NPS69-84-005

# NAVAL POSTGRADUATE SCHOOL

## Monterey, California

# USE OF THE TENSOR PRODUCT FOR NUMERICAL WEATHER PREDICTION BY THE FINITE ELEMENT METHOD - PART 2

R. E. Newton
June 1984

Report for Period
April 1984 - June 1984





…epared for:

…val Environmental Prediction Research Facility
…nterey, California 93943

NAVAL POSTGRADUATE SCHOOL
Monterey, California

Commodore R. H. Shumaker
Superintendent

D. A. Schrady
Provost

The work reported herein was supported by the Naval Environmental Prediction Research Facility.



This report was prepared by:

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)



| REPORT DOCUMENTATION PAGE | | READ INSTRUCTIONS BEFORE COMPLETING FORM |
|---|---|---|
| 1. REPORT NUMBER NPS69-84-005 | 2. GOVT ACCESSION NO. | 3. RECIPIENT'S CATALOG NUMBER |
| 4. TITLE (and Subtitle) USE OF THE TENSOR PRODUCT FOR NUMERICAL WEATHER PREDICTION BY THE FINITE ELEMENT METHOD - PART 2 | | 5. TYPE OF REPORT & PERIOD COVERED Interim, April 84 - June 84 |
| | | 6. PERFORMING ORG. REPORT NUMBER |
| 7. AUTHOR(s) R. E. Newton | | 8. CONTRACT OR GRANT NUMBER(s) N6685684WR84103 |
| 9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93943 | | 10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 61153W |
| 11. CONTROLLING OFFICE NAME AND ADDRESS Naval Environmental Prediction Research Facility, Monterey, California, 93943 | | 12. REPORT DATE June 1984 |
| | | 13. NUMBER OF PAGES |
| 14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) | | 15. SECURITY CLASS. (of this report) Unclassified |
| | | 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE |

16. DISTRIBUTION STATEMENT (of this Report)

Approved for Public Release; Distribution Unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Finite element, numerical weather prediction, tensor product

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This is Part 2 of a report-pair concerning application of the tensor product in solving large sets of simultaneous linear equations arising in finite element formulations of Numerical Weather Prediction problems. A rectangular region having a graded mesh with Dirichlet boundary conditions on all four edges is considered. Coefficient matrices are the "mass" matrix and the "stiffness" matrix of the finite element method. For the

DD FORM 1473, 1 JAN 73 EDITION OF 1 NOV 65 IS OBSOLETE

S/N 0102-LF-014-6601

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered) stiffness matrix, which appears in Poisson's equation, operation counts and storage requirements are compared with corresponding numbers for solutions by successive over-relaxation and Gaussian elimination. FORTRAN programs for implementation of the tensor product formulations are given.

(block 20, continued)

# USE OF THE TENSOR PRODUCT FOR NUMERICAL WEATHER PREDICTION BY THE FINITE ELEMENT METHOD - PART 2.

## Introduction

This is the second installment of a report-pair concerning implementation of tensor product factoring of coefficient matrices in applications of the finite element method to numerical weather prediction. It was noted in Part 1 (Ref. 1) that these techniques were introduced in numerical weather prediction by Staniforth and Mitchell (Ref. 2). Discussed in Part 1 are applications in which the "mass" matrix for a grid such as that shown in Fig. 1 is factored as the tensor product of two matrices.

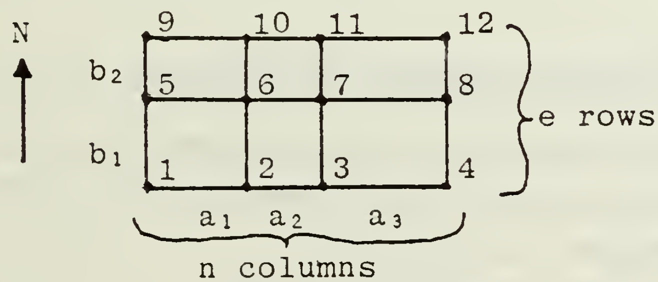


Fig. 1. Node numbering and spacing.

One of these matrices (MA) depends solely upon the nodal spacing in the east-west direction ($a_i$) and the other (MB) depends only on the north-south spacing ($b_i$). We began with the set of simultaneous linear equations

$$M\ w = v, \qquad \langle 1 \rangle$$

where M (the "mass" matrix) is symmetric, ne x ne, and w and v are column vectors of height ne. M and v are input quantities and w is sought. The tensor product representation of M is

$$M = MB * MA, \qquad \langle 2 \rangle$$

where MB and MA are tridiagonal, symmetric matrices, e x e and n x n, respectively. (The tensor product and matrices MA and MB are defined in Appendix A.) This representation allowed <1> to be rewritten as

$$MA\ W\ MB = V, \qquad \langle 3 \rangle$$

where W is n x e and the successive columns are subvectors of w corresponding to the rows of Fig. 1. V is also n x e and similarly derived from v. Boundary conditions considered were a cyclic condition in the east-west direction and either homogeneous Neumann conditions (normal derivative zero) or nonhomogeneous Dirichlet conditions (specified nonzero values) on the northern and southern edges.

The present report discards the cyclic east-west boundary condition and deals with two cases:

(1) Solutions of <3> with nonhomogeneous Dirichlet conditions on all four edges;

(2) Solution of Poisson's equation for the same region with nonhomogeneous Dirichlet conditions on all four edges.

