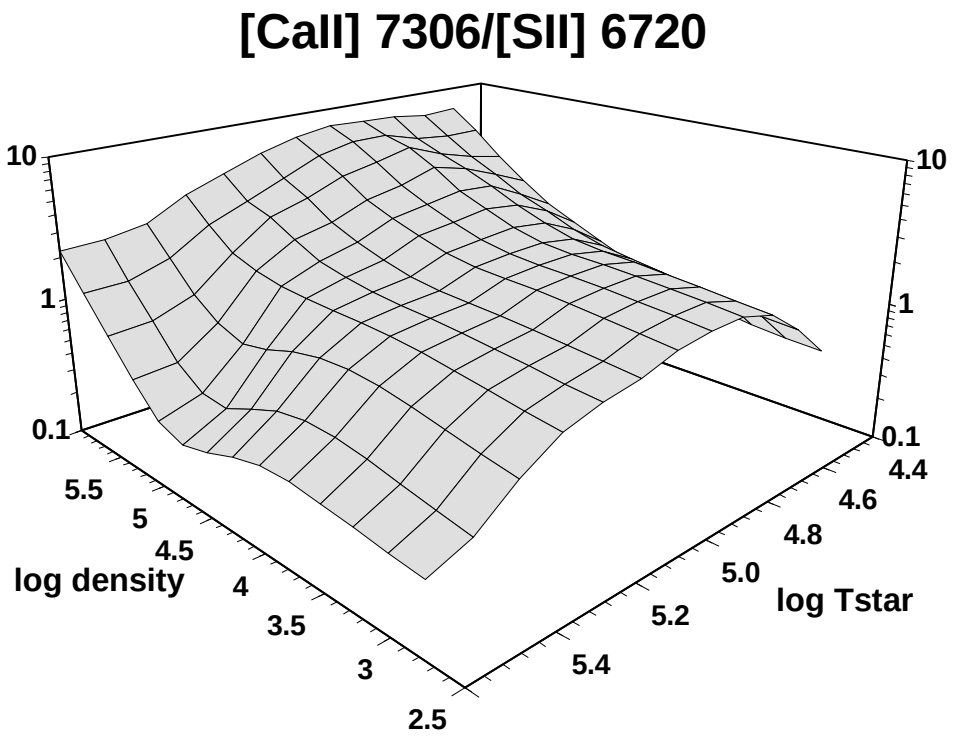
Hazy III

a brief introduction to Cloudy 90

Machine Environment and Test Cases

G.J. Ferland
Department of Physics and Astronomy
University of Kentucky, Lexington
http://www.pa.uky.edu/~gary/cloudy



CLOUDY is an evolving code. Updates are made on a roughly quarterly basis, while major revisions occur roughly every three years. You should confirm that you have the most recent version of the code by checking the web site http://www.pa.uky.edu/~gary/cloudy or asking to be placed on the CLOUDY mailing list.

October 21, 1997.

CLOUDY 90

G. J. Ferland
Department of Physics and Astronomy
University of Kentucky
Lexington

Table of Contents

List of Figures

List of Tables

1. MACHINE ENVIRONMENT

1.1. Overview

This section describes the machine environment needed to run CLOUDY. The code is designed to run in single precision on IEEE 32-bit computers, and has been tested on a Cray, Sun Sparcstation, SGI, DEC Alpha, HP Exemplar, and PC. Calculations on all machines produce essentially identical results.

The **`readme.htm`** file that is included with the Fortran source contains the most recent information about compiling and setting up the code.

1.2. Floating Point Environment

The floating-point environment should be set to ignore floating-point underflow but crash on any other floating-point error. Floating-point underflow is an unavoidable consequence of the attenuation of radiation as a beam of light is extinguished by an absorbing medium; underflow error checking should be disabled.

The code can treat certain situations differently if it is used on a long-word machine rather than an IEEE 32-bit machine. There is a logical variable ***bit32***, the sole element of the common block of the same name, which indicates whether (true) or not (false) the code is running on a 32-bit cpu. This test is performed in routine ***zero*** as the code is initialized.

Floating point overflow or division by zero *must never* occur, nor should library function domain errors (i.e., the log of a negative number). I would appreciate hearing about these errors. I can't fix it if I don't know it is broken. My Internet address is gary@cloud9.pa.uky.edu. Please send the input file and version number.

1.3. CLOUDY via ftp or the Web

The home page for CLOUDY is http://www.pa.uky.edu/~gary/cloudy. This is the preferred method of obtaining the code. This is the best place to retrieve the source since it is the only place certain to have the current version.

CLOUDY is an evolving code. Bugs are fixed as soon as they are discovered, and major revisions occur from time to time. It is important to check the web page several times per year to see what changes and improvements have occurred. I also maintain a mailing list to announce changes to the code. To be placed on this list send a request to cldmail@pa.uky.edu.

I would like to keep track of what the code is being used for, if only to try to make sure that it is within it range of validity. I would appreciate receiving preprints of any work done using CLOUDY.

The bibliographic reference to this version of CLOUDY, which should be cited in any paper that uses it, is given on the inside cover of this document. The citation should include the subversion number, i.e., version 90.09d.

1.4. CLOUDY in Lexington

1.4.1. Versions of the Code

Many versions of CLOUDY live on the UNIX network in Lexington. In general there will two versions of the code – a fully tested (older) version, and the current development version.

Software development goes through several phases. The initial version of a code is often referred to as an "alpha" version, the nearly de-bugged version as a "beta" version, and the fully tested version as the "gold" code. At any one time the last gold version of the code will live in a directory with a name indicating the version number. The version of CLOUDY is indicated by two parts - an integer indicating the version number, and a decimal part (perhaps with a letter following it) indicating the sub-version. Sub-versions to the gold code indicate fixes of bugs or stability problems, or major changes in the atomic data. The last gold version of CLOUDY was 90. An example of the current gold version number might be something like 90.09d. The source and executable for this version will live in a sub-directory with the name gary/c90. The current development version of the code lives in a directory called gary/cloudy.

Within the sub-directories the architecture is the same. The Fortran source files have names that end in .for. The executable has a name which indicates which processor it is intended for. Executable for a Sparc will have the name c.sun4, while those for the alpha will have the name c.alpha.

1.4.2. Running a single model

CLOUDY is often used to read in the parameters for a single model, and compute the result. The easiest way to do this is to create a small file which contains the input stream for that model. As a typical case consider a simple planetary nebula:

```
hden 4
radius 17
black body 100,000K, luminosity 38
```

Assume this is saved as the file **pn.in**. (CLOUDY stops reading the input steam when it reaches a blank line, or the end of file.)

I created a shell script with the name **run** which is in my "bin" directory, which I include on my path. The shell script **run** consists of the following:

```
echo reading input file $1.in
case $# in
0) echo there must be an input file ;;
1) /homeb/uwc0/gary/cloudy/c.sun4<$1.in >$1.out
   echo created output file $1.out ;;
2) /homeb/uwc0/gary/cloudy/c.sun4 < $1.in >$2.out
   echo created output file $2.out ;;
esac
echo $p
exit
```

run normally works by reading in a file called **file.in**, and creating a file called **file.out**.

If **run** is executed with no input parameters it will complain that at least one argument is needed and then stop. If there is one parameter it is treated as the name of the input and output files. So in the above example, typing

```
run pn
```

would read the input stream in **pn.in** and create an output file of results called **pn.out**. When two parameters occur the first is the name of the input stream and the second is the name of the output steam. The example

```
run pn test
```

would read the file **pn.in** and create the file **test.out**.

2. CLOUDY AS A SUBROUTINE

2.1. Overview

It is possible to use CLOUDY as a subroutine of other, much larger, codes. When used this way a series of subroutine calls, described next, are used to initialize the code, specify the initial conditions, drive the code, and examine the predictions.

It is also possible to drive the code by accessing its common blocks directly. In fact, this was the preferred method before version 80. The major problem with this approach, besides its inconvenience, is that the code and its common blocks are constantly changing, so that the main driver code will also have to be changed to keep up with CLOUDY. The following subroutines access the common blocks themselves, and (it is hoped) their use *will not* change with time.

A common strategy is to call the code to compute line intensities for a large matrix of parameters. The results of one such calculation is shown in Figure 1. Such grids can be computed in a few dozen hours on modern workstations, and offer far greater insight to physical effects of changing model parameters, than does a single model.

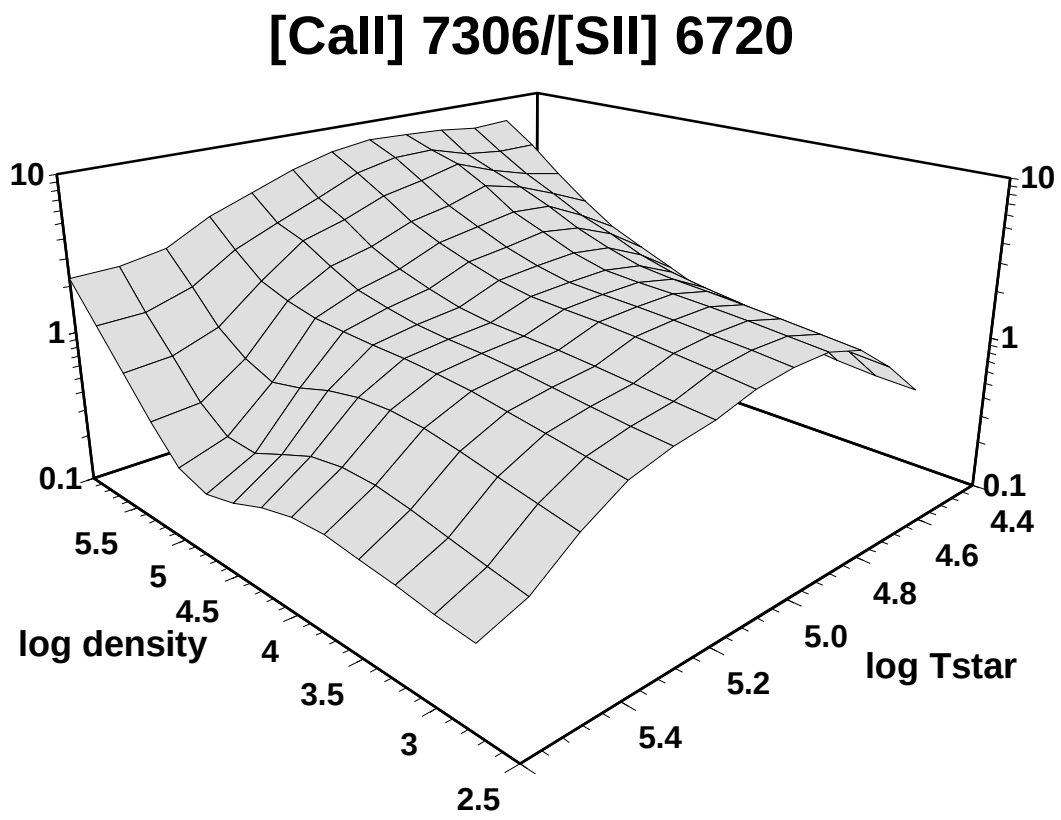


Figure 1 The results of a large grid of model calculations are shown. The x-y plane show the logs of the hydrogen density and stellar temperature. The z axis is the indicated line ratio, for solar abundances.

2.1.1. Remove the Old Main Program

The first step in calling the program as a subroutine is to replace the existing main program with one that you write. Somewhere within the source for the code lies the following main program:

```
************************************************************************
      program maincl
      implicit none
*main program that calls cloudy when used as a stand-alone program
*
*     local variables
      logical lgOK
      call cdInit
      call cdDriv( lgOK )
      write(qq,'('' Cloudy ends:'',i3,'' warnings,'',i3,'' cautions,'',
     1i3,'' temperature failures,'',i4,'' zones,'',i3,'' iterations.''
     2)') nwarn , ncaun , nfail , nzone, iter
*
      stop
      end
```

This is the code that is executed when CLOUDY is called as a stand-alone program. It must be deleted so that the program you write will be loaded instead. The following subsections outline how to write code for this new main program.

2.1.2. Computing a grid of models

It is often better to compute an entire grid of models at one time. In this case a main program is written which drives CLOUDY by calling a series of subroutines (described beginning on page 404 of this document). The version of the main program that is included in the source of CLOUDY should be deleted.

If the main program is called **`grid1.for`**, it could be compiled and linked with the CLOUDY object files as follows. I define a shell script named "**`compile`**" which contains the commands

```
echo Begin compilation of $1.for
f77 -fast -c $1.for
echo Compilation of $1.for is complete
```

so that **`grid1.for`** could be compiled with the command

```
compile grid1
```

Next it is necessary to link into CLOUDY itself. This is done with a command called **`linkedit`**, which consists of the following:

```
echo about to link $1.o using cloudy object files
f77 -fnonstd -fast -c -o c.sun4 `ls /homeb/uwc0/gary/cloudy/*.o`
echo created executable $1.exe
```

The command

```
linkedit grid1
```

will produce the executable file **`grid1.exe`**. Executing it as

```
grid1.exe > grid1.out
```

will produce the output file **`grid1.out`**.

2.2. Writing a New Main Program

2.2.1. Initializing the Code

Many variables must be initialized at the beginning of the calculation. This is done by calling subroutine ***cdInit***.

```
      call cdInit
```

Subroutine ***cdInit*** must be called every time a new calculation is to be performed, *before* calling any of the following subroutines, but after the results of previous calculations have been read.

2.2.2. Entering Commands

The input stream used to specify the initial conditions for the calculation consists of a series of 80 character free-format command lines. They must obey all the rules outlined in Part I, start in column 1, and contain meaningful information across all 80 columns. These command lines are entered by successive calls to routine ***cdRead***.

```
      character*80 chLine
      write(chLine,'(''constant pressure'')')
      call cdRead( chLine , nleft )
*
      hden = 3.5
      write(chLine,'(''hden='',F5.1)') hden
      call cdRead( chLine , nleft )
```

The first argument of ***cdRead*** is the line image, an eighty character variable. It must obey all the rules for normal command lines. It is assumed to end with column 80 or one of the special characters outlined in the section on command line format. The second integer variable in the call to ***cdRead*** is the number of command lines that can still be entered. It is not now possible to read in more than 1000 command lines because of limits to the size of the character arrays used to store them. This limit is stored as the variable ***nkrd*** that occurs in several parameter statements throughout the code. If more than 1000 lines are read in by calling ***cdRead*** then ***cdRead*** will stop after explaining why. It will be necessary to increase ***nkrd*** throughout the code if more than 1000 command lines are needed.

Warning!! The following is deceptively simple but *will not* work:

```
call cdRead('hden 4' , nleft )
call cdRead('blackbody 5' , nleft )
```

since the contents of the character string are not defined beyond the last specified character.

2.2.3. Controlling Output

CLOUDY normally writes its standard output on Fortran unit 6. This can be changed to another output unit number by calling routine ***cdOutp***. By combining this redirection with the Fortran **open** statement (which assigns unit numbers to specific output files) it is possible to have the standard output placed in any file.

```
      Open(56,file='test.out')
*     put output on FORTRAN unit 56
      call cdOutp( 56 )
```

CLOUDY normally speaks what's on its mind. However, it does have a quiet mode in which nothing at all is printed. This quiet mode is set by the logical argument to subroutine ***cdTalk***.

```
*     set no output at all
      call cdTalk( .false. )
*     have the code produce the normal printout
      call cdTalk( .true. )
```

The default is for CLOUDY to produce output, and ***cdTalk*** does not have to be called if this is desired. However, it does need to be called with the logical variable false if the quiet mode is desired.

2.2.4. Calling the Code

The calculation is performed when routine ***cdDriv*** is called. ***cdDriv*** is a subroutine with a single logical variable. The following shows an example of its use.

```
      logical lgOK
      call cdDriv(lgOK )
```

If the calculation was successful then the logical variable (in this example given the name lgOK) is set true by the code. If problems occurred and the results cannot be trusted then the variable is set false. This will only be set if the calculation suffered a complete meltdown. It is possible that warnings or cautions are present but ***lgOk*** is still true, and routine ***nwcns*** (page 409) should be called to check if they occurred. If output was turned off (with ***cdTalk***) then it will probably be necessary to turn output on and recompute the model to find out what went wrong.

2.2.5. Checking without Computing

If routine ***cdNoex*** is called after ***cdInit*** but before ***cdDriv*** then only the initial parts of a calculation will be performed when routine ***cdDriv*** is called.

```
      call cdInit
*     read in commands
      call cdRead .....
*     tell it not to execute
      call cdNoex
*     call the code
      call cdDriv
```

When ***cdDriv*** is called the code will generate the incident continuum, set the initial density, and the chemical composition. It will then stop just before the initial search for the physical conditions in the first zone. All of the initial printout, summarizing properties of the composition and continuum, will be generated. This provides a quick way to check that a large grid of models will be specified correctly, without actually fully computing the grid.

2.3. Checking Predictions

2.3.1. Reading the Emission-Line Array

The predicted line intensities or luminosities for all lines with non-zero values are stored within a set of common blocks which also contain the line identifiers, a four character label and integer wavelength. These are normally printed at the end of the calculation. It is also possible to obtain the line formation by calling subroutine ***cdLine***. In this case the label and wavelength of the line are specified, and the

subroutine returns the relative intensity, and the log of the absolute intensity or luminosity.

```
      character*4 chLabel
      logical lgOK
*
      chLabel = 'totl'
      length = 1216
      call cdLine( chLabel , length , relint , absint , lgOK)
```

The first variable is the line label, the four-character string (upper or lower case) as used by the code to identify the line. The second integer variable gives the wavelength of the line. Both of these must exactly match the label and wavelength used by CLOUDY to identify the line (see the chapter "Lines" in Part IV of this document). If ***cdLine*** finds the line then the logical variable that is the last argument in the call to ***cdLine*** is set to true, and this variable is set to false if the line is not found. This occurs if the line had zero intensity (and so may not have been stored) or if the line wavelength and label were mistyped. The third variable (***relint*** in the above example) is the relative intensity of the line (relative to the normalization line, usually Hβ, but set with the **normalize** command). The log of the intensity (erg cm^{-2} s^{-1}) or luminosity (erg s^{-1}) of the line is returned as the next variable (***absint*** in the above example). If the intensity of the line is zero or the line was not found then this variable will be set to -37.

A second routine, ***cdEms***, functions much the same as ***cdLine***, but returns the local emissivity (erg cm^{-3} s^{-1} for unit filling factor) of the line for the last computed zone. The first two arguments are same, but the third is the emissivity, and the last is the logical variable ***lgOK***.

2.3.2. Reading the Column Density Array

The predicted column densities can be accessed by calling the subroutine ***cdColm***.[1]

```
      character*4 chLabel
      logical lgOK
*
      chLabel = 'carb'
      ion = 3
      call cdColm( chLabel , ion , colum , lgOK )
      write(6,*)'The predicted column density is', COLUM
```

where the subroutine returns ***colum***, the predicted column density (linear, cm^{-2}). ***chLabel*** is a four character identifier which must agree with the first four characters (upper or lower case) of the name used to indicate the element in the printout. The integer variable ***ion*** is the spectroscopic designation of the level of ionization, i.e., 1 indicates C I, 3 indicates C III, etc. The last logical variable indicates whether the column density was returned OK.

2.3.3. Reading the Ionization Fractions

The predicted ionization fractions, averaged over radius or volume, can be accessed by calling the subroutine ***cdIonf***.

[1]The OK parameter was introduced in version 81.

```
      character*4 chLabel , chWeight*6
      logical lgOK
*
      chLabel = 'carb'
      ion = 3
      chWeight = 'volume'
      call cdIonf( chLabel , ion , frac , chWeight , lgOK )
      write(6,*)'The predicted ionization fraction is',
     1 frac
```

The subroutine returns ***frac***, the predicted ionization fraction A_{ion} / A_{tot}. ***chLabel*** is a four character identifier which must agree with the first four characters (upper or lower case) used to indicate the element in the printout. The integer variable ***ion*** is the spectroscopic designation of the level of ionization, i.e., 1 indicates C I, 3 indicates C III, etc. ***chWeight*** is a six character variable which can be either "radius" or "volume" (either upper or lower case). This determines whether the ionization fraction returned is weighted over radius or volume. The last logical variable indicates whether the ionization fraction was returned OK.

2.3.4. cdGetTe - reading the temperature

The electron temperature for the last computed zone is obtained by called routine ***cdGetTe***, whose single argument returns the predicted temperature, a single precision real variable.

2.3.5. cdGetPres – pressure

The pressure for the last computed zone is obtained by called routine ***cdGetPres***. This routine has three arguments, the total (gas plus radiation) pressure, the gas pressure, and the radiation pressure. All are single precision real variables.

2.4. Printing the Comments

After the calculation is complete, but before the emission lines are printed, the code generates a series of statements which indicate warnings, cautions, comments, and surprises. These should be examined to confirm that the calculation is probably all right. A series of subroutines allows the driving code to determine whether these comments were generated, what type they were, and to print then on an arbitrary IO unit.

2.4.1. Were comments generated?

Two routines are provided to check on problems in the calculation.

Subroutine ***cdNwcns*** will return the number of warnings, cautions, surprises, notes, and temperature and pressure failures:

```
      call cdNwcns( nw , nc , nn , ns , nte , npe )
```

where ***nw*** is the number of warnings generated (if this number is non-zero, then the calculation has serious problems), ***nc*** is the number of cautions generated (these are less severe than warnings, but are still a cause of concern), and ***nn*** and ***ns*** are the number of notes and surprises. The last two arguments are the number of temperature and pressure failures.

2.4.2. Printing the comments.

The comments may be printed on an arbitrary Fortran IO unit by calling the series of subroutines described here.

```
*     output the comments on unit 6, the normal output
      io = 6
*     print the reason the calculation stopped, and geometry
      call reageo( io )
*     print the warnings
      call warnot( io )
*     next print the cautions
      call caunot( io )
*     now print the surprising results
      call bangot( io )
*     now print the notes
      call noteot( io )
```

reageo(io). It is very important to understand why the calculation stopped. The first two lines after the last zone results give the reason the calculation stopped, and the type of geometry. This information will be printed on Fortran unit IO when this routine is called.

warnot(io) All warnings will be printed on unit IO.

caunot(io) All cautions will be printed on unit IO.

bangot(io) All surprises (denoted by a "!", which is called a "bang" in standard Unix-speak) are printed on unit IO.

noteot(io) The notes concerning the calculation will be printed on unit IO.

2.4.3. cdErrors(io) - printing a summary of any problems

Routine ***cdErrors***(io) will generate a summary of any problems that happened during a calculation. The argument is the punch unit to be used for the output. If problems occurred in the calculation, such as temperature or pressure failures, warnings, or cautions, these will be printed following the title for the calculation.

2.5. Example Call as a Subroutine

The following is an example of a very simple use of CLOUDY as a subroutine. The normal output will be produced on Fortran unit 6.

```
      program main
      logical lgOK
      character*80 chLine
*     initialize the code
      call cdInit
*     write commands into character variable LINE,
*     then feed into code with CDREAD
      write(chLine,'(''hden 4 '')')
      call cdRead(chLine , nleft )
      write(chLine,'(''black 5 '')')
      call cdRead(chLine , nleft )
      write(chLine,'(''stop zone 1 '')')
      call cdRead(chLine , nleft )
      write(chLine,'(''ionization -2 '')')
      call cdRead(chLine , nleft )
*     now call the main driver, output will be produced
      call cdDriv(lgOK)
*     print any errors on unit 11
      call cdErrors(11)
```

```
*     now read and print the Ly-alpha flux
      call cdline( 'totl',1216,R1216,A1216,OK)
      If(.not. lgOK) then
         write(6,'('' Did not find line.'')')
      else
         write(6,'('' Ly-alpha='',1P,E10.2)') R1216
      endif
      end
```

2.6. Co-adding Many Calculations; on-the-fly

It is possible to co-add the results of many calculations performed by successive calls to the subroutines described in this section. Two ways of doing this are possible. The first is on-the-fly, and is described here. A later subsection describes a strategy for creating a large grid of models, and then analyzing this grid ex post facto. The second method is more useful since intermediate results are stored permanently.

In the first method the series of calculations is initialized, then a grid of models are co-added with variable weights. ***wginit*** is used to initialize the series of calculations, ***wgadd*** is used to add in a specific model with variable weight, and ***wgline*** is used to examine the resulting (co-added) emission-line array.

2.6.1. Initializing the series

The series of calculations to be co-added are begun with the single call to routine ***wginit***. It has no arguments, and is called only once, before the first calculation that is to be co-added. This is to be contrasted with ***cdinit***, which is called before *each* individual calculation.

2.6.2. Adding the model

The emission lines predicted by a particular model are co-added (with weights) to the predictions of the previous set of models by calling routine ***wgadd***. It has a single argument, the weight to be assigned to the results of the previous calculation. In the end the integrated intensities of the set of models is given by

$$I_{sum} = \sum w_i \, I_{pred} \qquad (400)$$

where ω_i is the weight for the model *i*, and I_{pred} is the intensity predicted by model *i*. Note that the final result is not renormalized to a summed weight of unity.

2.6.3. Reading the line array

The predictions of the sum given above are examined by calling routine ***wgline***, which has the same series of arguments as ***cdline*** (described on page 407). When a series of models are computed, ***cdline*** can be called to examine the predictions of the previous calculation, while ***wgline*** is called to examine the cumulative results of the series of co-added models.

2.6.4. An example

The following example shows calls to the routines described in this section, used to co-add the predictions of a series of calculations.

```
*
*     call WGINIT at the start to set the series up
      call wgInit
*
*     this is the series of calculations
      doi=1,5
*       call cdInit before each one
        call cdInit
*       write commands into character variable line,
*       then feed into code with cdread
        hden = float(i)
        write(chLine,'(''hden '',F6.4)') hden
        call cdRead(chLine , nleft )
        write(chLine,'(''black 5 '')')
        call cdRead(chLine , nleft )
        write(chLine,'(''stop zone 1 '')')
        call cdRead(chLine , nleft )
        write(chLine,'(''ionization -2 '')')
        call cdRead(chLine , nleft )
*
*       now call the code
        call cdDriv( lgOK )
*
*       the weight for this calculation is given by
*       the following function
        weight = 1. / (i * i)
*       now co-add the previous calculation
*       with this weight
        call wgadd( weight )
      enddo
*     the series of calculations are now complete,
*     examine the cumulative results with wdline
      call wgline( 'totl' , 1216 , reline, absint , OK )
```

2.7. Analyzing Stored Grids of Calculations

This subsection describes a strategy for creating and saving a large grid of models, and then analyzing this grid ex post facto.

The **`punch results last`** command is described in Part I. A large number of the quantities computed by the code will be written into a file if this command is entered. Any post-processing software can read this file, but reading this file is facilitated by routine ***cdGett***, described next.

2.7.1. Reading the stored grid

The output produced by the **`punch results last`** command can be read with subroutine ***cdGett***. ***cdGett*** has two arguments. The first is the Fortran I/O unit number for the file containing the large grid of models. The second argument is a logical variable that indicates whether (true) or not (false) the end of file was encountered when the model results were read. If the logical variable is true then the end of file was encountered and no results were read. If it has not been encountered then there may be more models further down the input stream. The next model in the grid will be accessed by the next call to ***cdGett***.

2.7.2. Reading intensities, ionization fractions, and column densities

Once ***cdGett*** has been called the routines ***cdLine***, ***cdIonf***, and ***cdColm*** routines can be used to obtain emission line intensities, ionization fractions, or column densities from the stored file. A common block, ***getpar***, is associated with ***cdGett***. This common block contains information concerning the initial parameters of the current model, as specified in the input stream.

2.7.3. An example

One or more calculations can be within the file to be read. If a sequence of models is within the file then the next model in the sequence will be read by each call with the ***cdGett*** command. The following examples illustrate producing, then reading, the results of a series of models.

```
*
*     open a new file
      open(7,file='example.out')
*     this is the series of calculations
      doi=1,5
*       call cdInit before each to initialize the code
        call cdInit
*       write commands into character variable line,
*       then feed into code with cdread
*       following punches output on Fortran unit 7
        write(chLine,'(''punch results last 7'')')
        call cdRead(chLine , nleft )
        hden = float(i)
        write(chLine,'(''hden '',F6.4)')hden
        call cdRead(chLine , nleft )
        write(chLine,'(''black 5 '')')
        call cdRead(chLine , nleft )
        write(chLine,'(''stop zone 1 '')')
        call cdRead(chLine , nleft )
        write(chLine,'(''ionization -2 '')')
        call cdRead(chLine , nleft )
*
*       now call the code
        call cdDriv( lgOK )
      enddo
```

The result of this program will be a file, called **`example.out`** in Unix, which contains information about the model. The following program fragment will obtain the intensity of Lyα for all of the models.

```
*
      logical eof
*     read in the results until eof
      open(7,file='example.out')
  999 call cdGett(7,eof)
      if( eof ) then
        stop
      else
*       read and print the Ly-alpha flux
        call cdLine( 'totl',1216,R1216,A1216,OK)
        if(.not. ok) then
          write(6,'('' Did not find line.'')')
        else
          write(6,'('' Ly-alpha='',1P,E10.2)') r1216
        endif
        goto999
      endif
```

3. OTHER DETAILS

3.1. Overview

This section largely outlines internal details of code variables, and how these relate to overall quantities. These are described only after the relevant portion of the code has become fairly mature, and not likely to undergo further major revision.

