONTINUE
  200          CONTINUE
               DO 310 I1 = 2, N1
                  IF (I1 .EQ. N1) GO TO 230
                     DO 220 J = 1, M
                        DO 210 I = 1, M
                           R5(I,J) = C2(I,J,I1)
  210                   CONTINUE
  220                CONTINUE
  230             CONTINUE
                  DO 260 J = 1, M
                     DO 240 I = 1, M
                        C2(I,J,I1) = R6(I,J)
  240                CONTINUE
                     DO 250 I = 1, M
C FOR IN-LINE CAXPY, ACTIVATE NEXT 3 LINES AND DEACTIVATE FOLLOWING 1 .
C                       DO 245 II = 1, M
C                          C2(II,J,I1) = C2(II,J,I1) + R3(I,J)*C1(II,I,I1)
C 245                   CONTINUE
                        CALL CAXPY(M,R3(I,J),C1(1,I,I1),1,C2(1,J,I1),1)
  250                CONTINUE
  260             CONTINUE
                  DO 280 J = 1, M
                     DO 270 I = 1, M
C FOR IN-LINE CAXPY, ACTIVATE NEXT 3 LINES AND DEACTIVATE FOLLOWING 1 .
C                       DO 265 II = 1, M
C                          C1(II,J,I1) = C1(II,J,I1) + R2(I,J)*R6(II,I)
C 265                   CONTINUE
                        CALL CAXPY(M,R2(I,J),R6(1,I),1,C1(1,J,I1),1)
  270                CONTINUE
  280             CONTINUE
                  DO 300 J = 1, M
                     DO 290 I = 1, M
                        R6(I,J) = R5(I,J)
  290                CONTINUE
  300             CONTINUE
  310          CONTINUE
  320       CONTINUE
            DO 340 J = 1, M
               DO 330 I = 1, M
                  C2(I,J,1) = R3(I,J)
  330          CONTINUE
  340       CONTINUE
C
C           COMPUTE THE SOLUTION OF THE SYSTEM WITH THE
C           PRINCIPAL MINOR OF ORDER M*N .
C
            DO 360 J = 1, M
```

```
           DO 350 I = 1, M
              R3(I,J) = R1(I,J)
  350      CONTINUE
           X(J,N) = B(J,N)
  360   CONTINUE
        DO 380 I1 = 1, N1
           I2 = N - I1
           I3 = 1
           DO 370 I = 1, M
C FOR IN-LINE CAXPY, ACTIVATE NEXT 4 LINES AND DEACTIVATE FOLLOWING 2 .
C             DO 365 II = 1, M
C                X(II,N) = X(II,N) - X(I,I2)*A2(I3,I1)
C                I3 = I3 + 1
C 365         CONTINUE
              CALL CAXPY(M,-X(I,I2),A2(I3,I1),1,X(1,N),1)
              I3 = I3 + M
  370      CONTINUE
  380   CONTINUE
        CALL CGEFA(R3,M,M,R,II)
        CALL CGESL(R3,M,M,R,X(1,N),0)
        DO 400 I1 = 1, N1
           DO 390 I = 1, M
C FOR IN-LINE CAXPY, ACTIVATE NEXT 3 LINES AND DEACTIVATE FOLLOWING 1 .
C             DO 385 II = 1, M
C                X(II,I1) = X(II,I1) + X(I,N)*C2(II,I,I1)
C 385         CONTINUE
              CALL CAXPY(M,X(I,N),C2(1,I,I1),1,X(1,I1),1)
  390      CONTINUE
  400   CONTINUE
  410 CONTINUE
  420 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE CTSLC(A,X,R,M,L,LDA)
      INTEGER M,L,LDA
      COMPLEX A(LDA,L),X(M,L),R(1)
C
C     CTSLC SOLVES THE COMPLEX LINEAR SYSTEM
C     A * X = B
C     WITH THE CT - MATRIX A .
C
C     ON ENTRY
C
C        A       COMPLEX(2*M - 1,L)
C                THE FIRST ROW OF BLOCKS OF THE CT - MATRIX .
C                EACH BLOCK IS REPRESENTED BY ITS FIRST ROW
C                FOLLOWED BY ITS FIRST COLUMN BEGINNING WITH THE
C                SECOND ELEMENT. ON RETURN A HAS BEEN DESTROYED .
C
C        X       COMPLEX(M*L)
C                THE RIGHT HAND SIDE VECTOR B .