## Mass Matrix - Dirichlet Boundary Conditions

Effects of the Dirichlet boundary conditions on the solution process are most readily understood by considering the following partitioned form of <1>:

$$\begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix} \begin{bmatrix} w_b \\ w_c \end{bmatrix} = \begin{bmatrix} v_b \\ v_c \end{bmatrix} \qquad \langle 4 \rangle$$

In <4> the w vector has been rearranged so that all of the boundary values are in the subvector $w_b$ and the interior ("center") values are in $w_c$. A similar reordering has been applied to v and M. If the boundary values of w are prescribed, then $w_b$ is known and only $w_c$ remains to be found. Expanding the lower partition of <4> and placing the known terms on the right gives

$$M_{22} w_c = v_c - M_{21} w_b, \qquad \langle 5 \rangle$$

or, letting $v_c' = v_c - M_{21} w_b$, we have

$$M_{22} w_c = v_c'. \qquad \langle 5' \rangle$$

We consider now how the strategy just described can be applied when the tensor product resolution of M has been used to convert <1> into <3>. In the matrix W the prescribed boundary values occupy the first and last columns and the top and bottom rows. Denote this border matrix, including an (n-2) x (e-2) null matrix inside, by WB.

Calculate

$$VB = MA\ WB\ MB \qquad <6>$$

and now form

$$V' = V - VB. \qquad <7>$$

Now define a set of submatrices MA1, MB1, W1, and V1 obtained from MA, MB, W, and V', respectively, by removing the first and last columns and the top and bottom rows. The reduced problem becomes

$$MA1\ W1\ MB1 = V1 \qquad <8>$$

As described in Ref. 1, <8> may be solved by standard Gaussian elimination procedures. A computer program (GAUSS4) which carries out these calculations is listed in Appendix B. The subroutines of GAUSS4 are designed for substitution in the program devised by Hinsman (Ref. 5).

## Poisson's Equation - Dirichlet Boundary Conditions

As noted above, Staniforth and Mitchell (Ref. 2) appear to have been first in applying the tensor product resolution to Poisson's equation in a numerical weather prediction problem using the finite element method. Additional detail is given in earlier papers by Dorr (Ref. 3) and by Lynch, Rice, and Thomas (Ref. 4).

Finite element discretization of Poisson's equation for the region of Fig. 1 results in a set of simultaneous linear equations which may be written in matrix form as

$$K\ w = v, \qquad <9>$$

where vectors v and w are, respectively, given and unknown. As for <1>, each has length ne and the coefficient matrix K is ne x ne, symmetric, sparse, and block tridiagonal. K is called the "stiffness" matrix in finite element parlance.

It is easily shown that K is expressible as the sum of two tensor products as follows:

$$K = MB * SA + SB * MA. \qquad <10>$$

The new matrices SA and SB are symmetric, tridiagonal and depend only on the $a_i$ and $b_i$, respectively. Explicit formulas for SA and SB are given in Appendix A.

Using the definition of the tensor product and again converting the vectors w and v into the n x e rectangular matrices W and V, <9> may be written as

$$SA\ W\ MB + MA\ W\ SB = V. \qquad <11>$$

Before solving <11> we must first take account of the Dirichlet boundary conditions on the four edges of the region. As in solving <3>, the given boundary values are in the first and last columns and top and bottom rows of W. As before, we let WB be an n x e matrix containing the given boundary values, together with zeros at locations corresponding to interior nodes. Calculate

$$VB = SA\ WB\ MB + MA\ WB\ SB, \qquad <12>$$

and then form

$$V' = V - VB. \qquad <7>$$

The remaining step again parallels that used when applying the Dirichlet boundary conditions to <3>. Specifically, we introduce submatrices MA1, MB1, SA1, SB1, W1, and V1 obtained from MA, MB, SA, SB, W, and V', respectively, by removing the first and last columns and the top and bottom rows. The reduced problem becomes

$$SA1\ W1\ MB1 + MA1\ W1\ SB1 = V1. \qquad <13>$$

To solve <13> we first need the complete solution of the eigenproblem

$$SB1\ p_i = \lambda_i\ MB1\ p_i, \qquad <14>$$

where $p_i$ is the ith eigenvector and $\lambda_i$ is the corresponding eigenvalue. We write the complete solution in the form

$$SB1\ P = MB1\ P\ \Lambda, \qquad <14'>$$

where P is the (e-2) x (e-2) modal matrix whose columns are the $p_i$ and $\Lambda$ is the (diagonal) spectral matrix whose elements are the $\lambda_i$. We specify that the modal matrix is normalized so that

$$PT\ MB1\ P = I, \qquad <15>$$

where I is the identity matrix of order e-2 and PT is the transpose of P. If both sides of <13> are postmultiplied by P and <14'> is used to replace SB1 P, <13> becomes

$$SA1\ W1\ MB1\ P + MA1\ W1\ MB1\ P\ \Lambda = V1\ P. \qquad <16>$$

Let X = W1 MB1 P and U = V1 P, then <16> is equivalent to

$$(SA1 + \lambda_i\ MA1)\ x_i = u_i, \qquad i = 1, e-2, \qquad <17>$$

where $x_i$ and $u_i$ are, respectively, the ith columns of X and U. Since the coefficient matrix in <17> is tridiagonal, the Gaussian elimination process, i.e., factoring, forward reduction, and back-substitution, is computationally economical. The final step consists of a matrix multiplication to obtain

$$W1 = X\ PT. \qquad \text{<18>}$$

Since W1 contains the w values at all interior nodes and the boundary values were known in advance, the solution is complete. A FORTRAN program (GAUSS5) which implements the tensor product solution for Poisson's equation is given in Appendix C.

## Operation Counts and Storage Requirements - Poisson's Equation

In Ref. 1 comparisons of floating point operation counts and storage requirements were made for solutions of <1>. Substitution of the boundary conditions considered here in place of those considered in Ref. 1 has a negligible effect on both operation counts and storage requirements. Accordingly, no further comparison is given here for solutions of <1>.

