# APPENDIX C

## CODE DEVELOPMENT AND COMPUTER EXERCISES

### 1. Code Development Exercises

1. Take the computational region to be a rectangle on which the curvilinear coordinates are defined to be equal to the indices of the field arrays, i.e., $\xi$ = I and $\eta$ = J, with x = X (I,J) and y = Y (I,J).

Make provision for reading in values of x and y on any segments of the boundary of the computational region. Generate x and y in the interior by interpolating linearly between the top and bottom boundaries. Plot the grid.

2. Modify the code to allow horizontal (in the computational plane) interpolation, as well, the choice being specified by input.

3. Now add the choice of interpolation from the four corners (tensor product interpolation).

4. Finally add the choice of transfinite interpolation.

5. Generalize the interpolation to cubic Hermite interpolation, with the grid being orthogonal at the boundary.

6. Generalize the interpolation to use the hyperbolic tangent distribution function, rather than being linear, with specified relative spacing on each end.

7. Modify the code to provide for reading in x and y on any segment of any horizontal or vertical line in the computational region. Also provide for the interpolation tion to be done on any rectangular segment of the computational region (including a segment that is only a line.)

8. Add another field array TYPE (I,J) which is a flag to identify each point as one for which the x,y values are (1) fixed, e.g., specified points on the physical boundary, (2) out of the computation, e.g., points inside a slab, or (3) to be generated. Provide for the designation (1) and (2) to be made by input for any rectangular segment of the computational region, the default being to the designation (3).

9. Modify the dimensions of the field arrays so that an extra layer of points surrounds the computational region. Also add two more field arrays, ILINK (I,J) and JLINK (I,J). Provide for any segments of any horizontal or vertical lines to be designated as image points in TYPE by input, i.e., points for which the values of x and y are set equal to those at some other point. Also provide for the indices of these other points to be put in ILINK and JLINK by input.

10. Add an elliptic generator, based on Laplace equation, to the code. Use the algebraic generator (the interpolation) to provide the initial guess for point SOR iteration.

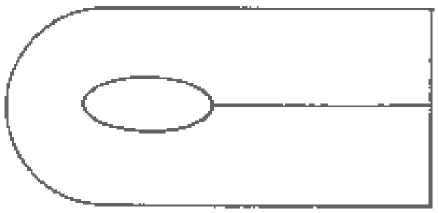
11. Add control functions to the elliptic generator. Let the control function be evaluated on the boundaries and interpolated into the field by transfinite interpolation.

## 2. Computer Exercises

1. Generate an algebraic grid between two concentric circles. Use linear interpolation between the circles.

2. Generate an algebraic grid between two ellipses, both of which are centered at the origin but which may have different eccentricities, using interpolation between the ellipses. Compare grids generated using linear and Hermite interpolation, the latter being orthogonal at the boundaries.

3. Generate a C-type algebraic grid for an ellipse inside an outer boundary formed by a semicircle replacing one side of a rectangle:



Compare (1) vertical interpolation in the computational region boundary, (2) horizontal interpolation, (3) tensor product interpolation, and (4) transfinite interpolation, using linear interpolation in each case. Note that (2) and (3) are totally unreasonable.

4. Generate an algebraic grid for a circular simply-connected region by (1) unidirectional interpolation, (2) tensor product interpolation, and (3) transfinite interpolation. Note that here only (3)gives a reasonable grid. Compare linear and Hermite interpolation for (3).

5. Repeat Exercise 4 with a triangular boundary.

6. Using the boundary configuration of Exercise 3, but with a hyperbolic tangent point distribution on the right-hand boundary of the physical region with smaller spacing at the centerline than at the top and bottom. Compare algebraic grids generated using (1) linear interpolation between the inner and outer boundaries, (2) nonlinear interpolation, based on the hyperbolic tangent, between the inner and outer boundaries, (3) transfinite interpolation with linear blending functions, and (4) transfinite interpolation using the boundary point distribution (in terms of relative arc length) as the blending functions. Note that only (2) and (4) preserve the boundary point distribution in the field.

