Quick-start guide to Cloudy

Spring meeting, Keeneland, Lexington

version 06.02

G.J. Ferland

Department of Physics and Astronomy
University of Kentucky, Lexington
http://www.nublado.org

Quick start guide to Cloudy
version 06.02

1. Introduction

Numerical simulations make it possible to understand complex physical environments starting from first principles. This is a quick-start guide to the spectral simulation code Cloudy, which is designed to just that. It determines the physical conditions (temperature, level of ionization, and chemical state) within a non-equilibrium gas, possibly exposed to an external source of radiation, and predicts the resulting spectrum.

This document begins with an introduction to the code's web site and is followed by a description of HAZY, the code's documentation. It goes on to an overview of how to set up the code and create two simple models. The following sections summarizes the parameters used to establish the boundary conditions for a simulation. This document only gives an outline of what the commands do – the indicated sections of HAZY should be consulted to find out more.

1.1. The web site

The source and documentation for Cloudy are available from the code's web site www.nublado.org. Go to the downloads / current version page and follow the instructions under "setting up this version". You will download the source, data files, the test suite, and the code's documentation. The code's test suite, a large body of simulations that exercise the code over it limits, should be run to confirm that you have a valid executable. The site's hot fixes and problems pages should also be viewed. They may contain "hot fixes", small changes to the source that fix bugs that were discovered after the code was released, or "problems", a statement of known problems with that version. There is also a discussion board for questions or comments.

1.2. Versions

Many versions of the code exist on the web site. Like all software, Cloudy goes through versions as it is developed. The fidelity of the simulations improves as processors become faster. Atomic and molecular rate coefficients become better as theory and experiment makes progress. As a result the code's predictions improve over time.

The ***current stable version*** is recommended. There may also be a ***release candidate version***, a nearly final version which may soon become the current stable version. All versions that have ever been the ***current stable version*** are still on the web site. Finally, the ***bleeding edge*** page gives access to the current development version (which is in an undefined state).

1.3. Setting up the code

Cloudy is open source. Its source, atomic data files, the suite of test cases, and the documentation HAZY, are available at the web site www.nublado.org – this is

the authoritative source for information about Cloudy. Go to the link "downloads" at the top of the page – this will take you to a page that gives several versions of the code. You probably want the current stable version. Go to the link indicated for the version you want – that page will give detailed information about downloading and setting up the code.

Setting up the code involves three broad steps. First download the files. There are four categories; source, atomic data, documentation, and the test suite. Each category is described with its own *readme*.htm* document. Optional stellar atmosphere files may also be downloaded but are only needed if you want to use these stellar continua.

First download the files. Then edit *path.c* in the source and change the character variable *chDataPath* to give the location of the atomic data. Next compile the source. The web page gives the best options for the most popular compilers. The code is written in ANSI C and requires a good C compiler. It has been extensively tested with the GNU gcc compiler, which I recommend. Finally, run the entire test suite to check that you have a valid executable. The Perl script *runall.pl* is included in the test suite and runs all tests. Edit the script so that it knows how to find the Cloudy executable. The script *checkall.pl* verifies the results. If *checkall.pl* announces success then you have a valid executable. The web site describes all of this is in much greater detail.

Next view the "hot fixes" and "problems" pages on the web site for the version of the code you downloaded. These contain corrections to the code's source that you must make as well as a statement of known problems with that version of the code.

1.4. Hazy

HAZY is the code's main documentation and comes in three volumes. Part 1 gives a complete list of all commands used to drive the code. Part 3 describes how to set up a calculation, defines the output that is generated, shows how to extract observed quantities from the predictions, and how to call Cloudy as part of a larger program. Part 2, which is badly out of date at the time of this writing, describes the physics behind the simulation.

The past few years have seen a major expansion of the code's capabilities, especially in infrared emission and molecular physics. Unfortunately, Part 2 of Hazy has not kept up. Parts 1 & 3 are up to date and new versions are posted on the web site several times per year. Part 2 is badly out of date since it has not had a high enough priority to be updated with available resources. I have not given up and do intend to update this eventually. For now, an ADS search on Ferland will find the most recent papers that describe advances in the physics.

1.5. Assumptions

The code computes the non-equilibrium ionization, thermal, and chemical state of a cloud that may (or may not) be exposed to an external radiation field. See Osterbrock & Ferland (2005; hereafter AGN3), for more details.

The usual assumption is that atomic processes have had time to reach steady state, so the density of a species or level *i* is given by a balance equation of the form

$$\frac{\partial n_i}{\partial t} = \sum_{j \neq i} n_j R_{ji} + Source - n_i \left(\sum_{j \neq i} R_{ij} + Sink \right) = 0 \ [\mathrm{cm^{-3}\ s^{-1}}] \qquad (1)$$

Here R_{ji} represents the rate [s^{-1}] that a species *j* goes to *i*, *Source* is the rate per unit volume [$cm^{-3}\ s^{-1}$] that new atoms appear in *i*, and *Sink* is the rate [s^{-1}] they are lost. This, together with equations representing conservation of energy and charge, fully specify the problem.

The web site has a page devoted to intended future developments. Fully time-dependent simulations, with densities having a non-zero time derivative, are under development. The code can also compute a flow (Henney et al. 2005) although this too is under development.

1.6. What Cloudy can do

Given this assumption, the code will determine the ionization, temperature, and chemical state of a cloud and then predict its spectrum. All of this is done self-consistently, with the minimum number of free parameters.

The following sections outline how to set up a calculation. You first create an input script that sets boundary conditions. These include the cloud's geometry, composition, density, and thickness. The radiation field striking the cloud, often its only source of heat and ionization, is set, and other sources of heat can also be included. The code then computes the structure of the cloud and its spectrum. Output options make it possible to save the spectrum or cloud properties for later analysis. Chapter 1 of Part 1 of HAZY gives a more extensive overview of the code. The format of the input script, the file the specifies the boundary conditions, is described in the chapter *Introduction to commands* of Part 1 of HAZY, and output options are described in the chapter *Controlling output* of Part 1 of HAZY.

The solution of the equations of statistical equilibrium, charge conservation, and conservation of energy, determines the level of ionization, the particle density, and gas kinetic temperature, the chemical state, populations of levels within atoms and the full spectrum, which often includes hundreds of thousands of lines. Thus, a very large number of observables result from a few free parameters – often the goal is to determine these free parameters from observations. This approach is outlined in Section 8 of the review article Ferland (2003; ARA&A, 41, 517, available here).

1.7. Definitions

There is a fair amount of jargon that goes along with quantitative spectroscopy and numerical simulations of a cloud. These are summarized in the chapter *Definitions* of Part 1 of HAZY. As a minimum you should know the definitions of the types of geometries (open and closed), continua (incident, transmitted, diffuse, and reflected), and the terms covering factor, density, and column density.

1.8. Running the program

Cloudy needs to be able to derive the following information before it can simulate conditions within a cloud; the shape of the continuum striking the cloud, the intensity of this continuum, the total hydrogen density, the composition of the gas, and the thickness of the cloud. Unless otherwise specified, the gas-phase abundances will be close to the solar values and grains will not be included.

The code's philosophy is for a reasonable set of conditions to be assumed by default. These are listed in Section 3.3 of Part 1 of HAZY. Commands are added to change these conditions.

1.8.1. Command format

A series of commands, entered one per line into a file that code reads, tell Cloudy what to do. The commands can be entered in any order. Each command is identified by the first 4 or 5 letters on the line. Numbers may also appear on the command line to specify parameters. The format for commands is described further in the chapter *Introduction to commands* of Part 1 of HAZY.

