CLM-P683

UKAEA

Preprint

THE REGENERATION OF FORTRAN SOFTWARE

K. V. ROBERTS

CULHAM LABORATORY
Abingdon Oxfordshire

1982

CLM-P 683

THE REGENERATION OF FORTRAN SOFTWARE

K V Roberts

Culham Laboratory
Abingdon, Oxford, OX14 3DB, UK
(EURATOM/UKAEA Fusion Association)

ABSTRACT

There is a substantial body of existing Fortran Software that has considerable scientific and commercial value, but whose potential is sometimes difficult to exploit to the full because of a lack of structure and internal documentation. This article discusses, by means of an example, how the OLYMPUS COMPOSITOR can be used to regenerate such software semi-automatically so that it meets improved documentation standards. Comments and headings can be edited in on-line, using a free format, and the COMPOSITOR then produces a clear standard layout in which the Fortran statement numbers are correlated with the decimally numbered sections and subsections of an individual routine, and the meaning of the code is clarified by appropriate indentation and cross-referencing. Using the OLYMPUS GENERATOR one can also restructure the COMMON blocks and construct indexes of variables. It is suggested that such techniques can materially enhance the usefulness of a great deal of Fortran software, including some of the programs already included in the CPC Program Library.

(Submitted for publication in Computer Physics Communications)

September 1982

Introduction.

There is by now a substantial body of available computer software, much of it written in Fortran, that is often of considerable scientific and commercial value but of quite variable quality so far as the structure and documentation are concerned.

This is true for example of the programs submitted for publication in the CPC Program Library. In many cases the authors have taken great care to make the listings as readable and intelligible as possible by the use of clear layout, adequate headings and comments, tables, indexes and other techniques of the kind recommended in the CPC Instructions to Authors (see 4th Revision, ref. [1]). At the other extreme there are sometimes no tables or indexes, little apparent structure or layout conventions, few or no comments, and statement numbers in arbitrary order.

Certainly the structure and documentation make almost no difference to the working of a program. One could remove all the comments, randomize the Fortran identifiers, statement numbers and layout, and mix up the different program activities, and it would be just as accurate and efficient as before. However it would in practice be much less useful since the following evident requirements could not be met:

1. Clear understanding of what the program does.
2. Guarantee of correctness.
3. Correction of any errors.
4. Conversion to other types of computer system.
5. Possible improvements in operating efficiency.
6. Adaptation to meet changing requirements.
7. Availability as a model for future work.
8. Efficient software project management.

The other OLYMPUS articles in this issue [2, 3, 4] describe utilities and conventions that facilitate the writing of new programs to meet these requirements. However, there remains the question of what to do with existing software when it is too expensive to discard and replace but also very costly to maintain in operation. This paper demonstrates how the COMPOSITOR [2] can be used to regenerate the structure of an old Fortran subroutine, and illustrates other techniques that can bring the documentation to an acceptable standard.

2. Initial Subroutine Version.

Many examples of poor programming style could be chosen. However, since it would be invidious to use anyone else's work for this purpose I have selected a subroutine from the very early 1DMHD code written by the author and his colleagues [5, 6], and which actually dates from about 1960. This is shown in Fig. 1. A few comments added later have been removed to restore the listing to its initial state, and those COMMON identifiers that are not used by the subroutine have been omitted to save space, but the listing is otherwise unchanged. It is by no means untypical of much of the Fortran software that is currently in use or even being written at the present time.

The listing is evidently unintelligible but some specific deficiencies are:

1. No decimal subprogram number to locate its position in the listing.
2. Only a brief outline of what the subroutine does: no details or explanation of its working.
3. Untidy COMMON.
4. No comments.
5. No documentation of the internal structure.
6. No explanation of variable, array or subprogram identifiers.
7. Statement numbers in random order.
8. No cross-references to external documentation.

3. Use of COMPOSITOR for Regeneration

The regeneration procedure is quite straightforward and consists of editing in free-format comments, section headings (#) and subsection headings (# #) at a VDU terminal, and then using the COMPOSITOR [2] to generate a clear layout automatically. In practice this is an iterative procedure since as the code gradually becomes better documented its working becomes easier to follow and the regeneration can proceed more easily. To save space I have shown only a part of the edited COMPOSITOR input file in Fig.2, and then the final version of the subroutine in Fig.3. No executable statements have been altered at this stage, except for a permutation of the statement numbers, the replacement of 'PRINT' by 'WRITE', and the use of a continuation line where the original statement was too long when shifted to col.10.

