```
c ***********************************************
                         PROGRAM ZAMS
c ***********************************************
c This program constructs homogeneous zero age main sequence
c models using the ``fitting'' technique outlined in
c Chapter 7.
c The input quantities are X and Y (the hydrogen and helium
c mass fractions), the model mass (in Solar units), and your
c guesses for central pressure, central temperature, total
c radius, and total luminosity.
c You may have to change the input and output routines
c to suit your machine and compiler. Otherwise, the language
c used here is standard.
c ***********************************************
c The independent variable is xi=ln(1-Mr/M) and the dependent
c variables are ln P, ln T ln r, and ln L (in cgs units).
c There are 201 points in the model and the mesh in xi is
c determined for you. The fitting point is also chosen
c for you because convergence may depend on where it is set.
c The fitting point is specified by ``QFIT'' which is the
c fractional mass interior to the fitting point.
c ***********************************************
c Note that bad guesses may can be fatal.
c ***********************************************

      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      LOGICAL IDIDIT, IPULS
      CHARACTER*20 OUTFILE, OUTPULS
c
      PARAMETER (NFILE=10, NPULS=11)
c
c NFILE is the unit number for model output and
c NPULS is the unit number for output to be used
c by the pulsation codes.
c
      COMMON  AMASS, AMTOT, AL, QFIT, X, Y, Z
      COMMON /PULS/ OUTPULS, IPULS
c
c First executable statement. Call the input routine
c after setting the number of mesh points (N).
c
      N=201
c
      CALL INPUT (NTRY, N, NFOUT, NFIN, NFILE, NPULS,
     *    OUTFILE)
      IDIDIT=.FALSE.
c
c If IDIDIT is .FALSE. then the model is not converged.
c You have NTRY's to converge to within VERG=5.d-3
c tolerance of the corrections to the logarithms of
c central pressure and temperature and for total radius
c and luminosity. If the model is converged, then one more
c pass is made to improve the result.
c
      DO 1234 ITRY=1,NTRY
c Integrate outwards.
         CALL GOOUT (NFOUT, 1)
c Integrate inward.
         CALL GOIN (N, NFIN, 1)
c Get the corrections and check for convergence.
         CALL CORRECT (ITRY, NFOUT, NFIN, IDIDIT, NFILE)
c Check to see if converged.
```



```
         IF (.NOT. IDIDIT)  GOTO 1234
c The calculation converged so give it one more shot.
         CALL GOOUT (NFOUT, 2)
         CALL GOIN (N, NFIN, 2)
         WRITE (6,1001) OUTFILE
 1001    FORMAT (' MODEL OUTPUT WRITTEN TO FILE ',A20)
         IF (IPULS) WRITE (6,1002) OUTPULS
 1002    FORMAT (' PULSATION OUTPUT WRITTEN TO ',A20)
c Call the output routine.
         CALL OUTPUT (N, NFOUT, NFIN, NFILE, NPULS)
         STOP 'HOORAY, I CAN STOP NOW!'
c Otherwise try to continue to converge.
 1234 CONTINUE
c If you reach here, you have not been successful in
c converging after NTRY's and the program will stop
c with a message.
      WRITE (6,1000)
 1000 FORMAT (' MODEL NOT CONVERGED.  BETTER LUCK NEXT TIME.')
c We're all done ... for better or for worse.
      STOP 'OH WELL, SUCH IS LIFE.'
      END
c ***********************************************
      SUBROUTINE INPUT (NTRY, N, NFOUT, NFIN, NFILE, NPULS,
     *    OUTFILE)
c ***********************************************
c This reads in the input: Pc, Tc, R, L (in Solar units),
c the total mass M (in Solar units), and X and Y.
c QFIT is calculated for you and NTRY (the maximum
c number of iterations) is set to 15.
c Feel free to change the latter.
c NFOUT is the mesh point of the fitting point.
c the maximum allowed number of iterations.
c ***********************************************
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      CHARACTER*20 OUTFILE, OUTPULS, YORN
      COMMON /BUCKET/ XI(201), DXI(201), YOUT(4,201),
     *   YIN(4,201), DYFIT(4), PARTIAL(4,4), AMR(201),
     *   RHO(201), OP(201), ALCONV(201), EPSGEN(201),
     *   DEL(201), DELAD(201), DELRAD(201)
      PARAMETER (AMSUN=1.989D33, ALSUN=3.847D33)
      COMMON  AMASS, AMTOT, AL, QFIT, X, Y, Z
      LOGICAL IPULS
      COMMON /PULS/ OUTPULS, IPULS
c
      WRITE (6,1004)
 1004 FORMAT (' THE TOTAL MASS IS (IN MSUN) ')
      READ (5,*) AMASS
      WRITE (6,1005)
 1005 FORMAT (' X(Hydrogen) AND Y(Helium) ARE ')
      READ (5,*) X, Y
      WRITE (6,1000)
 1000 FORMAT (' GUESS FOR CENTRAL PRESSURE (IN CGS)')
      READ (5,*) PC
      WRITE (6,1001)
 1001 FORMAT (' GUESS FOR CENTRAL TEMPERATURE  (IN K)')
      READ (5,*) TC
      WRITE (6,1002)
 1002 FORMAT (' GUESS FOR TOTAL RADIUS  (IN CM)')
      READ (5,*) R
      WRITE (6,1003)
 1003 FORMAT (' GUESS FOR TOTAL LUMINOSITY (IN LSUN)')
      READ (5,*) AL
```

```
c Determine QFIT and set NTRY.
      NTRY=15
      TEMP1=DLOG10(1.5D0)
      TEMP2=0.6D0/(1.0D0-TEMP1)
      IF (AMASS .GE. 10.0D0) THEN
          QFIT=0.8D0
        ELSEIF (AMASS .LE. 1.5D0) THEN
          QFIT=0.2D0
        ELSE
          QFIT=0.2D0+(DLOG10(AMASS)-TEMP1)*TEMP2
      END IF
c We now ask you for the name of the file to where the
c output should (eventually) be sent. Note that the
c STATUS of this file is set to 'UNKNOWN' in the OPEN
c statement.
      WRITE (6,1006)
 1006 FORMAT (' WHAT IS YOUR OUTPUT FILE NAME? ')
      READ (5,1007) OUTFILE
 1007 FORMAT (A20)
      OPEN (NFILE, FILE=OUTFILE, STATUS='UNKNOWN')
c Asks whether you wish output to be written that is
c to be used for pulsation calculations.
      WRITE (6,1008)
 1008 FORMAT (' DO YOU WANT PULSATION OUTPUT? (Y/N)')
      READ (5,1007) YORN
      IF ((YORN .EQ. 'y') .OR. (YORN .EQ. 'Y')) THEN
        IPULS=.TRUE.
        WRITE (6,1009)
 1009   FORMAT (' PULSATION FILENAME IS ')
        READ (5,1007) OUTPULS
        OPEN (NPULS, FILE=OUTPULS, STATUS='UNKNOWN')
        WRITE (NPULS,*) AMASS
        WRITE (NPULS,*) X,Y
      ELSE
        IPULS=.FALSE.
      ENDIF
c Convert to logs and cgs units.
      Z=1.0D0-X-Y
      YOUT(1,1)=DLOG(PC)
      YOUT(2,1)=DLOG(TC)
      YOUT(3,1)=0.0D0
      YOUT(4,1)=0.0D0
      YIN(1,1)=0.0D0
      YIN(2,1)=0.0D0
      YIN(3,1)=DLOG(R)
      YIN(4,1)=DLOG(AL*ALSUN)
      AMTOT=AMASS*AMSUN
c This the total mass in c.g.s.
c Set up XI and DXI arrays. The DXI are set up as
c two geometric series. XI(N) is undefined. XI(N-1) is
c at the mass 1-Mr/M=1.d-10. There is no simple way
c to set up the zoning so as to satisfy all kinds of
c models and pulsation codes. A logarithmic mesh
c of this sort is discussed in Kippenhahn, Weigert,
c and Hofmeister (1967) (see chapter 7).
      NFOUT=0
      NC=185
      EPS1=0.035D0
      EPS1P1=EPS1+1.0D0
      EPS2=-0.15D0
      EPS2P1=EPS2+1.0D0
      N1=N-1
```



```
      N2=N-2
      XI(N1)=DLOG(1.D-10)
      TEMP1=(EPS1P1**(NC-1)-1.0D0)/EPS1
      TEMP2=EPS1P1**(NC-2)*EPS2P1*
     *    (EPS2P1**(N1-NC)-1.0D0)/EPS2
      DXI(1)=XI(N1)/(TEMP1+TEMP2)
      DO 1 I=2,NC-1
         TEMP=EPS1P1**(I-1)
         DXI(I)=DXI(1)*TEMP
         XI(I)=DXI(1)*(TEMP-1.0D0)/EPS1
    1 CONTINUE
      XI(NC)=XI(NC-1)+DXI(NC-1)
      DO 2 I=NC,N-3
         J=I-NC+1
         DXI(I)=DXI(NC-1)*EPS2P1**J
         TEMP=EPS2P1*(EPS2P1**J-1.0D0)/EPS2
         XI(I+1)=XI(NC)+DXI(NC-1)*TEMP
    2 CONTINUE
      DXI(N-2)=EPS2P1**(N-NC-1)
      DO 3 I=2,N1
         IF (XI(I).LT.DLOG(1.D0-QFIT).AND.NFOUT.EQ.0) THEN
            NFOUT=I
            GOTO 4
         ENDIF
    3 CONTINUE
    4 WRITE (6,*) ' NFOUT=',NFOUT
c These are the interior masses.
      AMR(1)=0.0D0
      AMR(N)=1.0D0
      DO 5 I=2,N1
         AMR(I)=AMTOT*(1.0D0-DEXP(XI(I)))
    5 CONTINUE
      NFIN=N-NFOUT+1
c NFIN is the mesh number of the fitting point as measured
c from the surface.
