Communications
from the
University of London
Observatory
No. 75

ATLAS5* AT THE UNIVERSITY OF LONDON (SAM1)

I: PROGRAMMING AND IMPLEMENTATION

by

S.L. Wright and J.D. Argyros

* a program for the calculation of model stellar atmospheres

November, 1975
Department of Physics & Astronomy
University College London

TABLE OF CONTENTS

	ABSTRACT	1
	PREFACE	3
	NOTE	4
1	INTRODUCTION	5
2	ERRORS IN THE ORIGINAL CODE	7
3	SAM1 ON IBM AND CDC MACHINES	9
4	IMPLEMENTATION OF AN EFFECTIVE OVERLAY SCHEME	12
5	MODIFICATION OF THE STOP/RESTART PROCEDURE	17
6	GENERALISATION OF THE INPUT/OUTPUT PROCEDURES	19
7	ALTERATIONS TO THE FORMAT OF THE OUTPUT	22
8	THE EXTENDED OUTPUT FACILITY	23
9	CHANGES TO SUBROUTINE CONVEC	31
10	THE HANDLING OF OPACITY TABLES	32
11	ERROR MESSAGES	33
12	OPTIMISATION OF THE CODE	35
13	SUNDRY ALTERATIONS	40
14	CHANGES TO THE UTILITY PROGRAMS	42
15	IMPLEMENTATION OF SAM1 ON THE IBM 360/65	45
16	IMPLEMENTATION OF SAM1 ON THE CDC 6600	51
	Appendix 1	58
	Appendix 2	59
	Appendix 3	60
	References	61

PREFACE

It is a truism that any user of a computer program written by someone else, no matter how well it is written, finds aspects of it not to his liking. The wise user, perhaps, is the one who leaves the program as it is and gets on with his own work, but sometimes, even he may decide that he really has to make some alterations.

We obtained the general-purpose stellar atmosphere program ATLAS5, written by R.L. Kurucz, with the intention of using it as a test-bed, as it were, to examine the effects of various sources of continuous opacity on the structure of the atmosphere and on the emergent intensity. This meant that we wanted to be able to change the opacity routines, and to have output in a form convenient for our purposes. Also, the program would not run on our IBM 360/65 without modification.

Having made such 'necessary' changes, we decided to make further 'improvements'. We have made more changes than are really needed to operate the program, but we are confident that the changes make it more convenient for us to use as well as increasing its operating efficiency.

This document has been prepared primarily for ourselves, as a record of the changes which have been made. We feel that it may also be of interest to other users of the ATLAS5 program. To avoid confusion in our own minds we have called the modified version SAM1. A companion report by S.L. Wright contains a guide to operating the program.

We are grateful to Dr. Kurucz and his colleagues for supplying us with a copy of the original program.

W.B. Somerville

NOTE

A copy of the program described in this report can be obtained by sending a magnetic tape to:

S.L. Wright
Department of Physics & Astronomy
University College London,
Gower Street,
London WC1E 6BT

Please state whether the tape is to be written on an IBM or a CDC machine.

1: INTRODUCTION

ATLAS5 is a very general computer program for calculating 'classical' model atmospheres. A complete listing, and details of the program and its operation have been given by Kurucz (1970). We obtained a copy of ATLAS5 in November 1971, and since that time we have made a number of changes to it. This report describes only the changes that have been made to the programming. Changes in the physics that have been put into the program will be described in a later publication.

Changes to the program have been made for the following reasons:

1) to correct errors in the original version of the program,
2) to enable the program to be run on the IBM 360/65,
3) to implement an overlay scheme,
4) to improve the input/output flexibility,
4) to optimise explicitly some of the code for faster execution,
5) to satisfy demands made by colleagues for alternative output formats.

The improved performance of the program as a result of the alterations has occurred in several areas. Of these, the changes made to the output (chapters 7, 8, and 11) will be the ones most easily noticed by a user familiar with ATLAS5. In the simplest cases, improved error messages have been included; elsewhere, completely new formats for extended output have been added. Facilities have also been included to allow the user to read/write easily data from/to mass storage. The logical structure of ATLAS5 has been changed (chapters 3 and 4) to allow the successful implementation of overlay schemes on both the CDC 6600 and IBM 360/65 computers. Several techniques have been used to increase the speed of the program. These have resulted in execution times from 20% to 30% faster than the original program. The most important of these changes was the reformulation of the interpolation procedure to give greater accuracy and faster execution time.

To distinguish the two versions of the program, we have called our version SAM1 for Stellar Atmosphere Model version 1. Because of the large number of changes that have been made, it is impossible to describe them in detail here. Therefore we have described only the general nature of the changes and the approximate positions in the code where they have been made. As well as giving the changes, we describe the details of the implementation of SAM1 on the CDC and IBM machines of the University of London. It is assumed that the reader has access to a copy of the report by Kurucz (1970) and is familiar with its contents.

A user's guide to the program in its modified form is available (Wright, 1975).

2: ERRORS IN THE ORIGINAL CODE

Kurucz himself has reported some corrections to ATLAS5 in his memo to ATLAS5 users (July 15, 1973). These include changes to the temperature correction procedure, and the correction of statement 30 in subroutine HOT.

The errors in the original code which we have detected to date, most of which are relatively trivial, are summarised below.

a) If the PRINT 1 option is in effect the program skips the calculation of the fractional convective flux, thus the data under the heading FRACTION CONV FLUX in the summary table is in fact the absolute convective flux. The code was changed so that the fractional convective flux is calculated for all PRINT options.

b) After the call to DUMMYR in subroutine READIN, control should be returned to statement 98, and not to statement 97. The error arose because the variable CARD is local to READIN so that on return from DUMMYR it still contains the values it held prior to entering DUMMYR. However the COMMON block FREE which controls the free-field read from this array will now hold values pertaining to the last read executed in DUMMYR.

c) The calculation of GFLUX(J) in subroutine TCORR uses the variables DEL and D. As these variables are not arrays, only GFLUX(NRHOX) will be calculated correctly. To overcome this we store the quantity DEL*(1.+D/(D+DEL)) in an array DDEL(J), and use this in the calculation of GFLUX(J). This correction does not seem to have much effect on the model calculation.

d) Some unnecessary calculations are performed when IFSCAT=Ø. These have been avoided.

There are a number of disagreements between the text of Kurucz (1970) and the coding of the program. This has led to the following changes:

e) The tables of rates for atomic hydrogen should be printed for the PRINT 1 option (Kurucz, 1970 p.154), but only the heading of the table was in fact printed. We have decided to suppress this table completely for the PRINT 1 option.

f) One of the values in the table of HOT opacities had evidently been revised in the text but not in our copy of ATLAS5. The program was changed to agree with the text (Kurucz, 1970 p.239).

g) The formula used to calculate the number density of H_2 in subroutine H2RAYOP does not agree with Kurucz 1970 p.88, or the data on pp.59-60. The formula was changed to agree with the text.

h) The cut-off frequency for the H Rayleigh scattering opacity given by Kurucz, 1970 p.80, is different to that used in the program. The selection of a cut-off frequency is a difficult problem. A lot of our own work with SAM1 is concerned with pure hydrogen atmospheres, and if the cut-off frequency is left as it is, a completely unphysical kink is produced in the emergent flux distribution. We decided to set the cut-off at the Lyman jump.

i) In Kurucz, 1970 p.29 it is stated that no allowance is made for the integral from t_n to infinity in calculating the intensity integral. In fact the method does allow for this contribution, but does not include the contribution from zero to t_1. We have completely recoded this calculation (see chapter 12).

j) Kurucz, 1970 p.67 states that the height is arbitrarily zero at the first RHOX. In fact height is taken as zero where RHOX is zero. We have changed this so that height is zero where the Rosseland mean optical depth is 1.

N.B.

There are a number of errors, most of them apparently typographical, in Kurucz (1970), so the user should check equations before using them.

3: SAM1 ON IBM AND CDC MACHINES

ATLAS5 was written for and runs on a CDC machine. While it is true that the author made allowances for 'any unfortunate who has a 360' (Kurucz, 1970), in the event these proved insufficient. The incompatibilities between the two machines (IBM 360/65 and CDC 6600) are of such a magnitude that a large number of changes are needed to make the program run on an IBM machine. The incompatibilities which exist between the two versions of SAM1 for IBM and CDC machines arise for the following reasons:

a) Word length differences.

The CDC single precision word is 60 bits long, whereas the IBM single precision word is only 32 bits long. The program was written in single precision and runs satisfactorily on the CDC machine. However, because of the numerical complexity of a model stellar atmosphere calculation, truncation and round-off errors build up quite quickly in the smaller IBM word. The result of this is that significantly different numbers are produced on the IBM as compared to a CDC run with the same model parameters. Therefore to run the programs successfully on an IBM machine, it is necessary to make most of the variables double precision. This entails changing all the FORTRAN supplied functions to their double precision forms, changing E-format constants to D-format constants, altering FORMAT statements, etc. This proves to be the major source of the incompatibilities between the two versions.

b) Initialisation of variables in named COMMON blocks.

i) CDC FORTRAN allows variables in named COMMON blocks to be initialised in subroutines and only allows one BLOCK DATA subroutine per program.

ii) IBM FORTRAN does not allow variables in named COMMON blocks to be initialised in subroutines, but one can have any number of BLOCK DATA subroutines. However, a complete COMMON block must be initialised in only one BLOCK DATA

subroutine.

Thus for running on IBM machines all the subroutines that initialise named COMMON blocks (BLOCKE, BLOCKR, BLOCKJ, BLOCKH) must be changed to BLOCK DATA subroutines, and COMMON /MATX/ must be split into two COMMON blocks:

```
COMMON /MATJ/ CJ(1849)
COMMON /MATH/ CH(1849),XTAU(43),NXTAU
```

c) Size of subroutines and speed of compilation.

The IBM FORTRAN compilers do not like large subroutines with lots of initialisation using DATA statements. Therefore the array NNN in subroutine PFSAHA was put into a COMMON block and all the DATA statements were put into a BLOCK DATA subroutine.

d) Problems with the exponent.

The maximum and minimum values of the exponent for floating point numbers are very different for the two machines: ± 76 for IBM, and ± 300 for CDC. Thus while it is rare that exponent underflow or overflow occurs on a CDC machine, it is much more likely to happen on an IBM machine. To suppress exponent error messages and allow the calculation to continue it is necessary to make the following initialisation call to a system subroutine on the IBM machine:

```
CALL ERRSET (nume,256,-1,1)
```

where nume is the IBM error number. Exponent overflow, underflow and division by zero have been suppressed in this way. All test calculations performed on both machines have given identical results, indicating that the occurrence of these 'errors' is not significant.

e) Equivalence of integer and real variables.

The user should be careful when equivalencing integer and real variables on the IBM machine because the lengths of the two variable types are different.

Because of the incompatibilities mentioned above, it is necessary to have a different version of the program for each

machine. However a master version of the program has been written which can be pre-processed by a FORTRAN program to produce a working version of the program for either machine (see chapter 15).

The punched card output from the IBM version is in D-format, thus it cannot be read by the CDC version of the original program. This prompted us to modify subroutines IWORDF and FREEFF so that they could cope with E- or D-format and the differences between BCD and EBCDIC card codes. The punched card output from one version of the program can now be read directly by the other version without modification.

4: IMPLEMENTATION OF AN EFFECTIVE OVERLAY SCHEME

Introduction

The process of calculating a model stellar atmosphere is well suited to an overlay scheme, because the calculation can be split up into well-defined sections which can be executed sequentially. The scheme described in this chapter was originally designed for the IBM 360/65 to reduce the core requirement of the program, and so, to improve the turn-around time. Later this scheme was transferred to the CDC 6600 as described at the end of this chapter.

IBM 360/65 Overlay Scheme

The overlay scheme proposed by Kurucz, 1970 p.162, is all right in principle and will work for most cases. However it will not work if molecules are included in the calculation as explained in section b) below. The changes required to implement the scheme are as follows:

a) All the calls to subroutine POPS in the block KAPP were removed and put into the main program. The number densities of the species required by subroutines in block KAPP were passed by means of the COMMON blocks:

```
COMMON /COOL/ XNFPC(4Ø),XNFPMG(4Ø),XNFPAL(4Ø),XNFPSI(4Ø)
COMMON /LUKE/ XNFPN(4Ø),XNFPO(4Ø),XNFPM2(4Ø),XNFPS2(4Ø),
              XNFPC2(4Ø)
COMMON /HOT/ XNFP(4Ø,21)
```

b) When an overlay segment is called into core, a copy of the original absolute code of the segment is loaded. Thus the values of all local variables which were established during the previous execution of the segment are lost. It follows that Kurucz's overlay scheme will not work if molecules are included.

