OAK RIDGE NATIONAL LABORATORY
OPERATED BY
UNION CARBIDE CORPORATION
NUCLEAR DIVISION
POST OFFICE BOX X
OAK RIDGE, TENNESSEE 37830

ORNL/CSD/INF-81/11

DATE: January 7, 1982

SUBJECT: Periodic Parametric Cubic Spline Interpolation

TO: Distribution

FROM: C. W. Nestor, Jr.

ABSTRACT

In contour plotting and other applications it is often useful to interpolate a closed curve through a set of data points. The continuity properties of cubic splines produce interpolated curves that are smooth; the addition of periodic end conditions insures a smooth join where the curve closes. Given values of X and Y defining N points, and N+1 values of the parameter T, the subroutine PCUSPL will return the coefficients of the periodic cubic splines which interpolate X vs T and Y vs T. A possible choice for T is the cumulative point-to-point distance

$$T_1 = 0$$

$$T_i = T_{i-1} + \sqrt{(X_i - X_{i-1})^2 + (Y_i - Y_{i-1})^2}\ , \quad i = 2,3,\ldots,N$$

$$T_{N+1} = T_N + \sqrt{(X_1 - X_N)^2 + (Y_1 - Y_N)^2}$$

As an example of the use of PCUSPL, we show the results from interpolating through the vertices of regular polygons, plotting both the interpolated curve and the circle through the vertices. Program listings are included in the Appendix.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGMENT

The author wishes to thank P. W. Gaffney and M. R. Patterson for their careful review of this report.

Funds for the support of this work were provided by the Graphics Development Section of Computer Sciences.

INTRODUCTION

The DISSPLA graphics package provides for interpolation with parametric cubic splines (see Sect. 12.2, Part B of the User's Manual[1]) with two important limitations: first, a maximum of 51 points is allowed; and second, there is no provision for closed curves. The subroutine PCUSPL is designed to let the user dimension the arrays to be used as large as needed (anything you like, as long as you're willing to pay for the computer time) and to apply the cubic spline continuity conditions in a periodic way so that the generated curve is closed with a smooth join.

There is an excellent discussion of cubic spline interpolation in Forsythe, Malcolm and Moler[2]; for this reason, the derivation given in this report is fairly sketchy.

To show an example of the use of PCUSPL, we have plotted the interpolated curves through the vertices of regular polygons and plotted the circles through the vertices for comparison. The program to produce these plots is listed in the Appendix, in addition to PCUSPL.

---

[1] DISSPLA User's Manual, Integrated Software Systems Corporation, San Diego, California, 1978 Edition (Release 8.2).

[2] G. E. Forsythe, M. A. Malcolm and C. B. Moler, Computer Methods for Mathematical Computations, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1977, pp. 70-83.

DERIVATION

We use a parametric cubic interpolation function for each coordinate (X and Y) describing a curve through N+1 points:

$$X = X_i + b_i(t-t_i) + c_i(t-t_i)^2 + d_i(t-t_i)^3$$

$$(i = 1,2,\ldots,N)$$

$$Y = Y_i + \bar{b}_i(t-t_i) + \bar{c}_i(t-t_i)^2 + \bar{d}_i(t-t_i)^3$$

$$X_{N+1} = X_1$$

$$Y_{N+1} = Y_1$$

The parameter t must be increasing as the curve is traversed. A possible choice for t is the cumulative point-to-point distance:

$$t_i = 0;$$

$$t_i = t_{i-1} + \sqrt{(X_i - X_{i-1})^2 + (Y_i - Y_{i-1})^2} \quad (i = 2,\ldots,N); \text{ and}$$

$$t_{N+1} = t_N + \sqrt{(X_1 - X_N)^2 + (Y_1 - Y_N)^2}$$

For reasons of numerical stability and convenience we write the interpolation function in the form[2]

$$s(t) = w\, v_{i+1} + \bar{w}\, v_i + h^2_i\, [(w^3 - w)\ \sigma_{i+1} + (\bar{w}^3 - \bar{w})\ \sigma_i]$$

$$(i=1,2,\ldots,N;\ v_{N+1} = v_1)$$

where

v_i is the value of the dependent variable (either X or Y) at t_i;