      WRITE (NFILE,2000) AMASS, X, Y
 2000 FORMAT (' MASS/MSUN=', 0PF7.3, ',X=',F6.3,
     *    ',Y=',F6.3)
      WRITE (NFILE,998) NTRY, N, NFOUT, NFIN, QFIT
 998  FORMAT (' NTRY',I3,',N=',I3,',NFOUT=',I3,',NFIN=',I3,
     *    ',QFIT=',F6.3)
      WRITE (NFILE,2001) PC, TC
 2001 FORMAT (' GUESSES: Pc=',1PD10.3,',Tc=', D10.3)
      WRITE (NFILE,2002) R, AL
 2002 FORMAT ('     R=', 1PD10.3, ',L/LSUN=', D10.3)
      RETURN
      END
c ***********************************************
        SUBROUTINE OUTPUT(N, NFOUT, NFIN, NFILE, NPULS)
c ***********************************************
c This writes the output if you have succeeded in
c converging. The model output is written to "outfile"
c which was designated in SUBROUTINE INPUT. If
c you wanted pulsation output, it is written to
c "outpuls".
c ***********************************************
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      COMMON /BUCKET/ XI(201), DXI(201), YOUT(4,201),
     *   YIN(4,201), DYFIT(4), PARTIAL(4,4), AMR(201),
     *   RHO(201), OP(201), ALCONV(201), EPSGEN(201),
     *   DEL(201), DELAD(201), DELRAD(201)
      DIMENSION TOTLUM(201), RPULS(201)
```

```
      LOGICAL IPULS
      CHARACTER*20 OUTPULS
      COMMON /PULS/ OUTPULS, IPULS
      COMMON /PULSOUT/ XPULS(201), GPULS(201), SL2(201),
     *  BV2(201), UPULS(201), VPULS(201), CHIT(201),
     *  CHIRHO(201), G3(201)
      PARAMETER (PSIG4=7.1246D-4, GRAV=6.6726D-8,
     *   P4=12.56637062D0, ALSUN=3.847D33)
      COMMON  AMASS, AMTOT, AL, QFIT, X, Y, Z
c
c First see if data is to be written to a pulsation
c file for use in the pulsation code.
c
      ECONV=DLOG(10.0D0)
      IF (.NOT. IPULS) GOTO 2000
      WRITE (NPULS,*) N-2
      DO 2001 I=2,NFOUT
         RPULS(I)=DEXP(YOUT(3,I))
         R2=RPULS(I)**2
         R3=R2*RPULS(I)
         GPULS(I)=GRAV*AMR(I)/R2
         GAMMA1=CHIT(I)*G3(I)+CHIRHO(I)
         PSCALE=DEXP(YOUT(1,I))/GPULS(I)/RHO(I)
         XPULS(I)=YOUT(3,I)-YOUT(1,I)
         UPULS(I)=P4*R3*RHO(I)/AMR(I)
         VPULS(I)=RPULS(I)/PSCALE
         VPULS(I)=1.0D0/(1.0D0+VPULS(I))
         SL2(I)=GPULS(I)*PSCALE*GAMMA1/R2
         BV2(I)=-CHIT(I)*GPULS(I)*(DEL(I)-DELAD(I))
     *       /CHIRHO(I)/PSCALE
         WRITE (NPULS,3000) XPULS(I),RPULS(I),
     *    GPULS(I),RHO(I)
 3000    FORMAT (1P4D14.6)
 2001 CONTINUE
      DO 2002 I=2,NFIN-1
         NIN=NFIN-I+1
         NOUT=NFOUT+I-1
         RPULS(NOUT)=DEXP(YIN(3,NIN))
         R2=RPULS(NOUT)**2
         R3=R2*RPULS(NOUT)
         GPULS(NOUT)=GRAV*AMR(NOUT)/R2
         GAMMA1=CHIT(NOUT)*G3(NOUT)+CHIRHO(NOUT)
         PSCALE=DEXP(YIN(1,NIN))/GPULS(NOUT)/RHO(NOUT)
         XPULS(NOUT)=YIN(3,NIN)-YIN(1,NIN)
         UPULS(NOUT)=P4*R3*RHO(NOUT)/AMR(NOUT)
         VPULS(NOUT)=RPULS(NOUT)/PSCALE
         VPULS(NOUT)=1.0D0/(1.0D0+VPULS(NOUT))
         SL2(NOUT)=GPULS(NOUT)*PSCALE*GAMMA1/R2
         BV2(NOUT)=-CHIT(NOUT)*GPULS(NOUT)*(DEL(NOUT)
     *     -DELAD(NOUT))/CHIRHO(NOUT)/PSCALE
         WRITE (NPULS,3000) XPULS(NOUT),RPULS(NOUT),
     *    GPULS(NOUT),RHO(NOUT)
 2002 CONTINUE
      WRITE (NPULS,*) N-2
      DO  2003 I=2,N-1
        WRITE (NPULS,3000) XPULS(I), GPULS(I)/RPULS(I),
     *  GPULS(I)/SL2(I)/RPULS(I), RPULS(I)*BV2(I)/GPULS(I)
        WRITE (NPULS,3001)   UPULS(I), VPULS(I)
 3001   FORMAT (1P2D14.6)
 2003 CONTINUE
c
c Now write out the ZAMS model to file.
```



```
c
 2000 WRITE (NFILE,1000)
 1000 FORMAT ('           *****FINAL MODEL*****')
      WRITE (NFILE,1002) DEXP(YOUT(1,1)), DEXP(YOUT(2,1))
     *    , DEXP(YIN(3,1)), DEXP(YIN(4,1))
 1002 FORMAT (' Pc:', 1PD11.4,', Tc:',D11.4,', R:',
     *   D11.4,', L:', D11.4)
      TEFF=DEXP(.25D0*YIN(4,1)-0.5D0*YIN(3,1))/
     *   PSIG4**0.25D0
      DLOGAL=DLOG10(DEXP(YIN(4,1))/ALSUN)
      WRITE (NFILE,1007)  TEFF, DLOG10(TEFF),DLOGAL
 1007 FORMAT (' Teff:',1PD12.4, ', LOG(Teff):',0PF7.4,
     *   ', LOG(L/LSUN):', F7.4)
      WRITE (NFILE,1003)
 1003 FORMAT ('       1-Mr/M      LOG(r)   LOG(P)',
     *    2X,'LOG(T)'
     *   1X, 'LOG(RHO) LOG(L)')
      DO 1 I=2,NFOUT
         TOTLUM(I)=DEXP(YOUT(4,I))
         DO 3 J=1,4
            YOUT(J,I)=YOUT(J,I)/ECONV
    3    CONTINUE
         WRITE (NFILE,1004) I,1.D0-AMR(I)/AMTOT,
     *      YOUT(3,I), YOUT(1,I), YOUT(2,I),
     *      DLOG10(RHO(I)), YOUT(4,I)
    1 CONTINUE
 1004 FORMAT (I5,1PD18.8, 0PF9.5, 4F8.4)
      DO 2 I=2,NFIN-1
         NIN=NFIN-I+1
         NOUT=NFOUT+I-1
         TOTLUM(NOUT)=DEXP(YIN(4,NIN))
         DO 4 J=1,4
            YIN(J,NIN)=YIN(J,NIN)/ECONV
    4    CONTINUE
         WRITE (NFILE,1004) NOUT,1.D0-AMR(NOUT)/AMTOT,
     *      YIN(3,NIN), YIN(1,NIN), YIN(2,NIN),
     *      DLOG10(RHO(NOUT)),
     *      YIN(4,NIN)
    2 CONTINUE
      WRITE (NFILE,1010)
 1010 FORMAT (' ****************************************')
      WRITE (NFILE,1005)
 1005 FORMAT ('   LOG(EPS) LOG(OP) LOG(Lc)',3X,
     *     'Lc/Ltot',2X,'DEL',5X,'DELAD',7X,'DELRAD')
      DO 5 I=2,N-1
         TOTLUM(I)=ALCONV(I)/TOTLUM(I)
         IF (ALCONV(I) .GT. 0.0D0) THEN
            ALCONV(I)=DLOG10(ALCONV(I))
         ENDIF
         WRITE (NFILE,1006)I,DLOG10(EPSGEN(I)),
     *      DLOG10(OP(I)), ALCONV(I), TOTLUM(I), DEL(I),
     *      DELAD(I),DELRAD(I)
    5 CONTINUE
 1006 FORMAT (I4, 4F8.4, 2F8.5, F14.4)
      RETURN
      END
c **********************************************
       SUBROUTINE GOOUT (NFOUT, NGO)
c **********************************************
c This is the driver routine for the outwards integration.
c It does a first step using the central expansions.
c There are three different integrations performed:
```

```
c (1) vary the central pressure from its guessed value;
c (2) vary the central temperature; (3) use the guessed
c values for both central pressure and temperature. All
c three are usually done (when NGO is 1) except when the
c problem has converged (NGO is 2). Only (3) is done in
c that case (to get a final model).
c **********************************************
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      COMMON /BUCKET/ XI(201), DXI(201), YOUT(4,201),
     *   YIN(4,201), DYFIT(4), PARTIAL(4,4), AMR(201),
     *   RHO(201), OP(201), ALCONV(201), EPSGEN(201),
     *   DEL(201), DELAD(201), DELRAD(201)
      COMMON /DENSITY/VGUESS
      DIMENSION TY(2), TYOUT(4,2), DYOUT(2), YGO(4),
     *   WORK(27), IWORK(5)
      PARAMETER (AMSUN=1.989D33, ALSUN=3.847D33,
     *   CHANGE=0.0002D0)
      PARAMETER (GRAV=6.67259D-8, P43=4.1887902,
     *   PG23=1.397504D-7, AC8=1.81456D-3,
     *   P4=12.5663706D0)
      COMMON  AMASS, AMTOT, AL, QFIT, X, Y, Z
      COMMON /PULSOUT/ XPULS(201), GPULS(201), SL2(201),
     *  BV2(201), UPULS(201), VPULS(201), CHIT(201),
     *  CHIRHO(201), G3(201)
      EXTERNAL DERIVS
c YOUT(N,I) is (at the Nth point): I=1, ln P; i=2, ln T;
c I=3, ln r; I=4, ln L.
      DO 1 I=1,2
         TY(I)=YOUT(I,1)
    1 CONTINUE
      GOTO (10, 100) NGO
c Succesively perturb Pc and Tc by CHANGE.