3.2. Overall Structure

This section outlines the flow control in the higher levels of the code.

3.2.1. Program MainCl

When used as a stand-alone program, control passes to program ***MainCl***, which calls ***cdInit***, ***cdDriv***, and then stops. c***dInit*** sets the logical variable ***called*** false indicating that the input stream should be read in from standard input. ***cdDriv*** decides whether to read in commands from the input stream (by calling ***fillar***) and then calls routine ***cloudy*** (which computes a single model) or to call ***drvary***, the routine to optimize parameters to match a set of observations. The organization is shown in Figure 2.

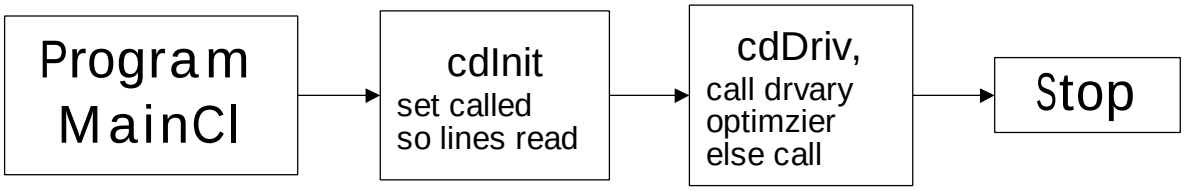


Figure 2 This figure shows the structure of the main program, MainCl.

3.2.2. Subroutine Cloudy

Most of the actual work performed in the computation of a model is done is the main subroutine CLOUDY (Figure 3). This subroutine controls the zone and iteration variables ***nzone*** and ***iter***. These are the only variables used or altered by this subroutine.

3.2.3. Subroutine PIonte

PIonte is the routine which converges the local pressure, or specification of the gas density. Its major loop calls routine ***presur***, which determines what the local density / pressure should be, and sets the variable ***presok*** to true if the conditions are correct. Next routine ***ionte*** is called. The routine loops until heating and cooling match (the variable ***tfail*** is set false), or a temperature failure occurs (***tfail*** is set true). The Boltzmann factors are evaluated next in routine ***boltgn***. The overall structure is shown in Figure **5**.

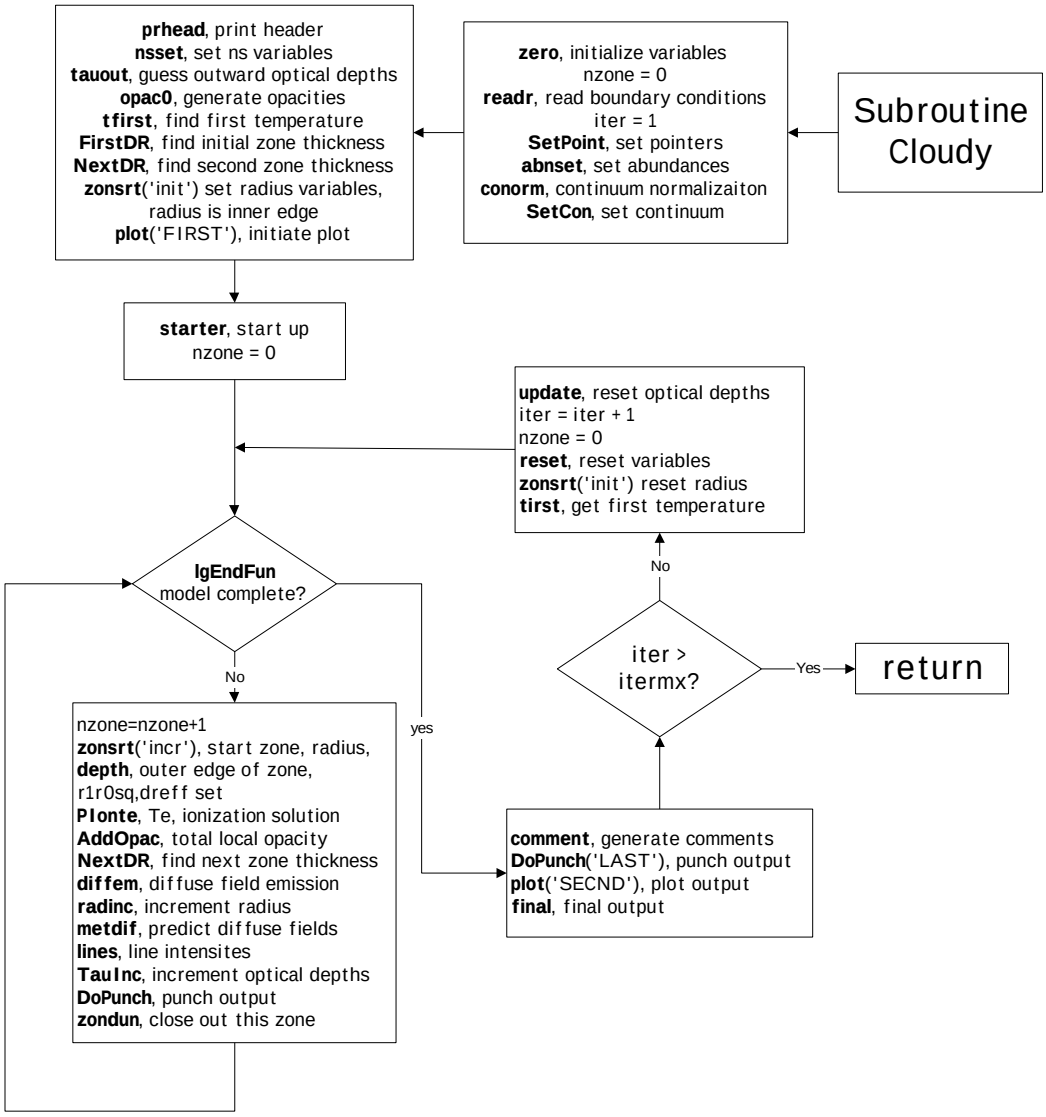


Figure 3 This figure shows the structure of subroutine Cloudy.

3.2.4. Subroutine ionte

ionte is the routine that a) calls ***ConvIoniz*** to converge the ionization, and b) determines the electron temperature by balancing heating and cooling. An overview is shown in Figure **4**. Ionte totally controls the value of ***lgDoPhoto***. When ***lgDoPhoto*** is true the code completely reevaluates all opacities and photoionization rates. When false the rates are left at previous values, safe for second iterations.

3.2.5. Line radiative transfer routines

Figure 6 shows the series of routines that are called to evaluate line radiative transfer.

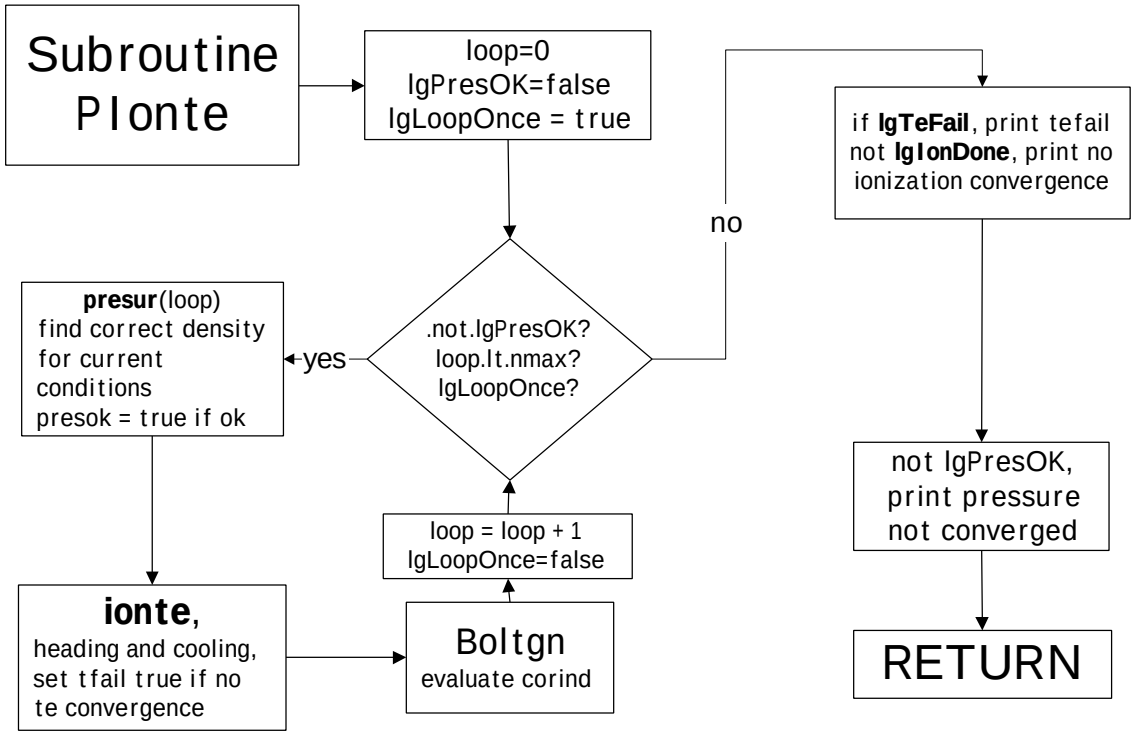


Figure 5 This figure shows the structure of subroutine Pionte. pionte .

3.3. Coding Conventions

CLOUDY is evolving towards a simple formulation of the Hungarian naming convention (Simonyi 1977). In this convention the first few characters of a variable name indicate the type and function of that variable.

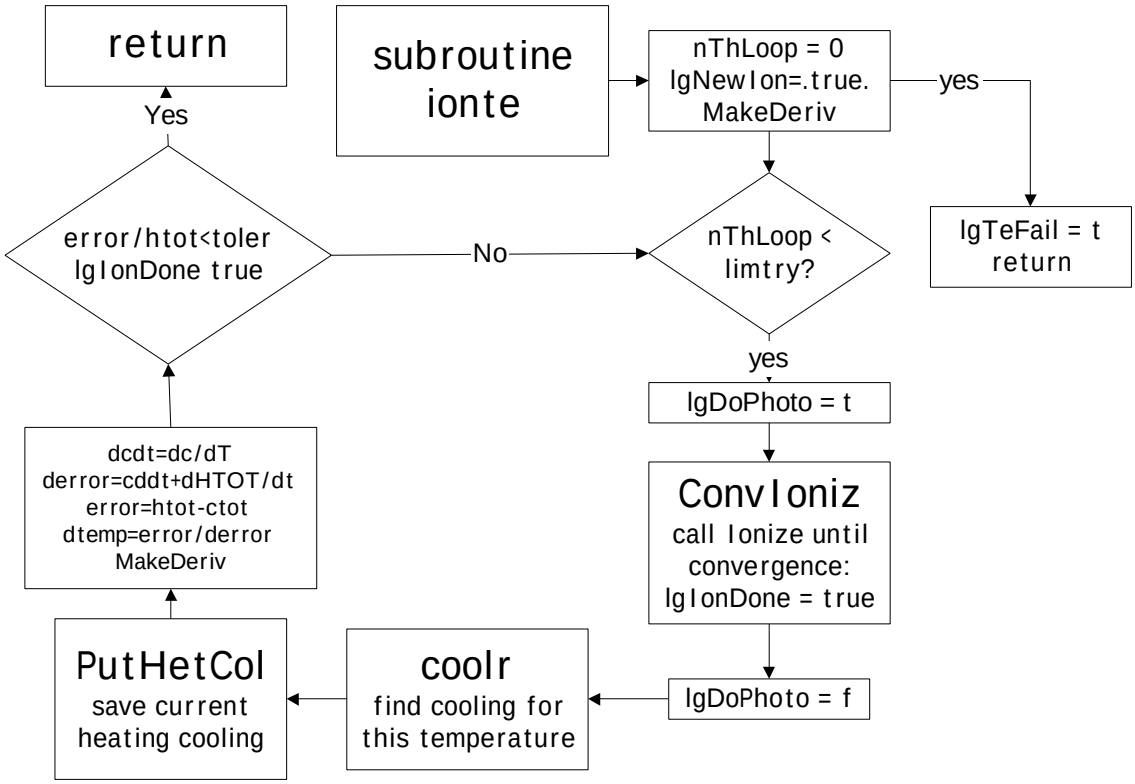


Figure 4 This figure shows the structure of subroutine ionte. ionte .

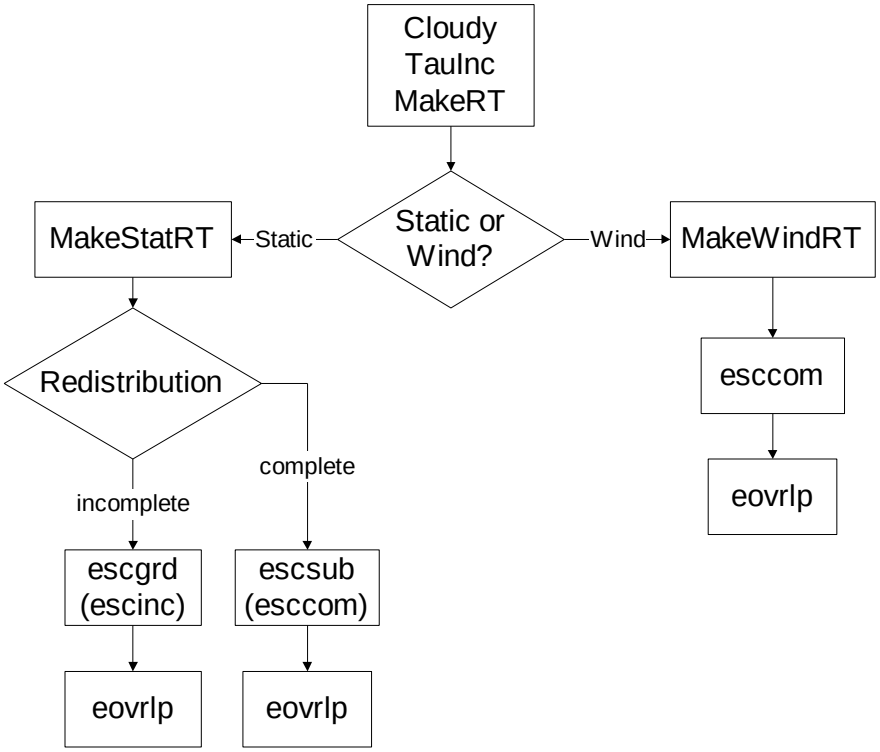


Figure 6 This figure shows the structure of the radiative transfer routines. radtrn

3.3.1. Strong Typing

In part the naming convention used in the code looks back to an under appreciated advantage in the FORTRAN II and FORTRAN 66 languages - the fully implicit designation of variable types by the first letter of its name. Implicit typing *is not* used in CLOUDY today - the current version is moving towards Fortran 90 and HPF, and makes full use of strong typing. Each subroutine and function begins with the statement **`implicit none`**. The naming convention forced by early versions of FORTRAN (integers begin with i-n, real numbers with other characters) is still useful since the type can be determined at a glance. This is used in the following.

3.3.2. Integers

Integers begin with the characters i, j, k, l, m, or n.

Counters begin with n. Examples include ***nLevel*** or ***nLoop***.

Loop indices are generally i, j, or k. Sometimes they are counters.

Pointers begin with ip. Examples include ***ipContinuum*** or ***ipCIV1549***.

3.3.3. Real, or floating point

These begin with letters between a through h, and o through z. Examples include ***PumpRate***, ***DestRate***, or ***CollisIoniz***.

In some cases floating numbers naturally will have names beginning with one of the letters reserved for integers. In this case a lower case x is used as the first character. Examples include ***xJumpDown***, ***xMoleDen***.

3.3.4. Characters

Character variables begin with "***ch***". Examples are ***chName*** or ***chReadInput***.

3.3.5. Logical variables

These begin with "l***g***". Examples are ***lgOK***, ***lgDone***.

3.3.6. Variables in equations

All variables must have exactly the same name in all routines, preferably by putting these in a single common and including this common with include statements. Variables are set in equations with a single space on either side of the equals sign. There are no exceptions to this rule. This makes it easy to find all occurrences of setting a given variable by using a pattern searcher such as **grep**.

3.3.7. Changes to the code

Changes to logical flow within the code that could impact results or convergence are indicated by a comment line just before the affected line, with the following style:

```
*     >>chng 95 dec 20 eden had eold, was undefined here, affect electron density convergence
```

The flag **>>chng yy** indicates a change. Here yy is the last two digets of the year. It is important that this style be followed consistently so that changes can be searched from the code with a pattern matcher such as **grep**, and then sorted by date.

3.3.8. Atomic data references

Codes such as CLOUDY only exist because of the foundation of basic atomic and molecular data. It is important to the survival of this field that the original sources of the basic data be cited, since this in turn affects their ability to generate support. The code follows the convention of preceeding all uses of atomic data with a citation to the original paper in the following form:

```
*     >>refer Berrington, K., AtData Nuc Data Tab 33, 195.
```

The flag >>refer indicates a reference. This style must be followed consistently so that a pattern search will generate a list of references used.

3.3.9. Sanity checks

Sanity checks are redundant tests for variable values that are totally impossible (Maguire 1993). Examples include negative collision strengths or electron temperatures. A major improvement to CLOUDY version 86 and later is the inclusion of large numbers of these sanity checks. These checks do not have a major impact on performance but they do slow the code down a bit. For production runs with a gold version of the code it would be reasonable to not include these checks. Unfortunately there are no standard FORTRAN 77 compiler directives for conditional compilations, so CLOUDY encloses sanity checks within a uniform set of comment lines marking the beginning and ending of the check, as shown below.

```
*begin sanity check
    if( te .lt. 0 ) then
       write(qq,'('' temperature is negative, this is impossible, ='',1p,e10.2)')te
       call showme
       stop
```

```
    endif
*end sanity check
```

I intend to follow this convention uniformly, without exception, to make it possible to remove sanity checks automatically.

3.3.10. Broken and Test Code

broken It is sometimes necessary to physically break the code, either by writing specific code to override the correct behavior or disable a physical process. When such code is entered, it should be accompanied by a call to routine ***broken***. This routine does nothing but set a flag that broken code is present. This flag generates a warning after the calculation is complete, to serve as a reminder of the presence of the broken code. This routine is not normally used.

TestCode Trial code is identified by a call to routine ***TestCode***. This routine does nothing but set a flag that test code is present. This flag generates a comment after the calculation is complete, to serve as a reminder of the presence of the test code. This routine is not normally used.

3.3.11. Routine descriptions

The first statement in each routine is **`implicit none`**. Each routine should have as its first comment following this, a line describing its main duties. This line will be picked up by the program that generates the list of routine names.

3.3.12. Version numbers

CLOUDY uses version numbers to keep track of changes to the code. The version number is stored in block data ***version***, in the common block ***date***. The variable ***chDate*** contains the date of the last major revision, and the variable ***chVersion*** is the actual version number. Both are character variables.

3.4. Geometry

This section defines the internal variables used to describe the geometry. The geometry is always spherical, but can be made effectively plane parallel by making the inner radius of the cloud much larger than its thickness.

Most variables having to do with the geometry are set and incremented in routine ***zonsrt***. The following gives the Fortran variable name, followed by its representation in this document.

rinner, $\mathbf{r_o}$ This is the separation between the center of symmetry (i.e., the center of the central object) and the inner edge of the cloud. It remains constant throughout the calculation. If an inner radius is not specified then it is given the default value of 10^{25} cm. This will usually result in a plane parallel geometry.

drad, $\mathbf{dr}$ This is the thickness of the current zone. Note that the zone size changes continuously throughout the calculation. Upper or lower limits to ***drad*** can be set with the **`drmax`** and **`drmin`** commands, described in Part I.

radius, $\mathbf{r}$ This is the distance between the center of symmetry and the *outer* edge of the current zone. For the first zone, ***radius*** has the value ***rinner + drad***.

depth*, Dr** This is the distance between the inner edge of the cloud and the *outer* edge of the current zone. For the first zone, ***depth has the value ***drad***.

A problem can arise under certain extreme circumstances. The depth variable ***depth*** must be increased for every zone, by adding the zone thickness ***drad***. Both variables are double precision. If the radio ***drad/depth*** falls below ~ 10^{-14} then the depth cannot be updated on most machines. The problem is that the sum ***depth*** + ***drad*** will be equal to ***depth*** because of numerical underflow. If this occurs (i.e., the zone thickness ***drad*** falls below ***depth*** / 10^{14}) the code will stop, with the comment than the zone thickness is too small relative to the depth. There is no obvious solution to this problem.

drnxt This will be the thickness of the next zone. The thickness of the zones is adjusted continuously during a calculation. Adaptive logic is used to ensure that the zones are large enough to be economical, but small enough to follow changes in the physical conditions across the nebula. This choice of the next zone thickness is done in routine ***NextDR***. The logic behind the choice of the zone thickness can be followed with either the **`trace dr`** or **`punch dr`** commands.

router This is the limit to the outer radius of the structure, as set before the calculation begins. The default value is effectively infinite, actually 10^{30} cm.

r1r0sq This is the sphericity ratio

$$\mathrm{R1R0SQ} = \left[\frac{\text{distance to center of zone}}{\text{inner radius}}\right]^2 = \left(\frac{\mathrm{RADIUS} - \mathrm{DRAD}/2}{\mathrm{RINNER}}\right)^2 \qquad (401)$$

pirsq This is the log of the inner area ($4\pi\, r_o^2$).

dReff This is the effective radius, $\delta\, r_{eff} = \delta\, r \times f(r)$ where f(r) is the filling factor.

dVeff This is the effective volume relative to the inner radius. The units of ***dVeff*** are cm, and it is equal to ***dReff*** if the geometry is plane parallel.

$$\mathrm{dVeff} = \left(\frac{radius - dRad/2}{rinner}\right)\left(\frac{\min\left(radius - dRad/2, \mathrm{cylind}\right)}{rinner}\right) dRad \times f(r) \qquad (402)$$

3.5. Column Densities

All column densities are updated in routine ***TauInc***. The main column densities used are the following.

colden The total hydrogen column density.

chi The neutral hydrogen (atomic) column density.

colhii The ionized hydrogen column density.

colh2 The hydrogen molecule column density.

chmin The negative hydrogen ion molecule.

chehp The HeH^+ column density.

ch2pls The H_2^+ column density.

3.6. Physical Conditions

3.6.1. Densities

densty The gas mass density in gm cm^{-3}.

eden This is the electron density, as evaluated in routine ***esum***. ***Eden*** is also controlled by other parts of the code, which allow it to change only gradually while looking for a new solution.

EdenTrue This is the correct electron density, and is evaluated in ***esum***. The electron density has converged when ***EdenTrue*** and ***eden*** are within ***EdenError*** of one another. ***EdenError*** is set in the large block data to 0.01. This variable is the sole member of the common block of the same name.

pden This is the number of particles per cubic centimeter. It is evaluated in ***TotalPressure***.

wmole This is a quantity related to the mean molecular weight, the mean AMU per particle. It is evaluated in ***TotalPressure***.

$$\mathrm{w}mole = \frac{\sum n_i m_i}{n_{tot}} \tag{403}$$

With these definitions the density ***densty*** (gm cm^{-3}) is the product of ***pden*** and ***wmole***.

edensqte This is the ratio

$$edensqte = \left(n_e + n_H 10^{-4}\right) T_e^{-0.5} \tag{404}$$

used in many collision rate equations across the code. It is evaluated in routine ***tfidle***. The second term in parenthesis approximately accounts for neutral collisions.

cdsqte This is the ratio

$$cdsqte = edsqte \times 8.629 \times 10^{-6} = \left(n_e + n_H 10^{-4}\right) T_e^{-0.5} 8.629 \times 10^{-6} \tag{405}$$

used in many collision rate equations across the code. It is evaluated in routine ***tfidle***.

3.6.2. Structure

The *struc* common block. This common block saves information about the structure of the model. It has several elements, each containing a saved quantity for a zone. These vectors are dimensioned ***nzlim*** long. ***nzlim*** is currently set to 2000, so only the first 2000 zones can be saved. This is the only thing that sets the limit to the number of zones whose information can be saved. Increase ***nzlim*** if more are needed.

ednstr The electron density of each zone is saved in this vector.

hiistr The H^+ density of each zone is saved in this vector.

histr The H^o density of each zone is saved in this vector.

heatsrt This is the total heating.

pdenstr This save the total number of particles per cubic centimeter.

radstr The effective thickness ***dReff*** (cm) of each zone is saved in this vector. This includes a filling factor if one was specified.

testr The temperature structure of the nebula is saved in this vector.

volstr The volume ***dVeff*** (cm^3) of each zone is saved in this vector. This includes a filling factor if one was specified.

3.6.3. Temperatures

te This is the local electron temperature.

tlast is the final temperature of the last computed zone. It is only meaningful for the second or greater zone.

alogte, ***alogete*** These are the base 10 and natural logs of the electron temperature. The array ***telogn*** contains powers of the base 10 log of the temperature.

telogn(i) This is a vector dimensioned 7 long, contained within the ***telogn*** common block. The n^{th} member of the array contains $\log(T_e)^n$.

alogete is the natural log of the temperature.

Routine ***zonsrt*** will propose a temperature for the next zone, the variable ***TeProp***, if the model is a constant density model. Routine ***tfidle*** sets all ancillary variables related to the temperature, such as alogte.

3.7. Zones and Iterations

iter This is the counter for the current iteration. It is set to one at the start of the first iteration, and is incremented in routine CLOUDY after the last printout, just after the limiting optical depths are updated by calling routine ***update***. The calculation stops when ***iter*** is greater than ***itermx*** *after* the iteration is complete, but before the counter is incremented. ***iter*** is the first variable in common block ***iter***.

npass This is the current iteration number. It is equal to one on the first iteration. The code stops *after* the current iteration if ***npass*** is greater than ***iter***. In common block ***iter***.

ItrDim This is the limit to the total number of iterations that can possibly be performed. It appears in parameter statement, and is used to declare the dimension of the vectors that store iteration information. It currently is set to 20.

itermx This is the limit to the number of iterations to be performed and is set by the user. ***itermx*** is the second variable in common block ***iter***. ***itermx*** is initialized to 0 in the large block data, so that the code normally stops after the first iteration. The value of ***itermx*** can be changed with the **`iterate`** command. ***itermx*** is set equal to the number entered on the command *minus one.* This is so that "**`iterate 1`**" will cause the code to stop after a single iteration (***iter*** is equal to 1 at the end of the first iteration, and the code will only stop if ***iter*** is greater than ***itermx*** after the iteration is complete).

nzone This is the current zone number, equal to one for the first zone. ***nzone*** is set and incremented in routine CLOUDY. It is the first variable in the common block

nzone. ***nzone*** is equal to zero during the search for the initial conditions, on all iterations. After the search has identified a solution the conditions in the first zone are computed with ***nzone*** set to unity.

nend(i) This is the limit to the number of zones in the current (i^{th}) iteration. It is a vector of dimension 20 (as set by the parameter variable ***itrmax***). Individual elements of the vector are set with the **`stop zone`** command. The current iteration stops when ***nzone*** is greater than or equal to ***nend(npass***). It is a variable in common block ***nzone***.

lgOpacOn This logical variable is set false in routine ***zero*** when the code is initialized. It is set true after the opacities are calculated for the first time, in routine ***addopc***.

lgLastIt This logical variable indicates whether (true) or not this is the last iteration. It is controlled by routine ***startr*** and is set true if ***iter*** is greater than ***itermx***, and false otherwise.

3.8. Search phase?

The logic used during the search for the initial conditions at the illuminated face of the cloud is quite different from that used when going from zone to zone across the cloud. Usually no good estimate of the initial conditions exists, but within the cloud conditions do not vary by much from zone to zone. One way to check whether the code has a valid estimate of the physical conditions, or whether the first step in the initial search for parameters is taking place, is to check the status of the variable ***lgSearch***. The initial search is underway if this variable is true. Another is to check whether ***nzone*** is greater than 0.

3.9. Composition variables

3.9.1. Solar default

Default abundances are stored in several arrays in block data ***scalar***. Solar abundances are stored in the array ***SolarSave(nelem***), where ***nelem*** is the atomic number. Other mixtures, such as ISM, HII Region, etc, are also entered in this common block, in other arrays. Each array is dimensioned ***limelm*** (currently 30), the number of elements included in the code.

When the code is initialized the contents of ***SolarSave*** are copied to the array ***solar***, which will contain the initial abundance mix for the current calculation. Gas phase depletion factors, used to modify the final abundance, are stored in the array ***depset(nelem***) and are set to unity in routine ***zero*** when the calculation is initialized. The final contents of ***solar*** will be absolute abundances by number, on a scale with hydrogen at unity.

When an element with atomic number ***nelem*** is turned off, the logical variable ***lgElmtOn(nelem)*** is set to false.