Solution of Poisson's equation (<9>) using the tensor product resolution <10> of K is more costly in terms of computation and storage than the previously studied applications to <1>. In Table 1 the number of floating point operations and the required number of coefficient matrix storage locations are compared for three different solution methods. These are SOR (successive over-relaxation), SKY (skyline storage and Gauss elimination), and TENSOR (the scheme described above). A floating point operation is defined to be one multiplication (or division) plus one addition (or subtraction). The exact operation counts would be polynomials in n and e. Only the highest degree terms are given in the table. Since it is not possible to predict the number of iterations per solution using SOR, the operation count given for that algorithm is for a single iteration. In

Table 1 a storage location corresponds to 8 bytes. For the comparison it is assumed that each floating point number requires 8 bytes of storage and an integer requires 4 bytes. The storage requirement given for SOR is based on the compact storage scheme described by Franke and Salinas (Ref. 6).

TABLE 1. Operation Counts and Storage Requirements.

| ALGORITHM | NUMBER OF OPERATIONS PER SOLUTION | NUMBER OF STORAGE LOCATIONS FOR COEFFICIENT MATRICES |
|---|---|---|
| SOR | 10 en (1) | 13 en |
| SKY | 2 $en^2$ | $en^2$ |
| TENSOR | 2 $en^2$ | $e^2$ |

Note: 1. Number of operations per iteration.

It is perhaps surprising to note that the number of operations for TENSOR is no fewer than for SKY. Turning attention to storage requirements reveals that for a large problem (e = n = 100, say) the SKY storage requirement for the stiffness matrix is 8 megabytes, compared with 1 megabyte for SOR and 80 kilobytes for TENSOR. It is this comparison which is the compelling reason for preferring TENSOR. It is acknowledged that there is overhead associated with the one-time solution of the eigenvalue problem <14>, but the tridiagonal form of matrices SB1 and MB1 makes the amount of computation comparable with that required for a single solution of Poisson's equation. Since two solutions of Poisson's equation are required at each time step, the overhead is clearly negligible.

It is not feasible to make a definitive comparison between the number of operations required for SOR and those required for the other two algorithms. If the number of iterations is less than 0.2 e, then SOR will be more economical and the storage tradeoff would need to be weighed.

## Conclusions

It has been demonstrated that Dirichlet boundary conditions on all edges of the region are easily incorporated in solution processes which use tensor product resolution of the coefficient matrix. For very large problems the tensor product algorithm uses much less core storage than alternative choices. The computational expense of a solution to Poisson's equation is substantially the same for Gaussian elimination and for the tensor product scheme. It is expected that successive over-relaxation is almost always more expensive.

## List of References

1. Newton, R. E., "Use of the Tensor Product for Numerical Weather Prediction by the Finite Element Method," NPS69-84-001, Naval Postgraduate School, April 1984.

2. Staniforth, A. N., and H. L. Mitchell, "A Semi-Implicit Finite Element Barotropic Model," Monthly Weather Review, v. 105, p. 154-169, February 1977.

3. Dorr, F. W., "The Direct Solution of the Discrete Poisson Equation on a Rectangle," SIAM Review, v. 29, p. 248-263, April 1970.

4. Lynch, R. E., J. R. Rice and D. H. Thomas, "Tensor Product Analysis of Partial Difference Equations," Bull. Amer. Math. Soc. v. 70, p. 378-384, 1964.

5. Hinsman, D. E., "Numerical Simulation of Atmospheric Flow on Variable Grids using the Galerkin Finite Element Method," Doctoral Dissertation, Naval Postgraduate School, March 1983.

6. Franke, Richard, and David Salinas, An Efficient Method for Solving Stiff Transient Field Problems arising from FEM Formulations," NPS53-79-002, Naval Postgraduate School, March 1979.

## APPENDIX A - TENSOR PRODUCT AND MATRIX DEFINITIONS

Tensor Product  The tensor product of matrices C and D may be represented in block partition form as

$$C * D = \begin{bmatrix} c_{11}D & c_{12}D & c_{13}D \\ c_{21}D & c_{22}D & c_{23}D \\ c_{31}D & c_{32}D & c_{33}D \end{bmatrix}$$

where the $c_{ij}$ are the elements of C. Note that, if C and D have dimensions r x s and t x u, respectively, the tensor product has dimensions rt x su.

Definitions for matrices MA and SA are given below. The corresponding expressions for MB and SB may be obtained by substituting "b" for "a" throughout and replacing n by e. (Symbols $a_i$ and $b_i$ are defined in Fig. 1.)

$$\underset{(n=4)}{MA} = \frac{1}{6}\begin{bmatrix} 2a_1 & a_1 & 0 & 0 \\ a_1 & 2(a_1+a_2) & a_2 & 0 \\ 0 & a_2 & 2(a_2+a_3) & a_3 \\ 0 & 0 & a_3 & 2a_3 \end{bmatrix}.$$

$$\underset{(n=4)}{SA} = \begin{bmatrix} \frac{1}{a_1} & -\frac{1}{a_1} & 0 & 0 \\ -\frac{1}{a_1} & \frac{1}{a_1}+\frac{1}{a_2} & -\frac{1}{a_2} & 0 \\ 0 & -\frac{1}{a_2} & \frac{1}{a_2}+\frac{1}{a_3} & -\frac{1}{a_3} \\ 0 & 0 & -\frac{1}{a_3} & \frac{1}{a_3} \end{bmatrix}$$