   10 DO 11 ICHANGE=1,2
         DYOUT(ICHANGE)=CHANGE*TY(ICHANGE)
         YOUT(ICHANGE,1)=TY(ICHANGE)+DYOUT(ICHANGE)
         CALL CENTRAL
         DO 12 IGO=2,NFOUT-1
            DO 13 I=1,4
               YGO(I)=YOUT(I,IGO)
   13       CONTINUE
            XSTART=XI(IGO)
            XFIN=XI(IGO+1)
            RELERR=5.D-9
            ABSERR=DABS(YGO(1))
            DO 20 I=2,4
               TEMP=DABS(YGO(I))
               ABSERR=DMIN1(ABSERR,TEMP)
   20       CONTINUE
            ABSERR=5.D-9*ABSERR
            ABSERR=DMAX1(ABSERR,1.0D-10)
            CALL RKF(DERIVS,4,YGO,XSTART,XFIN,RELERR,ABSERR,
     *         IFLAG,WORK,IWORK)
            IF (IFLAG.EQ.3.OR.IFLAG.EQ.4.OR.IFLAG.EQ.5)
     *         WRITE(6,1100) IFLAG,XFIN
            IFLAG=1
 1100 FORMAT (17H WATCH OUT,IFLAG=,I4,6H XFIN=,1PD10.2)
            DO 14 I=1,4
               YOUT(I,IGO+1)=YGO(I)
   14       CONTINUE
   12    CONTINUE
         DO 15 I=1,4
            TYOUT(I,ICHANGE)=YOUT(I,NFOUT)
```



```
   15    CONTINUE
         DO 16 I=1,2
            YOUT(I,1)=TY(I)
   16    CONTINUE
   11 CONTINUE
  100 DO 101 I=1,2
         YOUT(I,1)=TY(I)
  101 CONTINUE
c Now do the calculation with the original guesses at the
c center.
      CALL CENTRAL
      DO 102 IGO=2,NFOUT-1
         DO 103 I=1,4
            YGO(I)=YOUT(I,IGO)
  103    CONTINUE
         XSTART=XI(IGO)
         XFIN=XI(IGO+1)
            RELERR=5.D-9
            ABSERR=DABS(YGO(1))
            DO 120 I=2,4
               TEMP=DABS(YGO(I))
               ABSERR=DMIN1(ABSERR,TEMP)
  120       CONTINUE
            ABSERR=5.D-9*ABSERR
            ABSERR=DMAX1(ABSERR,1.0D-10)
            CALL RKF(DERIVS,4,YGO,XSTART,XFIN,RELERR,ABSERR,
     *         IFLAG,WORK,IWORK)
            IF (IFLAG.EQ.3.OR.IFLAG.EQ.4.OR.IFLAG.EQ.5)
     *         WRITE(6,1100) IFLAG,XFIN
            IFLAG=1
         DO 104 I=1,4
            YOUT(I,IGO+1)=YGO(I)
  104       CONTINUE
  102    CONTINUE
         GOTO (400,200) NGO
  400 DO 105 ICHANGE=1,2
         DO 106 I=1,4
            PARTIAL(I,ICHANGE)=(TYOUT(I,ICHANGE)
     *         -YOUT(I,NFOUT))/DYOUT(ICHANGE)
  106    CONTINUE
  105 CONTINUE
      GOTO 300
  200 VGUESS=8.3145D7*(3.D0+5.0D0*X)/4.0D0*
     *      DEXP(YOUT(2,2))/DEXP(YOUT(1,2))
      DO 201 IGO=2,NFOUT
         DO 202 I=1,4
            YGO(I)=YOUT(I,IGO)
  202    CONTINUE
c Compute the model quantities at the mesh points.
         CALL EOS (YGO, RHO(IGO), PE, DELAD(IGO),
     *      CHIT(IGO), CHIRHO(IGO), G3(IGO))
         CALL OPACITY (YGO, RHO(IGO), PE, OP(IGO))
         CALL EPSILON (YGO, RHO(IGO), EPSGEN(IGO))
c Convective or radiative?
         DELRAD(IGO)=DEXP(YGO(1)-4.D0*YGO(2)+YGO(4))*
     *      OP(IGO)/AMR(IGO)/AC8/PG23
         IF (DELRAD(IGO) .GT. DELAD(IGO)) THEN
            GLOCAL=GRAV*AMR(IGO)/DEXP(2.0D0*YGO(3))
c Find the actual del.
            CALL CONV(DELRAD(IGO),DELAD(IGO),YGO(1),YGO(2),
     *         RHO(IGO),OP(IGO),GLOCAL,DEL(IGO))
c Find the convective luminosity (if any).
```

```
          ALCONV(IGO)=DEXP(YGO(4))*(DELRAD(IGO)-DEL(IGO))/
     *        DELRAD(IGO)
         ELSE
          DEL(IGO)=DELRAD(IGO)
          ALCONV(IGO)=0.0D0
         ENDIF
  201 CONTINUE
  300 RETURN
      END
c ***********************************************
       SUBROUTINE GOIN (N, NFIN, NGO)
c This is the driver routine for the inward integrations.
c It starts at the mass point 1-Mr/M=1.d-10 which is taken
c as the photosphere. There are three different integrations
c performed: (1) vary the total radius keeping luminosity
c fixed; (2) vary the luminosity at fixed radius; (3) use
c the guessed values of R and LUM. All three are  done
c (when NGO is 1) except when the problem has converged.
c Only (3) is done in that case (to get the final model).
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      COMMON /BUCKET/ XI(201), DXI(201), YOUT(4,201),
     *   YIN(4,201), DYFIT(4), PARTIAL(4,4), AMR(201),
     *   RHO(201), OP(201), ALCONV(201), EPSGEN(201),
     *   DEL(201), DELAD(201), DELRAD(201)
      COMMON /DENSITY/VGUESS
      DIMENSION TY(2), TYIN(4,2), DYIN(2), YGO(4),
     *   WORK(27), IWORK(5)
      PARAMETER (AMSUN=1.989D33, ALSUN=3.86D33,
     *   CHANGE=0.0002)
      PARAMETER (GRAV=6.67259D-8, P43=4.1887902,
     *   PG23=1.397504D-7, AC8=1.81456D-3,
     *   P4=12.5663706D0)
      COMMON  AMASS, AMTOT, AL, QFIT, X, Y, Z
      COMMON /PULSOUT/ XPULS(201), GPULS(201), SL2(201),
     *  BV2(201), UPULS(201), VPULS(201), CHIT(201),
     *  CHIRHO(201), G3(201)
      EXTERNAL DERIVS
c
c YIN is the analogue of YOUT in GOOUT and it marches
c in from the surface.
      DO 1 I=1,2
         TY(I)=YIN(I+2,1)
    1 CONTINUE
      GOTO (10, 100) NGO
   10 DO 11 ICHANGE=1,2
         DYIN(ICHANGE)=CHANGE*TY(ICHANGE)
         YIN(ICHANGE+2,1)=TY(ICHANGE)+DYIN(ICHANGE)
         CALL SURFACE
         DO 12 IGO=2,NFIN-1
            DO 13 I=1,4
               YGO(I)=YIN(I,IGO)
   13       CONTINUE
            NIN=N-IGO+1
            XSTART=XI(NIN)
            XFIN=XI(NIN-1)
            RELERR=5.D-9
            ABSERR=DABS(YGO(1))
            DO 20 I=2,4
               TEMP=DABS(YGO(I))
               ABSERR=DMIN1(ABSERR,TEMP)
   20       CONTINUE
            ABSERR=5.D-9*ABSERR
```



```
            ABSERR=DMAX1(ABSERR,1.0D-10)
            CALL RKF(DERIVS,4,YGO,XSTART,XFIN,RELERR,ABSERR,
     *         IFLAG,WORK,IWORK)
            IF (IFLAG.EQ.3.OR.IFLAG.EQ.4.OR.IFLAG.EQ.5)
     *         WRITE(6,1100) IFLAG,XFIN
            IFLAG=1
            DO 14 I=1,4
               YIN(I,IGO+1)=YGO(I)
   14       CONTINUE
   12    CONTINUE
         DO 15 I=1,4
            TYIN(I,ICHANGE)=YIN(I,NFIN)
   15    CONTINUE
         DO 16 I=1,2
            YIN(I+2,1)=TY(I)
   16    CONTINUE
   11 CONTINUE
  100 DO 101 I=1,2
         YIN(I+2,1)=TY(I)
  101 CONTINUE
c Now do the calculation with the original guesses at the
c surface.
      CALL SURFACE
      DO 102 IGO=2,NFIN-1
         DO 103 I=1,4
            YGO(I)=YIN(I,IGO)
  103    CONTINUE
         NIN=N-IGO+1
         XSTART=XI(NIN)
         XFIN=XI(NIN-1)
            RELERR=5.D-9
            ABSERR=DABS(YGO(1))
            DO 120 I=2,4
               TEMP=DABS(YGO(I))
               ABSERR=DMIN1(ABSERR,TEMP)
  120       CONTINUE
            ABSERR=5.D-9*ABSERR
            ABSERR=DMAX1(ABSERR,1.0D-10)
            CALL RKF(DERIVS,4,YGO,XSTART,XFIN,RELERR,ABSERR,
     *         IFLAG,WORK,IWORK)
            IF (IFLAG.EQ.3.OR.IFLAG.EQ.4.OR.IFLAG.EQ.5)
     *         WRITE(6,1100) IFLAG,XFIN
 1100 FORMAT (17H WATCH OUT,IFLAG=,I4,6H XFIN=,1PD10.2)
            IFLAG=1
         DO 104 I=1,4
            YIN(I,IGO+1)=YGO(I)
  104       CONTINUE
  102    CONTINUE
         GOTO (400,200) NGO
  400 DO 105 ICHANGE=1,2
         DO 106 I=1,4
            PARTIAL(I,ICHANGE+2)=-(TYIN(I,ICHANGE)
     *         -YIN(I,NFIN))/DYIN(ICHANGE)
  106    CONTINUE
  105 CONTINUE
      GOTO 300
  200 VGUESS=8.3145D7*(3.D0+5.0D0*X)/4.0D0*
     *      DEXP(YIN(2,2))/DEXP(YIN(1,2))
      DO 201 IGO=2,NFIN-1
         DO 202 I=1,4
            YGO(I)=YIN(I,IGO)
  202    CONTINUE
```

```
      NIN=N-IGO+1
c Compute the model quantities at the mesh points.)