3.9.2. Routine abnset

Routine ***abnset*** is called by routine ***cloudy*** after all commands have been entered. This routine sets the final abundances to be used in the first zone of the calculation. The following variables are used.

dmetal This is the scale factor entered with the **metals** command when a number but no keyword appears on the line. This multiplies the abundances of all elements heavier than helium. It has no effect on hydrogen or helium.

depset If the **metals** command is entered and no numbers appear, but the keyword **deplete** occurs instead, then this array of scale factors is set to the contents of the array ***deplon***. The contents of ***depset*** are set to unity in routine ***zero*** when the calculation is initialized.

deplon This in turn stores the default ISM depletion factors for the ***limelm*** elements. Its contents are set in block data ***scalar***.

scalem This is an array of ***limelm*** scale factors, and is set when the **element scale** command is entered. The contents of this array are set to unity in routine ***zero*** when the calculation is initialized.

lgAbnSolar This logical variable is false if the abundances have been altered, and is true if they are left at the default solar mixture. It is used for sanity checks within the code.

xIonFracs This is a two dimensional vector containing information about gas phase ionic abundances. ***xIonFracs***(nelem,0) is the gas-phase density (cm^{-3}) of all stages of ionization of element ***nelem***. ***xIonFracs***(nelem, n) is the gas-phase density of the nth ionization stage of that element, where the atom is 1.

Routine ***abnset*** first modifies the contents of ***solar*** by the scale factors. The helium abundance is altered by both ***depset*** and ***scalem***, while all heavier elements are modified by these and ***dmetal*** as well. Then ***xIonFracs(nelem,0***) is set to the density (cm^{-3}) of each element, the product ***hden*** and ***solar(nelem***). This is the total abundance of that element, in all stages of ionization and molecular forms.

The initial chemical composition is printed by routine ***PrintElem***, which is called by ***abnset***.

3.10. Covering factors

Two covering factors enter into the calculations. These are referred to as the geometric covering factor, and the radiative transfer covering factor.

3.10.1. Geometric covering factor

This covering factor linearly affects the luminosity of emission lines. The nebula intercepts a fraction $\Omega_{geo}/4\pi$ of the luminosity radiated by the central object. Within the code the geometric covering factor is referred to by the variable ***covgeo***.

The code actually works in units of intensity radiated by a unit area of illuminated face of the cloud to avoid exponential range problems with IEEE

machines. If the predicted intensity of a line (erg s^{-1} cm^{-2}) is given by I then the line luminosity will be

$$L = 4\pi r_{inner}^2 \frac{\Omega_{geo}}{4\pi} I \qquad (406)$$

where r_{inner} is the inner radius.

The default value of the geometric covering factor is unity, and it can be changed with the **`covering factor`** and **`sphere`** commands.

3.10.2. Radiative transfer covering factor

The radiative transfer covering factor has only second order effects on the intensity of emission lines. This is the covering factor which takes into account interactions with diffuse fields produced on the symmetric far side of the nebula. Within the code it is referred to by the variable ***covrt***.

The default value of the radiative transfer covering factor is zero, appropriate for an open geometry. For a closed geometry it is set to unity. The radiative transfer covering factor only affects the model through the diffuse fields. For a closed geometry all radiation is included in the outward beam, and for an open geometry only half. This covering factor has no effects on the calculations, other than the amount of diffuse fields transferred outward. Physically for an open geometry the fraction of radiation escaping in the inward direction is then lost to the system. In an open geometry the nebula is symmetric, and escaping radiation is exactly matched by radiation impinging from the far side of the geometry.

4. PROBLEMS

4.1. Overview

This section describes some of the errors that can cause CLOUDY to stop. Floating point errors should never occur. Several other internal errors, which the code is designed to catch and then complain about, can also occur. Finally, it is possible that the code will stop because of thermal stability problems.

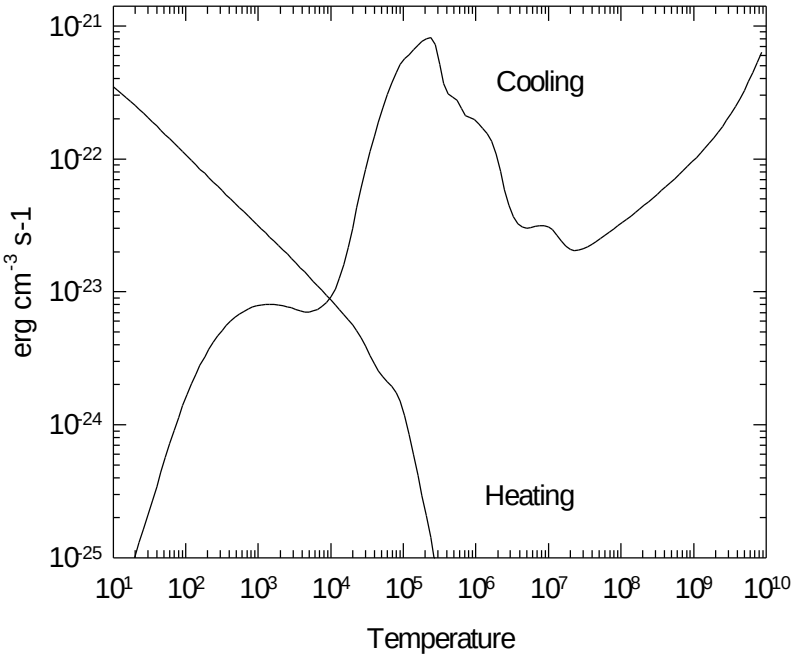


Figure 8 A typical cooling function for low density photoionized gas. The cooling and heating rates (erg cm^{-3} s^{-1}) are shown. cooling

4.2. Thermal stability and temperature convergence

4.2.1. Types of thermal maps

Three types of thermal maps, showing the heating or cooling of gas as a function of temperature, can be produced by CLOUDY. Each is the answer to a different question.

Figure 8 shows the heating and cooling rates as a function of temperature for a photoionized gas in which the electron temperature was varied. This figure was produced by running the test case **`map.in`** shown on page 486 below. The gas has constant density and flux of ionizing photons, so only one temperature is meaningful. The map.in file uses the **`punch map`** command to vary the temperature and determine heating and cooling rates. This is exactly what the code does to determine the equilibrium

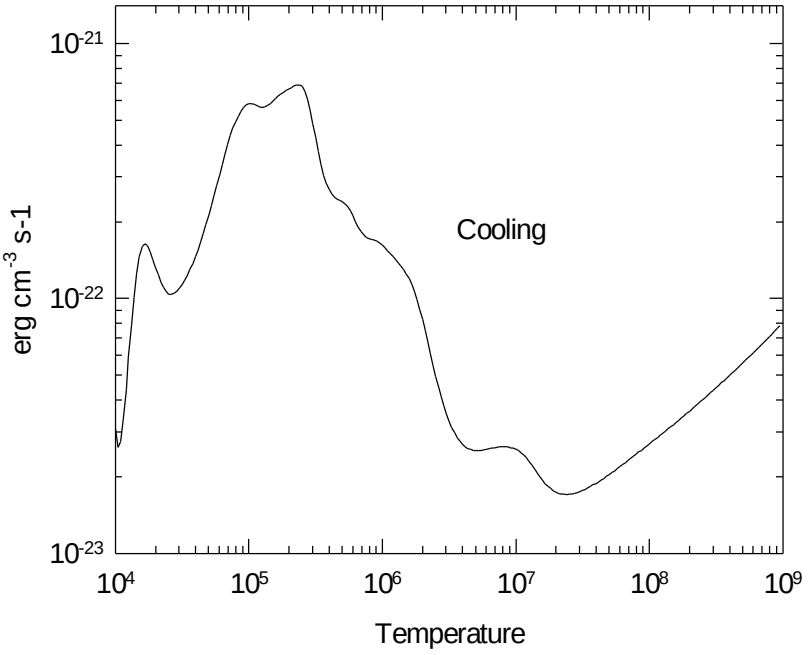


Figure 7 A typical cooling function for low density collisionally ionized gas. coolcurve

temperature, so this plot can be useful to find out why the code ran into problems. This is why the command was introduced.

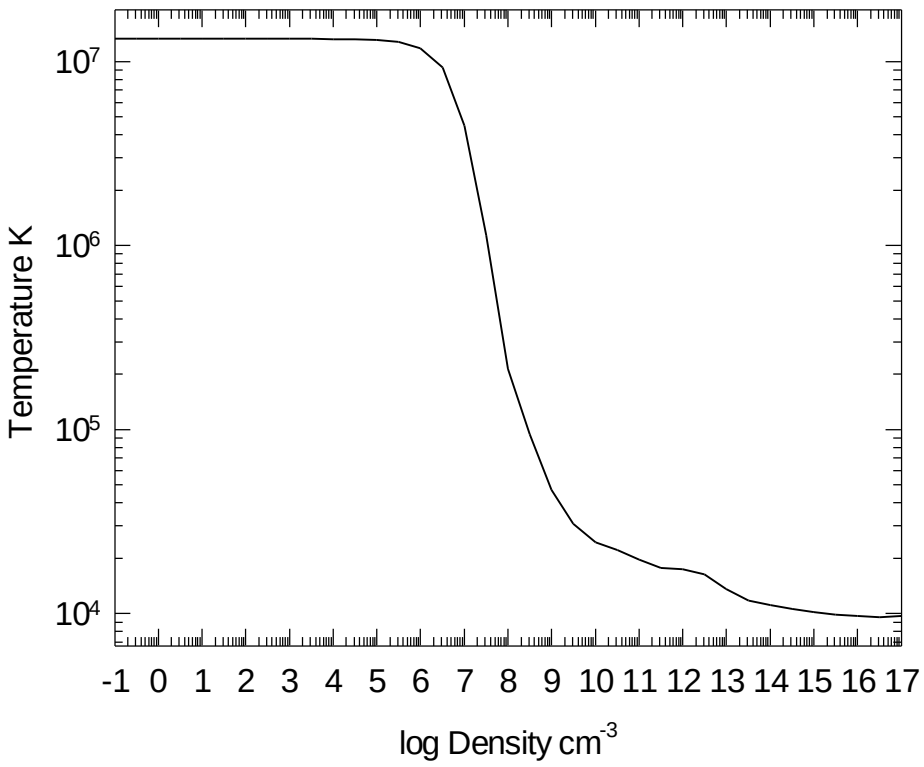


Figure 9 Equilibrium temperate as a function of density. kmtplot

Gas in collisional equilibrium has a well defined cooling rate that is only a function of temperature. The sample program **coolcurve.for** (see page 467 below) does such a calculation, and Figure 7 shows it. Here the electron temperature is assumed to be set by some physics external to the problem. Each temperature, and the entire ionization solution, is valid, under this assumption.

The third map is the type of thermal stability map shown by Krolik, McKee, and Tarter (1981) and plotted in Figure 9. The program that generated these results is shown on page 481 and the source for this test is given in the file **kmt.for**. Here the equilibrium temperature is determined self-consistently for gas over a wide range of densities, but a single flux of ionizing photons (or equivalently, distance from the central object).

4.2.2. No Temperature Convergence

A temperature failure occurs when the heating-cooling balance is not within a certain tolerance, set by the **tolerance** command, after 20 tries. Normally CLOUDY will punt after an excessive number of temperature failures occur. The limit to the number of failures is reset with the **failures** command, which sets the variable ***limfal***. The default value is 20. (When CLOUDY stops because of excessive failures it first produces a map of heating-cooling space to give an indication of where the equilibrium temperature should have been.)

Temperature failures most often occur for temperatures in the range 1 to 4×10^3 K, and 10^5 to 10^6 K. These are ranges where the cooling function permits more that one thermal solution (see, for example, Williams 1967). Figure 8 shows a typical cooling function for gas in photoionization equilibrium.

A peak is reached at a temperature near 10^3 K. This can occur when the fine-structure lines are major coolants. At lower temperatures their cooling rate goes up

exponentially (as expected), until roughly 10^3 K, when their Boltzmann factors are near unity. Above this temperature their cooling rate is nearly proportional to the Coulomb focusing factor $T^{-1/2}$, and the cooling *decreases* until the temperature is high enough for optical forbidden lines to become important (at roughly 4000 K). A similar phenomenon occurs near the ~ 10^5 to 10^6 K peak in the cooling function.

When failures occur because more than one temperature solution is possible, the reported failures are a physical (not numerical) problem. CLOUDY will try to deal with this problem by forcing the temperature to values below the peak in the cooling function. Increasing the number of allowed failures (with the **failures** command) to prevent the code from stopping prematurely is permissible as long as the global energy balance is preserved. A warning will be issued at the end of the calculation if the heating-cooling balance is not preserved.

4.2.3. Thermal Stability

The thermal solution may be unstable when the temperature derivative of the net cooling function (cooling minus heating) is negative (Field 1965). Possibly unstable solutions are indicated by a "u" just before the equilibrium temperature in the zone printout. (The temperature derivative is for isochoric, not isobaric, conditions.) Comments are printed at the end of the calculation if possibly unstable thermal solutions are present in the calculation.

4.2.4. Temperature Jumps

The code will generate a caution or comment if the electron temperature changes discontinuously from one zone to the next. This can be caused by a real physical change of state of the gas such as those that occur near the peaks in the cooling curve. There is no obvious solution to this type of jump since it is physical. Jumps can also occur if the grid zoning becomes too coarse for changes in the physical conditions, although the code should be protected against this.

4.2.5. Map Output

If an excessive number of temperature failures occur (the default limit is 20) then the program stops and produces a map of the heating and cooling as a function of temperature for the last computed zone. The limit to the number of failures allowed before the code punts is reset with the **failures** command. The map is described here. The start of the output from the test case **map** is shown below.

```
90.02x map of heating vs cooling
 te, heating, cooling.
 Cloudy punts, Te= 9.254E+03 HTOT= 9.123E-24 CTOT= 9.118E-24 nzone=   1
 COOLNG array is
     O  4    25  0.340 O  3  5007  0.182 O  3    88  0.075 H FB     0  0.057 S  4    10  0.048 O  3    51  0.042 S  3  9532  0.035
     H ff     0  0.022 S  3    33  0.020 Ne 3    15  0.019 Hefb     0  0.015 N  3    57  0.015 Ne 3  3869  0.013 S  3    18  0.013
     Ne 5    24  0.010 Ne 5    14  0.009 C  3  1910  0.008 Heff     0  0.007 Si 2    34  0.006 Fe 5  3892  0.006 O  2  3727  0.005
 Line heating array follows
    Te     Heat------------------> Cool---------------------->    dH/dT    dC/DT       Ne         NH      HII       Helium
1.0000E+01 3.4774E-22   1   1 0.636 4.6095E-26 H FB    0A 0.723 -8.19E-24  1.56E-27 9.1178E-01 1.0000E+00 -0.07 -0.40 -0.24 -1.75
1.0209E+01 3.4490E-22   1   1 0.635 4.6814E-26 H FB    0A 0.720 -7.98E-24  1.65E-27 9.1353E-01 1.0000E+00 -0.07 -0.40 -0.23 -1.73
1.0423E+01 3.4233E-22   1   1 0.635 4.7510E-26 H FB    0A 0.717 -7.74E-24  1.74E-27 9.1491E-01 1.0000E+00 -0.07 -0.41 -0.23 -1.73
```

The output begins with a listing of the strongest coolants. Then the program steps through increasing temperatures and prints the heating, cooling, and ionization of the gas. From this information it should be possible to determine the temperature where the equilibrium thermal solution should have been. Each solution is completely self-consistent (except that heating and cooling do not balance). Both the local attenuated radiation field and collisional ionization contribute to the ionization

balance at each temperature. All processes contribute to the thermal balance, including collisional ionization. The map is at constant density.

The first column gives the temperature. Columns 2 and 6 give the volume heating and cooling. Both have units erg s^{-1} cm^{-3}. Columns 3 and 4 constitute a pointer to the main heating source. Columns 7 and 8 give the label and wavelength of the strongest coolant. Columns 5 and 9 give the fraction of the total heating or cooling due to these agents. Columns 10 and 11 give the heating and cooling derivatives. Columns 12 and 13 give the electron and hydrogen densities (cm^{-3}) and the remaining columns give the logs of the hydrogen and helium ionization fractions. The location of the probable thermal solution is indicated by a comment surrounded by dashed lines.

4.3. Floating Point Errors

CLOUDY has been tested on machines ranging from 80486-based PCs to Cray's. *Floating point errors do not occur.* The logic within the code is designed to identify problems, and complain, but not fail. The logic is only as good as the tests they were designed to pass. It is inevitable that circumstances will occur for which the logic now in the code is not sufficient. It is possible that the code will fail when these circumstances occur. I would be grateful for reports of any such failures, since they inevitably identify shortcomings in the code, and lead to its improvement. My internet address is gary@cloud9.pa.uky.edu.

4.4. Optical depth convergence problems

The code generally will not converge if it has not done so within ten or so iterations. The most common reason for convergence problems is that the specified column density or thickness is very near a prominent ionization front. In this case very small changes in the physical conditions results in large changes in the optical depths. The code will not have convergence problems if an optical depth is used as a stopping criterion instead.

4.5. Negative Populations of H, He

It is possible that the code will stop because negative level populations were predicted for atoms or ions of hydrogen and helium. This is not supposed to occur, but sometimes happens because of numerical instabilities in the matrix inversion routine. Please send me the input stream and version of CLOUDY.

4.6. I can't fix it if I don't know its broken.

Machines are growing faster much more rapidly than people are getting smarter. Reliability in the face of complexity is the major challenge to the development of any large-scale computer code. There can be little doubt that CLOUDY contains bugs.

The code is well tested in many simple limits, and behaves in the correct manner. Simulations of H II regions, planetary nebulae, and other simple objects, are in good agreement with predictions of other photoionization codes (Ferland et al. 1995).

Bugs can be discovered by strange behavior in situations where the code has not been well-tested. The discovery of the existence of problems is itself a major challenge. If problems arise or the code crashes then it is likely that a problem has been isolated. I would appreciate learning about such problems since they identify shortcomings which usually lead to improvements in the code (or the documentation). My internet address is gary@cloud9.pa.uky.edu.

5. REVISIONS TO CLOUDY

5.1. Overview

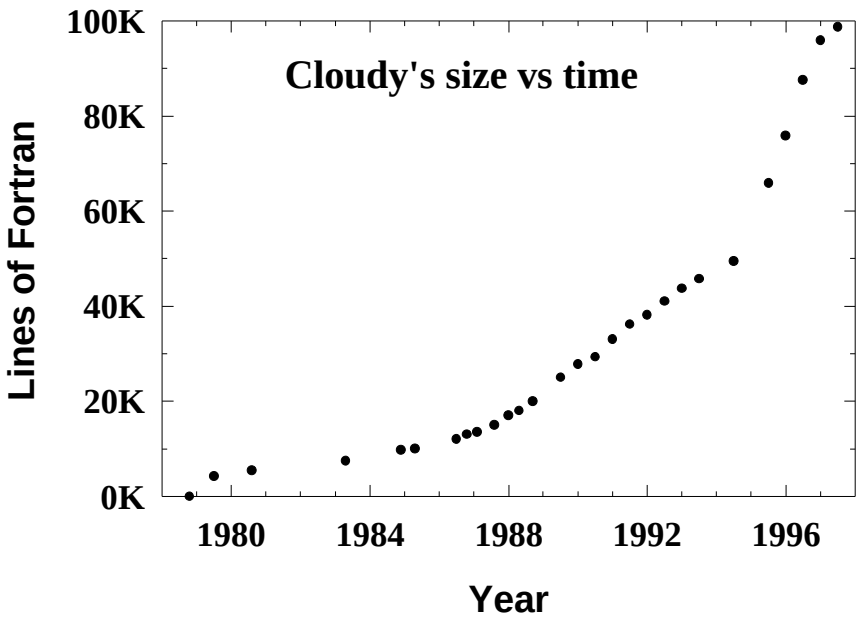


Figure 10 The size of the code, indicated by the number of lines of executable Fortran, as a function of time. size

This section outlines some of the major versions of CLOUDY, and gives an indication of the direction development will take in the next few years. Its development began in August of 1978, at the Institute of Astronomy, Cambridge, and has been continued at The University of Kentucky, The Ohio State University, and during extended visits to the Joint Institute for Laboratory Astrophysics, the Royal Greenwich Observatory, and Cerro Tolodo Interamerican Observatory. Figure 10 shows the evolution of the code, as indicated by its size as a function of time[2].

5.2. CLOUDY and Moore's Law

Moore's Law is due to Gordon Moore, one of the founders of Intel Corporation. He observed that modern CPU's become about twice as powerful every 18 months. This trend has held true for the past twenty years, and shows no sign of failing.

By this standard the growth of CLOUDY has been conservative, in that it is growing slower on the same timescale. As an example, the Meudon 1985 Meeting planetary nebula test (page 493) has always taken about one minute to compute.

5.3. Major Past Versions

- 67 August 1987. Hydrogen atom goes to LTE in limit of large electron densities. Many small bugs uncovered as result of careful comparison with Netzer's ION.

- 68 Cambridge, Fall 1987. Development work in progress.

[2] Before mid-1995 the size was the total number of lines in the distributed source. After 1995 the size only includes the number of lines excluding block datas. The number of lines of block data became very large, thanks to Dima and Katya Verner, so that the total size is roughly 2.5 times the indicated size.

- 69 December 1987. Hydrogen atom goes to LTE in limit of large photon densities. Ferland and Rees (1988).
- 70 September 1988. He II Lyα transfer improved. Improved form of escape probabilities with explicit damping constants. H^- and improved free-free heating. Many high excitation metal lines transferred. (Rees, Netzer, and Ferland 1989; Ferland and Persson 1989).
- 71 December 1988. Photon array rewritten, now Compton exchange problem is exact for black bodies with temperatures between 2.7 K and 10^{10} K. He II Lyα radiation pressure included.
- 72 January 1989. Static version, minor bug fixes.
- 73 1989. Major rewrite of helium treatment. Dust changed to two populations, scattering and absorption included. Default radius and thickness increased by ten orders of magnitude. He^+ goes to LTE. Development work on getting helium to go to LTE. IR power law for default AGN continuum now $\nu^{+2.5}$ below 100 micron break. No ***anumm*** array, all continuum one array. **`table star`**. ***coolr*** broken up. Helium to LTE for high electron density.
- 74 1990 January. Hydrogen double precision, many bug fixes. 10 tables among the continua. Kurucz (1989) atmospheres. Improved dust treatment, including photoionization and charge.
- 75 1990, JILA visit. Cosmic abundances changed to Grevesse and Anders. Major bug in constant pressure for HII regions, PNs, etc; did not affect BLR. Calculation of νf_ν (Hβ) was incorrect. Molecules at low temperatures. HDEN now $n(H^o) + n(H^+) + n(H^-) + 2n(H_2) + 2n(H_2^+)$. Improved Rayleigh scattering treatment. Dielectronic recombination for sulfur (guess). Many changes in dust; Orion paper (Baldwin et al. 1990). Subordinate lines changed to Hummer's K2 function.
- 76 1990, static version from end of JILA visit.
- 77 through Nov. 1990. **`optimize`** option added using Bob Carswell's code. Gaunt factor for brems input spectrum. Reflected continuum predicted. Frequency partition adjusted. X-Ray optical depth now at 0.5 keV. Read in table of points from previous calculation. ***opsav*** deleted, now single pointer ***opsv*** for all opacities. Numerical array to 100 MeV. Hummer Lyα escape destruction prob. ***tautot*** arrays now both in and outward directions. Bound Compton included for all ionization levels. Mean ionization arrays rewritten to make sense. Bug in wind velocity fixed, result now exact. C, O outward diffuse fields changed to OTS.
- 78 Through May 91. Continuum escape probability formal H-only opacity. H recombination, cooling over wide range of temp. X-Ray optical depth back to 1 keV. **`abundances no dust`** no longer changes abundances of depleted elements. OI-Lyβ treatment is now six-level atom. Default table AGN changed. Continuum normalization rewritten. Milne relation for diffuse fields of H, all He. Fe Kα divided into hot and cold. Beams paper (Ferland,

Peterson, Horne, et al. 1992) Many high ionization lines included as OTS and outward ionization sources.

- 79 Summer 91. H molecules completed, C, N, O molecules included. Continuum binning changed for Ca, Fe L-shell ionization potentials. Extensive testing.
- 80 July 91, static version .05. Version 80.06 fixed many small problems discovered by several people. New collision strength for [NeV] put in. This static version ended with 80.09, in January of 1993.
- 81 Late 1991. New collision strengths for [NeV] IR lines (10x larger). Also for [CII], [NIII], and [OIV]. Some are 2x larger. Opacity arrays totally rewritten with eye to Opacity Project data. NIII paper (Ferland 1992).
- 82 Early 1992. X-Ray opacity arrays rewritten. Improved pressure convergence. HJBAR now function. Error in cooling due to collisional ionization of H, He. Sodium and nickel added. /TAU/ array broken up. All heavy element opacities converted to table look-up. Revised collision strengths for NIII lines, improved treatment of atom. NLR abundances deleted, ISM put in their place. Summary comments now driven by subroutine. Cap on 911 OTS field. Note on [FeXI] maser (Ferland 1993). Luminosity command separated into luminosity and intensity commands.
- 83 Autumn 1992, Cambridge and CTIO visits. Hydrogen molecule network completed, Ferland, Fabian, and Johnstone (1993). Collision strengths for fine structure lines changed to Blum and Pradhan 1992, Hollenbach and McKee 89; these changed temperatures for cold ISM by factors of 2. Heavy element molecule network as in Hollenbach and McKee 1989. Code works in fully molecular limit. Kevin Volk's stars (Atlas 91, and Werner models). More accurate treatment of secondary ionization after Voit visit. [OI] lines each include escape prob. Opacities, destruction probabilities, evaluated within all loops, code far more stable, but roughly three times slower. Transferred HeI 2.06 line correctly, after Shields papers.
- 84 1993 Feb 13, Static version following CTIO visit.
- 85 1994, Revisions following Lexington meeting. Outward only now default continuum transport.
- 86 1995, All of first thirty elements are now in code. Photoionization database changed to Dima Verner's ***phfit***.
- 87 1995 summer, Map now converges electron densities. Negative populations of OI and FeII atoms solved. Dima's 6k lines included in cooling and radiative acceleration. Total rewrite of ***nextdr*** logic. Ionization predictor corrector logic completely rewritten. Dima's ***phfit*** now fitted to all Opacity Project data.
- 88 1995 Fall-winter. Kirk's extensive grids of BLR models run. March 1996 visit to Tel Aviv to compare results with Hagai. Kirk carefully went over atomic data base. Iron recombination changed to Arnaud and Raymond (1992). Default iron abundance down 33%. Entire line data base revised.

- 89 1995 winter-spring. These were mainly beta versions with no major changes, but many fixes to problems.
- 90 1996 June 17, static version, extensive year of debugging. Ferland et al. (C90) paper.

5.4. Version 90 versus 84

5.4.1. Commands

The **abundances** command now needs 29 numbers by default. A new command "**init**" allows a commonly used set of commands to be saved as a single file and used by a variety of scripts.

5.4.2. Continuum Transport

Versions 86 and before used a modified version of on-the-spot approximation (OTS) for the Lyman continua of hydrogen and helium. This method was numerically stable and gave results in excellent agreement with Van Blerkom and Hummer (1967). This has been changed to outward-only to obtain better agreement with predictions of Pat Harrington's and Bob Rubin's codes (Ferland et al. 1995). The OTS code is still in place and will be used if the **diffuse ots** command is entered, but outward-only is the default. The two methods result in temperatures at the illuminated face which can differ by as much as several thousand degrees, but the resulting spectra are surprisingly similar.

5.4.3. Hydrogen

The model hydrogen atom has been generalized to an arbitrary multi-level atom (Ferguson and Ferland 1996). The **hydrogen levels** command is used to specify the number of levels to be used. The collision strengths have been changed to Vriens and Smeets (1980) for levels higher than 3, and Callaway (1994) for collisions with 1,2 and 3.

Predicted infrared line intensities are now correct for all densities and temperatures greater than 10^3 K. Versions before 89 used a well l-mixed hydrogen atom, and its predictions were not correct for some infrared lines at low densities.

The routine that computes the free-free gaunt factors has been extended to include the full range the code can handle.

5.4.4. The helium ion

The helium ionization balance at low photon and particle densities, and at high particle densities, has always been exact, and this continues to be the case. There was a problem in the helium ion for high radiation densities, in versions 87 and before. The code used three pseudo levels to represent the levels between 7 and 1000, for H, He, and He^+. This seemed to work well for the atoms for the cases of high densities, but testing has shown that it did not represent the physics of the high radiation density limit well. The problem is that the pseudo-levels had very large statistical weights, they represented line energies in the far infrared, and had A's appropriate for lower levels. As a result they had very large induced rates when the photon occupation numbers were large, and this affected populations of lower

levels. As a result the atom became too ionized - as much as a factor of two for He^+. The following test illustrates this problem:

```
title helium ionization in high photon density limit
print departure coef
set dr 0
stop zone 1
constant temper 4
hden  11.000
phi(h)  20.750 range 1
stop thickness 11.7
table agn
```

In versions 88 and later no pseudo levels are used for any hydrogen or helium atom or ion.