APPENDIX B

PROGRAM to SOLVE M w = v with DIRICHLET BOUNDARY CONDITIONS

Listing: GAUSS4 FORTRAN

```
C  MAIN PROGRAM  MASS MATRIX USING TENSOR PRODUCT RESOLUTION
C
C  THIS PROGRAM IS DESIGNED TO TEST THE SCHEME (TENSOR)
C  WHICH RESOLVES THE MASS MATRIX INTO A TENSOR PRODUCT IN
C  ORDER TO SOLVE THE SYSTEM OF EQUATIONS M w = v.IN
C  THIS PROGRAM THERE ARE DIRICHLET BOUNDARY CONDITIONS ON
C  ALL 4 EDGES OF THE REGION.  THE PRESCRIBED BOUNDARY
C  VALUES ARE GIVEN IN THE CORRESPONDING LOCATIONS IN V.
C  THE SUBROUTINES MAY BE INSERTED IN THE PROGRAM DEVISED
C  BY HINSMAN.
C
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON/CM1A/NLAT,NLONG
      COMMON/CM8/A(Z1),B(Z1)
      COMMON AG(ZB),BG(ZC),GAD(ZK),GBD(ZL),MA(ZM),MB(ZN)
      DIMENSION V(ZP)
      READ(5,*)NLONG, NLAT
      LATX=NLAT+1
      WRITE(6,1000)
1000  FORMAT(//,'   MASS MATRIX - TENSOR PRODUCT RESOLUTION'
     1,/)
      WRITE(6,1001)NLONG,NLAT
      READ(5,*)A,B
      WRITE(6,500)A
      WRITE(6,503)B
503   FORMAT(/,'   B:  ',(24F3.0))
500   FORMAT(/,'   A:  ',(24F3.0))
1001  FORMAT('   NLONG =',I3,'      NLAT =',I3,/)
C  CONSTRUCT FACTORS, GAD AND GBD, OF MASS MATRIX
      CALL AMTRX3
      WRITE(6,501)AG
501   FORMAT(/,'   AG:  ',(12F4.1))
      WRITE(6,504)BG
504   FORMAT(/,'   BG:  ',(12F4.1))
      WRITE(6,1002)GAD
1002  FORMAT(/,'   GAD',/,(3X,6F7.3))
      WRITE(6,1004)GBD
1004  FORMAT(//,'   GBD',/,(3X,6F7.3))
      WRITE(6,1003)MA
      WRITE(6,1006)MB
      K=(NLAT-1)*NLONG
      L=NLAT*NLONG
      READ(5,*)V
      WRITE(6,510)V
C  CORRECT RIGHT-HAND SIDE FOR DIRICHLET CONDITION
      LONGM=NLONG-1
      DO 2 J=2,LONGM
      V(J+NLONG)=V(J+NLONG)-(GAD(2*J-1)*V(J-1)+GAD(2*J-2)
     1*V(J)+GAD(2*J+1)*V(J+1))*GBD(3)
2     V(K+J)=V(K+J)-(GAD(2*J-1)*V(L+J-1)+GAD(2*J-2)*V(L+J)
     1+GAD(2*J+1)*V(L+J+1))*GBD(2*LATX-1)
      CU=GBD(3)
      CL=GBD(2*LATX-1)
      GBD(3)=0.
      GBD(2*LATX-1)=0.
      DO 3 J=2,NLAT
      V((J-1)*NLONG+2)=V((J-1)*NLONG+2)-(GBD(2*J-1)*V((J-2)
     1*NLONG+1)+GBD(2*J-2)*V((J-1)*NLONG+1)+GBD(2*J+1)
     2*V(J*NLONG+1))*GAD(3)
3     V(J*NLONG-1)=V(J*NLONG-1)-(GBD(2*J-1)*V((J-1)*NLONG)
     1+GBD(2*J-2)*V(J*NLONG)+GBD(2*J+1)*V((J+1)*NLONG))
     2*GAD(2*NLONG-1)
      GBD(3)=CU
      GBD(2*LATX-1)=CL
      WRITE(6,510)V
C  PERFORM LDLT FACTORING OF GAD AND GBD
4     CALL FACT1(GAD,NLONG)
      CALL FACT1(GBD,LATX)
      WRITE(6,1002)GAD
      WRITE(6,1004)GBD
```

```
C   PERFORM FORWARD REDUCTION AND BACK-SUBSTITUTION USING
C   FACTORS OF GAD
       CALL BACKA1(GAD,V)
       WRITE(6,510)V
C   PERFORM FORWARD REDUCTION AND BACK-SUBSTITUTION USING
C   FACTORS OF GBD
       CALL BACKB1(GBD,V)
6      WRITE(6,510)V
510    FORMAT(/,' V: ',5F8.2,/,(4X,5F8.2))
1003   FORMAT(/,'    MA:',2X,36I3)
1006   FORMAT(/,'    MB:',2X,36I3)
       STOP
       END
C ******************************************************************
       SUBROUTINE FACT1(A,NN)
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
C   SUBROUTINE FACT1 PERFORMS L*D*LT FACTORING ON A SUBMATRIX
C   OF A SYMMETRIC TRIDIAGONAL MATRIX STORED IN SKYLINE FORM.
C   THE SUBMATRIX IS FORMED BY OMITTING THE FIRST AND LAST
C   COLUMNS AND ROWS OF THE INPUT MATRIX.
C .- - INPUT VARIABLES - -                                          .
C .    A(NWK) = INPUT MATRIX STORED IN COMPACTED FORM               .
C .    NN     = NUMBER OF COLUMNS (OR ROWS) IN INPUT MATRIX         .
C .    NWK    = NUMBER OF ELEMENTS BELOW SKYLINE (2*NN - 1)         .
C .   - - OUTPUT - -                                                .
C .    A(NWK) = D AND L - FACTORS OF INPUT SUBMATRIX                .
C .                                                                 .
C . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
       IMPLICIT REAL*8(A-H,O-Z)
       DIMENSION A(1)
C
C      PERFORM L*D*LT FACTORIZATION OF STIFFNESS MATRIX
C
       LONGMM=NN-2
       A(3)=0.