Exponential notation cannot be used – the number 2e3 will be interpreted as a two followed by a three. The code generally avoids exponents by using logs of numbers. It is important to read the documentation to see what is expected for numerical input. For instance, temperatures are interpreted as logs if they are less than or equal to 10 and as the temperature if greater.

1.8.2. A single model

To run a single model I enter the commands into a file with a name like *sim.in*. The following gives the commands for a simple simulation:

```
title - this is the input stream for a planetary nebula
//
// the double slash is one format for comments,
//
// set the temperature of the central star
blackbody, temp = 100,000K
//
// set the log of the total luminosity (erg / s) of the central star
luminosity total 38
//
// this sets a "typical" PN chemical composition
// including grains
abundances planetary nebula
//
// log of inner radius of nebula in cm
radius 17
```

```
//
// log of hydrogen density - cm^-3 -
// constant H density is done by default
hden 4
//
// this is a sphere with large covering factor
sphere
```

Lines beginning with "//" are comments. The other lines give commands that are described further below. The file ends with a blank line or the end of file.

If the executable is called *cloudy.exe* then a single simulation could be run with the command

```
cloudy.exe < sim.in > sim.out
```

where the file *sim.in* contains the commands given above and the main output will go to the file *sim.out*. This method of running the code is described in the chapter *Running a Single Model* in Part 3 of HAZY.

I also keep a "run" script on my path – in Unix the contents of the *run* file are the following

```
/u/home1/gary/cloudy/current/main.exe < $1.in > $1.out
```

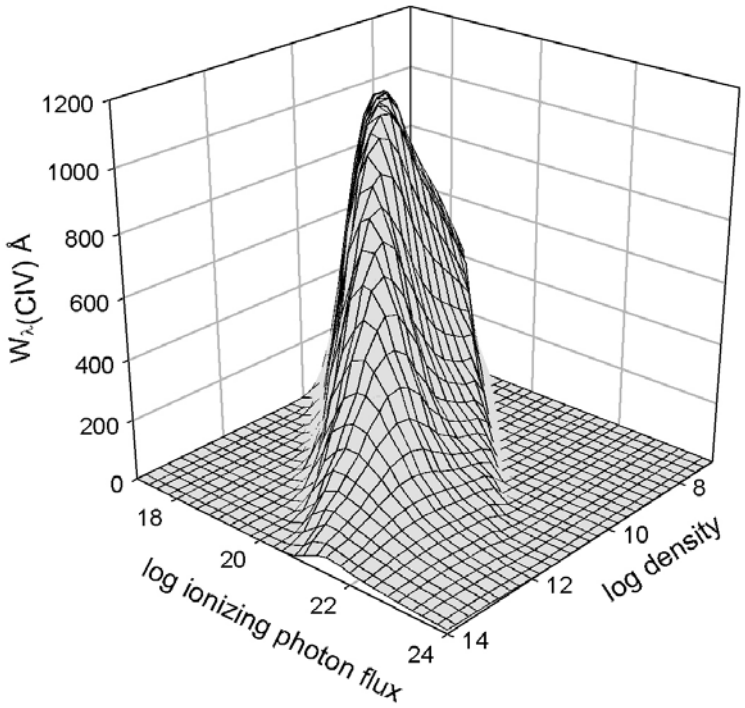
and in Windows the following is in *run.bat*:

```
c:\projects\cloudy\current\debug\current.exe < %1.in > %1.out
```

In both cases the code could be run with the command "*run sim*". The code would read commands from *sim.in* and write its output to *sim.out*.

1.8.3. Grids of models

The greatest physical insight is often obtained from looking at results of grids of calculations to discover physical trends or correlations. An example is shown at the right, which shows the equivalent width of one of the strongest emission lines in quasars as a function of two cloud parameters (Hamann & Ferland 1999). The code is designed to be used as a sub-program of other, larger, codes. The command format is the same when it is used as a subprogram but commands are entered by calling a



The predicted equivalent width of C IV λ1549 as a function of cloud density and the flux of ionizing photons striking the cloud. See Ferland (2003) for further details.

special routine. Cloudy is then executed and the predictions are retrieved after the simulation finishes. This method of running the code is described in the chapter *Cloudy as a Subroutine* in Part 3 of HAZY. The test suite downloads include a set of sample programs that illustrate this use of the code.

1.8.4. Heads up!

Exponential notation cannot be used to enter numbers in a command. The number 3e4 will be interpreted as two numbers, a 3 followed by a 4. Many commands would accept this number as the log of 30,000 (the number 4.477).

Most commands expect ***numerical parameters*** on the command line to be ***in a particular order, although they can often be omitted from right to left***. Be sure to follow the rules for each command. Default values are assumed when an optional parameter is missing.

The program is designed to be autonomous and self-aware. It continuously audits itself as a calculation progresses. The end of the calculation may have comments on various aspects of the physics. ***If problems occur*** then the code may generate a warning or caution. These are described further on page 24 below.

1.9. Citing the use of Cloudy

The ***current stable version*** should always be used in publications. Copies of all versions of the code that have ever been the current stable version are on the web site. Papers should cite the code giving the version number and a reference to the last major review of Cloudy's development[1], Ferland et al. (1998). An example would be "We used version 05.07a of Cloudy, last described by Ferland et al. (1998)". Then, years from now, when someone wants to know how an answer was obtained, the version used to obtain a result can be retrieved from the web site.

Citations are important for sustaining the development of Cloudy. I use them to support proposals to funding agencies to continue the code's development. Please follow the citation example shown above.

1.10. Collaborations

Cloudy is a totally open source code. The bleeding edge web page gives yesterday's version of the code (the ***last*** version) and the last version of the code that passed the test suite (***last_good***). Some of the most useful additions to the code have been surprises introduced by volunteers. There is a great deal of work left to be done. You are welcome to help out!

---

[1] At one time the preferred citation was the reference to the on-line version of Hazy. The ADS does not track such citations even though, in the case of Cloudy, there are roughly a thousand of them. ADS citation rates are used for some pretty important decisions. Please do not cite Hazy. Cite the published paper instead.

2. Two very simple models

As a simple introduction let's create a very simple model of a planetary nebula (see Chapter 10 of AGN3) and of a cloud in the Broad Emission Line Region of a quasar (Chapter 13 of AGN3). Use a simple editor like vi, emacs, or notepad to create a file with the following commands. As a minimum you must specify a few parameters, the continuum source and the density, radius, and composition of the nebula. We go over these briefly here.

2.1. A simple planetary nebula

The temperature and luminosity of the central star must be specified since starlight is the main source of heat and ionization for a planetary nebula. The central star is a cooling while dwarf. We can approximate it as a hot blackbody at the Eddington limit for one solar mass. We set the stellar continuum shape and luminosity with separate commands, as follows:

```
blackbody, T=100,000K
luminosity total 38
```

This sets the continuum shape to a 10^5 K blackbody. Its luminosity is normalized to a total luminosity of 10^{38} erg s^{-1}. This is the luminosity case described on page 14 below. Luminosity and shape commands are described in the section beginning on page 19 below.

Next we need to specify the properties of the nebula, the shell of gas surrounding the star. We will need to specify an inner radius since the continuum luminosity is specified. A typical inner radius is a fraction of a parsec, so let's use 10^{17} cm. A typical hydrogen density, measured from ratios of forbidden lines (AGN3 Chapter 5), is something on the order of 10^5 cm^{-3}. Images suggest that the gas fully covers the star so we include the **sphere** command. Finally, the code would assume the chemical composition of the sun if we didn't change it. It is actually quite different and includes dust. There are a series of built-in mixtures of gas and dust that are specified by the **abundances** command (page 17 below). We will use the one typical of planetary nebulae. The input commands would be the following:

```
radius 17
hden 5
sphere
abundances planetary nebula
```

We did not set an outer radius so that the calculation extends until the gas kinetic temperature falls below the value of 4000 K. The calculation will end when the electron temperature falls below this. Generally this will be near the H^+ - H^0 ionization front. Many other stopping criteria could have been used but this is a simple model.