         CALL EOS (YGO, RHO(NIN), PE, DELAD(NIN),
     *      CHIT(NIN), CHIRHO(NIN), G3(NIN))
         CALL OPACITY (YGO, RHO(NIN), PE, OP(NIN))
         CALL EPSILON (YGO, RHO(NIN), EPSGEN(NIN))
c Convective or radiative?
         DELRAD(NIN)=DEXP(YGO(1)-4.D0*YGO(2)+YGO(4))*
     *      OP(NIN)/AMR(NIN)/AC8/PG23
         IF (DELRAD(NIN) .GT. DELAD(NIN)) THEN
            GLOCAL=GRAV*AMR(NIN)/DEXP(2.0D0*YGO(3))
            CALL CONV(DELRAD(NIN),DELAD(NIN),YGO(1),YGO(2),
     *         RHO(NIN),OP(NIN),GLOCAL,DEL(NIN))
c Find the convective luminosity (if any).
            ALCONV(NIN)=DEXP(YGO(4))*(DELRAD(NIN)-DEL(NIN))/
     *         DELRAD(NIN)
         ELSE
            DEL(NIN)=DELRAD(NIN)
            ALCONV(NIN)=0.0D0
         ENDIF
  201 CONTINUE
  300 RETURN
      END
c ***********************************************
c ***********************************************
      SUBROUTINE CORRECT (ITRY, NFOUT, NFIN, IDIDIT,
     *   NFILE)
c ***********************************************
c This routine computes the changes in Pc, Tc, R, and L to
c head towards convergence. It also checks for convergence.
c If converged, then IDIDIT is set to .TRUE. .
c The method is to set up four linear equations
c containing information about how the four
c parameters change across the fitting point as the
c central and surface values are independently varied.
c ***********************************************
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      COMMON /BUCKET/ XI(201), DXI(201), YOUT(4,201),
     *   YIN(4,201), DYFIT(4), PARTIAL(4,4), AMR(201),
     *   RHO(201), OP(201), ALCONV(201), EPSGEN(201),
     *   DEL(201), DELAD(201), DELRAD(201)
      LOGICAL IDIDIT
      DIMENSION YTEMP(4),IPVT(4)
      PARAMETER (AMSUN=1.989D33, ALSUN=3.86D33, VERG=5.D-3)
      COMMON  AMASS, AMTOT, AL, QFIT, X, Y, Z
c
      DO 1 I=1,4
         DYFIT(I)=YIN(I,NFIN)-YOUT(I,NFOUT)
    1 CONTINUE
      DO 2 I=1,2
         YTEMP(I)=YOUT(I,1)
    2 CONTINUE
      DO 3 I=3,4
         YTEMP(I)=YIN(I,1)
    3 CONTINUE
      WRITE (NFILE,1000) ITRY
 1000 FORMAT (' ****ITERATION NUMBER = ', I4)
      WRITE (NFILE,1005)
      WRITE (6,1005)
 1005 FORMAT (' MISMATCH AT FITTING POINT')
      WRITE (NFILE,1001) (DYFIT(I), I=1,4)
      WRITE (6,1001) (DYFIT(I), I=1,4)
```



```
 1001 FORMAT (1P4D15.6)
c Call the LINPACK routines to get corrections.
      CALL DGEFA (PARTIAL, 4, 4, IPVT, INFO)
      IF (INFO .NE. 0) THEN
        STOP ' MATRIX CONTAINS ZERO ON DIAGONAL. ENDING CALCULATION'
      ENDIF
      CALL DGESL (PARTIAL, 4, 4, IPVT, DYFIT, 0)
      WRITE (NFILE,1002) ITRY
      WRITE (6,1002) ITRY
 1002 FORMAT (' Pc, Tc, R, L, FOR ITRY = ',I4)
      WRITE (NFILE,1001) YOUT(1,1), YOUT(2,1), YIN(3,1),
     *   YIN(4,1)
      WRITE (6,1001) YOUT(1,1), YOUT(2,1), YIN(3,1),
     *   YIN(4,1)
      WRITE (NFILE,1004) ITRY
      WRITE (6,1004) ITRY
 1004 FORMAT (' RAW CORRECTIONS FOR ITRY = ', I4)
      WRITE (NFILE,1001) (DYFIT(I), I=1,4)
      WRITE (6,1001) (DYFIT(I), I=1,4)
c Test if converged.
      DO 4 I=1,4
         IF(DABS(DYFIT(I)) .GT. VERG) GOTO 6
    4 CONTINUE
c Converged.
      IDIDIT=.TRUE.
   11 DO 7 I=1,2
         YOUT(I,1)=YTEMP(I)+DYFIT(I)
    7 CONTINUE
      DO 8 I=3,4
         YIN(I,1)=YTEMP(I)+DYFIT(I)
    8 CONTINUE
      WRITE (NFILE,1006) ITRY
 1006 FORMAT (' APPLIED CORRECTIONS FOR ITRY =',I4)
      WRITE (NFILE,1001) (DYFIT(I),I=1,4)
      WRITE (NFILE,1007)
 1007 FORMAT (' GUESSES FOR NEXT ITERATION')
      WRITE (NFILE,1001) YOUT(1,1),YOUT(2,1),YIN(3,1),
     *   YIN(4,1)
      RETURN
    6  DO 9 I=1,4
         IF (DABS(DYFIT(I)) .GT. 0.10D0) GOTO 30
    9 CONTINUE
      GOTO 11
   30 AMAX=DABS(DYFIT(1))
      DO 31 I=2,4
         AMAX=DMAX1(AMAX,DABS(DYFIT(I)))
   31 CONTINUE
      DO 32 I=1,4
         DYFIT(I)=0.10D0*DYFIT(I)/AMAX
   32 CONTINUE
      GOTO 11
      END
c ***********************************************
c ***********************************************
          SUBROUTINE CENTRAL
c ***********************************************
c This routine uses the expansions at the center to
c find the variables at the first point away  from
c the center. (See Sec. 7.3.1.)
c ***********************************************
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      COMMON /BUCKET/ XI(201), DXI(201), YOUT(4,201),
```

```
     *   YIN(4,201), DYFIT(4), PARTIAL(4,4), AMR(201),
     *   RHO(201), OP(201), ALCONV(201), EPSGEN(201),
     *   DEL(201), DELAD(201), DELRAD(201)
      DIMENSION YGO(4)
      COMMON /DENSITY/VGUESS
      PARAMETER (GRAV=6.67259D-8, P43=4.1887902,
     *   PG23=1.397504D-7, AC8=1.81456D-3)
c GRAV=G, P43=4PI/3, PG23=2 PI*G/3. AC8=8ac
      COMMON /PULSOUT/ XPULS(201), GPULS(201), SL2(201),
     *  BV2(201), UPULS(201), VPULS(201), CHIT(201),
     *  CHIRHO(201), G3(201)
      COMMON  AMASS, AMTOT, AL, QFIT, X, Y, Z
c
      DO 1 I=1,4
         YGO(I)=YOUT(I,1)
    1 CONTINUE
c First find the central density. Make a guess.
         VGUESS=8.3145D7*(3.D0+5.0D0*X)/4.0D0*
     *      DEXP(YOUT(2,1))/DEXP(YOUT(1,1))
      CALL EOS (YGO, RHO(1), PE, DELAD(1), CHIT(1),
     *  CHIRHO(1), G3(1))
c Find the next radius point.
      YOUT(3,2)=DLOG(AMR(2)/P43/RHO(1))/3.0D0
c Compute the pressure at the next point.
      DP=PG23*RHO(1)*RHO(1)*DEXP(2.0D0*YOUT(3,2))
      YOUT(1,2)=DLOG(DEXP(YOUT(1,1))-DP)
      CALL OPACITY (YGO, RHO(1), PE, OP(1))
      CALL EPSILON (YGO, RHO(1), EPSGEN(1))
c Find the luminosity at the next point.
      YOUT(4,2)=3.0D0*YOUT(3,2)+DLOG(P43*RHO(1)*EPSGEN(1))
c Establish whether the center is radiative or convective.
      DELRAD(1)=DEXP(YOUT(1,1)-4.0D0*YOUT(2,1))*OP(1)*
     *   EPSGEN(1)/AC8/PG23
c DEL is the lesser of DELAD and DELAD.
      DEL(1)=DMIN1(DELAD(1),DELRAD(1))
c Find the temperature at the next point.
      DT=PG23*DEL(1)*RHO(1)*RHO(1)*DEXP(YOUT(2,1)+2.0D0*
     *   YOUT(3,2)-YOUT(1,1))
      YOUT(2,2)=DLOG(DEXP(YOUT(2,1))-DT)
c We have all the YOUT at the point once removed
c from the center.
      RETURN
      END
c ***********************************************
c ***********************************************
       SUBROUTINE SURFACE
c ***********************************************
c This gets us to the first mass point from the surface.
c To make things very simple, we assume that the
c mass point 1-Mr/M=10**(-10) is at the photosphere
c and that photospheric boundary conditions are OK.
c ***********************************************
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION YGO(4)
      COMMON /BUCKET/ XI(201), DXI(201), YOUT(4,201),
     *   YIN(4,201), DYFIT(4), PARTIAL(4,4), AMR(201),
     *   RHO(201), OP(201), ALCONV(201), EPSGEN(201),
     *   DEL(201), DELAD(201), DELRAD(201)
      COMMON  AMASS, AMTOT, AL, QFIT, X, Y, Z
      COMMON /DENSITY/VGUESS
      PARAMETER (GRAV=6.67259D-8, PSIG4=7.1246D-4,
     *   R=8.314D7)
```



```
c
      YIN(3,2)=YIN(3,1)
      YIN(4,2)=YIN(4,1)
      YGO(1)=1.0
      YGO(3)=YIN(3,2)
      YGO(4)=YIN(4,2)
      G=GRAV*AMTOT/DEXP(2.D0*YIN(3,2))
      TEFF=DEXP(.25D0*YIN(4,1)-0.5D0*YIN(3,1))/
     *   PSIG4**0.25D0
      YIN(2,2)=DLOG(TEFF)
      YGO(2)=YIN(2,2)
c We have to find the photospheric density and pressure
c given Teff and gravity. Do by iteration (Newton's method).
c Set part of a Kramers' b-f + f-f opacity as first guess.