The solution is to put all the variables that need to be saved into COMMON blocks:

```
      COMMON /MOLEQ/ EQUIL(6,1ØØ),CODE(1ØØ),KCOMPS(1ØØ),
                     LOCJ(1ØØ),NUMMOL
      COMMON /MOLEQU/ IDEQUA(1ØØ),NEQNEQ,NEQUA,NEQUA1
      COMMON /READ/ IREAD
```

These COMMON blocks must be included in subroutine MOLEC, NMOLEC and in the main program.

c) For convenience the array XNMOL was put into a COMMON block:

```
      COMMON /NMOLEQ/ XNMOL(4Ø,1ØØ)
```

This COMMON block has been put only into subroutines MOLEC and NMOLEC. If the program is changed so that the molecular number densities are needed in another block, e.g. to calculate molecular opacities, then this COMMON block will have to be included in the main program, which will increase substantially the core requirement of the program.

d) The local variable ITEMP1 which occurs in all the opacity subroutines was put into the COMMON block TEMP and all the initialising DATA statements were removed.

e) Because of the modifications given by Kurucz (1973), the subroutine TTAUP is now called from subroutine TCORR. Thus TTAUP must be put in the root segment of the overlay.

Given the above changes, the overlay version of SAM1 runs correctly. However the scheme does not use core as efficiently as possible because parts of subroutines TCORR, STATEQ, PUTOUT, ROSS, and RADIAP are resident in core all the time. As these parts of the subroutines are only required to finish off an iteration, they can be made into separate subroutines and put

into one block FINISH. The main motivation for this reorganisation is that the extended output facility described in chapter 8 needs substantial extra coding. If this coding were put into the root segment the core requirement would increase substantially. With this reorganisation all the extra coding goes into a new subroutine, OUTPUT, which is put into block FINISH. The changes required were as follows:

f) All the variables containing frequency integrals were put into COMMON blocks.

```
COMMON /TEMCOR/ OLDT1(4Ø),T1(4Ø),RJMINS(4Ø),RDABH(4Ø),
                RDIAGJ(4Ø)
COMMON /RATES/ QRADIK(4Ø,6),QRADKI(4Ø,6),DQRAD(4Ø),
               DQRD(4Ø),QRDKHM(4Ø),QRDHMK(4Ø)
```

g) The first parts (up to statement number 3Ø) of subroutines ROSS and RADIAP were integrated into a new subroutine RADOSS. The calculation of FLXRAD has been moved from subroutine TCORR and put into this subroutine.

h) The last part of the main program was made into a new subroutine FINISH. The subroutines HIGH and TURB, and the last parts of subroutines ROSS and RADIAP were integrated into this new subroutine.

i) The last parts of subroutines TCORR, STATEQ, and PUTOUT were made into new subroutines FTCORR, FSTAT, and OUTPUT respectively.

j) All the new subroutines were put into block FINISH.

The final overlay scheme for SAM1 is given on page 16. If the core requirement of this scheme is too large for the user's computer, then it is possible to overlay segments JOSH and KAPP. The only change required in the program is to move the statement: ITEMP1=ITEMP outside the frequency loop in the

main program. This scheme is not recommended as the temperature dependant part of the opacity will have to be calculated at every frequency, leading to a large increase in the execution time of the program.

CDC Overlay Scheme

The overlay scheme described above is set up using the IBM Job Control Language (see chapter 15), and no changes are required to the FORTRAN itself. In contrast to this, the standard overlay system of the CDC SCOPE operating systems is unsophisticated. To use this system changes are required to the FORTRAN, and there are fairly stringent restrictions on how different overlay segments can communicate with each other. An overlay version of SAM1 using this system has been developed, and the details are available on request.

With recent versions of the SCOPE operating systems (SCOPE 3.4 or later) a segmented loader has been provided. Using this program the overlay problem on the CDC machine reduces practically to the same problem encountered on the IBM machine, see chapter 16.

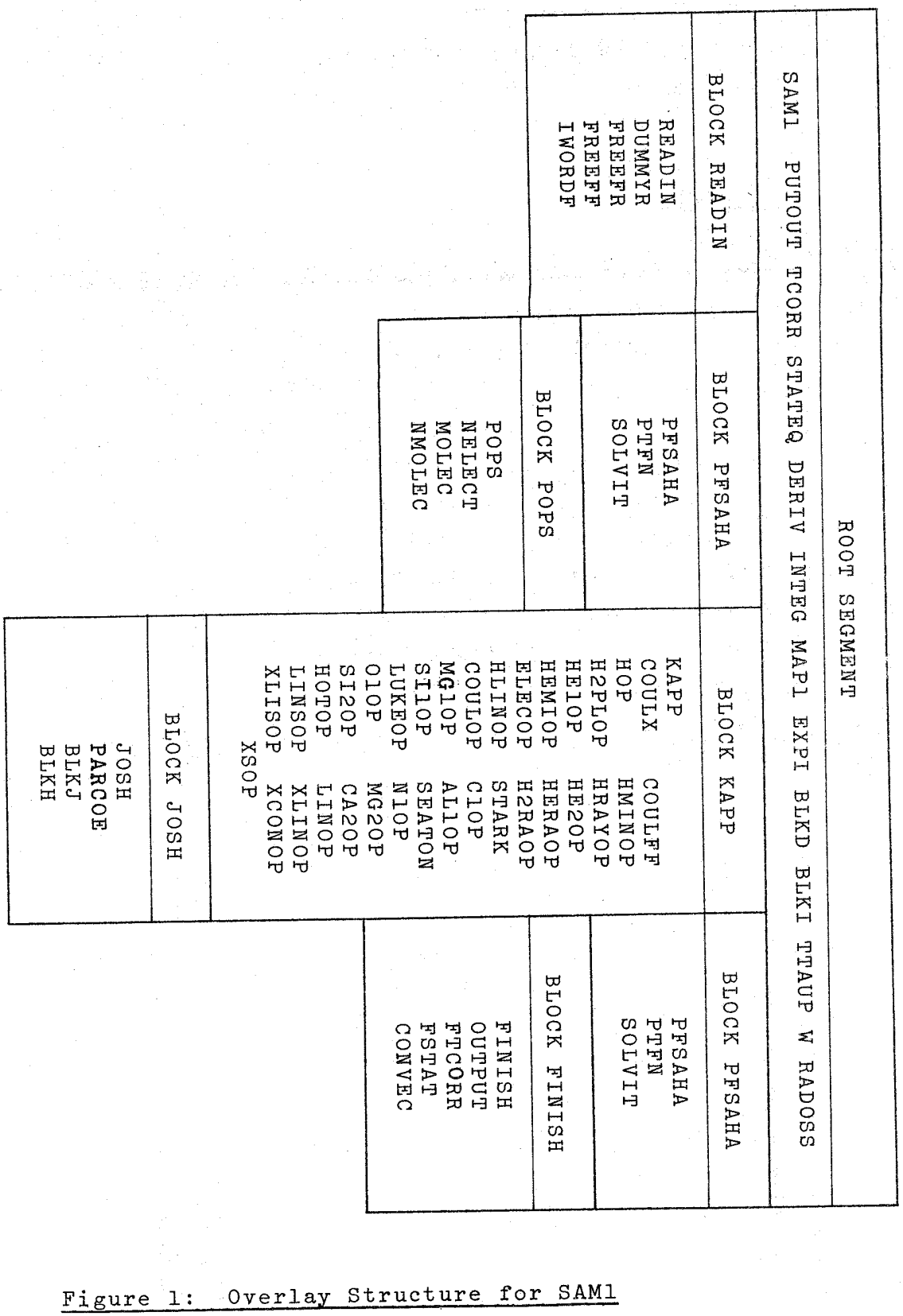


Figure 1: Overlay Structure for SAM1

5: MODIFICATION OF THE STOP/RESTART PROCEDURE

There are two approaches one can take in running model atmosphere programs:

a) To run the program with the full number of iterations thought necessary, and to punch out the model after the last iteration. More iterations can then be performed if necessary.

b) To run the program several times in succession, with a small number of iterations in each run, until a converged model is produced.

The disadvantages of the first approach are that computer time may be wasted performing unnecessary iterations, and that if the computer 'crashes' during the run the model must be started from the beginning again. The disadvantage of the second approach is that with the original version of the program a lot of unnecessary cards are produced in obtaining a converged model. The modifications to the program discussed in the next chapter remove this disadvantage, and make it so easy to restart a model calculation that we always use the second approach. This approach also has the advantage that the calculation is monitored after every few iterations, so that deviations from the expected rate of convergence, and the numerical stability of the model, can be followed.

Unfortunately, as the program stood, the first few iterations after restarting were wasted in establishing approximately the internal configuration existing at the last iteration before the calculation was stopped. The empirical speeding-up procedure used for non-convective calculations requires the sign of the temperature correction, at each depth, from the previous iteration; but when the program was stopped this information was not available in the restart data set. Consequently, on restarting, the speeding-up procedure could not be applied, the rate of convergence was slowed and subsequent iterations produced values different from those from a single continuous run.

By including the variable OLDT1 in the data set output by the program this problem no longer arises. Changes were required in OUTPUT to punch out the additional information, and in READIN to read it is again. The variable OLDT1 is passed to all the necessary subroutines through the COMMON block TEMCOR:

```
COMMON /TEMCOR/ OLDT1(4Ø),T1(4Ø),RJMINS(4Ø),RDABH(4Ø),
                RDIAGJ(4Ø)
```

If a model is to be calculated in more than one run it is desirable to have iteration numbers on the output which run in sequence; in ATLAS5, numbering starts from 1 each time the model is restarted. Also the program must know whether or not it can apply the empirical speeding-up procedure. So we tell the program not only the number of iterations it has to perform, but also the number of the first iteration to be performed in this particular run. If SAM1 is told to start from iteration number 1, the empirical speeding-up procedure is not used in the first iteration, but the procedure is used if it is to start from any other iteration number. We give SAM1 this information with a slightly changed ITERATIONS control card. The format of the card is now:

ITERATIONS n1 STARTING AT n2

The first number, n1, gives the total number of iterations to be attempted. The second number, n2, gives the number of the first iteration to be performed in this run. E.g.

ITERATIONS 2Ø STARTING AT 12

This card tells SAM1 to calculate up to iteration number 2Ø starting at iteration number 12, i.e. 9 iterations will be performed in this run. In our current implementation the maximum iteration number allowed is 3Ø. It is necessary to change READIN to recognise the modified control card. The iteration loop in the main program now needs to run from ITERLO to NUMITS. The COMMON block ITER has been changed to:

```
COMMON /ITER/ IFPRNT(3Ø),IFPNCH(3Ø),NUMITS,ITERLO,ITER
```

6: GENERALISATION OF THE INPUT/OUTPUT PROCEDURES

Although ATLAS5 has a very flexible input scheme, the user is limited to reading information from only one logical unit. Effectively this means that punched cards must be used for input and for output, as at most institutions, this is the medium easiest to handle when additions or changes need to be made to a data set, before reading the information back into the program. We therefore decided to change the program to allow input to be taken from a number of logical units. The scheme that was devised requires only a small number of changes to READIN, FREEFR, and DUMMYR, and it works as follows:

When the program is run, READIN starts reading control cards from logical unit 5. On encountering a control card of the form:

MODEL READ FROM UNIT nl

READIN reads control cards from logical unit nl until a BEGIN card is read. This BEGIN card causes READIN to stop reading from unit nl, and continue reading from unit 5, starting at the card after the MODEL control card. nl can be any logical unit that is defined by the Job Control Cards including unit 5. When the program is reading data initiated by a MODEL control card, any code words that are not recognised by READIN (or DUMMYR) are ignored and the run is not terminated. Thus using the MODEL control card one can read in the data produced by any of the PUNCH options. If, for example, one had produced a converged model atmosphere with the PUNCH 2 option and wanted to scale this model to a new effective temperature, then the following control cards could be used:

```
                MODEL  5

cards          FLUX ...
output          :
from
PUNCH2         TEFF ...  GRAVITY  ...
option          :
               BEGIN        ITERATION  15  COMPLETED

               SCALE ...

               ITERATIONS  ...  PRINT  ...

               BEGIN

               END
```

The cards beginning with FLUX are ignored by READIN. Any number of MODEL control cards can appear in the input data set. Thus one could, for example, store frequency sets and opacity tables on disk files, and read in the required information by including the appropriate MODEL control cards in the input data set.