We will have the code create two output files in addition to its standard output. To do this we include **punch** commands, which specify what to output and a file name (see page 24 below).

```
punch overview “pn.ovr”
punch continuum “pn.con” units microns
```

The overview file makes it possible to create plots showing the temperature and ionization structure of the cloud. The continuum file will show the incident and total continuum and lines emitted by the star and nebula.

Place all of these commands into a single file with the name “pn.in”. The lines should be written one after another and there should not be any blank or empty lines before the end of the commands. The file will contain the following.

```
blackbody, T=100,000K
luminosity total 38
radius 18
hden 5
sphere
abundances planetary nebula
punch overview “pn.ovr”
punch continuum “pn.con” units microns
```

Run the program as follows:

```
cloudy.exe < pn.in > pn.out
```

You will end up with three output files, the main output file **pn.out** and the two punch files **pn.ovr** and **pn.con**. Have a look at **pn.out** – the output is described further in the section starting on page 23 below. Use the two punch files to create plots. The first column of the overview file gives the radius in cm. Another column gives the gas temperature. Make a plot showing the temperature as a function of radius. The first column of the continuum file gives the wavelength in microns since this was requested on the **punch** command. Column seven gives the total emission. Make a plot showing the emission as a function of wavelength. Both axes will probably need to be logs due to the large dynamic range.

2.2. A quasar cloud

Next compute a simple model of a quasar emission-line cloud. The incident continuum shape is a power law. Most of the literature in this field describes the continuum intensity in terms of an ionization parameter, the ratio of photon flux to hydrogen density, so we will too.

```
table power law
ionization parameter -2
```

A number of different continuum shapes are stored in the code as look-up tables. This **table** command uses one. The ionization parameter has the effect of specifying the continuum as a flux, photons cm^{-2} s^{-1}, so this is an example of the intensity case (page 14 below) and a starting radius does not need to be specified.

We still need to specify a hydrogen density and any stopping criteria. A typical density is 10^{10} cm^{-3}. Most published calculations stop at a certain column density, so let's set 10^{22} cm^{-2}. This is large enough for an H^+ - H^0 ionization front to be present. Solar abundances are fine so we will not change these. The covering factor, the fraction of 4π sr covered by gas, is small so the **sphere** command is not included. So we have

```
hden 10
stop column density 22
```

We will include the same two punch files but with different file names

```
punch overview "blr.ovr"
punch continuum "blr.con" units microns
```

Put these into a file with the name *blr.in* and run the code like we did above. Examine the main output *blr.out* and make the same plots of the temperature structure and emitted continuum.

These are close to being the simplest models that can be done. The following sections go over each of the parameters described above and point to sections of HAZY that go into more detail.

3. Geometry

3.1. Zones and iterations

The code works by dividing a cloud into a large number of thin layers called zones. There is a default limit of 1400 zones. The code will generate a warning if the calculation stops because the default limit to the number of zones was reached since this was probably not intended. Warnings, cautions, surprises, and notes are described on page 24 below.

By default the code will make one iteration, one complete simulation of a cloud. If line or continuum transfer is important then more than one iteration will be necessary for a valid solution since the total optical depths must be known. The code will complain if too few iterations were performeed.

3.1.1. Commands normally used

set nend changes the default limit to the number of zones.

stop zone tells the code to stop at a particular zone. This is mainly used for debugging, or, in the case **stop zone 1**, to check on conditions at the illuminated face of a cloud.

iterate sets the number of iterations to be performed. The default is a single iteration and more will be needed when radiative transfer effects are important. There is a special version of the command, **iterate to convergence**, which tells the code to recompute the model until line and continuum optical depths become stable.

3.1.2. Heads up!

The code will ***generate a warning*** if it stops because it ***reaches the default limit to the number of zones***, since this probably was not intended. If this occurs, use the **`set nend`** command to increase the limit to the number of zones.

The code ***will generate warnings or cautions if significant line optical depths changed in the last iteration***. Increase the number of iterations if this occurs with the **`iterate`** command, or use the **`iterate to convergence`** command.

3.2. The geometry – intensity & luminosity cases

Two cases, intensity and luminosity, exist. In the intensity case the inner radius does not need to be specified and the incident radiation field striking a unit area of cloud is specified. The emission per unit area (erg cm^{-2} s^{-1}) is predicted[2]. In the luminosity case the inner radius of the cloud and the total luminosity of the central source of radiation are both specified. The luminosities of emission lines (erg s^{-1}) are predicted.

The gas covering factor $\Omega/4\pi$ (AGN3 Section 5.9) is the fraction of 4π sr covered by gas as seen from the central object. If the central object has a total luminosity L then the nebula intercepts $L\Omega/4\pi$ of the radiation field. The covering factor will linearly affect line luminosities but have only second-order effects on line intensities.

If the code can determine the separation (cm) between the continuum source and the cloud then it will predict emission-line luminosities. Otherwise it will predict line intensities. This is described further in Section 2.3 of Part 1 of HAZY.

3.2.1. Commands normally used

`radius` sets the inner radius of the cloud. Line luminosities can be predicted if this is specified.

`covering factor` sets the covering factor of the cloud, $\Omega/4\pi$. The predicted emission-line luminosities scale linearly with the covering factor.

3.2.2. Heads up!

3.3. Open or closed geometry

The chapter *Definitions* of Part 1 of HAZY defines open and closed geometries and the chapter *Geometry* goes into more details. These considerations affect the transfer of diffuse fields and only change the predicted line intensities at the ~10% level.

---

[2] The emission per unit area is called the "intensity" in this document, in the code, and in Hazy. For an optically thick slab this is more properly termed the emittance and for an optically thin source this is $4\pi J$ where J is the mean intensity as defined in most radiative transfer texts.

An open geometry is the default. This is one where diffuse emission from the illuminated face of the cloud escapes from the system without striking other clouds.

A closed geometry is one where gas covers most of the continuum source. This is computed when the **sphere** command is included. In this case diffuse emission from the illuminated face of the cloud crosses a central cavity and strikes gas on the far side.

There are two classes of closed geometries, static and expanding. In the expanding case, the default when the **sphere** command is included, resonance line photons that cross the central hole will not be absorbed by gas on the far side due to Doppler shifts, perhaps due to expansion, between the near and far side. In the static case emission lines cross the central hole and are absorbed on the far side. These considerations have little effect on most lines but do affect emissivity of higher-*n* Lyman line of hydrogen.

This is described further in Part 1 of HAZY where the **sphere** command is discussed.

3.3.1. Commands normally used

sphere [expanding, static] tells the code to assume a closed geometry. The shell is assumed to be expanding rapidly enough that lines escaping from one side of the shell do not interact with the gas on the far side of the central hole. If the **static** keyword appears on the command linie then lines escaping from one side can be absorbed by gas on the far side.

3.3.2. Heads up!

These considerations affect the transport of the diffuse fields and have only second-order effects on the predicted spectrum or physical conditions in the gas. If you are uncertain about the geometry, try the simulation with and without the **sphere** command, and try both **sphere static** and **sphere expanding**. Is this distinction important? It usually is not.

3.4. Is the gas static or a wind? Is it turbulent?

The cloud is normally assumed to be static and that the lines are broadened only by thermal motions. A component of microturbulence can be added and a wind can be computed.