7. Generate an algebraic grid for a square inside a rectangle using linear interpolation between the inner and outer boundaries. Note the propagation of the boundary slope discontinuities into the field. Generate a grid from an elliptic generation system for the same boundary point distribution and note the difference.

8. Generate an algebraic grid for a square inside a circle using linear interpolation between

the inner and outer boundaries. Show that it is possible to position the points on the circle such that the grid overlaps the corners of the square. Generate a grid from an elliptic generation system for the same boundary point distribution and note the difference.

## 3. Listing of Routine for Computer Exercises

```
      SUBROUTINE INTERP
      PARAMETER(NI=20,NJ=20,N=5)
      COMMON/COORD/X(NI,NJ),Y(NI,NJ)
      COMMON/CONST/CHOICE,IMAX,JMAX,NA,DS1,DS2
      COMMON/ATTR/IAL(N),IAX(N),IAY(N),JAL(M),JAX(N),JAY(N)
      COMMON/COEF/AI(N),BI(N),CI(N),DI(N),AJ(N),BJ(N)
      COMMON/COEF/CJ(N),DJ(N)
      DIMENSION P(NI,NJ),Q(NI,NJ),XX(0:NI,O:NJ),YY(0:MI,O:NJ)
      DIMENSION X1(NI,NJ),X2(NI,NJ),Y1(NI,NJ),Y2(NI,NJ)
      INTEGER CHOICE
C
C     BOUNDARY INTERPOLATION
C
C      X           X ARRAY OF XI-ETA COORDINATE
C      Y           Y ARRAY OF XI-ETA COORDINATE
C      IMAX        MAX. NUMBER OF GRID IN XI AXIS
C      JMAX        MAX. NUMBER OF GRID IN ETA AXIS
C      NA          MAX. NUMBER OF ATTRACTIONS
C      DSl         SPECIFIED LENGTH OF INITIAL INTERVAL
C      DS2         SPECIFIED LENGTH OF FINAL INTERVAL
C      ATTR        ARRAY OF ATTRACTION TO LINES/POINTS
C      COEF        ARRAY OF COEFFICIENT FOR ATTRACTION
C
C      CHOICE
C         1        VERTICAL INTERPOLATION
C         2        HORIZONTAL INTERPOLATION
C         3        TENSOR PRODUCT INTERPOLATION
C         4        TRANSFINITE INTERPOLATION
C         5        HERMITE CUBIC INTERPOLATION
C         6        HYPERBOLIC TANGENT INTERPOLATION
C         7        ELLIPTIC GRID GENERATION ( SOR ITERATION )
C         8        ATTRACTION TO COORDINATES
C
       IF(CHOICE.EQ.1) GO TO 100
       IF(CHOICE.EQ.2) GO TO 200
       IF(CHOICE.EQ.3) GO TO 300
       IF(CHOICE.EQ.H) GO TO 400
       IF(CHOICE.EQ.5) GO TO 500
       IF(CHOICE.EQ.6) GO TO 600
       IF(CHOICE.EQ.7) GO TO 700
       IF(CHOICE.EQ.S) GO TO 800
C
C      **** VERTICAL INTERPOLATION ****
C
 100   DO 110 I 1,IMAX
       DO 110 J 1,JMAX
       RJ1=FLOAT(JMAX-J)/FLOAT(JMAX-1)
       RJ2=FLOAT(J-1)/FLOAT(JMAX-1)
C      *** ( EQ. 8-1 )
       X(I,J)=RJ1*X(I,l)+RJ2*X(I,JMAX)
 110   Y(I,J) RJl*Y(I.1)+RJ2*Y(I,JMAX)
       RETURN
C
C      **** HORIZONTAL INTERPOLATION ****
```