      AKAPPA=(4.0D25*Z+4.0D22*(X+Y))*(1.0D0+X)*TEFF**(-3.5)
      AMU=4.0D0/(1.0D0+3.0D0*X)
      RHO1=DSQRT(2.0D0*G*AMU/R/AKAPPA/TEFF/3.0D0)
      TEMP=2.D0*G/3.D0
      DO 1 I=1,100
         DRHO=0.01D0*RHO1
         V1=1.0D0/RHO1
         RHO2=RHO1+DRHO
         V2=1.0D0/RHO2
         CALL GETEOS(TEFF,V1,P1,PE1,DUM1,DUM2,DUM3)
         CALL GETEOS(TEFF,V2,P2,PE2,DUM1,DUM2,DUM3)
         CALL OPACITY(YGO,RHO1,PE1,OP1)
         CALL OPACITY(YGO,RHO2,PE2,OP2)
         DOPDRHO=(OP2-OP1)/DRHO
         DPDRHO=(P2-P1)/DRHO
         DELRHO=(TEMP/OP1-P1)/(DPDRHO+TEMP*DOPDRHO/OP1**2)
         IF (DABS(DELRHO/RHO1) .GT. 0.2D0)
     *      DELRHO=0.2D0*DSIGN(RHO1,DELRHO)
         RHO1=RHO1+DELRHO
         V1=1.0D0/RHO1
         IF (DABS(DELRHO/RHO1) .LT. 1.0D-7) THEN
            CALL GETEOS(TEFF,V1,P1,PE1,DUM1,DUM2,DUM3)
            YIN(1,2)=DLOG(P1)
            VGUESS=V1
            RETURN
         ENDIF
    1 CONTINUE
      WRITE (6,1000)
 1000 FORMAT (' PHOTOSPHERIC CALCULATION OF DENSITY DIED')
      STOP 'SORRY'
      END
c ***********************************************
c ***********************************************
        SUBROUTINE DERIVS (X, Y, DYDX)
c ***********************************************
c This supplies the derivatives for the Runge-Kutta
c integrator.
c ***********************************************

      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION Y(4), DYDX(4)
      COMMON  AMASS, AMTOT, AL, QFIT, XTEMP, YTEMP, Z
      PARAMETER (GRAV=6.67259D-8, P43=4.1887902,
     *   PG23=1.397504D-7, AC8=1.81456D-3,
     *   P4=12.5663706D0)
c
      CALL EOS (Y, RHO, PE, DELAD, CHIT, CHIRHO, G3)
      CALL OPACITY (Y, RHO, PE, OP)
```

```
      CALL EPSILON (Y, RHO, EPS)
c This is e to the xi (x).
      EXI=DEXP(X)
c This is M times exi.
      AMXI=AMTOT*EXI
c This is Mr at x.
      AMR=AMTOT-AMXI
c Convective or radiative?
      DELRAD=DEXP(Y(1)-4.D0*Y(2)+Y(4))*OP/AMR/AC8/PG23
         IF (DELRAD .GT. DELAD) THEN
            GLOCAL=GRAV*AMR/DEXP(2.0D0*Y(3))
            CALL CONV(DELRAD,DELAD,Y(1),Y(2),
     *           RHO,OP,GLOCAL,DEL)
         ELSE
            DEL=DELRAD
         ENDIF
      DYDX(1)=AMXI*GRAV*AMR/P4/DEXP(4.0D0*Y(3)+Y(1))
      DYDX(2)=DYDX(1)*DEL
      DYDX(3)=-AMXI/P4/RHO/DEXP(3.0D0*Y(3))
      DYDX(4)=-AMXI*EPS/DEXP(Y(4))
      RETURN
      END
c ***********************************************
C ***********************************************
       SUBROUTINE EOS (YGO, RHO, PE, DELAD, CHIT,
     *    CHIRHO, G3)
C
C FINDS RHO GIVEN PRESSURE AND TEMPERATURE. X AND Y
C ARE PASSED THROUGH COMMON. HYDROGEN IS ASSUMED TO HAVE
C ONE STATE WHEREAS HELIUM HAS TWO STATES (TWO ELECTRONS
C IN GROUND.) THE ROUTINE PASSES THE ELCTRON PRESSURE
C (TO BE USED IN THE OPACITY) AND DELAD. THE EQUATION
C OF STATE IS THAT OF AN IDEAL MONATOMIC GAS PLUS
C RADIATION. IONIZATION EFFECTS ARE INCLUDED IN
C THE CALCULATION OF THE THERMODYNAMIC DERIVATIVES.
C ***********************************************
C THE PRIMARY INARDS OF THIS ROUTINE ARE DUE TO
C W. D. PESNELL ... WHOM WE THANK.
c ***********************************************
c
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION  YGO(4)
      COMMON  AMASS, AMTOT, AL, QFIT, X, Y, Z
      PARAMETER (ACCUR=1.0D-6)
      COMMON /DENSITY/VGUESS
      PWANT=YGO(1)
      T=DEXP(YGO(2))
      DO 1 ITRY=1,50
         CALL GETEOS (T, VGUESS, P, PE, PV, PT, CV)
         PV=VGUESS*PV/P
         FP=DLOG(P)-PWANT
         DELV=-FP/PV
         IF (DABS(DELV) .GT. 0.30) DELV=0.30*DELV/DABS(DELV)
         VGUESS=DLOG(VGUESS)+DELV
         VGUESS=DEXP(VGUESS)

         IF (DABS(DELV) .LT. ACCUR) THEN
            CALL GETEOS (T, VGUESS, P, PE, PV, PT, CV)
            PWANT=DEXP(YGO(1))
            FP=P-PWANT
            DELV=-FP/PV
            VGUESS=VGUESS+DELV
```



```
            RHO=1.0D0/VGUESS
c These are chit, chirho, and Gamma3-1.
            CHIT=PT*T/PWANT
            CHIRHO=-PV/(PWANT*RHO)
            G3=PWANT*CHIT/(RHO*T*CV)
            DELAD=CHIT+CHIRHO/G3
            DELAD=1.0D0/DELAD
            RETURN
         ENDIF
    1 CONTINUE
      WRITE (6,1000) T, DEXP(PWANT),X,Y
 1000 FORMAT (' EOS DID NOT CONVERGE. UGH!. T=',
     *   1PD11.3,' P=', D11.3/' X=',0pf7.3,' Y=',f7.3)
      WRITE (6,1001)
 1001 FORMAT (' CALCULATION STOPPED')
      STOP 'SORRY'
      END
c ***********************************************
      SUBROUTINE GETEOS (TIN,VIN,P,PE,PV,PT,ET)
C ***********************************************
c This routine and the other EOS routines are due
c to W. Dean Pesnell.
C
C     EQUATION OF STATE
C     ARGUMENTS...TIN (DEGREES K), VIN=1/RHO (CM**3/GM)
C     METALS...
C        NA,AL ALWAYS IONIZED
C        MG,SI,FE INCLUDED AS SINGLE ELEMENT
C        ALL OTHERS IGNORED
C
C          TABLE OF RETURNED QUANTITIES.
C
C      FUNCTION       NAME   DERIVATIVE WITH RESPECT TO
C                                 TEMP.       SP. VOL.
C-------------------------------------------------------
C      PRESSURE     I    P I     PT      I     PV      I
C      INT. ENERGY  I    E I     ET      I     EV      I
C      ELEC. PRS.   I   PE I     PET     I     PEV     I
C      MOLAR DENSITYI   BP I     BPT     I     BPV     I
C-------------------------------------------------------
C
C
C         X = HYDROGEN MASS FRACTION
C         Y = HELIUM MASS FRACTION
C         Z = METALLIC MASS FRACTION
C         PARGRM = MEAN MOLECULAR MOLAR DENSITY WITHOUT
C                  ELECTRONS
C
C         R = GAS CONSTANT 8.31434E7
C         A = STEFAN-BOLTZMAN CONSTANT 7.56471E-15
C         BK = BOLTZMAN S CONSTANT 8.6170837E-5
C         AVAGD = AVAGADRO S NUMBER 6.02217E23
C         AD3 = A/3
C
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      COMMON  DUM1, DUM2, DUM3, DUM4, X, Y, Z
      PARAMETER (ZERO = 0.D0,ONE = 1.D0, TWO=2.D0,
     *   THRE=3.D0, FOR=4.D0, TEN=10.D0, AHF=0.5D0,
     *   QRT=0.25D0 )
      PARAMETER ( R = 8.31434D7, A = 7.56471D-15,
     *   BK = 8.6170837D-5, AVAGD = 6.02217D23,
     *   AD3 = A/3.D0 )
```

```
      DATA T3OUT,T4OUT/1.665795163D-25,3.802592017D-28/
      DATA T2OUT,T5OUT/ 5.347896D-35,6.614536D-34/
C
C          IONIZATION POTENTIALS FOR HYDROGEN AND HELIUM
C
      DATA XH,XHE,XHE2/13.595D0,24.581D0,54.403D0/
      DATA C1,C2,C3/4.0092926D-9,1.00797D0,4.0026D0/
      DATA XM,CM,ZPZP/7.9D0,0.7D0,0.12014D0/
      DATA PREC / 1.D-10 /
      DATA ONHLF/1.5D0/
      V = VIN
      T = TIN
      IF( V .LE. ZERO ) GO TO 11
      IF( T .LE. ZERO ) GO TO 10
      FRE = ZERO
      ENT = ZERO
      PARGRM = X/C2 + Y/C3
      RMUC = ONE/PARGRM
      RT = R*T
      TT4 = T**4
      TK = ONE/(T*BK)
      SQT = DSQRT(T)
C    C1=ORIGINAL(C1(0.33334622))/R
      T1 = V*SQT**3*C1
      T2 = T2OUT
      IF( T .GT. 2.D3 ) T2 = DEXP(-XH*TK)
      T3 = T3OUT
      IF( T .GT. 5.D3 ) T3 = DEXP(-XHE*TK)
      T4 = T4OUT
      IF( T .GT. 1.D4 ) T4 = DEXP(-XHE2*TK)
      T5 = T5OUT
      IF( T .GT. 1.2D3 ) T5 = DEXP(-XM*TK)
      D = T1*T2
      B = FOR*T1*T3
      C = B*T1*T4
      DD = TWO*CM*T1*T5
      ZNA = Z*2.48D-3/24.969D0
      ZMG = Z*ZPZP/45.807D0
C
C          CONVERGE ON ELECTRON DENSITY USING THE SAHA
C             EQUATION.
