DEPARTMENT OF MATHEMATICS

GAUSSIAN ELIMINATION WITH PIVOTING FOR MULTI–DIAGONAL SYSTEMS

C. I. Christov

Numerical Analysis Report 5/94

UNIVERSITY OF READING

GAUSSIAN ELIMINATION WITH PIVOTING FOR MULTI–DIAGONAL SYSTEMS

C. I. Christov

Instituto Pluridisciplinar, Universidad Complutense,
Paseo Juan XXIII, No 1, 28040, Madrid, Spain

Numerical Analysis Report 5/94

Visiting the Institute for Computational Fluid Dynamics

On leave from the Institute of Meteorology and Hydrology, Bulgarian Academy of Sciences, Sofia 1184, BULGARIA.

1 Introduction

The sparse systems of linear algebraic equations appear in many different fields. For this reason devising algorithms for their solution enjoys a considerable attention and a few ingenious algorithms have been proposed. The real problem arise, however, when the size of system becomes very large. The systems that appear in finite-difference or finite-element approximations of differential equations are particularly stupendous. Fortunately, they are very simple and have multidiagonal matrices. Acknowledging the last property one can create specialized fast algorithms for direct solution. The purpose of the present note is to present such an algorithm.

2 Algorithm

Consider linear systems of the type

$$
\begin{aligned}
b_1^{M_1+1} u_1 + \ldots + b_1^{M_1+M_2+1} u_{M_2+1} &= b_1^{M_1+M_2+2} \\
b_2^{M_1} u_1 + b_2^{M_1+1} u_2 + \ldots + b_2^{M_1+M_2+1} u_{M_2+2} &= b_2^{M_1+M_2+2} \\
\ldots \qquad \ldots \quad & \quad \ldots \\
b_{M_1+1}^1 u_1 + b_{M_1+1}^2 u_2 + \ldots + b_{M_1+1}^{M_1+M_2+1} u_{M_1+M_2+1} &= b_{M_1+1}^{M_1+M_2+2} \\
\ldots \qquad \ldots \quad & \quad \ldots \\
b_i^1 u_{i-M_1} + b_i^2 u_{i-M_1+1} + \ldots + b_i^{M_1+1} u_i + \ldots + b_i^{M_1+M_2+1} u_{i+M_2} &= b_i^{M_1+M_2+2}, \\
\ldots \qquad \ldots \quad & \quad \ldots \\
b_{N-M_2-1}^1 u_{N-M_1-M_2} + b_{N-M_1-M_2+1}^2 u_{N-M_1-M_2+1} + \ldots + b_{N-M_2-1}^{M_1+1} u_{N-M_2} & \\
+ \ldots + b_{N-M_2-1}^{M_1+M_2-1} u_N &= b_{N-M_2-1}^{M_1+M_2+2} \\
\ldots \qquad \ldots \quad & \quad \ldots \\
b_{N-M_2}^1 u_{N-M_1-M_2+1} + \ldots + b_{N-M_2}^{M_1+M_2} u_N &= b_{N-M_2-1}^{M_1+M_2+2} \\
b_{N-M_2+1}^1 u_{N-M_1-M_2+2} + \ldots + b_{N-M_2+1}^{M_1+M_2+1} u_N &= b_{N-M_2+1}^{M_1+M_2+2} \\
\ldots \qquad \ldots \quad & \quad \ldots \\
b_N^1 u_{N-M_1} + \ldots + b_N^{M_1+1} u_N &= b_N^{M_1+M_2+2}
\end{aligned}
\tag{1}
$$

System (1) is a multidiagonal one with M_2 diagonals above the main diagonal and M_1 – bellow it. We concern ourselves with the non-trivial case when $M_1 \neq 0$ and $M_2 \neq 0$. Otherwise one would have the simplest case of upper- or lower-diagonal matrix, respectively.

When $M_1 = M_2 = 1$, system (1) reduces to the well known *three*-diagonal system for which there exist a very simple elimination procedure (called sometimes Thomas algorithm [1]). The said procedure is implemented in two steps: forward and backward "sweeps". It is called in the Russian-language literature "*progonka*" [2] (not to be confused with "*perestroika*"). It is well known that the *three*-diagonal *progonka* is stable to round-off errors only iff the main diagonal is dominating. When it is not the case the algorithm can be generalized to incorporate pivoting (see, e.g., [1,2]). Then it is called non-monotonous *progonka*. In [2] is shown the way of generalization of the algorithm of non-monotonous *progonka* to the case of *five*-diagonal systems. In the present short note we show the generalization of the algorithm with pivoting to the case of M_1 sub-diagonals and M_2 super-diagonals, where M_1, M_2 are arbitrary numbers $(M_1 + M_2 < N - 1)$.

Generalizing the notations used in [2] for the *five*-diagonal case, we recast the original system in the form

$$\begin{aligned}
a_{11}u_{\theta^1_{i-1}} + a_{12}u_{\theta^2_{i-1}} + \ldots + a_{1,M_2}u_{\theta^{M_2}_{i-1}} + b_i^{M_1+M_2+1}, u_{i+M_2} &= \varphi_1, \\
\ldots \qquad \ldots \qquad &\ldots \\
a_{M_1+M_2,1}u_{\theta^1_{i-1}} + a_{M_1+M_2,2}u_{\theta^2_{i-1}} + \ldots + b^2_{i+M_1+M_2-1}u_{i+M_2} &+ \\
+ \ldots + b^{M_1+M_2+1}_{i+M_1+M_2-1}u_{i+M_1+2M_2} &= \varphi_{M_1+M_2} \\
b^1_{i+M_1+M_2}u_{i+M_2} + \qquad + b^{M_1+M_2+1}_{i+M_1+M_2}u_{i+M_1+2M_2} &= b^{M_1+M_2+2}_{i+M_1+M_2} \\
\ldots \qquad \ldots \qquad &\ldots
\end{aligned} \tag{2}$$

It is seen that for $i = 1$ one can render the first $M_1 + M_2$ equations of (1) into the first $M_1 + M_2$ equations of (2). upon setting

$$\theta^j_0 = j \ , \quad j = 1, 2, \ldots, M_2 \tag{3}$$

The forward "sweep" of the elimination procedure consists in resolving the first of the equations (2) for one of the unknowns. In most general form this gives

$$u_{\theta_i^{M_2+1}} = \alpha_i^1 u_{\theta_i^1} + \ldots + \alpha_i^{M_2} u_{\theta_i^{M_2}} + \beta_i, \tag{4}$$

where indices $\theta_i^1, \ldots, \theta_i^{M_2}, \theta_i^{M_2+1}$, take values from the set of numbers $\theta^1_{i-1}, \ldots, \theta^{M_2}_{i-1}, i$. The essence of the pivoting is to chose the new indices (subscript i) in a manner so as to have all of the coefficients $| \alpha_i^j | \leq 1, \quad j = 1, 2, ..., M_2$. In other words, when resolving the first of equations (2) we find the maximal coefficient. If the maximal coefficients turn out to be more than one, then we take as maximal that one which is situated on the left of the others (i.e., with the smaller value of the superscript).