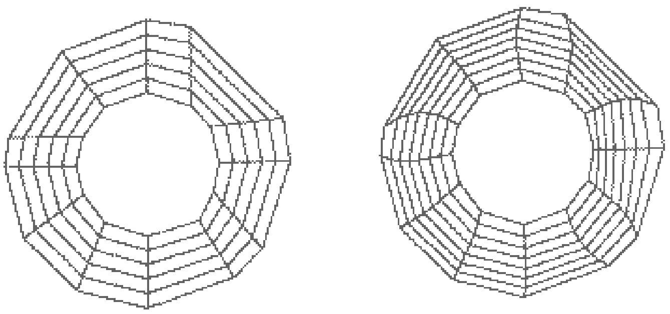
```
C
 200   DO 210 I=1,JMAX
       DO 210 J=1,IMAX
       RI1=FLOAT(IMAX-I)/FLOAT(IMAX-1)
       RI2=FLOAT(I-1)/FLOAT(IMAX-1)
C      *** ( EQ. 8-1 )
       X(I,J)=RI1*X(1,J)+RI2*X(IMAX,J)
 210   Y(I,J)=RI1*Y(1,J)+RI2*Y(IMAX,J)
       RETURN
C
C      **** TENSOR PRODUCT INTERPOLATION ****
C
 300   DO 310 I=1,IMAX
       DO 310 J=1,JMAX
       RI1=FLOAT(IMAX-I)/FLOAT(IMAX-1)
       RI2=FLOAT(I-1)/FLOAT(IMAX-1)
       RJ1=FLOAT(JMAX-J)/FLOAT(JMAX-1)
       RJ2=FLOAT(J-1)/FLOAT(JMAX-1)
C      *** ( EQ. 8-69 )
       X(I,J)=RI1*RJ1*X(1,1)+RI1*RJ2*X(1,JMAX)
      *+RI2*RJ1*X(IMAX,1)+RI2*RJ2*X(IMAX,JMAX)
       Y(I,J)=RI1*RJ1*Y(1,1)+RI1*RJ2*Y(1,JMAX)
      *+RI2*RJ1*Y(IMAX,1)+RI2*RJ2*Y(IMAX,JMAX)
 310   CONTINUE
       RETURN
C
C      **** TRANSFINITE INTERPOLATION ****
C
 400   DO 410 I=1,IMAX
       DO 410 J=1,JMAX
       RI1=FLOAT(I-1)/FLOAT(IMAX-1)
       RI2=FLOAT(IMAX-I)/FLOAT(IMAX-1)
       X1(I,J)=RI1*X(IMAX,J)+RI2*X(1,J)
 410   Y1(I,J)=RI1*Y(IMAX,J)+RI2*Y(1,J)
       DO 420 I=1,IMAX
       DO 420 J=1,JMAX
       RJ1=FLOAT(J-1)/FLOAT(JMAX-1)
       RJ2=FLOAT(JMAX-J)/FLOAT(JMAX-1)
       X2(I,J)=RJ1*(X(I,JMAX)-X1(I,JMAX))+RJ2*(X(I,1)-X1(I,1))
 420   Y2(I,J)=RJ1*(Y(I,JMAX)-Y1(I,JMAX))+RJ2*(Y(I,1)-Y1(I,1))
C      *** ( EQ. 8-73 )
       DO 430 I=1,IMAX
       DO 430 J=1,JMAX
       X(I,J)=X1(I,J)+X2(I,J)
 430   Y(I,J)=Y1(I,J)+Y2(I,J)
       IF(CHOICE.NE.4) GO TO 740
       RETURN
C
C      ***  HERMITE CUBIC INTERPOLATION (ORTHOGONAL BOUNDARY) ***
C
 500   DO 510 I=1,IMAX
       DO 510 J=1,JMAX
       XX(I,J)=X(I,J)
 510   YY(I,J)=Y(I,J)
       DO 520 J=1,JMAX
       XX(O,J)=XX(IMAX-1,J)
       YY(O,J)=YY(INAX-1,J)
       XX(IMAX+1,J)=XX(2,J)
 520   YY(IMAX+1,J)=YY(2,J)
       DO 530 I=1,IMAX
       DO 530 J=1,JMAX
```

```