It was also decided that SAM1 should output a restart data set after each iteration to ensure that a model could be restarted from the latest available iteration. Therefore OUTPUT was changed so that the output from the PUNCH 1 option is written to logical units 12 and 13 alternately after successive iterations. A message is printed below the summary table for each iteration indicating which logical unit the restart data has been written to. Two logical units are used to ensure that under any circumstances, data from the last or last but one iteration is available intact. We normally have logical units 12 and 13 as system resident disk files, in which case this system provides an extremely quick and convenient restart facility. For example, suppose one does 5 iterations on a model and decides that 5 further iterations are required for satisfactory convergence. One looks at the output to see where the latest restart data has been written (unit 12 in this case) and then submits a job with the following control cards:

```
MODEL READ FROM UNIT 12

ITERATIONS 10 STARTING AT 6

PRINT ... PUNCH ...

BEGIN

END
```

7: ALTERATIONS TO THE FORMAT OF THE OUTPUT

1) It may be required to analyse the results from the program on a different computer. Therefore the output should, as far as possible, be in a form readily accepted by any machine. This means that only a subset of all characters should be used: A to Z, Ø to 9, -,. , and that numbers should be in F format as far as possible. This leads to the following changes:

a) The surface flux is now output in the format:

FLUX 9117.638 -6.Ø752Ø

where the first number is the wavelength in nanometers, and the second number is the $\log_{10}$(surface flux), H_ν .

b) The surface intensity is now output in the format:

INTENSITY 9117.638 1.ØØØ -7.761452 etc.

where the first number is the wavelength in nanometers, and the following pairs of numbers are the μ's and their corresponding $\log_{10}$(surface intensity).

2) Because of the modifications described in chapter 4, one needs an extra card at each depth to hold the variable OLDT1. The rest of the card is used to output some other variables: FLXERR, FLXDRV, BHYD(I),I=1,6, BMIN.

Because the departure coefficients are now output for every model, one does not need the special punchout for NLTEON=1.

3) A new control word has been added which allows SAM1 to read in cards punched by ATLAS5. The control word is ATLAS and should be placed in front of an ATLAS5 data deck. This option is cancelled by the execution of the BEGIN control word.

8: THE EXTENDED OUTPUT FACILITY

Introduction

ATLAS5 prints out the model only at the depth points used in the calculation. It is often convenient to have a model atmosphere tabulated at suitable depths on a mean or monochromatic optical depth scale. It was therefore decided to write code to map the model onto a mean or monochromatic optical depth grid specified by the user. The only restriction is that the frequency of the monochromatic depth scale must be one of the frequencies used in calculating the model.

The PRINT 3 and 4 options of ATLAS5 generate an unwieldy amount of information which is difficult to digest. Therefore it was felt desirable to be more selective in the data produced. The original output format has been retained as an option, however, as it can be useful when debugging a new opacity subroutine, for example. The extended output options are brought into play by new PRINT and PUNCH options. Thus the PRINT 13 option prints the variables associated with PRINT 3, but only at depths and frequencies specified by the user. The PRINT 14 option (corresponding to PRINT 4) prints an opacity table at depths and frequencies specified by the user for all the opacity sources included in the calculation. To achieve this reorganisation of the data, two scratch files need to be defined on logical units 1Ø and 11.

The additional information required by the extended output options is read in using a version of the subroutine DUMMYR. Code has also been written to allow the information to be punched out for subsequent analysis.

Additional Control Cards

The control cards required by the extended output option must appear between the two special control cards:

CALL DUMMYR

which causes the program to transfer control to the subroutine DUMMYR, and:

RETURN

which passes control back to READIN. The only control cards that can appear between these two are:

a) SAMPLE n1 n2 n3 ...

The control word SAMPLE requires a list of n1 frequency identifiers (maximum of 500), n2, n3, ... which correspond to the frequencies that the user wants in the output. If PRINT 13 or 14 is selected, this control word with its list of frequency identifiers (even if only one) must appear.

b) DEPTH n1 n2
n3 n4 ....

The control word DEPTH requires a list of n1 (maximum of 40) optical depths, n3, n4, ... at which the user wishes the information to be output. n2 is the frequency identifier which corresponds to the frequency of the monochromatic optical depths. If n2=Ø, the optical depths are given on a Rosseland mean optical depth scale.

PRINT and PUNCH Options for Extended Output

a) PRINT 12

Output as for PRINT 2 plus the model mapped onto the optical depth grid specified by the DEPTH control card(s).

b) PRINT 13

As for PRINT 12, plus tables giving the radiation field parameters mapped as specified by DEPTH, and at frequencies specified by the SAMPLE control card(s).

c) PRINT 14

As for PRINT 12, plus tables giving the $\log_{10}$(opacity) at each depth and frequency (specified on the DEPTH and SAMPLE

control cards). One table is produced for each opacity source included in the calculation.

PUNCH 3

Output as for PUNCH 2, plus a punchout of the information given by the PRINT 13 option.

PUNCH 4

Output as for PUNCH 2, plus a punchout of the information given by the PRINT 14 option.

N.B.

If PUNCH 3 or 4 is specified the corresponding PRINT option (13 or 14) must also be specified.

Sample Calculations

In this section some sample calculations are described giving details of the output associated with the extended output options, PRINT 12, 13, and 14. The output from these options is printed after the normal output from SAM1.

a) The first example is a model of the star Procyon, illustrating the PRINT 13 option. The following control cards were included in the input data set defining the model:

```
CALL DUMMYR
SAMPLE  4   23  3Ø  36  42
DEPTH  16  DEPTHS AT FREQUENCY IDENT  3Ø
1.D-2  1.585D-2  2.512D-2  3.981D-2  6.31ØD-2
1.D-1  1.585D-1  2.512D-1  3.981D-1  6.31ØD-1
1.     1.585     2.512     3.981     6.31Ø     1.D1
RETURN
```

The first page of the output appears as shown on page 27. After the normal output from SAM1, a table giving the model mapped onto the specified depth grid is printed (see page 28). In this example the frequency identifier 3Ø corresponds to a wavelength of 5ØØ nm. This table will be printed if any of the extended output options is in effect. Page 28 also shows a part of the output associated with PRINT 13. In this table TAUSTD is the depth specified by the user, and TAUNU is the corresponding optical depth at the frequency at which the variables are tabulated. If the frequency at which the variables are tabulated is the same as the frequency of the monochromatic optical depth grid then TAUNU is the same as TAUSTD so we print TAUROSS instead of TAUNU.

b) The second example is a model for the Sun. Page 29 is the first page of the output. The optical depth scale in this case is a Rosseland mean optical depth scale. Page 30 shows the model mapped onto this depth grid, and also gives a part of the output associated with the PRINT 14 option.

THE FOLLOWING DATA HAS BEEN READ BY DUMMYR

FREQUENCY IDENTIFIERS 4
23 30 36 42

16 OPTICAL DEPTHS AT FREQUENCY IDENTIFIED BY 30
1 1.000E-02 2 1.585E-02 3 2.512E-02 4 3.981E-02 5 6.310E-02 6 1.000E-01 7 1.585E-01 8 2.512E-01
9 3.981E-01 10 6.310E-01 11 1.000E+00 12 1.585E+00 13 2.512E+00 14 3.981E+00 15 6.310E+00 16 1.000E+01

TEFF 6450. LOG G 4.00 LTE

TITLE LTE MODEL ATMOSPHERE FOR PROCYON.

XSCALE 1.000 H .900 HE .100
LI-11.60 BE -9.60 B -9.20 C -3.50 N -4.12 O -3.28 F -6.60 NE -3.50 NA -6.18 MG -4.57
AL -5.65 SI -4.50 P -6.62 S -4.84 CL -6.60 AR -5.30 K -7.00 CA -5.72 SC -9.01 TI -7.55
V -8.13 CR -6.58 MN -7.17 FE -4.50 CO -8.40 NI -6.97 CU -7.30 ZN -7.63 GA -9.11 GE -8.73
AS -9.70 SE -8.80 BR -9.40 KR -8.80 RB -9.42 SR -9.23 Y -9.60 ZR -9.60 NB-10.30 MO-10.00
TC-20.00 RU-10.40 RH-11.20 PD-10.70 AG-11.30 CD -9.98 IN-10.34 SN-10.34 SB-10.40 TE-10.00
I -10.60 XE-10.00 CS-10.90 BA-10.15 LA-10.60 CE-10.40 PR-11.20 ND-10.50 PM-20.00 SM-11.00
EU-11.30 GD-10.90 TB-11.60 DY-10.80 HO-11.50 ER-11.10 TM-11.90 YB-10.90 LU-11.70 HF-11.40
TA-11.70 W -10.90 RE-11.40 OS-10.70 IR-10.80 PT-10.40 AU-11.30 HG-11.10 TL-11.50 PB-10.15
BI-11.30 PO-20.00 AT-20.00 RN-20.00 FR-20.00 RA-20.00 AC-20.00 TH-11.70 PA-20.00 U -12.00
NP-20.00 PU-20.00 AM-20.00 CM-20.00 BK-20.00 CF-20.00 ES-20.00

H1 1 H2PLUS 1 HMINUS 1 HRAY 1 HE1 1 HE2 1 HEMINUS 0 HERAY 0 COOL 1 LUKE 1
HOT 0 ELECTRON 1 H2RAY 0 HLINES 1 LINES 0 LINESCAT 0 XLINES 0 XLSCAT 0 XCONT 0 XSCAT 0

IFCORR 1 IFPRES 1 IFSURF 0 IFSCAT 1 IFCONV 1 MIXLTH 1.00 IFMOL 0
IFTURB 0 TRBFDG 0.00 TRBPOW 0.00 TRBSND 0.00 TRBCON 0.00
ITERATIONS 11 TO 11
IFPRNT 1 1 1 1 1 1 1 1 1 1 13 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
IFPNCH 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

FREQID PROCYN NUMNU 86 NULO 1 NUHI 76

1	3.288047E+13	1.644025E+13	23	4.496880E+14	9.201967E+12	45	1.050000E+15	4.977288E+13	67	1.900000E+15	8.480297E+13
2	3.288053E+13	1.542103E+12	24	4.535439E+14	2.561294E+12	46	1.100000E+15	4.989606E+13	68	1.972315E+15	2.660077E+13
3	3.670000E+13	3.998834E+12	25	.567819E+14	2.088044E+12	47	1.150000E+15	5.223402E+13	69	1.972318E+15	1.424415E+13
4	4.059321E+13	6.614761E+12	26	4.605108E+14	1.151333E+13	48	1.192579E+15	1.569611E+13	70	2.020000E+15	6.964705E+13
5	5.137578E+13	1.346479E+13	27	4.800000E+14	1.886007E+13	49	1.192581E+15	2.172816E+13	71	2.076098E+15	1.989230E+13
6	6.710306E+13	1.356110E+13	28	5.000000E+14	3.212309E+13	50	1.250000E+15	6.475939E+13	72	2.076102E+15	4.743675E+13
7	9.133472E+13	4.428176E+13	29	5.500000E+14	6.107234E+13	51	1.300000E+15	4.548211E+13	73	2.200000E+15	1.277701E+14
8	1.315219E+14	1.517814E+13	30	5.595850E+14	2.797748E+13	52	1.350000E+15	5.000737E+13	74	2.300000E+15	1.224355E+14
9	1.315221E+14	7.266987E+12	31	6.105754E+14	4.961573E+12	53	1.400000E+15	5.257096E+13	75	2.419598E+15	4.585766E+13
10	1.506495E+14	2.048866E+13	32	6.167301E+14	5.426781E+12	54	1.442999E+15	1.587131E+13	76	2.419602E+15	1.814854E+13
11	1.742980E+14	3.445283E+13	33	6.232692E+14	1.321259E+13	55	1.443001E+15	3.614600E+13	77	2.465399E+15	4.501286E+13
12	2.055029E+14	1.177264E+13	34	6.500000E+14	3.993314E+13	56	1.515290E+15	3.614600E+13	78	2.500000E+15	4.846377E+13
13	2.055033E+14	9.324904E+12	35	7.000000E+14	5.150270E+13	57	1.515294E+15	1.235988E+13	79	2.600000E+15	1.481648E+14
14	2.300000E+14	2.856851E+13	36	7.494810E+14	5.596898E+13	58	1.550000E+15	4.714918E+13	80	2.725397E+15	4.601001E+13
15	2.600000E+14	3.681093E+13	37	7.980397E+14	1.819702E+13	59	1.600000E+15	3.869734E+13	81	2.725403E+15	2.259495E+13
16	3.000000E+14	3.371661E+13	38	7.980413E+14	1.198602E+13	60	1.650000E+15	8.845605E+13	82	2.800000E+15	1.134625E+14
17	3.700000E+14	3.781169E+13	39	8.220117E+14	1.198602E+13	61	1.750000E+15	8.185507E+13	83	2.921954E+15	1.093823E+14
18	1.[illegible]	1.360311E+13	40	8.220133E+14	9.530200E+12	62	1.787967E+15	4.159281E+12	84	3.000000E+15	6.670436E+12
19	1.[illegible]	1.[illegible]	41	8.500000E+14	4.086072E+13	63	1.787971E+15	1.115443E+13	85	3.081115E+15	2.330843E+14
20	4.[illegible]	[illegible]	42	8.993770E+14	5.317090E+13	64	1.820000E+15	4.059911E+13	86	3.288047E+15	7.745553E+13
21	4.[illegible]	[illegible]	43	9.500000E+14	4.973740E+13	65	1.848849E+15	9.028552E+12			
22	4.[illegible]	[illegible]	44	1.000000E+15	4.[illegible]	66	1.848853E+15	1.206176E+13			