We distinguish two main cases:

- I. The maximal coefficient is one of the coefficients a_{1,j_2}, say $a_{1,k}$;
- II. The maximal coefficient is $b_i^{M_1+M_2+1}$.

For the case I the forward-sweep coefficients α of (4) and the "new" values for their indices θ are calculated according to the formulas

$$\begin{aligned}
\alpha_i^{j_2} &= -\frac{a_{1,j_2}}{a_{1,k}} \ , \qquad 1 \leq j_2 < k \\
\alpha_i^{j_2} &= -\frac{a_{1,j_2+1}}{a_{1,k}}, \qquad k \leq j_2 \leq M_2 - 1 \\
\alpha_i^{M_2} &= -\frac{b_i^{M_1+M_2+1}}{a_{1,k}}, \qquad \beta_i = \frac{\varphi_1}{a_{1,k}} \ .
\end{aligned} \tag{5}$$

$$\theta_i^{M_2+1} = \theta_{i-1}^k , \quad \theta_i^{M_2} = i + M_2 \quad \text{for} \quad k \le M_2$$
$$\theta_i^j = \theta_{i-1}^j \quad \text{for} \quad 1 \le j < k; \tag{6}$$
$$\theta_i^j = \theta_{i-1}^{j+1} \quad \text{for} \quad k \le j \le M_2 - 1 .$$

and it is obvious that $| \alpha_i^j | \le 1$.

For $j_1 \le M_1 + M_2 - 1$ the "new" values of coefficients a_{j_1,j_2} and φ_{j_1} are calculated in this case as follows

$$a_{j_1,j_2} = a_{j_1+1,j_2} + \alpha_i^{j_2} a_{j_1+1,k} \quad 1 \le j_2 < k;$$
$$a_{j_1,j_2} = a_{j_1+1,j_2+1} + \alpha_i^{j_2} , a_{j_1+1,k} \quad k \le j_2 < M_2 - 1; \tag{7}$$
$$a_{j_1,M_2} = b_{i+j_1}^{M_1+M_2+1-j_1} + \alpha_i^{M_2} a_{j_1+1,k}$$
$$\varphi_{j_1} = \varphi_{j_1+1} - \beta_i a_{j_1+1,k} \quad j_1 < M_1 + M_2 .$$

The last row for the "new" version of system (2) is supplied by the respective row of the system (1), namely

$$a_{M_1+M_2,j_2} = 0 , \quad 1 \le j_2 \le M_2 - 1 ,$$
$$a_{M_1+M_2,M_2} = b_{i+M_1+M_2}^1 , \quad \varphi_{M_1+M_2} = b_{i+M_1+M_2}^{M_1+M_2+2} . \tag{8}$$

For the case II we have for $1 \le j_2 \le M_2$ and then

$$\alpha_i^{j_2} = -\frac{a_{1,j_2}}{b_i^{M_1+M_2+1}} , \quad \beta_i = \frac{\varphi_1}{b_i^{M_1+M_2+1}} ,$$
$$\theta_i^{M_2+1} = i + M_2 , \quad \theta_i^j = \theta_{i-1}^j \tag{9}$$

and the "new" coefficients for $1 \le j_1 \le M_1 + M_2 - 1$ are given by

$$a_{j_1,j_2} = a_{j_1,j_2} + \alpha_i^{j_2} , b_{i+j_1}^{M_1+M_2+1-j_1} ,$$
$$\varphi_{j_1} = \varphi_{j_1+1} - \beta_i b_{i+j_1}^{M_1+M_2+1-j_1} . \tag{10}$$

Respectively, the last row of system (2) gives

$$a_{M_1+M_2,j_2} = b_{i+M_1+M_2}^1 a_i^{j_2}, \quad 1 \le j_2 \le M_2 ; \quad \varphi_{M_1+M_2} = b_{i+M_1+M_2}^{M_1+M_2+2} \beta_i . \tag{11}$$

Thus one step of the forward "sweep" is completed. The forward sweeping is conducted until all of the equations with numbers $i \le N - 1$) are exhausted. Then (4) and the last of the equations (2) with $i = N$ form a closed system for $u_{\theta_N^{M_2}}$ which appears to be exactly equal to β_N. The backward "sweep" goes according to (4). Due to the fact that all α's are less than or equal to one, the backward "sweep" is stable to round-off errors.

The FORTRAN code implementing the above described algorithm is presented in Appendix A.

3 Tests and Comparisons

In order to display the performance of the algorithm we consider here three tests arising from difference approximations to boundary value problems. The first two tests are related to the numerical solution of the differential equation $u''' = 6$, but with two different sets of boundary conditions. The first set consist of one b.c. $u = 0$ at $x = 0$ and two b.c. $u = u' = 0$ at $x = 1$, while the second set consists of two b.c. $u = u' = 0$ at $x = 0$ and one $u = 0$ – at $x = 1$. The analytical solutions for these test problems are $u = x^3 - 2x^2 + x$ and $u = x^3 - x^2$, respectively. The first problem results into four-diagonal linear algebraic system with one sub-diagonal and two super-diagonals. The second problem has two sub-diagonals and one super-diagonal. The FORTRAN codes and the results for the two test are presented in Appendices B1 and B2, respectively.