External documentation was already available for the 1DMHD code (although written well after 1960) and this has been referred to in the cross-references shown in Fig.3. In general it may be necessary to produce much of the external documentation in parallel with the regeneration of the code itself.

The original blank COMMON has been subdivided into separate labelled and numbered blocks, each with a specific purpose, and these are included in the code by means of the C/ INSERT statements shown in Fig. 3. The regeneration of COMMON and the production of indexes can be carried out by the GENSIS generator [3], the relevant index entries being shown in Fig. 4, and some of the output in Fig.5.

Modification of Executable Statements.

Once the listing is in a clear state it may be useful and practicable to improve the coding further by changing some of the executable statements. A general procedure should be to carry out one or more standard test runs before the regeneration begins, and to repeat these at each stage in order to assist in the detection of any errors that have inadvertently been introduced.

The only change illustrated here (Fig.6) is the replacement of arithemetic IF and computed GO TO statements by logical IF (which was not available when the original code was written), although in some cases a much more substantial re-organization might be desirable, for example changing COMMON identifiers using an Editor to ensure that they are consistent throughout the code, or breaking down long subroutines into shorter ones.

Dialect Statements.

In its published version [2] the COMPOSITOR cannot handle non-standard dialect Fortran statements that it does not recognize. This mainly applies to non-standard declarations which usually can be tidied up afterwards quite simply with an Editor. Non-standard executable statements will usually be treated correctly but those that contain references to statement numbers may require editing. The COMPOSITOR program is however designed in such a way that it can readily be adapted to handle specific Fortran 66 dialects or be extended to deal with Fortran 77.

Concluding Remarks.

It is believed that the regeneration and maintenance of software will be an increasingly important and economically necessary task in the future, and that tools such as the COMPOSITOR [2] and GENSIS generator [3] will prove useful for this type of work. Some of the programs that are currently being submitted to the CPC Library would benefit from such preprocessing, and this would materially lighten the task of editors and referees as well as that of

future users. Authors and subscribers might also like to consider the use of these Utility programs to produce regenerated adaptations of some of the programs that are already in the Library.

The computer output was produced on an ADLER SE1005 typewriter with a carbon ribbon, interfaced to the Culham PRIME computer via a British Telecom 300 baud modem.

REFERENCES

[1] Computer Physics Communications, Instructions to Authors, 4th Revision, Comput. Phys. Commun. 27 (1982) 1.

[2] The OLYMPUS FORTRAN COMPOSITOR, M.H.Hughes and K.V.Roberts, CLM-P690

[3] The OLYMPUS FORTRAN GENERATOR, M.H.Hughes and K.V.Roberts, CLM-P689

[4] OLYMPUS FORTRAN Conventions, M.H.Hughes and K.V.Roberts, CLM-P685

[5] K.Hain, G.Hain, K.V.Roberts and S.J.Roberts, Zeits. für Naturf. 15a (1960) 1039.