*** [illegible] TABLE [illegible] ***

TEFF 6450. LOG G 4.000 LTE MODEL ATMOSPHERE FOR PROCYON. ITERATION 11

	STANDARD DEPTH	TEMP	PRESSURE	ELECTRON NUMBER	DENSITY	ROSSELAND MEAN	BHYD						BMIN
1	1.000E-02	5206.2	6.929E+03	1.326E+12	2.131E-08	2.316E-02	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
2	1.585E-02	5229.0	9.067E+03	1.669E+12	2.776E-08	2.849E-02	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
3	2.512E-02	5263.0	1.180E+04	2.125E+12	3.590E-08	3.529E-02	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
4	3.981E-02	5312.5	1.527E+04	2.746E+12	4.603E-08	4.414E-02	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
5	6.310E-02	5383.1	1.964E+04	3.629E+12	5.841E-08	5.601E-02	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
6	1.000E-01	5482.6	2.504E+04	4.969E+12	7.311E-08	7.287E-02	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
7	1.585E-01	5620.0	3.151E+04	7.189E+12	8.976E-08	9.886E-02	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
8	2.512E-01	5807.2	3.888E+04	1.130E+13	1.072E-07	1.435E-01	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
9	3.981E-01	6058.2	4.666E+04	1.982E+13	1.233E-07	2.291E-01	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
10	6.310E-01	6393.6	5.411E+04	3.969E+13	1.354E-07	4.123E-01	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
11	1.000E+00	6837.2	6.048E+04	9.064E+13	1.415E-07	8.506E-01	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
12	1.585E+00	7429.1	6.534E+04	2.352E+14	1.403E-07	2.081E+00	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
13	2.512E+00	8240.2	6.851E+04	6.900E+14	1.316E-07	6.361E+00	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
14	3.981E+00	9279.8	7.020E+04	2.037E+15	1.166E-07	2.298E+01	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
15	6.310E+00	10451.0	7.096E+04	5.044E+15	9.746E-08	7.713E+01	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
16	1.000E+01	11607.0	7.133E+04	9.290E+15	7.774E-08	1.759E+02	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000

WAVELENGTH 666.668 FREQUENCY 4.496880E+14

	TAUSTD	TAUNU	ABTOT	ALPHA	BNU	SNU	JNU	HNU
1	1.000E-02	1.231E-02	3.045E-02	5.511E-03	2.157E-05	2.156E-05	1.950E-05	1.049E-05
2	1.585E-02	1.960E-02	3.774E-02	4.409E-03	2.198E-05	2.197E-05	1.991E-05	1.048E-05
3	2.512E-02	3.119E-02	4.703E-02	3.525E-03	2.258E-05	2.257E-05	2.052E-05	1.045E-05
4	3.981E-02	4.959E-02	5.903E-02	2.814E-03	2.347E-05	2.347E-05	2.142E-05	1.041E-05
5	6.310E-02	7.881E-02	7.499E-02	2.237E-03	2.478E-05	2.478E-05	2.273E-05	1.035E-05
6	1.000E-01	1.252E-01	9.739E-02	1.762E-03	2.669E-05	2.668E-05	2.464E-05	1.026E-05
7	1.585E-01	1.987E-01	1.315E-01	1.366E-03	2.944E-05	2.944E-05	2.743E-05	1.011E-05
8	2.512E-01	3.152E-01	1.892E-01	1.038E-03	3.342E-05	3.342E-05	3.148E-05	9.880E-06
9	3.981E-01	5.000E-01	2.977E-01	7.850E-04	3.914E-05	3.914E-05	3.737E-05	9.538E-06
10	6.310E-01	7.940E-01	5.237E-01	6.139E-04	4.748E-05	4.747E-05	4.593E-05	9.050E-06
11	1.000E+00	1.265E+00	1.041E+00	5.312E-04	5.962E-05	5.961E-05	5.839E-05	8.402E-06
12	1.585E+00	2.032E+00	2.403E+00	5.163E-04	7.766E-05	7.767E-05	7.676E-05	7.608E-06
13	2.512E+00	3.345E+00	7.090E+00	5.077E-04	1.054E-04	1.054E-04	1.045E-04	6.472E-06
14	3.981E+00	5.774E+00	2.780E+01	4.226E-04	1.452E-04	1.450E-04	1.444E-04	4.640E-06
15	6.310E+00	1.044E+01	1.162E+02	2.977E-04	1.947E-04	1.944E-04	1.941E-04	2.758E-06
16	1.000E+01	1.895E+01	3.568E+02	2.232E-04	2.474E-04	2.472E-04	2.471E-04	1.586E-06

THE FOLLOWING DATA HAS BEEN READ BY DUMMYR

FREQUENCY IDENTIFIERS 25
19 23 29 33 35 37 39 41 43 45 47 49 51 53 55 57 59 61 63 65 67 70 72 74
76

8 OPTICAL DEPTHS AT FREQUENCY IDENTIFIED BY 0
1 1.000E-03 2 1.000E-02 3 1.000E-01 4 5.000E-01 5 1.000E+00 6 5.000E+00 7 1.000E+01 8 1.000E+02

TEFF 5770. LOG G 4.44 LTE

TITLE SOLAR TYPE MODEL ATMOSPHERE WITH KURUCZIS HYDROGEN LINE FUDGE

XSCALE 1.000 H .900 HE .100
LI-11.60 BE -9.60 B -9.20 C -3.50 N -4.12 O -3.28 F -6.60 NE -3.50 NA -6.18 MG -4.57
AL -5.65 SI -4.50 P -6.62 S -4.84 CL -6.60 AR -5.30 K -7.00 CA -5.72 SC -8.01 TI -7.55
V -8.13 CR -6.58 MN -7.17 FE -4.50 CO -8.40 NI -6.97 CU -7.30 ZN -7.63 GA -9.11 GE -8.73
AS -9.70 SE -8.80 BR -9.40 KR -8.80 RB -9.42 SR -9.23 Y -9.60 ZR -9.60 NB-10.30 MO-10.00
TC-20.00 RU-10.40 RH-11.20 PD-10.70 AG-11.30 CD -9.98 IN-10.34 SN-10.34 SB-10.40 TE-10.00
I -10.60 XE-10.00 CS-10.90 BA-10.15 LA-10.60 CE-10.40 PR-11.20 ND-10.50 PM-20.00 SM-11.00
EU-11.30 GD-10.90 TB-11.60 DY-10.80 HO-11.50 ER-11.10 TM-11.90 YB-10.90 LU-11.70 HF-11.40
TA-11.70 W -10.90 RE-11.40 OS-10.70 IR-10.80 PT-10.40 AU-11.30 HG-11.10 TL-11.50 PB-10.15
BI-11.30 PO-20.00 AT-20.00 RN-20.00 FR-20.00 RA-20.00 AC-20.00 TH-11.70 PA-20.00 U -12.00
NP-20.00 PU-20.00 AM-20.00 CM-20.00 BK-20.00 CF-20.00 ES-20.00

H1 1 H2PLUS 1 HMINUS 1 HRAY 1 HE1 1 HE2 1 HEMINUS 0 HERAY 0 COOL 1 LUKE 0
HOT 0 ELECTRON 1 H2RAY 0 HLINES 1 LINES 0 LINESCAT 0 XLINES 0 XLSCAT 0 XCONT 0 XSCAT 0

IFCORR 1 IFPRES 1 IFSURF 0 IFSCAT 1 IFCONV 1 MIXLTH 1.00 IFMOL 0
IFTURB 0 TRBFDG 0.00 TRBPOW 0.00 TRBSND 0.00 TRBCON 0.00
ITERATIONS 6 TO 6
IFPRNT 1 1 1 1 1 14 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
IFPNCH 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

FREQID SOLAR NUMNU 86 NUL0 1 NUHI 76

1	3.288047E+13	1.644025E+13	23	4.500000E+14	9.191423E+12	45	1.050000E+15	4.975351E+13	67	1.900000E+15	8.480297E+13
2	3.288053E+13	1.542103E+12	24	4.535439E+14	2.345265E+12	46	1.100000E+15	4.989606E+13	68	1.972315E+15	2.660077E+13
3	3.670000E+13	3.998834E+12	25	4.567919E+14	2.112665E+12	47	1.150000E+15	5.223402E+13	69	1.972318E+15	1.424415E+13
4	4.059321E+13	6.614761E+12	26	4.605108E+14	1.150621E+13	48	1.192579E+15	1.569611E+13	70	2.020000E+15	6.964705E+13
5	5.137578E+13	1.346479E+13	27	4.800000E+14	1.889196E+13	49	1.192581E+15	2.172916E+13	71	2.076098E+15	1.989230E+13
6	6.710306E+13	1.356110E+13	28	5.000000E+14	3.198905E+13	50	1.250000E+15	6.475939E+13	72	2.076102E+15	4.743675E+13
7	9.133472E+13	4.428176E+13	29	5.500000E+14	6.143883E+13	51	1.300000E+15	4.548211E+13	73	2.200000E+15	1.277701E+14
8	1.315219E+14	1.517814E+13	30	6.000000E+14	2.798808E+13	52	1.350000E+15	5.000737E+13	74	2.300000E+15	1.224355E+14
9	1.315221E+14	7.266987E+12	31	6.105754E+14	4.672930E+12	53	1.400000E+15	5.257096E+13	75	2.419598E+15	4.585766E+13
10	1.506495E+14	2.048866E+13	32	6.167301E+14	5.446832E+12	54	1.442999E+15	1.587131E+13	76	2.419602E+15	1.814854E+13
11	1.742980E+14	3.445283E+13	33	6.232692E+14	1.323912E+13	55	1.443001E+15	3.614600E+13	77	2.465399E+15	4.501286E+13
12	2.055029E+14	1.177264E+13	34	6.500000E+14	3.981311E+13	56	1.515290E+15	3.614600E+13	78	2.500000E+15	4.846377E+13
13	2.055033E+14	9.324904E+12	35	7.000000E+14	5.200452E+13	57	1.515294E+15	1.235998E+13	79	2.600000E+15	1.481648E+14
14	2.300000E+14	2.856851E+13	36	7.500000E+14	5.582004E+13	58	1.550000E+15	4.714918E+13	80	2.725397E+15	4.601001E+13
15	2.600000E+14	3.681093E+13	37	7.980397E+14	1.795130E+13	59	1.600000E+15	3.869734E+13	81	2.725403E+15	2.259495E+13
16	3.000000E+14	3.371661E+13	38	7.980413E+14	1.198602E+13	60	1.650000E+15	8.845605E+13	82	2.800000E+15	1.134625E+14
17	3.300000E+14	3.781169E+13	39	8.220117E+14	1.198602E+13	61	1.750000E+15	8.185507E+13	83	2.921954E+15	1.093823E+14
18	3.653305E+14	1.360311E+13	40	8.220133E+14	9.419266E+12	62	1.787967E+15	4.159281E+12	84	3.000000E+15	6.670436E+12
19	3.653309E+14	1.310898E+13	41	8.500000E+14	4.114497E+13	63	1.787971E+15	1.115443E+13	85	3.081115E+15	2.330843E+14
20	4.000000E+14	3.757340E+13	42	9.000000E+14	5.324043E+13	64	1.820000E+15	4.069911E+13	86	3.288047E+15	7.745553E+13
21	4.170000E+14	8.784831E+12	43	9.500000E+14	4.918284E+13	65	1.848849E+15	9.028562E+12			
22	4.340000E+14	1.882303E+13	44	1.000000E+15	5.000000E+13	66	1.848853E+15	1.206176E+13			

*** DEFAULT OPACITY TABLE WILL BE USED ***

TEFF 5770. LOG G 4.440 SOLAR TYPE MODEL ATMOSPHERE WITH KURUCZ'S HYDROGEN LINE FUDGE ITERATION 6

	ROSS DEPTH	TEMP	PRESSURE	ELECTRON NUMBER	DENSITY	ROSSELAND MEAN				BHYD			BMIN
1	1.000E-03	4602.9	4.085E+03	5.330E+11	1.421E-08	1.234E-02	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
2	1.000E-02	4646.9	1.442E+04	1.587E+12	4.970E-08	3.471E-02	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
3	1.000E-01	4918.1	5.095E+04	5.326E+12	1.659E-07	9.967E-02	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
4	5.000E-01	5594.1	1.196E+05	1.778E+13	3.423E-07	2.348E-01	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
5	1.000E+00	6138.2	1.606E+05	4.700E+13	4.188E-07	4.969E-01	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
6	5.000E+00	8328.3	2.131E+05	1.363E+15	4.067E-07	1.007E+01	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
7	1.000E+01	9757.2	2.191E+05	5.568E+15	3.473E-07	4.995E+01	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
8	1.000E+02	17044.0	2.263E+05	4.347E+16	1.168E-07	3.933E+02	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000