C
C        R       COMPLEX(MAX(2*M - 2,2*L))
C                A WORK VECTOR .
C
C        M       INTEGER
C                THE ORDER OF THE BLOCKS OF THE MATRIX A .
C
C        L       INTEGER
C                THE NUMBER OF BLOCKS IN A ROW OR COLUMN
C                OF THE MATRIX A .
C
C        LDA     INTEGER
C                THE LEADING DIMENSION OF THE ARRAY A .
C
C     ON RETURN
C
C        X       THE SOLUTION VECTOR .
C
C     TOEPLITZ PACKAGE. THIS VERSION DATED 07/23/82 .
C
C     SUBROUTINES AND FUNCTIONS
C
C        TOEPLITZ PACKAGE ... SALWC,TSLC
C        FORTRAN ... FLOAT
C
C     INTERNAL VARIABLES
C
      INTEGER I1,I2
      REAL RL
C
      RL = FLOAT(L)
C
C     REDUCE THE CT - MATRIX TO A BLOCK-DIAGONAL MATRIX
C     BY THE INVERSE DISCRETE FOURIER TRANSFORMATION .
C
      CALL SALWC(A,R,R(L+1),2*M - 1,L,LDA,-1)
C
```

```
C     COMPUTE THE DISCRETE FOURIER TRANSFORMATION OF
C     THE RIGHT HAND SIDE VECTOR .
C
      CALL SALWC(X,R,R(L+1),M,L,M,1)
C
C     SOLVE THE BLOCK-DIAGONAL SYSTEM, BLOCKS OF WHICH
C     ARE T - MATRICES .
C
      DO 10 I2 = 1, L
         CALL TSLC(A(1,I2),X(1,I2),R,M)
   10 CONTINUE
C
C     COMPUTE THE SOLUTION OF THE GIVEN SYSTEM BY
C     THE INVERSE DISCRETE FOURIER TRANSFORMATION .
C
      CALL SALWC(X,R,R(L+1),M,L,M,-1)
C
      DO 30 I2 = 1, L
         DO 20 I1 = 1, M
            X(I1,I2) = X(I1,I2)/RL
   20    CONTINUE
   30 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE CCSLC(A,X,R,M,L,LDA)
      INTEGER M,L,LDA
      COMPLEX A(LDA,L),X(M,L),R(1)
C
C     CCSLC SOLVES THE COMPLEX LINEAR SYSTEM
C     A * X = B
C     WITH THE CC - MATRIX A .
C
C     ON ENTRY
C
C        A       COMPLEX(M,L)
C                THE FIRST ROW OF BLOCKS OF THE CC - MATRIX .
C                EACH BLOCK IS REPRESENTED BY ITS FIRST ROW .
C                ON RETURN A HAS BEEN DESTROYED .
C
C        X       COMPLEX(M*L)
C                THE RIGHT HAND SIDE VECTOR B .
C
C        R       COMPLEX(MAX(M,2*L))
C                A WORK VECTOR .
C
C        M       INTEGER
C                THE ORDER OF THE BLOCKS OF THE MATRIX A .
C
C        L       INTEGER
C                THE NUMBER OF BLOCKS IN A ROW OR COLUMN
C                OF THE MATRIX A .
C
C        LDA     INTEGER
C                THE LEADING DIMENSION OF THE ARRAY A .
C
C     ON RETURN
C
C        X       THE SOLUTION VECTOR .
C
C     TOEPLITZ PACKAGE. THIS VERSION DATED 07/23/82 .
C
C     SUBROUTINES AND FUNCTIONS
C
C        TOEPLITZ PACKAGE ... CSLC,SALWC
C        FORTRAN ... FLOAT
C
C     INTERNAL VARIABLES
C
      INTEGER I1,I2
      REAL RL
C
      RL = FLOAT(L)
C
C     REDUCE THE CC - MATRIX TO A BLOCK-DIAGONAL MATRIX
C     BY THE INVERSE DISCRETE FOURIER TRANSFORMATION .
C
      CALL SALWC(A,R,R(L+1),M,L,LDA,-1)
C
C     COMPUTE THE DISCRETE FOURIER TRANSFORMATION OF
```

```
C     THE RIGHT HAND SIDE VECTOR .
C
      CALL SALWC(X,R,R(L+1),M,L.M,1)
C
C     SOLVE THE BLOCK-DIAGONAL SYSTEM, BLOCKS OF WHICH
C     ARE C - MATRICES .