5.4.5. Heavy elements

The atomic data base, the organization of aspects of the code dealing with storing heavy element ionization, and all the associated routines, have been totally rewritten. The lightest 30 elements are now included. Photoionization data are from Verner et al. (1996), recombination data partially from Verner and Ferland (1996), and roughly 10^4 lines of the heavy elements have been added (Verner, Verner, and Ferland 1996).

The number of resonance lines has increased by more than an order of magnitude. All resonance lines listed by Verner, Verner, and Ferland (1996) are included. As a result of these many additional lines the cooling function tends to be larger and smoother.

All lines are now fully transferred, and include pumping by the attenuated incident continuum as a general excitation mechanism. Pumping can be a significant contributor to the formation of weak high excitation lines.

The default solar mixture has been changed to Grevesse and Noel (1993). The biggest change is in the iron abundance. Previous versions had used a higher photospheric abundance. The current version is the 1993 suggested meteoritic abundance.

5.4.6. Free-free, line heating and cooling

These are counted in a different but equivalent manner. Now the *difference* between cooling and heating is used, since this is more numerically stable at high radiation densities. This difference has no physical affect on the predictions, but the printed contributors to the total heating and cooling do appear different.

5.4.7. Excited state photoionization cross sections

OP data are now used. For the excited state of Mg^+ this is nearly ten times smaller than old screened hydrogenic values. This affects the intensity of Mg II λ2798 in some BLR calculations.

5.4.8. The O^+ photoionization cross section

The Reilman and Manson photoionization cross sections, used before version 87, show a jump in the photoionization cross section at the 2s - 2p edge, and low values above that threshold extending up to the valence electron threshold. Opacity Project cross sections are used in the current version of the code, and these do not show the 2s edge (the OP calculations find that the 2s and 2p electrons are highly correlated). The cross section remains large up to the valence threshold. The difference approaches a factor of two, and this affects high ionization parameter clouds since O^+ is the dominant opacity for some energies.

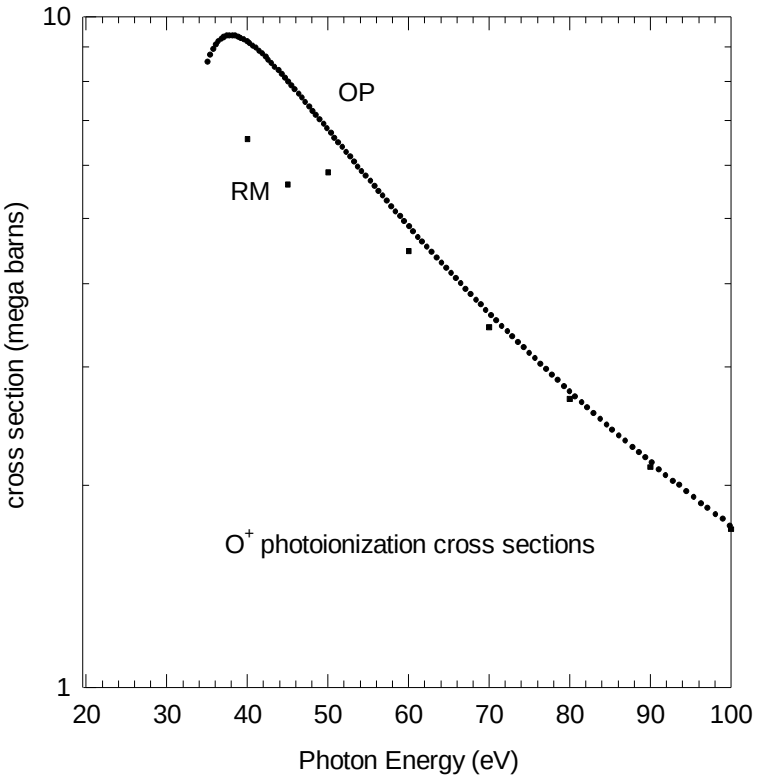


Figure 11 This figure shows the O^+ photoionization cross sections now used, compared to the Reilman and Manson values, used in version 84 and before. o2photo

Verner, et al. (1996) comment on all other cases where the photoionization cross sections have changed. There are generally atoms and first ions where Opacity Project data are now available.

5.5. Version 84 versus 80

5.5.1. Commands

In previous versions of the code both luminosity (quantity radiated into 4π sr) and intensity (a surface flux) were specified with the same command. The code decided which was intended by checking the resulting ionization parameter. This method never failed to the best of my knowledge, but, as the code grows more capable of considering ever more extreme cases, there might eventually come a time when it made the wrong decision. All luminosity-intensity commands have now been split up. For instance, the Q(H) command is now two commands, Q(H) (for number of photons radiated into 4π sr) and ϕ(H), for the surface flux. These commands are all discussed in Part I of HAZY.

5.5.2. Mg II l2798

The biggest difference between the two versions is in the predicted intensity of Mg II λ2798. The intensity of this line is now a factor of two stronger in many models. The new version uses L-shell photoabsorption cross sections from Reilman and Manson (1979), and the older version used inner shell cross sections extrapolated from the table in Weisheit (1974). The cross sections differ by a factor of nearly 3, in the sense that Mg now tends to be more neutral, and Mg II stronger. As a result of the increased cooling by λ2798 other lines formed in the same region tend to be weaker.

5.5.3. General Results

The following may affect certain specific models, but did not result in changes in any of the "standard" test cases.

The treatment of line-continuum fluorescence in the optically thin limit has been much improved, following Ferland (1992). This can affect hydrogen line emission in clouds that are optically thin in the Lyman continuum.

The treatment of molecules has been vastly improved. The code now goes to the fully molecular (H_2 and CO) limit, and reproduces the Tielens and Hollenbach (1985a, b) PDR results for Orion.

The elements Na and Ni have been added.

5.6. Future Improvements

- Helium transition probabilities as a function of temperature and density.
- Molecule network goes to LTE for all temperatures.
- Transfer Lyα correctly (incomplete redistribution).
- Transfer other hydrogen lines correctly (complete redistribution).
- Heavy elements go to LTE via pseudo states.
- Something must be done about low-temperature dielectronic recombination.

5.7. Known Modes for CLOUDY 90

- Time dependent model turned off.
- HeII 10-level atom line predictions do not agree with case B at low density. (*l*-mixing is not a good assumption for low densities.)
- Molecular abundances can go negative for very large CO fractions.
- HeII line radiation pressure not reliable.
- 2s level of hydrogen not exactly unity in radiation-dominated LTE limit. Induced two-photon emission.

5.8. Making a Revision

5.8.1. The code

- Compile code with array bounds checking. Run all test cases.
- Confirm that **`punch transmitted`** and **`table read`** work correctly together.
- Confirm that **`compile stars`** still works.

- Run all test cases on Alpha, Cray, HP, Sparc, SGI, and PC. Confirm no differences. The Microsoft Power Station Fortran is important since it catches many problems tolerated by Unix compilers.
- Confirm ***fabden*** stops immediately.
- Confirm all Press et al. code removed. There are three sets of source, normally called **press.for** (this is actually used), **press.source,** and **press.none**. The second file contains their source and must not be redistributed. The third file contains a series of empty routines so that the loader will think the codes are present.
- Verify **AAAA.for** has correct version.
- Summarize changes in **c**90**rev.htm** in the web site. Use comments in **version.for**. This is the public primary documentation for changes to the code.
- Go to the **include** directory, and execute the shell script **do.all**. This will create a file called **cloudy.f**, which should be renamed to something like **c9034.for**, and copied to the **ftp/source** subdirectory. Update the **readme.htm** file there. Use the script **maketar** in to source subdirectory to tar the files into **source.tar**. The file list in **maketar** will need to be updated, and the sample test case **parispn.in** computed. Compress the resulting output file **source.tar** and copy it to the ftp site. The script **cp.ftp** will copy it to the correct directory. Create a new subdirectory with the version number of the code, and move **source.tar**, the test case output, and the Fortran file there.
- Create the test case files for each of the platforms. Each lives in a separate subdirectory off the hazy directory, and contains a script called **maketar**. This will create a compressed tar file with the name of the platform included. Place this in the ftp site with the script **cp.ftp**.
- Tar all input files into single file with name **tests.tar**. The command is **tar -cvf tests.tar *.in *.for**. Compress and copy to **ftp**.
- The ftp account is /home31/sys/ftp/gary/c90 on **gradj**. This should be mirrored in the account **ftp** in Gary's home directory.

5.8.2. Printing Hazy

- ***Compile the line list from `lines.for` and bring these into HAZY.*** To do this, go to the **programs** subdirectory off the **hazy** main directory, and run **do.all** to create formatted output file call **lines.out** for HAZY. Edit the links in HAZY **lines.doc** to update this.
- ***Create the list of transferred lines.*** Compile the line data list for the transferred lines by running the input file **punchline.in** in the CLOUDY main directory. This will create a punch file named **LineData.txt**. This is the file

that will be linked into HAZY. Edit the links in HAZY's file **lines.doc** to update the list of transferred lines.

- ***Update the list of subroutine names.*** Do this by running **do.all** in the **include** directory. The list of routine names in is the file **routines.txt**, and this is automatically generated by **do.all**. This was probably done to update the source, as described in the previous section. Routine **makesort** sorts the file **fort.92** into **routines.txt** that is used by HAZY. Edit links in HAZY **routines.doc** to update this.
- ***Update all comparison tables.*** These are in the section starting on page 440. Go over these to confirm that line predictions are OK.
- ***Update all test case input scripts.*** These are listed in the section beginning on page 457 of this document. Edit the links to manually update them.
- ***Confirm that all cross-linked variables are ok***. Change labels in HAZY **headinfo.doc**.
- ***Summarize changes to the code***. These are listed in **version.for** and should go into the past major revisions in this section of HAZY.

6. COMPARISON CALCULATIONS

6.1. Overview

This section presents comparisons between the current predictions of the code, and results from other independent calculations. The "other" calculations are from the compendium resulting from the Lexington meeting on model nebulae (Ferland et al. 1995).

The scatter among the calculations, as well as the changes that have occurred in the predictions made by CLOUDY, are in some sense an indication of the stability and reliability of these types of extreme non-LTE calculations. The largest discrepancies between current predictions made by CLOUDY and the other models from the Meudon meeting (which were computed in 1985) are due to changes which have occurred in the atomic data base between 1985 and the present. In general, the strongest lines are in very good agreement (as they must because of energy conservation) while weak lines (which are very sensitive to changes in the computed temperature and ionization structure) scatter by nearly a factor of two.

6.2. Cool HII Region

This is an HII region ionized by a very cool star. It is one of the Lexington Meeting test cases and is computed with the input script **`coolhii.in`** in the code's test suite. This is the simplest model since helium is predominantly neutral. The entry ***L(total)*** comes from the "Stoy" printed entry.

Table 1 HII Region Ionized by Cool Star

		Mean	Ferland	Harrington	Netzer	Pequignot	Rubin
L(Hβ)	E36	4.90	4.98	4.93	4.85	4.83	4.93
[NII]	6584+	0.87	0.91	0.82	0.97	0.82	0.84
[OII]	3727+	1.21	1.16	1.22	1.32	1.14	1.21
[NeII]	12.8 μ	0.30	0.35	0.29	0.29	0.29	0.29
[SII]	6720+	0.57	0.64	0.55	0.61	0.52	0.52
[SIII]	18.7 μ	0.31	0.27	0.36	0.17	0.37	0.37
[SIII]	34 μ	0.51	0.47	0.60	0.27	0.61	0.62
[SIII]	9532+	0.57	0.48	0.55	0.64	0.60	0.56
L(total)	E36	21.3	21.3	21.7	20.7	21.0	21.8
T(in)		6860	6952	6749	6980	6870	6747
T(H+)		6767	6740	6742	6950	6660	6742
<He+>/<H+>		0.047	0.041	0.044	0.068	0.048	0.034
R(out)	E18	8.96	8.93	8.94	9.00	8.93	9.00

Table 2 Cool HII Region vs Cloudy

		Mean	STD	90.00	90.03
L(Hβ)	E36	4.90	0.06	4.98	5.01
[NII]	6584+	0.87	0.07	0.91	0.87
[OII]	3727+	1.21	0.07	1.15	1.08
[NeII]	12.8 μ	0.30	0.03	0.26	0.26
[SII]	6720+	0.57	0.05	0.49	0.47
[SIII]	18.7 μ	0.31	0.09	0.36	0.43
[SIII]	34 μ	0.51	0.15	0.59	0.61
[SIII]	9532+	0.57	0.06	0.56	0.59
L(total)	E36	21.29	0.45	21.5	21.7
T(in)		6860	110	7353	7274
T(H+)		6767	108	6699	6623
<He+>/<H+>		0.047	0.01	0.049	0.049
R(out)	E18	8.96	0.04	8.92	8.91

6.3. Paris HII Region

This compares current predictions of the code with those of other participants at the Meudon meeting on photoionization calculations for the case of a simple spherical HII region. The input used to generate this model HII region is shown in the sample input section with the name **`parishii.in`** on page 492.

Table 3 Paris Meeting HII Region

		Meu	Lex	Ferland	Harrington	Netzer	Pequignot	Rubin
L(Hβ)	E37	2.06	2.04	2.06	2.04	2.02	2.02	2.05
HeI	5876	0.116	0.111	0.109	0.119	0.101	0.116	
CII	2326+	0.17	0.17	0.19	0.17	0.16	0.14	0.18
CIII]	1909+	0.051	0.07	0.059	0.059	0.078	0.065	0.076
[NII]	122 μ		0.031	0.033			0.036	0.031
[NII]	6584+	0.73	0.80	0.88	0.74	0.87	0.78	0.73
[NIII]	57 μ	0.30	0.28	0.27	0.29	0.26	0.30	0.30
[OII]	3727+	2.01	2.20	2.19	2.14	2.3	2.11	2.26
[OIII]	51.8 μ	1.10	1.06	1.04	1.11	0.99	1.08	1.08
[OIII]	88.4 μ	1.20	1.20	1.07	1.28	1.16	1.25	1.26
[OIII]	5007+	2.03	2.09	1.93	1.96	2.29	2.17	2.10
[NeII]	12.8 μ	0.21	0.21	0.23	0.19	0.22	0.20	0.20
[NeIII]	15.5 μ	0.44	0.41	0.43	0.43	0.37	0.42	0.42
[NeIII]	3869+	0.096	0.091	0.103	0.086	0.100	0.079	0.087
[SII]	6720+	0.14	0.18	0.23	0.16	0.22	0.17	0.13
[SIII]	18.7 μ	0.55	0.53	0.48	0.56	0.5	0.55	0.58
[SIII]	34 μ	0.93	0.87	0.82	0.89	0.81	0.88	0.94
[SIII]	9532+	1.25	1.31	1.27	1.23	1.48	1.27	1.30
[SIV]	10.5 μ	0.39	0.38	0.37	0.42	0.36	0.41	0.33
Sum	Sum	11.59	11.87	11.58	11.70	12.17	11.90	12.01
L(total)	E37	24.1	24.4	24.1	24.1	24.8	24.3	24.6
T(in)		7378	7655	7815	7741	7670	7650	7399
T(H+)		7992	8052	8064	8047	8000	8060	8087
He+>/<H+>			0.76	0.71	0.77	0.76	0.75	0.83
R(out)	E18	1.45	1.46	1.46	1.46	1.47	1.46	1.46

Table 4 Paris HII Region vs Cloudy

		Mean	STD	76.03	80.06	84.15	90	90.03
L(Hβ)	E37	2.04	0.02	2.06	2.04	2.01	2.02	2.02
HeI	5876	0.11	0.01	0.12	0.12	0.12	0.11	0.11
CII]	2326+	0.17	0.02	0.24	0.17	0.17	0.16	0.16
CIII]	1909+	0.07	0.01	0.06	0.11	0.09	0.072	0.073
[NII]	122 μ	0.03	0.00			0.03	0.029	0.029
[NII]	6584+	0.80	0.07	0.77	0.80	0.77	0.82	0.799
[NIII]	57 μ	0.28	0.02			0.29	0.29	0.29
[OII]	3727+	2.20	0.08	2.29	2.43	2.31	2.08	2.05
[OIII]	51.8 μ	1.06	0.05	1.15	1.09	1.11	1.26	1.24
[OIII]	88.4 μ	1.20	0.09			1.06	1.15	1.13
[OIII]	5007+	2.09	0.15	2.44	2.16	2.00	2.44	2.45
[NeII]	12.8 μ	0.21	0.02	0.22	0.23	0.23	0.18	0.18
[NeIII]	15.5 μ	0.41	0.02	0.47	0.45	0.45	0.29	0.29
[NeIII]	3869+	0.09	0.01	0.13	0.12	0.11	0.089	0.093
[SII]	6720+	0.18	0.04	0.27	0.29	0..129	0.13	0.13
[SIII]	18.7 μ	0.53	0.04	0.54	0.53	0.52	0.51	0.61
[SIII]	34 μ	0.87	0.05			0.88	0.82	0.86
[SIII]	9532+	1.31	0.10	1.39	1.39	1.40	1.2	1.29
[SIV]	10.5 μ	0.38	0.04	0.09	0.11	0.36	0.46	0.24
Sum	Sum	11.87	0.24	10.0	9.9	11.8	12.0	12.0
L(total)	E37	24.44	0.31	20.9	20.4	23.9	24.4	24.5
T(in)		7655	157			6547	7980	8246
T(H+)		8052	32			7530	8062	8072
<He+>/<H+>		0.76	0.04			0.75	0.73	0.73
R(out)	E18	1.46	0.00			1.45	1.46	1.46

6.4. Blister HII Region

This is one of the Lexington Meeting test cases, and is meant to be similar to inner regions of the Orion Nebula and is called **`blister.in`** in the test suite.

Table 5 Blister HII Region

		Mean	Ferland	Harrington	Netzer	Pequignot	Rubin
I(Hβ)		4.69	4.59	4.81	4.69	4.67	4.70
HeI	5876	0.12	0.13	0.11	0.12	0.12	
CII	2326+	0.16	0.14	0.20	0.10	0.15	0.23
CII	1335+	0.15	0.17	0.14	0.13	0.16	
CIII]	1909+	0.18	0.22	0.17	0.18	0.15	0.20
[NII]	6584+	0.81	0.58	0.94	0.74	0.90	0.87
[NIII]	57 μ	.033	.035	.033	.033	.032	.034
[OII]	7330+	0.12	0.10	0.13	0.09	0.12	0.14
[OII]	3727+	0.86	0.73	0.98	0.69	0.86	1.04
[OIII]	51.8 μ	0.29	0.31	0.29	0.28	0.28	0.28
[OIII]	5007+	4.18	4.74	3.90	4.40	3.90	3.96
[NeII]	12.8 μ	0.34	0.32	0.33	0.35	0.33	0.35
[NeIII]	15.5 μ	1.06	1.24	1.07	0.96	1.04	1.00
[NeIII]	3869+	0.34	0.48	0.32	0.35	0.26	0.29
[SIII]	18.7 μ	0.33	0.31	0.34	0.31	0.33	0.35
[SIII]	9532+	1.46	1.41	1.46	1.51	1.42	1.53
[SIV]	10.5 μ	0.51	0.54	0.52	0.51	0.53	0.43
Sum	Sum	10.78	11.32	10.80	10.63	10.46	10.71
I(total)		51.3	52.6	52.4	50.4	49.4	50.3
T(in)		7911	8206	7582	8200	8200	7366
T(H+)		8303	8324	8351	8310	8200	8328
<He+>/<H+>		0.85	0.94	0.78	0.93	0.79	0.84
ΔR	E17	2.99	2.88	3.08	2.93	2.98	3.09

Table 6 Blister HII Region vs Cloudy

		Mean	STD	90	90.03
I(Hβ)		4.69	0.08	4.71	4.70
HeI	5876	0.12	0.01	0.14	0.14
CII]	2326+	0.16	0.05	0.16	0.12
CII	1335+	0.15	0.02	0.2	0.18
CIII]	1909+	0.18	0.03	0.27	0.28
[NII]	6584+	0.81	0.14	0.5	0.49
[NIII]	57 μ	0.03	0.00	0.04	0.040
[OII]	7330+	0.12	0.02	0.11	0.11
[OII]	3727+	0.86	0.15	0.68	0.69
[OIII]	51.8 μ	0.29	0.01	0.33	0.33
[OIII]	5007+	4.18	0.38	5.73	5.78
[NeII]	12.8 μ	0.34	0.01	0.13	0.14
[NeIII]	15.5 μ	1.06	0.11	0.49	0.49
[NeIII]	3869+	0.34	0.08	0.23	0.24
[SIII]	18.7 μ	0.33	0.02	0.3	0.31
[SIII]	9532+	1.46	0.05	1.27	1.37
[SIV]	10.5 μ	0.51	0.04	0.66	0.43
Sum	Sum	10.78	0.32	10.80	11.1
I(total)		51.29	1.39	52.6	52.2
T(in)		7910	406	8447	8520
T(H+)		8302	59	8391	8433
<He+>/<H+>		0.85	0.07	0.95	0.95
ΔR	E17	2.99	0.09	3.02	3.03

6.5. Paris Planetary Nebula

This compares current predictions of the code with those of other participants at the Meudon (1985) and Lexington (1993) meetings on photoionization calculations, for the case of ionization by a very hot black body. The input used to generate this model planetary nebula is shown in the sample input section and is called **`parispn.in`** in the test suite. The model results are very sensitive to the detailed transfer of HeII Lyα; this line is the dominant heat source across the He^{++} region of the model nebula. The parameters were chosen to be roughly similar to NGC 7027, a very well studied object. The input stream is listed on page 493.

Table 7 Paris Meeting Planetary Nebula

Line		Meudon	Lexington	Ferland	Harrington	Netzer	Pequignot
L(Hβ)	E35	2.60	2.68	2.63	2.68	2.73	2.68
HeII (35)	erg/s	0.87	0.88	0.83	0.88	0.94	0.85
He I	5876	0.11	0.11	0.11	0.10	0.10	0.11
He II	4686	0.33	0.33	0.32	0.33	0.35	0.32
C II]	2326+	0.38	0.33	0.33	0.43	0.27	0.30
C III]	1909+	1.70	1.77	1.82	1.66	1.72	1.87
C IV	1549+	1.64	2.33	2.44	2.05	2.66	2.18
[N II]	6584+	1.44	1.49	1.59	1.45	1.47	1.44
N III]	1749+	0.11	0.12	0.13	0.13	0.11	0.13
[NIII]	57 μ		0.13	0.12	0.13	0.13	0.13
N IV]	1487+	0.12	0.19	0.20	0.15	0.21	0.19
N V	1240+	0.09	0.17	0.18	0.12	0.23	0.15
[O I]	6300+	0.15	0.14	0.15	0.12	0.14	0.14
[O II]	3727+	2.23	2.25	2.23	2.27	2.31	2.18
[O III]	5007+	20.9	20.76	21.1	21.4	19.4	21.1
[O III]	4363	0.16	0.15	0.16	0.16	0.14	0.16
[O III]	52 μ	1.43	1.43	1.42	1.44	1.40	1.46
[O IV]	26 μ	3.62	3.67	3.52	3.98	3.32	3.86
O IV]	1403+	0.13	0.26	0.20	0.23	0.26	0.33
O V]	1218+	0.09	0.20	0.20	0.11	0.29	0.19
[Ne III]	15.5 μ	2.51	2.78	2.75	2.76	2.80	2.81
[Ne III]	3869+	2.59	2.69	3.33	2.27	2.74	2.44
Ne IV]	2423+	0.56	0.78	0.72	0.74	0.91	0.74
[Ne V]	3426+	0.73	0.67	0.74	0.60	0.73	0.61
[Ne V]	24.2 μ	1.67	0.87	0.94	0.76	0.81	0.99
Mg II	2798+	1.48	1.58	2.33	1.60	1.22	1.17
[Mg IV]	4.5 μ	0.09	0.12	0.12	0.13		0.12
[Si II]	34.8 μ	0.13	0.19	0.16	0.26	0.19	0.17
Si II]	2335+	0.11	0.16	0.15		0.16	0.16
Si III]	1892+	0.20	0.41	0.39	0.32	0.46	0.45
Si IV	1397+	0.15	0.18	0.20	0.15	0.21	0.17
[S II]	6720+	0.39	0.35	0.21	0.45	0.33	0.43
[S III]	18.7 μ	0.49	0.48	0.48	0.49	0.46	0.49
[S III]	9532+	2.09	1.96	2.04	1.89	2.05	1.87
[S IV]	10.5 μ	1.92	1.98	1.92	2.21	1.81	1.98
L(total)	E35	129	137	139	136	135	136
T(in)	E4		1.81	1.83	1.78	1.84	1.78
T(H+)	E4		1.25	1.22	1.21	1.35	1.21
<He+>/<H+>			0.72	0.74	0.74	0.71	0.71
R(out)	E17		4.06	4.04	4.04	4.07	4.07

Table 8 Paris Planetary vs Cloudy

Line		Lexington	STD	74.23	76.03	80.06	84.15	90	90.03
L(Hβ)	E35	2.68	0.04	2.57	2.66	2.52	2.34	2.55	2.53
HeII (35)	erg/s	0.88	0.05				0.83	0.79	0.79
He I	5876	0.11	0.01	0.11	0.11	0.11	0.11	0.11	0.11
He II	4686	0.33	0.01	0.29	0.32	0.35	0.36	0.31	0.31
C II]	2326+	0.33	0.07	0.36	0.35	0.37	0.35	0.28	0.28
C III]	1909+	1.77	0.09	1.57	1.48	1.72	1.72	1.82	1.84
C IV	1549+	2.33	0.27	2.24	2.76	2.48	2.19	2.48	2.63
[N II]	6584+	1.49	0.07	1.42	1.40	1.44	1.48	1.45	1.29
N III]	1749+	0.12	0.01	0.10	0.08	0.10	0.11	0.11	0.12
[NIII]	57 μ	0.13	0.00				0.12	0.12	0.13
N IV]	1487+	0.19	0.03	0.16	0.12	0.11	0.15	0.23	0.23
N V	1240+	0.17	0.05	0.14	0.09	0.06	0.09	0.17	0.17
[O I]	6300+	0.14	0.01	0.15	0.15	0.16	0.16	0.15	0.15
[O II]	3727+	2.25	0.06	2.24	2.19	2.35	2.40	1.92	2.16
[O III]	5007+	20.76	0.91	21.2	21.1	20.4	20.8	23.6	22.6
[O III]	4363	0.15	0.01	0.14	0.13	0.15	0.16	0.18	0.17
[O III]	52 μ	1.43	0.03	1.40	1.35	1.37	1.35	1.36	1.32
[O IV]	26 μ	3.67	0.30			3.42	3.65	3.23	3.42
O IV]	1403+	0.26	0.05	0.19	0.22	0.11	0.15	0.24	0.25
O V]	1218+	0.20	0.07	0.17	0.11	0.07	0.11	0.19	0.19
[Ne III]	15.5 μ	2.78	0.03	2.77	2.70	2.67	2.71	1.9	1.88
[Ne III]	3869+	2.69	0.47	3.21	3.01	3.24	3.32	2.73	2.74
Ne IV]	2423+	0.78	0.09	0.62	0.51	0.51	0.63	0.78	0.80
[Ne V]	3426+	0.67	0.08	0.64	0.54	0.53	0.64	0.8	0.80
[Ne V]	24.2 μ	0.87	0.11	0.24	0.25	1.01	1.04	1.06	1.05
Mg II	2798+	1.58	0.54	0.83	1.82	1.96	2.33	2.23	2.22
[Mg IV]	4.5 μ	0.12	0.00	0.12	0.13	0.14	0.13	0.13	0.12
[Si II]	34.8 μ	0.19	0.04	0.16	0.16	0.16	0.17	0.16	0.16
Si II]	2335+	0.16	0.01	0.15	0.14	0.16	0.18	0.16	0.16
Si III]	1892+	0.41	0.07	0.32	0.42	0.42	0.42	0.46	0.50
Si IV	1397+	0.18	0.03	0.17	0.24	0.22	0.15	0.18	0.23
[S II]	6720+	0.35	0.11	0.38	0.68	0.66	0.36	0.35	0.35
[S III]	18.7 μ	0.48	0.02	0.58	0.71	0.67	0.47	0.45	0.47
[S III]	9532+	1.96	0.10	1.69	2.11	2.07	1.97	1.75	1.88
[S IV]	10.5 μ	1.98	0.17	1.64	1.32	1.53	1.78	1.86	2.16
L(total)	E35	137	1.60	117	124	128	121	135	144
T(in)	E4	1.81	0.03				1.49	1.84	1.82
T(H+)	E4	1.25	0.07				1.28	1.21	1.22
<He+>/<H+>		0.72	0.02				0.72	0.74	0.74
R(out)	E17	4.06	0.02				3.90	4	3.99

6.6. Paris NLR Model

This compares current predictions of the code with those of other participants at the Meudon meeting on photoionization calculations, for a model similar to the NLR of active nuclei. Results for other codes are from the 1985 Meudon meeting. The input stream is called **`parisnlr.in`** in the test suite is listed on page 493.