       DO 50 J=2,LONGMM
       TEMP=A(2*J+1)/A(2*(J-1))
       A(2*J)=A(2*J)-TEMP*A(2*J+1)
       IF(A(2*J))120,120,50
120    WRITE(IOUT,2000)N,A(KN)
       STOP
50     A(2*J+1)=TEMP
2000   FORMAT(//,' STOP - MATRIX NOT POSITIVE DEFINITE',//,
      1' NONPOSITIVE PIVOT FOR EQUATION  ',I4,//,' PIVOT = ',
      2E20.12)
       RETURN
       END
C *******************************************************************
       SUBROUTINE BACKA1(A,V)
C *******************************************************************
C   THIS SUBROUTINE PERFORMS THE FORWARD REDUCTION AND BACK-
C   SUBSTITUTION USING THE FACTORS OF GAD
C
       IMPLICIT REAL*8(A-H,O-Z)
       COMMON/CM1A/NLAT,NLONG
       DIMENSION A(1),V(1)
C
C      DEFINE LIMITS FOR DO-LOOPS
C
       NTM=NLAT-1
       LONGM=NLONG-1
       LONGMM=NLONG-2
C
C      REDUCE RIGHT-HAND-SIDE LOAD VECTOR
C
       DO 100 K=1,NTM
       DO 20 J=3,LONGM
20     V(K*NLONG+J)=V(K*NLONG+J)-V(K*NLONG+J-1)*A(2*J-1)
C
C      DIVIDE BY DIAGONAL ELEMENTS
C
       DO 40 J=1,LONGMM
40     V(K*NLONG+J+1)=V(K*NLONG+J+1)/A(2*J)
C
C      BACK-SUBSTITUTE
C
```

```
      DO 60 J=3,LONGM
      L=(K+1)*NLONG-J+1
      M=2*(NLONG-J)+3
60    V(L)=V(L)-V(L+1)*A(M)
100   CONTINUE
      RETURN
      END
C
C ***********************************************************************
      SUBROUTINE BACKB1(A,V)
C ***********************************************************************
C  THIS SUBROUTINE PERFORMS THE FORWARD REDUCTION AND BACK-
C  SUBSTITUTION USING THE FACTORS OF GBD.
C
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON/CM1A/NLAT,NLONG
      DIMENSION A(1),V(1)
C
C  DEFINE NEEDED INDEX VARIABLES
C
      LATX=NLAT+1
      LONGM=NLONG-1
C
C     REDUCE RIGHT-HAND-SIDE LOAD VECTOR
C
      DO 100 K=2,LONGM
      DO 20 J=3,NLAT
20    V(K+(J-1)*NLONG)=V(K+(J-1)*NLONG)-V(K+(J-2)*NLONG)
     1*A(2*J-1)
C
C  DIVIDE BY DIAGONAL ELEMENTS
C
      DO 40 J=2,NLAT
40    V(K+(J-1)*NLONG)=V(K+(J-1)*NLONG)/A(2*J-2)
C
C     BACK-SUBSTITUTE
C
      DO 60 J=3,NLAT
60    V(K+(LATX-J)*NLONG)=V(K+(LATX-J)*NLONG)
     1-V(K+(1+LATX-J)*NLONG)*A(2*(LATX-J)+3)
100   CONTINUE
      RETURN
      END
C ***********************************************************************
      SUBROUTINE AMTRX3
C ***********************************************************************
C  THIS SUBROUTINE FORMS THE MASS MATRIX IN THE FORM OF A
C  TENSOR PRODUCT OF THE GBD MATRIX AND THE GAD MATRIX.
C  THE FIRST OF THESE IS NLAT + 1 BY NLAT + 1, SYMMETRIC,
C  AND TRIDIAGONAL.  THE SECOND IS NLONG BY NLONG, SYMMET-
C  RIC, AND TRIDIAGONAL.  NOTE THAT THERE ISNO CYCLIC
C  BOUNDARY CONDITION IN THE EAST-WEST DIRECTION.  BOTH GBD
C  AND GAD ARE STORED IN SKYLINE VECTOR FORM (UPPER TRIANGLE
C  WITH SPACE FOR FILL-IN). INTEGER ADDRESS VECTORS MB AND
C  MA ARE ALSO GENERATED.
C
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON/CM1A/NLAT,NLONG
      COMMON/CM8/A(Z1),B(Z1)
      COMMON AG(ZB),BG(ZC),GAD(ZK),GBD(ZL),MA(ZM),MB(ZN)
C     DIMENSION BG(NLAT),AG(NLONG),GBD(2*NLAT-1),
C    1GAD(3*NLONG-3),MA(NLONG+1),MB(NLAT+2)
C
      LATX=NLAT+1
      LONGM=NLONG-1
C   FIND BG = (ELEMENT HEIGHT)/6.
      LONGM=NLONG-1
      DO 2 J=1,NLAT
2     BG(J)=B(1+LONGM*(J-1))/3.
C   GENERATE GBD
      GBD(1)=2.*BG(1)
      DO 4 J=2,NLAT
      K=2*(J-1)
      GBD(K)=2.*(BG(J-1)+BG(J))
4     GBD(K+1)=BG(J-1)
      GBD(2*NLAT)=2.*BG(NLAT)
```

```
      GBD(2*NLAT+1)=BG(NLAT)
C     FIND AG = (ELEMENT WIDTH)/6.
      DO 10 J=1,LONGM
10    AG(J)=A(J)/3.
C     GENERATE GAD
      GAD(1)=2.*AG(1)
      DO 12 J=2,LONGM
      K=2*(J-1)
      GAD(K)=2.*(AG(J-1)+AG(J))
12    GAD(K+1)=AG(J-1)
      GAD(2*LONGM)=2*AG(LONGM)
      GAD(2*LONGM+1)=AG(LONGM)
C     GENERATE DIRECTORY VECTORS
      MB(1)=1
      DO 16 J=1,NLAT
16    MB(J+1)=2*J
      MB(NLAT+2)=2*(NLAT+1)
      MA(1)=1
      DO 18 J=2,NLONG
18    MA(J)=2*(J-1)
      MA(NLONG+1)=2*NLONG
      RETURN
      END
```