OPACITY H1

DEPTH 1.000E-03 1.000E-02 1.000E-01 5.000E-01 1.000E+00 5.000E+00 1.000E+01 1.000E+02

	FREQUENCY	WAVE	OPACITY VALUES							
19	3.653393E+14	820.587	-5.28791	-5.16147	-4.43409	-2.92758	-1.95337	.68234	1.75462	3.37399
23	4.500000E+14	666.206	-5.54287	-5.41628	-4.68786	-3.17858	-2.20207	.44260	1.51996	3.15619
29	5.500000E+14	545.077	-5.79180	-5.66512	-4.93611	-3.42510	-2.44701	.20461	1.28644	2.93943
33	6.232692E+14	481.000	-5.94827	-5.82155	-5.09234	-3.58070	-2.60195	.05307	1.13724	2.80054
35	7.000000E+14	428.275	-6.09423	-5.96750	-5.23820	-3.72620	-2.74705	-.08959	.99650	2.66888
37	7.980397E+14	375.661	-6.25969	-6.13296	-5.40361	-3.89142	-2.91201	-.25262	.83526	2.51711
39	8.220117E+14	364.706	-6.29713	-6.17040	-5.44105	-3.92884	-2.94939	-.28968	.79854	2.48238
41	8.500000E+14	352.697	-3.84978	-3.74368	-3.13344	-1.87032	-1.05411	1.15126	2.04514	3.21347
43	9.500000E+14	315.571	-3.98582	-3.87972	-3.26946	-2.00626	-1.18993	1.01644	1.91139	3.08676
45	1.050000E+15	285.517	-4.10874	-4.00265	-3.39238	-2.12915	-1.31277	.89416	1.78980	2.97052
47	1.150000E+15	260.689	-4.22091	-4.11482	-3.50455	-2.24131	-1.42491	.78232	1.67839	2.86339
49	1.192581E+15	251.381	-4.26586	-4.1597[illegible]	-3.54949	-2.28626	-1.46985	.73745	1.63366	2.82013
51	1.300000E+15	230.610	-4.37273	-4.26663	-3.65637	-2.39314	-1.57673	.63068	1.52712	2.71661
53	1.400000E+15	214.138	-4.46485	-4.35876	-3.74849	-2.48527	-1.66887	.53858	1.43514	2.62673
55	1.443001E+15	207.756	-4.50253	-4.39644	-3.78618	-2.52295	-1.70655	.50090	1.39749	2.58980
57	1.515294E+15	197.845	-4.56351	-4.45741	-3.84715	-2.58394	-1.76754	.43991	1.33653	2.52987
59	1.600000E+15	187.370	-4.63149	-4.52539	-3.91513	-2.65192	-1.83553	.37190	1.26855	2.46283
61	1.750000E+15	171.310	-4.74376	-4.63766	-4.02741	-2.76420	-1.94782	.25957	1.15623	2.35166
63	1.787971E+15	167.672	-4.77070	-4.66461	-4.05435	-2.79115	-1.97477	.23262	1.12926	2.32491
65	1.848849E+15	162.151	-4.81277	-4.70667	-4.09642	-2.83322	-2.01684	.19053	1.08717	2.28309
67	1.900000E+15	157.786	-4.84709	-4.74099	-4.13074	-2.86754	-2.05117	.15619	1.05282	2.24894
70	2.020000E+15	148.412	-4.92421	-4.81811	-4.20786	-2.94467	-2.12830	.07902	.97563	2.17207
72	2.076102E+15	144.402	-4.95875	-4.85266	-4.24241	-2.97922	-2.16285	.04445	.94106	2.13760
74	2.300000E+15	130.345	-5.08814	-4.98205	-4.37180	-3.10862	-2.29226	-.08501	.81155	2.00829
76	2.419602E+15	123.902	-5.15232	-5.04623	-4.43598	-3.17280	-2.35645	-.14922	.74732	1.94406

9: CHANGES TO SUBROUTINE CONVEC

In order to calculate thermodynamic derivatives it is necessary to perturb slightly the pressure and temperature, and to obtain new number densities. Kurucz introduces an approximation by using only a small set of equilibrium equations (including only H, H_2, and He) to recalculate the number densities. This approximation is very good for those cases that have been tried. However, if there are situations where the number density of one or more of the species considered is very small, e.g. a pure hydrogen atmosphere, or a very hot normal abundance model, then the method that Kurucz uses to calculate the number densities breaks down.

Therefore it was decided to re-write that section of CONVEC which calculates the number densities. The method used in the new version is identical to the technique used by subroutine NELECT. All the species considered by NELECT are included in the calculation. Because of this change CONVEC needs to call subroutine PFSAHA, so the overlay structure of the program had to be changed to allow for this. The increase in the execution time of the program caused by this change is not very significant.

It was decided to give the user an extra option in the convection calculation, that of not calculating any convective variables. This entails changing the control card for convection. It now has the form:

CONVECTION n1 n2

where n1 is the ratio of mixing length to pressure scale height, and n2 is the option number (Ø, 1, or 2). For IFCONV=Ø, no convective variables are calculated. For IFCONV=1, the convective variables are calculated unless IFPRESS=Ø or MIXLTH=Ø. For IFCONV=2, convective variables are calculated and convection is included in the temperature correction. The default is IFCONV=1, with MIXLTH=1.Ø, i.e. the same as for ATLAS5.

10: THE HANDLING OF OPACITY TABLES

In ATLAS5 the following steps were required to change the opacity table:

a) A utility program (OPTAB) was run to calculate an opacity table.

b) This table was inserted into subroutine TTAUP.

c) TTAUP was recompiled and the new program was run.

It is bad in principle that one should have to recompile part of a general program in order to provide different data for the program to use. Therefore it was decided to change OPTAB to produce either:

a) A table suitable for insertion into subroutine TTAUP, as in ATLAS5 (which is useful if one is calculating a large number of models which require a particular opacity table).
or:
b) A deck of cards containing the opacity values which can be read by READIN.

The changes to OPTAB are described in chapter 14, and the other changes required are as follows:

a) The opacity deck produced by OPTAB is prefaced by the control card:

```
KAPPA  nl
```

where nl is the number of temperature and pressure points. READIN has to be able to recognise this control card and read in its associated data. The data is transferred to the root segment via the COMMON block:

```
COMMON /KAPPA/ TABT(3Ø),TABP(3Ø),KTAB(3Ø,3Ø),NTAB,IK
```

b) The default opacity table must be moved to a BLOCK DATA subroutine.

11: ERROR MESSAGES

ATLAS5 provides very little diagnostic information when an error occurs. It was decided to provide more diagnostics, to help the user to locate the error. This led to the following changes:

a) Subroutine TTAUP

The pressure at each depth is calculated iteratively with an infinite loop in ATLAS5. If the iterations fail to converge to the required accuracy, the program will continue performing iterations until the job exceeds its time allocation. This was changed to restrict the maximum number of iterations to 100. If this number is exceeded then the following message is printed and the job is terminated:

PRESSURE FAILS TO CONVERGE FOR DEPTH nl

where nl is the depth identification number.

b) Subroutine TCORR

If the depth points are not well chosen then D=TAUNU(J+1)-TAUNU(J) can be negative. Hence an error occurs when the $\log_{10}(D)$ is calculated. A test has been inserted to determine the sign of D. If it is negative then the following error message is printed and the run is terminated:

THE MODEL HAS BLOWN UP AT DEPTH nl

c) Subroutine NELECT

The error message was changed to:

XNE DOES NOT CONVERGE FOR DEPTH nl

d) Subroutine MOLEC

The error message was changed to:

NO DATA FOR MOLECULE nl

where nl is the molecular code.

e) Subroutine CONVEC

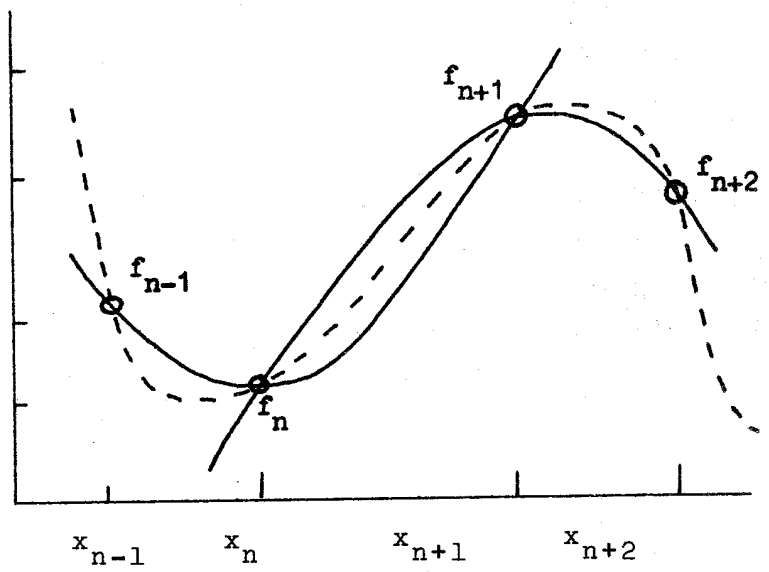
This subroutine now uses the same logic as NELECT for calculating number densities. The error message is:

CONVEC: XNE DOES NOT CONVERGE AT DEPTH nl

12: OPTIMISATION OF THE CODE

Because the program is used a great deal, it was decided that it would be worth while optimising the program for faster execution time. This can be done by improving the numerical analysis, and by improving the FORTRAN coding. The changes in the numerical analysis which were suggested by S.D. Pepper (private communication) are given in detail below.

a) Double Parabolic Fitting



A reasonable representation of a function between x_n and x_{n+1} can be obtained by fitting a 'backward' parabola through x_{n-1}, x_n, x_{n+1}, and a 'forward' parabola through x_n, x_{n+1}, x_{n+2}. These parabolas and then weighted inversely by their second derivatives. The analysis given by Kurucz, 1970 p.16 retains a fixed coordinate system for all the parabolas. However there is great advantage to be obtained in setting the origin of the coordinate system to x_n when fitting a function over the range x_n to x_{n+1}.

We define a forward parabola on x_n, x_{n+1}, x_{n+2} with origin at x_n,

$f = a_n^f + b_n^f(x-x_n) + c_n^f(x-x_n)^2$, where:

$$c_n^f = \left[\frac{f_{n+2} - f_{n+1}}{x_{n+2} - x_{n+1}} - \frac{f_{n+1} - f_n}{x_{n+1} - x_n} \right] \Big/ (x_{n+2} - x_n)$$

$$b_n^f = \frac{f_{n+1} - f_n}{x_{n+1} - x_n} - c_n^f.(x_{n+1} - x_n)$$

$$a_n^f = f_n$$

The backward parabola on x_{n-1}, x_n, x_{n+1} can be obtained from the previous forward parabola defined on those points but with origin x_{n-1}, by the transformations:

$$c_n^b = c_{n-1}^f$$

$$b_n^b = c_{n-1}^f.(x_n - x_{n-1}) + \frac{f_n - f_{n-1}}{x_n - x_{n-1}}$$

$$a_n^b = a_n^f$$

The final result is:

$$f = f_n + b_n.(x - x_n) + c_n.(x - x_n)^2$$

with $b_n = b_n^f + w_n.(b_n^b - b_n^f)$ and

$c_n = c_n^f + w_n.(c_n^b - c_n^f)$

$$\text{where} \quad w_n = \frac{\left|c_n^f\right|}{\left|c_n^f\right| + \left|c_n^b\right|}$$

This method is much faster than the original one, because the terms in the expressions are simpler, and data used in calculating one parabola can be used to get the next one. This method is also more accurate because of fewer operations giving smaller truncation and round-off errors, and also because one is not defining the quantities with respect to an origin which could be a long way away.

1 The method described above has been applied to MAP1, INTEG, and PARCOE. Where there is not sufficient information to fit double parabolas we have used linear interpolation or extrapolation.