[6] K.V.Roberts, Journ. Nucl. Energy. Part C 5 (1963) 365.

```
*IDENTIFIERDF12GS
*FORTRAN
      SUBROUTINE DF12GS
      COMMON KGAS,UXDA,T,UXA,SWN,ACC,RELAX,N1,FACHC,KHC,NHCORE,
     1N,KBA,KB,FLUXN,VNE,RHOM1,RZA,NO3,GA5,DELTAT
      COMMON VISCN(51),MARKER(51),VN(51),RHON(51),DELTA1(51),
     1VISC2(51),R2(51),QN(51),DN(51),ALPHA(51),BETA(51),RHO(51),
     2S4(51),V(51),S1(51),AN(51),AN1(51),DELTA2(51),AJO1(51),AN2(51),
     3RHON2(51),VN2(51)
C        DF12GS SOLVES THE FOLLOWING EQN.S
C
C             DVN      D (RHON.K.TN)    DPN
C        RHON.--- = - --(---------) - --- + G.(V-VN)
C             DT       DR(    M     )   DR
C
C        DRHON      RHON D
C        ----- = - ----.--(R.VN) - S
C        DT          R    DR
C
C        DTN                TN D            G
C        --- = - (GAMMA-1).--.--(R.VN) - ----.(TN-TI)
C        DT                 R  DR          RHON
C
C
      GO TO (8,2),KGAS
 2    Z1=EXPF(-UXDA*T)
      FLUXN=UXA*Z1
      Z2=SWN**2
      Z12=0.5*ACC*RELAX
      DO 10 K=2,N1
      VISCN(K)=0.0
      Z=MARKER(K)-1
      Z1=(VN(K+1)-VN(K-1))*Z
      IF(Z1)7,10,10
 7    VISCN(K)=-0.5*RHON(K)*Z1*Z2/DELTA1(K+1)
 10   CONTINUE
      VISCN(1)=VISCN(2)
      VISCN(N)=VISCN(N1)
      CALL DF12VS(2)
      DO 999 K=1,N
      VISC2(K)=VISCN(K)/3.0
999   CONTINUE
      DO 9 K=2,N1
      VISCN(K)=VISC 2(K-1)+VISC2(K)+VISC2(K+1)
 9    CONTINUE
      VISCN(1)=VISCN(2)
      VISCN(N)=VISCN(N1)
      DO 32 K=1,N1
      Z=(VN(K+1)-VN(K))/DELTA1(K+1)+0.5*(VN(K+1)+VN(K))/R2(K)
      QN(K)=-Z*VISCN(K)/RHON(K)
      DN(K)=Z*0.25*DELTAT
32    CONTINUE
      QN(N)=0.0
      X=0.5*DELTAT
      Z=DELTAT**2/8.0
      ALPHA(1)=0.0
      BETA(1)=0.0
      X2=2.0*DELTAT*FACHC
      ZZ4=4.0*DN(KHC)*QN(KHC)
      DO 41 K=2,N
      MKR=MARKER(K)
      GO TO (31,36),KGAS
36    GO TO (31,33),MKR
31    X4=0.0
      X5=0.0
      X6=0.0
      GO TO 35
33    X4=2.0*(S4(K)*RHO(K)+S4(K-1)*RHO(K-1))
      X5=2.0*V(K)
      X6=X*(S1(K)-S1(K-1))
35    Z1=AN(K)+AN(K-1)+AN1(K)+AN1(K-1)
      Z2=2.0*(RHON(K)+RHON(K-1))
      ZZ1=AN(K)+AN1(K)-AN(K-1)-AN1(K-1)
      ZZ5=4.0*DN(K)*QN(K)
      ZZ8=GA5*(AN(K)+AN1(K))
      ZZ9=GA5*(AN(K-1)+AN1(K-1))
      Z5=X*((ZZ1+2.0*Z1*(RHON(K)-RHON(K-1)-X6)/Z2-ZZ5+ZZ4)/DELTA2(K)
     1-X4*X5/Z2)
      Z6=Z*(ZZ8+Z1*2.0*RHON(K)/Z2)/DELTA2(K)
     1+X2*VISCN(K)/(Z2*DELTA2(K))
      Z7=Z*(ZZ9+Z1*2.0*RHON(K-1)/Z2)/DELTA2(K)
     1+X2*VISCN(K-1)/(Z2*DELTA2(K))
      ZZ4=ZZ5
      V2D=VN(K+1)
      GO TO(311,312),NHCORE
312   IF(K-2)310,310,204
310   X2=X2*10.0
      E2=0.0
      Z7=0.0
      E1=Z6/DELTA1(K+1)+Z6/(2.0*R2(K))
      GO TO 304
311   IF(K-N1)204,210,205
210   X2=0.1*X2
      GO TO 204
205   E1=0.0
      Z6=0.0
      V2D=0.0
      GO TO 206
204   E1=Z6/DELTA1(K+1)+Z6/(2.0*R2(K))
206   E2=Z7/DELTA1(K)-Z7/(2.0*R2(K-1))
304   E3=E1+E2-Z6/R2(K)+Z7/R2(K-1)+X*X4/Z2+1.0
      F=-Z5+E1*V2D-(E3-2.0)*VN(K)+E2*VN(K-1)
      ALPHA(K)=E1/(E3-E2*ALPHA(K-1))
      BETA(K)=(F+E2*BETA(K-1))/(E3-E2*ALPHA(K-1))
41    CONTINUE
      V2D=0.0
      K=N
102   VN2(K)=V2D*ALPHA(K)+BETA(K)
      V2D=VN2(K)
      K=K-1
      IF (K-1) 103,103,102
103   GO TO (68,69),KBA
69    GO TO (68,70),KB
68    Z1=RHON(N)*DELTA1(N)
      Z2=FLUXN*DELTAT
      VN2(N)=(VN2(N)*Z1-VNE*Z2)/(Z1+Z2)
 70   GO TO (300,301),NHCORE
301   GO TO (368,369),KBA
369   GO TO (368,300),KBHC
368   Z1=RHON(1)*DELTA1(2)
      Z2=FLUXN*DELTAT
      VN2(2)=(VN2(2)*Z1+VNE*Z2)/(Z1+Z2)
300   DO 1 K=KHC,N1
      MKR=MARKER(K)
      GO TO (3,6),MKR
3     AN2(K)=AJO1(K)
      RHON2(K)=RHOM1
      GO TO 1
6     Z2=(VN2(K+1)-VN2(K))/DELTA1(K+1)+0.5*(VN2(K+1)+VN2(K))/R2(K)*RZA
      DHN2=Z2*0.25*DELTAT
      Z11=ABSF(DN(K)+DHN2)
      IF (Z11-Z12) 4,4,5
5     PRINT 201,Z11,K
      NO3=2
4     QN2=-Z2*VISCN(K)/RHON(K)
      Z1=(DN(K)+DHN2)*(QN(K)+QN2)*GA5
      Z=1.0+GA5*(DN(K)+DHN2)
      AN2(K)=(AN1(K)*(2.0-Z)-Z1)/Z
      Z=1.0+DN(K)+DHN2
      RHON2(K)=(RHON(K)*(2.0-Z)
     1         -S1(K)*DELTAT)/Z
      RHON2(K)=MAX1F(RHON2(K),RHOM1)
1     CONTINUE
      CALL DF12BC(3)
8     RETURN
201   FORMAT(29H  NEUTRAL COMPRESSION FACTOR=,E12.4,4H  K=,I3)
      END
```