The third case demonstrates the performance for *nine*-diagonal systems. We consider the eight-order boundary value problem

$$\frac{d^8u}{dx^8} = 1 \ , \quad u = \frac{d^2u}{dx^2} = \frac{d^4u}{dx^4} = \frac{d^6u}{dx^6} = 0 \quad \text{for} \quad x = 0, 1 \ . \tag{12}$$

Eq. 12 has an analytical solution

$$u = \frac{(x^8 - 4x^7 + 14x^5 - 28x^3 + 17x)}{40320}$$

The problem with eq. 12 is that its direct difference approximation on uniform mesh of spacing h

$$u_{i-4} - 8u_{i-3} + 28u_{i-2} - 56u_{i-1} + 70u_i - 56u_{i+1} + 28u_{i+2} - 8u_{i+3} + u_{i+4} = h^8 \tag{13}$$

has a matrix whose determinant is of order of $O(h^8)$ which brings into picture the significance of propagation of the round-off error, which spoils completely the algorithm for total number of grid points larger than 200. For this reason, before constructing the difference scheme we render the original equation 12 into a system of four second-order equations, namely

$$\frac{d^2u}{dx^2} = v \quad \frac{d^2v}{dx^2} = w \quad \frac{d^2w}{dx^2} = s \quad \frac{d^2s}{dx^2} = 1 \ . \tag{14}$$

with b.c.

$$u = v = w = s = 0 \quad \text{for} \quad x = 0, 1 \; . \qquad (15)$$

Approximating the second derivatives in eq.14 by central differences one gets four three-point difference equations at each grid point. By means of a new difference function

$$W_{4i-3} \equiv u_i \; , \quad W_{4i-2} \equiv v_i \; , \quad W_{4i-1} \equiv w_i \; , \quad W_{4i} \equiv s_i \; ,$$

one renders the system of nine diagonals. The difference with the direct approximation eq. 13 is that the new system has four times larger number of unknowns – $4N$, where N is the total number of grid points. The solver, however is so efficient that we were able to solve the problem with $N = 20001$, which means 80004 equations without bothering about the round-off error, since the eq. 14 yields to a system with determinant of order of $O(h^2)$.

Appendix C presents the FORTRAN code of the main programme implementing the test and the result obtained for $N = 5001, h = 0.0002, 4N = 20004$. One sees that the differences with the analytical solution are in the seventh significant digit which is compatible with truncation error of order of H^2. For the sake of demonstrating the role of the round-off error, we present in the last column the eight derivative as approximated by the nine-point difference approximation eq. 13. One sees that the double precision is far from being enough to provide for accurate numerical differentiation of eight order. That is an explanation why the direct difference approximation fails to provide solution for $h > 0.01$ when the determinant of the matrix is of order of 10^{-16} which is the limit for double precision.

4 Conclusions

The algorithm has been tested for many other difference systems and performed very well. The presented here tests suffice to claim the correctness of the programme code.

Acknowledgment Part of this work was done during the visit of the author to Laboratoire de Modélisation en Mécanique, Université Pierre et Marie Curie, Paris. Author acknowledges also the hospitality of Dept. of Mathematics of University of Reading, UK where part of the tests were performed and the final version of the manuscript was written.

REFERENCES

[1] Roache P., Computational Fluid Dynamics, *Hermosa* , 1972.

[2] Samarskii A. A., Nikolaev E. N., *Numerical Methods for Grid Equations,* Moscow, *Nauka*, 1978 (in Russian). English translation: Birkhäuser, Basel, 1989

