Cloudy Quick-start Guide

Spring meeting, Keeneland, Lexington

http://www.nublado.org

Sunday, January 09, 2005

1 INTRODUCTION..4

1.1 Hazy 4

1.2 Assumptions 4

1.3 What Cloudy can do 5

1.4 Setting up the code...5

1.5 Versions and citations..6

1.6 Definitions 6

1.7 Driving the code 6

1.7.1 Command format...7

1.7.2 A single model..7

1.7.3 Grids of models 7

1.7.4 Heads up! 8

1.8 Collaborations..8

2 GEOMETRY 8

2.1 zones and iterations 8

2.1.1 Commands normally used 8

2.1.2 Heads up! 9

2.2 Is the geometry spherical or plane-parallel? 9

2.2.1 Commands normally used 9

2.2.2 Heads up! 9

2.3 Open or closed geometry 9

2.3.1 Commands normally used 10

2.3.2 Heads up! 10
2.4 Is the gas static or a wind? Is it turbulent? 10
2.5 What sets the outer edge to the cloud? Why should the calculation stop? 10
2.5.1 Commands normally used 11
2.5.2 Heads up! 11
3 COMPOSITION AND DENSITY 11
3.1 Chemical composition 12
3.1.1 Commands normally used 12
3.1.2 Heads up! 12
3.2 What is the cloud's density? Does it vary with depth? 12
3.2.1 Commands normally used 13
3.2.2 Heads up! 13
4 THE INCIDENT RADIATION FIELD 13
4.1 The luminosity / intensity of the radiation field 13
4.1.1 Commands normally used 14
4.1.2 Heads up! 14
4.2 The shape of the radiation field 14
4.2.1 Commands normally used 14
4.2.2 Heads up! 15
5 OTHER COMMANDS 15
5.1 Radiative transfer 15
5.1.1 Commands normally used 15
5.1.2 Heads up! 16
5.2 Miscellaneous 16
5.2.1 Commands normally used 16
5.2.2 Heads up! 16
6 THE CODE'S PREDICTIONS 16
6.1 The default printout 16
6.1.1 Commands normally used 17
6.1.2 Heads up! 17
6.2 Warnings, cautions, surprises 17
6.3 Observed quantities 17
6.4 Punch output 17
6.4.1 Commands normally used 18
7 EXAMPLES 18
7.1 The test suite 18
7.2 One of the models … 18
7.2.1 run orion_hii_pdr_pp.in 19
7.2.2 Examine the main output 19
7.2.3 The punch output 20
7.3 Heads ups for classes of objects 22

7.3.1 BLR 22
7.3.2 NLR 22
7.3.3 PDRs..22
7.3.4 Galactic nebulae 23
8 REFERENCES 23

1 Introduction

This is a quick-start guide to the spectral simulation code Cloudy. It is intended as an introduction to the code's main documentation, HAZY, which comes in three volumes. The most important parameters that are specified when setting up a simulation are described in the following sections. Each section summarizes a set of parameters, gives a short list of the most frequently used commands that specify them, and concludes with a summary of hidden pitfalls. 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 Hazy

HAZY is Cloudy's documentation and is available on the code's web site. Part 1 of HAZY gives a complete list of all commands used to drive the code. This part is totally up to date. Part 3 describes how to set up a calculation, the output that is generated, and how to extract observed quantities from the predictions. This part is fairly up to date. Part 2 describes the physics behind the simulation but is badly out of date. Updating Part 2 does not have a high enough priority to be done with available resources, although 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.

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

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

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 prescribes 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 a goal.

1.3 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. This involves establishing a scenario for the physical system under study, using the code to predict the structure of the cloud and its spectrum, and finally use the code's output to obtain the desired results. The cloud geometry, including its composition, density, and thickness, are specified. Next, the radiation field striking the cloud, often its only source of heat and ionization, is set. Other sources of heat can also be added. Output options are established so that the spectrum or cloud properties can be saved for later analysis. Finally, the simulation is performed by running the code. Chapter 1 of Part 1 of HAZY gives a more extensive overview.

The solution of the equations of statistical equilibrium, charge conservation, and conservation of energy, determines the level of ionization, the density and temperature of free electrons, the chemical state, populations of levels within atoms, ions, and molecules, 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. This approach to understanding physical conditions is outlined in the review article Ferland (2003; ARA&A, 41, 517, available here).