Table 9 Paris Meeting NLR Model

Line		Netzer	Pequignot	Binette	Kraemer	Mean
Hβ	erg/s/cm^2	0.129	0.134	0.124	0.12	0.127±0.006
Hβ	4861	1.00	1.00	1.00	1.00	1.00
Lyα	1216	35.3	33.1	-	24.0	30.8±6.0
He I	5876	0.095	0.098	0.092	0.090	0.094±0.004
He II	4686	0.36	0.32	0.38	0.37	0.358±0.026
C II]	2326	0.96	0.77	1.70	1.06	1.12±0.40
C II	1335	0.14	0.14	0.20	0.08	0.14±0.05
C III]	1909	4.59	4.99	6.50	4.91	5.25±0.85
C IV	1549	7.03	7.20	5.30	7.20	6.68±0.93
[N I]	5200	0.31	0.33	0.82	0.37	0.46±0.24
[N II]	6548	2.68	1.52	1.77	1.63	1.90±0.53
N III]	1749	0.40	0.40	0.43	0.48	0.428±0.038
N IV]	1487	0.45	0.43	0.51	0.48	0.468±0.035
N V	1240	0.32	0.30	0.32	0.28	0.305±0.019
[O I]	63.2 μm	-	0.62	0.14	0.10	0.29±0.29
[O I]	6300	1.32	0.90	1.62	1.04	1.22±0.32
[O II]	7325	0.11	0.094	0.16	0.10	0.116±0.03
[O II]	3727	3.4	2.62	4.41	2.73	3.29±0.82
[O III]	52μm	2.5	2.54	2.31	2.65	2.50±0.14
[O III]	5007	27.36	27.36	23.28	27.76	26.44±2.11
[O III]	4363	0.42	0.41	0.44	0.44	0.428±0.015
O III]	1663	0.97	0.95	0.92	1.01	0.963±0.038
[O IV]	25.9μm	5.69	5.19	5.49	-	5.46±0.25
O IV]	1403	0.53	0.44	0.51	0.66	0.534±0.092
O V]	1218	0.33	0.32	0.45	0.24	0.335±0.086
O VI	1035	0.17	0.17	0.22	0.10	0.165±0.049
[Ne II]	12.8μm	0.28	0.18	0.48	0.13	0.268±0.155
[Ne III]	15.5μm	2.8	2.62	1.83	1.25	2.13±0.72
[Ne III]	3869	2.70	2.59	2.27	1.67	2.31±0.46
Ne IV]	2423	0.82	0.79	1.03	1.12	0.94±0.16
[Ne V]	24.2μm	3.54	2.64	3.54	-	3.24±0.52
[Ne V]	3426	1.17	1.02	1.13	1.05	1.095±0.066
Mg II	2798	1.58	1.43	1.51	1.10	1.40±0.21
Si II	34.8μm	1.73	0.97	0.51	-	1.07±0.62
Si II	2335	0.21	0.17	0.09	-	0.16±0.06
Si III]	1892	0.15	0.19	0.69	0.14	0.29±0.26
Si IV	1397	0.21	0.14	0.02	0.13	0.13±0.08
S II	6720	1.00	0.62	1.29	0.37	0.82±0.41
S II	4070	0.07	0.04	0.078	0.03	0.055±0.023
S III	18.7μm	0.75	0.49	0.68	0.65	0.64±0.11
S III	9532	2.25	1.38	1.73	1.62	1.74±0.37
S IV	10.5μm	1.39	0.73	0.94	1.57	1.16±0.39

Table 10 Paris NLR Model vs CLOUDY

Line		Mean	84.15	90	90.03
Hβ	erg/s/cm^2	0.127±0.006	0.133	0.136	0.135
Hβ	4861	1.00	1.00	1.00	1.00
Lyα	1216	30.8±6.0	32.3	34.2	34.5
He I	5876	0.094±0.004	0.104	0.098	0.098
He II	4686	0.358±0.026	0.351	0.33	0.33
C II]	2326	1.12±0.40	0.766	0.69	0.67
C II	1335	0.14±0.05	0.126	0.14	0.14
C III]	1909	5.25±0.85	5.02	4.67	4.67
C IV	1549	6.68±0.93	8.42	7.12	7.47
[N I]	5200	0.46±0.24	0.14	0.14	0.15
[N II]	6548	1.90±0.53	2.32	2.54	2.42
N III]	1749	0.428±0.038	0.45	0.36	0.38
N IV]	1487	0.468±0.035	0.553	0.50	0.53
N V	1240	0.305±0.019	0.391	0.26	0.30
[O I]	63.2 μm	0.29±0.29	0.36	0.34	0.34
[O I]	6300	1.22±0.32	1.02	0.05	0.04
[O II]	7325	0.116±0.03	0.111	0.097	0.11
[O II]	3727	3.29±0.82	2.99	2.60	2.72
[O III]	52μm	2.50±0.14	2.34	2.31	2.22
[O III]	5007	26.44±2.11	25.3	26.5	26.0
[O III]	4363	0.428±0.015	0.45	0.44	0.44
O III]	1663	0.963±0.038	1.05	1.04	1.07
[O IV]	25.9μm	5.46±0.25	5.76	5.37	5.86
O IV]	1403	0.534±0.092	0.54	0.45	0.52
O V]	1218	0.335±0.086	0.453	0.24	0.28
O VI	1035	0.165±0.049	0.222	0.13	0.15
[Ne II]	12.8μm	0.268±0.155	0.220	0.19	0.17
[Ne III]	15.5μm	2.13±0.72	3.13	2.11	2.12
[Ne III]	3869	2.31±0.46	3.99	3.11	3.23
Ne IV]	2423	0.94±0.16	1.19	1.07	1.15
[Ne V]	24.2μm	3.24±0.52	2.74	2.69	2.75
[Ne V]	3426	1.095±0.066	1.45	1.19	1.25
Mg II	2798	1.40±0.21	1.76	1.78	1.67
Si II	34.8μm	1.07±0.62	1.12	1.07	1.05
Si II	2335	0.16±0.06	0.218	0.20	0.20
Si III]	1892	0.29±0.26	0.401	0.37	0.46
Si IV	1397	0.13±0.08	0.140	0.13	0.19
S II	6720	0.82±0.41	1.50	1.10	1.00
S II	4070	0.055±0.023	0.1004	0.106	0.096
S III	18.7μm	0.64±0.11	0.625	0.65	0.67
S III	9532	1.74±0.37	1.92	1.68	1.62
S IV	10.5μm	1.16±0.39	1.73	1.52	1.21

6.7. Lexington NLR Model

This is the NLR model computed at the 1994 Lexington meeting, and is called **`nlr.in`** in the test suites.

Table 11 Lexington NLR Model

		Lexington	Binette	Ferland	Netzer	Pequignot	Viegas
I(Hβ)	E0	1.36	1.33	1.31	1.37	1.43	1.34
Lyα	1216	33.70	38.3	32.1	32.4	31.5	34.2
HeI	5876	0.12	0.11	0.13	0.12	0.13	0.13
HeII	4686	0.24	0.25	0.25	0.25	0.23	0.24
HeII	1640	1.62	1.60	1.74	1.53	1.56	1.67
CIII]	1909+	2.90	2.90	2.99	2.87	2.83	2.90
CIV	1549+	3.35	2.70	3.85	3.69	3.17	3.36
[NII]	6584+	2.55	1.40	3.20	3.10	2.67	2.40
NIII]	1749+	0.23	0.24	0.24	0.22	0.22	0.22
NIV]	1487+	0.21	0.20	0.23	0.22	0.21	0.21
[OI]	6300+	1.65	2.20	1.61	1.67	1.31	1.46
[OI]	63 μ	1.12	0.25	1.13		1.44	1.64
[OII]	3727+	1.42	1.60	1.44	1.58	1.30	1.20
OIII]	1663+	0.56	0.35	0.63	0.61	0.57	0.63
[OIII]	5007+	33.54	31.4	34.5	33.0	32.8	36.0
[OIII]	4363	0.32	0.30	0.34	0.31	0.30	0.33
OIV	1403+	0.36	0.49	0.30	0.36	0.42	0.25
[NeIII]	15.5 μ	1.89	1.50	2.01	1.94	2.05	1.95
[NeIII]	3869+	2.13	1.90	2.51	2.16	1.72	2.34
[Ne IV]	2423+	0.44	0.52	0.42	0.47	0.41	0.38
[NeV]	3426+	0.52	0.59	0.55	0.53	0.44	0.50
MgII	2798+	1.84	3.50	1.72	1.23	1.12	1.61
[SiII]	34.8 μ	0.90	1.00	0.96	1.07	0.96	0.52
[SII]	6720+	1.29	2.40	1.01	0.93	0.99	1.10
[SIII]	9532+	1.91	1.60	2.15	2.06	1.67	2.08
[SIII]	18.7 μ	0.49	0.36	0.61	0.57	0.52	0.37
[SIV]	10.5 μ	1.02	0.86	1.24	0.82	0.94	1.22
I(total)	E0	130	131	128	128	131	133
T(in)	E4	1.70	1.71	1.70	1.72	1.68	1.68
T(H+)	E4	1.18		1.24	1.06	1.20	1.23

Table 12 Lexington NLR vs Cloudy

		Mean	STD	90	90.03
I(Hβ)	E0	1.36	0.05	1.38	1.36
Lyα	1216	33.70	2.76	32.1	32.1
HeI	5876	0.12	0.01	0.12	0.12
HeII	4686	0.24	0.01	0.24	0.24
HeII	1640	1.62	0.09	1.7	1.73
CIII]	1909+	2.90	0.06	2.55	2.62
CIV	1549+	3.35	0.45	3.71	3.96
[NII]	6584+	2.55	0.72	3.18	3.05
NIII]	1749+	0.23	0.01	0.19	0.20
NIV]	1487+	0.21	0.01	0.26	0.28
[OI]	6300+	1.65	0.34	1.63	1.61
[OI]	63 μ	1.12	0.61	0.72	0.73
[OII]	3727+	1.42	0.17	1.28	1.35
OIII]	1663+	0.56	0.12	0.64	0.66
[OIII]	5007+	33.54	1.76	35	34.8
[OIII]	4363	0.32	0.02	0.32	0.33
OIV	1403+	0.36	0.09	0.35	0.40
[NeIII]	15.5 μ	1.89	0.22	1.39	1.44
[NeIII]	3869+	2.13	0.32	1.9	1.97
[Ne IV]	2423+	0.44	0.06	0.43	0.47
[NeV]	3426+	0.52	0.06	0.57	0.60
MgII	2798+	1.84	0.96	1.54	1.48
[SiII]	34.8 μ	0.90	0.22	0.88	0.88
[SII]	6720+	1.29	0.63	1	0.91
[SIII]	9532+	1.91	0.26	1.92	2.01
[SIII]	18.7 μ	0.49	0.12	0.64	0.68
[SIV]	10.5 μ	1.02	0.20	1.16	1.05
I(total)	E0	130	1.86	132	97
T(in)	E4	1.70	0.02	1.7	1.703
T(H+)	E4	1.18	0.08	1.22	1.06

6.8. The DQ Her Shell

This is more or less the model of the DQ Her nebula proposed by Ferland et al. (1984). The input stream for this model is given on page 470 and is called **dqher.in** in the test suite. The big difference between C90 and previous versions is in the intensity of Hβ predicted. The code no longer assumes case B when the temperature is too low to do the matrix solution. The nebula is optically thin to many Lyman

Table 13 The DQ Her Shell

Line		80.09	84.15	90	90.03
Hβ (+30)	4861	1.65	1.62	1.19	0.93
Totl	4861				0.658
Case B	4861			1.00	1.00
Lyα	1216	40.9	20.7	20.6	20.6
He I	5876	0.786	0.315	0.29	0.25
He II	4686	0.166	0.085	0.077	0.086
C II	158μ	0.777	0.62	0.079	0.089
C II	1335		0.0.062	0.12	0.014
[N I]	5200	0.144		0.047	0.040
[N II]	122μ	7.85	3.43	2.98	3.13
[N III]	57μ	5.30	3.78	3.87	8.46
[O II]	3727	1.16	0.304	0.28	0.30
[O III]	88μ	12.1	6.36	7.09	4.46
[O III]	52μ	12.4	6.60	7.44	5.04
[Si II]	35μ	0.586	0.79	0.59	0.77
[S III]	34μ	0.114	0.167	0.13	0.21
[Fe II]		0.121	0.187	0.31	0.13
<Te>	K		643	552	796

lines and these escape, robbing flux from Hβ.

6.9. The Kwan and Krolik Standard Model

Table 14 gives the spectrum of the Kwan and Krolik (1981) standard model, called **kk.in** in the test suite. The input stream is given on page 480.

Table 14. The Kwan and Krolik Standard Model.

Line	KK81	80	84.15	90	90.03
Lyα 1216	5.512+7[3]	5.78	5.59	5.83	5.87
Lyα 1216	100	100	100	100	100
Hβ 4861	10.3	6.37	6.05	5.90	5.85
Hα 6563	42.8	19.9	19.56	24.6	24.4
BaC	47.0	38.8	39.5	62.2	59.8
PaC	30.7	20.2	20.0	39.2	37.7
2s-1s	3.3	2.35	3.19	2.90	2.95
free-free	29.0	25.3	23.3	42.8	43.2
He I 10830	4.4	2.86	2.84	2.53	2.60
He I 5876	0.9	0.697	0.845	0.63	0.63
He II 4686	-	0.365	0.316	0.30	0.30
He II 1640	2.5	3.17	2.77	2.53	2.54
C II] 2326	3.3	2.70	0.686	3.26	3.19
C II 1335	-	0.661	0.611	0.67	0.67
C III 977	5.1	7.53	6.52	5.95	5.79
C III] 1909	13.0	20.7	15.9	15.0	14.1
C IV 1549	67.0	95.7	76.3	73.6	72.7
N III] 1750	1.5	4.26	4.95	4.58	4.40
N III 990	-	0.324	1.35	1.75	1.44
N IV] 1486	5.8	4.88	5.24	5.56	5.43
N V 1240	8.4	3.61	4.95	3.19	3.90
O I 1304	6.9	4.01	3.93	7.74	7.56
O III] 1663	9.5	18.4	18.3	19.4	19.4
O IV] 1402	5.2	4.81	5.84	5.98	6.40
O V] 1218	6.7	3.07	4.63	2.60	3.30
O VI 1034	15.0	2.85	4.71	1.95	3.06
Mg II 2798	18.0	17.6	9.71	32.3	30.3
Si III] 1892	-	14.6	15.7	12.2	11.9
Si IV 1397	5.5	14.2	9.77	8.91	8.86
Fe II	23.9	9.58	11.3	12.2	14.9
Fe Kα		2.11	1.74	1.92	1.93

[3]Line intensity in erg s^{-1} cm^{-2}. The entries which follow are relative to a scale where Lyα=100.

6.10. Rees, Netzer, and Ferland, low density

This is the lower density model listed in Table 1 of Rees et al. (1989). It is **`rnfa.in`** in the sample input streams.

Table 15 Rees, Netzer, and Ferland, low density case.

	1989	C90	90.03	Ion
	log n = 10			
Lyα	1.000	1.000	1.00	1.000
Hβ	0.030	0.028	0.028	0.026
Hα	0.197	0.134	0.133	0.180
Pα	0.022	0.015	0.015	0.020
Ba C	0.127	0.140	0.138	0.125
Pa C	0.052	0.057	0.052	0.025
ff	0.082	0.093	0.100	0.089
HeI 5876	0.007	0.006	0.0064	0.008
HeII 4686	0.005	0.004	0.0040	0.004
HeII 1640	0.039	0.035	0.0351	0.034
CII 1335	0.009	0.018	0.0104	0.008
CII] 2326	0.014	0.011	0.0104	0.011
CIII 977	0.030	0.026	0.0266	0.035
CIII] 1909	0.106	0.104	0.103	0.103
CIV 1549	0.424	0.460	0.462	0.453
NIII] 1750	0.012	0.013	0.0135	0.014
NIV] 1486	0.009	0.013	0.0123	0.009
NV 1240	0.002	0.002	0.0025	0.003
OI 1304	0.033	0.013	0.0124	0.013
OI 8446	0.005		0.005	0.005
OIII] 1663	0.055	0.067	0.065	0.060
OIV] 1402	0.011	0.014	0.0144	0.017
Mg II 2798	0.076	0.200	0.195	0.160
SiIII 1207	0.012	0.010	0.0106	0.013
SiIII] 1892	0.085	0.086	0.0874	0.090
SiIV 1397	0.048	0.044	0.046	0.044
FeII		0.054	0.0169	0.052
Kα		0.001	0.0011	0.001
sum	1.244	2.416	2.606	2.323
Lyα	4.85E+07	4.42+07	4.40	4.55+07
I(total)	6.03E+07	6.25+07	7.07+7	6.02+07

6.11. Rees, Netzer, and Ferland, high density

This is the higher density model listed in Table 1 of Rees et al. (1989). It is **`rnfb.in`** in the sample input streams.

Table 16 Rees, Netzer, and Ferland, high density case.

	1989	90.0	90.03	Ion
	log n = 12			
Lyα	1.000	1.000	1.000	1.000
Hβ	0.063	0.021	0.021	0.054
Hα	0.175	0.062	0.0617	0.134
Pα	0.009	0.005	0.0048	0.008
Ba C	2.938	2.470	2.43	2.680
Pa C	1.313	1.110	1.09	0.590
ff	1.563	1.340	1.33	1.300
HeI 5876	0.038	0.035	0.035	0.032
HeII 4686	0.030	0.031	0.0304	0.017
HeII 1640	0.266	0.188	0.185	0.140
CII 1335	0.082	0.124	0.0486	0.099
CII] 2326	0.003	0.001	0.0015	0.002
CIII 977	0.225	0.105	0.105	0.150
CIII] 1909	0.023	0.010	0.0098	0.014
CIV 1549	0.938	0.591	0.588	0.860
NIII] 1750	0.006	0.003	0.0029	0.004
NIV] 1486	0.006	0.004	0.0037	0.004
NV 1240	0.031	0.023	0.0026	0.021
OI 1304	0.033	0.020	0.0201	0.028
OI 8446	0.006	0.003	0.0026	0.005
OIII] 1663	0.039	0.024	0.0244	0.024
OIII 835	0.041	0.007	0.0197	0.002
OIV] 1402	0.023	0.015	0.017	0.020
Mg II 2798	0.088	0.136	0.136	0.170
SiIII 1207	0.088	0.049	0.048	0.068
SiIII] 1892	0.113	0.039	0.0381	0.061
SiIV 1397	0.300	0.190	0.186	0.230
FeII		0.017	0.021	0.015
Kα		0.004	0.0042	0.003
sum	8.188	7.516	7.4671	7.521
Lyα	8.54+08	1.11+09	1.11	1.08+09
I(total)	7.00+09	7.21+09	7.17	7.04+09

7. SAMPLE INPUT

7.1. Overview

These are the input streams that are used to exercise CLOUDY under both standard and extreme conditions. They are contained in the file **`tests.in`** that is found in the web site. These test cases should be run to validate the machine and compiler used to run CLOUDY.

7.2. *atlas* Atlas 1991 Model Atmospheres

```
title Model of a Compact HII Region
c Approximate model of a hot star ionizing its surroundings.
c Assume H density of 1e+6 per cc.  This run tests the new Kurucz model option.
c I choose the T_eff and log(g) values arbitrarily.  It implies a star
c with luminosity 32300 L_sun, radius 6.4745 R_sun, and mass 3.745 M_sun.
set path="/home16/users/gary/cloudy/"
table star atlas 30400.0 4.2
luminosity 4.509202522 solar
sphere
radius -3.0 parsec
hden 6.0
print last iteration
iterate
plot continuum range .1
punch overview last file='atlas.ovr'
punch results last file='atlas.rlt'
c atlas.in
```

This is a test that the code can correctly access the large block of Atlas model atmosphere continua described by Kurucz (1991). Kevin Volk provided it as part of his original coding of these stellar atmosphere files.

Checks:

- **`table star atlas`** command works.

7.3. *blister* Lexington Meeting Blister model

```
title conditions similar to Orion nebula blister
init path file='c84.ini'
dielec kludge 0
turbulence 10
sphere
blackbody 40,000
phi(h) 13.0
hden 4
abundances hii region no grains
abundances 1 1 1 1 1 1 mg=.0001 al1 si.01 s1 ar1
continue ca1 fe.001 ni1
iterate
print last iteration
print faint .01
punch overview last file='blister.ovr'
punch results last file='blister.rlt'
print line sum
TOTL 5876
C  2 2326
C  2 1335
TOTL 1909
N  2 6584
N  2 6548
N  3  570
TOTL 7325
TOTL 3727
O  3  518
O  3 5007
O  3 4959
Ne 2  128
Ne 3  156
Ne 3 3869
Ne 3 3968
S  3  187
S  3 9532
S  3 9069
S  4  105
end of lines
c blister.in
```

This is one of the test cases from the Lexington Meeting suite of nebulae (Ferland et al. 1995). It is a grain-free blister HII region, similar to inner regions of the Orion Nebula, except for the absence of grains.

The set of lines entered with the **`print line sum`** command lists the most powerful coolants in this model. This is one of the tabulated quantities in the Lexington Meeting, and is a fundamental test of energy conservation in the code. The ratio of the sum of these lines to Hβ is equivalent to the Stoy ratio, used for determining stellar temperatures.

7.4. *blr* Ferland and Persson BLR model

```
title final F+P BLR model table 3
stop column density 25.5
iterate to convergence
print last
ionization parameter -0.5
hden 9.5
table agn
plot continuum
punch overview last file='blr.ovr'
punch results last file='blr.rlt'
c blr.in
```

Ferland and Persson (1989) presented this calculation. The differences between the present predictions and those given by FP are largely due to improved treatment of Balmer line escape and destruction.

7.5. *brems* hot interstellar medium

```
title generate continuum due to hot ism in high Z,z starburst
coronal equilibrium, t=1,000,000K
stop column density 21
hden 1
punch continuum last file='brems.ovr'
punch results last file='brems.rlt'
abundances starburst Z=10 Zsun
c brems.in
```

This model generates a large column constant density cloud similar to the hot phase of the interstellar medium. The continuum is punched to generate one of the figures in Part 2 of HAZY.

7.6. *casea* Case A

```
title case A
c Seaton 1959 results; 4pi j(beta) 5.56E-26
c for this model total H-beta=4.745
c b(2)=3.73E-3 (3)=3.69E-2 (4)=0.091 (5)=0.145 (6)=0.193
c n.b. very different results if not l-mixed
no induced processes ;N.B. case A is a fiction; no incuded transitions
hydrogen collisions off ; must turn off all collisions except 2s-2p
stop lyman optical depth -6
constant temperature = 10,000K
print departure coefficients
stop zone 1
black body, T=50,000K
ionization parameter -2
hden = 15; must be high to mix 2s, 2p
set dr 0
abundances all -10
c casea.in
```

Case A is a mathematical fiction; when the Lyman lines are optically thin continuum pumping must be important if the gas is ionized. Fluorescence is turned off with the **`no induced processes`** command. The density is set to a very high value (10^{15} cm^{-3}) so that the 2s-2p states are well *l*-mixed, in keeping with standard case A assumptions. As a result, collisional excitation would dominate the level populations, and hydrogen collisions must be turned off with the "hydrogen collisions off" command. The Lyα optical depth is set to a small value. The **`set dr`** command sets the zone thickness to 1 cm. The abundances are set to a very small value so that the electron density is equal to the hydrogen density.

Checks:

- Departure coefficients for H, He levels
- Neutral fractions
- Hβ emissivity

7.7. *casebn2* low density Case B

```
title log density case B, T=5000, log n=2
case b hummer
no induced processes
constant temperature = 5,000
stop zone 1
black body, T=50,000K
ionization parameter -2
hden = 2
abundances all -10
set dr 0
hydrogen levels 50
punch results file="casebn2.rlt"
c casebn2.in
```

This test case compares the predictions of the multi-level hydrogen atom with the Storey and Hummer (1995) results. The **`set dr`** command sets the zone thickness to 1 cm. The **`case b`** command sets Lyman line optical depths to very large values.

Checks:

- Neutral fractions
- Hβ emissivity
- Relative intensities of hydrogen lines

7.8. *casebn8* high density Case B

```
title high density case B
case b hummer
constant temperature = 10,000
stop zone 1
black body, T=50,000K
ionization parameter -2
hden = 8
abundances all -10
set dr 0
hydrogen levels 50
punch results file="casebn8.rlt"
c casebn8.in
```

This test case compares the predictions of the multi-level hydrogen atom with the Storey and Hummer (1995) results. The **`set dr`** command sets the zone thickness to 1 cm. The **`case b`** command sets Lyman line optical depths to very large values.

Checks:

- Neutral fractions
- Hβ emissivity
- Relative line intensities

7.9. *casec* Case C

```
title case C
stop lyman optical depth -6
constant temperature = 10,000K
print departure coefficients
stop zone 1
black body, T=50,000K
ionization parameter -2
hden = 15; must be high to mix 2s, 2p
set dr 0
abundances all -10
c casec.in
```

This is Case C, what really happens when optically thin gas is irradiated by a continuum with Lyman line continuum fluorescence allowed.

Checks:

- Departure coefficients for H, He levels
- Neutral fractions
- Hβ emissivity

7.10. *collion.for* Grid of collisional ionization models

```
      program main
      logical ok
      include "/home16/users/gary/cloudy/limelm.par"
      include "/home16/users/gary/cloudy/IonFracs.com"
      include "/home16/users/gary/cloudy/elnam.com"
      parameter (nmax=100)
      real xionsave(nmax,limelm,limelm+1)
      real tesave(nmax)
      character*80 line , chLab(31)*5
*     redirect output to file collion.out
      icldy = 12
      open(icldy,file='collion.out')
      hden = 1.
      telog = 3.6
      teinc = 0.2
      nte = 1.
      do while (telog.le.9.0 .and. nte.lt.nmax )
        call cdinit
*       option to redirect output to file openned above
        call cdoutp( icldy )
*       tell it so be quiet
        call cdtalk( .false. )
*       option to not execute code beyond header
*       call cdnoex
        write(line,'(''title run code for coll ion grid'',
     1  F7.2)') hden
        call cdread( line , nleft )
        write(line,'(''no photoionization'')')
        call cdread( line , nleft )
        write(line,'(''table agn'')')
        call cdread( line , nleft )
        write(line,'(''phi(h) '',f7.2)') -4.
        call cdread( line , nleft )
        write(line,'(''set trim -20 '')')
        call cdread( line , nleft )
        write(line,'(''stop zone 1 '')')
        call cdread( line , nleft )
        write(line,'(''dielec kludge off '')')
        call cdread( line , nleft )
        write(line,'(''hden '',f5.2)') hden
        call cdread( line , nleft )
        write(line,'(''constant temper '',f5.2)') telog
        call cdread( line , nleft )
        call cddriv( ok )
        do nelem=1,limelm
          do ion=1,limelm+1
            xionsave(nte, nelem, ion ) =
     1      xIonFracs(nelem,ion)/xIonFracs(nelem,0)
          enddo
        enddo
        tesave(nte) = telog
        telog = telog + teinc
        nte = nte + 1
      enddo
      do nelem=1,limelm
        if( nelem+1.le.15 .and. (nelem/2)*2.eq.nelem ) then
          write(icldy,'(/)')
          write(icldy,'(''   Element'',i4,3x,a10)') nelem,chElNam(nelem)
        else
          if( nelem.ne.1 ) then
            write(icldy,'(''1  Element'',i4,3x,a10)') nelem,
     1      chElNam(nelem)
          else
            write(icldy,'(''   Element'',i4,3x,a10)') nelem,
     1      chElNam(nelem)
          endif
        endif
        write(icldy,'(''    Te'',15i5)') (i,i=1,min(15,nelem+1))
        doi=1,nte-1
          doion=1,min(15,nelem+1)
            if(xionsave(i,nelem,ion).gt.1e-9 ) then
              write(chLab(ion),'(f5.2)') -log10(xionsave(i,nelem,ion))
            else
              chLab(ion) = '  .0 '
            endif
          enddo
```

```
      write(icldy,'(2x,f5.2,15a5)') tesave(i),
     1   (chLab(ion),ion=1,min(15,nelem+1))
        enddo
        if( nelem+1.gt.15 ) then
          write(icldy,'(/)')
          write(icldy,'(''   Element'',i4,3x,a10)') nelem,chElNam(nelem)
          write(icldy,'(''   Te '',16i5)') (i,i=16,nelem+1)
          doi=1,nte-1
            doion=16,nelem+1
              if(xionsave(i,nelem,ion).gt.1e-9 ) then
                   write(chLab(ion),'(f5.2)') -log10(xionsave(i,nelem,ion))
                 else
                   chLab(ion) = '  .0 '
                 endif
               enddo
               write(icldy,'(2x,f5.2,16a5)') tesave(i),
      1         (chLab(ion),ion=16,nelem+1)
             enddo
          endif
        enddo
        end
```

This drives the code for collisional equilibrium. The results are directly comparable to Arnaud and Raymond (1992) or Arnaud and Rothenflug (1985).