C
      DO 10 I2 = 1, L
         CALL CSLC(A(1,I2),X(1,I2),R,M)
   10 CONTINUE
C
C     COMPUTE THE SOLUTION OF THE GIVEN SYSTEM BY
C     THE INVERSE DISCRETE FOURIER TRANSFORMATION .
C
      CALL SALWC(X,R,R(L+1),M,L,M,-1)
C
      DO 30 I2 = 1, L
         DO 20 I1 = 1, M
            X(I1,I2) = X(I1,I2)/RL
   20    CONTINUE
   30 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE CGSLC(A,X,R,M,L,LDA)
      INTEGER M,L,LDA
      COMPLEX A(LDA,L),X(M,L),R(1)
C
C     CGSLC SOLVES THE COMPLEX LINEAR SYSTEM
C     A * X = B
C     WITH THE CG - MATRIX A .
C
C     ON ENTRY
C
C        A        COMPLEX(M**2,L)
C                 THE FIRST ROW OF BLOCKS OF THE CG - MATRIX .
C                 EACH BLOCK IS REPRESENTED BY COLUMNS .
C                 ON RETURN A HAS BEEN DESTROYED .
C
C        X        COMPLEX(M*L)
C                 THE RIGHT HAND SIDE VECTOR B .
C
C        R        COMPLEX(MAX(M,2*L))
C                 A WORK VECTOR .
C
C        M        INTEGER
C                 THE ORDER OF THE BLOCKS OF THE MATRIX A .
C
C        L        INTEGER
C                 THE NUMBER OF BLOCKS IN A ROW OR COLUMN
C                 OF THE MATRIX A .
C
C        LDA      INTEGER
C                 THE LEADING DIMENSION OF THE ARRAY A .
C
C     ON RETURN
C
C        X        THE SOLUTION VECTOR .
C
C     TOEPLITZ PACKAGE. THIS VERSION DATED 07/23/82 .
C
C     SUBROUTINES AND FUNCTIONS
C
C        TOEPLITZ PACKAGE ... SALWC
C        LINPACK ... CGEFA,CGESL
C        FORTRAN ... FLOAT
C
C     INTERNAL VARIABLES
C
      INTEGER I1,I2,II
      REAL RL
C
      RL = FLOAT(L)
C
C     REDUCE THE CG - MATRIX TO A BLOCK-DIAGONAL MATRIX
C     BY THE INVERSE DISCRETE FOURIER TRANSFORMATION .
C
      CALL SALWC(A,R,R(L+1),M**2,L,LDA,-1)
C
```

```
C     COMPUTE THE DISCRETE FOURIER TRANSFORMATION OF
C     THE RIGHT HAND SIDE VECTOR .
C
      CALL SALWC(X,R,R(L+1),M,L,M,1)
C
C     SOLVE THE BLOCK-DIAGONAL SYSTEM, BLOCKS OF WHICH
C     ARE G - MATRICES .
C
      DO 10 I2 = 1, L
         CALL CGEFA(A(1,I2),M,M,R,II)
         CALL CGESL(A(1,I2),M,M,R,X(1,I2),0)
   10 CONTINUE
C
C     COMPUTE THE SOLUTION OF THE GIVEN SYSTEM BY
C     THE INVERSE DISCRETE FOURIER TRANSFORMATION .
C
      CALL SALWC(X,R,R(L+1),M,L,M,-1)
C
      DO 30 I2 = 1, L
         DO 20 I1 = 1, M
            X(I1,I2) = X(I1,I2)/RL
   20    CONTINUE
   30 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE SALWC(A,R1,R2,M,L,LDA,JOB)
      INTEGER M,L,LDA,JOB
      COMPLEX A(LDA,L),R1(L),R2(L)
C
C     SALWC COMPUTES THE DIRECT OR INVERSE DISCRETE FOURIER
C     TRANSFORMATION FOR ROWS OF A COMPLEX RECTANGULAR MATRIX .
C
C     ON ENTRY
C
C        A       COMPLEX(M,L)
C                THE INPUT MATRIX .
C
C        R1      COMPLEX(L)
C                A WORK VECTOR .
C
C        R2      COMPLEX(L)
C                A WORK VECTOR .
C
C        M       INTEGER
C                THE NUMBER OF ROWS OF THE MATRIX A .