1.4 Setting up the code

The source for the code, its atomic data files, the test suite, 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.

There are three broad steps involved in setting up the code. The first is to download the files. There are several categories of files; source, atomic data, documentation, and test suite. These should be saved in different directories, and each categories is described with a *readme*.htm* document. Finally, optional stellar atmosphere files may also be downloaded. These are needed for certain commands, which use these stellar continua, to function. They are not needed if these stellar continua are not going to be used.

After downloading the source, edit *path.c* to point to the location of the atomic data, and then compile the entire 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. Then set up any desired stellar atmosphere data files. Finally run the entire test suite to check that you have a valid executable. The Perl script *runall.pl* runs the entire test suite and the script *checkall.pl* verifies the results. Edit the *runall.pl* script so that it points to the Cloudy executable. If *checkall.pl* announces success then you have a valid executable. The web site describes all of this is in more detail.

1.5 Versions and citations

Several versions of the code exist on the web site. Like all software, the code goes through versions as it is developed. The fidelity of the simulations improves as processors become faster, and atomic rate coefficients become better as atomic theory or experiment makes progress.

The ***current stable version*** is the recommended version. There may also be a ***beta 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 in the ***old versions*** page. Finally, the ***bleeding edge*** page gives access to the current development version (which may be in an undefined state).

The ***current stable version*** should be used in publications, which should cite the code giving the version number and a reference to the last major review of Cloudy's development. An example would be "We used version 96.01 of Cloudy, last described by Ferland et al. (1998)[1]". Then, years from now, when someone wants to know how an answer was obtained, the version used to obtain it can be retrieved from the ***old versions*** part of the web site. *All "current stable versions" that have ever been on the web site are still on 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 support. Please follow the citation example shown above.

1.6 Definitions

There is a fair amount of jargon that goes along with quantitative spectroscopy and numerical simulations of a cloud. These are summarized in Chapter 2 DEFINITIONS of Part 1 of HAZY. As a minimum you should know the definitions of; open vs closed geometry, incident, transmitted, diffuse, and reflected continua, covering factor, density, and column density.

1.7 Driving the code

Cloudy needs to be able to derive the following information before it can compute a model; the shape of the continuum illuminating the cloud, the

[1] At one time, the preferred citation was the reference to the on-line version of Hazy. The ADS does not track such citations. Please cite the last major review of the code, which is now Ferland et al. (1998; PASP, 110, 761, available here.

intensity of this continuum, and the total hydrogen density. Unless otherwise specified, the gas phase abundances will be close to the solar values and grains will not be included.

1.7.1 Command format

A series of commands, entered one per line, tell Cloudy what to do. They can be entered in any order. Each command is identified by the first 4-5 letters on the line. Numbers may also appear on the command line to specify parameters. The format for commands is described further in Chapter 3 of Part 1 of HAZY.

1.7.2 A single model

To run a single model I enter a series of commands into a file with a name like *sim.in*. An example follows:

```
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
black body, temp = 100,000K,
//
// set the total luminosity (erg / s) of the central star
luminosity total 38
//
// this sets "typical" PN abundances, with grains
abundances planetary nebula
//
// log of starting radius in cm
radius 17
//
// log of hydrogen density - cm^-3
hden 4
//
// this is a sphere with large covering factor
sphere
```

The file ends with a blank line or the end of file. If the executable is called *cloudy.exe* then a single simulation would be run with the command

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

This method of running the code is described in Chapter 2 of Part 3 of HAZY.

1.7.3 Grids of models

The greatest physical insight is often obtained from looking at trends in grids of calculations. The code is designed to be used as a sub-program of other, larger, codes. The formats of the commands are the same when it is used as a subprogram but they are entered by calling a special routine. Cloudy is then called and the predictions are retrieved when the simulation finishes. This method of running the code is described in Chapter 3 of Part 3 of HAZY.

1.7.4 Heads up![2]

Exponential notation cannot be used to enter numbers in a command. The number 3e4 will be interpreted as 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 ***parameters*** on the command line to be ***in a particular order, and they often can be omitted from right to left***. Be sure to follow the rules for each command. Default values are assumed when an optional parameter is left off the line.

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

1.8 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!

2 Geometry

2.1 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 800 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.

By default the code will make ***one complete simulation*** of a cloud, ***called an iteration***. If line or continuum transfer is important then it is necessary to perform more than one iteration to obtain a valid solution. The code will complain if more iterations need to be performed.