3.4.1. Commands normally used

turbulence adds a component of microturbulence to line broadening. This will make line pumping by the incident continuum more important and line trapping less important, so it does affect the spectrum. The parameter is the turbulent velocity u_{turb} in km s^{-1}.

wind simulates an expanding wind. A static cloud is assumed by default. The equations of motion determine the velocity as a function of depth. LVG or Sobolev approximation escape probabilities are used for the line transfer.

3.4.2. Heads up!

If turbulence is included then a turbulent pressure term is added to the gas equation of state, the relationship between density and pressure. This affects constant-pressure clouds. The gas equation of state is discussed in the *Optical Depths and Radiative Transfer* chapter in Part 1 of HAZY.

The velocity entered in the **turbulence** command is the component of turbulence u_{turb} that enters with the thermal width u_{th} as

$$u = \sqrt{u_{th}^2 + u_{turb}^2} \text{ [km s}^{-1}\text{]}$$

in setting the total line width u.

The velocity that appears on the **turbulence** command is in km s^{-1} because observed velocities are always expressed in these units. Cloudy actually works in cgs units.

3.5. What sets the outer edge to the cloud? Why should the calculation stop?

The code starts at the illuminated face of the cloud and works its way into deeper regions. The integration must stop for some reason. Many different stopping criteria can be specified and the code will stop when the first one is reached. Cloudy was originally designed to interpret optical / UV emission lines in quasars. These lines are produced in warm ionized gas so the default is to stop when the gas temperature falls below 4000 K. This will often be near the hydrogen ionization front. This would be a mistake if you want to consider cool atomic or molecular regions.

You should understand what you want to set the outer edge of the cloud and then confirm that the code reached this point. The introduction to the chapter *Stopping Criteria* of Part 1 of HAZY goes into this in more detail.

3.5.1. Commands normally used

radius sets the inner and outer radius of the cloud.

stop temperature sets the lowest electron temperature to allow. The calculation will stop then the electron temperature falls below this value. The default is 4000 K.

stop thickness sets the thickness of the cloud, the distance from the illuminated face to the outer edge.

`stop column density` sets a limit to the hydrogen column density. The default column density has units cm^{-2} and includes hydrogen in all forms, ionized, atomic, and molecular.

`double` This doubles the computed line optical depths at the end of an iteration. This command should be used if the region being simulated is only a layer on a much larger structure. This is the case, for instance, in a PDR calculation, where an unmodeled molecular cloud is assumed to lie beyond the PDR. Lines emitted from the shielded face of the computed PDR will be quite optically thick. Emission from this layer will be suppressed if this command is used since we then assume that the shielded face is the mid plane of the cloud. Were it not included the code would assume that the outer edge of the model is the outer edge of the cloud and these optically-thick lines would freely radiate from the shielded face. This is described further in the chapter *Optical Depths and Radiative Transfer* of Part 1 of HAZY.

3.5.2. Heads up!

The ***calculation will stop when it reaches a depth where any of the stopping criteria are satisfied***. Understand why the calculation stopped. Did it stop for the reason you expected or did it stop prematurely because another criterion was met or because the calculation had problems? Is the calculation a complete simulation of the region? The code will explain why it stopped in the first lines after the last zone. A sample printout of the last zone and the explanation for why the calculation stopped follows.

```
####259  Te:2.980E+01 Hden:1.000E+05 Ne:2.441E-01 R:1.000E+30 R-R0:1.408E+17 dR:7.588E+15 NTR: 27 Htot:9.894E-24 T912: 9.14e+04###
 Hydrogen      6.68e-01 2.44e-06 H+o/Hden 6.68e-01 6.63e-14 H-    H2 1.66e-01 7.65e-14 H2+ HeH+ 8.86e-15 Ho+ ColD 9.92e+21 1.01e+17
 Helium        1.00e+00 5.20e-08 0.00e+00 He I2SP3 1.14e-15 Comp H,C 1.78e-30 5.98e-31 Fill Fac 1.00e+00 Gam1/tot 1.58e-02
 He singlet n  1.00e+00 5.81e-22 9.19e-27 8.61e-29 4.08e-28 3.12e-28 He tripl 1.14e-15 1.12e-26 1.53e-28 1.66e-27 8.53e-28
 Pressure      NgasTgas 2.70e+06 P(total) 3.73e-10 P( gas ) 3.73e-10 P(Radtn) 2.14e-17 Rad accl 3.99e-14 ForceMul 8.66e+00
               Texc(La) 2.95e+03 T(contn) 3.00e+01 T(diffs) 6.30e-01 nT (c+d) 7.39e+06 Prad/Gas 5.75e-08 Pmag/Gas 0.00e+00
 Lithium       2.20e-01 7.80e-01 0.00e+00 0.00e+00 Berylliu 6.14e-01 3.86e-01 0.00e+00 0.00e+00 0.00e+00 sec ion: 1.34e-17
 Carbon        0.00e+00 0.00e+00 0.00e+00 0.00e+00 0.00e+00 0.00e+00 0.00e+00 0.00e+00 H2O+/O   0.00e+00 OH+/Otot 0.00e+00 Hex(tot) 0.00e+00
 Nitrogen      0.00e+00 0.00e+00 0.00e+00 0.00e+00 0.00e+00 0.00e+00 0.00e+00 0.00e+00 O2/Ototl 0.00e+00 O2+/Otot 0.00e+00
    model of cloud with primordial abundances exposed to background at Z=10
   Calculation stopped because lowest Te reached.    Iteration 2 of 2
   The geometry is plane-parallel.
```

4. Composition and density

What is the chemical composition of the gas? Should grains be included? Should PAHs be included? Commands that set the composition are discussed in the chapter *Chemical Composition* of Part 1 of HAZY. The default composition is close to solar and grains are not included.

The density at the illuminated face of the cloud, and a prescription of how this density varies with depth, must also be given. A density and composition that do not change with depth into the cloud will be assumed by default.

4.1. Chemical composition

Elemental abundances are given by number relative to hydrogen.

4.1.1. Commands normally used

abundances sets the abundances of all elements to the set of numbers given on the line. If no numbers are present but a keyword is then the composition is set to a standard mixture. Examples include the local ISM or a typical planetary nebula. Grains are included for some abundances sets. Consult the *Chemical Composition* chapter of Part 1 of HAZY.

element sets the abundance of a particular element, removes the element from the calculation, or sets its ionization state.

grains determines the type and abundance of grains. If grains are included then their abundance will be the same at all positions and quantum heating will be included when it is important. The **function** keyword will make the grain abundance depend on position and the **no qheat** keyword will turn off quantum heating. By itself this command specifies classical or large grains but does not include PAHs.

grains PAH includes PAHs. By default their abundance will be proportional to the ratio $n(H^0)/n(H_{tot})$ as suggested by observations of the Orion Bar.

4.1.2. Heads up!

Grain sublimation: A warning will be printed if grains are included and become hotter than their sublimation temperature. But they will not be removed from the calculation. The effects of grain sublimation can be mimicked by making the grain abundance vary with depth with the **function** option on the **grains** command, but this requires recoding a routine. Note also that if grains are destroyed the material contained within them must be returned to the gas phase to be self consistent. This is also not done automatically. Finally, a note will be printed if grains are not present but could exist since they would have been below their sublimation temperature.

Gas-phase abundances and grains: It is possible to leave the gas-phase composition at its solar value, appropriate if all elements are in the gas phase, but also set a population of grains with the **grains** command. This is not consistent – when grains are present the elements that comprise them are depleted from the gas phase. If you use both solar abundances and grains then the code will complain but still perform the calculation.