      RJJ=FLOAT(J-1)/FLOAT(JMAX-1)
C     *** ( EQ. 8-6 a and b, n=2 )
      PHI1=(1.+2.*RJJ)*(1.-RJJ)*(1.-RJJ)
      PHI2=(3.-2.*RJJ)*RJJ*RJJ
      PSI1=(1.-RJJ)*(1.-RJJ)*RJJ
      PSI2=(RJJ-1.)*RJJ*RJJ
C
C     ** CAL. NORMAL DERIV. **
C
      XXI1=.5*(XX(I+1,1)-XX(I-1,1))
      XXI2=.5*(XX(I+1,JMAX)-XX(I-1,JMAX))
      YXI1=.5*(YY(I+1,1)-YY(I-1,1))
      YXI2=.5*(YY(I+1,JMAX)-YY(I-1,JMAX))
      UNIT1=SQRT(XXI1*XXI1+YXI1*YXI1)
      UNIT2=SQRT(XXI2*XXI2+YXI2*YXI2)
C     *** ( EQ. 3-108 )
      XN1=-YXI1/UNIT1*DS1
      XN2=-YXI2/UNIT2*DS2
      YN1=XXI1/UNIT1*DS1
      YN2=XXI2/UNIT2*DS2
C     *** ( EQ. 8-5 )
      XX(I,J)=PHI1*XX(I,1)+PHI2*XX(I,JMAX)+PSI1*XN1+PSI2*XN2
 530  YY(I,J)=PHI1*YY(I,1)+PHI2*YY(I,JMAX)+PSI1*YN1+PSI2*YN2
      DO 540 I=1,IMAX
      DO 540 J=1,JMAX
      X(I,J)=XX(I,J)
 540  Y(I,J)=YY(I,J)
      RETURN
C
C     **** HYPERBOLIC TANGENT SPACING INTERPOLATION ****
C
 600  TOL=1.0E-10
C     *** ( EQ. 8-49, 50 and 51
      A=SQRT(DS2/DS1)
      B=1./(FLOAT(JMAX-1)*SQRT(DS1*DS2))
C     *** INITIAL GUESS BY SERIES EXPANSION
      DELTA=SQRT(6.*(B-1.))
      DO 610 IT=1,20
      RESID=SINH(DELTA)/(DELTA*B)-1.
      IF(ABS(RESID)LT.TOL) GO TO 630
 610  CALL AITKEN(DELTA,RESID,DELTO,RO,RSO)
      PRINT 620, RESID,DELTA,IT-1
 620  FORMAT(//, 5X, 'DELTA IS NOT CONVERGE ?', 5X, 2E15.5,
     *5X, I3, //)
      GO TO 660
 630  CONTINUE
C     *** ( EQ. 8-52, 53 and 54 )
      DO 650 I=1,IMAX
      DO 650 J=2,JMAX-1
      RATIO FLOAT(J-1)/FLOAT(JMAX-1)
      U=.5*(1.+TANH(DELTA"(RATIO-.5))/TANH(.5*DELTA))
      S=U/(A+(1.-A)*U)
      X(I,J)=X(I,1)+(X(I,JMAX)-X(I,1))*S
 650  Y(I,J)=X(I,1)+(Y(I,JMAX)-Y(I,1))*S
 660  RETURN
C
C     **** ELLIPTIC GRID GENERATION ( SOR ITERATION ) ****
C     *** CAL. P AND Q ON THE BOUNDARY **
C
 700  DO 710 I=1,IMAX
      DO 710 J=1,JMAX
```