APPENDIX C
PROGRAM - POISSON'S EQUATION with DIRICHLET BOUNDARY CONDITIONS

Listing: GAUSS5 FORTRAN

```
C   MAIN PROGRAM    STIFFNESS MATRIX USING TENSOR PRODUCT
C                   RESOLUTION
C
C   THIS PROGRAM IS DESIGNED TO TEST THE SCHEME WHICH
C   RESOLVES THE STIFFNESS MATRIX INTO A SUM OF TWO TENSOR
C   PRODUCTS IN ORDER TO SOLVE THE SYSTEM OF EQUATIONS
C   K W = V.   THERE ARE DIRICHLET BOUNDARYCONDITIONS ON
C   ALL 4 EDGES OF THE REGION.  THE PRESCRIBED BOUNDARY
C   VALUES ARE GIVEN IN THE CORRESPONDING LOCATIONS IN V.
C   THE SUBROUTINES MAY BE INSERTED IN THE PROGRAM DEVISED
C   BY HINSMAN.
C
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON/CM1A/NLAT,NLONG
      COMMON/CM8/A(Z1),B(Z1)
      COMMON AG(ZB),BG(ZC),GA1(ZK),SA1(ZK),GB1(ZL),SB1(ZL)
      DIMENSION V(ZP),W1(ZQ),P(ZR),D(ZS),U(ZT)
      READ(5, *)NLONG, NLAT
      LATX=NLAT+1
      WRITE(6,1000)
1000  FORMAT(//,'   STIFFNESS MATRIX - TENSOR PRODUCT
     1RESOLUTION',/)
      WRITE(6,1001)NLONG,NLAT
      READ(5,*)A,B
      WRITE(6,500)A
      WRITE(6,503)B
503   FORMAT(/,'    B:  ',(24F3.0))
500   FORMAT(/,'    A:  ',(24F3.0))
1001  FORMAT('    NLONG =',I3,'      NLAT =',I3,/)
C  CONSTRUCT FACTORS, GA1, GB1, SA1, AND SA2 OF STIFFNESS
C  MATRIX
      CALL AMTRX4
      WRITE(6,501)AG
501   FORMAT(/,'    AG:  ',(12F4.1))
      WRITE(6,504)BG
504   FORMAT(/,'    BG:  ',(12F4.1))
      WRITE(6,1002)GA1
1002  FORMAT(/,'  GA1',/,(3X,6F7.3))
      WRITE(6,1012)SA1
1012  FORMAT(/,'  SA1',/,(3X,6F7.3))
      WRITE(6,1004)GB1
1004  FORMAT(//,'  GB1',/,(3X,6F7.3))
      WRITE(6,1014)SB1
1014  FORMAT(//,'  SB1',/,(3X,6F7.3))
C  LOAD BORDER VECTOR W1
      READ(5,*)V
      CALL BORDER(W1,V)
      WRITE(6,510)V
      L1=4*NLONG
      L2=L1+1
      L3=L1+4*LATX
      WRITE(6,520)(W1(L),L=1,L1)
      WRITE(6,521)(W1(L),L=L2,L3)
520   FORMAT(//,'  W1',/,(3X,6F8.2))
521   FORMAT(3X,5F8.2)
510   FORMAT(//,'  V:',/,(4X,6F10.5))
      CALL MULT1(W1,V,GA1,SB1)
C     WRITE(6,520)(W1(L),L=1,L1)
C     WRITE(6,521)(W1(L),L=L2,L3)
C     WRITE(6,510)V
      CALL BORDER(W1,V)
C     WRITE(6,520)(W1(L),L=1,L1)
C     WRITE(6,521)(W1(L),L=L2,L3)
      CALL MULT1(W1,V,SA1,GB1)
      WRITE(6,510)V
      WRITE(6,520)(W1(L),L=1,L1)
      WRITE(6,521)(W1(L),L=L2,L3)
      READ(5,*)P
      WRITE(6,530)P
530   FORMAT(/,'  P:  ',/,(6X,3F12.4))
```

```
      READ(5,*)D
      WRITE(6,531)D
531   FORMAT(/,'  D:   ',3F12.4)
C
C  FORM U = VINT*P
C
      LONGM=NLONG-1
      LONGMM=NLONG-2
      NLATM=NLAT-1
      DO 30 L=1,NLATM
      JP1=(L-1)*NLATM
      KU1=(L-1)*(NLONG-2)-1
      DO 29 K=2,LONGM
      TEMP=0.
      DO 28 J=1,NLATM
      JV=J*NLONG+K
      JP=JP1+J
28    TEMP=TEMP+V(JV)*P(JP)
29    U(KU1+K)=TEMP
30    CONTINUE
      WRITE(6,532)U
532   FORMAT(/,'  U:   ',/,(6X,4F12.4))
      CALL FFFDB(U,D,GA1,SA1)
      WRITE(6,532)U
C
C  PUT FINAL RESULT IN V
C
      DO 40 L=1,NLATM
      DO 39 K=1,LONGMM
      TEMP=0.
      DO 38 J=1,NLATM
38    TEMP=TEMP+P((J-1)*NLATM+L)*U((J-1)*LONGMM+K)
39    V(L*NLONG+K+1)=TEMP
40    CONTINUE
      WRITE(6,510)V
      STOP
      END
C
C*********************************************************************
      SUBROUTINE BORDER(W1,V)
C*********************************************************************
C
C  THIS SUBROUTINE CLEARS THE BORDER VECTOR W1 AND
C  SUBSTITUTES THE BOUNDARY VALUES FROM V.
C
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON/CM1A/NLAT,NLONG
      DIMENSION W1(ZQ),V(ZP)
      LATX=NLAT+1
      NC=4*NLONG
      NR=4*LATX
      NB=NC+NR
      DO 4 J=1,NB
4     W1(J)=0.
      DO 6 J=1,NLONG
6     W1(J)=V(J)
      DO 8 J=1,NLONG
      L=3*NLONG+J
      K=NLAT*NLONG+J
8     W1(L)=V(K)
      DO 10 J=1,LATX
      L=NC+J
      K=(J-1)*NLONG+1
10    W1(L)=V(K)
      DO 12 J=1,LATX
      L=NC+3*LATX+J
      K=J*NLONG
12    W1(L)=V(K)
      RETURN
      END
C ********************************************************************
      SUBROUTINE AMTRX4
C ********************************************************************
C  THIS SUBROUTINE FORMS THE MATRICES GA1, GB1, SA1, AND SB1
C  THAT ARE FACTORS IN THE TENSOR PRODUCTS USED TO FORM THE
C  COEFFICIENT MATRIX ("STIFFNESS" MATRIX) FOR THE POISSON
```

```
C   EQUATION.  ALL OF THESE MATRICES ARE SYMMETRIC AND
C   TRIDIAGONAL.
C
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON/CM1A/NLAT,NLONG
      COMMON/CM8/A(Z1),B(Z1)
      COMMON AG(ZB),BG(ZC),GA1(ZK),SA1(ZK),GB1(ZL),SB1(ZL)
C     DIMENSION BG(NLAT),AG(NLONG),GB1(2*NLAT-1),
C    1GA1(3*NLONG-3)
C
      LATX=NLAT+1
      LONGM=NLONG-1
C    FIND BG = (ELEMENT HEIGHT)/6.
      NM=NLONG-1
      DO 2 J=1,NLAT
2     BG(J)=B(1+NM*(J-1))/3.
C    GENERATE GB1 AND 6*SB1
      GB1(1)=2.*BG(1)
      SB1(1)=1./BG(1)
      DO 4 J=2,NLAT
      K=2*(J-1)
      GB1(K)=2.*(BG(J-1)+BG(J))
      GB1(K+1)=BG(J-1)
      SB1(K)=1./BG(J-1)+1./BG(J)