Appendix A

Fortran Code of Solver

```
C
C
      SUBROUTINE DPRNUL(B,NAL,A,FI,NDIM,N,NU,NL)
C
C     ------------------------------------------------
C     THIS SUBROUTINE SOLVES A (NU+NL+1)-DIAGONAL LINEAR
C     ALGEBRAIC SYSTEM BY MEANS OF A SPECIAL KIND OF GAUSSIAN
C     ELIMINATION WITH PIVOTING CALLED 'NON-MONOTONOUS PROGONKA'
C     LITERATURE: A.A.Samarsky, E.Nikolaev,  Numerical Methods
C     grid equations, Moskow, "Nauka", 1978 (in Russian),
C     English Translation: Birkh\"auser, Basel, 1989
C
C     The system should be represented in the following form:
C
C     B(I,1)*U(I-NL) + B(I,2)*U(I-NL+1) + . . . . +
C      + B(I,NL)*U(I-1) + B(I,NL+1)*U(I) + B(I,NL+2)*U(I+1)
C      + . . . + B(I,NL+NU)*U(I+NU-1) + B(I,NL+NU+1)*U(I+NU)
C          = B(I,NL+NU+2),                        for I=1,...,N
C
C     where U(I) is the sought set function and B(.,.) is the
C     array of coefficients organized diagonalwise.
C
C     -------------------------------------------------
C       REMARK:
C       It is self-evident that B(1,1)=...=B(1,NL)=0. ,
C      . . . , B(2,1)=...=B(2,NL-1)=0.,    B(NL,1)=0.
C       and
C       B(N,NU+2)=...=B(N,NL+NU+1)=0. ,
C      . . . , B(N-1,NL+3)=...=B(N-1,NL+NU+1)=0.,
C       B(N-NL,NL+NU+1)=0.
C
C       The above mentioned members of array B(.,.) are
C       automatically avoided in computations and it is not
C       necessary to explicitly set them equal to zero
C     -------------------------------------------------
C
C     NDIM is the total number of columns of the arrays B and NAL
C          (as specified in the user's programme).  This number
C          does not necessarily coinside with the actual number
C          number of linear equations to be solved during the
C          particular run of the programme which allows certain
C          flexibility when changing the dimensions of the systems.
```

```
C
C     N is the actual number of algebraic equations (columns of
C          arrays B and NAL) to be solved in the particular run.
C
C     NU is the number of diagonals above the main one
C          (not including the latter)
C
C     NL is the number of diagonals below the main one
C          (not including the latter)
C
C     NL+NU+1  is the total number of diagonals
C
C     REMARK: It is more convenient to arrainge the system in
C          a way so as to have NU lesser than NL
C
C
        IMPLICIT REAL*8(A-H,P-Z)
        DIMENSION B(NDIM,NL+2*NU+4),NAL(NDIM,NU+1)
        DIMENSION A(NU+NL,NU),FI(NU+NL)
C
C     The columns of numbers NL+NU+3 <=J <= NL+2*NU+3 of array
C     B(.,J) are used for the purposes of the elimination. The
C     solution obtained is stored in B(.,NL+2*NU+4).  Respectively,
C     the integer array NAL(.,J), J=1 to J=NU+1 contains the
C     'addresses' of the elements of the array B(.,.) needed as
C     cross-references during the pivoting.
C
C
        N1= N+1
        NDD = NDIM
        IF ((N1).LE.NDD) GO TO 201
        WRITE (*,104) (N1),NDD
104     FORMAT(1X,'THE NUMBER N+1=',I5,' IS GREATER THAN ND=',I5)
        STOP
201     CONTINUE
C
C
        M = NU + 1
        MAIN = NL+1
        MM2= M -2
        M2 = NU+NL
        MT2 = M2+2
        MM1 = NU
        M2P1 = M2+1
        M2M1 = M2-1
        M3 = NL + 2*NU + 4
```

```
C
C
      UNE = 1.D0
      ZERO = 0.D0
C
C
      DO 9 J2 = 1,M2
      DO 90 J1 = 1,MM1
      JAX = J1-J2+MAIN
      IF (JAX.LT.1) GO TO 901
      A(J2,J1) = B(J2,JAX)
      GO TO 90
901   CONTINUE
      A(J2,J1) = ZERO
90    CONTINUE
      FI(J2) = B(J2,MT2)
9     CONTINUE
C
      DO 91 J1=1,MM1
91    NAL(1,J1)=J1
      nal(1,m) = m
C
      DO 1 I=1,N
C
      RMAX=DABS(A(1,1))
      K=1
      DO 10 J1=2,MM1
C
      R1=DABS(A(1,J1))
      IF (RMAX.Gt.R1) GO TO 10
      K=J1
      RMAX=R1
10    CONTINUE
      R1=DABS(B(I,M2+1))
      IF (RMAX.Gt.R1) GO TO 100
      K=M
      RMAX=R1
100   CONTINUE
C
      KP1 = K+1
      KM1 = K-1
C
       if (rmax.eq.zero) WRITE (*,191) I,K
191   FORMAT(1X,'CURRENT POINT ',I5,3X,'ELIMIN. INDEX ',I3)
C
C
```

```
        NALTMP = NAL(I,M)
        NAL(I,M) = I+MM1

        IF (K.EQ.M) GO TO 14
        AUX = UNE/A(1,K)
        DO 11 J1 = 1,MM1
        JACT = J1+1
        IF (J1.LT.K) JACT = J1
        IF (J1.LT.MM1) B(I,MT2+J1) = -A(1,JACT)*AUX
        IF (J1.EQ.MM1) B(I,MT2+MM1) = - B(I,M2+1)*AUX
        NAL(I+1,J1) = NAL(I,JACT)
11      CONTINUE
C
        B(I,MT2+M) = FI(1)*AUX
        NAL(I+1,M) = NAL(I,K)
        NAL(I,M) = NALTMP
C
C
        DO 12 J2 = 1,M2M1
        IF ((I+J2).GT.N) GO TO 13
        DO 120 J1 = 1,MM2
        JADD = 1
        IF (J1.LT.K) JADD = 0
120     A(J2,J1) = A(J2+1,J1+JADD) + B(I,MT2+J1)*A(J2+1,K)
        A(J2,MM1) = B(I+J2,M2+1-J2) + B(I,MT2+MM1)*A(J2+1,K)
        FI(J2) = FI(J2+1) - B(I,MT2+M)*A(J2+1,K)
12      CONTINUE
        IF ((I+M2).GT.N) GO TO 13
        DO 123 J1=1,MM2
123     A(M2,J1) = ZERO
        A(M2,MM1) = B(I+M2,1)
        FI(M2) = B(I+M2,MT2)
C
13      CONTINUE
        GO TO 15
C
14      CONTINUE
        AUX = UNE/B(I,M2+1)
        DO 141 J1 = 1,MM1
        B(I,MT2+J1) = -A(1,J1)*AUX
        NAL(I+1,J1) = NAL(I,J1)
141     CONTINUE
C
        B(I,MT2+M) = FI(1)*AUX
        NAL(I+1,M) = NAL(I,M)
        NAL(I,M) = NALTMP
```

```
C
C       Backward 'sweep'
C
        DO 142 J2 = 1,M2M1
        IF ((I+J2).GT.N) GO TO 143
        DO 1420 J1 = 1,MM1
        JADD = 1
        IF (J1.LT.K) JADD = 0
1420    A(J2,J1) = A(J2+1,J1) + B(I,MT2+J1)*B(I+J2,M2-J2+1)
        FI(J2) = FI(J2+1) - B(I,MT2+M)*B(I+J2,M2-J2+1)
142      CONTINUE
        IF ((I+M2).GT.N) GO TO 143
        DO 1423 J1=1,MM1
1423     A(M2,J1) = B(I+M2,1)*B(I,MT2+J1)
        FI(M2) = B(I+M2,MT2) - B(I+M2,1)*B(I,MT2+M)
C
143      CONTINUE
C
15      CONTINUE
C
1       CONTINUE
C
C
        DO 2 I1=1,N
        I=N-I1+1
        B(NAL(I+1,M),M3) = B(I,MT2+M)
        DO 20 J1=1,MM1
        IF (NAL(I+1,J1).LE.N) B(NAL(I+1,M),M3)
     *      = B(NAL(I+1,M),M3) + B(I,MT2+J1)*B(NAL(I+1,J1),M3)
20      CONTINUE
2       CONTINUE
        RETURN
        END
```