Fig.1 **Original Version of Subroutine DF12GS.** The main deficiencies are a lack of overall structure, the absence of comments, statement numbers in random order and an untidy COMMON.

```
C-----
*Bypass this routine if plasma is fully ionised
      GO TO (8,2),KGAS
£Space/time scale parameters
*Decay of neutral flux from wall, Eq. (12.7)
 2    Z1=EXPF(-UXDA*T)
      FLUXN=UXA*Z1
*Neutral shock width
      Z2=SWN**2
*Maximum compression rate
      Z12=0.5*ACC*RELAX
£Neutral viscosity
££Von-Neumann shock term, Eq. (5.20)
      DO 10 K=2,N1
      VISCN(K)=0.0
      Z=MARKER(K)-1
      Z1=(VN(K+1)-VN(K-1))*Z
*Zero, if region of neutral expansion
      IF(Z1)7,10,10
 7    VISCN(K)=-0.5*RHON(K)*Z1*Z2/DELTA1(K+1)
 10   CONTINUE
      VISCN(1)=VISCN(2)
      VISCN(N)=VISCN(N1)
££Modify viscosity if required
*Dummy routine, can be used to introduce any viscosity term
      CALL DF12VS(2)
££Introduce some smoothing
*Temporary store
      DO 999 K=1,N
      VISC2(K)=VISCN(K)/3.0
 999  CONTINUE
*Smooth the viscosity
      DO 9 K=2,N1
      VISCN(K)=VISC 2(K-1)+VISC2(K)+VISC2(K+1)
 9    CONTINUE
*Boundary extrapolation
      VISCN(1)=VISCN(2)
      VISCN(N)=VISCN(N1)
£Neutral equation of motion
*Main implicit method is used, with no iteration
££Preliminaries
      DO 32 K=1,N1
*Div Vn
      Z=(VN(K+1)-VN(K))/DELTA1(K+1)+0.5*(VN(K+1)+VN(K))/R2(K)
*Viscous pressure term
      QN(K)=-Z*VISCN(K)/RHON(K)
*Expansion
      DN(K)=Z*0.25*DELTAT
 32   CONTINUE
      QN(N)=0.0
C
*Initialize for scan
      X=0.5*DELTAT
      Z=DELTAT**2/8.0
      ALPHA(1)=0.0
      BETA(1)=0.0
      X2=2.0*DELTAT*FACHC
      ZZ4=4.0*DN(KHC)*QN(KHC)
££Scan radially outwards
      DO 41 K=2,N
C
*Is this point fully ionised?
      MKR=MARKER(K)
      GO TO (31,36),KGAS
 36   GO TO (31,33),MKR
C
*Yes, skip
 31   X4=0.0
      X5=0.0
      X6=0.0
      GO TO 35
C
*No, charge exchange
 33   X4=2.0*(S4(K)*RHO(K)+S4(K-1)*RHO(K-1))
      X5=2.0*V(K)
*Ionisation
      X6=X*(S1(K)-S1(K-1))
££Coefficients used each step (page 8.5)
```