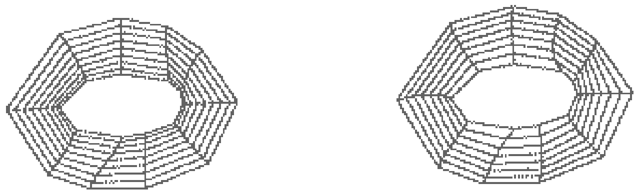
```
      XXI=.5*(X(I+1,J)-X(I-1,J))
      XXIXI=X(I+1,J)-2.*X(I,J)+X(I-1,J)
      XETA=.5*(X(I,J+1)-X(I,J-1))
      XETA2=X(I,J+1)-2.*X(I,J)+X(I,J-1)
      YXI=.5*(X(I+1,J)-Y(I-1,J))
      XXIXI=Y(I+1,J)-2.*X(I,J)+Y(I-1,J)
      YETA=.5*(Y(I,J+1)-Y(I,J-1))
      YETA2-Y(I,J+1)-2.*Y(I,J)+Y(I,J-1)
      IF(ABS(XETA2).LT.10E-3) XETA2=0.
      IF(ABS(YETA2).LT.10E-3) YETA2=0.
      RXI2=XXI*XXI+YXI*YXI
      RETA2=XETA*XETA+YETA*YETA
C     *** ( EQ. 8-70 )
      P(I,J)=(XXI*XXIXI+XXI*YXIXI)/RXI2
      Q(I,J)=(XETA*XETA2+YETA*YETA2)/RETA2
 710  CONTINUE
C
C     **    INTERPOLATE P AND Q BETWEEN BOUNDARY **
C           P : VERTICAL, Q : HORIZONTAL
C
      DO 720 I=1,IMAX
      DO 720 J=l,JMAX
      RJ1=FLOAT(JMAX-J)/FLOAT(JMAX-1)
      RJ2=FLOAT(J-1)/FLOAT(JMAX-1)
      RI1=FLOAT(IMAX-I)/FLOAT(IMAX-1)
      RI2=FLOAT(I-1)/FLOAT(IMAX-1)
      P(I,J)=RJl*P(I,l)+RJ2*P(I,JMAX)
      Q(I,J)=RI1*Q(1,J)+RI2*Q(IMAX,J)
 720  CONTINUE
C
C     **    INITIAL GUESS WITH TRANSFINITE INTERPOLATION **
C
      GO TO 400
 740  CONTINUE
C
C     *** ITERATION ( SOR ) ***
C
      ITMAX=200
      TOL=10.E-5
      W=1.8
      DO 760 IT=1,ITMAX
      ERRX=0.
      ERRY=0.
      DO 750 J=2,JMAX-1
      DO 750 I=2,IMAX-1
      XXI=.5*(X(I+1,J)-X(I-1,J))
      YXI=.5*(Y(I+1,J)-Y(I-1,J))
      XXIXI=X(I+1,J)+X(I-1,J)
      YXIXI=Y(I+1,J)+Y(I-1,J)
      XETA=.5*(X(I,J+1)-X(I,J-1))
      YETA .5*(Y(I,J+1)-Y(I,J-1))
      XXIETA=.25*(X(I+1,J+1)-X(I+1,J-1)-X(I-1,J+1)+X(I-1,J-l))
      YXIETA=.25*(Y(I+1,J+1)-Y(I+1,J-1)-Y(I-1,J+1)+Y(I-1,J-1))
      XETA2=X(I,J+1)+X(I,J-1)
      YETA2=Y(I,J+1)+Y(I,J-1)
C     *** ( EQ. 6-18 and 6-20 )
      G11=XXI*XXI+YXI*YXI
      G22=XETA*XETA+YETA*YETA
      G12=XXI*XETA+YXI*YETA
      XTEMP=.5*(G22*(P(I,J)*XXI+XXIXI)+G11*(Q(I,J)*XETA+XETA2)
     *-2.*G12*XXIETA)/(G11+G22)
```

```
      YTENP=.5*(G22*(P(I,J)*YXI+YXIXI)+G11*(Q(I,J)*YETA+YETA2)
     *-2.*Gf2*YXIETA)/(Gl1+G22)
      XTENP=W*XTEMP+(1.-W)*X(I,J)
      YTEMP=W*YTEMP+(1.-W)*Y(I,J)
      ERRX=AMAXO(ERRX,ABS(XTEMP-X(I,J)))
      ERRY=AMAXO(ERRY,ABS(YTEMP-Y(I,J)))
      X(I,J)=XTEMP
      Y(I,J)=YTEMP
 750  CONTINUE
      IF(ERRX.LT.TOL.AND.ERRY.LT.TOL) GO TO 78O
 760  CONTINUE
      PRINT 770,ERRX,ERRY,IT-1
 770  FORMAT(//, 5X, 'X AND Y ARE NOT CONVERGE ?', 2E15.5,
     *5X, I5, //)
 780  CONTINUE
      IF(CHOICE.EQ.8) GO TO 830
      RETURN
C
C     **** ATTRACTION TO COORDINATE LINE/POINT ****
C
 800  DO 810 I=1,IMAX
      DO 810 J=1,JMAX
      P(I,J)=0.
 810  Q(I,J)=0.
      DO 820 NS=1,NA
      DO 820 I=1,IMAX
      DO 820 J=1,JMAX
      XL=FLOAT(I-IAL(NS))
      XI=FLOAT(I-IAX(NS))
      XJ=FLOAT(J-IAY(NS))
      YL=FLOAT(J-JAL(NS))
      YI=FLOAT(I-JAX(NS))
      YJ=FLOAT(J-JAY(NS))
C     *** ( EQ. 6-30 )
      P(I,J)=P(I,J)-AI(NS)*(XL/ABS(XL))*EXP(-CI(NS)*ABS(XL))
     *-BI(NS)*(XI/ABS(XI))*EXP(-DI(NS)*SQRT(XI*XI+XJ*XJ))
      Q(I,J)=Q(I,J)-AJ(NS)*(YL/ABS(YL))*EXP(-CJ(NS)*ABS(YL))
     *-BJ(NS)*(YJ/ABS(YJ))*EXP(-DJ(NS)*SQHT(YI*YI+YJ*YJ))
 820  CONTINUE
      GO TO 400
 830  CONTINUE
      RETURN
      END
```