Fortran Code of First Test with Four Diagonals (Two Diagonals above the Main One)

```
      PARAMETER(ND=20001)
      IMPLICIT REAL*8(A-H,P-Z)
      DIMENSION B(ND,9),NAL(ND,3),A(3,2),FI(3)
C
      N=5001
      NP1=N+1
      N1=N-1
      HX=1.D0/FLOAT(N-1)
      HX3=HX**3
      NPRINT = (N-2)/50 +1
C
      OPEN(3,FILE='test4a.dat')
C
C-------------------------------------------------------
C     Solving a boundary value problem represented
C     on the difference level by a four-diagonal matrix:
C     u''' = 1;  u(0) = 0,  u(1)=u'(1)=0 .
C     whose exact solution is:
C     u=(X**3 - X**2)
C-------------------------------------------------------
C
      FI0=0.
      FI1=0.
      PSI0=0.
      PSI1=0.
C
C     Boundary conditions at the left-hand end of
C     interval x=0 (u=FI0):
      B(1,2)=1
      B(1,5)=FI0
      DO 1 J=2,N1
      B(J,1)=-1
      B(J,2)= 3
      B(J,3)=-3
      B(J,4)= 1
      B(J,5)= 6*HX3
C
1     CONTINUE
C
C     Boundary conditions at the right-hand end of
C     interval x=1 (u'=PSI1) and (u=PSI0):
```

```
      B(N,1) = -1
      B(N,3) = 1
      B(N,5) = PSI1
      B(NP1,1) = 1
      B(NP1,5) = PSI0
C
      CALL DPRNUL(B,NAL,A,FI,ND,NP1,2,1)
C
      WRITE (3,100)
      XIN=0
      DO 2 J=1,N
      XACT=XIN+(J-1)*HX
      B(J,7) = XACT
      UAN=(XACT**3 - 2*XACT**2 + XACT)
      B(J,8)=UAN
2     CONTINUE
C
      DO 3 J=2,N1
      B(J,5)= (-B(J-1,9) + 3*B(J,9) - 3*B(J+1,9)
     *         + B(J+2,9))/HX3
3     CONTINUE
C
      DO 4 J1 = 1,N,NPRINT
      IF ((J1.EQ.1).OR.(J1.EQ.N)) THEN
      WRITE(3,102) B(J1,7),B(J1,9),B(J1,8)
      ELSE
      WRITE(3,101) B(J1,7),B(J1,9),B(J1,8),B(J1,5)
      ENDIF
4     CONTINUE
C
100   FORMAT(20X,'COMPUTED   ',6X,'ANALYTICAL ',2X,
     *          'THIRD DERIVATIVE')
101   FORMAT(2X,E10.3,3X,D17.10,2X,D17.10,3X,E11.4)
102   FORMAT(2X,E10.3,3X,D17.10,2X,D17.10)
103   FORMAT(I7)
      CLOSE(3)
      STOP
      END
```

```
                    COMPUTED          ANALYTICAL    THIRD DERIVATIVE
  0.000E+00   0.0000000000D+00   0.0000000000D+00
  0.200E-01   0.1920800084D-01   0.1920800000D-01    0.6000E+01
  0.400E-01   0.3686400164D-01   0.3686400000D-01    0.6000E+01
  0.600E-01   0.5301600239D-01   0.5301600000D-01    0.6000E+01
  0.800E-01   0.6771200311D-01   0.6771200000D-01    0.6000E+01
```

0.100E+00	0.8100000378D-01	0.8100000000D-01	0.6000E+01
0.120E+00	0.9292800442D-01	0.9292800000D-01	0.6000E+01
0.140E+00	0.1035440050D+00	0.1035440000D+00	0.6000E+01
0.160E+00	0.1128960056D+00	0.1128960000D+00	0.6000E+01
0.180E+00	0.1210320061D+00	0.1210320000D+00	0.6000E+01
0.200E+00	0.1280000066D+00	0.1280000000D+00	0.6000E+01
0.220E+00	0.1338480070D+00	0.1338480000D+00	0.6000E+01
0.240E+00	0.1386240074D+00	0.1386240000D+00	0.6000E+01
0.260E+00	0.1423760078D+00	0.1423760000D+00	0.6000E+01
0.280E+00	0.1451520082D+00	0.1451520000D+00	0.6000E+01
0.300E+00	0.1470000085D+00	0.1470000000D+00	0.6000E+01
0.320E+00	0.1479680088D+00	0.1479680000D+00	0.6000E+01
0.340E+00	0.1481040091D+00	0.1481040000D+00	0.6000E+01
0.360E+00	0.1474560093D+00	0.1474560000D+00	0.6000E+01
0.380E+00	0.1460720095D+00	0.1460720000D+00	0.6000E+01
0.400E+00	0.1440000097D+00	0.1440000000D+00	0.6000E+01
0.420E+00	0.1412880098D+00	0.1412880000D+00	0.6000E+01
0.440E+00	0.1379840099D+00	0.1379840000D+00	0.6000E+01
0.460E+00	0.1341360100D+00	0.1341360000D+00	0.6000E+01
0.480E+00	0.1297920100D+00	0.1297920000D+00	0.6000E+01
0.500E+00	0.1250000100D+00	0.1250000000D+00	0.6000E+01
0.520E+00	0.1198080100D+00	0.1198080000D+00	0.6000E+01
0.540E+00	0.1142640099D+00	0.1142640000D+00	0.6000E+01
0.560E+00	0.1084160098D+00	0.1084160000D+00	0.6000E+01
0.580E+00	0.1023120097D+00	0.1023120000D+00	0.6000E+01
0.600E+00	0.9600000956D-01	0.9600000000D-01	0.6000E+01
0.620E+00	0.8952800938D-01	0.8952800000D-01	0.6000E+01
0.640E+00	0.8294400917D-01	0.8294400000D-01	0.6000E+01
0.660E+00	0.7629600893D-01	0.7629600000D-01	0.6000E+01
0.680E+00	0.6963200866D-01	0.6963200000D-01	0.6000E+01
0.700E+00	0.6300000836D-01	0.6300000000D-01	0.6000E+01
0.720E+00	0.5644800802D-01	0.5644800000D-01	0.6000E+01
0.740E+00	0.5002400766D-01	0.5002400000D-01	0.6000E+01
0.760E+00	0.4377600726D-01	0.4377600000D-01	0.6000E+01
0.780E+00	0.3775200683D-01	0.3775200000D-01	0.6000E+01
0.800E+00	0.3200000637D-01	0.3200000000D-01	0.6000E+01
0.820E+00	0.2656800588D-01	0.2656800000D-01	0.6000E+01
0.840E+00	0.2150400536D-01	0.2150400000D-01	0.6000E+01
0.860E+00	0.1685600480D-01	0.1685600000D-01	0.6000E+01
0.880E+00	0.1267200421D-01	0.1267200000D-01	0.6000E+01
0.900E+00	0.9000003592D-02	0.9000000000D-02	0.6000E+01
0.920E+00	0.5888002939D-02	0.5888000000D-02	0.6000E+01
0.940E+00	0.3384002253D-02	0.3384000000D-02	0.6000E+01
0.960E+00	0.1536001535D-02	0.1536000000D-02	0.6000E+01
0.980E+00	0.3920007837D-03	0.3920000000D-03	0.6000E+01
0.100E+01	0.0000000000D+00	0.0000000000D+00	