4     SB1(K+1)=-1./BG(J-1)
      GB1(2*NLAT)=2.*BG(NLAT)
      GB1(2*NLAT+1)=BG(NLAT)
      SB1(2*NLAT)=1./BG(NLAT)
      SB1(2*NLAT+1)=-1./BG(NLAT)
      J2=2*NLAT+1
      DO 6 J=1,J2
6     SB1(J)=SB1(J)/6.
C    FIND AG = (ELEMENT WIDTH)/6.
      DO 10 J=1,LONGM
10    AG(J)=A(J)/3.
C    GENERATE GA1 AND 6*SA1
      GA1(1)=2.*AG(1)
      SA1(1)=1./AG(1)
      DO 12 J=2,LONGM
      K=2*(J-1)
      GA1(K)=2.*(AG(J-1)+AG(J))
      GA1(K+1)=AG(J-1)
      SA1(K)=1./AG(J-1)+1./AG(J)
12    SA1(K+1)=-1./AG(J-1)
      GA1(2*LONGM)=2*AG(LONGM)
      GA1(2*LONGM+1)=AG(LONGM)
      SA1(2*LONGM)=1./AG(LONGM)
      SA1(2*LONGM+1)=-SA1(2*LONGM)
      J2=2*NLONG-1
      DO 14 J=1,J2
14    SA1(J)=SA1(J)/6.
      RETURN
      END
C
C*****************************************************************
      SUBROUTINE MULT1(W1,V,A,B)
C
C   SUBROUTINE PREMULTIPLIES W1 MATRIX BY TRUNCATED A MATRIX
C   (FIRST AND LAST ROWS OMITTED), POSTMULTIPLIES PRODUCT BY
C   TRUNCATED B MATRIX (FIRST AND LAST COLUMNS OMITTED), AND
C   SUBTRACTS INTERIOR ELEMENTS OF W1 FROM CORRESPONDING
C   ELEMENTS OF V.
C*****************************************************************
C
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON/CM1A/NLAT,NLONG
      DIMENSION W1(ZQ),V(ZP),A(1),B(1)
      LATX=NLAT+1
      LONGM=NLONG-1
      FCU=W1(1)
      RCU=W1(3*NLONG+1)
      L1=4*NLONG
      L2=L1+1
      L3=L1+4*LATX
      DO 2 J=2,LONGM
      K=2*(J-1)
```

```
      L=3*NLONG+J
      FCC=A(K+1)*FCU+A(K)*W1(J)+A(K+3)*W1(J+1)
      RCC=A(K+1)*RCU+A(K)*W1(L)+A(K+3)*W1(L+1)
      FCU=W1(J)
      RCU=W1(L)
      W1(J)=FCC
2     W1(L)=RCC
      LASTA=2*NLONG-1
      DO 4 J=2,NLAT
      L=4*NLONG+J
      LL=L+LATX
      K=L+3*LATX
      KK=K-LATX
      W1(LL)=A(3)*W1(L)
4     W1(KK)=A(LASTA)*W1(K)
C     WRITE(6,520)(W1(L),L=1,L1)
C     WRITE(6,521)(W1(L),L=L2,L3)
520   FORMAT(//,'  INTERMEDIATE RESULT, W1',/,(3X,5F8.2))
521   FORMAT(3X,6F8.2)
      LASTB=2*LATX-1
      NC=4*NLONG
      W1(NLONG+2)=B(3)*W1(2)+B(2)*W1(NC+LATX+2)+B(5)
     1*W1(NC+LATX+3)
      W1(2*NLONG-1)=B(3)*W1(NLONG-1)+B(2)*W1(NC+2*LATX+2)
     1+B(5)*W1(NC+2*LATX+3)
      W1(2*NLONG+2)=B(LASTB-2)*W1(NC+2*LATX-2)+B(LASTB-3)
     1*W1(NC+2*LATX-1)+B(LASTB)*W1(3*NLONG+2)
      W1(3*NLONG-1)=B(LASTB-2)*W1(NC+3*LATX-2)+B(LASTB-3)
     1*W1(NC+3*LATX-1)+B(LASTB)*W1(NC-1)
      J2=NLONG-2
      DO 6 J=3,J2
      W1(J+NLONG)=B(3)*W1(J)
6     W1(J+2*NLONG)=B(LASTB)*W1(J+3*NLONG)
      URL=W1(NC+LATX+2)
      BRL=W1(NC+2*LATX+2)
      J2=LATX-2
      DO 8 J=3,J2
      ND=2*(J-1)
      NCU=NC+LATX+J
      URC=B(ND+1)*URL+B(ND)*W1(NCU)+B(ND+3)*W1(NCU+1)
      BRC=B(ND+1)*BRL+B(ND)*W1(NCU+LATX)+B(ND+3)
     1*W1(NCU+LATX+1)
      URL=W1(NCU)
      BRL=W1(NCU+LATX)
      W1(NCU)=URC
8     W1(NCU+LATX)=BRC
      W1(NC+LATX+2)=W1(NLONG+2)
      W1(NC+2*LATX-1)=W1(2*NLONG+2)
      W1(NC+2*LATX+2)=W1(2*NLONG-1)
      W1(NC+3*LATX-1)=W1(3*NLONG-1)
C
C   CORRECT V
C
      NLATM=NLAT-1
      DO 10 J=3,NLATM
      L=NC+LATX+J
      K=(J-1)*NLONG+2
      V(K)=V(K)-W1(L)
      L=LATX+L
      K=J*NLONG-1
10    V(K)=V(K)-W1(L)
      J2=NLONG-1
      DO 12 J=2,J2
      L=NLONG+J
      V(L)=V(L)-W1(L)
      L=2*NLONG+J
      K=(NLAT-1)*NLONG+J
12    V(K)=V(K)-W1(L)
      RETURN
      END
C************************************************************************
      SUBROUTINE FFFDB(X,E,GA,SA)
C
C   THIS SUBROUTINE SOLVES A SUCCESSION OF ONE-DIMENSIONSAL
C   PROBLEMS.  THE RELEVANT COEFFICIENT MATRIX C IS FIRST
C   FORMED, THEN FACTORED, FOLLOWED BY FORWARD REDUCTION,
```

```
C  DIVISION BY THE DIAGONAL ELEMENTS, AND BACK SUBSTITUTION.
C  THE PROCESS IS CARRIED OUT NLATM TIMES.
C
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON/CM1A/NLAT,NLONG
      DIMENSION X(1),E(1),GA(1),SA(1),C(ZU)
      NLATM=NLAT-1
      LONGM=NLONG-1
      LONGMM=NLONG-2
      DO 50 L=1,NLATM
C
C  FORM COEFFICIENT MATRIX C
C
      D1=E(L)
      C(1)=SA(2)+D1*GA(2)
      J2=2*NLONG-5
      DO 2 J=2,J2
2     C(J)=SA(J+2)+D1*GA(J+2)
C
C  FACTOR C
C
      TEMP=C(3)/C(1)
      C(2)=C(2)-TEMP*C(3)
      IF(C(2))7,7,3
3     C(3)=TEMP
      J2=LONGMM-1
      DO 5 J=2,J2
      TEMP=C(2*J+1)/C(2*(J-1))
      C(2*J)=C(2*J)-TEMP*C(2*J+1)
      IF(C(2*J))7,7,5
5     C(2*J+1)=TEMP
      GO TO 8
7     WRITE(6,1000)J,C(2*J)
1000  FORMAT(//,' STOP - MATRIX NOT POSITIVE DEFINITE',//,
     1' NONPOSITIVE PIVOT FOR EQUATION  ',I3,//,' PIVOT = '',
     2D20.12)
      STOP
C
C  PERFORM FORWARD REDUCTION
C
8     J2=(L-1)*LONGMM
      DO 10 J=2,LONGMM
10    X(J2+J)=X(J2+J)-X(J2+J-1)*C(2*(J-1)+1)
C
C  DIVIDE BY DIAGONAL ELEMENTS
C
      X(J2+1)=X(J2+1)/C(1)
      DO 12 J=2,LONGMM
12    X(J2+J)=X(J2+J)/C(2*(J-1))
C
C  BACK-SUBSTITUTE
C
      DO 14 J=2,LONGMM
      JB=J2+LONGM-J
14    X(JB)=X(JB)-X(JB+1)*C(2*(LONGM-J)+1)
50    CONTINUE
      RETURN
      END
```