7.11. *comphi* Compton Limit; highest temperature

```
title compton exchange near low temperature limit
black linear 3 lte
brems 5
ionizat -5
hden -10
eden 0
stop zone 1
tolerance 0.0001
iterate
print last iteration
c complo.in
```

This is the highest Compton temperature that can be computed in LTE on an IEEE 32-bit processor. This tests the code in the high-temperature Compton limit. Temperatures as high as 10^{10} K can be computed successfully on CPUs with longer word lengths, such as a Cray or the new 64 bit processors.

Checks:

- The equilibrium temperature should be exactly $10^{7.4}$ K (2.512×10^{7} K).

7.12. *complo* Compton Limit; lowest temperature

```
title compton exchange near low temperature limit
black linear 3 lte
brems 5
ionizat -5
hden -10
eden 0
stop zone 1
tolerance 0.0001
iterate
print last iteration
c complo.in
```

This tests the code in the low temperature Compton limit. The gas is illuminated by a 3 K blackbody in thermodynamic equilibrium. The equilibrium temperature should be *exactly* 3 K. It fails a bit high because the frequency bandwidth of the code only extends down to 1.001×10^{-5} Ryd. It is necessary to add an extra component of free electrons to test the code in this limit with the **eden** command. This causes the warning that a physical process (charge conservation) has been disabled.

Checks:

- The equilibrium temperature should be exactly 3 K.

7.13. *compton* Compton limit

```
title Compton limit, test continuum partition
c test of thermal equil in compton limit; temp should EXACTLY equal 2E5K
c check continuum partition;
c energy range, photon densities, luminosities, follow
c 0.25-1.  Q=26.6470 L=15.8190     c 1-1.807  Q=26.8867 L=16.3766
c 1.807-4  Q=27.3042 L=17.0948     c 4-20     Q=27.2041 L=17.3179
c 20 --    Q=22.9038 L=22.9038     c total lumin 17.5597
c nufnu(912A) = 1.8029E+16
black body t=200,000K lte
grains -5 no heating no cooling
hden = 6
stop zone 1
print faint .1
print last
metals -10
iterate
stop lyman continuum optical depth = -6
print departure coef
set dr 0
tolerance .0001  % decrease tolerance on heat-cool match
c compton.in
```

This tests the behavior of the code in the Compton limit. The incident continuum is a blackbody in strict thermodynamic equilibrium. Strict thermodynamic equilibrium is expected for all constituents of the gas. The input stream also lists the expected photon fluxes for the incident continuum; this tests the normalization of the continuum, and its distribution. Grains are included to confirm their behavior in the LTE limit. The **`set dr`** command sets the zone thickness to 1 cm.

Checks:

- Luminosity, photon flux, over various energy intervals, $4\pi J$ at 912Å.
- Electron temperature exactly 2×10^5 K.
- Grain temperature forced to 2×10^5 K by radiative processes.

7.14. *conserv* Energy conservation

```
title test that energy is conserved
hden 1.0
constant pressure
iterate
print last iteration
black body, temp=5.5
intensity total -4
stop temperature 3 linear
punch overview last file='conserv.ovr'
punch results last file='conserv.rlt'
c conserv.in
```

This checks that energy is conserved. The code always checks that it did not radiate more energy than was absorbed. This calculation extends well past the photo-dissociation zone into fully molecular gas, so that all of the incident radiation is absorbed.

7.15. *coolcurve.for* Collisional ionization cooling curve

```
      program main
      logical ok
      include "/home16/users/gary/cloudy/tcool.com"
      include "/home16/users/gary/cloudy/phycon.com"
      character*80 line
*     redirect output to file coolcurve.out
      icldy = 12
      open(icldy,file='coolcurve.out')
      hdden = 1.
      telog = 4.
      nte = 1.
      do while (telog.le.9.0  )
        call cdinit
*       option to redirect output to file openned above
        call cdoutp( icldy )
*       tell it so be quiet
        call cdtalk( .false. )
*       option to not execute code beyond header
*       call cdnoex
        write(line,'(''title run code for cooling curve grid'',
     1  F7.2)') telog
        call cdread( line , nleft )
        write(line,'(''no photoionization'')')
        call cdread( line , nleft )
        write(line,'(''table agn'')')
        call cdread( line , nleft )
        write(line,'(''phi(h) '',f7.2)') -4.
        call cdread( line , nleft )
        write(line,'(''stop zone 1 '')')
        call cdread( line , nleft )
        write(line,'(''hden '',f5.2)') hdden
        call cdread( line , nleft )
        write(line,'(''constant temper '',f6.3)') telog
        call cdread( line , nleft )
        call cddriv( ok )
*       this is what they usually plot
        if(eden*hden.gt.0. ) then
          answ = ctot/eden/hden
        else
          answ = 0.
        endif
          write(icldy,'(1p,2e10.2)') telog,  answ
          telog = telog + 0.02
       enddo
       end
```

This drives the code for conditions of collisional equilibrium and computed the resulting cooling function.

7.16. *coronal* Model solar coronal region

```
title model of solar corona
set path="/home16/users/gary/cloudy/"
coronal equilibrium, 1,000,000K
radius 11 12
table star atlas T=5770 g=4.43
luminosity solar linear 1
hden log=10 power= -2
sphere
iterate
print last iteration
c coronal.in
```

This is a very rough model of the solar corona.

Checks:

- **`table star atlas`** and **`coronal equilibrium`** commands work.

7.17. *corners4.for* Four Corners of BLR grid

```
      program main
      logical ok
      parameter (nflx=7 , nhden=7 )
      real hden(nhden) , flx(nflx)
      character*80 line
*     redirect output to file corner4.out
      icldy = 12
      open(icldy,file='corners4.out')
      ierr = 13
      open(ierr,file='corners4.err')
      doized=0,1
        dond=1,nhden
          hden(nd) = 8. + float(nd-1)
          donion=1,nflx
            flx(nion) = 18. + float(nion-1)
            call cdInit
*           option to redirect output to file opened above
            call cdOutp( icldy )
*           option to not execute code beyond header
*           call cdNoex
            write(line,'(''title run code for 4 corners, n,f='',
     1      2F7.2,'' Z='',f7.2)') hden(nd) , flx(nion),10.**float(ized)
            call cdRead( line , nleft )
            write(line,'(''agn kelvin 5.89 -1.21 -0.30 '')')
            call cdRead( line , nleft )
            write(line,'(''phi(h) '',f7.2)') flx(nion)
            call cdRead( line , nleft )
            write(line,'(''background z=4 '')')
            call cdRead( line , nleft )
            if(ized.eq.1 ) then
              write(line,'(''abundan starburst 10'')')
              call cdRead( line , nleft )
            endif
            write(line,'(''iterate to convergence '')')
            call cdRead( line , nleft )
            write(line,'(''print last iteration '')')
            call cdRead( line , nleft )
            write(line,'(''stop column density 24 '')')
            call cdRead( line , nleft )
            write(line,'(''hden '',f5.2)') hden(nd)
            call cdRead( line , nleft )
            write(line,'(''failures 5 no map '')')
            call cdRead( line , nleft )
            call cdDriv( ok )
            call cdErrors(ierr)
          enddo
        enddo
      enddo
      end
```

This program runs the code over a very broad range of parameters, and two metallicities (solar, and an evolved starburst). It has a large column density and iterates the model to convergence.

7.18. *coolhii* Lexington Meeting cool HII region

```
title cool HII region model from Lexington Meeting
c Hbeta 4.93E36, L(total)4.30xHbeta
iterate   ; must iterate since fine structure lines are opticall thick
init  file='c84.ini'
dielec kludge 0
sphere
black body, T=20,000K
q(h) 49
hden = 2
radius = 18.477121
abund -1 C-3.6576 N-4.39794 O-3.481146 ne-4.30103 na=-8 mg-8 al=-8
continue si-8 s-5.04576 ar-8 ca=-8 fe-8 ni=-8
plot continuum range .1
print faint .01
print last iteration
punch overview last file='coolhii.ovr'
punch results last file='coolhii.rlt'
print line sum :this is the sum of lines in table 2 of the Lexington meeting
N  2 6584
N  2 6548
TOTL 3727
Ne 2  128
S  2 6720
S  3  187
S  3  334
S  3 9532
S  3 9069
end of lines
c coolhii.in
```

This is one of the test cases from the Lexington Meeting suite of nebulae. It is a grain-free HII region ionized by a very cool star. Hydrogen is ionized but not helium so this tests the transport of the H Lyman continuum. The set of lines is entered with the **`print line sum`** command to test energy conservation.

7.19. *dqher* DQ Her nebula

```
title (roughly) Ferland et al. DQ Her model
c model of room temperature ionized cloud arund old nova DQ Her
c roughly that of Ferland et al. 1984 Ap.J. 281, 194.
c tests behaviour of code at very low temperatures
stop temp = 100
c flat continuum, followed by nu^-2 power law
interpolate (0, 0) (0.3, 0) (8,000,000 -14.8)
luminosity total 34
hden = 2.
filling factor 0.667
covering factor 0.667
radius 16.5682, thickness=16.14613
abundances nova
grains
iterate
turbulence 300
print last
punch overview last file='dqher.ovr'
punch results last file='dqher.rlt'
normalize to 4861 label="Ca B"
age 70 years %it exploded in the 1930's
c dqher.in
```

This tests the code's behavior in the limit posed by the metal rich low density nebula surrounding DQ Her (Ferland et al. 1984).

Checks:

- Thickness exact
- Thermal stability

7.20. *drive.for* Cloudy over all dusty NLR parameters

```
      program main
      logical lgOK
      parameter (nu=10 , nhden = 10 )
      real hden(nhden) , R(nu)
      character*80 line
*     vary density and distance to dusty NLR cloud
      icldy = 12
      ierr = 84
      open(icldy,file='drive.out')
      open(ierr,file='drive.err')
      donden=1,nhden
        hden(nden) = 0.5 + float(nden) / 2.
        dondist=1,nu
          R(ndist) = -0.5 + float(ndist) / 4.
          call cdInit
*         option to redirect output to file openned above
          call cdOutp( icldy )
*         option to not execute code beyond header
*         call cdNoex
          write(line,'(''title exercise code for dusty NLR, n,f='',
     1    2F7.2)') hden(nden) , R(ndist)
          call cdRead( line , nleft )
          write(line,'(''luminosity total 44'')')
          call cdRead( line , nleft )
          write(line,'(''table agn '')')
          call cdRead( line , nleft )
          write(line,'(''radius log parsecs'',f7.2)') R(ndist)
          call cdRead( line , nleft )
          write(line,'(''abundances orion '')')
          call cdRead( line , nleft )
          write(line,'(''iterate convergence '')')
          call cdRead( line , nleft )
          write(line,'(''hden '',f5.2)') hden(nden)
          call cdRead( line , nleft )
          write(line,'(''print last iteration '')')
          call cdRead( line , nleft )
          call cdDriv( lgOK )
          call cdErrors(ierr)
        enddo
      enddo
      end
```

This runs a large grid of dusty NLR models and is a check that the code performs well over a very broad range of physical conditions. The models are generated in physical space so the luminosity of the central object is specified, and the separation between it and the cloud is specified instead of the ionization parameter.

7.21. *eden* Electron conservation

```
title Martin Gaskell's funny model
c used to test that electron density convergence is ok
c hydrogen line spectrum strongly pumped by continuum
black 4000
lumin 27.2
constant temper 5500
print faint -3
radius 15
hden 5.13
abundances all 1000
stop zone 1
set dr 0
c eden.in
```

This is mainly a test of the ability of the code to converge a model with a *very strange* electron density. The electrons are mainly contributed by heavy elements, and the gas is only slightly ionized.

Checks:

- Electron density is correct.
- Hydrogen line spectrum strongly pumped by continuum.

7.22. *fluc* Density Fluctuations

```
title Paris meeting Planetary nebula with density fluctuations
init file='c84.ini'
fluctuations 16.5 4 3
sphere
black body, T=150,000K radius = 10
black 5
luminosity total 38
radius 17
abund -1 C-3.523 N-4. O-3.222 ne-3.824 na=-10 mg-4.523 al=-10
continue si-4.523 s-4.824 ar-10 ca=-10 fe-10 ni=-10
punch overview last file='fluc.ovr'
punch results last file='fluc.rlt'
c fluc.in
```

The boundary conditions are similar to those for the Paris meeting PN (see page 493), a homogeneous grain-free PN. This model has density fluctuations, with values chosen so that the mean n_e n_p V are the same for the nebulae with and without fluctuations.

Checks:

- Check that the **`fluctuations`** command works.
- How do results compare with homogeneous Paris pn?

7.23. *globule* Globule density model

```
title test of globule command
globule initial density=5 scale depth=14
black 40,000k
abundances hii region
ionization parameter -1.5
punch overview last file='globule.ovr'
punch results last file='globule.rlt'
c globule.in
```

This model uses the **globule** command, and tests that the zoning logic works for this extreme case, and that the code is able to converge the **globule** model.

7.24. *grid0.for* check that calculations repeat

```
      program main
      logical lgOK
      character*80 line
*
*     vary density and flux of ionizing photons
*     and check that subsequent models with the same parameters
*     produce exactly the same results
*
*     grid0.ot1, grid0.ot2 are first and second output files
*
      open(12,file='grid0.ot1')
      open(13,file='grid0.ot2')
      ierr = 66
      open(ierr,file='grid0.err')
*
*     density  and U increments
      deninc = 3.
      uinc = 2.
      icldy = 11
      do n=1,2
        icldy = icldy + 1
        u = -8.
        do while( u.lt. 2.-uinc-0.01 )
          u = u + uinc
          den = -4.
          do while (den.lt. 14.-deninc-0.01)
            den = den + deninc
            call cdInit
*           call cdTalk(.false.)
*           option to redirect output
            call cdOutp( icldy )
*           option to not execute code beyond header
*           call cdNoex
            write(line,'(''title den U= '',1p,2e10.2)')den,u
            call cdRead( line , nleft )
            write(line,'(''ionizat'',f5.2)') u
            call cdRead( line , nleft )
            write(line,'(''table agn '')')
           call cdRead( line , nleft )
            if( den.lt.7. ) then
              write(line,'(''abundance orion '')')
              call cdRead( line , nleft )
            endif
            write(line,'(''iterate to convergence '')')
           call cdRead( line , nleft )
            write(line,'(''hden '',f5.2)') den
            call cdRead( line , nleft )
            write(line,'(''print last iteration '')')
            call cdRead( line , nleft )
            call cdDriv( lgOK )
            call cdErrors( ierr )
          enddo
        enddo
```

```
 enddo
end
```

This uses the code to generate two grids as a check that the same parameters produce the same results when the code is used as a subroutine. This is an important test of a core feature of the code. Two output files, **grid0.ot1** and **grid0.ot2**, are created. These should be exactly the same. Use the OS's compare command to verify this.

7.25. *hemis* Hydrogen emissivity

```
title test hydrogen atom continuous emissivity
black 5
ionization parameter -2
sphere                  ;want full outward flux
constant temperature 4
abundances all -10      ;pure hydrogen
hden 7                  ;suppress two-photon emission
* this is necessary to stop model at H ionization front
* constant temperature models will go on to infinity
stop eden 6
print last iteration
c output the predicted continuum
punch emitted continuum last iteration file='hemis.dta'
c stop collisional ionization and excitation of lines
hydrogen collisions off
iterate
c hemis.in
```

This checks that the predicted hydrogen continuum is in good agreement with exact results in the optically thin nebular limit.

Checks:

- This output was used to generate figure *hemis* in Part I of HAZY.
- Continuum relative to Hβ should agree with Ferland (1980) filter averaged results.
- Hβ should agree with Case B predictions, and Q(H) 4861.

7.26. *highn* High density limit

```
title high electron density approach to lte
c test from Ferland and Rees 88, collisions drive H to LTE
c collisions should drive all departure coef to unity
constant temperature = 50,000
print departure coefficients
stop zone 1
stop lyman optical depth -6
black body t=50,000
ionization parameter -5
hden = 19
abundances all -10
iterate
print last
set dr 0
c highn.in
```

This model is a test of the behavior of hydrogen and helium in the high density, collision dominated, limit. The temperature is preset, the hydrogen density is set to a very high value, and the ionization parameter is very low. The resulting model is collision dominated, so this case checks that the collision physics occurs in detailed balance. The predicted departure coefficients should all equal unity. The **`set dr`** command sets the zone thickness to 1 cm.

Checks:

- Hydrogen departure coefficients exactly unity.
- Helium departure coefficients near unity. (Density not high enough to bring helium departure coefficients exactly to unity.)
- H^-, H_2, $H2^+$ $H3^+$, and HeH^+ departure coefficients exactly unity.

7.27. *hiiregions.for* grid of HII region models

```
      program main
      logical ok
      parameter (nutot=9 , nstartot=5 )
      character*80 line
      hden = 1.
*     redirect output to file corner4.out
      icldy = 12
      open(icldy,file='hiiregions.out')
      ierr = 13
      open(ierr,file='hiiregions.err')
      doigrain=0,1
        donu=1,nutot
          donstar=1,nstartot
            tstar = nstar*2e4 - 1e4
            u = 2. - nu*0.5
            call cdInit
*           option to redirect output to file opened above
            call cdoutp( icldy )
*           option to not execute code beyond header
*           call cdNoex
            write(line,'(''title run code for hii regions'',
     1      1p,2e10.2,i3)') tstar,u,igrain
            call cdRead( line , nleft )
            write(line,'(''blackbody '',f7.4)')log10(tstar)
            call cdRead( line , nleft )
            write(line,'(''ionization '',f7.2)') u
            call cdRead( line , nleft )
            if(ized.eq.1 ) then
                write(line,'(''abundan orion'')')
                call cdRead( line , nleft )
              endif
              write(line,'(''iterate to convergence '')')
              call cdRead( line , nleft )
              write(line,'(''set nend 1500 '')')
              call cdRead( line , nleft )
              write(line,'(''print last iteration '')')
              call cdRead( line , nleft )
              write(line,'(''stop temperature 3 linear '')')
              call cdRead( line , nleft )
              write(line,'(''hden '',f5.2)') hden
              call cdRead( line , nleft )
              write(line,'(''failures 5 no map '')')
              call cdRead( line , nleft )
              call cdDriv( ok )
              call cdErrors(ierr)
            enddo
          enddo
        enddo
        end
```

This computes a grid of HII region models using various blackbodies, solar and dusty abundances, with a very low stopping temperature.

7.28. *hizlte.for* high Z cloud exposed to blackbody in TE

```
      program main
      include "/home16/users/gary/cloudy/phycon.com"
      character*80 line
      logical ok
      icldy = 10
      open(icldy,file='hizlte.out')
      ierr = 13
      open(ierr,file='hizlte.err')
      write(icldy,'(''         input  deduced'')')
      doi=1,8
        tein = 3.5 + float(i)/2.
        call cdInit
*       this will be a quiet run - turn off printout
        call cdTalk( .false. )
*       option to not execute code beyond header
*       call cdNoex
        write(line,'(''title high metal clouds into lte te='',
     1  F7.2)')  tein
        call cdRead( line , nleft )
        write(line,'(''blackbody lte'',f6.2)')tein
        call cdRead( line , nleft )
        write(line,'(''iterate '')')
        call cdRead( line , nleft )
        write(line,'(''abundances starburst 10 '')')
        call cdRead( line , nleft )
        write(line,'(''stop zone 1 '')')
        call cdRead( line , nleft )
        write(line,'(''hden '',f5.2)') 10.
        call cdRead( line , nleft )
        call cdDriv( ok )
        call cdErrors( ierr )
        write(icldy,'('' log(t):'',2f7.3)')tein,log10(te)
      enddo
      end
```

This is a series of calculations of a very high metallicity gas exposed to a true blackbody. The gas should equilibriate at the blackbody temperature. It is driven to this limit by interactions between the radiation field and the thousands of lines produced by the gas. This is also an example of using the code in a very quiet mode - very little output is generated.

7.29. *hizqso* Very high metallicity quasar cloud

```
title high Z quasar cloud
c hydrogen is a minor constituent in this high metallicity gas
c the electron density is well above the hydrogen density
table agn
abundances starburst 35
hden 8
ionization parameter 2
iterate
c increase limit to zones because of high ionization parameter
set nend 1000
print last iteration
punch overview last file='hizqso.ovr'
punch results last file='hizqso.rlt'
c hizqso.in
```

This is a model of a dense very high metallicity cloud. Hydrogen is a minor constituent in this gas. It is included to confirm that the code can handle such an extreme situation.

7.30. *induc* High radiation field limit

```
title constant temper black body limit from Ferland and Rees 1988
c tests whether departure coef are forced to unity by induced processes
constant temper 50,000
stop zone 1
grains -5 no heating no cooling
hden 10
stop lyman continuum optical depth -6
black body, t=50,000 lte
metals -10
print departure coef
print faint -1
print last
iterate
set dr 0
c induc.in
```

This example tests whether induced processes force level populations of hydrogen and helium to LTE when they are irradiated by a blackbody in strict thermodynamic equilibrium. The density is low enough value for radiation to dominate the rate equations coupling levels with each other and the continuum. The expectation is for all departure coefficients to equal unity. A small amount of grains are included to check that the grain thermal balance is handled properly in this radiation-dominated limit.

Checks:

- Departure coefficients exactly unity.
- Grain temperatures are *exactly* 5×10^4 K.

7.31. *ism* ISM cloud

```
title interstellar cloud irradiated by ism background
table ism
extinguish by a column of 22
cosmic rays, background
hden 0
abundances hii region no grains; this turns on Orion abundances
grains ism ;this over rides above default, forcing ism grains
sphere ;set sphere since matter in all directions
stop temperature linear 10
stop thickness 0.1 linear parsecs
case b
print last
print faint .1
punch overview last file='ism.ovr'
punch results last file='ism.rlt'
c ism.in
```

This is a test of the behavior of the code in the extreme of photoionization by a relatively hard continuum, at low densities. The continuum is the galactic background, attenuated by a column density of 10^{22} cm^{-2}. Ionization by galactic background cosmic rays is included. **`Case b`** appears since this region is deep in the ISM, and the Lyman lines are quite thick. This example checks whether the ionization balance, thermal balance, and electron density sum, are performed correctly in this limit.

Checks:

- Numerical stability of solution
- Thickness exact

7.32. *kk* Kwan and Krolik BLR model

```
title Kwan+Krolik Ap.J. 250, 478 BLR model
init file='c84.ini'
constant gas pressure
f(nu) -7.32148
abundances he-1 c-3.699 n-4 o-3.1549 ne-4 na=-8 mg-4.5229
continue al-10 si-4.4229 s-10 ar-10 ca-10 fe-4.5229 ni=-8
stop column density 23
interpolate (0 -5) (.05 -5) (.1 0) (1 -0.5) (7.353 -2.233)
continue (735 -3.233) (800 -15) (8,000,000 -15)
print last iteration
normalise 1216 100
iterate to convergence
plot continuum range -3
hden 9.60206
punch overview last file='kk.ovr'
punch results last file='kk.rlt'
c kk.in
```

This is the "standard" BLR model presented by Kwan and Krolik (1981).

Checks:

- Compare line intensities to previous versions of CLOUDY by entering into table on page 454.
- A code caution that the resulting total pressure was not constant is to be expected. The KK calculation assumed constant gas pressure, but radiation pressure was significant. Because of this the sum of gas plus radiation pressure was not constant (but gas pressure was).

7.33. *kmt.for* Cooling and pressure curve similar to KMT 1981

```
      program main
      logical ok
      parameter (nhden=37 )
      character*80 line
*     redirect output to file kmt.out
      icldy = 12
      open(icldy,file='kmt.out')
      ierr = 13
      open(ierr,file='kmt.err')
      flx = 20.
      dond=1,nhden
        hden = -1. + float(nd-1) / 2.
        call cdInit
*       option to redirect output to file opened above
        call cdOutp( icldy )
*       tell it so be quiet
        call cdTalk( .false. )
*       option to not execute code beyond header
*       call cdNoex
        write(line,'(''title run code for KMT style grid'',
     1  F7.2)') hden
        call cdRead( line , nleft )
        write(line,'(''table agn'')')
        call cdRead( line , nleft )
        write(line,'(''phi(h) '',f7.2)') flx
        call cdRead( line , nleft )
        write(line,'(''background z=4 '')')
        call cdRead( line , nleft )
        write(line,'(''stop zone 1 '')')
        call cdRead( line , nleft )
        write(line,'(''hden '',f5.2)') hden
        call cdRead( line , nleft )
        call cdDriv( ok )
        call cdErrors( ierr )
        call cdGetTe( te )
        call cdGetPres( ptot , pgas, prad)
        write(icldy,'(1p,5e11.4)') hden, te, ptot,pgas,prad
      enddo
      end
```

This is the correct way to generate the pressure law grid used to search for thermal stability. The only output generated is the list of density, electron temperature, and pressures. It produces results that can be compared with Krolik, McKee, and Tarter (1981).

7.34. *lalpha* Ly alpha forest cloud

```
title Ly alpha forest cloud
c ionized by AGN power law plus cosmic background
background, z=2 ; this includes both black body and AGN parts
double  ; mimic two-siced photoionization
hden -2
stop neutral column density 15
metals -1.5  ;reduce all heavy elements by 1.5 dex
print faint -1
iterate to convergence; must iterate since optically thin
print last
punch overview last file='lalpha.ovr'
punch results last file='lalpha.rlt'
c lalpha.in
```

This example demonstrates the behavior of the code in the low-density limit. The ionizing source is the cosmic background at a redshift of z = 2. The cloud is assumed to be optically thin to ionizing radiation, in keeping with the Gunn-Peterson test, so the **double** command is included, and an iteration is performed to converge the optical depth scale. Continuum fluorescent excitation of lines is important because the cloud is so thin.

7.35. *laser1* Hydrogen ionization

```
title test of H ionization in optically thin limit
c H cross section is 2.09E-18 cm^2, rec coef is 4.18E-14
c answer is neutral fraction 2.00E-4
laser 1.5 Ryd
phi(h) 10
hden 1
constant temperature = 4
iterate to convergence
stop zone 1
abundances all -10
set dr 0
print last
c laser1.in
```

This checks the calculation of the hydrogen photoionization equilibrium. The continuum is a laser peaked at 1.5 Ryd, where the hydrogen photoionization cross section is 2.09×10^{-18} cm^{-2}.

Checks:

- The hydrogen neutral fraction is nearly 2.00×10^{-4} (not exact since laser has finite width).
- Hβ emissivity close to high density case A. The predicted TOTL 4861 intensity should be nearly 2.2 times the expected case B intensity.

7.36. *laser2* H, He ionization

```
title test of H and HeI ionization in optically thin limit
c H cross section is 0.927E-18 cm^2, rec coef is 4.18E-13
c answer is Ho/H+ = 4.51e-4
c HeI cross section is 6.54E-18 cm^2, rec coef is 4.32E-13
c answer is Heo/He+ = 6.61e-5
laser 2.0 Ryd
phi(h) 10
hden 1
constant temperature = 4
iterate to convergence
stop zone 1
abundances all -10
set dr 0
print last
c laser2.in
```

This checks the calculation of the hydrogen and helium photoionization equilibrium. The continuum is a laser peaked at 2.0 Ryd, and so can only ionize hydrogen and atomic helium.

Checks:

- The hydrogen neutral fraction is nearly $H^o / H^+ = 4.51 \times 10^{-4}$ (not exact since laser has finite width).
- Hβ emissivity close to high density case A. The predicted TOTL 4861 intensity should be nearly 2.2 times the expected case B intensity.
- Helium ionization should be $He^o / He^+ = 6.61 \times 10^{-4}$.

7.37. *laser3* H, He ionization

```
title test of H, HeI, and HeII ionization in optically thin limit
c H cross section is 1.0E-18 cm^2, rec coef is 4.18E-13
c answer is n(Ho)/n(H+)=4.18e-3
c HeI cross section is 1.51E-18 cm^2, rec coef is 4.32e-13
c answer is n(Heo)/n(He+)=2.86e-4
c HeII cross section is 1.30E-18 cm^2, rec coef is 2.20e-12
c answer is n(He+)/n(He2+)=1.69e-3
laser 4.3 Ryd
phi(h) 10
hden 1
constant temperature = 4
iterate  to convergence
stop zone 1
abundances all -10
set dr 0
print last
c laser3.in
```

This checks the calculation of the hydrogen and helium photoionization equilibrium. The continuum is a laser peaked at 4.3 Ryd, where it can fully ionize both hydrogen and helium.

Checks:

- The hydrogen neutral fraction is nearly 4.18×10^{-4} (not exact since laser has finite width).
- Helium ion: The ratio He^{+}/He^{++} should be 1.69×10^{-3} and the ratio He^{o}/He^{+} should be 2.86×10^{-4}.
- Hβ emissivity should be close to high-density case A. The predicted TOTL 4861 intensity should be nearly 2.2 times the expected case B intensity.

7.38. *ldl* Low density limit

```
title check that equilibrium works at very low densities
hden = -1, proportional to radius to the -2
metals -10 :need this to turn off line re-emission by heavies
stop eden -5
filling factor -5
radius 16
blackbody, t=5
luminosity 38
sphere
punch overview last file='ldl.ovr'
punch results last file='ldl.rlt'
c ldl.in
```

This checks the behavior of the code at very low densities. Ionization and temperature should not change with density. This is because of r^{-2} density dependence, causing the cloud to have constant ionization parameter throughout. The solutions of the first and last zones will be similar unless underflow has affected the results.