PAH abundances: There is good observational evidence that PAHs only exist in the H^0 region. Observations of the Orion Bar suggest that they are destroyed in the H^+ region and coagulate into larger grains in molecular regions. By default the PAH abundance depends on the ratio $n(H^0)/n(H_{tot})$. Other forms can be used by changing the routine called when the **function** keyword is used.

4.2. What is the cloud's density? Does it vary with depth?

The density of hydrogen is used to set the cloud density. The default is for the hydrogen density to be the constant value given by the **hden** command. Optional commands tell the code to assume constant pressure, include a magnetic field or turbulence in the gas equation of state, or to vary the density with a function specified by the user. Most of these are discussed in Chapter 8 *Density Laws* of Part 1 of HAZY.

4.2.1. Commands normally used

hden specifies the log of the hydrogen density (cm^{-3}). Constant density is the default.

constant pressure The cloud will have constant pressure. The equation of state includes pressure terms from thermal gas motions, turbulent motions, a magnetic field, the nearly isotropic radiation pressure produced by trapped lines, and the outward push of the incident radiation field on the gas. The keyword **gas** says to keep the gas pressure, rather than the total pressure, constant. This is discussed in the *Optical Depths and Radiative Transfer* chapter in Part 1 of HAZY.

filling factor The gas is normally assumed to fully fill the available space. This command sets a filling factor *f*, the fraction of the volume that contains gas (AGN3 Section 5.9). The remainder of the volume is a vacuum.

magnetic field These are ignored by default. Magnetic fields will contribute to the total pressure, and, optionally, to the turbulent velocity field. If the temperature is high enough then cyclotron cooling may become important. This is discussed in the chapter *Thermal Solutions* of Part 1 of HAZY.

4.2.2. Heads up!

hden gives the ***total hydrogen density***, defined as

$$n(\mathrm{H}) = n\left(\mathrm{H}^0\right) + n\left(\mathrm{H}^+\right) + 2n\left(\mathrm{H}_2\right) + \sum_{other} n\left(\mathrm{H}_{other}\right) \ \ [\mathrm{cm}^{-3}] \qquad (2)$$

where the sum includes H in other molecules. In nearly all cases hydrogen will be in one of the first three forms.

5. The incident radiation field

The radiation field is specified by its shape, which describes how it depends on wavelength or frequency, and by its intensity or luminosity. The shape and intensity are usually specified with different commands. The chapter *Defining the Continuum* of Part 1 of HAZY gives an overview of how this is done. More than one continuum source can be included. The following sections describe how to specify the shape and intensity of the radiation field.

5.1. The luminosity or intensity of the radiation field

The two geometries are described on page 14 above. In the ***luminosity case*** the luminosity (erg s^{-1}) or number of photons (s^{-1}) emitted by the central object into 4π sr, and the inner radius of the cloud (cm), are both specified in the boundary conditions. The luminosities (erg s^{-1}) of emission lines are predicted. In the ***intensity case*** the energy (erg cm^{-2} s^{-1}) flux or photon (cm^{-2} s^{-1}) flux striking the surface of the cloud is specified. The inner radius does not need to be given and the line intensity or emittance (erg cm^{-2} s^{-1}) is predicted.

There are many ways to specify the continuum intensity or luminosity. These are described in the chapter *Continuum Luminosity* of Part 1 of HAZY. The subsection *Intensity vs luminosity commands* of Part 1 of HAZY describes the distinction between these two cases.

By default most luminosity or intensity commands specify the quantity integrated over hydrogen-ionizing energies. Most also have the **`range`** option, which allows this energy range to be changed.

5.1.1. Commands normally used

`Q(H)` specifies the number of hydrogen-ionizing photons emitted by the central object into 4π sr (s^{-1}).

`phi(H)` is the intensity (cm^{-2} s^{-1}) equivalent of the **`Q(H)`** command. It sets the flux of hydrogen-ionizing photons striking the face of the cloud.

`luminosity` specifies the luminosity emitted by the central object into 4π sr (erg s^{-1}). By default this is the luminosity in H^0-ionizing radiation.

`intensity` is the intensity (erg cm^{-2} s^{-1}) equivalent of the **`luminosity`** command. It gives the intensity of radiation striking the cloud face. Note that this "intensity" is 4π times larger than the true mean intensity *J*, which has units erg cm^{-2} s^{-1} sr^{-1}.

`ionization parameter` This sets the dimensionless ratio of densities of ionizing photons to hydrogen, $U \equiv \varphi\left(H^0\right)\big/ cn\left(\mathrm{H}_{tot}\right)$. This can sometimes be useful since clouds with the same ionization parameter have similar levels of ionization and temperature. This is equivalent to an intensity command.

5.1.2. Heads up!

5.2. The shape of the radiation field

The continuum shape can be interpolated from a table of points, specified as a fundamental form such as a blackbody or bremsstrahlung, or taken from a previous Cloudy calculation. Methods of setting the shape are described in the chapter *Continuum Shape* of Part 1 of HAZY.

The shape should be specified between the code's energy limits of 10 m (yes, meters) and 100 MeV, if possible. The code will complain but compute the model if the continuum is not specified over the full energy range.

5.2.1. Commands normally used

`interpolate` sets the continuum shape by interpolating on a table giving pairs of frequency – flux points. This is the most commonly used method of setting the continuum shape since the results of other calculations, such a stellar atmosphere, can be directly entered.

`CMB` adds the cosmic microwave background for any redshift.

`blackbody` The shape will be a blackbody. This command has a number of options that allow the intensity of the radiation field to be specified at the same time using blackbody relationships.

`background` is a simple estimate of the X-ray – UV background at any redshift. It includes the CMB.

`table` are a series of continua that are specified as built-in tables. Some examples include the AGN / starburst background at any redshift *z*, some stellar continua, and the local ISM galactic background.

`table AGN` enters the Mathews & Ferland (1987) quasar continuum.

`table HM96` employs the Haardt & Madau (1996) background, which may be more accurate than that used in the **`background`** command. The redshift is the number on the line.

`table ISM` is the local ISM background.

`extinguish` will extinguish the incident continuum by photoelectric absorption due to a column density of neutral gas. This command is often used to remove hydrogen-ionizing radiation in a PDR calculation, in which is it assumed that an unmodeled H^+ region has extinguished much of the incident radiation field.

`cosmic ray background` will include the effects of galactic background cosmic rays. These are important when the calculation extends into molecular gas. The chemistry of the cold ISM is driven by a series of ion-molecule reactions that are initiated by cosmic-ray ionization. The required ions will not exist if no source of ionization is present and the chemistry network may collapse. The code will complain, but try to compute the model, if the calculation extends into cool regions without background cosmic rays.

5.2.2. Heads up!

Some shape commands also specify an intensity. An example is **`table HM96`** – that command specifies both the shape and intensity of the quasar background at some redshift. Commands that set both are listed in subsection *Keeping shape and*

intensity commands together of Part 1 of HAZY, which also describes a ***possible disaster***.

The shape of the ***incident continuum should be specified over the entire energy range*** considered by the code, 10 m to 100 MeV. An easy way to do this is to include, as a minimum, the cosmic microwave background and local ISM diffuse continuum, the two commands described next.

The ***cosmic microwave background should always be included*** with the **`CMB`** command. This is not done by default.

`table ISM` and **`cosmic rays background`** ***should probably be included*** for objects within our galaxy.

The ***quasar background continuum should probably also be included***, either with the **`background`** or **`table HM`** commands.

Intensity and shape commands should be kept together for simplicity.

6. Other commands

6.1. Radiative transfer

All line-formation processes, including line trapping, collisional deexcitation, continuum pumping, and destruction by background opacities, are included.