$h_i = t_{i+1} - t_i$; and

$w = (t-t_i)/h_i$, $\bar{w} = 1 - w$

so that as t ranges over the subinterval $t_i \le t \le t_{i+1}$, w goes from 0 to 1 and $\bar{w}$ goes from 1 to 0. We apply the cubic spline continuity conditions (function, first and second derivatives) to obtain a system of N linear equations in the N unknown σ_i:

$$\begin{bmatrix} 2(h_1+h_N) & h_1 & 0 & 0\ldots & 0 & 0 & h_N \\ h_1 & 2(h_1+h_2) & h_2 & 0\ldots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0\ldots & h_{N-2} & 2(h_{N-2}+h_{N-1}) & h_{N-1} \\ h_N & 0 & 0 & 0\ldots & 0 & h_{N-1} & 2(h_{N-1}+h_N) \end{bmatrix} \begin{bmatrix} \sigma_1 \\ \sigma_2 \\ \vdots \\ \sigma_{N-1} \\ \sigma_N \end{bmatrix}$$

$$= \begin{bmatrix} \Delta_1 - \Delta_N \\ \Delta_2 - \Delta_1 \\ \vdots \\ \Delta_{N-1} - \Delta_{N-2} \\ \Delta_N - \Delta_{N-1} \end{bmatrix}$$

where

$$\Delta_i = (v_{i+1} - v_i)/h_i \qquad i = 1,2,\ldots,N-1$$

$$\Delta_N = (v_1 - v_N)/h_N$$

Note that the first and last elements of the main diagonal are different from those for non-periodic spline interpolation[2], and that the elements in the upper right and lower left corners are not zero. This results from the application of the continuity conditions at the joining point (X_1, Y_1). The matrix of this set of equations is symmetric, nearly tridiagonal, and strictly diagonally dominant; the solution can be calculated by Gaussian elimination, without pivoting, using an additional vector of N elements. A subroutine (PTRI) for the solution of the linear equations is given in the Appendix. Once the σ_i are calculated, the coefficients of the cubic polynomials can be calculated from

$$b_i = (v_{i+1} - v_i)/h_i - h_i\ (\sigma_{i+1} + 2\sigma_i)$$

$$c_i = 3\ \sigma_i \qquad\qquad i = 1,2,\ldots,N$$

$$d_i = (\sigma_{i+1} - \sigma_i)/h_i$$

where the b_i, c_i and d_i are calculated using x_i for the v_i, and $\bar{b}_i$, $\bar{c}_i$ and $\bar{d}_i$ are calculated using y_i for the v_i. After these coefficients have been calculated by PCUSPL, the user can call the subroutine SEVAL (Ref. 2, p. 79; CORLIB notes) with several values of t to compute x's and y's for a set of intermediate points; these can then be plotted with a call to the DISSPLA subroutine CURVE. See the sample program (PCPLT) in the Appendix.

USE

All floating-point arrays used by PCUSPL must be defined as double precision by the user. Given two arrays X and Y of N elements each, the coordinates of the points to be interpolated, the user must calculate N+1 values of the parameter t, as indicated by the following program fragment:

```
          :
          T(1) = 0.D0
          NM1 = N - 1
          DO 100 I = 1, NM1
          T(I+1) = T(I) + DSQRT ((X(I+1) - X(I))**2 + (Y(I+1) - Y(I))**2)
     100  CONTINUE
          T(N+1) = T(N) + DSQRT ((X(1) - X(N))**2 + (Y(1) - Y(N))**2)
          :
```

The calling sequence is

```
CALL PCUSPL (X, Y, T, N, BX, CX, DX, BY, CY, DY, A, B, V)
```

where X, Y and T are as defined above; BX, CX and DX will contain, on return, the coefficients b_i, c_i and d_i; BY, CY and DY will contain $\bar{b}_i$, $\bar{c}_i$ and $\bar{d}_i$; the arrays A, B and V are used as workspace. Except for N, the number of points, all variables are double-precision arrays and must be dimensioned at least as large as N. (T must be dimensioned at least N+1.)