2 Using this technique to calculate the intensity we obtain the following analysis:

$$I_\nu(\mu) = \int_0^\infty S_\nu \; e^{-\tau_\nu/\mu} \; \frac{d\tau_\nu}{\mu}$$

Splitting this integral up into three ranges, and using the weighted parabolas approximation to S_ν in the middle range,

$$I_\nu(\mu) = \frac{1}{\mu}\int_0^{t_1} S_\nu \; e^{-t_\nu/\mu} \; dt_\nu$$
$$+ \frac{1}{\mu}\sum_{i=1}^{n-i}\int_{t_i}^{t_{i+1}} \left[S_i + b_i(t_\nu - t_i) + c_i(t_\nu - t_i)^2\right] e^{-t_\nu/\mu} \; dt_\nu$$
$$+ \frac{1}{\mu}\int_{t_n}^\infty S_\nu \; e^{-t_\nu/\mu} \; dt_\nu$$

Doubly integrating by parts the second term we obtain:

$$\text{2nd term} = -\sum_{i=1}^{n-1}\left\{\left[e^{-t_\nu/\mu}(S_i + b_i(t_\nu-t_i) + c_i(t_\nu-t_i)^2)\right]_{t_i}^{t_{i+1}}\right.$$

$$\left. + \left[\mu e^{-t_\nu/\mu}(b_i + 2c_i(t_\nu-t_i))\right]_{t_i}^{t_{i+1}} + \left[\mu^2 e^{-t_\nu/\mu}.2c_i\right]_{t_i}^{t_{i+1}}\right\}$$

which can be rewritten as:

$$e^{-t_1/\mu}S_1 + \sum_{i=2}^{n-1} e^{-t_i/\mu}(S_i-(S_{i-1}+b_{i-1}(t_i-t_{i-1})+c_{i-1}(t_i-t_{i-1})^2))$$

$$-e^{-t_n/\mu}(S_{n-1}+b_{n-1}(t_n-t_{n-1})+c_{n-1}(t_n-t_{n-1})^2) + \mu e^{-t_1/\mu}b_1$$

$$+ \sum_{i=2}^{n-1} \mu e^{-t_i/\mu}(b_i-(b_{i-1}+2c_{i-1}(t_i-t_{i-1})))$$

$$-\mu e^{-t_n/\mu}(b_{n-1}+2c_{n-1}(t_n-t_{n-1})) + \mu^2 e^{-t_1/\mu}2c_1$$

$$+ \sum_{i=2}^{n-1} \mu^2 e^{-t_i/\mu}2(c_i-c_{i-1}) - \mu^2 e^{-t_n/\mu}2c_{n-1}$$

If in the first term we approximate S_ν by a straight line extrapolated from the first two points we obtain:

$$\frac{1}{\mu}\int_0^{t_1}\left[S_1+b_1(t_\nu+t_1)\right]e^{-t_\nu/\mu}\,dt_\nu$$

$$= S_1 - e^{-t_1/\mu}S_1 - b_1t_1 - e^{-t_1/\mu}b_1 + \mu b_1$$

Similarly if we approximate S_ν by a straight line in the third term:

$$\frac{1}{\mu}\int_{t_n}^{\infty}\left[S_n+b_n(t_\nu-t_n)\right]e^{-t_\nu/\mu}dt_\nu = e^{-t_n/\mu}\,S_n + e^{-t_n/\mu}\,b_n$$

Finally combining these terms and noting that $c_n=0$ (for a straight line), and that by definition:

$$S_i = S_{i-1} + b_{i-1}(t_i-t_{i-1}) + c_{i-1}(t_i-t_{i-1})^2$$

we obtain:

$$I_\nu(\mu) = \sum_{i=2}^{n} \mu e^{-t_i/\mu}((b_i-(b_{i-1}+2c_{i-1}(t_i-t_{i-1})))+2\mu(c_i-c_{i-1}))$$

$$+ S_1 - b_1 t_1 + \mu b_1$$

This method does not suffer from any cancellation effects, and is very quick.

b) Optimisation

Some FORTRAN compilers optimise the code more effectively than others. This can lead to quite large differences in execution time. We have tried to eliminate this effect by optimising the FORTRAN code at source level. Thus we have eliminated common sub-expressions from statements, etc. The IBM 360/65 takes longer to perform a division than a multiplication. Therefore where feasible we have converted division to multiplications.

In order to fit the program into a convenient partition size on the IBM 360, and so to improve the turn-around time, it was necessary to make some of the temporary variables used in the opacity calculation single precision. This causes the program to execute more slowly because of the single to double precision conversions that are now required. However this effect is not very large. The accuracy of the calculation is not affected significantly by this change.

13: SUNDRY ALTERATIONS

It was found convenient to make the following further alterations:

a) The calculation of abundance in the BEGIN section of READIN was changed to allow the user to input an abundance of zero for any element.

b) For a possible future change of the temperature correction method, we have made the following changes:

1) The IFSURF=1 option (which calculates the surface flux only) has been removed. The calculation of surface intensity only is now IFSURF=1 (previously it was IFSURF=2).

2) The CORRECTION control card now has the format:

CORRECTION n1

n1=Ø indicates that no temperature correction is to be performed. n1=1 indicates that the standard temperature correction method is to be used.

c) An extra option has been added to READ STARTING. The format is now:

READ STARTING n1 n2 n3,n4 n5,n6 ..,..

n2 is the number of depth points. n3,n4 (etc) are the pairs of depth variable and temperature. n1 indicates the depth variable; n1=1 means TAUROS, and n1=2 means RHOX.

d) The default PRINT option is now IFPRNT=1 for all iterations, except the last when it is IFPRNT=2. The default PUNCH option is now IFPNCH=Ø for all iterations.

e) If an opacity table is not read in, the message:

DEFAULT OPACITY TABLE WILL BE USED

is printed under the initial summary table.

f) Arguments to subroutines READIN, KAPP, and JOSH were removed and passed over by COMMON blocks:

```
COMMON /DISFNC/ STEPWT,NSTEPS,N
COMMON /MODE/ MODE
```

g) When TAUROS is calculated, TAUROS(1) was always set to zero. This can cause problems when using SCALE or CHANGE to shift the depth grid higher in the atmosphere. Therefore this fudge was taken out. To ensure that SCALE and CHANGE give accurate mappings, we convert variables to $\log_{10}$ before mapping them, and then convert back again.

h) The user of ATLAS5 will probably have noticed that most atmospheres produced by the program have a kink in the temperature structure near the top of the atmosphere. This is apparently caused by the fact that Kurucz sets the optical depth at the top of the atmosphere to zero before calculating the radiation field.

We have re-coded that section which introduced this kink to eliminate the effect.

i) The change made under h) causes a problem to appear in SUBROUTINE PUTOUT during the calculation of RHOX1 (the value of $\log_{10}$(RHOX) at τ_ν=1), when certain strong lines are included in the calculation. Instead of interpolating in RHOX and then taking the $\log_{10}$, the interpolation is now carried out directly in the $\log_{10}$.

14: CHANGES TO THE UTILITY PROGRAMS

FRESET

No major changes have been made. Some intermediate tables have been deleted and only the final table is now printed. The format of the punched card output has been changed so as to punch two sets of frequencies and weights per card. A test has been inserted so that no cards are punched if any of the weights is negative.

PRETAB

The program has been changed so that the information controlling the calculation, and the optical depths required for the calculation, are read in. Thus the program does not need to be recompiled every time it is run. The first data card must contain NTAU, IFJ, IFH, IFPUN in 4I4 format, where

NTAU is the number of depth points;

IFJ is set to 1 if the Λ matrix is to be calculated, otherwise Ø;

IFH is set to 1 if the Φ matrix is to be calculated, otherwise Ø;

IFPUN is set to Ø if no punched card output is required,
to 1 if the matrices are to be punched in the form of DATA statements,
to 2 if the matrices are to be written to logical unit 7 in the format 1P 8ØD16.9 (1P 8ØE16.9 for CDC).

The next set of data cards must contain the NTAU values of the depth points in the format 8D1Ø.3 (8E1Ø.3 for CDC).

The only other changes to PRETAB are to the format statements controlling the IFPUN=1 option. The IFPUN=2 option, which writes the depth points and both matrices to a file (usually a disk file) is useful if it is decided that it would be easier to read this data in, rather than compile the large

block data subroutines.

OPTAB

A new input subroutine, READA, has been written which reads in all the data required by OPTAB. READA will recognise the code words: OPACITY, MOLECULES, CALCULATE, ABUNDANCE, READ FREQUENCIES, BEGIN, END, TITLE, WAVELENGTH.

All the above code words except CALCULATE have the same meaning and format as before. The CALCULATE option for OPTAB has the format:

CALCULATE cw1 n1 n2 n3
n4 n5 ...

n1 is the number of temperature and pressure points. n2 is the starting number and n3 the finishing number of the part of the table to be calculated in this run. This card must be followed by cards containing n1 values of $\log_{10}$(temperature) followed by n1 values of $\log_{10}$(pressure). cw1 is a code word which indicates the format of the output. If the code word is DECK then the table will be output in a format suitable to be read in by READIN. If the code word is TABLE, the output will be in a form suitable for insertion into a block data subroutine.

The overlay structure for OPTAB is shown in figure 2.

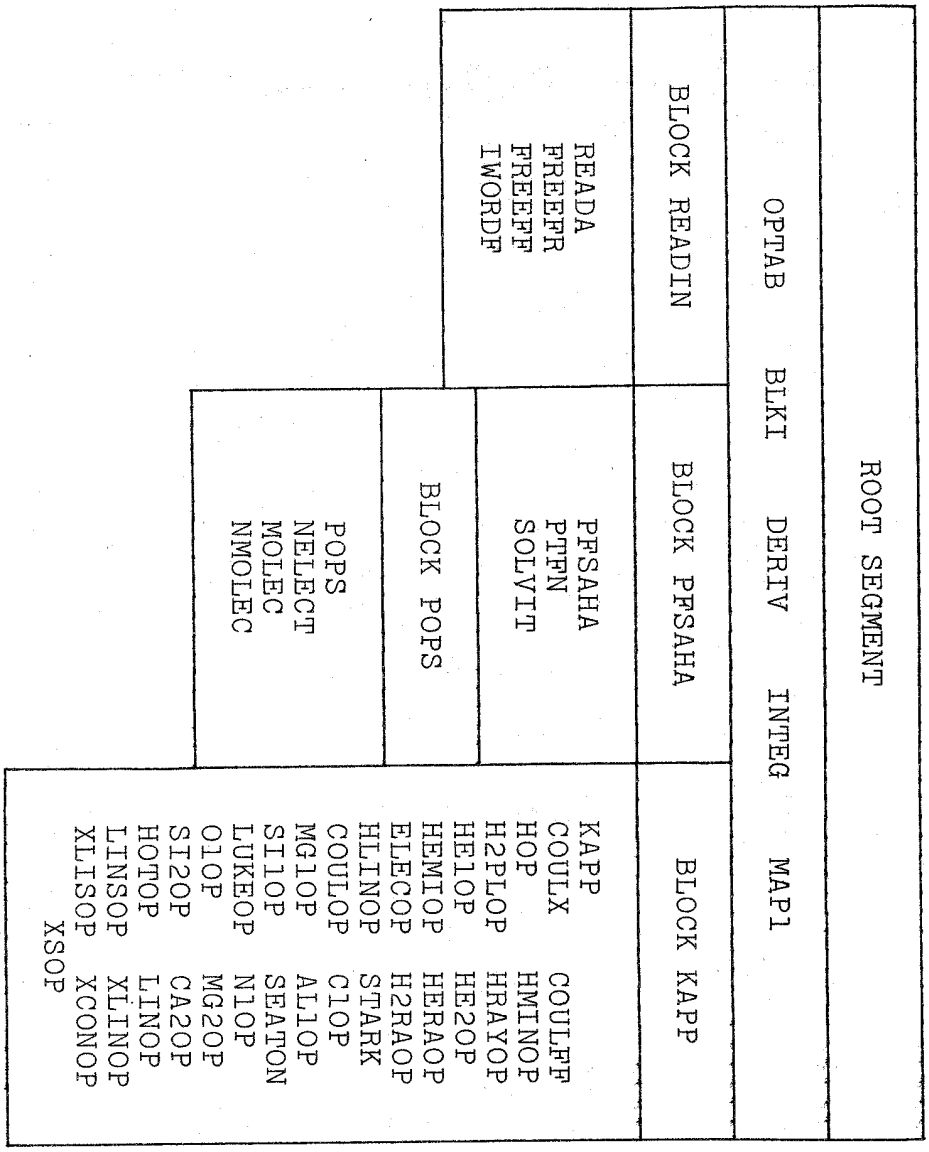


Figure 2: Overlay Structure for OPTAB

15: IMPLEMENTATION OF SAM1 ON THE IBM 360/65

Introduction

The master version of the program mentioned in Chapter 3 consists mainly of a double precision version of the program; in addition instructions specific to the CDC machine are prefaced by CCDC starting in column 1, while IBM specific instructions are prefaced by CIBM. A FORTRAN program called STRIP has been written which accepts this master version as input and outputs an executable version of the program for the specified machine.

The whole system is on a MASTER tape consisting of a set of card images of the program STRIP followed by the Master version of the program. It is not possible to execute this Master version. To obtain an executable version of SAM1 we compile and execute a small FORTRAN program which accepts the MASTER tape on logical unit 8 and writes STRIP (12 subroutines) to logical unit 9 with the Master version going to logical unit 1Ø. The next step is to compile and run program STRIP. It accepts the Master version of the program on logical unit 1Ø and writes the executable version to logical unit 11.