Fig.2 **Edited COMPOSITOR Input File**. The extra lines all begin in col.1 and start with C*& or # (shown as £). Only part of the file is reproduced here.

```
C/ MODULE C2S13
C
      SUBROUTINE DF12GS
C
C 2.13 Solve neutral equations
C
C/ INSERT COMCON
C/ INSERT COMPLA
C/ INSERT COMNEU
C/ INSERT COMNUM
C/ INSERT COMESH
C/ INSERT COMWRK
C/ INSERT COMDGN
C
C.      DF12GS SOLVES THE FOLLOWING EQN.S
C.
C.           DVN      D (RHON.K.TN)    DPN
C.      RHON.--- = - --(----------) - --- + G.(V-VN)
C.           DT       DR(   M     )    DR
C.
C.      DRHON     RHON D
C.      ----- = - ----.--(R.VN) - S
C.      DT          R   DR
C.
C.      DTN                TN D           G
C.      --- = - (GAMMA-1).--.--(R.VN) - ----.(TN-TI)
C.      DT                 R  DR          RHON
C
C     References are to D.L.Fisher, COS Note 12/66
C     The neutral equations are discussed in Section 8, page 8.3
C-----------------------------------------------------------------------
C     Bypass this routine if plasma is fully ionised
      GO TO (660,100),KGAS
C
C-----------------------------------------------------------------------
CL              1.          Space/time scale parameters
C
C     Decay of neutral flux from wall, Eq. (12.7)
  100 Z1=EXPF(-UXDA*T)
      FLUXN=UXA*Z1
C     Neutral shock width
      Z2=SWN**2
C     Maximum compression rate
      Z12=0.5*ACC*RELAX
C
C-----------------------------------------------------------------------
CL              2.          Neutral viscosity
C
CL                    2.1       Von-Neumann shock term, Eq. (5.20)
      DO 211 K=2,N1
      VISCN(K)=0.0
      Z=MARKER(K)-1
      Z1=(VN(K+1)-VN(K-1))*Z
C     Zero, if region of neutral expansion
      IF(Z1)210,211,211
  210 VISCN(K)=-0.5*RHON(K)*Z1*Z2/DELTA1(K+1)
  211 CONTINUE
      VISCN(1)=VISCN(2)
      VISCN(N)=VISCN(N1)
C
CL                    2.2       Modify viscosity if required
C     Dummy routine, can be used to introduce any viscosity term
      CALL DF12VS(2)
C
CL                    2.3       Introduce some smoothing
C     Temporary store
      DO 230 K=1,N
      VISC2(K)=VISCN(K)/3.0
  230 CONTINUE
C     Smooth the viscosity
      DO 231 K=2,N1
      VISCN(K)=VISC 2(K-1)+VISC2(K)+VISC2(K+1)
  231 CONTINUE
C     Boundary extrapolation
      VISCN(1)=VISCN(2)
      VISCN(N)=VISCN(N1)
C
C-----------------------------------------------------------------------
CL              3.          Neutral equation of motion
C
C     Hain implicit method is used, with no iteration
C
CL                    3.1       Preliminaries
      DO 310 K=1,N1
C     Div Vn
      Z=(VN(K+1)-VN(K))/DELTA1(K+1)+0.5*(VN(K+1)+VN(K))/R2(K)
C     Viscous pressure term
      QN(K)=-Z*VISCN(K)/RHON(K)
C     Expansion
      DN(K)=Z*0.25*DELTAT
  310 CONTINUE
      QN(N)=0.0
C
C     Initialize for scan
      X=0.5*DELTAT
      Z=DELTAT**2/8.0
      ALPHA(1)=0.0
      BETA(1)=0.0
      X2=2.0*DELTAT*FACHC
      ZZ4=4.0*DN(KHC)*QN(KHC)
C
CL                    3.2       Scan radially outwards
      DO 360 K=2,N
C
C     Is this point fully ionised?
      MKR=MARKER(K)
      GO TO (321,320),KGAS
  320 GO TO (321,322),MKR
C
C     Yes, skip
  321 X4=0.0
      X5=0.0
      X6=0.0
      GO TO 330
C
C     No, charge exchange
  322 X4=2.0*(S4(K)*RHO(K)+S4(K-1)*RHO(K-1))
      X5=2.0*V(K)
C     Ionisation
      X6=X*(S1(K)-S1(K-1))
C
CL                    3.3       Coefficients used each step (page 8.5)
  330 Z1=AN(K)+AN(K-1)+AN1(K)+AN1(K-1)
      Z2=2.0*(RHON(K)+RHON(K-1))
      ZZ1=AN(K)+AN1(K)-AN(K-1)-AN1(K-1)
      ZZ5=4.0*DN(K)*QN(K)
      ZZ8=GA5*(AN(K)+AN1(K))
      ZZ9=GA5*(AN(K-1)+AN1(K-1))
C
      Z5=X*((ZZ1+2.0*Z1*(RHON(K)-RHON(K-1)-X6)/Z2-ZZ5+ZZ4)/DELTA2(K)
     +  -X4*X5/Z2)
      Z6=Z*(ZZ8+Z1*2.0*RHON(K)/Z2)/DELTA2(K)
     +  +X2*VISCN(K)/(Z2*DELTA2(K))
      Z7=Z*(ZZ9+Z1*2.0*RHON(K-1)/Z2)/DELTA2(K)
     +  +X2*VISCN(K-1)/(Z2*DELTA2(K))
C
      ZZ4=ZZ5
      V2D=VN(K+1)
C
CL                    3.4       Boundary points
C     Is this a hardcore run?
      GO TO(342,340),NHCORE
C     Yes
  340 IF(K-2)341,341,350
C
C     Inner boundary (hardcore only, Section 13)
  341 X2=X2*10.0
      E2=0.0
      Z7=0.0
      E1=Z6/DELTA1(K+1)+Z6/(2.0*R2(K))
      GO TO 352
C
  342 IF(K-N1)350,343,344
C     Reduce viscosity for outer interval
  343 X2=0.1*X2
      GO TO 350
C
C     Outer boundary
  344 E1=0.0
      Z6=0.0
      V2D=0.0
      GO TO 351
C
CL                    3.5       Evaluate E and F (Eq. 8.18)
  350 E1=Z6/DELTA1(K+1)+Z6/(2.0*R2(K))
  351 E2=Z7/DELTA1(K)-Z7/(2.0*R2(K-1))
  352 E3=E1+E2-Z6/R2(K)+Z7/R2(K-1)-X*X4/Z2+1.0
      F=-Z5+E1*V2D-(E3-2.0)*VN(K)+E2*VN(K-1)
C
CL                    3.6       Evaluate ALPHA and BETA
      ALPHA(K)=E1/(E3-E2*ALPHA(K-1))
      BETA(K)=(F+E2*BETA(K-1))/(E3-E2*ALPHA(K-1))
```