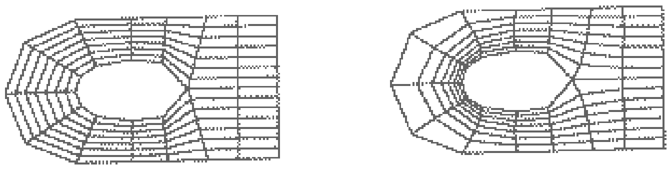
1.



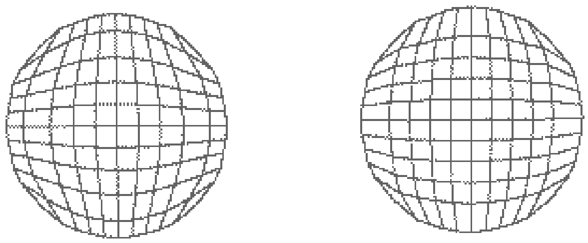
2.



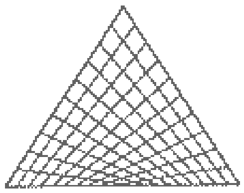
3.



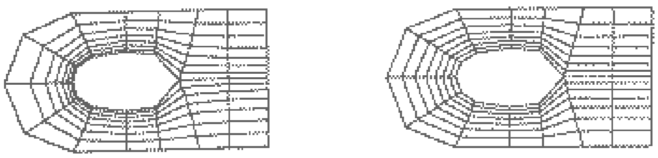
4.



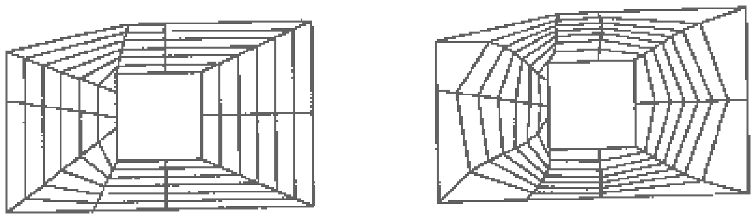
5.



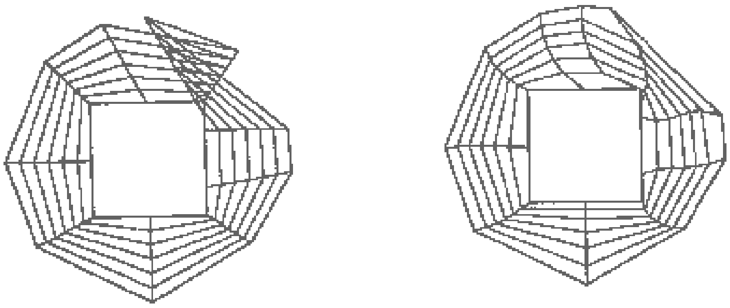
6.



7.



8.



Attraction