C
C        L       INTEGER
C                THE NUMBER OF COLUMNS OF THE MATRIX A .
C
C        LDA     INTEGER
C                THE LEADING DIMENSION OF THE ARRAY A .
C
C        JOB     INTEGER
C                = 1 FOR DIRECT FOURIER TRANSFORMATION .
C                = -1 FOR INVERSE FOURIER TRANSFORMATION .
C
C     ON RETURN
C
C        A       THE TRANSFORMED ROWS OF THE MATRIX .
C
C     TOEPLITZ PACKAGE. THIS VERSION DATED 07/23/82 .
C
C     SUBROUTINES AND FUNCTIONS
C
C        FORTRAN ... CMPLX,COS,FLOAT,SIN
C
C     INTERNAL VARIABLES
C
      INTEGER I,I1,I2
      REAL P,RI,RL,V1,V2
      COMPLEX E,F
C
      IF (L .EQ. 1) GO TO 60
      RL = FLOAT(L)
C
      R1(1) = (1.0E0,0.0E0)
      RI = 0.0E0
      DO 10 I1 = 2, L
         RI = RI + 1.0E0
C
```

```
C        MINIMIZE ERROR IN FORMING MULTIPLES OF 2*PI .
C
         P = ((201.E0/32.E0)*RI + 1.93530717958647692528E-3*RI)/RL
C
         V1 = COS(P)
         V2 = SIN(P)
         IF (JOB .EQ. (-1)) V2 = -V2
         R1(I1) = CMPLX(V1,V2)
   10 CONTINUE
      DO 50 I = 1, M
         DO 30 I1 = 1, L
            E = R1(I1)
            F = A(I,1)
            DO 20 I2 = 2, L
               F = E*F + A(I,I2)
   20       CONTINUE
            R2(I1) = E*F
   30    CONTINUE
         DO 40 I1 = 1, L
            A(I,I1) = R2(I1)
   40    CONTINUE
   50 CONTINUE
   60 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE CTGSLC(A,X,R,M,L,K,LDA)
      INTEGER M,L,K,LDA
      COMPLEX A(LDA,K),X(M,L,K),R(1)
C
C     CTGSLC SOLVES THE COMPLEX LINEAR SYSTEM
C     A * X = B
C     WITH THE CTG - MATRIX A .
C
C     ON ENTRY
C
C        A       COMPLEX(M**2*(2*L - 1),K)
C                THE FIRST ROW OF OUTER BLOCKS OF THE CTG - MATRIX .
C                EACH OUTER BLOCK IS REPRESENTED BY ITS FIRST ROW
C                OF INNER BLOCKS FOLLOWED BY ITS FIRST COLUMN
C                OF INNER BLOCKS BEGINNING WITH THE SECOND BLOCK .
C                EACH INNER BLOCK IS REPRESENTED BY COLUMNS .
C                ON RETURN A HAS BEEN DESTROYED .
C
C        X       COMPLEX(M*L*K)
C                THE RIGHT HAND SIDE VECTOR B .
C
C        R       COMPLEX(MAX(M**2*(2*L + 3) + M,2*K))
C                A WORK VECTOR .
C
C        M       INTEGER
C                THE ORDER OF THE INNER BLOCKS OF THE MATRIX A .
C
C        L       INTEGER
C                THE NUMBER OF INNER BLOCKS IN A ROW OR COLUMN
C                OF AN OUTER BLOCK OF THE MATRIX A .
C
C        K       INTEGER
C                THE NUMBER OF OUTER BLOCKS IN A ROW OR COLUMN
C                OF THE MATRIX A .
C
C        LDA     INTEGER
C                THE LEADING DIMENSION OF THE ARRAY A .
C
C     ON RETURN
C
C        X       THE SOLUTION VECTOR .
C
C     TOEPLITZ PACKAGE. THIS VERSION DATED 07/23/82 .
C
C     SUBROUTINES AND FUNCTIONS
C
C        TOEPLITZ PACKAGE ... SALWC,TGSLC
C        FORTRAN ... FLOAT
C
C     INTERNAL VARIABLES
C
      INTEGER I1,I2,I3,ML,MM
      REAL RK
C
```

```
      RK = FLOAT(K)
      MM = M**2
      ML = M*L
C
C     REDUCE THE CTG - MATRIX TO A BLOCK-DIAGONAL MATRIX
C     BY THE INVERSE DISCRETE FOURIER TRANSFORMATION .