Processing the Master version

The JCL required to perform the operations described above follows:

```
i)    //UCAQØ35M JOB (12Ø,SAM1,t,1),'descriptive title'
ii)   /*SETUP  TAPE9,U327  SL,READONLY,VOL=SER=STAMØ2
iii)  /*SETUP  TAPE9,U328  SL,WRITE,VOL=SER=STAMØ3
iv)   //ONE EXEC FORTRAN
v)    //C.SYSIN DD *

            FORTRAN program in Appendix 1

vi)   /*
```

```
vii)   //G.FTØ8FØØ1 DD UNIT=TAPE9,DSN=MASTER,
       //  VOL=SER=STAMØ2,LABEL=(1,SL,,IN),DISP=OLD,
       //  DCB=(RECFM=FB,LRECL=8Ø,BLKSIZE=2ØØØ)
viii)  //G.FTØ9FØØ1 DD UNIT=DRUM,SPACE=(CYL,(1Ø,2)),DISP=(NEW,PASS),
       //  DCB=(RECFM=FB,LRECL=8Ø,BLKSIZE=2ØØØ)
ix)    //G.FT1ØFØØ1  DD UNIT=DRUM,SPACE=(CYL,(1Ø,2)),DISP=(NEW,PASS),
       //  DCB=(RECFM=FB,LRECL=8Ø,BLKSIZE=2ØØØ)
x)     //TWO EXEC FORTRAN
xi)    //C.SYSIN DD DSN=*.ONE.G.FTØ9FØØ1,DISP=(OLD,PASS)
xii)   //G.FT1ØFØØ1 DD DSN=*.ONE.G.FT1ØFØØ1,DISP=(OLD,PASS)
xiii)  //G.FT11FØØ1 DD UNIT=TAPE9,DSN=SAM1S,VOL=SER=STAMØ3,
       //  LABEL=(1,SL,,IN),DISP=(NEW,KEEP),
       //  DCB=(RECFM=FB,LRECL=8Ø,BLKSIZE=2ØØØ)
xiv)   //G.FT12FØØ1 DD UNIT=DRUM,SPACE=(CYL,(1Ø,2)),DISP=NEW,
       //  DCB=(RECFM=VS,LRECL=84,BLKSIZE=1684)
xv)    //G.SYSIN DD *
xvi)   IBM
xvii)  /*
xviii) //
```

Card i) is a suitable JOB card where "t" is the CPU time required to run the job and "l" is the number of lines (in thousands) of line printer output required. Suitable values for this particular job are t=5 minutes, l=2.

Cards ii) and iii) request the operators to mount the required magnetic tapes.

Card iv) invokes a FORTRAN compile, link, and go sequence.

Card v) defines the input stream of the compiler to be on cards immediately following this card.

Card vi) marks the end of the compiler input. The program reads the MASTER tape from unit 8 as defined by card vii) and writes STRIP to unit 9 (a scratch disk file defined by cards viii) with the Master version of SAM1 being written to logical unit 1Ø (a scratch disk file defined by card ix).

Card x) re-invokes the compile, link, and go sequence for FORTRAN. This time the input is taken from logical unit 9 (card xi) which contains the program STRIP. When STRIP begins execution, it takes its main input from logical unit 1Ø (the

Master version of SAM1 as defined by card xii) and outputs the executable version of the program onto unit 11 (the tape SAM1S defined by card xiii).

Card xiv) defines a scratch disk file on logical unit 12 which is used by STRIP for performing binary I/O.

Card xv) defines the logical unit 5 input stream.

Card xvi) is input to STRIP and indicates to the program that an IBM version of the program is required.

Cards xvii) and xviii) ensure that the job is read in correctly.

As the IBM 360/65 is our on-campus computer, it is easier to handle the program using cards, so that the tape SAM1S is then dumped to cards. Consequently, the JCL which follows reflects this.

Creation of load module library

The sets of subroutines making up the blocks defining the overlay structure given on page 16 , are compiled separatel as members of a load module library using the following JCL:

```
i)    //UCAQØ351 JOB (12Ø,t,1),'S.L. WRIGHT'
ii)   //ONE EXEC FORTMOD,LIB=UCPGMLIB
iii)  //C.SYSIN DD *

            FORTRAN programs

iv)   /*
v)    //L.SYSIN DD *
vi)     NAME UCAQØ351
vii)  /*
viii) //
```

The values for the JOB card parameters vary with the module being compiled. However, upper limits for these parameters are t=5, and l=3.

The catalogued procedure FORTMOD (see card ii) compiles the FORTRAN program and writes the object module to a library UCPGMLIB with the name UCAQØ351 (defined by card vi). UCPGMLIB is a library for the storage of user's programs and is on a permanently mounted disk. Using JOB's of this type we compile and store the following modules:

ROOT segment	(subroutines SAM1 to RADOSS)	to UCAQØ351
BLOCK READIN	(subroutines READIN to IWORDF)	to UCAQØ352
BLOCK PFSAHA	(subroutines PFSAHA to SOLVIT)	to UCAQØ353
BLOCK POPS	(subroutines POPS to NMOLEC)	to UCAQØ354
BLOCK FINISH	(subroutines FINISH to CONVEC)	to UCAQØ355
BLOCK KAPP	(subroutines KAPP to XSOP)	to UCAQØ356
BLOCK JOSH	(subroutines JOSH to PARCOE)	to UCAQØ357

The modules are also stored under the same names in UCMODLIB which is another library on a permanently mounted disk reserved for the temporary storage (up to 30 days) of user's programs. In addition we compile FRESET (subroutines FRESET, FREEFR, FREEFF) to UCAQØ358 and the ROOT segment of OPTAB (subroutines OPTAB, DERIV, INTEG, MAP1, READA) to UCAQØ359 on UCPGMLIB.

Creation of executable modules

We can now link together these modules to obtain a completely executable program. The following JCL can be used:

SAM1:

```
i)      //UCAQØ35E JOB (17Ø,SAME,t,1),'S.L. WRIGHT'
ii)     //ONE EXEC LINKL,OVLY=OVLY,LIB=UCPGMLIB,CYLS=1Ø
iii)    //SYSLIB DD DSN=SYS1.FORTLIB,DISP=SHR
iv)     //SYSLMOD DD DSN=UCPGMLIB,DISP=SHR
v)      //SYSIN DD *
vi)       INCLUDE MODLIB(UCAQØ351)
vii)      OVERLAY ONE
viii)     INCLUDE MODLIB(UCAQØ352)
ix)       OVERLAY ONE
```

```
x)          INCLUDE MODLIB(UCAQØ353)
xi)         OVERLAY TWO
xii)        INCLUDE MODLIB(UCAQØ354)
xiii)       OVERLAY TWO
xiv)        INCLUDE MODLIB(UCAQØ355)
xv)         OVERLAY ONE
xvi)        INCLUDE MODLIB(UCAQØ356)
xvii)       INCLUDE MODLIB(UCAQØ357)
xviii)      NAME UCAQØ35A(R)
xix)      /*
xx)       //
```

Card ii) invokes the linkage editor.

Card iii) specifies the library which is used to resolve the externals; in this case, it is the FORTRAN library.

Card iv) specifies the library where the linked program is to be written.

Cards vi) to xvii) define the overlay structure of the program and card xviii) gives it the name of UCAQØ35A in the library UCPGMLIB. A similar procedure is followed for FRESET.

FRESET:

```
//UCAQØ35F JOB (17Ø,FRST,t,1),'S.L. WRIGHT'
//ONE EXEC LINKL,LIB=UCPGMLIB
//SYSLIB DD DSN=SYS1.FORTLIB,DISP=SHR
//SYSLMOD DD=UCPGMLIB,DISP=SHR
//SYSIN DD *
  INCLUDE MODLIB(UCAQØ358)
  NAME UCAQØ358(R)
/*
//
```

The procedure for OPTAB is almost identical to that used for SAM1. The principal difference is the deletion of READIN and DUMMYR which are replaced by READA.

OPTAB:

```
//UCAQØ35Ø JOB (17Ø,OPTB,t,1),'S.L. WRIGHT'
//ONE EXEC LINKL,OVLY='OVLY,LET',LIB=UCPGMLIB
//SYSLIB DD DSN=SYS1.FORTLIB,DISP=SHR
//SYSLMOD DD DSN=UCPGMLIB,DISP=SHR
```

```
//SYSIN DD *
  INCLUDE MODLIB(UCAQØ359)
  OVERLAY ONE
  INSERT READA
  REPLACE DUMMYR
  REPLACE READIN
  INCLUDE MODLIB(UCAQØ352)
  OVERLAY ONE
  INCLUDE MODLIB(UCAQØ353)
  INCLUDE MODLIB(UCAQØ354)
  OVERLAY ONE
  INCLUDE MODLIB(UCAQØ356)
  NAME UCAQØ359(R)
/*
//
```

Running the programs

A description of the procedure used to run the programs just described is given in the Communications of the University of London Observatory, No. 76, Appendix III.

Running the programs with modifications

If the changes made are large, compile the modules to UCMODLIB. Then the modules should be INCLUDEd and/or INSERTed in the appropriate places in the link editing sequence.

16: IMPLEMENTATION OF SAM1 ON THE CDC 6600

SAM1 as a file for UPDATE

UPDATE is a system utility developed by CDC and used to create, manipulate, and maintain library files. It is a versatile program allowing the user to change as little as one card image or to replace/delete/insert large sections of code. It is particularly suited to the manipulation of very large programs. For these reasons, SAM1 is an ideal candidate to be put into the system as a file for UPDATE.

The usual mode of operation is to create and maintain by the UPDATE program a file containing the program to be used. This file consists of card images in a compressed symbolic code with sufficient information to allow each card image to be referenced uniquely by UPDATE. Such a file can be created on disk or tape. On disk, the file is in random format. If the file is to be written to tape, it must be generated in sequential format firstly on disk, and then copied to the magnetic tape. It is written to tape as one binary record. UPDATE requires about 40000 (octal) words of core in which to run.

The first step in the implementation of SAM1 on the CDC 6600 is to obtain a copy of the source code by processing the MASTER tape. This is then stored as a file for UPDATE.

After processing the Master version of SAM1 using STRIP with the CDC option, the resultant program contains an UPDATE control card (*DECK subroutine name) in front of each subroutin This makes it very easy to use the output file from STRIP as the input file for UPDATE to produce a NEW Program Library (NEWPL). However, this does mean that if UPDATE is not used in a CDC implementation, then the series of *DECK cards must be removed somehow before compiling SAM1.

Processing the Master version

The following JCL produces an UPDATE file from the MASTER tape:

```
i)      JOB(UCAQØ35,Jj,LC1,Tt,M66ØØ)
ii)     ATTACH(OLDPL,SAM1MASTER,CY=n)
iii)    UPDATE(L=A123,C=TAPE8,*=#
iv)     REWIND(TAPE8)
v)      FTN(SL)
vi)     LGO.
vii)    REWIND(TAPE9)
viii)   REWIND(TAPE1Ø)
ix)     REWIND(LGO)
x)      FTN(I=TAPE9)
xi)     LGO.
xii)    REWIND(TAPE11)
xiii)   REQUEST(SAM1,*PF)
xiv)    UPDATE(N=SAM1,L=A123,*=*)
xv)     CATALOG(SAM1,SAM1UPDATE)
xvi)    - EOR -
xvii)   #COMPILE STRIP.END
xviii)  - EOR -
           FORTRAN program in Appendix 1
xix)    - EOR -
xx)     CDC
xxi)    - EOR -
xxii)   *READ TAPE11
xxiii)  - EOR -
xxiv)   - EOI -
```

Card i) is a suitable JCL card where "j" is the job category, "l" is the line count, and "t" is the CPU time in seconds. The M-parameter specifies the computer on which this job is to be run.

Card ii) indicates that in a previous run the Master version of the tape was put up as a permanent file for use by UPDATE. Here, cycle "n" of the Master file is given the local file name OLDPL and is made available to the job.

Card iii) invokes the program UPDATE which is used to extract the program STRIP from the Master file. The output from this run is written to logical unit 8 (TAPE8). The argument *=# indicates that UPDATE directives are prefixed with the "#" sign. Card xvii) is the directive for this UPDATE run.

Card v) causes the FORTRAN compiler FTN to run taking its input from cards, the program situated between cards xviii) and xix). The object code output from the compiler is written to the default file LGO.

Card vi) causes this object program to be loaded and executed. This particular program takes its principal input from logical unit 8 and writes program STRIP to logical unit 9 (TAPE9), and the Master version of SAM1 onto logical unit 1Ø (TAPE1Ø).