2.1.1 Commands normally used

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

`stop zone` This 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.

`iterate` It is often necessary to iterate on the solution (do more than one calculation of the structure) since total optical depths are not known until after

[2] *Heads up!* is an expression from the North American playground. It is most often screamed in the loudest possible voice when a pop fly ball is hit in baseball. A baseball can come down at ~100 mi/hr and, if it hits you in the right place, could kill you. It tells innocent bystanders to pay sharp attention to what's going on.

the first calculation of the structure of a cloud is complete. This is often only necessary when line transfer is important and lines are optically thick. The code will complain if optical depths of significant lines or continua change on the last iteration. There is a special version of the command, **`iterate to convergence`**, in which the code will recompute the model until line and continuum optical depths become stable.

2.1.2 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.

2.2 Is the geometry spherical or plane-parallel?

Here, spherical is defined as a geometry where gas covers some fraction of 4π sr as seen from the central object. If the central object has a total luminosity L then the gas intercepts $L\Omega/4\pi$ of the radiation field, where $\Omega/4\pi$ is the gas covering factor. The luminosities (erg s^{-1}) of the emission lines are predicted in this case.

A plane-parallel geometry is one where the problem is solved per unit area of cloud surface. A starting radius is not specified and the surface emittance (erg cm^{-2} s^{-1}) of emission lines is predicted.

If the code can determine an inner radius then it will predict emission-line luminosities. Otherwise it will predict the emittance, the emission per unit area. This is described further in 2.3 of Part 1 of HAZY.

2.2.1 Commands normally used

`radius` This sets the inner radius of the cloud. Only then can line luminosities be predicted.

`covering factor` This sets the covering factor of the cloud. The predicted emission-line luminosities scale linearly with the covering factor.

2.2.2 Heads up!

2.3 Open or closed geometry

Chapter 2 of Part 1 of HAZY defines open and closed geometries. An open geometry is the default. If the geometry is closed, one where gas covers most of the continuum source, then the **`sphere`** command (described in Chapter 9 GEOMETRY of Part 1) should be included. This affects the transfer of the diffuse fields. This physics changes the final predictions at the ~10% level.

For a closed geometry you should specify whether the shell is static or expanding (the default). This affects the emissivity of some lines and is described further in Part 1 of HAZY where the **`sphere`** command is discussed.

2.3.1 Commands normally used

`sphere [expanding, static]` This tells the code to assume a closed geometry, one where diffuse continuous emission from the “far side” of the central hole in a spherical geometry is included. By default 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 then lines escaping from one side can be absorbed by gas on the far side.

2.3.2 Heads up!

These considerations affect the transport of the diffuse fields and have only second-order effects on the predicted spectrum or ionization of the gas. If you are uncertain, try the same simulation with and without the **`sphere`** command, and with both **`sphere static`** and **`sphere expanding`**, to see whether this makes any difference. It usually does not.

2.4 Is the gas static or a wind? Is it turbulent?

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

`turbulence` – this adds a component of microturbulence to line broadening. This will make line pumping by the incident continuum more important and line trapping less important.

`wind` – The cloud is assumed to be static by default. This command simulates an expanding wind. The equations of motion determine the velocity as a function of depth.

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

The calculation must stop for some reason. Many different stopping criteria can be specified and the code will stop when the first one is satisfied. Cloudy was originally designed to interpret optical / UV emission lines in quasars. These lines are produced in warm ionized gas, so the default is for the calculation 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 include cool atomic or molecular regions.

You should understand what sets the outer edge of the cloud and then confirm that the code reached this point. The introduction to Chapter 12 STOPPING CRITERIA of Part 1 of HAZY goes into this in more detail.

2.5.1 *Commands normally used*

`radius` This sets the inner and outer radii of the cloud.

`stop temperature` This sets the lowest temperature to allow. The calculation stops when the electron temperature falls below this.

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

`stop column density` This sets a limiting column density.

`double` This doubles the computed line optical depths at the end of an iteration. This command should be used if the region being calculated is only a layer on a much larger structure. This is the case, for instance, in a PDR calculation, where an unmodeled molecular cloud lies beyond the PDR. This tells the code to double the computed line optical depths at the end of an iteration, so that lines at the shielded face of the computed structure are quite optically thick. This is described further in Chapter 10 OPTICAL DEPTHS AND RADIATIVE TRANSFER of Part 1 of HAZY.