C
      CALL SALWC(A,R,R(K+1),MM*(2*L - 1),K,LDA,-1)
C
C     COMPUTE THE DISCRETE FOURIER TRANSFORMATION OF
C     THE RIGHT HAND SIDE VECTOR .
C
      CALL SALWC(X,R,R(K+1),ML,K,ML,1)
C
C     SOLVE THE BLOCK-DIAGONAL SYSTEM, BLOCKS OF WHICH
C     ARE TG - MATRICES .
C
      DO 10 I3 = 1, K
         CALL TGSLC(A(1,I3),X(1,1,I3),R,M,L,MM)
   10 CONTINUE
C
C     COMPUTE THE SOLUTION OF THE GIVEN SYSTEM BY
C     THE INVERSE DISCRETE FOURIER TRANSFORMATION .
C
      CALL SALWC(X,R,R(K+1),ML,K,ML,-1)
C
      DO 40 I3 = 1, K
         DO 30 I2 = 1, L
            DO 20 I1 = 1, M
               X(I1,I2,I3) = X(I1,I2,I3)/RK
   20       CONTINUE
   30    CONTINUE
   40 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE CCTSLC(A,X,R,M,L,K,LDA)
      INTEGER M,L,K,LDA
      COMPLEX A(LDA,K),X(M,L,K),R(1)
C
C     CCTSLC SOLVES THE COMPLEX LINEAR SYSTEM
C     A * X = B
C     WITH THE CCT - MATRIX A .
C
C     ON ENTRY
C
C        A       COMPLEX((2*M - 1)*L,K)
C                THE FIRST ROW OF OUTER BLOCKS OF THE CCT - MATRIX .
C                EACH OUTER BLOCK IS REPRESENTED BY ITS FIRST ROW
C                OF INNER BLOCKS. EACH INNER BLOCK IS REPRESENTED
C                BY ITS FIRST ROW FOLLOWED BY ITS FIRST COLUMN
C                BEGINNING WITH THE SECOND ELEMENT .
C                ON RETURN A HAS BEEN DESTROYED .
C
C        X       COMPLEX(M*L*K)
C                THE RIGHT HAND SIDE VECTOR B .
C
C        R       COMPLEX(MAX(2*M - 2,2*L,2*K))
C                A WORK VECTOR .
C
C        M       INTEGER
C                THE ORDER OF THE INNER BLOCKS OF THE MATRIX A .
C
C        L       INTEGER
C                THE NUMBER OF INNER BLOCKS IN A ROW OR COLUMN
C                OF AN OUTER BLOCK OF THE MATRIX A .
C
C        K       INTEGER
C                THE NUMBER OF OUTER BLOCKS IN A ROW OR COLUMN
C                OF THE MATRIX A .
C
C        LDA     INTEGER
C                THE LEADING DIMENSION OF THE ARRAY A .
C
C     ON RETURN
C
C        X       THE SOLUTION VECTOR .
C
C     TOEPLITZ PACKAGE. THIS VERSION DATED 07/23/82 .
C
C     SUBROUTINES AND FUNCTIONS
C
C        TOEPLITZ PACKAGE ... CTSLC,SALWC
C        FORTRAN ... FLOAT
C
C     INTERNAL VARIABLES
C
      INTEGER I1,I2,I3,M2,ML
      REAL RK
C
```

```
      RK = FLOAT(K)
      M2 = 2*M - 1
      ML = M*L
C
C     REDUCE THE CCT - MATRIX TO A BLOCK-DIAGONAL MATRIX
C     BY THE INVERSE DISCRETE FOURIER TRANSFORMATION .
C
      CALL SALWC(A,R,R(K+1),M2*L,K,LDA,-1)
C
C     COMPUTE THE DISCRETE FOURIER TRANSFORMATION OF
C     THE RIGHT HAND SIDE VECTOR .
C
      CALL SALWC(X,R,R(K+1),ML,K,ML,1)
C
C     SOLVE THE BLOCK-DIAGONAL SYSTEM, BLOCKS OF WHICH
C     ARE CT - MATRICES .
C
      DO 10 I3 = 1, K
         CALL CTSLC(A(1,I3),X(1,1,I3),R,M,L,M2)
   10 CONTINUE
C
C     COMPUTE THE SOLUTION OF THE GIVEN SYSTEM BY
C     THE INVERSE DISCRETE FOURIER TRANSFORMATION .