After calculating the coefficients of the cubics, the user can call the function subprogram SEVAL to calculate a new set of coordinates for plotting:

```
          DOUBLE PRECISION TP(129), T(  ), X( ), Y( ),...
          DIMENSION XP(129), YP(129)
          :
          (Statements defining TP array)

          DO 100 K=1,NP
            XP(K) = SEVAL (N, TP(K), T, X, BX, CX, DX)
            YP(K) = SEVAL (N, TP(K), T, Y, BY, CY, DY)
     100  CONTINUE
          :
          (Calls to DISSPLA setup routines)
          :
          CALL CURVE (XP, YP, 129, 0)
          :
```

Source programs for PCUSPL and PTRI are available on the X-10 PDP-10 in the two files DSKD:PCUSPL.FOR[6137,17] and DSKD:PTRI.FOR[6137,17]. The program to produce the regular polygon comparison plots is stored in DSKD:PCPLT.FOR[6137,17]. The subroutine DSKD:FILLEM.FOR[6137,17] is also needed. When using SEVAL to calculate additional coordinates for plotting, follow the directions in the HELP file for CORLIB.

TEST RESULTS

As an example of the use of PCUSPL, we have plotted in Figures 1 and 2 the curves obtained from interpolating periodic parametric cubic splines through points representing the vertices of regular polygons with 3, 4, 5 and 6 vertices. For comparison, we have also plotted a circle through the vertices. The two curves are nearly indistinguishable for the six points representing a hexagon.

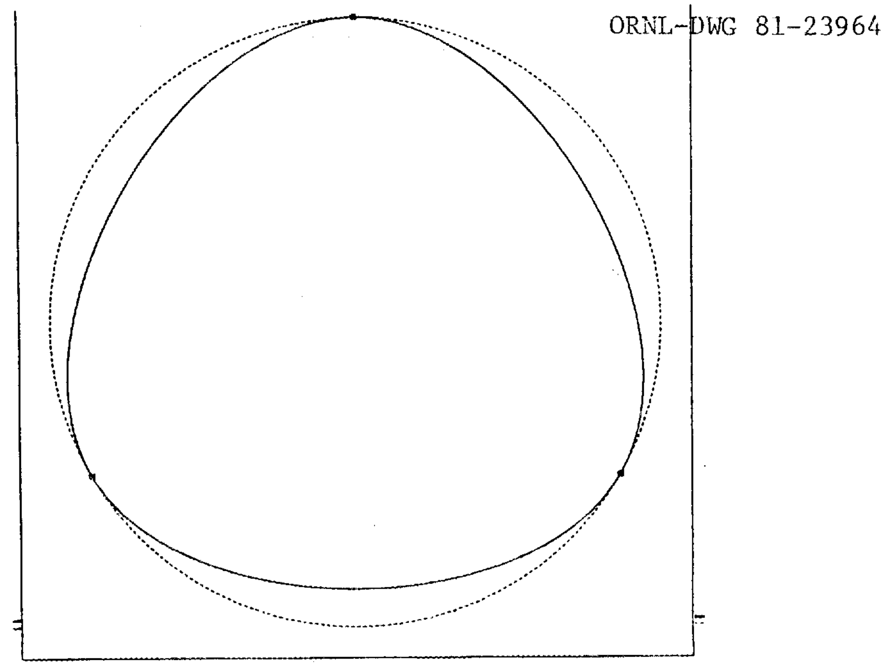

Figure 1a. Periodic Cubic Spline (Solid) and Circle (Dash) for Equilateral Triangle (N=3)