Fig.3 **Regenerated Version of Subroutine DF12GS.** This version is produced automatically when the file of Fig.2 is submitted as input to the COMPOSITOR. Except for the initial letter conventions it satisfies the OLYMPUS conventions of ref. [4], and being more readable than Fig.1 it can be used as the basis for further improvements.

```
  360    CONTINUE
C
C-----------------------------------------------------------------------
CL              4.         Solve for velocity, scanning inwards
C
        V2D=0.0
        K=N
  400   VN2(K)=V2D*ALPHA(K)+BETA(K)
        V2D=VN2(K)
        K=K-1
        IF (K-1) 510,510,400
C
C-----------------------------------------------------------------------
CL              5.         Effect of emission on boundary velocities
C
CL                   5.1      Outer boundary
C     Is the boundary Lagrangian?
  510   GO TO (512,511),KBA
  511   GO TO (512,520),KB
C
C     No, Boundary condition on Vn
C     Allow for neutral influx from wall, Eq. (12.8)
  512   Z1=RHON(N)*DELTA1(N)
        Z2=FLUXN*DELTAT
        VN2(N)=(VN2(N)*Z1-VNE*Z2)/(Z1+Z2)
C
CL                   5.2      Inner boundary
C     Is it a hardcore run (Section 13)?
  520   GO TO (600,521),NHCORE
C     Yes, is the boundary Lagrangian?
  521   GO TO (523,522),KBA
  522   GO TO (523,600),KBHC
C
C     No, Hardcore boundary condition on Vn
  523   Z1=RHON(1)*DELTA1(2)
        Z2=FLUXN*DELTAT
        VN2(2)=(VN2(2)*Z1+VNE*Z2)/(Z1+Z2)
C
C-----------------------------------------------------------------------
CL              6.         Temperature and density equations
C
  600   DO 650 K=KHC,N1
        MKR=MARKER(K)
        GO TO (610,620),MKR
C
CL                   6.1      Set convenient values if fully ionised
  610   AN2(K)=AJO1(K)
        RHON2(K)=RHOM1
        GO TO 650
C
CL                   6.2      Adiabatic compression
  620   Z2=(VN2(K+1)-VN2(K))/DELTA1(K+1)+0.5*(VN2(K+1)+VN2(K))/R2(K)
     +  *RZA
        DHN2=Z2*0.25*DELTAT
C
CL                   6.3      Check compression during timestep
        Z11=ABSF(DN(K)+DHN2)
        IF (Z11-Z12) 631,631,630
C     Results not accepted
  630   WRITE(NOUT,9000) Z11,K
        NO3=2
C     Set warning marker
  631   QN2=-Z2*VISCN(K)/RHON(K)
C
CL                   6.4      Compression/viscous heating (T, Eq. 8.12)
        Z1=(DN(K)+DHN2)*(QN(K)+QN2)*GA5
        Z=1.0+GA5*(DN(K)+DHN2)
C     Neutral temperature
        AN2(K)=(AN1(K)*(2.0-Z)-Z1)/Z
C
CL                   6.5      Compression/ionization (Rho, Eq. 8.1)
        Z=1.0+DN(K)+DHN2
        RHON2(K)=(RHON(K)*(2.0-Z)
     +          -S1(K)*DELTAT)/Z
C     Density cannot be negative
        RHON2(K)=MAX1F(RHON2(K),RHOM1)
C
  650   CONTINUE
C
CL                   6.6      Boundary conditions on Rho, T (sec. 12)
        CALL DF12BC(3)
C
  660   RETURN
C
 9000   FORMAT(29H  NEUTRAL COMPRESSION FACTOR=,E12.4,4H  K=,I3)
        END
```