C
      CALL SALWC(X,R,R(K+1),ML,K,ML,-1)
C
      DO 40 I3 = 1, K
         DO 30 I2 = 1, L
            DO 20 I1 = 1, M
               X(I1,I2,I3) = X(I1,I2,I3)/RK
   20       CONTINUE
   30    CONTINUE
   40 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE CCCSLC(A,X,R,M,L,K,LDA)
      INTEGER M,L,K,LDA
      COMPLEX A(LDA,K),X(M,L,K),R(1)
C
C     CCCSLC SOLVES THE COMPLEX LINEAR SYSTEM
C     A * X = B
C     WITH THE CCC - MATRIX A .
C
C     ON ENTRY
C
C        A       COMPLEX(M*L,K)
C                THE FIRST ROW OF OUTER BLOCKS OF THE CCC - MATRIX .
C                EACH OUTER BLOCK IS REPRESENTED BY ITS FIRST ROW
C                OF INNER BLOCKS. EACH INNER BLOCK IS REPRESENTED
C                BY ITS FIRST ROW. ON RETURN A HAS BEEN DESTROYED .
C
C        X       COMPLEX(M*L*K)
C                THE RIGHT HAND SIDE VECTOR B .
C
C        R       COMPLEX(MAX(M,2*L,2*K))
C                A WORK VECTOR .
C
C        M       INTEGER
C                THE ORDER OF THE INNER BLOCKS OF THE MATRIX A .
C
C        L       INTEGER
C                THE NUMBER OF INNER BLOCKS IN A ROW OR COLUMN
C                OF AN OUTER BLOCK OF THE MATRIX A .
C
C        K       INTEGER
C                THE NUMBER OF OUTER BLOCKS IN A ROW OR COLUMN
C                OF THE MATRIX A .
C
C        LDA     INTEGER
C                THE LEADING DIMENSION OF THE ARRAY A .
C
C     ON RETURN
C
C        X       THE SOLUTION VECTOR .
C
C     TOEPLITZ PACKAGE. THIS VERSION DATED 07/23/82 .
C
C     SUBROUTINES AND FUNCTIONS
C
C        TOEPLITZ PACKAGE ... CCSLC,SALWC
C        FORTRAN ... FLOAT
C
C     INTERNAL VARIABLES
C
      INTEGER I1,I2,I3,ML
      REAL RK
C
      RK = FLOAT(K)
      ML = M*L
C
```

```
C     REDUCE THE CCC - MATRIX TO A BLOCK-DIAGONAL MATRIX
C     BY THE INVERSE DISCRETE FOURIER TRANSFORMATION .
C
      CALL SALWC(A,R,R(K+1),ML,K,LDA,-1)
C
C     COMPUTE THE DISCRETE FOURIER TRANSFORMATION OF
C     THE RIGHT HAND SIDE VECTOR .
C
      CALL SALWC(X,R,R(K+1),ML,K,ML,1)
C
C     SOLVE THE BLOCK-DIAGONAL SYSTEM, BLOCKS OF WHICH
C     ARE CC - MATRICES .
C
      DO 10 I3 = 1, K
         CALL CCSLC(A(1,I3),X(1,1,I3),R,M,L,M)
   10 CONTINUE
C
C     COMPUTE THE SOLUTION OF THE GIVEN SYSTEM BY
C     THE INVERSE DISCRETE FOURIER TRANSFORMATION .
C
      CALL SALWC(X,R,R(K+1),ML,K,ML,-1)
C
      DO 40 I3 = 1, K
         DO 30 I2 = 1, L
            DO 20 I1 = 1, M
               X(I1,I2,I3) = X(I1,I2,I3)/RK
   20       CONTINUE
   30    CONTINUE
   40 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE CCGSLC(A,X,R,M,L,K,LDA)
      INTEGER M,L,K,LDA
      COMPLEX A(LDA,K),X(M,L,K),R(1)
C
C     CCGSLC SOLVES THE COMPLEX LINEAR SYSTEM
C     A * X = B
C     WITH THE CCG - MATRIX A .
C
C     ON ENTRY
C
C        A       COMPLEX(M**2*L,K)
C                THE FIRST ROW OF OUTER BLOCKS OF THE CCG - MATRIX .
C                EACH OUTER BLOCK IS REPRESENTED BY ITS FIRST ROW
C                OF INNER BLOCKS. EACH INNER BLOCK IS REPRESENTED
C                BY COLUMNS. ON RETURN A HAS BEEN DESTROYED .