6.1.1. Commands normally used

`Case B` artificially sets the optical depths of hydrogen Lyman lines to very large values. Under some circumstances this will force the hydrogen recombination lines to their Case B intensities, the limit where all Lyman lines scatter often enough to be degraded into Lα and Balmer lines (AGN3, Section 4.2).

This command is only intended for setting up "homework" problems or test cases and should never be used in a simulation of a real object. If the calculated region does not contain dust then the mean intensity of Lα will become very large when this is used. This may result in unphysical photoionization rates for third or fourth-row elements.

`turbulence` and **`wind`** These commands are discussed on page 15 above. The code assumes that only thermal motions broaden lines unless an additional component of motion is added with one of these commands.

6.1.2. Heads up!

The ***Case B command has many artificial side-effects***, especially when grains are not present. It should not be used except in special test cases.

6.2. Making the calculation faster

It is possible to make the calculation run faster, but at the expense of its physical fidelity. This is described in the subsection *Performance speedups* in Part I of HAZY.

6.3. Miscellaneous commands

6.3.1. Commands normally used

`atom` changes the treatment of some model atoms. It has many keywords. **`atom H-like`** and **`atom He-like`** allow some aspects of the H-like and He-like isoelectronic sequences to be changed. **`H2`** and **`FeII`** turns on large (and computationally expensive) models of H_2 (Shaw et al. 2005) and Fe II (Verner et al. 1999) emission.

`init` Frequently-used commands can be stored in an initialization file. The **`init`** command will include the contents of this file in the input stream. The file name appears on the command line within a pair of double quotes.

Comments, lines that start with special symbols that tell the code to ignore them, can be included within the input stream. The section *Introduction to Commands* of Part 1 of HAZY describes their syntax and gives other details.

6.3.2. Heads up!

Only one init file can occur within an input stream.

7. The code's predictions

7.1. The default printout

When the code is executed on a command line, as in

```
cloudy.exe < test.in > test.out
```

the file *test.in* contains the code's input commands (read from *stdin* in C) and its output (written to *stdout* in C) goes to *test.out*. The output is fully described in the chapter *Output* of Part 3 of HAZY. The default output includes a copy of the input commands, a list of the abundances of the chemical elements and grains, the physical conditions in the first and last zone, a statement of why the calculation stopped, the intensity or luminosity of the stronger emission lines, and the mean ionization, temperature, and column density of many species.

7.1.1. Commands normally used

`title` enters a title that is printed at various places.

`print` commands change some aspects of the printout. It can sort the emission lines, change their format, and modify what information is printed.

`normalize` The log of the radiated luminosity or intensity of each emission line is printed after the line's label. The intensity of the line relative to a

normalization line follows. In astronomical optical emission-line spectroscopy the normalization line is usually Hβ and this is the code's default. The **`normalize`** command can specify any other line to normalize the spectrum.

7.1.2. Heads up!

Understand why the calculation stopped. This is given in the first comments after the last zone and is described further in the introduction to the chapter *Stopping Criteria* of Part 1 of HAZY. An example of the last zone printout and the statement of the reason why the calculation stopped is shown in section 3.5.2 on page 17 above.

7.2. Warnings, cautions, surprises, and notes

Examine the comments after the last zone for any warnings, cautions, or surprises. The code is designed to be autonomous and self-aware – it does many internal sanity checks to make sure that the calculation is valid (Ferland 2001, *Spectroscopic Challenges of Photoionized Plasmas*, ASP Conf 247, available here). If there are problems the code will say so by issuing warnings, comments that start with "W-", cautions, comments starting with "C-", or surprises, starting with "!". This is described further in the section *Warnings, Cautions, Surprises, and Notes* of Part 3 of HAZY. Warnings indicate that something is seriously wrong with the calculation. Cautions indicate that the code is on thin ice. Surprises indicate novel or interesting aspects of the results.

7.3. Observed quantities

The chapter *Observed Quantities* of Part 3 of HAZY explains how to relate quantities predicted by the code to observed properties. The chapter *The Emission Lines* of Part 3 of HAZY gives an overview of the labels used to indicate various emission lines.

7.4. Punch output

A typical calculation generates far too much information for even a small part to be included in the main printout. Instead, a series of **punch**[3] commands are used to create ancillary files that contain various predictions. These are described in the chapter *Controlling Output* in Part 1 of HAZY. The punch information is written into a file whose name appears within double quotes on the command line.

---

[3] The first generations of computers had a large machine that "punched" numbers onto Hollerith cards. This was an important way to save information since disk drives were so small. These punch commands are in the spirit of those punch machines.

7.4.1. Commands normally used

punch continuum gives the incident, transmitted, and total spectrum. The photon energy is normally given in Rydbergs but can be changed to keV, microns or Angstroms with the **units** option.

punch overview gives an overview of the calculation. This includes the electron temperature and density, the ionization of several elements, and abundances of some molecules, as a function of depth into the cloud.

punch element gives the abundance of ions and some molecules of a particular element as a function of depth into the cloud.

punch molecules gives the densities of a large number of molecular species as a function of depth into the cloud.

7.4.2. Heads up!

Emission lines appear in the output produced by the **punch continuum** command. The ***emission line to continuum contrast*** ratio depends on the size of the continuum mesh because the continuum cells are too coarse to resolve the lines. See the section *Line to Continuum Contrast* in the chapter *Observed Quantities* of Part 3 of HAZY.

8. Example calculations

This section describes the code's test suite, a series of simulations that are used to automatically validate the code every time it is changed, and then goes over one model in detail.

8.1. The test suite

The test suite is a large group of simulations of various astrophysical environments. They are included in download files on the code's web site. All of their file names end with "*.in*". The first part of the name indicates the type of model – for instance, all BLR models start with "*blr_*", all PDR models start with "*pdr_*", etc. The naming convention should be clear if you do a listing of all the files in the test suite directory ("**ls *.in**" at the command prompt).

The test suite is broken into three parts. ***Auto*** contains over two hundred simulations and will run in about half a day. This runs every night here in Lexington. ***Slow*** includes simulations with the large H_2 and Fe II atoms and takes more than a day to run. ***Programs*** contains a set of sample programs that test various aspects of using the code as a subroutine of larger programs.

Use the Perl script *runall.pl* to run all simulations in the auto test suite. This is an important step in setting up the code since it confirms that a valid executable was produced. A code as large as Cloudy is likely to discover bugs in a compiler, especially when highly optimized code is produced.

Each simulation contains **assert** commands that give the expected answer. If the predictions are wrong the code will print a string saying that an asserted quantity has changed. **Assert** commands provide an automatic way to revalidate the code every time it is changed. The script *checkall.pl* will confirm that all quantities had their expected value. The asserts can be removed if you change models parameters so that results change too. The script *tests_remove_asserts.pl* will do this.

8.2. Use the test suite simulations as examples

The file *doc_tsuite.htm* within the test suite distribution contains a list of all the test cases along with a summary of what they do and why they are set up the way they are. This file should be examined to get an idea of how the commands described above are used.

8.3. One of the models …

This section considers *orion_hii_pdr_pp.in*, one of the models in the test suite. A much longer discussion with more details is given in the chapter *Output* in Part 3 of HAZY. This simulates a plane-parallel molecular cloud with an H II region on its surface and a PDR between the two. Radiation from a nearby O star and galactic background cosmic rays are the only sources of heat and ionization. The calculation follows a ray of light into the cloud, and begins with the H^+ region, an ionized layer with a temperature of $T \approx 10^4$ K, continues into the PDR, a largely atomic region with a $T < 10^3$ K, and ends in the molecular cloud, with $T < 100$ K.