Fig.3 Continued

```
VERSION 1* 05/AUG/82 KVR CULHAM

ARRAY DIMENSIONS
ND=51

INDEX OF COMMON VARIABLES

S 2.1 Physics control

COMMON/COMCON/

I KGAS      1 fully ionised, 2 still partial
I NHCORE    1 if hardcore run, 2 if not
I KBA       Boundary, 1 Eulerian, 2 can be Lagrangian
R T         Time
I KB        Outer boundary, 1 Eulerian, 2 Lagrangian
I KBHC      Hardcore, 1 Eulerian, 2 Lagrangian
IA MARKER(ND) 1 cell fully ionised, 2 partial

S 2.2 Plasma variables

COMMON/COMPLA/

R GA5       Gamma-1
RA RHO(ND)    Plasma density
RA V(ND)      Plasma velocity

S 2.3 Neutral variables

COMMON/COMNEU/

R UXDA      Neutral flux decay rate
R UXA       Neutral flux coefficient
R FACHC     Neutral viscosity factor
R FLUXN     Neutral flux from wall
R VNE       Velocity of neutrals from wall
RA VISCN(ND)    Neutral viscosity
RA VN(ND)       Neutral velocity
RA RHON(ND)     Neutral density
RA S4(ND)       Charge exchange rate/RHO
RA S1(ND)       Ionization rate
RA AN(ND)       Neutral temperature
RA AN2(ND)      New AN
RA RHON2(ND)    New neutral density
RA VN2(ND)      New neutral velocity

S 3.1 Numerical control

COMMON/COMNUM/

R SWN       Neutral shock width
R ACC       Maximum variable change/step
R RELAX     Maximum timestep increase/step
R RHOM1     Minimum permitted density

S 3.2 Mesh

COMMON/COMESH/

I N1        N-1
I N         Number of mesh points
R DELTAT Timestep
RA R2(ND)  Half-integral mesh points
RA DELTA1(ND) Integral space interval
RA DELTA2(ND) Half-integral space interval

S 4.1 Working space

COMMON/COMWRK/

RA VISC2(ND)    Working space
RA QN(ND)       DIV(VN)*VISC/RHON
RA ON(ND)       DIV(VN)*DELTAT/4
RA ALPHA(ND)    Working space for implicit solution
RA BETA(ND)     Working space for implicit solution
RA AN1(ND)      AN + thermal conduction
RA AJO1(ND)     AJO + thermal conduction

S 5.1 Diagnostics

COMMON/COMDGN/

I NO3       Run termination parameter
```