Fortran Code of Second Test with Four Diagonals (One Diagonal above the Main One)

```
      PARAMETER(ND=20001)
      IMPLICIT REAL*8(A-H,P-Z)
      DIMENSION B(ND,8),NAL(ND,2),A(3,2),FI(3)
C
      N=5001
      NP1=N+1
      N1=N-1
      HX=1.D0/FLOAT(N-1)
      HX3=HX**3
      NPRINT = (N-2)/50 +1
C
      OPEN(3,FILE='test4b.dat')
C
C-------------------------------------------------------
C     Solving a boundary value problem represented
C     on the difference level by a four-diagonal matrix:
C     u''' = 1;  u(0)=u'(0)=0,  u(1)=0.
C     whose exact solution is:
C     u= (X**3 - 2*X**2 + X)
C-------------------------------------------------------
C
      FI0=0.
      FI1=0.
      PSI0=0.
      PSI1=0.
C
C     Boundary conditions at the left-hand end of
C     interval x=0 (u=FI0, u'=FI1):
      B(1,4) = 1
      B(1,5) = FI0
      B(2,4) = 1
      B(2,2) =-1
      B(2,5) = FI1
C
      DO 1 J=3,N
      B(J,1)=-1
      B(J,2)= 3
      B(J,3)=-3
      B(J,4)= 1
      B(J,5)= 6*HX3
C
```

```
1       CONTINUE
C
C       Boundary conditions at the right-hand end of
C       interval x=1 (u=PSI0):
        B(NP1,3) = 1
        B(NP1,5) = PSI1
C
        CALL DPRNUL(B,NAL,A,FI,ND,NP1,1,2)
C
        WRITE (3,100)
        XIN=0
        DO 2 J=2,NP1
        XACT=XIN+(J-2)*HX
        B(J-1,6) = XACT
        UAN=(XACT**3 - XACT**2)
        B(J-1,7)=UAN
2       CONTINUE
C
        DO 3 J=2,Np1
        B(J-1,5)= (-B(J-2,8) + 3*B(J-1,8) - 3*B(J,8)
     *          + B(J+1,8))/HX3
3       CONTINUE
C
        DO 4 J1 = 1,N,NPRINT
        IF ((J1.EQ.1).OR.(J1.EQ.N)) THEN
        WRITE(3,102) B(J1,6),B(J1+1,8),B(J1,7)
        ELSE
        WRITE(3,101) B(J1,6),B(J1+1,8),B(J1,7),B(J1,5)
        ENDIF
4       CONTINUE
C
100     FORMAT(20X,'COMPUTED   ',6X,'ANALYTICAL ',3X,
     *              'THIRD DERIVATIVE')
101     FORMAT(2X,E10.3,4X,D17.10,2X,D17.10,4X,E11.4)
102     FORMAT(2X,E10.3,4X,D17.10,2X,D17.10)
103     FORMAT(I7)
        CLOSE(3)
        STOP
        END
```

```
                   COMPUTED          ANALYTICAL     THIRD DERIVATIVE
  0.000E+00     0.0000000000D+00   0.0000000000D+00
  0.200E-01    -0.3920007852D-03  -0.3920000000D-03    0.6000E+01
  0.400E-01    -0.1536001541D-02  -0.1536000000D-02    0.6000E+01
  0.600E-01    -0.3384002267D-02  -0.3384000000D-02    0.6000E+01
  0.800E-01    -0.5888002963D-02  -0.5888000000D-02    0.6000E+01
```

0.100E+00	-0.9000003630D-02	-0.9000000000D-02	0.6000E+01
0.120E+00	-0.1267200427D-01	-0.1267200000D-01	0.6000E+01
0.140E+00	-0.1685600487D-01	-0.1685600000D-01	0.6000E+01
0.160E+00	-0.2150400545D-01	-0.2150400000D-01	0.6000E+01
0.180E+00	-0.2656800600D-01	-0.2656800000D-01	0.6000E+01
0.200E+00	-0.3200000652D-01	-0.3200000000D-01	0.6000E+01
0.220E+00	-0.3775200701D-01	-0.3775200000D-01	0.6000E+01
0.240E+00	-0.4377600746D-01	-0.4377600000D-01	0.6000E+01
0.260E+00	-0.5002400789D-01	-0.5002400000D-01	0.6000E+01
0.280E+00	-0.5644800829D-01	-0.5644800000D-01	0.6000E+01
0.300E+00	-0.6300000865D-01	-0.6300000000D-01	0.6000E+01
0.320E+00	-0.6963200899D-01	-0.6963200000D-01	0.6000E+01
0.340E+00	-0.7629600929D-01	-0.7629600000D-01	0.6000E+01
0.360E+00	-0.8294400956D-01	-0.8294400000D-01	0.6000E+01
0.380E+00	-0.8952800981D-01	-0.8952800000D-01	0.6000E+01
0.400E+00	-0.9600001002D-01	-0.9600000000D-01	0.6000E+01
0.420E+00	-0.1023120102D+00	-0.1023120000D+00	0.6000E+01
0.440E+00	-0.1084160103D+00	-0.1084160000D+00	0.6000E+01
0.460E+00	-0.1142640105D+00	-0.1142640000D+00	0.6000E+01
0.480E+00	-0.1198080105D+00	-0.1198080000D+00	0.6000E+01
0.500E+00	-0.1250000106D+00	-0.1250000000D+00	0.6000E+01
0.520E+00	-0.1297920106D+00	-0.1297920000D+00	0.6000E+01
0.540E+00	-0.1341360106D+00	-0.1341360000D+00	0.6000E+01
0.560E+00	-0.1379840106D+00	-0.1379840000D+00	0.6000E+01
0.580E+00	-0.1412880105D+00	-0.1412880000D+00	0.6000E+01
0.600E+00	-0.1440000104D+00	-0.1440000000D+00	0.6000E+01
0.620E+00	-0.1460720103D+00	-0.1460720000D+00	0.6000E+01
0.640E+00	-0.1474560101D+00	-0.1474560000D+00	0.6000E+01
0.660E+00	-0.1481040098D+00	-0.1481040000D+00	0.6000E+01
0.680E+00	-0.1479680096D+00	-0.1479680000D+00	0.6000E+01
0.700E+00	-0.1470000093D+00	-0.1470000000D+00	0.6000E+01
0.720E+00	-0.1451520089D+00	-0.1451520000D+00	0.6000E+01
0.740E+00	-0.1423760086D+00	-0.1423760000D+00	0.6000E+01
0.760E+00	-0.1386240081D+00	-0.1386240000D+00	0.6000E+01
0.780E+00	-0.1338480077D+00	-0.1338480000D+00	0.6000E+01
0.800E+00	-0.1280000072D+00	-0.1280000000D+00	0.6000E+01
0.820E+00	-0.1210320066D+00	-0.1210320000D+00	0.6000E+01
0.840E+00	-0.1128960060D+00	-0.1128960000D+00	0.6000E+01
0.860E+00	-0.1035440054D+00	-0.1035440000D+00	0.6000E+01
0.880E+00	-0.9292800477D-01	-0.9292800000D-01	0.6000E+01
0.900E+00	-0.8100000407D-01	-0.8100000000D-01	0.6000E+01
0.920E+00	-0.6771200334D-01	-0.6771200000D-01	0.6000E+01
0.940E+00	-0.5301600256D-01	-0.5301600000D-01	0.6000E+01
0.960E+00	-0.3686400175D-01	-0.3686400000D-01	0.6000E+01
0.980E+00	-0.1920800089D-01	-0.1920800000D-01	0.6000E+01
0.100E+01	0.0000000000D+00	0.0000000000D+00	