Comments within the input script explain the commands used to set up the simulation. This section discusses the various output files created by the simulation.

8.3.1. run orion_hii_pdr_pp.in

Run the *orion_hii_pdr_pp.in* script with the command

```
cloudy.exe < orion_hii_pdr_pp.in > orion_hii_pdr_pp.out
```

You will end up with many punch output files with names *orion_hii_pdr_pp.** and a main output file called *orion_hii_pdr_pp.out*.

8.3.2. Examine the main output

Examine the file *orion_hii_pdr_pp.out*. First, confirm that the calculation stopped for the intended reason. The reason is given after the last zone results, and, for this simulation, should be because the outer radius was reached.

The simulation may have had pressure convergence failures where the cloud passed through a ***thermal front***. Thermal fronts, and the problems they cause, are described in the chapter *Problems* of Part 3 of HAZY. Convergence problems are announced with lines that begin with the string "PROBLEM".

Next, identify some of the strongest emission lines in the spectrum. These are listed towards the bottom of the output, following the string "`Emission Line Spectrum`". Notice that two iterations were performed and make sure that you are looking at the last iteration (similar information is printed for each iteration). The command **`print last`** would tell the code to only print results of the last iteration but is not used in this test. The emission-line intensities are given relative to Hβ. Lα is only about twice as strong as Hβ, rather than the intensity ratio ~34 expected for Case B, because Lα is efficiently absorbed by dust. [O III] λ5007 is one of the strongest lines in the spectrum, as expected for an H II region.

Two blocks of emission-line intensities are printed because this is a dusty open geometry. The first block "`Emergent line intensities`" gives the spectrum that emerges from the illuminated face of the cloud. Some fraction of each line is emitted towards the hemisphere containing the molecular cloud. The grain albedo is used to compute the fraction that is reflected back towards the illuminated face. The second block of lines "`Intrinsic line intensities`" gives the total emission in all directions but does not include the effects of extinction due to the molecular cloud. This spectrum would be observed after correcting for reddening.

Some integrated properties of the cloud are listed towards the end of the file. Column densities of various constituents are given. The line with "`Log10 Column density (cm^-2)`" gives column densities of H^0, H^+, and H_2 – the cloud is predominantly molecular. Mean temperatures are also given following the line "`Log10 Mean Temperature (over volume)`"– the H^+ region has a mean temperature of nearly 10^4 K, the H^0 region has a mean temperature a bit under 10^3 K, and the H_2 region has a mean temperature of around 20 K. The output ends with a list of the asserted quantities – these compare the predictions of your executable with its historical predicted quantities.

The chapter *Output* in Part 3 of HAZY goes over the code's output in detail. Have a look.

8.3.3. The punch output

The input script contains many **`punch`** commands. Some are only intended as debugging aids, but others contain a wealth of physical information about the results of the simulation. The **`punch`** commands are described in the Chapter *Controlling Output* of Part 1 of HAZY while the chapter *Observed Quantities* of Part 3 of HAZY explains how to extract some observed quantities from all of this output.

The first line in a **`punch`** file gives a title for the columns of numbers that follow. In most files each line gives quantities for a single zone. The numbers are tab-delimited to make it easier to enter into a data base or plotting program. These tabs may appear confusing if viewed with an editor that is not aware of the tab settings. The following sections go over individual output file.

orion_hii_pdr_pp.con: This is produced by the **punch continuum** command and gives the incident, reflected, transmitted, and total continua. These are defined in the chapter *Definitions* of Part 1, and are described in both chapters *Controlling Output* of Part 1 and *Observed Quantities* of Part 3 of HAZY. The **units** option changes the energy scale from Rydbergs (the default) to microns. These wavelengths are the x-axis in Figure 1. The plot shows column two, the incident stellar continuum, as the smoother line, and column seven, the total emitted continuum, as the line with a great deal of structure. This is mainly the "reflected" spectrum, the continuum emergent from the illuminated face of the H II region. Extinction by dust in the molecular cloud prevents much light from emerging from the shielded face. The y-axis gives νf_ν with units erg cm^{-2} s^{-1}.

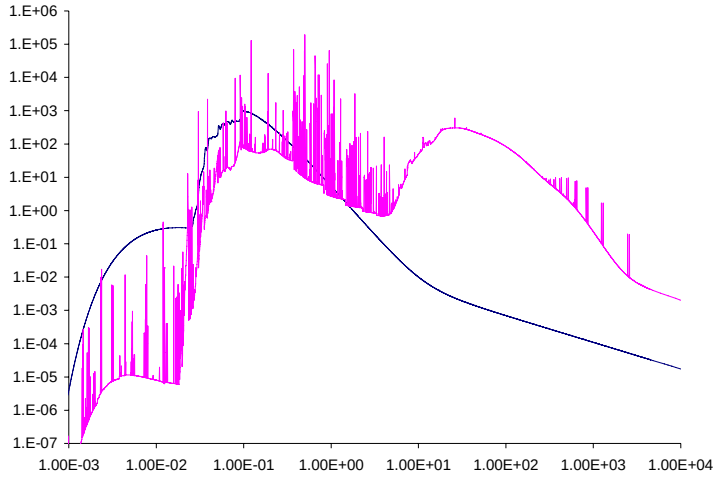


Figure 1 The incident (smooth) and emitted continua contained in the *orion_hii_pdr_pp.con* file. This is produced by the **punch continuum** command

orion_hii_pdr_pp.ovr: This is produced by the **punch overview** command. It gives the electron temperature and density, the hydrogen density, the heating rate, and the ionization distribution of H, He, C, and O. Figure 2 shows the hydrogen ionization and chemical structure contained in this file. The x-axis gives the log of the depth into the cloud in cm. The y-axis gives the log of the fraction of hydrogen in the form of H^+, H^0, and H_2. The hydrogen ionization front occurs at a depth of ~ 2×10^{17} cm. There is a small H^0 region and the rest of the cloud contains mostly H_2.

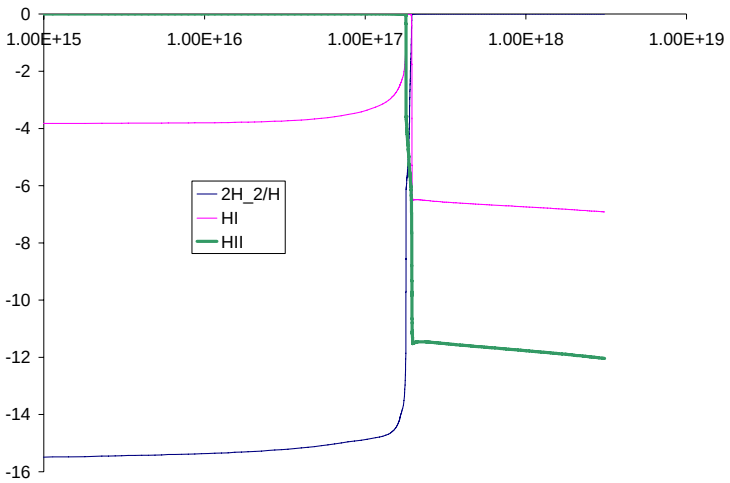


Figure 2 – The hydrogen ionization structure. The x-axis is the depth in cm and the y-axis gives the log of the fraction of H in H^+, H^0, and H_2.

orion_hii_pdr_pp.grntem file gives temperatures of grains as a function of depth. These, together with the electron temperature contained in the overview file, were used to create Figure 3. The highest curve is the electron temperature and is ~10,000 K across the H^+ region, falls to ~500 K in the small H^0 region, then to below 100 K in the H_2 region.