Fig.4 **Master Index MINDEX.** This is the input file used by the GENERATOR to construct COMMON blocks, documentation and other standard sections of the program [3].

```
C-------------------------------------------------------------------
CL                        ALPHABETIC INDEX OF COMMON VARIABLES
C
C VERSION 1* 05/AUG/82 KVR CULHAM
C
C  ACC              Maximum variable change/step              R  3.1
C  AJO1(ND)         AJO + thermal conduction                  RA 4.1
C  ALPHA(ND)        Working space for implicit solution       RA 4.1
C  AN(ND)           Neutral temperature                       RA 2.3
C  AN1(ND)          AN + thermal conduction                   RA 4.1
C  AN2(ND)          New AN                                    RA 2.3
C  BETA(ND)         Working space for implicit solution       RA 4.1
C  DELTA1(ND)       Integral space interval                   RA 3.2
C  DELTA2(ND)       Half-integral space interval              RA 3.2
C  DELTAT           Timestep                                  R  3.2
C  DN(ND)           DIV(VN)*DELTAT/4                          RA 4.1
C  FACHC            Neutral viscosity factor                  R  2.3
C  FLUXN            Neutral flux from wall                    R  2.3
C  GA5              Gamma-1                                   R  2.2
C  KB               Outer boundary, 1 Eulerian, 2 Lagrangian  I  2.1
C  KBA              Boundary, 1 Eulerian, 2 can be Lagrangian I  2.1
C  KBHC             Hardcore, 1 Eulerian, 2 Lagrangian        I  2.1
C  KGAS             1 fully ionised, 2 still partial          I  2.1
C  MARKER(ND)       1 cell fully ionised, 2 partial           IA 2.1
C  N                Number of mesh points                     I  3.2
C  N1               N-1                                       I  3.2
C  NHCORE           1 if hardcore run, 2 if not               I  2.1
C  NO3              Run termination parameter                 I  5.1
C  QN(ND)           DIV(VN)*VISC/RHON                         RA 4.1
C  R2(ND)           Half-integral mesh points                 RA 3.2
C  RELAX            Maximum timestep increase/step            R  3.1
C  RHO(ND)          Plasma density                            RA 2.2
C  RHOM1            Minimum permitted density                 R  3.1
C  RHON(ND)         Neutral density                           RA 2.3
C  RHON2(ND)        New neutral density                       RA 2.3
C  S1(ND)           Ionization rate                           RA 2.3
C  S4(ND)           Charge exchange rate/RHO                  RA 2.3
C  SWN              Neutral shock width                       R  3.1
C  T                Time                                      R  2.1
C  UXA              Neutral flux coefficient                  R  2.3
C  UXDA             Neutral flux decay rate                   R  2.3
C  V(ND)            Plasma velocity                           RA 2.2
C  VISC2(ND)        Working space                             RA 4.1
C  VISCN(ND)        Neutral viscosity                         RA 2.3
C  VN(ND)           Neutral velocity                          RA 2.3
C  VN2(ND)          New neutral velocity                      RA 2.3
C  VNE              Velocity of neutrals from wall            R  2.3
C
```

```
C-------------------------------------------------------------------
CL                  C2.3     Neutral variables
C VERSION 1* 05/AUG/82 KVR CULHAM
      COMMON/COMNEU/
     R   AN     ,   AN2    ,   FACHC  ,   FLUXN  ,   RHON   ,   RHON2  ,
     R   S1     ,   S4     ,   UXA    ,   UXDA   ,   VISCN  ,   VN     ,
     R   VN2    ,   VNE
      DIMENSION
     R   AN(51),    AN2(51),   RHON(51), RHON2(51),           S1(51),
     R   S4(51),    VISCN(51),          VN(51),    VN2(51)
```

Fig.5 **Examples of GENERATOR Output.** Two automatically constructed files are shown. (a) Alphabetic Index of Variables INDVAR. (b) COMMON Block COMNEU.

```
C-------------------------------------------------------------------
CL              5.         Effect of emission on boundary velocities
C
CL                   5.1      Outer boundary
C     Is the boundary Lagrangian?
  510    IF((KBA.NE.1).AND.(KB.NE.1)) GO TO 520
C     No, Boundary condition on Vn
C     Allow for neutral influx from wall, Eq. (12.8)
         Z1=RHON(N)*DELTA1(N)
         Z2=FLUXN*DELTAT
         VN2(N)=(VN2(N)*Z1-VNE*Z2)/(Z1+Z2)
C
CL                   5.2      Inner boundary
C     Is it a hardcore run (Section 13)?
  520    IF(NHCORE.EQ.1) GO TO 600
C
C     Yes, is the boundary Lagrangian?
         IF((KBA.NE.1).AND.(KB.NE.1)) GO TO 600
C     No, Hardcore boundary condition on Vn
         Z1=RHON(1)*DELTA1(2)
         Z2=FLUXN*DELTAT
         VN2(2)=(VN2(2)*Z1+VNE*Z2)/(Z1+Z2)
C
```

Fig.6 **Code Modifications.** After the code has been clarified using the COMPOSITOR it can be further improved by modification of the executable statements.