INITIAL DISTRIBUTION LIST

| | | Copies |
|---|---|---|
| 1. | Defense Technical Information Center<br>Cameron Station<br>Alexandria, Virginia 22314 | 2 |
| 2. | Research Administration, Code 012A<br>Naval Postgraduate School<br>Monterey, California 93943 | 1 |
| 3. | Professor R. E. Newton, Code 69Ne<br>Naval Postgraduate School<br>Monterey, California 93943 | 10 |
| 4. | Professor R. T. Williams, Code 63Wu<br>Naval Postgraduate School<br>Monterey, California 93943 | 5 |
| 5. | Professor A. L. Schoenstadt, Code 53Zh<br>Naval Postgraduate School<br>Monterey, California 93943 | 1 |
| 6. | Professor D. Salinas, Code 69Zc<br>Naval Postgraduate School<br>Monterey, California 93943 | 1 |
| 7. | Professor R. H. Franke, Code 53Fe<br>Naval Postgraduate School<br>Monterey, California 93943 | 1 |
| 8. | Superintendent<br>Naval Postgraduate School<br>Monterey, California 93943<br>ATTN Code 0142 Library | 2 |
| 9. | Commanding Officer<br>Naval Environmental Prediction Research Facility<br>Monterey, California 93943 | 10 |
| 10. | Doctor A. N. Staniforth<br>Recherche en Prevision Numerique<br>Atmospheric Environment Service<br>Dorval, Quebec H9P 1J3<br>CANADA | 1 |
| 11. | Library, Code 0142<br>Naval Postgraduate School<br>Monterey, CA 93943 | 1 |