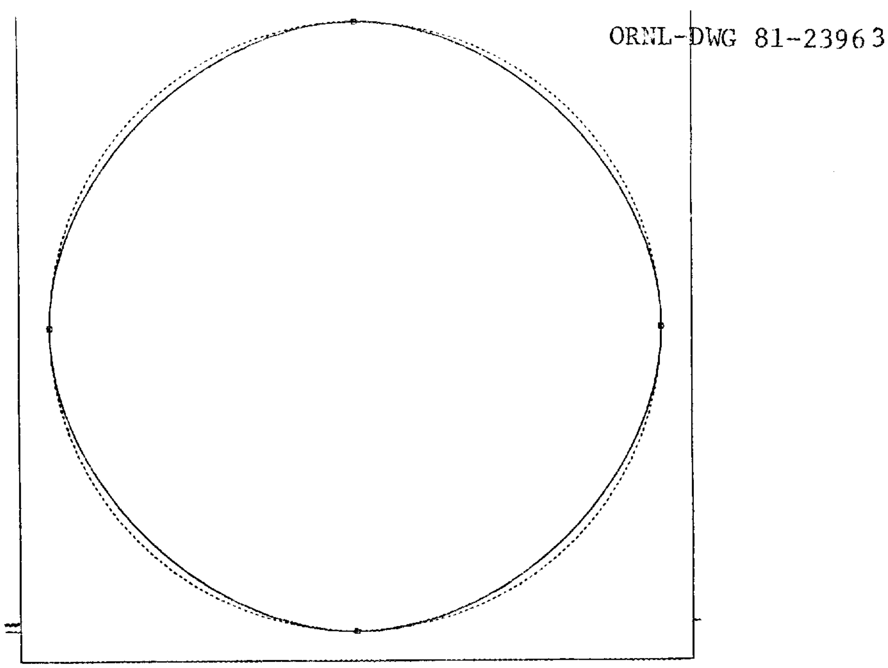

Figure 1b. Square (N=4)

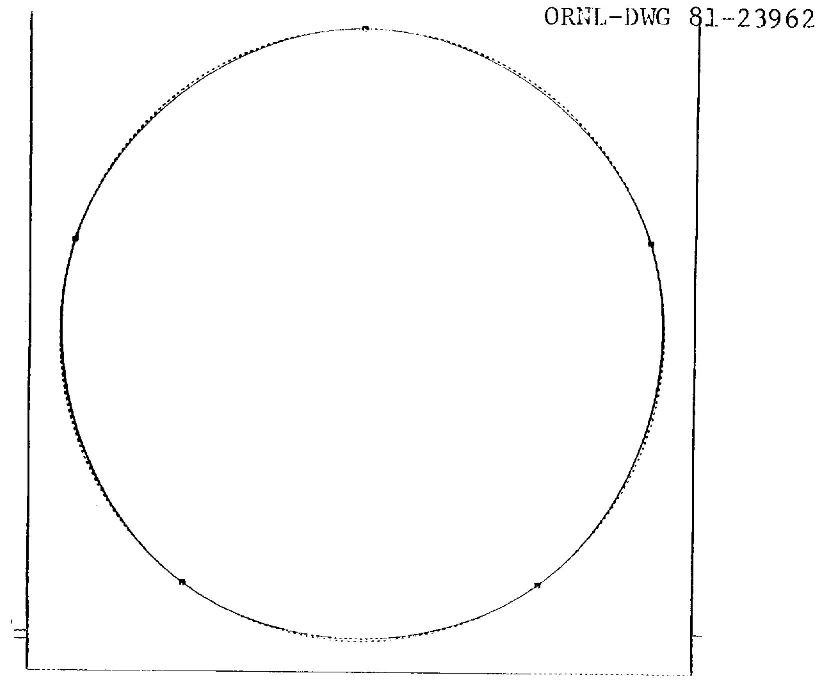

Figure 2a. Regular Pentagon (N=5)