C
C          GES IS THE MOLAR DENSITY OF ELECTRONS.
C
      GES = (X+Y*AHF)/(ONE+Y/(FOR*C))
      IF( GES .LT. X ) GES = AHF*(DSQRT(D*(D+FOR*X))-D)
      IF( GES .LT. 1.D-6*Z ) GES = 1.D-6*Z
      XC2 = X/C2
      YC3 = Y/C3
C
C          NEWTON METHOD FOR ELECTRON DENSITY.
C
      DO 1 I=1,25
         T2 = C/GES+GES+B
         GEP = XC2*D/(GES+D)+YC3*(B+TWO*C/GES)/T2
     @         + ZMG*DD/(GES+DD) + ZNA
         T1 = ONE+XC2*D/(D+GES)**2+YC3/T2*
     @       (TWO*C/GES**2+(B+TWO*C/GES)*(ONE-C/GES**2)/T2)
     @     + ZMG*DD/(GES+DD)**2
         DGES = (GEP-GES)/T1
         GES = GES+DGES
         IF( DABS(DGES)/GES .LT. PREC ) GOTO 3
```



```
   1  CONTINUE
      GOTO 12
   3  CONTINUE
C
C  ELECTRON PRESSURE
C
      PE = RT*GES/V
C
C      TOTLN = 1/MU = X/C2+Y/C4+Z/C3+GES
C
      TOTLN = PARGRM+GES
      XX = D/(GES+D)
      T2 = GES+B+C/GES
      YY = B/T2
      ZZ = C/(GES*T2)
      WW = DD/(GES+DD)
C
C          DERIVATIVES OF THE SAHA EQUATION FOR THE
C          PRESSURE AND INTERNAL ENERGY TEMPERATURE AND
C          DENSITY DERIVATIVES.
C
      T1 = YC3*(B+TWO*C/GES)
      QC0 = ONE+XC2*XX/(GES+D)+ZMG*WW/(GES+DD)+YC3/T2*
     @       (TWO*C/GES**2+(B+TWO*C/GES)*(ONE-C/GES**2)/T2)
      QC1 = XC2*(ONE-XX)/(GES+D)
      QC4 = ZMG*(ONE-WW)/(GES+DD)
      QC2 = (YC3-T1/T2)/T2
      QC3 = (YC3*TWO-T1/T2)/(GES*T2)
      QGV = (QC1*D+QC2*B+QC3*TWO*C+QC4*DD)/(QC0*V)
      QP1 = D*(ONHLF+XH*TK)/T
      QP2 = B*(ONHLF+XHE*TK)/T
      QP3 = C*(THRE+(XHE+XHE2)*TK)/T
      QP4 = DD*(ONHLF+XM*TK)/T
      QGT = (QC1*QP1+QC2*QP2+QC3*QP3+QC4* QP4)/QC0
C
C          ELECTRON PRESSURE DERIVATIVES.
C
      PET = ONE + QGT/GES
      PEV =-ONE + QGV/GES
C
C          PRESSURE DUE TO THE IDEAL GAS
C
      P = RT*TOTLN/V
      PT = P/T+RT*QGT/V
      PV = RT*QGV/V-P/V
C
C          BP IS R/MU
C
      BP = R*TOTLN
      BPV = R*QGV
      BPT = R*QGT
C
C           ADD THE RADIATION PRESSURE
C
      P = P+AD3*TT4
      PT = PT+FOR*AD3*TT4/T
C
C          IONIZATION ENERGY
C
      EI = (R/BK)*(XH*XX*XC2+YC3*(XHE*YY+(XHE+XHE2)*ZZ)
     @       +ZMG*XM*WW + ZNA*5.524D0 )
C          TOTAL INTERNAL ENERGY
```

```
      E = ONHLF*RT*TOTLN + A*V*TT4 + EI
      EV = T*PT-P
      QXT = ((ONE-XX)*QP1-XX*QGT)/(GES+D)
      DT2 = QGT*(ONE-C/GES**2)+QP2+QP3/GES
      QYT = (QP2-B*DT2/T2)/T2
      QZT = (QP3-C*QGT/GES-C*DT2/T2)/(T2*GES)
      QWT = ((ONE-WW)*QP4-WW*QGT)/(GES+DD)
      EIT = (R/BK)*(XH*QXT*XC2+YC3*(XHE*QYT+(XHE+XHE2)*QZT)+
     @      ZMG*XM*QWT)
      ET = ONHLF*R*(TOTLN+T*QGT)+FOR*A*V*TT4/T+EIT
      RETURN
 10   WRITE(6,100) T,V
100   FORMAT(1H0,27HNEGATIVE TEMP IN EOS     T=,1PE10.3,2X,
     *   2HV=,E10.3)
      STOP
 11   WRITE(6,101) T,V
101   FORMAT(1H0,27HNEGATIVE DENSITY IN EOS  T=,1PE10.3,2X,
     *   2HV=,E10.3)
      STOP
 12   WRITE(6,102) T,V
102   FORMAT(1H0,27HNO CONVERGENCE IN EOS    T=,1PE10.3,2X,
     *   2HV=,E10.3)
      STOP
      END
c ************************************************
c ************************************************
      SUBROUTINE OPACITY (YGO,RHO,PE,OP)
c ************************************************
c This routine computes the opacity (OP) given the
c density (RHO), temperature (T), Hydrogen mass fraction
c (X), Helium mass fraction (Y), and electron pressure (Pe
c from the equation of state routine called EOS).
c Taken from Stellingwerf 1975, {\sl Ap.J.}, {\bf 195},
c 441., with corrections given in Stellingwerf 1975,
c {\sl Ap.J.}, {\bf 199}, 705.
c This routine is to be used for 0.6<X<0.8, 0.2<Y<0.4,
c and 0.001<Z<0.02.
c You may use it outside these bounds but, if you do, use
c with caution!
c Constructing these sorts of fits is not for the amateur.
c Try it!
c ************************************************
c Temperature and volume in these fits are given in 10^4 K
c and 1/RHO.
c ************************************************
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION YGO(4)
      COMMON  AMASS, AMTOT, AL, QFIT, X, Y, Z
c Set up variables.
      T4=DEXP(YGO(2))/1.0D4
      V=1.0D0/RHO
      V1=V**0.35D0
      V2=DSQRT(V1)
      Y1=6.294D-5-(6.0D-5*Y)
      Y2=3.53D6*Y-3.0447D5
      Z1=21.0D0*Z+0.979D0
      Z2=105.0D0*Z+0.895D0
c ************************************************
c  Compute kappa (OP) from equation D3 of Stellingwerf
c  1975 which involves some continued fractions
c  (temp1, temp2).
c ************************************************
```



```
      TEMP1=Y1*V1*DSQRT(T4)*T4**3+1.0D0/(760.0D0*T4**5
     *   +316.0D0/V2)
      TEMP1=1.0D0/(10.0D0*T4**6+1.0D0/TEMP1)
      TEMP2=1780.0D0*DSQRT(T4)*T4*T4/Z1+1.0D0/(Z1*Y2/T4**10
     *   +2.13D-3*Z2*V2/DSQRT(T4)/T4**4)
      TEMP2=47.3D0/T4**8+1.0D0/TEMP2
      TEMP2=1.0D0/(4.0D3+1.0D0/TEMP2)
      OP=PE*(4.819D-13*V/T4+TEMP1+TEMP2)
      RETURN
      END
c ************************************************
c ************************************************
        SUBROUTINE EPSILON (YGO, RHO, EPS)
c ************************************************
c This computes the energy generation rate for
c the sum of the pp-chains and CNO cycles.
c The simplified formulae from chapter 6
c are used.
c ************************************************
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      COMMON  AMASS, AMTOT, AL, QFIT, X, Y, Z
      DIMENSION YGO(4)
c
c If the temperature is below 1.2x10**6, then
c neglect the energy generation rate.
      IF (YGO(2) .LT. 14.0D0) THEN
         EPS=1.D-20
         RETURN
      ENDIF
      T9=DEXP(YGO(2))/1.0D9
      T913=T9**(1.0D0/3.0D0)
      T923=T913*T913
      EPSPP=2.4D4*RHO*X*X*DEXP(-3.380/T913)/T923
      EPSCNO=4.4D25*RHO*X*Z*DEXP(-15.228D0/T913)/T923
      EPS=EPSPP+EPSCNO
      RETURN
      END
c ************************************************
c ************************************************
      SUBROUTINE CONV(DELRAD,DELAD,P0,T0,RHO,OP,G,DEL)
c Solves the cubic to find DEL given DELRAD, DELAD, etc.,
c using the mixing length theory of Chapter .
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      PARAMETER (AC24=6.84D7)
      P=DEXP(P0)
      T=DEXP(T0)
      DDEL=DELRAD-DELAD
      C=DDEL*(P/T)**5/T**3
      C=C*AC24*(OP/G)**2/RHO
c Make initial guesses for X and then iterate.
      IF (C .GT. 1.0) THEN
         X=C**0.333333
      ELSE
         X=C
      ENDIF
      DO 1 ITRY=1,50
        F=X*(X*X+X+1.D0)-C
        DF=X*(3.D0*X+2.D0)+1.0D0
        DX=-F/DF
        IF (DABS(DX/X) .GT. 0.3D0)
     *    DX=0.3D0*DSIGN(X,DX)
        X=X+DX
```

```
        IF (DABS(DX/X) .LT. 2.0D-11) GOTO 2
    1 CONTINUE
      WRITE (6,1000) X
 1000 FORMAT(' CONVECTION CUBIC NOT CONVERGED X=',1PD9.2)
      STOP 'SORRY'
    2 ALAMBDA=X*(X+1.D0)
      DEL=DDEL*ALAMBDA/C+DELAD
      RETURN
      END
c ***********************************************
c The following subroutines comprise the heart of the
c Runge-Kutta integrator. They are due to H. A. Watts
c and L. F. Shampine of Sandia Laboratories, Albuquerque,
c New Mexico. For a textbook reference, discussion, and
c annotated listing of these programs, see G. E. Forsythe,
c M. A. Malcolm, and C. B. Moler (1977), "Computer Methods
c for Mathematical Computation" [Prentice Hall: N.J.],
c pp. 127--147.