C
C        X       COMPLEX(M*L*K)
C                THE RIGHT HAND SIDE VECTOR B .
C
C        R       COMPLEX(MAX(M,2*L,2*K))
C                A WORK VECTOR .
C
C        M       INTEGER
C                THE ORDER OF THE INNER BLOCKS OF THE MATRIX A .
C
C        L       INTEGER
C                THE NUMBER OF INNER BLOCKS IN A ROW OR COLUMN
C                OF AN OUTER BLOCK OF THE MATRIX A .
C
C        K       INTEGER
C                THE NUMBER OF OUTER BLOCKS IN A ROW OR COLUMN
C                OF THE MATRIX A .
C
C        LDA     INTEGER
C                THE LEADING DIMENSION OF THE ARRAY A .
C
C     ON RETURN
C
C        X       THE SOLUTION VECTOR .
C
C     TOEPLITZ PACKAGE. THIS VERSION DATED 07/23/82 .
C
C     SUBROUTINES AND FUNCTIONS
C
C        TOEPLITZ PACKAGE ... CGSLC,SALWC
C        FORTRAN ... FLOAT
C
C     INTERNAL VARIABLES
C
      INTEGER I1,I2,I3,ML,MM
      REAL RK
C
      RK = FLOAT(K)
      MM = M**2
```

```
      ML = M*L
C
C     REDUCE THE CCG - MATRIX TO A BLOCK-DIAGONAL MATRIX
C     BY THE INVERSE DISCRETE FOURIER TRANSFORMATION .
C
      CALL SALWC(A,R,R(K+1),MM*L,K,LDA,-1)
C
C     COMPUTE THE DISCRETE FOURIER TRANSFORMATION OF
C     THE RIGHT HAND SIDE VECTOR .
C
      CALL SALWC(X,R,R(K+1),ML,K,ML,1)
C
C     SOLVE THE BLOCK-DIAGONAL SYSTEM, BLOCKS OF WHICH
C     ARE CG - MATRICES .
C
      DO 10 I3 = 1, K
         CALL CGSLC(A(1,I3),X(1,1,I3),R,M,L,MM)
   10 CONTINUE
C
C     COMPUTE THE SOLUTION OF THE GIVEN SYSTEM BY
C     THE INVERSE DISCRETE FOURIER TRANSFORMATION .
C
      CALL SALWC(X,R,R(K+1),ML,K,ML,-1)
C
      DO 40 I3 = 1, K
         DO 30 I2 = 1, L
            DO 20 I1 = 1, M
               X(I1,I2,I3) = X(I1,I2,I3)/RK
   20       CONTINUE
   30    CONTINUE
   40 CONTINUE
      RETURN
      END
```

Distribution for ANL-83-16

Internal:

J. M. Boyle
M. K. Butler (10)
W. J. Cody
W. R. Cowell
J. J. Dongarra
K. W. Dritz
B. S. Garbow (45)
K. L. Kliewer
A. B. Krisciunas
P. C. Messina
D. M. Pahis
T. M. Woods (2)
G. W. Pieper
R. J. Royston
ANL Patent Department
ANL Contract File
ANL Libraries
TIS Files (6)

External:

DOE-TIC, for distribution per UC-32 (186)
Manager, Chicago Operations Office, DOE
Mathematics and Computer Science Division Review Committee:
J. C. Browne, U. Texas, Austin
S. Gerhart, Wang Institute, Tynsboro, MA
L. P. Kadanoff, U. of Chicago
W. C. Lynch, Xerox Corp., Palo Alto
J. M. Ortega, U. Virginia
D. L. Wallace, U. of Chicago
M. F. Wheeler, Rice U.
T. Aird, IMSL, Houston (10)
O. B. Arushanian, Moscow State U.
D. Austin, Office of Basic Energy Sciences, DOE
J. R. Bunch, UCSD
G. Cybenko, Tufts U.
P. Gaffney, ORNL
S. Hammarling, NAG, Oxford, England
G. Michael, LLL
W. Miller, U. of Arizona
C. Moler, U. of New Mexico
M. K. Samarin, Moscow State U.
E. E. Tyrtyshnikov, Moscow State U.
C. Van Loan, Cornell U.
V. V. Voevodin, Academy of Sciences, U.S.S.R.