Checks:

- Solutions of first and last zones should be very similar.
- Cray agrees with IEEE machines.

7.39. *liner* Liner model

```
title liner model
c a constant pressure (gas+radiation) model of a liner cloud
c in the spirit of Ferland and Netzer 1983
table agn
metals 0.3
constant pressure
iterate to convergence ;must iterate to get radiation pressure correctly
hden 6
stop column density 23
ionization parameter -3
print last
punch overview last file='linr.ovr'
punch results last file='linr.rlt'
c liner.in
```

This is a model somewhat like the Liner parameters proposed by Ferland and Netzer (1983). A second iteration is performed to allow the calculation of the line radiation pressure.

7.40. *lte* Lte thermal limit

```
title thermal equil black body limit from Ferland and Rees 1988
c this tests whether thermal processes go to lte
stop zone 1
hden 10
stop lyman optical depth -6
black body, t=50,000 lte
abundances all -10
print departure coef
print last
iterate
set dr 0
c lte.in
```

This is the ultimate test of the behavior of the code in the strict thermodynamic equilibrium limit. The temperature is not held constant, so the resulting equilibrium temperature determines whether cooling processes are treated properly in the detailed balance limit. The equilibrium temperature should be exactly 5×10^4 K, and all departure coefficients should equal unity. A small amount of grains are included to check that the grain thermal balance is handled properly in this limit.

Checks:

- Electron temperature exactly 5×10^4 K.
- Departure coefficients unity.

7.41. *ltemetl* high metallicity thermodynamic equilibrium

```
title thermodynamic equilibrium with metals
c this tests whether thermal processes go to lte
stop zone 1
hden 10
stop lyman optical depth -6
black body, t=20,000 lte
abundances starburst 5
print departure coef
print last
iterate
set dr 0
c ltemetl.in
```

This checks that the code goes to strict thermodynamic equilibrium for the case of a metal rich gas exposed to a true black body. The many heavy element lines should dominate cooling, so this is a test that the multilevel atoms go to LTE in the radiation-dominated limit.

Checks:

- Temperature should equilibrate at 20,000 K.
- Departure coefficients should equal unity.

7.42. *map* Map of heating and cooling

```
title map of heating vs cooling
hden 0
table agn
ionization parameter -2.5
c change plot to punch to generate plot for hazy
punch map file='map.dta'
*plot map
stop zone 1
set nmaps 1000
c map.in
```

This is a test of the continuity of the code over a very large range of temperature. It was used to produce the thermal map shown on page 426.

Checks:

- No breaks in the heating and cooling curves where various approximations change.

7.43. *matchn2* Match FOS Orion spectra

```
title find density and temperature from HST FOS spectra of [NII] in orion
constant temper 4 vary
hden 3.5 vary
stop zone 1
ionization parameter -3
black 35,000
normalize to 6584 = 89.5
optimize lines
N  2 5755 1.69
N  2 2140 2.026
end of lines
```

This is an example of using a very large cannon to knock off a very small flea. The code is asked to determine the density and temperature that would reproduce to [NII] line ratios. The **normalize** command is used to set the resulting intensity of [NII] λ6584 to 89.5 in the relative scale. On this scale, intensities of [N II] 5755 and N II] 2140 of 1.69 and 2.026 are desired. The density and temperature of a single zone model is varied to produce this observed ratio.

7.44. *n5548* Ferland et al model of NGC 5548

```
title BLR model from Ferland et al. 1992
hden = 11.0
print last iteration
ionization parameter -1.0
stop column density 26.0
iterate to convergence
table agn break 100 microns
punch unit 11, overview last file='n5548.ovr'
punch unit 12, results last file='n5548.rlt'
c n5548.in
```

This is similar to one of the models presented in Ferland et al. (1992) for the well-studied Seyfert galaxy NGC 5548.

7.45. *nlr* Lexington Meeting NLR model

```
title Hagai's nlr model for Lexington Meeting
init file='c84.ini'
diel kludge 0
hden 4
stop column 22
abundances -1 c-3.52 n-4 o-3.097 ne-4 na-9
cont mg-4.523 al-8 si-4.523 s-4.824 a-9 ca-8 fe-8 ni-9
interpolate (0 -10) (0.08 -10) (0.1 1) (3676 -4.935) (4700 -11)
continue (4750 -20) (7,400,000 -30)
phi(h) 12.47712
iterate
print faint .01
print last iteration
punch overview last file='nlr.ovr'
punch results last file='nlr.rlt'
c nlr.in
```

This is one of the test cases from the Lexington Meeting suite of nebulae. It is a grain-free NLR model.

7.46. *nova* Eta Car Nova Envelope

```
title dense nova shell
table star kurucz 35000
absolute bolometric magnitude -8.1
radius 12
hden 10
sphere
iterate to convergence
print last iteration
punch overview last file='nova.ovr'
punch results last file='nova.rlt'
c nova.in
```

This is a model illuminated by a relatively soft radiation field. It is dense and grain-free. Some interesting features:

- H^- heating dominates in outer ionized regions.
- The significant H_2^+ heating in outer ionized regions.
- The [OI] lines are optically thick and not in 3:1 ratio.
- Checks that **`table star Kurucz`** command works.

7.47. *oldblr* Old BLR model

```
title Netzer and Ferland PASP 96, 593, Table 1
abundances cameron
constant gas pressure
hden 9.5
ionization parameter -1.92
stop lyman optical 6
iterate to convergence
print last iteration
power law -1, 100 0.01
plot continuum
punch overview last file='oldblr.ovr'
punch results last file='oldblr.rlt'
c oldblr.in
```

This is an example of a “conventional” BLR calculation. The parameters are similar to those of Table 1 of Netzer and Ferland (1984). Notice that the ratio of Lyα to Hβ ratio is much larger than observed.

7.48. *optim* Optimization driver

```
title test of optimization driver
optimize iterations 30 ; increase limit to number of iterations
optimize intensity -1 0.1 ;want H-bet flux of .1 erg cm^-2 s^-1
*
optimize column density ;read in sets of desired column densities
hydrogen 2  20.8  0.1
end of column dens
*
optimize lines ; read in sets of desired relative line intensities
O  3 5007 3.1415
end of lines
*
hden 3 vary  ;the density, ionization parameter, and bb temp
black 4.7 vary
ionizat -2 vary
c optim.in
```

Vary three parameters to try to reproduce a desired Hβ intensity, ionized hydrogen column density, and 5007 / Hβ intensity ratio.

7.49. *orion* Orion Nebula

```
title conditions similar to Orion nebula blister
sphere
table star kurucz 39,600K
phi(h) 13.0
turbulence 8 km/sec
hden 4
abundances hii region
constant pressure
iterate
print last iteration
punch overview last file='orion.ovr'
punch results last file='orion.rlt'
c orion.in
```

This is a model similar in spirit to the blister geometry HII region model computed by Baldwin et al. (1991). Orion grains are turned on with the **abundances** command. The **constant pressure** command does a hydrostatic equilibrium structure. The predicted emission line spectrum is affected by the reddening of the internal grains. The resulting Peimbert analysis produces artificial results as a result.

Checks:

- Pressure convergence
- Helium neutral fraction

7.50. *orionff* Orion Nebula with filling factor

```
title Orion blister with filling factor
sphere
filling factor 0.01
table star kurucz 39,600K
phi(h) 13.0
turbulence 8 km/sec
hden 4
abundances hii region
constant pressure
iterate
print last iteration
punch overview last file='orionff.ovr'
punch results last file='orionff.rlt'
print line inward
c orionff.in
```

This is the Orion model with a filling factor. The geometry is plane parallel so the results should have absolutely no dependence on the filling factor.

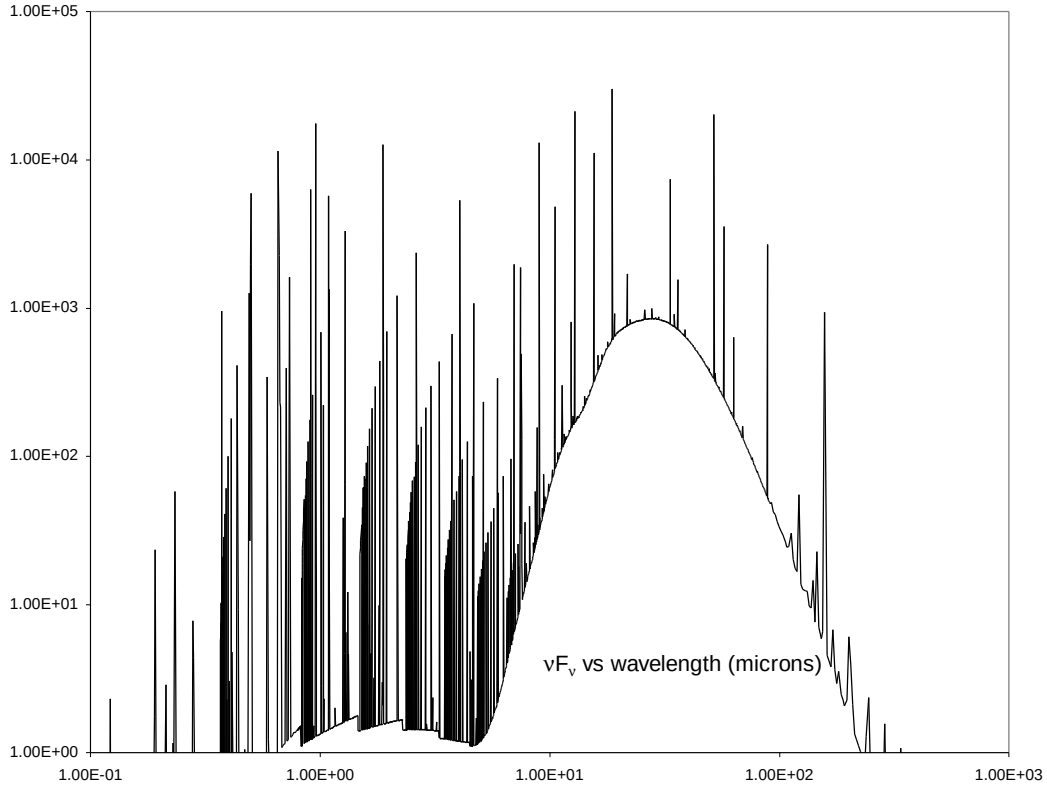
Checks:

- Depth of cloud increases by factor of the filling factor.
- Predictions are identical to Orion model.
- Inward fractions are ok NB! This is one of the few cases where this is printed!

7.51. *orionpdr* Orion Nebula going into the PDR

```
title constant density orion into pdr
stop temperature 50
hydrogen levels limit
set PunchLWidth 10 km/sec
sphere
table star kurucz 39,600K
phi(h) 13.0
turbulence 8 km/sec
hden 4
abundances hii region
brems 6
phi(h) 10
punch overview last file='orionpdr.ovr'
punch results last file='orionpdr.rlt'
punch continuum units microns last file='orionpdr.cnt'
c orionpdr.in
```

This is the Orion model with an X-ray continuum, going well into the PDR. The continuum is punched out with the energy scale in microns rather than Rydbergs. The number of levels in the hydrogen atom is set to the maximum limit to produce more lines in the plot. The intrinsic line width used to set the line to continuum contrast is set to 10 km/sec with the **`set PunchLWidth`** command. A plot of the punch output is below.



7.52. *parishii* Paris meeting HII region

```
title "New" Paris meeting HII region
init file='c84.ini'
c "standard" HII region model of the Pequignot Meudon Conferance
sphere
black body, T=40,000K radius = 12.113943
hden = 2
radius = 18.477121
abund -1 C-3.6576 N-4.39794 O-3.481146 ne-4.30103 na=-8 mg-8 al=-8
continue si-8 s-5.04576 ar-8 ca=-8 fe-8 ni=-8
plot continuum range .1
iterate   ; must iterate since fine structure lines are opticall thick
print last
punch overview last file='parishii.ovr'
punch results last file='parishii.rlt'
print line sum
TOTL 5876
C  2 2326
TOTL 1909
N  2  121
N  2 6584
N  2 6548
N  3  570
TOTL 3727
O  3 5007
O  3 4959
O  3  518
O  3  883
Ne 2  128
Ne 3  156
Ne 3 3869
Ne 3 3968
S  2 6720
S  3  187
S  3  334
S  3 9532
S  3 9069
S  4  105
end of lines
dielectronic kludge 0
c parishii.in
```

This is one of the "standard" models computed at the Paris and Lexington meetings on photoionization and shock calculations. Table 4 compares the predictions of the current version of CLOUDY with predictions of a few of the other codes. It is necessary to iterate since some fine structure lines are optically thick. The set of lines entered with the **`print line sum`** command is used to obtain the total luminosity in detected lines, a measure of the Stoy temperature.

Checks:

- Hβ close to case B, Q(H) 4861, intensities.
- Enter answers in Table **7**.

7.53. *parisnlr* Paris meeting NLR model

```
title paris meeting NLR model
init file='c84.ini'
iterate
print last iteration
interpolate (0 -10) (0.073 -10) (0.074 1) (7352 -4)
continue (7400 -15) (7,353,000 -20)
stop lyman continuum optical depth 4
hden 3
abund -1 C-3.5229 N-4. O-3.22185 ne-3.82391 na=-8 mg-4.5229 al=-8
continue si-4.5229 s-4.82391 ar-8 ca=-8 fe-7 ni-8
ionization parameter -2
punch overview last file='parisnlr.ovr'
punch results last file='parisnlr.rlt'
c parisnlr.in
```

This is the NLR model presented in the Meudon meeting on model nebulae. The **`init`** file is entered to make the code behave more like version 84.

Checks:

- **`init`** file works

7.54. *parispn* Paris meeting planetary nebula

```
title "New" Paris meeting Planetary nebula
c recompute "standard" PN model of the Pequignot Meudon Conferance
init file='c84.ini'
sphere
black body, T=150,000K radius = 10
hden = 3.4771213
radius = 17
abund -1 C-3.523 N-4. O-3.222 ne-3.824 na=-10 mg-4.523 al=-10
continue si-4.523 s-4.824 ar-10 ca=-10 fe-10 ni=-10
plot continuum range .1
punch overview last file='parispn.ovr'
punch results last file='parispn.rlt'
dielectronic kludge off
print diffuse continuum
c parispn.in
```

This is one of the “standard” models computed at the Paris meeting on photoionization and shock calculations. Table 7 compares the predictions of the current version of CLOUDY with predictions of a few of the other codes.

7.55. *pdr* Tielens and Hollenbach PDR

```
title Tielens  and Hollenbach pdr model,Table 2, paper b
init file="c84.ini"
*first continuum is FIR hot grain continuum
blackbody, t=75K
intensity 2.7 (total)
*this is hot star continuum
black 30,000
*intensity  2.204 range  0.38 to 0.912 Ryd
nuf(nu) = 2.322 at 0.651 ryd
*this will remove all ionizing radiation
extinguish 24 0
hden 5.362
grains orion
abundances he -1.01 c -3.52 n-8 o-3.30 ne-8 na-8 mg-5.89
continue al-8 si -6.10 s -5.10 ar-8 ca -8 fe -6.60 ni -8
turbulence 2.7 km/sec
normalize to 157
sphere
case b
cosmic rays, background
stop temperature 10 linear
punch overview last file='pdr.ovr'
punch results last file='pdr.rlt'
c pdr.in
```

This is the Tielens and Hollenbach (1985a, b) standard model of the Orion photodissociation region (PDR). The **`case b`** command appears since the Lyman lines are vastly optically thick in this environment. If the command is removed the Lyman lines will be optically thin, which will result in fast fluorescent excitation of the (nearly totally neutral) hydrogen atom.

7.56. *pnots* OTS version of the Paris Planetary Nebula

```
title Paris meeting Planetary nebula with ots
c recompute "standard" PN model of the Pequignot Meudon Conferance
diffuse ots
init file='c84.ini'
sphere
black body, T=150,000K radius = 10
hden = 3.4771213
radius = 17
abund -1 C-3.523 N-4. O-3.222 ne-3.824 na=-10 mg-4.523 al=-10
continue si-4.523 s-4.824 ar-10 ca=-10 fe-10 ni=-10
plot continuum range .1
punch overview last file='pnots.ovr'
punch results last file='pnots.rlt'
c pnots.in
```

This is the on-the-spot version of the Paris Planetary Nebula.

Checks:

- Q(H) total 4861 luminosity is close to expected value
- Line spectrum similar to that predicted by default conditions.

7.57. *primal* primordial cloud

```
title primordial abundances
background 10
table agn
ionization parameter -2
abundances primordial
hden 7
punch overview last file='primal.ovr'
punch results last file='primal.rlt'
iterate to convergence
print last iteration
c primal.in
```

This is a high redshift cloud irradiated by the cosmic background and AGN light.

7.58. *rauch* model of very hot planetary nebula

```
title very hot PN model
set path="/home16/users/gary/cloudy/"
table star rauch T=250,000K g=7.5
luminosity 4.5 solar
sphere
radius 13.5
hden 3.0
print last iteration
iterate
abundances planetary
plot continuum range .1
punch overview last file='rauch.ovr'
punch results last file='rauch.rlt'
c rauch.in
```

This is a model of a very hot planetary nebula, and checks that the code is able to access the Rauch grid of stellar atmosphere models.

7.59. *reflector* Compton reflector model

```
title model of Compton reflector
power law -1. hi=1000000000
hden 11
ionization parameter 1.
stop total column density 25
constant temper 5
plot reflected continuum range 1 100,000 Ryd
print faint -1
punch continuum last file='rflect.dta'
set PunchLWidth 100 km/sec
```

This is a model of the Compton reflector in AGN. It is a constant temperature since models of this region often make that assumption. A plot in part I shows the incident and reflected portions of the continuum. The code will complain that the cloud is Compton thick since it is not really designed to simulate this situation.

7.60. *rnfa* Rees, Netzer, and Ferland Table 1

```
title table 1 of Rees et al. ApJ 347, 648
init file='c84.ini'
abundances he-1, c-3.328 n-4.0088 o-3.0809 ne-4 na=-20
continue mg -4.3768 al-5.5686 si-4.36653 s-4.76955
continue a-5.4202 ca-5.6383 fe-4.4815 ni=-20
hden 10
ionization parameter -2
stop column density 23
iterate to convergence
table agn
```

```
print last iteration
punch overview last file='rnfa.ovr'
punch results last file='rnfa.rlt'
normalise 1216
c rnfa.in
```

This is the lower density cloud computed in Rees et al. (1989). Table 1 of that paper lists the predictions, which were a mean of those of Hagai Netzer's ION and roughly version 76 of CLOUDY. The lines are generally still in good agreement with the predictions of that paper. In particular the changes in the line fluxes shown in Figure 1 of that paper are reproduced quite well.

7.61. *rnfb* Rees, Netzer, and Ferland Table 1

```
title table 1 of Rees et al. ApJ 347, 648
init file='c84.ini'
abundances he-1, c-3.328 n-4.0088 o-3.0809 ne-4 na=-20
continue mg -4.3768 al-5.5686 si-4.36653 s-4.76955
continue a-5.4202 ca-5.6383 fe-4.4815 ni=-20
hden 12
ionization parameter -2
stop column density 23
iterate to convergence
table agn
print last iteration
punch overview last file='rnfb.ovr'
punch results last file='rnfb.rlt'
normalise 1216
c rnfb.in
```

This is a very dense cloud, and was computed in Rees et al. (1989). Table 1 of that paper lists the predictions, which were a mean of those of Hagai Netzer's ION and roughly version 76 of CLOUDY. The lines are generally still in good agreement with the predictions of that paper. In particular the changes in the line fluxes shown in Figure 1 of that paper are reproduced quite well. The fluxes of Lyα and Hβ are *not* reproduced with great precision by this model because of changes in collision rates for hydrogen and especially the form of the escape probability function for subordinate lines. As Figure 1 of RNF showed the line intensities are very sensitive to density for these parameters.

7.62. *secndary* Ionization by suprathermal electrons

```
title ionization by suprathermal electrons only
set csupra -5
black 50,000
ionization paramete -10
hden 5
stop zone 1
constant temperature 5,000K
c secndary.in
```

This is a cloud with a very weak incident continuum but a component of suprathermal secondary electrons added with the **`set csupra`** command. This tests whether the code correctly handles the secondary ionization limit.

7.63. *sqrden* Density falls as inverse square

```
title test with density falling as R^-2, and filling factor
hden 5 -2
filling factor -2
blackbody, t=5, luminosity=38
abundances planetary
radius 16 18
sphere
c sqrden.in
```

Checks:

- Zone thickness budgeting handled OK.

7.64. *strom* Pure H Stromgren sphere

```
title check pure hydrogen Stromgren sphere
c the answer is R(Stromgren) = 4.16E17 cm
hden 4
sphere static
radius 16
blackbody 50,000k
q(h) 49
stop eden 3.
constant temper 7500
abundances all -10
print last iteration
iterate
punch overview last file='strom.ovr'
punch results last file='strom.rlt'
c strom.in
```

This case checks that the code computes the geometry and emissivity correctly for a pure hydrogen spherical shell. The low temperature is chosen to avoid collisional ionization. The model stops at the H^o-H^+ ionization front.

Checks

- Outer radius should be 4.16×10^{17} cm.
- Predicted Hβ, case B Hβ, and Q(H) Hβ, all agree.

7.65. *varyN.for* forbidden line ratios at various densities

```
      program main
      logical ok
      character*80 line
      logical ok
*
*     vary density at constant temperature
*
      hden = -1.
      dowhile( hden.lt.8. )
        hden = hden + 1
        call cdInit
        call cdTalk(.false.)
*       option to redirect output
        icldy = 10
        call cdOutp( icldy )
        write(line,'(''table agn '')')
        call cdRead( line , nleft )
        write(line,'(''ioniz '',f7.2)') -2.
        call cdRead( line , nleft )
        write(line,'(''constant temper 4 '')')
        call cdRead( line , nleft )
        write(line,'(''stop zone 1 '')')
        call cdRead( line , nleft )
        write(line,'(''hden '',f5.2)') hden
        call cdRead( line , nleft )
        call cdDriv( ok )
        call cdLine('S II', 6731 , rel6731,abs6731,ok)
        call cdLine('S II', 6716 , rel6716,abs6716,ok)
        call cdLine('Ar 4', 4740 , rel4740,abs4740,ok)
        call cdLine('Ar 4', 4711 , rel4711,abs4711,ok)
        write(20,'(1p,10e11.3)')hden, rel6731/max(1e-25,rel6716),
     1  rel4740/max(1e-25,rel4711)
      enddo
      end
```

This is an example of calling the code at various densities to obtain line ratios for diagnositc ions. It was originally designed to check the behavior of the five level atoms used for [OII] and [SII], although it could be used for any species. The **`constant temperature`** command is used to set the temperature.

7.66. *varyNU.for* exercies the code over wide N, U

```
      program main
      logical ok
      character*80 line
*
*     vary density and ionization parameter
      icldy = 83
      ierr = 82
     open(icldy,file='varyNU.out')
     open(ierr,file='varyNU.err')
     write(icldy,'(''   density  log(U)    H-beta    5007      1549'')')
     1)
      uinc = 1.
      hdinc = -2.
      hden = 16.
*     density loop
      do while( hden.gt.-4. )
        hden = hden + hdinc
        u = -6.
*      ionization parameter loop
        do while( u.le.0. )
        u = u + uinc
          call cdInit
*         produce no output at all
          call cdTalk( .false.)
*         option to not execute code beyond header
*         call cdNoex
          write(line,'(''ioniz '',f7.2)') u
          call cdRead( line , nleft )
          write(line,'(''iterate to convergence '')')
          call cdRead( line , nleft )
          write(line,'(''print last iteration  '')')
          call cdRead( line , nleft )
        write(line,'(''stop column density 23 '')')
          call cdRead( line , nleft )
          write(line,'(''table agn '')')
          call cdRead( line , nleft )
          write(line,'(''hden '',f5.2)') hden
          call cdRead( line , nleft )
          call cdDriv( ok )
          call cdErrors( ierr )
          call cdLine('O  3', 5007 , rel5007,abs5007,ok)
          call cdLine('TOTL', 1549 , rel1549,abs1549,ok)
          call cdLine('TOTL', 4861 , rel4861,abs4861,ok)
          call cdLine('Inci', 4860 , rel4860,abs4860,ok)
          write(icldy,'(1p,9e10.2)') hden,u,rel4861/rel4860*4861,
     1    rel5007*4861./rel4860, rel1549*4861./rel4860
        enddo
      enddo
      end
```

This tests the code over its full density range for a 32-bit CPU, and over the range of ionization parameter likely to produce significant optical line emission. The program checks whether warnings or cautions were generated, and prints them if they were. Finally a table giving the equivalent widths of Hβ, [OIII] 5007, and CIV 1549 is produced.

7.67. *vbhum* Van Blerkom and Hummer structure

```
title test against Van Blerkom and Hummer, fig 4
diffuse ots
c test hydrogen ground state rec effic against vb+h exact results
c this is their case e) - "zero condition"
c their answer for H0/Htot at the illuminated edge is approx 5.8E-4,
c and a Stromgren radius of approximately 7.7E16 cm
```

```
hden 4
blackbody 50,000k
phi(h) 12.30103
stop eden 3.5
constant temper 4
abundances all -10
print last iteration
iterate
c vbhum.in
```

This is a test of the treatment of the diffuse fields, their transfer, and their effects on the ionization structure of a nebula. The comparison is made against the exact calculation published by Van Blerkom and Hummer (1967). The geometry is open, that is, similar to that assumed in most BLR calculations.

The **`diffuse ots`** command is entered in order to reproduce the Van Blerkom and Hummer results. The default assumption, outward only, does not agree as well. I changed the default from OTS to outward only to be in better agreement with predictions by Harrington and Rubin at the Lexington meeting. They have not checked whether their codes are in agreement with the Van Blerkom and Hummer paper.

Checks:

- Neutral fraction at illuminated face 5.8×10^{-4}.
- Location of ionization front at 7.8×10^{16} cm.
- "TOTL 4861" and "CA B 4861" agree; both slightly lower than "Q(H) 4861".
- Answers with OTS agree with 1967 results.

7.68. *werner* Werner Hot Star Model

```
title test run with Werner stellar atmosphere
set path="/home16/users/gary/cloudy"
*changed 1997 may 13 so that it interpolates to get continuum
table star werner 190,000 g=7.5
hden 4
luminosity total 38
radius 17
abundances planetary
sphere
plot continuum range 0.1
punch overview last file='werner.ovr'
punch results last file='werner.rlt'
c werner.in
```

This checks that the code can access Kevin Volk's Werner atmospheres.

Checks:

- **`table star Werner`** atmosphere works.

7.69. *wind* Highly Ionized Wind

```
title test of equations of motion in a wind
c test of wind code
c radiative acceleration (e- only) is 9.54E-7 cm s^-2
c terminal velocity (e- only) is 7.6 km s^-1
hden 4
table agn
luminosiy (total) 45
radius (parsecs) 1
stop thickness (parsecs) -1
no radiation pressure
wind 0.1
constant temperature 8
c wind.in
```

This tests the management of the radiative acceleration of an electron scattering wind, and the resulting velocity. The parameters were chosen so that electron scattering is the dominant opacity source, so that the equations can be solved both numerically (in the example) and analytically (the expected solution given above). In a realistic wind the gas would be more neutral and line driving would dominate. The actual acceleration is slightly below that due to Thomson scattering alone because the gas opacity at high energies is slightly below Thomson.

Checks:

- The radiative acceleration is correct (e- 9.54×10^{-7} cm s^{-2}).
- The terminal velocity should be 7.57 km s^{-1}.
- Force multiplier near unity (no line driving since so highly ionized).
- Thickness of cloud correct (R-Ro + dr/2 should be 3.086×10^{17} cm).