2.5.2 *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 code aborted? 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 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.
```

3 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 Chapter 7 CHEMICAL COMPOSITION of Part 1 of HAZY. The default composition is close to that of the solar atmosphere 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. Constant density will be assumed by default.

3.1 Chemical composition

3.1.1 Commands normally used

`abundances` This sets the abundances of all elements to some standard mixture. Some examples include the local ISM or a typical planetary nebula. The mix may include classical grains but will not include PAHs.

`element` This is used to set the abundance of a particular element, turn if off, or set its ionization state.

`grains` This determines the type and number of grains. If grains are turned on 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.

3.1.2 Heads up!

Grain sublimation: A warning will be printed if grains are included and become hotter than their sublimation point. A note will be printed if grains are not present but would have been cooler than their sublimation temperature.

Gas-phase abundances and grains: It is possible to leave the composition near solar, where 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. The code will complain if you do this, but will go ahead with the calculation.

PAH abundances: There is good observational evidence that PAHs only exist near the H^0 – H^+ ionization front. They are thought to be destroyed in the H^+ region and coagulate into larger grains in molecular regions. The keyword **`function`**, an option on the **`grains`** command, makes the PAH abundance depend on the ratio $n(H^0)/n(H_{tot})$.

Quantum heating: This physics is computationally very expensive. The simulation will run significantly faster if quantum heating is disabled with the **`no qheat`** keyword on the **`grains`** or **`abundances`** commands. Quantum heating only affects the continuum in the 1 – 10 μm region. This continuum will be under predicted if quantum heating is turned off but the rest of the calculation should still be valid.

Turn off elements with trivial abundances. The calculation will run faster if insignificant or highly depleted elements are turned off with the **`element xxx off`** command.

3.2 What is the cloud’s density? Does it vary with depth?

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 or include a magnetic field in the pressure law, or to vary the density explicitly through the cloud. Most of these are discussed in Chapter 8 DENSITY LAWS of Part 1 of HAZY.

3.2.1 Commands normally used

hden This specifies the log of the hydrogen density. By default, the density does not depend on depth.

constant pressure This tells the code to assume constant pressure. The keyword **gas** says to keep the gas pressure, rather than the total pressure, constant.

filling factor The gas is normally assumed to fully fill the available space. This command sets a filling factor *f*. In this case the gas is assumed to be highly clumped, so that much of the volume is a vacuum. The parameter *f* is the fraction of the volume that contains gas.

magnetic field These are normally ignored. Magnetic fields can be included and will contribute to the total pressure and, optionally, to the turbulent velocity field. This is discussed in Chapter 10 THERMAL SOLUTIONS of Part 1 of HAZY.

3.2.2 Heads up!

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

$$n(H) = n\left(H^o\right) + n\left(H^+\right) + +2n\left(H_2\right) + \sum_{other} n\left(H_{other}\right) \quad [\mathrm{cm}^{-3}]$$

where the sum is over all other molecules that contain hydrogen. In nearly all cases hydrogen will mostly be in one of the first three forms.

4 The incident radiation field

The radiation field is specified by its shape, which describes its frequency dependence, and by its intensity or luminosity. Chapter 4 DEFINING THE CONTINUUM of Part 1 of HAZY gives an overview of how this is done. More than one continuum source can be included in the net radiation field. The following first describes how to specify the intensity of the radiation field, and the next section describes setting its shape.

4.1 The luminosity / intensity of the radiation field

There are two cases. 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, are both specified. The luminosities of emission lines are then predicted. In the ***intensity case***, where the flux of energy (erg cm^{-2} s^{-1}) or photons (cm^{-2} s^{-1}) is specified, the inner radius does not need to be given, and the line emittance (erg cm^{-2} s^{-1}) is predicted. Section 5.2 of Part 1 of HAZY describes this distinction.

There are many ways to specify the continuum intensity or luminosity. These are described in Chapter 5 CONTINUUM LUMINOSITY of Part 1 of HAZY.