Fortran Code of Second Test with Nine Diagonals (Four Diagonals bellow and above the Main One)

```
      PARAMETER(NDS=5001)
      PARAMETER(ND=NDS*4+1)
      IMPLICIT REAL*8(A-H,P-Z)
      DIMENSION B(ND,16),NAL(ND,5),A(8,4),FI(8)
C
      OPEN(1,FILE='test9.par')
      READ (1,103) M
      READ (1,103) N
      CLOSE(1)
C
      N1=N-1
      N2=N-2
      N3=N-3
      N4=N-4
     OPEN(3,FILE='test9a.dat')
C
      MP1=M+1
      HX=1.D0/FLOAT(N-1)
C
      HX2=HX**2
      HX8 = HX**8
C
C     SOLVING A PROBLEM WITH NINE-DIAGONAL MATRIX
C
C     u'' = v
C     v'' = w
C     w'' = s
C     s'' = 1
C
C     u(0)=v(0)=w(0)=s(0)=0,  u(1)=v(1)=w(1)=s(1)=0.
C
C     which is equivalent to u'''''''' = 1.
C
C     the exact solution is:
C
C     u=(X**8 - 4*X**7 + 14*X**5 - 28*X**3 + 17*X)/40320
C
      FI0=0.
      FI1=0.
      FI2=0.
      FI3=0.
```

```
        PSI0=0.
        PSI1=0.
        PSI2=0.
        PSI3=0.
C
C------Boundary conditions at the left-hand end of interval
C------ x=0:
C
C------u=FI0:
      B(1,5)=1
      B(1,10)=FI0
C
C------v=FI1
      B(2,5)=1
      B(2,10)=FI1
C
C-------w=FI2
      B(3,5)=1
      B(3,10)=FI2
C
C-------s=FI3
      B(4,5)=1
      B(4,10)=FI3
C

      DO 1 J=2,N1
        j3 = J*4 - 3
        j2 = j*4 - 2
        j1 = j*4 - 1
        j0 = j*4
C
C-------This is the portion of programme implmenting u''=v
        B(J3,1)=1
        B(J3,2)=0
        B(J3,3)=0
        B(J3,4)=0
        B(J3,5)=-2
        B(J3,6)=-HX2
        B(J3,7)=0
        B(J3,8)=0
        B(J3,9)=1
        B(J3,10)=0
C
C--------This is the portion of programme implementing v''=w
        B(J2,1)=1
        B(J2,2)=0
```

```
      B(J2,3)=0
      B(J2,4)=0
      B(J2,5)=-2
      B(J2,6)=-HX2
      B(J2,7)=0
      B(J2,8)=0
      B(J2,9)=1
      B(J2,10)=0
C
C--------This is the portion of programme implementing w''=s
      B(J1,1)=1
      B(J1,2)=0
      B(J1,3)=0
      B(J1,4)=0
      B(J1,5)=-2
      B(J1,6)=-HX2
      B(J1,7)=0
      B(J1,8)=0
      B(J1,9)=1
      B(J1,10)=0
C
C--------This is the portion of programme implementing s''=1
      B(J0,1)=1
      B(J0,2)=0
      B(J0,3)=0
      B(J0,4)=0
      B(J0,5)=-2
      B(J0,6)=0
      B(J0,7)=0
      B(J0,8)=0
      B(J0,9)=1
      B(J0,10)=HX2
C
1     CONTINUE
C
C------Boundary conditions at the rright-hand end of interval
C------ x=1:
C
C------u=PSI0:
      B(4*N-3,5)=1
      B(4*N-3,10)=PSI0
C
C------v=PSI1
      B(4*N-2,5)=1
      B(4*N-2,10)=PSI1
C
```

```
C-------w=PSI2
        B(4*N-1,5)=1
        B(4*N-1,10)=PSI2
C
C-------s=PSI3
        B(4*N,5)=1
        B(4*N,10)=PSI3
C
        CALL DPRNUL(B,NAL,A,FI,ND,4*N,4,4)
C
        WRITE (3,100)
        XIN=0
        DO 2 J=1,N
        XACT=XIN+(J-1)*HX
        B(J,13) = XACT
        UAN=(XACT**8-4*XACT**7+14*XACT**5
     *          -28*XACT**3+17*XACT)/40320.
        B(J,15)=UAN
2       CONTINUE
C
        NPRINT = (N-2)/50 +1
C
        DO 3 J1=NPRINT+1,N1,NPRINT
        J=4*J1-3
        JM4 = J-16
        JM3 = J-12
        JM2 = J -8
        JM1 = J -4
        JP1 = J +4
        JP2 = J +8
        JP3 = J+12
        JP4 = J+16
        B(J1,14)=(B(JM4,16)-8*B(JM3,16)+28*B(JM2,16)
     *          -56*B(JM1,16)+70*B(J,16)-56*B(JP1,16)
     *          +28*B(JP2,16)-8*B(JP3,16)+B(JP4,16))/HX8
3       CONTINUE
C
        DO 4 J1 = 1,N,NPRINT
        J= 4*J1 -3
        IF ((J1.LE.4).OR.(J1.GE.(N-3))) THEN
        WRITE(3,102) B(J1,13),B(J,16),B(J1,15)
        ELSE
        WRITE(3,101) B(J1,13),B(J,16),B(J1,15),B(J1,14)
        ENDIF
4       CONTINUE
C
```

```
100    FORMAT(20X,'COMPUTED   ',4X,'ANALYTICAL ',4X,
     *               'EIGHT DERIVATIVE')
101    FORMAT(2X,E10.3,3X,D17.10,2X,D17.10,3X,E11.4)
102    FORMAT(2X,E10.3,3X,D17.10,2X,D17.10)
103    FORMAT(I7)
       CLOSE(3)
      STOP
      END
```