The calculation includes graphitic and silicate size-resolved grains. Their temperatures are shown as the lower cluster of curves. They have a range of temperatures ~100 – 200 K across the H^+ region and grow cooler in the molecular region. The gas and dust temperatures approach one another deep in the molecular cloud.

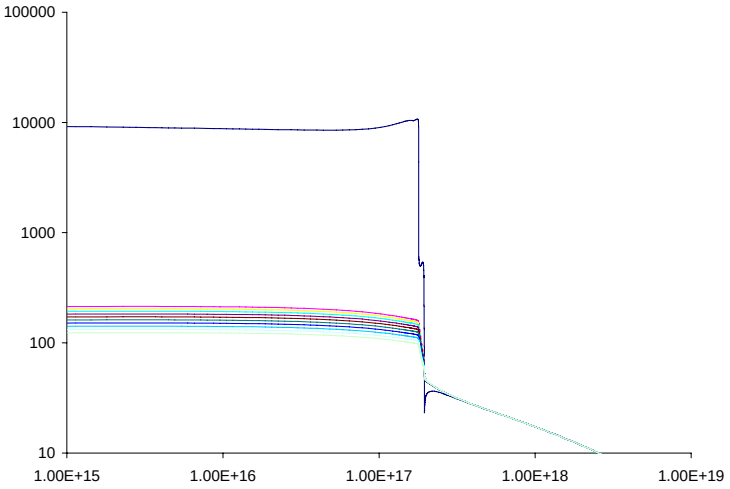


Figure 3 – temperatures of grains (the lower cluster of curves) and electrons (the higher curve). They become nearly equal in the molecular cloud.

orion_hii_pdr_pp.mol; This file gives densities of all molecules included in the calculation as a function of depth and was used to produce Figure 4. It also includes several measures of extinction due to dust.

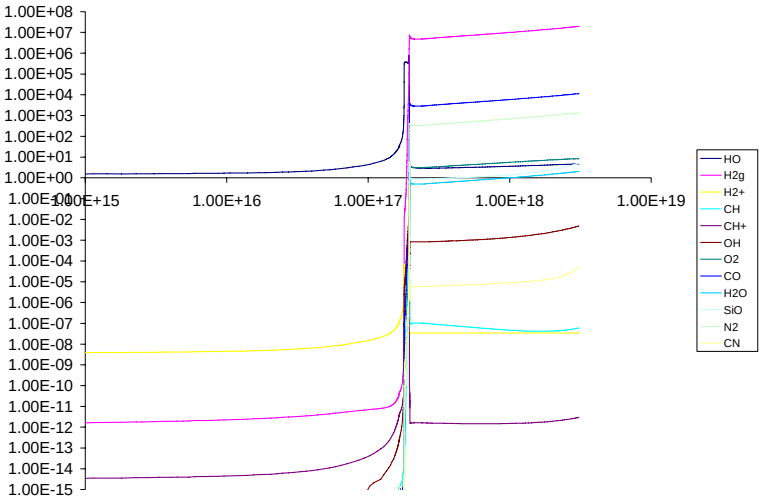


Figure 4 – densities of the molecules across the cloud.

These are only some of the predictions that come from this calculation. Feel free to explore by changing parameters and assumptions and by adding other output options.

8.4. Heads ups for classes of objects

The simulations in the test suite fall into several broad categories of objects. These are grouped together by the first part of their filename. Following subsections describe important considerations for setting up each of these classes of simulations.

8.4.1. BLR of Active Nuclei

The density is high enough for free-free absorption of low-energy radiation to be a significant heating process. As a result the infrared continuum can have a surprising effect on the temperature of these clouds. Extrapolating a reasonable power-law continuum into the infrared may result in runaway free-free heating. This, and other practical aspects of BLR clouds, is discussed in Ferland (1999a, available here) and in the last two chapters of AGN3.

8.4.2. NLR of Active Nuclei

Does dust exist in the ionized gas? Depletion patterns are not clear, as discussed by Ferguson et al. (1997, available here). The NLR is discussed further in the last two chapters of AGN3.

8.4.3. PDRs of Star-forming regions

It is critical that background cosmic rays, or a source of X-rays, be included if the simulation is to extend into a molecular cloud. The chemistry of cold interstellar matter is driven by a series of ion-molecule reactions. Interactions with ions have smaller activation barriers than interactions between neutrals. Ions will not exist if a source of ionization is not present and the chemistry network will collapse. Always include the **`cosmic ray`** and **`background`** commands.

The **`double`** command should be entered if the calculation stops before the outer edge of the molecular cloud is reached. See the discussion on page 17 above.

The culture in the PDR modeling community is to treat the PDR as an isolated phenomenon rather than an extension of the H II region. Nature does not do this, but you can force the code to do it by removing all hydrogen-ionizing radiation from the incident continuum. This is done with the **`extinguish`** command (see page 21 above).

If you do extinguish the hydrogen-ionizing radiation and start a PDR at that point you will usually find a thin layer of fairly highly-ionized hydrogen. This is caused by continuum pumping of the Lyman lines which then populate the metastable 2*s* level of hydrogen. This is called "Case C" in the ionized cloud community (see Ferland 1999b available here) and is described further in AGN3. Excited levels can then be photoionized by relatively soft radiation such as the Balmer continuum or Lα. Classical PDR calculations do not consider this physics. It can be disabled by assuming that the Lyman lines are very optically thick and self-shielded. Do this with the **`case B`** command (page 22 above). The **`case B`** command should not be included in any realistic simulation, which should start with the H^+ region and extend into the PDR (as done by Abel et al, 2005, ApJS, 161, 65, available here).

8.4.4. Galactic nebulae

Cosmic rays and the ISM background should be included. Dust is almost certainly present in the ionized gas, and should be included, along with depleted abundances of the refractory elements. This can be done by selecting a mix of gas and solids that includes dust (with one of the **`abundances`** commands) or by explicitly including these (with a combination of the **`grains`**, **`abundances`**, and **`depletion`** commands).

9. References

Abel, N. P.; Ferland, G. J.; Shaw, G.; & Hoof, P. A. M. Van, 2005, ApJS, 161, 65

Ferguson, J. W., Korista, K. T., Baldwin, J. A., & Ferland, G. J. 1997, ApJ, 487, 122

Ferland, G. J. 1999a, in *Quasars and Cosmology*, ASP 162, p 147 ed G Ferland & J Baldwin (astro-ph/0307450)

Ferland 1999b, PASP, 111, 1528

Ferland, G.J., 2003, ARA&A, 41, 517

Ferland, G. J. Korista, K.T. Verner, D.A., Ferguson, J.W. Kingdon, J.B. & Verner, E.M. 1998, PASP, 110, 761

Haardt, Francesco, & Madau, Piero, 1996, ApJ, 461, 20

Hamann, F., & Ferland, G. J. 1999, ARAA, 37, 487

Henney, W.J., Arthur, S.J., Williams, R.J.R., & Ferland, G.J. 2005, ApJ, 621, 328 (astro-ph/0501034)

Mathews, W. G., & Ferland, G. J. 1987, ApJ, 323, 456

Osterbrock, D.E., & Ferland, G.J. 2006, *Astrophysics of Gaseous Nebulae and Active Galactic Nuclei*, 2nd edition (Mill Valley: University Science Books) (AGN3)

Shaw, G. Ferland, G.J. Abel, N.P. Stancil, P.C. & van Hoof, P.A.M. 2005, ApJ, 624, 794, (astro-ph/0501485)

Verner, E.M. Verner, D.A. Korista, K.T. Ferguson , J.W. Hamann, F. & Ferland, G.J. 1999, ApJS 120, 101