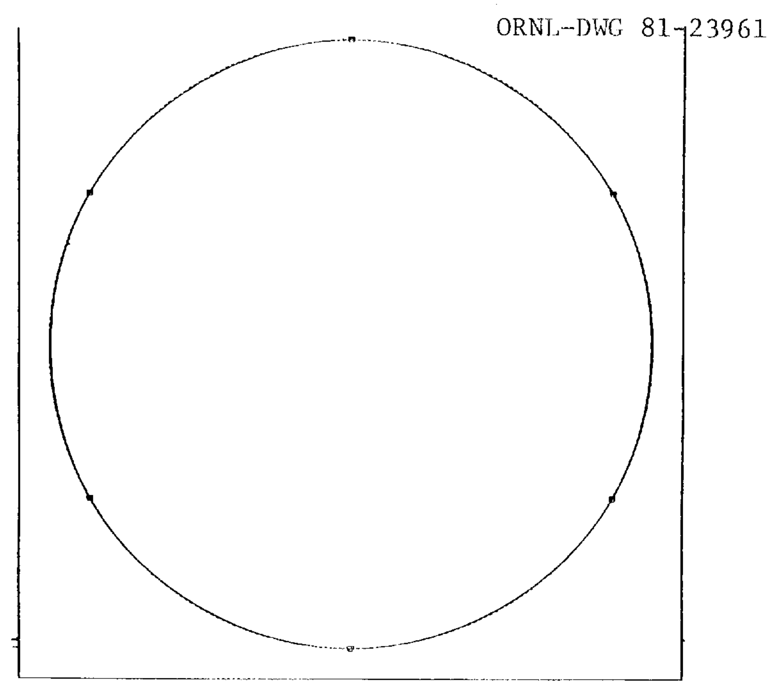

Figure 2b. Regular Hexagon (N=6)