C ********************************************************
      SUBROUTINE RKF(F,NEQN,Y,T,TOUT,RELERR,ABSERR,
     @        IFLAG,WORK,IWORK)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION Y(NEQN),WORK(1),IWORK(5)
      EXTERNAL F
      K1M=NEQN+1
      K1=K1M+1
      K2=K1+NEQN
      K3=K2+NEQN
      K4=K3+NEQN
      K5=K4+NEQN
      K6=K5+NEQN
      CALL RKFS(F,NEQN,Y,T,TOUT,RELERR,ABSERR,IFLAG,
     @    WORK(1),WORK(K1M),WORK(K1),WORK(K2),WORK(K3),
     @    WORK(K4),WORK(K5),WORK(K6),WORK(K6+1),
     @    IWORK(1),IWORK(2),IWORK(3),IWORK(4),IWORK(5))
      RETURN
      END
C-----------------------------------------------------
C
      SUBROUTINE RKFS(F,NEQN,Y,T,TOUT,RELERR,ABSERR,
     @    IFLAG,YP,H,F1,F2,F3,F4,F5,SAVRE,SAVAE,NFE,
     @    KOP,INIT,JFLAG,KFLAG)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      LOGICAL HFAILD,OUTPUT
      DIMENSION Y(NEQN),YP(NEQN),F1(NEQN),F2(NEQN),
     @    F3(NEQN),F4(NEQN),F5(NEQN)
      EXTERNAL F
      DATA U26/2.D-13/ , REMIN/2.D-13/
      DATA MAXNFE/3000/
      IF (NEQN .LT. 1) GO TO 10
      IF ((RELERR .LT. 0.D0)  .OR.  (ABSERR .LT. 0.D0))
     @  GO TO 10
      MFLAG=IABS(IFLAG)
      IF ((MFLAG .GE. 1) .AND. (MFLAG .LE. 7)) GO TO 20
   10 IFLAG=7
      RETURN
   20 IF (MFLAG .EQ. 1) GO TO 50
      IF (T .EQ. TOUT) GO TO 10
      IF(MFLAG .NE. 2) GO TO 25
      IF (INIT .EQ. 0) GO TO 45
      IF (KFLAG .EQ. 3) GO TO 40
      IF ((KFLAG .EQ. 4) .AND.  (ABSERR .EQ. 0.D0))
```



```
     @    GO TO 30
      IF ((KFLAG .EQ. 5)  .AND. (RELERR .LE. SAVRE)
     @     .AND. (ABSERR .LE. SAVAE)) GO TO 30
      GO TO 50
   25 IF (IFLAG .EQ. 3) GO TO 40
      IF ((IFLAG .EQ. 4) .AND. (ABSERR .GT. 0.D0))
     @    GO TO 45
   30 WRITE (6,1000) IFLAG,T
 1000 FORMAT (16H RKF SAYS IFLAG=,I5,3H X=,1PD12.4)
      STOP
   40 NFE=0
      IF (MFLAG .EQ. 2) GO TO 50
   45 IFLAG=JFLAG
   50 JFLAG=IFLAG
      KFLAG=0
      SAVRE=RELERR
      SAVAE=ABSERR
      RER=DMAX1(RELERR,REMIN)
      DT=TOUT-T
      IF (MFLAG .EQ. 1) GO TO 60
      IF (INIT .EQ. 0) GO TO 65
      GO TO 80
   60 INIT=0
      KOP=0
      A=T
      CALL F(A,Y,YP)
      NFE=1
      IF (T .NE. TOUT) GO TO 65
      IFLAG=2
      RETURN
   65 INIT=1
      YMAX=0.D0
      YPN=0.D0
      DO 70 K=1,NEQN
        YPN=DMAX1(DABS(YP(K)),YPN)
70      YMAX=DMAX1(DABS(Y(K)),YMAX)
      ETN=RER*YMAX+ABSERR
      H=DABS(DT)
      IF(ETN.GE.YPN*H**5) GO TO 80
      H=DMAX1((ETN/YPN)**0.2D0,U26*DMAX1(DABS(T),H))
80    H=DSIGN(H,DT)
      IF (DABS(H).GE.DABS(DT)) KOP=KOP+1
      IF (KOP.NE.100) GO TO 85
      IFLAG=6
      RETURN
85    IF (DABS(DT).GT.U26*DABS(T))GO TO 95
      DO 90 K=1,NEQN
90      Y(K)=Y(K)+DT*YP(K)
      A=TOUT
      CALL F(A,Y,YP)
      NFE=NFE+1
      GO TO 300
95    OUTPUT=.FALSE.
      SCALE=2.D0/RER
      AE=SCALE*ABSERR
100   HFAILD=.FALSE.
      HMIN=U26*DABS(T)
      DT=TOUT-T
      IF (DABS(DT).GE.2.D0*DABS(H)) GO TO 200
      IF (DABS(DT).GT.DABS(H)/0.9D0) GO TO 150
      OUTPUT=.TRUE.
      H=DT
```

```
      GO TO 200
150   H=0.5D0*DT
200   IF (NFE.LE.MAXNFE) GO TO 220
      IFLAG=3
      KFLAG=3
      RETURN
220   CALL FEHL(F,NEQN,Y,T,H,YP,F1,F2,F3,F4,F5,F1)
      NFE=NFE+5
      EEOET=0.D0
      DO 250 K=1,NEQN
        ET=DABS(Y(K))+DABS(F1(K))+AE
        IF (ET.GT.0.D0) GO TO 240
        IFLAG=4
        KFLAG=4
        RETURN
240     EE=DABS((-2090.D0*YP(K)+(21970.D0*F3(K)-15048.D0
     @      *F4(K)))+(22528.D0*F2(K)-27360.D0*F5(K)))
250     EEOET=DMAX1(EEOET,EE/ET)
      ESTTOL=DABS(H)*EEOET*SCALE/752400.D0
      IF (ESTTOL.LE.1.D0) GO TO 260
      HFAILD=.TRUE.
      OUTPUT=.FALSE.
      S=0.1D0
      IF (ESTTOL.LT.59049.D0)S=0.9D0/ESTTOL**0.2D0
      H=S*H
      IF (DABS(H).GT.HMIN) GO TO 200
      IFLAG=5
      KFLAG=5
      RETURN
260   T=T+H
      DO 270 K=1,NEQN
270     Y(K)=F1(K)
      A=T
      CALL F(A,Y,YP)
      NFE=NFE+1
      IF (HFAILD) GO TO 290
      S=5.D0
      IF (ESTTOL.GT.1.889568D-04) S=0.9D0/ESTTOL**0.2D0
      H=DSIGN(DMAX1(S*DABS(H),HMIN),H)
290   IF (OUTPUT) GO TO 300
      IF (IFLAG.GT.0) GO TO 100
      IFLAG=-2
      RETURN
300   T=TOUT
      IFLAG=2
      RETURN
      END
C-----------------------------------------------------
C
      SUBROUTINE FEHL(F,NEQN,Y,T,H,YP,F1,F2,F3,F4,F5,S)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION Y(NEQN),YP(NEQN),F1(NEQN),F2(NEQN),
     @   F3(NEQN),F4(NEQN),F5(NEQN),S(NEQN)
      CH=0.25D0*H
      DO 10 K=1,NEQN
   10     F5(K)=Y(K)+CH*YP(K)
      CALL F(T+0.25D0*H,F5,F1)
      CH=0.09375D0*H
      DO 20 K=1,NEQN
   20     F5(K)=Y(K)+CH*(YP(K)+3.D0*F1(K))
      CALL F(T+0.375D0*H,F5,F2)
      CH=H/2197.D0
```



```
      DO 30 K=1,NEQN
   30      F5(K)=Y(K)+CH*(1932.D0*YP(K)+(7296.D0*F2(K)
     @      -7200.D0*F1(K)))
      CALL F(T+12.D0/13.D0*H,F5,F3)
      CH=H/4104.D0
      DO 40 K=1,NEQN
   40      F5(K)=Y(K)+CH*((8341.D0*YP(K)-845.D0*F3(K))+
     @         (29440.D0*F2(K)-32832.D0*F1(K)))
      CALL F(T+H,F5,F4)
      CH=H/20520.D0
      DO 50 K=1,NEQN
   50    F1(K)=Y(K)+CH*((-6080.D0*YP(K)+(9295.D0*F3(K)
     @      -5643.D0*F4(K)))+(41040.D0*F1(K)-28352.D0
     @      *F2(K)))
      CALL F(T+0.5D0*H,F1,F5)
      CH=H/7618050.D0
      DO 60 K=1,NEQN
   60     S(K)=Y(K)+CH*((902880.D0*YP(K)+(3855735.D0
     @       *F3(K)-1371249.D0*F4(K)))+(3953664.D0*F2(K)
     @       +277020.D0*F5(K)))
      RETURN
      END
C ************************************************
C Here follow the LINPACK routines to solve linear equations.
C They are used to get corrections on guesses. For a discussion
C of the use of these routines see Dongarra et al. 1979,
C "LINPACK User's Guide" (Philadelphia, SIAM).