For many luminosity or intensity commands the default is for the quantity to be integrated over hydrogen-ionizing energies. Most of these commands also have the **`range`** option, which allows this energy range to be changed.

4.1.1 Commands normally used

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

`phi(H)` This is the per unit area (cm^{-2} s^{-1}) equivalent of the **`Q(H)`** command. It gives the flux of photons striking the face of the cloud.

`luminosity` This specifies the luminosity emitted by the central object into 4π sr (erg s^{-1}).

`intensity` This is the per unit area (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 gives the dimensionless ratio of densities of ionizing photons to hydrogen. 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, so emission lines will be predicted per unit area.

4.1.2 Heads up!

4.2 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 derived from built-in tables. Methods of setting the shape are described in Chapter 6 CONTINUUM SHAPE of Part 1 of HAZY.

The shape should be specified between the code's energy limits of 10 m and 100 MeV, if possible. The code will complain but compute the model if this is not done.

4.2.1 Commands normally used

`interpolate` The continuum shape will be set by interpolating on a table giving pairs of frequency – flux points.

`fireball` This adds the cosmic microwave background for any redshift.

`blackbody` The shape will be a blackbody. If the **`LTE`** option also appears then the intensity is also specified to make the radiation field a true blackbody.

`background` This is a simple estimate of the X-ray – UV background at high redshift.

`table` These are a series of continua that are specified as built-in tables. Some examples include the AGN background at redshift z = 2, some stellar continua, and the local ISM galactic background.

`table AGN` This uses the Mathews & Ferland (1987) mean quasar continuum.

`table HM96` This uses the Haardt & Madau (1996) background, which may be more accurate than that used in the `background` command.

`table ISM` This gives the diffuse background within the local ISM.

`extinguish` This 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 region has extinguished much of the incident radiation field.

`cosmic ray background` This will include the effects of galactic background cosmic rays. These are baryons, not photons, of course. It is important to include these if the calculation extends into predominantly molecular gas. The chemistry of the cold ISM is driven by a series of ion-molecule reactions. The required ions will not be present if no source of ionization is present and the reaction network will collapse. The code will complain (but try to compute the model) if the calculation extends into cool regions without background cosmic rays.

4.2.2 Heads up!

Some shape commands also specify the intensity. An example is `table HM96` – that command specifies both the shape and intensity of the background. Commands that set both are listed in Section 4.3.2 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 background radiation and local ISM diffuse continuum, the two commands described next.

The ***cosmic microwave background should always be included***, either with the `fireball` or `background` commands. This is not done by default.

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

Intensity and shape commands should be kept together for simplicity.

5 Other commands

5.1 Radiative transfer

5.1.1 Commands normally used

`Case B` This tells the code to artificially set 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. Case B is the limit where all Lyman lines scatter often enough to be degraded into Lα and Balmer lines. 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.

`turbulence` and **`wind`** These commands are discussed on page 10 above. The code assumes that only thermal motions broaden lines unless a component of motion is added with one of these commands.

5.1.2 Heads up!

The ***Case B command has many artificial side-effects***. It should not be used except in special test cases.

5.2 Miscellaneous

5.2.1 Commands normally used

`atom` This provides a way to change the detailed treatment of some model atoms. It has many keywords – **`H-like`** and **`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 and Fe II emission.

`init` It is possible to store frequently used commands in a special initialization file. The **`init`** command will include the contents of that file within the input stream. The name of the initialization file appears on the command line within a pair of double quotes. This is described in Section 3.6 of Part 1 of HAZY.

Comments: These can be included within the input stream. Section 3.4.5 of Part 1 of HAZY describes their syntax.

5.2.2 Heads up!

Only one init file can occur within an input stream.

6 The code’s predictions

6.1 The default printout

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