APPENDIX. PROGRAM LISTINGS

```
C        PCUSPL.FOR (22 OCTOBER 1981)
C
      SUBROUTINE PCUSPL
     1   (X, Y, T, N, BX, CX, DX, BY, CY, DY, A, B, V)
      IMPLICIT REAL*8 (A-H, O-Z)
      DIMENSION
     1  A(1),           X(1),        Y(1),       T(1),        BX(1),
     2  CX(1),          DX(1),       BY(1),      CY(1),       DY(1),
     3  B(1),           V(1)
C
C        PERIODIC PARAMETRIC CUBIC SPLINE TO INTERPOLATE
C        THE N + 1 POINTS (X, Y), WITH X(N+1) = X(1), Y(N+1) = Y(1).
C
C        FOR DERIVATION, NOTATION ET C., SEE FORSYTHE, MALCOLM
C        AND MOLER, COMPUTER METHODS FOR MATHEMATICAL COMPUTATIONS.
C
C        ON ENTRY:
C
C        X AND Y ARE THE COORDINATES OF THE POINTS;
C        (REAL*8 ARRAYS, DIMENSIONED AT LEAST N)
C
C        T CONTAINS THE VALUES OF THE PARAMETER FOR THE POINTS.
C        (REAL*8 ARRAY, DIMENSIONED AT LEAST N+1)
C        A POSSIBLE CHOICE IS THE POLYGONAL ARC LENGTH
C
C
C          T(1) = 0; T(I) = T(I-1) + DSQRT((X(I) - X(I-1))**2 +
C                                          (Y(I) - Y(I-1))**2);
C                    T(N+1) = T(N) + DSQRT((X(1) - X(N))**2 +
C                                          (Y(1) - Y(N))**2
C
C        N MUST CONTAIN THE NUMBER OF POINTS; AND
C
C        THE ARRAYS A, B, V, BX, CX, DX, BY, CY, AND DY
C        MUST BE DIMENSIONED AT LEAST N IN THE CALLING PROGRAM
C
C        ON RETURN:
C
C        BX, CX, AND DX WILL CONTAIN THE COEFFICIENTS OF THE CUBIC
C        POLYNOMIALS FOR X, AND
C        BY, CY, AND DY WILL CONTAIN THE COEFFICIENTS OF THE CUBIC
C        POLYNOMIALS FOR Y.
C
      NM1 = N - 1
      DO 100 J=1,NM1
        BX(J) = T(J+1) - T(J)
        DX(J) = (X(J+1) - X(J))/BX(J)
        DY(J) = (Y(J+1) - Y(J))/BX(J)
```

```
 100  CONTINUE
      BX(N) = T(N+1) - T(N)
      DX(N) = (X(1) - X(N))/BX(N)
      DY(N) = (Y(1) - Y(N))/BX(N)
      DO 110 I=2,NM1
        A(I) = BX(I-1) + BX(I-1) + BX(I) + BX(I)
        BY(I) = A(I)
        B(I) = BX(I)
        CX(I) = DX(I) - DX(I-1)
        CY(I) = DY(I) - DY(I-1)
 110  CONTINUE
      A(N) = BX(N-1) + BX(N-1) + (BX(N) + BX(N))
      BY(N) = A(N)
      A(1) = BX(1) + BX(1) + (BX(N) + BX(N))
      BY(1) = A(1)
      B(1) = BX(1)
      CX(1) = DX(1) - DX(N)
      CY(1) = DY(1) - DY(N)
      CX(N) = DX(N) - DX(N-1)
      CY(N) = DY(N) - DY(N-1)
      CALL PTRI (A, B, BX(N), CX, V, N)
C
C        THE ARRAY CX NOW CONTAINS SIGMA(X)
C
      Q = BX(N)
      CALL PTRI (BY, B, Q, CY, V, N)
C
C        THE ARRAY CY NOW CONTAINS SIGMA(Y)
C
      CX1 = CX(1)
      CY1 □ CY(1)
      DO 115 I=1,NM1
        HT = BX(I)
        BX(I) = DX(I) - BX(I)*(CX(I+1) + CX(I) + CX(I))
        DX(I) = (CX(I+1) - CX(I))/HT
        CX(I) = 3.DO*CX(I)
        BY(I) = DY(I) - HT*(CY(I+1) + CY(I) + CY(I))
        DY(I) = (CY(I+1) - CY(I))/HT
        CY(I) = 3.DO*CY(I)
 115  CONTINUE
      HT = BX(N)
      BX(N) = DX(N) - BX(N)*(CX1 + CX(N) + CX(N))
      DX(N) = (CX1 - CX(N))/HT
      CX(N) = 3.DO*CX(N)
      BY(N) = DY(N) - HT*(CY1 + CY(N) + CY(N))
      DY(N) = (CY1 - CY(N))/HT
      CY(N) = 3.DO*CY(N)
      RETURN
      END
```

```
C        PCPLT.FOR (3 FEB. 1981)
C
      DOUBLE PRECISION
     1  X(6),        Y(6),        T(7),        BX(6),       CX(6),
     2  DX(6),       BY(6),       CY(6),       DY(6),       A(6),
     3  B(6),        V(6),        TP(257),     TWOPI,       THET,
     4  CT,          ST,          CDT,         SDT,         Q
      DIMENSION
     1  XP(257),     YP(257)
      DATA TWOPI /6.28318530717958D0/