	COMPUTED	ANALYTICAL	EIGHT DERIVATIVE
0.000E+00	0.0000000000D+00	0.0000000000D+00	
0.200E-01	0.8426986071D-05	0.8426985238D-05	-0.7941E+10
0.400E-01	0.1682067212D-04	0.1682067046D-04	-0.7941E+10
0.600E-01	0.2514789126D-04	0.2514788877D-04	-0.1323E+10
0.800E-01	0.3337574221D-04	0.3337573891D-04	-0.7941E+10
0.100E+00	0.4147172062D-04	0.4147171652D-04	-0.3970E+11
0.120E+00	0.4940384850D-04	0.4940384361D-04	-0.6617E+11
0.140E+00	0.5714080140D-04	0.5714079575D-04	-0.1376E+12
0.160E+00	0.6465203264D-04	0.6465202625D-04	0.4765E+11
0.180E+00	0.7190789422D-04	0.7190788711D-04	0.2223E+12
0.200E+00	0.7887975382D-04	0.7887974603D-04	0.2647E+11
0.220E+00	0.8554010761D-04	0.8554009916D-04	0.1323E+12
0.240E+00	0.9186268820D-04	0.9186267912D-04	-0.7412E+11
0.260E+00	0.9782256761D-04	0.9782255795D-04	-0.7941E+11
0.280E+00	0.1033962547D-03	0.1033962445D-03	-0.9529E+11
0.300E+00	0.1085617868D-03	0.1085617760D-03	-0.1059E+12
0.320E+00	0.1132988149D-03	0.1132988037D-03	-0.3176E+11
0.340E+00	0.1175886832D-03	0.1175886716D-03	-0.3176E+11
0.360E+00	0.1214145008D-03	0.1214144889D-03	-0.2329E+12
0.380E+00	0.1247612075D-03	0.1247611952D-03	-0.4235E+11
0.400E+00	0.1276156316D-03	0.1276156190D-03	0.6353E+11
0.420E+00	0.1299665410D-03	0.1299665282D-03	-0.4235E+12
0.440E+00	0.1318046862D-03	0.1318046732D-03	-0.8470E+11
0.460E+00	0.1331228361D-03	0.1331228229D-03	-0.1059E+11
0.480E+00	0.1339158056D-03	0.1339157924D-03	0.0000E+00
0.500E+00	0.1341804758D-03	0.1341804625D-03	-0.7412E+11
0.520E+00	0.1339158056D-03	0.1339157924D-03	-0.5294E+11
0.540E+00	0.1331228361D-03	0.1331228229D-03	0.3176E+11
0.560E+00	0.1318046862D-03	0.1318046732D-03	0.1271E+12
0.580E+00	0.1299665410D-03	0.1299665282D-03	-0.4235E+11
0.600E+00	0.1276156316D-03	0.1276156190D-03	0.0000E+00
0.620E+00	0.1247612075D-03	0.1247611952D-03	-0.2859E+12
0.640E+00	0.1214145008D-03	0.1214144889D-03	-0.5294E+11
0.660E+00	0.1175886832D-03	0.1175886716D-03	0.1535E+12
0.680E+00	0.1132988149D-03	0.1132988037D-03	-0.2171E+12
0.700E+00	0.1085617868D-03	0.1085617760D-03	-0.2647E+11

0.720E+00	0.1033962547D-03	0.1033962445D-03	-0.3176E+11
0.740E+00	0.9782256761D-04	0.9782255795D-04	-0.1694E+12
0.760E+00	0.9186268820D-04	0.9186267912D-04	-0.1429E+12
0.780E+00	0.8554010761D-04	0.8554009916D-04	-0.1694E+12
0.800E+00	0.7887975382D-04	0.7887974603D-04	-0.1059E+11
0.820E+00	0.7190789422D-04	0.7190788711D-04	0.7941E+11
0.840E+00	0.6465203264D-04	0.6465202625D-04	-0.1059E+12
0.860E+00	0.5714080140D-04	0.5714079575D-04	-0.1323E+11
0.880E+00	0.4940384850D-04	0.4940384361D-04	0.6882E+11
0.900E+00	0.4147172062D-04	0.4147171652D-04	-0.3176E+11
0.920E+00	0.3337574221D-04	0.3337573891D-04	-0.7941E+10
0.940E+00	0.2514789126D-04	0.2514788877D-04	-0.6882E+11
0.960E+00	0.1682067212D-04	0.1682067046D-04	-0.1588E+11
0.980E+00	0.8426986071D-05	0.8426985238D-05	-0.1257E+11
0.100E+01	0.0000000000D+00	0.0000000000D+00	