C ************************************************

      subroutine dgefa(a,lda,n,ipvt,info)
      integer lda,n,ipvt(1),info
      double precision a(lda,1)
c
c     dgefa factors a double precision matrix by gaussian elimination.
c
c     dgefa is usually called by dgeco, but it can be called
c     directly with a saving in time if  rcond  is not needed.
c     (time for dgeco) = (1 + 9/n)*(time for dgefa) .
c
c     on entry
c
c        a       double precision(lda, n)
c                the matrix to be factored.
c
c        lda     integer
c                the leading dimension of the array  a .
c
c        n       integer
c                the order of the matrix  a .
c
c     on return
c
c        a       an upper triangular matrix and the multipliers
c                which were used to obtain it.
c                the factorization can be written  a = l*u  where
c                l  is a product of permutation and unit lower
c                triangular matrices and  u  is upper triangular.
c
c        ipvt    integer(n)
c                an integer vector of pivot indices.
c
c        info    integer
```

```
c                 = 0  normal value.
c                 = k  if  u(k,k) .eq. 0.0 .  this is not an error
c                      condition for this subroutine, but it does
c                      indicate that dgesl or dgedi will divide by zero
c                      if called.  use  rcond  in dgeco for a reliable
c                      indication of singularity.
c
c     linpack. this version dated 08/14/78 .
c     cleve moler, university of new mexico, argonne national lab.
c
c     subroutines and functions
c
c     blas daxpy,dscal,idamax
c
c     internal variables
c
      double precision t
      integer idamax,j,k,kp1,l,nm1
c
c
c     gaussian elimination with partial pivoting
c
      info = 0
      nm1 = n - 1
      if (nm1 .lt. 1) go to 70
      do 60 k = 1, nm1
         kp1 = k + 1
c
c        find l = pivot index
c
         l = idamax(n-k+1,a(k,k),1) + k - 1
         ipvt(k) = l
c
c        zero pivot implies this column already triangularized
c
         if (a(l,k) .eq. 0.0d0) go to 40
c
c           interchange if necessary
c
            if (l .eq. k) go to 10
               t = a(l,k)
               a(l,k) = a(k,k)
               a(k,k) = t
   10       continue
c
c           compute multipliers
c
            t = -1.0d0/a(k,k)
            call dscal(n-k,t,a(k+1,k),1)
c
c           row elimination with column indexing
c
            do 30 j = kp1, n
               t = a(l,j)
               if (l .eq. k) go to 20
                  a(l,j) = a(k,j)
                  a(k,j) = t
   20          continue
               call daxpy(n-k,t,a(k+1,k),1,a(k+1,j),1)
   30       continue
         go to 50
   40    continue
```



```
            info = k
   50    continue
   60 continue
   70 continue
      ipvt(n) = n
      if (a(n,n) .eq. 0.0d0) info = n
      return
      end

      subroutine dgesl(a,lda,n,ipvt,b,job)
      integer lda,n,ipvt(1),job
      double precision a(lda,1),b(1)
c     This is the double precision routine DGESL from LINPACK
c
c     dgesl solves the double precision system
c     a * x = b  or  trans(a) * x = b
c     using the factors computed by dgeco or dgefa.
c
c     on entry
c
c        a       double precision(lda, n)
c                the output from dgeco or dgefa.
c
c        lda     integer
c                the leading dimension of the array  a .
c
c        n       integer
c                the order of the matrix  a .
c
c        ipvt    integer(n)
c                the pivot vector from dgeco or dgefa.
c
c        b       double precision(n)
c                the right hand side vector.
c
c        job     integer
c                = 0         to solve  a*x = b ,
c                = nonzero   to solve  trans(a)*x = b  where
c                            trans(a)  is the transpose.
c
c     on return
c
c        b       the solution vector  x .
c
c     error condition
c
c        a division by zero will occur if the input factor contains a
c        zero on the diagonal.  technically this indicates singularity
c        but it is often caused by improper arguments or improper
c        setting of lda .  it will not occur if the subroutines are
c        called correctly and if dgeco has set rcond .gt. 0.0
c        or dgefa has set info .eq. 0 .
c
c     to compute  inverse(a) * c  where  c  is a matrix
c     with  p  columns
c           call dgeco(a,lda,n,ipvt,rcond,z)
c           if (rcond is too small) go to ...
c           do 10 j = 1, p
c              call dgesl(a,lda,n,ipvt,c(1,j),0)
c        10 continue
c
c     linpack. this version dated 08/14/78 .
```

```
c     cleve moler, university of new mexico, argonne national lab.
c
c     subroutines and functions
c
c     blas daxpy,ddot
c
c     internal variables
c
      double precision ddot,t
      integer k,kb,l,nm1
c
      nm1 = n - 1
      if (job .ne. 0) go to 50
c
c        job = 0 , solve  a * x = b
c        first solve  l*y = b
c
         if (nm1 .lt. 1) go to 30
         do 20 k = 1, nm1
            l = ipvt(k)
            t = b(l)
            if (l .eq. k) go to 10
               b(l) = b(k)
               b(k) = t
   10       continue
            call daxpy(n-k,t,a(k+1,k),1,b(k+1),1)
   20    continue
   30    continue
c
c        now solve  u*x = y
c
         do 40 kb = 1, n
            k = n + 1 - kb
            b(k) = b(k)/a(k,k)
            t = -b(k)
            call daxpy(k-1,t,a(1,k),1,b(1),1)
   40    continue
      go to 100
   50 continue
c
c        job = nonzero, solve  trans(a) * x = b
c        first solve  trans(u)*y = b
c
         do 60 k = 1, n
            t = ddot(k-1,a(1,k),1,b(1),1)
            b(k) = (b(k) - t)/a(k,k)
   60    continue
c
c        now solve trans(l)*x = y
c
         if (nm1 .lt. 1) go to 90
         do 80 kb = 1, nm1
            k = n - kb
            b(k) = b(k) + ddot(n-k,a(k+1,k),1,b(k+1),1)
            l = ipvt(k)
            if (l .eq. k) go to 70
               t = b(l)
               b(l) = b(k)
               b(k) = t
   70       continue
   80    continue
   90    continue
```



```
  100 continue
      return
      end

      double precision function ddot(n,dx,incx,dy,incy)
c This is the LINPACK routine DDOT
c
c     forms the dot product of two vectors.
c     uses unrolled loops for increments equal to one.
c     jack dongarra, linpack, 3/11/78.
c
      double precision dx(1),dy(1),dtemp
      integer i,incx,incy,ix,iy,m,mp1,n
c
      ddot = 0.0d0
      dtemp = 0.0d0
      if(n.le.0)return
      if(incx.eq.1.and.incy.eq.1)go to 20
c
c        code for unequal increments or equal increments
c          not equal to 1
c
      ix = 1
      iy = 1
      if(incx.lt.0)ix = (-n+1)*incx + 1
      if(incy.lt.0)iy = (-n+1)*incy + 1
      do 10 i = 1,n
        dtemp = dtemp + dx(ix)*dy(iy)
        ix = ix + incx
        iy = iy + incy
   10 continue
      ddot = dtemp
      return
c
c        code for both increments equal to 1
c
c
c        clean-up loop
c
   20 m = mod(n,5)
      if( m .eq. 0 ) go to 40
      do 30 i = 1,m
        dtemp = dtemp + dx(i)*dy(i)
   30 continue
      if( n .lt. 5 ) go to 60
   40 mp1 = m + 1
      do 50 i = mp1,n,5
        dtemp = dtemp + dx(i)*dy(i) + dx(i + 1)*dy(i + 1) +
     *   dx(i + 2)*dy(i + 2) + dx(i + 3)*dy(i + 3) + dx(i + 4)*dy(i + 4)
   50 continue
   60 ddot = dtemp
      return
      end

      subroutine daxpy(n,da,dx,incx,dy,incy)
c This is the LINPACK routine DAXPY
c
c     constant times a vector plus a vector.
c     uses unrolled loops for increments equal to one.
c     jack dongarra, linpack, 3/11/78.
c
      double precision dx(1),dy(1),da
```

```
      integer i,incx,incy,m,mp1,n
c
      if(n.le.0)return
      if (da .eq. 0.0d0) return
      if(incx.eq.1.and.incy.eq.1)go to 20
c
c        code for unequal increments or equal increments
c          not equal to 1
c
      ix = 1
      iy = 1
      if(incx.lt.0)ix = (-n+1)*incx + 1
      if(incy.lt.0)iy = (-n+1)*incy + 1
      do 10 i = 1,n
        dy(iy) = dy(iy) + da*dx(ix)
        ix = ix + incx
        iy = iy + incy
   10 continue
      return
c
c        code for both increments equal to 1
c
c
c        clean-up loop
c
   20 m = mod(n,4)
      if( m .eq. 0 ) go to 40
      do 30 i = 1,m
        dy(i) = dy(i) + da*dx(i)
   30 continue
      if( n .lt. 4 ) return
   40 mp1 = m + 1
      do 50 i = mp1,n,4
        dy(i) = dy(i) + da*dx(i)
        dy(i + 1) = dy(i + 1) + da*dx(i + 1)
        dy(i + 2) = dy(i + 2) + da*dx(i + 2)
        dy(i + 3) = dy(i + 3) + da*dx(i + 3)
   50 continue
      return
      end

      integer function idamax(n,dx,incx)
c
c     finds the index of element having max. absolute value.
c     jack dongarra, linpack, 3/11/78.
c
      double precision dx(1),dmax
      integer i,incx,ix,n
c
      idamax = 0
      if( n .lt. 1 ) return
      idamax = 1
      if(n.eq.1)return
      if(incx.eq.1)go to 20
c
c        code for increment not equal to 1
c
      ix = 1
      dmax = dabs(dx(1))
      ix = ix + incx
      do 10 i = 2,n
         if(dabs(dx(ix)).le.dmax) go to 5
```



```
         idamax = i
         dmax = dabs(dx(ix))
    5    ix = ix + incx
   10 continue
      return
c
c        code for increment equal to 1
c
   20 dmax = dabs(dx(1))
      do 30 i = 2,n
         if(dabs(dx(i)).le.dmax) go to 30
         idamax = i
         dmax = dabs(dx(i))
   30 continue
      return
      end

      subroutine  dscal(n,da,dx,incx)
c
c     scales a vector by a constant.
c     uses unrolled loops for increment equal to one.
c     jack dongarra, linpack, 3/11/78.
c
      double precision da,dx(1)
      integer i,incx,m,mp1,n,nincx
c
      if(n.le.0)return
      if(incx.eq.1)go to 20
c
c        code for increment not equal to 1
c
      nincx = n*incx
      do 10 i = 1,nincx,incx
        dx(i) = da*dx(i)
   10 continue
      return
c
c        code for increment equal to 1
c
c
c        clean-up loop
c
   20 m = mod(n,5)
      if( m .eq. 0 ) go to 40
      do 30 i = 1,m
        dx(i) = da*dx(i)
   30 continue
      if( n .lt. 5 ) return
   40 mp1 = m + 1
      do 50 i = mp1,n,5
        dx(i) = da*dx(i)
        dx(i + 1) = da*dx(i + 1)
        dx(i + 2) = da*dx(i + 2)
        dx(i + 3) = da*dx(i + 3)
        dx(i + 4) = da*dx(i + 4)
   50 continue
      return
      end
```