      CALL COMPRS
      CALL PHYSOR (.5, .5)
      CALL PAGE (11., 11.)
C
 100  CONTINUE
      WRITE (5, 1)
 1    FORMAT (1H0, 'ENTER N: ', $)
      READ (5, 2) N
 2    FORMAT (I)
      IF (N .EQ. 0) GO TO 900
      CALL TITLE (0, 0, 0, 0, 0, 0, 10., 10.)
      CALL GRAF (-5., 10., 5., -5., 10., 5.)
      THET = TWOPI/DFLOAT(N)
      CDT = DCOS (THET)
      SDT = DSIN (THET)
      CT = 1.D0
      ST = 0.D0
      T(1) = 0.D0
      Y(1) = 0.D0
      X(1) = 5.D0
      DO 105 I=2,N
        Q = CT*CDT - ST*SDT
        ST = ST*CDT + CT*SDT
        CT = Q
        X(I) = 5.D0*CT
        Y(I) = 5.D0*ST
        T(I) = T(I-1) + DSQRT ((X(I) - X(I-1))**2 +
     1                         (Y(I) - Y(I-1))**2)
```

```
 105  CONTINUE
      TP(257) = T(N) + DSQRT ((X(N) - X(1))**2 +
     1                         (Y(N) - Y(1))**2)
      T(N+1) = TP(257)
      CALL PCUSPL (X, Y, T, N, BX, CX, DX, BY, CY, DY,
     1  A, B, V)
      TP(1) = 0.D0
      CALL FILLEM (TP, 256)
      DO 110 I=1,257
        XP(I) = SEVAL (N, TP(I), T, X, BX, CX, DX)
        YP(I) = SEVAL (N, TP(I), T, Y, BY, CY, DY)
 110  CONTINUE
      CALL CURVE (XP, YP, 257, 0)
      DO 107 I=1,N
        XP(I) = X(I)
        YP(I) = Y(I)
 107  CONTINUE
      CALL MARKER (0)
      CALL CURVE (XP, YP, N, -1)
      CALL DASH
      CALL CIRC (0., 0., 5., .02, XP, YP, 257)
      CALL RESET ('DASH')
      CALL ENDPL (0)
      GO TO 100
C
 900  CONTINUE
      CALL DONEPL
      STOP
      END
```

```
C        PTRI.FOR (15 DECEMBER 1980)
C
      SUBROUTINE PTRI (A, B, C, Y, V, N)
      IMPLICIT REAL*8 (A-H, O-Z)
      DIMENSION A(5), B(5), V(5), Y(5)
      DO 100 I=1,N
        V(I) = 0.D0
  100 CONTINUE
      V(1) = C
      V(N-1) = B(N-1)
      NM2 = N - 2
      DO 105 I=1,NM2
        R = B(I)/A(I)
        A(I+1) = A(I+1) - R*B(I)
        Y(I+1) = Y(I+1) - R*Y(I)
        R = V(I)/A(I)
        V(I+1) = V(I+1) - R*B(I)
        A(N) = A(N) - R*V(I)
        Y(N) = Y(N) - R*Y(I)
  105 CONTINUE
      R = V(N-1)/A(N-1)
      A(N) = A(N) - R*V(N-1)
      Y(N) = Y(N) - R*Y(N-1)
      Y(N) = Y(N)/A(N)
      Y(N-1) = (Y(N-1) - V(N-1)*Y(N))/A(N-1)
      DO 110 K=1,NM2
        I = NM2 + 1 - K
        Y(I) = (Y(I) - V(I)*Y(N) - B(I)*Y(I+1))/A(I)
  110 CONTINUE
      RETURN
      END
```

```
C         FILLEM.FOR (28 MAY 1981)
C
      SUBROUTINE FILLEM (X, NINT)
      IMPLICIT REAL*8 (A-H, O-Z)
      DIMENSION X(1)
C
C         ON ENTRY:     X(1) = XMIN
C                   X(NINT+1) = XMAX
C
C         ON RETURN, THE INTERMEDIATE ELEMENTS WILL CONTAIN
C         EQUALLY-SPACED VALUES OF X WITH SPACING
C
C         DX = (XMAX - XMIN)/NINT
C
      IF (NINT .LE. 1) RETURN
      T = 1.D0/DFLOAT (NINT)
      DT = T
      CT = 1.D0 - DT
      XLAST = X(NINT+1)
      DO 100 N=2,NINT
        X(N) = T*XLAST + CT*X(1)
        T = T + DT
        CT = CT - DT
  100 CONTINUE
      RETURN
      END
```

DISTRIBUTION

1. V. E. Anderson
2. W. H. Atkins
3. J. D. Bell
4. A. A. Brooks/CSD K-25 Library
5. H. P. Carter/G. E. Whitesides/ CSD X-10 Library
6. B. A. Casstevens
7. B. A. Clark
8. J. S. Crowell
9. P. W. Gaffney
10. N. M. Larson
11. E. C. Long
12. L. B. Maddox
13. C. H. Malarkey
14. G. S. McNeilly
15. L. D. Merriman
16. P. S. Meszaros
17. B. D. Murphy
18. F. W. Nehring

19-24. C. W. Nestor, Jr.

25. L. W. Owen
26. M. R. Patterson
27. R. O. Sayer
28. L. A. Schnautz

29-30. Lab Records

31. Lab Records - RC
32. Y-12 Central Files
33. ORNL Patent Office
34. Central Research Library