8. REFERENCES

Abbott, D. C., 1982, ApJ 259, 282
Adams, T., 1972, ApJ 174, 439
Aldrovandi, S., & Pequignot, D., 1972, A&A 17, 88
Aldrovandi, S., & Pequignot, D., 1974, Revista Brasileira de Fisica, 4, 491
Ali, B., Blum, R.D., Bumgardner, T.E., Cranmer, S.R., Ferland, G.J., Haefner, R.I., & Tiede, G.P., 1991, Publ. A.S.P. 103, 1182
Allen, C.W., 1976, *Astrophysical Quantities*, Third Edition (London: Athlone Press).
Aller, L.H., 1984, in *Physics of Thermal Gaseous Nebulae*, (Reidel: Dordrecht).
Aller, L.H., & Czyzak, S.J. 1983, ApJ Sup. 51, 211.
Arimoto, N., & Yoshii, Y., 1987, A&A 173, 23
Arnaud, M., & Raymond, J., 1992, ApJ 398, 394
Arnaud, M., & Rothenflug, R., 1985, A&AS 60, 425
Avni, Y., & Tananbaum, H., 1986, ApJ 305, 83
Avni, Y., Worrall, D. M., & Morgan, W. A., ApJ 1995, 454, 673
Avrett, E.H., & Loeser, R., 1988, ApJ 331, 211
Bajtlik, S., Duncan, R.C., & Ostriker, J.P. 1988, ApJ 327, 570
Balbus, S.A., & McKee, C.F., 1982, ApJ 252, 529
Baldwin, J., Ferland, G.J., Martin, P.G., Corbin, M., Cota, S., Peterson, B.M., & Slettebak, A., 1991, ApJ 374, 580
Baldwin J.A., Ferland, G.J., Korista, K.T., Carswell, R., Hamann, F., Phillips, M., Verner, D., Wilkes, B., & Williams, R.E., 1996, ApJ 461, 683
Baldwin, J. A., Ferland, G.J., Korista K. T., and Verner, D., 1995ApJ 455, L119
Baldwin, J., Wampler, J., and Gaskell, C.M., 1989, ApJ 338, 630
Bässgen, G., Bässgen, M., & Grewing, M., 1988, Ast.Ap. 200, 51
Bates, D.R., Kingston, A. E., & McWhirter, R.W.P., 1962, Proc. R. Soc., A267, 297
Bechtold, J., Weymann, R.J., Lin, Z., & Malkan, M. A., 1987, ApJ 315, 180
Bell, K.L., Kingston, A. E., & McIlveen, W. A., 1975, J. Phys. B 8, 358
Berger, M.J., & Seltzer, S.M., 1965, NASA SP-3012
Bergeron, J., & Collin-Souffrin, S., 1971, A&A 14, 167
Berrington, K., and Pelan, A., 1995, A&AS 114, 367
Bethe, H., 1930, Ann. Phys. 5, 325
Bica, E., 1988, A&A 195, 76
Bieniek, R.J., & Dalgarno, A., 1979, ApJ 228, 635
Binette, L., Prieto, A., Szuszkiewicz, E., & Zheng, W., 1989, ApJ 343, 135
Black, J.H., 1978, ApJ 222, 125
Black, J.H., 1987, in *Interstellar Processes*, Hollenbach & Thronson, editors
Bohm, D., & Aller, L.H., 1947, ApJ 105, 131
Bonihala, J. R. M., Ferch, R., Salpeter, E. E., Slater, G., & Noerdlinger, P., 1979, ApJ 233, 649
Borkowski, K.J., & Harrington, J.P., 1991, ApJ 379, 168
Boyd, R., & Ferland, G.J., 1987, ApJ 318, L21
Bowen, I. S., 1960, ApJ 132, 1
Bregman, J.D., Allamandola, L.J., Tielens, A.G.G.M., Geballe, T.R., and Witteborn, F.C., 1989, ApJ 344, 791
Broad, J.T., & Reinhardt, W. P., 1976, Phys Rev A 14, 2159
Brown, R.L., & Mathews, W.G., 1970, ApJ 160, 939
Burgess, A., 1965, ApJ 141, 1588
Burgess, A., & Summers, H.P., 1969, ApJ 157, 1007
Burgess, A., & Summers, H.P., 1976, MNRAS 174, 345
Butler, S. E., Bender, C.F., & Dalgarno, A., 1979, ApJ 230, L59
Butler, S. E., & Dalgarno, A., 1979, ApJ 234, 765
Butler, S. E., Heil, T. G., & Dalgarno, A., 1980, ApJ 241, 442
Butler, S. E., & Dalgarno, A., 1980, ApJ 241, 838
Callaway, J. 1994, At Dat Nuc Dat Tab 57, 9

Cameron, A.G.W., 1982, in *Essays in Nuclear Astrophysics*, ed CA Barnes, DD Clayton, & DN Schramm, (Cambridge Univ Press; London)
Canfield, R.C., & Puetter, R.C., 1980, ApJ 236, L7
Cardelli, J. A., 1994, Science 264, 209
Cardelli, J. A., et al. 1991, ApJ 377, L57
Carswell R.F. and Ferland, G.J. 1988, MNRAS 235, 1121
Castor, J.I., 1970, MNRAS 149, 111
Chaffee, F. H., & White, R. E., 1982, ApJ Sup. 50, 169
Chan, E.S., Avrett, E.H., & Loeser, R., 1991, A&A 247, 580
Chapman, R. D., & Henry, R.J.W., 1971, ApJ 168, 169
Chidichimo, M.C., 1981, J. Phys. B. 14, 4149
Clavel, J., & Santos-Lleo, M., 1990, A&A 230, 3
Clegg, R.E.S., 1987, MNRAS 229, 31p
Clegg, R.E.S., & Harrington, J.P., 1989, MNRAS 239, 869
Cohen, E.R., and Taylor, B.N., 1987, Rev Mod Phys 57, 1121
Cota, S.A., 1987, Ph.D. Thesis, OSU
Cota, S.A., & Ferland, G.J., 1988, ApJ, 326, 889
Cowie, L.L., & Songaila, A., 1986, ARAA 24, 499
CrinkLaw, G., Federman, S. R., & Joseph, C. L., 1994, Ap J 424, 748
Crosas, M., & Weisheit, J.C., 1993, MNRAS 262, 359
Cruddace, R., Paresce, F., Bowyer, S., & Lampton, M., 1974, ApJ 187, 497
Dalgarno, A., & Kingston, A. E., 1963, Observatory 83, 39
Dalgarno, A., & McCray, R.A., 1973, ApJ 181, 95
Dalgarno, A., & Roberge, W.G., 1979, ApJ 233, L25
Davidson, K., 1972, ApJ 171, 213
Davidson, K., 1975, ApJ 195, 285
Davidson, K., 1977, ApJ 218, 20
Davidson, K., & Netzer, H., 1979, Rep. Prog. in Physics 51, 715
Davidson, K., & Fesen, R.A., 1985, Ann. Rev. A&A 23, 119
Desert, F.-X., Boulanger, F., and Puget, J.L., 1990, A&A 237, 215
Dove, J.E., Rush, A., Cribb, P., & Martin, P.G., 1987, ApJ 318, 379
Dove, J.E., & Mandy, M.E., 1986, ApJ, 311, L93
Draine, B.T., 1978, ApJS 36, 595
Draine, B.T., & Lee, H.M., 1984, ApJ 285, 89
Draine, B.T., & Salpeter, E. E., 1979, ApJ 231, 77
Draine, B.T., & Sultin, B., 1987, ApJ 320, 803
Drake, S.A., & Ulrich, R.K., 1980, ApJS 42, 351
Elitzur, M., 1982, Rev. Mod. Phys 54, 1125
Elitzur, M., 1984, ApJ 280, 653
Elitzur, M, 1992, *Astronomical Masers*, Kluwer, Dordrecht
Elitzur, M., Ferland, G.J., Mathews, W.G., & Shields, G., 1983, ApJ 272, L55
Elitzur, M., & Ferland, G.J., 1986, ApJ 305, 35
Elvis, M. et al. 1994, ApJS 95, 1
Fabian, A. C., Pringle, J.E., & Rees M.J., 1976, MNRAS 175, 43
Federman, S. R., et al. 1993, ApJ 413, L51
Ferguson, J., Ferland, G.J., & A. K. Pradhan, 1995, ApJ 438, L55
Ferguson, J., & Ferland, G.J., 1997, ApJ 479 363
Ferguson, J.W., Korista, K.T., Baldwin, J.A., & Ferland, G.J., 1997, ApJ 487, 122
Ferland, G.J., 1977, ApJ 212, L21
Ferland, G.J., 1979, MNRAS 188, 669
Ferland, G.J., 1980a, MNRAS 191, 243
Ferland, G.J., 1980b, BAAS 12, 853
Ferland, G.J., 1980c, PASP 92, 596
Ferland, G.J., 1986, PASP 98, 549
Ferland, G.J., 1986, ApJ, 310, L67
Ferland, G.J., 1992, ApJ 389, L63
Ferland, G.J., 1993, ApJS 88, 49

Ferland, G.J., Baldwin J. A., Korista, K.T., Hamann, F., Carswell, R., Phillips, M., Wilkes, B., & Williams, R. E., 1996, ApJ 461, 683
Ferland, G., Binette, L., Contini, M., Harrington, J., Kallman, T., Netzer, H., Pequignot, D., Raymond, J., Rubin, R., Shields, G., Sutherland, R., & Viegas, S., 1995, in *The Analysis of Emission Lines*, Space Telescope Science institute Symposium Series, R. Williams & M. Livio, editors (Cambridge University Press)
Ferland, G.J., & Elitzur, M., 1984, ApJ 285, L11
Ferland, G.J., Fabian, A. C., & Johnstone, R.M., 1994, MNRAS 266, 399
Ferland, G.J. Korista, K.T. and Peterson, B.M., 1990, ApJ 363, L21
Ferland, G.J., Korista, K.T., Verner, D. A., and Dalgarno, A., 1997, ApJ 481, L115
Ferland, G.J., Lambert, D.L., Netzer, H., Hall, D. N. B., & Ridgway, S.T., 1979a, ApJ 227, 489
Ferland, G.J., Lambert, D.L., Slovak, M., Shields, G. A., & McCall, M., 1982, ApJ 260, 794
Ferland, G.J., & Mushotzky, R.F., 1982, ApJ 262, 564
Ferland, G.J., & Mushotzky, R.F., 1984, ApJ 286, 42
Ferland, G.J., & Netzer, H., 1979, ApJ 229, 274
Ferland, G.J., & Netzer, H., 1983, ApJ 264, 105
Ferland, G.J., Netzer, H., & Shields, G. A., 1979, ApJ 232, 382
Ferland, G.J., Peterson, B.M., Horne, K., Welsh, W.F., & Nahar, S.N., 1992, ApJ 387, 95
Ferland, G.J. & Persson, S. E., 1989, ApJ 347, 656
Ferland, G.J. & Rees, M.J., 1988, ApJ 332, 141
Ferland, G.J., & Shields, G. A., 1978, ApJ 226, 172
Ferland, G.J., & Shields, G. A., 1985, in *Astrophysics of Active Galaxies & Quasi-stellar Objects*, J.S. Miller, Ed.
Ferland, G.J., & Truran, J.W., 1981, ApJ 244, 1022
Ferland, G.J., Williams, R. E., Lambert, D. L., Shields, G. A., Slovak, M., Gondhalekar, P.M., & Truran, J.W., 1984, ApJ 281, 194
Field, G. B., 1965, ApJ 142, 431
Francis, P.J. 1993, ApJ 407, 519
Gaetz, T.J., & Salpeter, E. E., 1983, ApJS 52, 155
Garstang, R.H., 1958, MNRAS, 118, 57
Gavrila, M., 1967, Phys Rev 163, 147, also JILA Report #86, Sept 19, 1966
Ginzburg, V. I., & Syrovatskii, S.I., 1964, *The Origin of Cosmic Rays*, (Oxford: Pergamon)
Gould, R. S., 1978, ApJ 219, 250
Gredel, R., Lepp, S., & Dalgarno, A., 1987, ApJ 323, L137
Gredel, R., Lepp, S., Dalgarno, A., & Herbst, E., 1989, ApJ 347, 289
Greenhouse, M., et al. 1993, ApJS 88, 23
Grevesse, N., & Anders, E., 1989, *Cosmic Abundances of Matter*, AIP Conference Proceedings 183, p. 1, Ed. C. J. Waddington, (New York: AIP)
Grevesse, N. & Noels, A. 1993 in Origin & Evolution of the Elements, ed. N. Prantzos, E. Vangioni-Flam, & M. Casse (Cambridge Univ. Press, p. 15)
Guhathakurta, P., and Draine, B.T., 1989, ApJ 345, 230
Guilbert, P.W., 1986, MNRAS 218, 171
Guilbert, P., & Rees, M.J., 1988, MNRAS 233, 475
Habing, H.J., 1968, Bull. Astr. Inst. Netherlands 19, 421
Halpern, J. P., & Grindlay, J.E., 1980, ApJ 242, 1041
Hamann, F., & Ferland, G., 1992, ApJ391, L53
Hamann, F., & Ferland, G.J., 1993, ApJ 418, 11
Harrington, J. P., 1969, Ap J 156, 903
Harrington, J. P., 1973, MNRAS 162, 43
Heitler, W., 1954, *The Quantum Theory of Radiation* (Oxford: Oxford University Press)
Hjellming, R.M., 1966, ApJ 143, 420
Hollenbach, D., & McKee, C.F., 1979, ApJS 41, 555
Hollenbach, D., & McKee, C.F., 1989, ApJ 342, 306
Hubbard, E.N., & Puetter, R.C., 1985, ApJ 290, 394
Hummer, D.G., 1962, MNRAS 125, 21
Hummer, D.G., 1968, MNRAS 138, 73
Hummer, D.G., 1988, ApJ 327, 477

Hummer, D. G, Berrington, K. A., Eissner, W., Pradhan, A. K., Saraph H. E., Tully, J. A., 1993, A&A, 279, 298
Hummer, D.G., & Kunasz, 1980, ApJ 236, 609
Hummer, D.G., & Seaton, M.J., 1963, MNRAS 125, 437
Hummer, D.G., & Storey, P., 1987, MNRAS 224, 801
Hutchings, J.B., 1976, ApJ 205, 103
Ikeuchi, S., & Ostriker, J.P., 1986, ApJ 301, 522
Jackson, J. D., 1975, *Classical Electrodynamics* (New York: Wiley)
Janev, R.K., Langer, W. D., Post, D. E., & Evans, K., 1987, *Elementary Processes in Hydrogen–Helium Plasmas* (Springer--Verlag; Berlin)
Jenkins, E.B., 1987, in *Interstellar Processes*, D. Hollenbach & H. Thronson, Eds, (Dordrecht: Reidel), p.533
Johnson, L.C., 1972, ApJ 174, 227
Johnstone, R.M., Fabian, A. C., Edge, A. C., & Thomas, P.A., 1992, MNRAS 255, 431
Kallman, T.R., & McCray, R., 1982, ApJS 50, 263
Karzas, W.J., & Latter, R., 1961, ApJS 6, 167
Kaastra, J.S., & Mewe, R., 1993, A&AS 97, 443
Kato, T., 1976, ApJS 30, 397
Kellerman, K.I., 1966, ApJ 146, 621
Khromov, G.S., 1989, Space Science Reviews 51, 339
Kingdon, J.B., & Ferland, G.J., 1991, PASP 103, 752
Kingdon, J.B., & Ferland, G. J., 1993, ApJ, 403, 211
Kingdon, J.B., Ferland, G. J., & Feibelman, W.A., 1995, ApJ, 439, 793
Kingdon, J.B., & Ferland, G.J., 1995, ApJ 450, 691
Kingdon, J.B., & Ferland, G.J., 1996, ApJS 106, 205
Korista, K.T., & Ferland, G.J., 1989, ApJ 343, 678
Korista, K.T., Baldwin, J., and Ferland, G.J., 1997, ApJ in press
Krolik, J., McKee, C.M., & Tarter, C.B., 1981, ApJ 249, 422
Kurucz, R.L., 1970, SAO Special Reports 309
Kurucz, R.L., 1979, ApJS 40, 1
Kurucz, R.L., 1991, in *Proceedings of the Workshop on Precision Photometry: Astrophysics of the Galaxy*, A.C. Davis Philip, A.R. Upgren, & K.A. James (Davis, Schenectady), p 27
Kwan, J., & Krolik, J., 1981, ApJ 250, 478
Lambert, D.L., & Pagel, B.E.J., 1968, MNRAS 141, 299
La Franca, Franceshini, A., Cristiani, S., & Vio, R., 1995, A&A 299, 19
Lame N.J., and Ferland, G.J., 1991 ApJ 367, 208
Landini, M., & Monsignori Fossi, B., 1990, A&AS 82, 229
Landini, M., & Monsignori Fossi, B., 1991, A&AS 91, 183
Lanzafame, A. ., Tully, J. A., Berrington, K. A., Dufton, P. L., Byrne, P. B., & Burgess, A., 1993, MNRAS 264, 402
Laor, A., & Draine, B.T., 1993, ApJ 402, 441
Latter, W.B., & Black, J.H., 1991, ApJ 372, 161
Lea, S., & Holman, G., 1978, ApJ 222, 29
Lennon, D. J., & Burke, V.M., 1991, MNRAS 251, 628
Lenzuni, P., Chernoff, D.F., & Salpeter, E.E., 1991, ApJS 76, 759
Levich, E.V., & Sunyaev, R.A., 1970, Astrophy Let 7, 69
Lepp, S., & Shull, J.M., 1983, ApJ 270, 578
Lightman, A. P., & White, T.R., 1988, ApJ 335, 57
Lites, B.W., & Mihalas, D., 1984, Solar Physics 93, 23
Liu, X.-W., Storey, P.J., Barlow, M.J., & Clegg, R.E.S., 1995, MNRAS, 272, 369
Longair, M. S., 1981, *High Energy Astrophysics*, (Cambridge, Cambridge University Press)
Lotz, W., 1967, ApJS 14, 207
MacAlpine, G. M., 1971, ApJ 175, 11
Maguire, S., 1993, *Writing Solid Code*, Microsoft Press
Mallik, D.C.V., & Peimbert, M., 1988, Rev Mexicana 16, 111
Martin, P.G., 1979, *Cosmic Dust* (Oxford, Clarendon Press)
Martin, P.G., 1988, ApJS 66, 125

Martin, P.G., & Ferland, G.J., 1980, ApJ 235, L125
Martin, P.G., & Whittet, D.C.B., 1990, ApJ 357, 113
Masters, A. R., Pringle, J. E., Fabian, A. C., & Rees, M. J., 1977, MNRAS 178, 501
Mathews, W.G., Blumenthal, G. R., & Grandi, S.A., 1980, ApJ 235, 971
Mathews, W.G., & Ferland, G.J., 1987, ApJ 323, 456
Mathis, J. S., 1982, ApJ 261, 195
Mathis, J. S., 1985, ApJ 291, 247
Mathis, J. S. Rumpl, W., & Nordsieck, K.H., 1977, ApJ 217, 425
Mathis, J. S., & Wallenhorst, S.G., 1981 ApJ 244, 483
Matteucci, F., & Tornambe, A., 1987, A&A 185, 51
Matteucci, F., & Greggio, A., 1986, A&A 154, 279
Mendoza, C., 1983, in *Planetary Nebulae*, IAU Sym 103, D. R. Flower, Ed., p 143, (Dordrecht: Reidel)
Mihalas, D., 1972, *Non-LTE Model Atmospheres for B & O Stars*, NCAR-TN/STR-76
Mihalas, D., 1978, *Stellar Atmospheres*, 2nd Edition (W.H. Freeman: San Francisco)
Mihalszki, J. S., & Ferland, G. J., 1983, PASP 95, 284
Morrison, R., & McCammon, D., 1983, ApJ 270, 119
Morton, D. C., York, D. G., & Jenkins, E.B., 1988, ApJS 68, 449
Nahar, S.N., & Pradhan, A. K., 1992, ApJ 397, 729
Netzer, H., 1990, in *Active Galactic Nuclei, Saas-Fee Advanced Course 20*, Courvorsier, T.J.-L., & Mayor, M., (Springer-Verlag; Berlin)
Netzer, H., Elitzur, M., & Ferland, G.J., 1985, ApJ 299, 752
Netzer, H., & Ferland, G.J., 1984, PASP 96, 593
Neufeld, D. A., 1989, Harvard Research Exam
Neufeld, D. A., & Dalgarno, A., 1989, Phys Rev A, 35, 3142
Nussbaumer, H., & Storey, P.J. 1983, A&A 126, 75
Nussbaumer, H., & Storey, P.J. 1984, A&AS 56, 293
Nussbaumer, H., & Storey, P.J. 1986, A&AS 64, 545
Nussbaumer, H., & Storey, P.J. 1987, A&AS 69, 123
Oliveira, S., & Maciel, W.J., 1986, Ap&SS 126, 211
Oliva, E., Pasquali, A., and Reconditi, M., 1996, A&A 305, 210
Osterbrock, D. E., 1951, ApJ 114, 469
Osterbrock, D. E., 1989, *Astrophysics of Gaseous Nebulae & Active Galactic Nuclei*, University Science Press
Osterbrock, D. E., & Flather, E., 1959, ApJ 129, 26
Osterbrock, D. E., Tran, H.D., & Veilleux, S., 1992, ApJ 389, 305
Ostriker, J. P., & Ikeuchi, S., 1983, ApJ 268, L63
Pacholczyk, A.G., 1970, *Radio Astrophysics* (San Francisco: Freeman)
Palla, F., Salpeter, E. E., & Stahler, S. W., 1983, ApJ 271, 632
Peebles, P.J.E., 1971, *Physical Cosmology*, (Princeton U. Press; Princeton)
Peimbert, M., ApJ 150, 825
Pengelly, R.M., 1964, MNRAS 127, 145
Pequignot, D., 1986, *Wordshop on Model Nebulae*, (Paris: l'Observatoire de Paris) p363
Pequignot, D., & Aldrovandi, S.M.V., 1986, A&A 161, 169
Pequignot, D., Petitjean, P., and Boisson, C., 1991, A&A 251, 680
Peterson, J.R., Aberth, W., Moseley, J., & Sheridan, J., 1971, Phys Rev A, 3, 1651
Press W.H., Teukolsky, S.A., Vetterling, W.T., & Flannery, B.P., 1992, *Numerical Recipes*, (Cambridge University Press; Cambridge)
Puetter, R.C., 1981, ApJ 251, 446
Puy, D., Alecian, G., Le Bourlot, J., Leorat, J., Pineau des Forets, G., 1993, A&A 267, 337
Raymond, J.C., Cox, D. P., & Smith, B.W., 1976, ApJ 204, 290
Rees, M.J., Netzer, H., & Ferland, G.J., 1989, ApJ 347, 640
van Regemorter, H., 1962, ApJ 136, 906
Rauch, T., 1997 A&A 320, 237
Rephaeli, Y., 1987, MNRAS 225, 851
Reilman, R.F., & Manson, S.T., 1979, ApJS 40, 815, errata 46, 115; 62, 939
Roberge, W.G., Jones, D., Lepp, S., & Dalgarno, A., 1991, ApJS 77, 287
Rossi, B., 1952, *High-Energy Particles* (New York; Prentice-Hall)

Rowan, T., 1990, *Functional Stability Analysis of Numerical Algorithms*, Ph.D. Thesis, Department of Computer Sciences, University of Texas at Austin
Rubin, R.H., 1968, ApJ 153, 671
Rubin, R.H., 1983, ApJ 274, 671
Rubin, R.H., Simpson, J.R., Haas, M.R., & Erickson, E.F., 1991, ApJ 374, 564
Rybicki, G. B., & Hummer, D. G. 1991, A&A, 245, 171
Rybicki, G. B., & Hummer, D. G. 1992, A&A, 262, 209
Rybicki, G. B., & Hummer, D. G. 1994, A&A, 290, 553
Rybicki, G. B., & Lightman, A.P., 1979, *Radiative Processes in Astrophysics* (Wiley, New York)
Sanders, D. B., et al. 1989, ApJ 347, 29
Saraph, H. E., 1970, J.Phys.B. 3, 952
Savage, B. D., and Sembach, K. R., 1996, ARAA 34, 279
Sciortino, S., et al., 1990, ApJ 361, 621
Scott, J.S., Holman, G.D., Ionson, J. A., & Papadopoulos, K., 1980, ApJ 239, 769
Schuster, A., 1905, ApJ 21, 1
Schutte, W.A., Tielens, A.G.G.M., and Allamandola, L.J., 1993 ApJ 415, 397
Schwarzschild, M., 1965, *Structure & Evolution of the Stars*, (New York: Dover)
Seaton, M.J., 1959, MNRAS 119, 90
Seaton, M.J., 1987, J.Phys. B 20, 6363
Sellmaier, F.H., Yamamoto, T., Pauldrach, A.W.A., & Rubin, R.H. 1996, A&A, 305, L37
Shine, R.A., & Linsky, J.L., 1974, Solar Physics 39, 49
Shull, J.M., 1979, ApJ 234, 761
Shull, J.M., & Van Steenberg, M. E., 1982, ApJS 48, 95
Shull, J.M., & Van Steenberg, M. E., 1985, ApJ 298, 268
Sellgren, K., Tokunaga, A.T., and Nakada, Y., 1990, ApJ 349, 120
Sikora, M., Begelman, M.C., & Rudak, B., 1989, ApJ, 341, L33
Simonyi, C., 1977, *Meta-Programming: A Software Production Method*, Thesis, Stanford University
Snow, T. P., & Dodger, S. L.., 1980, ApJ 237, 708
Snow, T. P., & York, D. G., 1981, ApJ 247, L39
Snow, T.P., & Witt, A., 1996, ApJ 468, L65
Spitzer, L., 1948, ApJ 107, 6
Spitzer, L., 1962, *Physics of Fully Ionized Gasses*, (Interscience: New York)
Spitzer, L., 1978, *Physical Processes in the Interstellar Medium*, (Wiley: New York)
Spitzer, L., 1985, ApJ 290, L21
Spitzer, L., & Tomasko, M.G., 1968, ApJ 152, 971
Stecher, T.P., & Williams, D.A., 1967, ApJ 149, 29
Stoy, R.H., 1933, MNRAS 93, 588
Storey, P.J., 1994, A&A 282, 999
Storey, P.J., & Hummer, D. G., 1991, Comput. Phys. Commun. 66, 129
Storey, P.J., & Hummer, D. G., 1995, MNRAS 272, 41
Swings, P., & Struve, O., 1940, ApJ 91, 546
Tarter, C.B., & McKee, C.F., 1973, ApJ 186, L63
Tielens, A.G.G.M., & Hollenbach, D., 1985a, ApJ 291, 722
Tielens, A.G.G.M., & Hollenbach, D., 1985b, ApJ 291, 746
Tout, C.A., Pols, O.R., Eggleton, P.P. and Han, Z., 1996, MNRAS 281, 257
Turner, J., & Pounds, K., 1989, MNRAS 240, 833
Van Blerkom, D., & Hummer, D. G., 1967, MNRAS 137, 353
van Regemorter, H., 1962, ApJ 136, 906
Vedel, H., Hellsten, U., and Sommer-Larsen, J., 1994, MNRAS 271, 743
Vernazza, J.E., Avrett, E.H., & Loeser, C.B., 1981, ApJS 45, 635
Verner, D.A., Yakovlev, D.G., Band, I.M., & Trzhaskovshaya, M.B., 1993, Atomic Data Nuc Data Tables 55, 233
Verner, D.A., & Yakovlev, 1995, A&AS 109, 125
Verner, D.A., & Ferland, G.J., 1996, ApJS 103, 467
Verner, D. A., Ferland, G., Korista, K., & Yakovlev D. G., 1996, ApJ, 465, 487
Verner, D.A., Verner, K., & Ferland, G.J., 1996, Atomic Data Nuc Data Tables, 64, 1
Voronov, G.S., 1997, ADNDT 65, 1

Voit, G. M., 1991, ApJ 377, 1158
Vriens, L., & Smeets, A.H.M., 1980, Phys Rev A 22, 940
Watson, W. D., 1972, ApJ 176, 103
Weisheit, J.C., 1974, ApJ 190, 735
Weisheit, J.C., and Collins, L.A., 1976, ApJ 210, 299
Weisheit, J.C., & Dalgarno, A., 1972, Ap. Letters, 12, 103
Weisheit, J., Shields, G. A., & Tarter, C.B., 1981, ApJ 245, 406
Werner, K., & Heber, U., 1991, in *Stellar Atmospheres: Beyond Classical Models*, p 341, NATO ASI Series C, eds. L. Crivellari, I. Hubney, & D.G. Hummer, (Kluwer; Dordrect)
White, R. E., 1986, ApJ 307, 777
Wiese, W.L., Smith, M.W., and Glennon, B.M., 1966, NSRDS-NBS 4
Wilkes, B.J., Ferland, G.J., Truran, J., & Hanes, D., 1981, MNRAS 197, 1
Wilkes, et al 1994, ApJS 92, 53
Wilkinson, D.T., 1987, in *13th Texas Symposium on Relavistic Astronphysics*, M.P. Ulmer, ed., (World Scientific; Singapore), p209
Williams, R.E., 1967, ApJ 147, 556
Williams, R.E., 1992, ApJ 392, 99
Wills, B., Netzer, H., & Wills D., 1985, ApJ 288, 94
Winslow, A.M., 1975, Lawrence Livermore Lab. report UCID-16854
Wishart, A.W., 1979, MNRAS 187, 59p
Wolfire, M. G., Tielens, A., & Hollenbach, D., 1990, ApJ 358, 116
Worral et al 1987, ApJ 313, 596
Wyse, A. B., 1941, PASP 53, 184
York, D. G., Meneguzzi, M., & Snow, T. 1982, ApJ 255, 524
Xu, Y., & McCray, R., 1991, ApJ 375, 190
Zamorani, G., et al. 1981, ApJ 245, 357
Zycki, P.T., Krolik, J.H., Zdziarski, A.A., and Kallman, T.R., 1994, 437, 597
Zygelman, B., & Dalgarno, A., 1990, ApJ 365, 239

9. INDEX

— D—

— E—

— F—

— T—

— W—

— X—

— Z—