Cards vii) to ix) reposition the files to their beginning

Card x) once again invokes the FORTRAN compiler. This time it takes its input from TAPE9, the program STRIP. On execution the program takes its input from logical unit 1Ø (the Master version of SAM1), and writes the CDC compatible FORTRAN source code of SAM1 to logical unit 11. In this case the secondary input stream is taken from cards and is the card xx).

Card xiv) once again causes the program UPDATE to be run. The parameters indicate that a NEW Program Library is to be generated on the file SAM1 and that the "*" is used to indicate which cards are UPDATE directives for this run.

Card xxii) is the UPDATE directive for this run. This card causes the UPDATE program to take its input from logical unit 11 (TAPE11). We note that the version of the program written to TAPE11 has interspersed within it the UPDATE directives: *DECK subroutine name.

Cards xiii) and xv) make the NEW Program Library SAM1 a permanent file with name SAM1UPDATE.

Cards xvi), xviii), xix), xxi), and xxiii) are End-Of-Record cards used to separate the data sets for the various programs called during the Job Control Sequence (cards i to xv).

Card xxiv) is an End-Of-Information card which ensures that the job is read in correctly.

Suitable JOB card parameters are j=6, l=15ØØ, and t=6Ø.

Compile and generate an overlay file

The last stage in the implementation of SAM1 is to create a binary version of the program. The following JCL uses the segmented loader SEGLOAD to create an overlayed version of the program.

SAM1:

```
i)      JOB(UCAQØ34,Jj,Tt,LC1,M66ØØ)
ii)     ATTACH(OLDPL,SAM1UPDATE,ID=UCAQØ35)
iii)    UPDATE.
iv)     FTN(I=COMPILE,B=SAM)
v)      REQUEST(SAM1,*PF)
vi)     SEGLOAD(B=SAM1)
vii)    LOAD(SAM/R)
viii)   NOGO.
ix)     CATALOG(SAM1,SAM1)
x)      - EOR -
xi)     *COMPILE SAM1.XSOP
xii)    - EOR -
           data cards for the Segmented Loader as given
           in Appendix 2.
xiii)   - EOR -
xiv)    - EOI -
```

As before the first card is the JOB card in which are given the parameters and their values necessary to run this job.

Card ii) makes available the file SAM1UPDATE which the job will subsequently refer to as the file OLDPL. The ID parameter allows the user access to the permanent file from another jobnumber.

Card iii) causes the UPDATE program to be executed with all defaults in operation. This means that it expects an OLD Program Library on the file OLDPL and will write the card image form of the program to the file COMPILE.

Card xi) is the directive for this UPDATE run.

Card iv) runs the FORTRAN compiler FTN which takes its input from the file COMPILE containing the source code of the program SAM1. The binary output is written to the file called SAM.

Cards vi) to viii) are used to generate the overlayed file. SEGLOAD is the name of the segmented loaded which produces the overlayed file SAM1.

Cards vii) and viii) link the separate elements of the program together by loading them into core but suppressing execution. It is necessary to load the program to complete the overlay building process. The data for the segmented loader follows card xii) and is listed in Appendix 2. The following JOB card parameters apply: j=9, t=12Ø, l=3ØØØ.

To create an executable module of the program FRESET, the following JCL can be used:

FRESET:

```
JOB(UCAQØ34,Jj,Tt,M66ØØ)
ATTACH(OLDPL,SAM1UPDATE,ID=UCAQØ35)
UPDATE(K)
REQUEST(BIN,*PF)
FTN(I=COMPILE,SL,B=BIN)
CATALOG(BIN,FRESET)
- EOR -
*COMPILE FRESET,PARCO,FREEFR,FREEFF
- EOR -
- EOI -
```

Suitable values for the JOB card parameters are j=6, and t=3Ø.

The module for the program OPTAB is created in a similar way to that for the main program SAM1. The following JOB can be used:

OPTAB:

```
JOB(UCAQØ34,Jj,Tt,M66ØØ)
ATTACH(OLDPL,SAM1UPDATE,ID=UCAQØ35)
UPDATE(C=ONE,K)
FTN(I=ONE,B=OTAB)
REQUEST(OPTAB,*PF)
SEGLOAD(B=OPTAB)
LOAD(OTAB/R)
NOGO.
```

```
CATALOG(OPTAB,OPTAB)
- EOR -
*COMPILE OPTAB.BLOCKDO
*COMPILE DERIV,INTEG,MAP1
*COMPILE FREEFR.IWORDF
*COMPILE PFSAHA.SOLVIT
*COMPILE POPS.NMOLEC
*COMPILE KAPP.XSOP
- EOR -
   data cards for the Segmented Loader as given
   in Appendix 3.
- EOR -
- EOI -
```

Typical values for the JOB card parameters are j=9, and t=12Ø.

Running the program

The programs described in this implementation can be executed in the manner described in Appendix III, Comm. U.L.O., No. 76.

Running SAM1 with modifications

It may happen that a user will want to change or add to SAM1. To debug the altered program, it is convenient to be able to make the change, compile the program, generate the overlayed module and then execute it. This can be done using the following JCL:

```
i)      JOB(UCAQØ34,Jj,Tt,LC1,M66ØØ)
ii)     ATTACH(OLDPL,SAM1UPDATE,ID=UCAQØ35)
iii)    UPDATE.
iv)     FTN(I=COMPILE,B=SAM)
v)      SEGLOAD(=SAM1)
vi)     LOAD(SAM/R)
vii)    EXECUTE.
```

viii) - EOR -
*IDENT identifier

changes to SAM1 using UPDATE control cards and data cards.

*COMPILE SAM1.XSOP

ix) - EOR -
data for the Segmented Loader as given in Appendix 2.

x) - EOR -
data for SAM1 as described in Comm. U.L.O. No. 76.

xi) - EOR -

xii) - EOI -

The JCL is very similar to that used in the previous section on the generation of the overlay file containing SAM1. The principal differences are that the file containing the overlayed program is not made permanent at the end of the job. Therefore it is not necessary to include a REQUEST card for a permanent file or to include the CATALOG card which makes the file permanent. In place of the "NOGO" card of the previous section we have here card vii), which causes the overlayed program to be loaded to complete the overlay buildin process and then to commence execution.

In the various data sets, the data for the Segmented Loader (appearing between cards ix and x) remains the same as that for the generation of the overlay file of the previous section. The data set for the UPDATE program will contain the UPDATE directives and the program changes interspersed according to the conventions expected by UPDATE. The last card of this data set for UPDATE is the same as that used in the previous example and writes a copy of the modified prograı to the file COMPILE in a form acceptable to the FORTRAN compiler. Between cards x) and xi) appears the data set for SAM1. The JOB card parameters for such a job will depend on the magnitude of the changes, the number of iterations required, and the type of data supplied to SAM1.

Appendix 1

```
**       PROGRAM SETUP(INPUT,OUTPUT,TAPE6=OUTPUT,TAPE8,TAPE9,TAPE1Ø)
  C   THIS PROGRAM SPLITS UP A SAM1 MASTER TAPE INTO TWO SECTIONS
  C   PROGRAM STRIP IS PUT ONTO TAPE9 AND THE REST OF THE MASTER
  C   TAPE IS PUT ONTO TAPE1Ø
         INTEGER CARD(8Ø),BLANK,IEND(3)
         DATA BLANK,IEND/' ','E','N','D'/
         IR=8
         IW=9
**C   BLANK OUT FIRST 4 COLUMNS OF FIRST TWO CARDS
**       DO 1 I=1,2
**       READ (IR,1ØØØ) CARD
**       DO 2 J=1,4
**     2 CARD(J)=BLANK
**     1 WRITE (IW,1ØØØ) CARD
    1ØØØ FORMAT (8ØA1)
   C   TRANSFER PROGRAM STRIP  (12 SUBROUTINES) TO TAPE9
         ISUB=13
     1ØØ NSUB=1
     2ØØ IF (NSUB.EQ.ISUB) GO TO 5ØØ
     3ØØ READ (IR,1ØØØ) CARD
   C   END OF FILE 8?
         DO 4 I=1,3
         IF (CARD(I).NE.IEND(I)) GO TO 7ØØ
       4 CONTINUE
         WRITE (IW,1ØØØ) CARD
         WRITE (6,1ØØ1) IW
    1ØØ1 FORMAT (' END OF FILE WRITTEN TO UNIT ',I3)
         STOP
   C   END OF SUBROUTINE?
     7ØØ DO 3 I=7,9
         IF (CARD(I).NE.IEND(I-6)) GO TO 4ØØ
       3 CONTINUE
         WRITE (IW,1ØØØ) CARD
         NSUB=NSUB+1
         WRITE (6,1ØØ2) NSUB,IW
    1ØØ2 FORMAT (' NSUB= ',I3,'  WRITTEN TO FILE ',I3)
         GO TO 2ØØ
     4ØØ WRITE (IW,1ØØØ) CARD
         GO TO 3ØØ
    C   TRANSFER REST OF FILE 8 TO FILE 1Ø
     5ØØ IW=IW+1
         ISUB=1ØØ
         GO TO 1ØØ
         END
```

Cards marked (**) should not be included if the program is run on an IBM 360 computer.

Appendix 2

The following card images define the tree structure and specify in which overlay segment, each SUBROUTINE and COMMON block is to be placed for the program SAM1.

```
GUAVA     TREE       SAM1-(READIN,KAPP,PFSAHA-(POPS,FINISH))
SAM1      GLOBAL     ABROSS,ABTOT,ALTDEP,CONV,DEPART,ELEM,FLUX,FREQ,HEIGHT
,,HOT,IF,IFOP,IONS,IRNTOP,ITER,JUNK,KAPPA,LUKE,MUS,OPS,OPTOT,PTOTAL,RAD,
,RATES,READ,RHOX,SETFRE,STATE,TAUSHJ,TEFF,TEMCOR,TEMP,TURBPR,WAVEY,XABUN
,D,DISFNC,COOL,FRESET,MOLEQ,MOLEQU,IOUNIT,FREAUX,MODE
SAM1      INCLUDE    SAM1,PUTOUT,TCORR,STATEQ,DERIV,INTEG,MAP1,EXPI,W,TTAU
,P,BLOCK1,RADOSS
READIN    INCLUDE    READIN,DUMMYR,FREEFR,FREEFF,IWORDF
PFSAHA    INCLUDE    PFSAHA,PARTFN,SOLVIT
POPS      INCLUDE    POPS,NELECT,MOLEC,NMOLEC
FINISH    INCLUDE    FINISH,OUTPUT,FTCORR,FSTAT,CONVEC
KAPP      INCLUDE    KAPP,COULX,COULFF,JOSH,BLOCKJ,BLOCKH,HOP,HMINOP,H2PLO
,P,HRAYOP,HE1OP,HE2OP,HEMIOP,HERAOP,ELECOP,H2RAOP,HLINOP,STARK,COOLOP,C1
,OP,MG1OP,AL1OP,SI1OP,SEATON,LUKEOP,N1OP,O1OP,MG2OP,SI2OP,CA2OP,HOTOP,LI
,NOP,LINSOP,XLINOP,XLISOP,XCONOP,XSOP
          END
```

Appendix 3

The following data set for the Segmented Loader must be used when creating the overlay module for the program OPTAB.

```
OPAC      TREE      OPTAB-(READA,POPS,KAPP)
OPTAB     GLOBAL    COOL,DISFNC,DEPART,ELEM,FREAUX,FREQ,FRESET,HOT,IF,IFO
,P,IONS,JUNK,KAPPA,LUKE,OPS,OPTOT,PRESS,RHOX,STATE,TEMP,WAVEY,XABUND,ITE
,R
OPTAB     INCLUDE   OPTAB,DERIV,INTEG,MAP1,BLOCKDO
READA     INCLUDE   READA,FREEFR,FREEFF,IWORDF
POPS      INCLUDE   POPS,MOLEC,NMOLEC,PFSAHA,PARTFN
KAPP      INCLUDE   KAPP,COULX,COULFF,HOP,HMINOP,H2PLOP,HRAYOP,HE1OP,HE2O
,P,HEMIOP,HERAOP,ELECOP,H2RAOP,HLINOP,STARK,COOLOP,C1OP,MG1OP,AL1OP,SI1O
,P,SEATON,LUKEOP,N1OP,O1OP,MG2OP,SI2OP,CA2OP,HOTOP,LINOP,LINSOP,XLINOP,X
,LISOP,XCONOP,XSOP
          END
```

References

Kurucz, R.L., 1970 SAO Special Report No. 309.

Wright, S.L., 1975 Comm. University of London Observatory, No. 76.

CDC Cyber 70 Series, 6000 Series, Loader Reference Manual, Publication Number 60344200, Revision B, 1972.

IBM System/360 OS Linkage Editor and Loader Manual, Order No. GC28-6538-8. 9th edition, 1969. Revised by TNL GN28-0423, 1970.