```
cloudy.exe < input > output
```

the code’s standard output, which is written to *stdout* in C, will go to the output file, “*output*” in this example. This output is fully described in Chapter 4 OUTPUT of Part 3 of HAZY. The default output includes a copy of the input commands, a list of the abundances of the elements, the physical conditions in the first and last zone, the intensity or luminosity of the stronger emission lines, and the mean ionization, temperature, and column density of many species.

6.1.1 Commands normally used

`title` The remainder of the line is a title that is printed at various places.

`print` This command changes 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 optical spectroscopy the normalization line is usually Hβ and this is the default. Other lines can be specified with this command.

6.1.2 Heads up!

Understand why the calculation stopped. This is given in the first comments after the last zone. This is described further in the introduction to Chapter 12 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 2.5.2 on page 11 above.

6.2 Warnings, cautions, surprises

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. If there are problems the code will say so by issuing warnings (these are comments that start with "W-"), cautions (comments that start with "C-"), or surprises (these start with "!"). This is described further in section 4.7 of Part 3 of HAZY. Warnings are a major source of concern – 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.

6.3 Observed quantities

Chapter 5 OBSERVED QUANTITIES of Part 3 of HAZY explains how to relate quantities predicted by the code to observed properties. Chapter 6 THE EMISSION LINES of Part 3 of HAZY gives a (slightly out of date) overview of the labels used to indicate various emission lines.

6.4 Punch output

There is far too much information generated in a typical calculation for even a small part of it to be included in the main printout. Instead, a series of **`punch`**[3] commands that are used to create ancillary files that contain various predictions.

[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.

These are described in Chapter 13 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.

6.4.1 Commands normally used

`punch continuum` This gives the incident, transmitted, and total continua. The energy axis is normally in Rydbergs but the **`units`** option allows other units, including keV, microns or Angstroms, to be used instead.

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

`punch molecules` The densities of a large number of molecular species are output as a function of depth into the cloud.

7 Examples

7.1 The test suite

The test suite is a large body of simulations of various astrophysical environments. 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 tests all contain **`assert`** commands. These tell the code the answer to expect. Use the Perl script *runall.pl* to run all simulations in the test suite. The script *checkall.pl* will go through the output and confirm that all quantities had their expected value. If predicted quantities differ by more than some amount, 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. They can be removed if you change models parameters so that results change too. The script *tests_remove_asserts.pl* will do this.

7.2 One of the models …

This section considers *orion_hii_pdr_pp.in*, one of the models in the test suite, in detail. This simulates a plane-parallel molecular cloud with an H II region and PDR on its surface. A nearby O star and galactic background cosmic rays are the only sources of heat and ionization. The calculation begins with the H II region, an ionized layer with a temperature of $T \sim 10^4$ 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 document the commands used to set up the simulation. This section discusses the various output files created by the simulation.

7.2.1 *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 and a main output file called *orion_hii_pdr_pp.out*.

7.2.2 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 likely had pressure convergence failures where the cloud passed through a ***thermal front***. Thermal fronts, and the problems they cause, are described in Chapter 8 PROBLEMS of Part 3 of HAZY. Convergence problems are announced with lines that begin with the string "PROBLEM". This is one of the few simulations in the test suite that has convergence problems.

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. The command **`print last`** would tell the code to only print results of the last iteration, but is not used here. 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 of this 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 the version of the code you ran with its historical predicted quantities.

Chapter 4 of Part 3 of HAZY goes over the output in detail. Have a look.

7.2.3 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 Chapter 13 of Part 1 of HAZY while Chapter 5 of Part 3 of HAZY explains how to extract some observed quantities from all of this output.

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

orion_hii_pdr_pp.con: This file is produced by the **`punch continuum`** command and gives the incident, reflected, transmitted, and total continua. These are defined in Chapter 2 of Part 1, and are described in both Chapter 13 of Part 1 and Chapter 5 of Part 3 of HAZY. The **`units`** option changed the energy scale from Rydbergs (the default) into 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, since dust extinction by 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}.

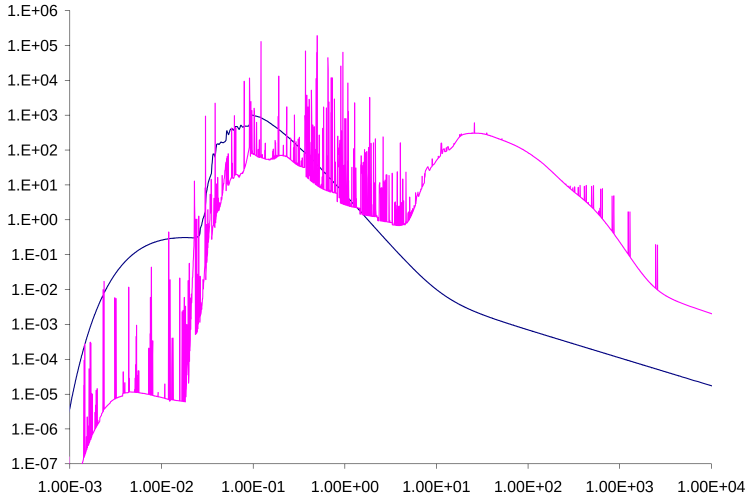

Figure 1 Contents of the *orion_hii_pdr_pp.con* file, produced by the **`punch continuum`** command

orion_hii_pdr_pp.ovr: This file is produced by the **`punch overview`** command. It gives the electron temperature and density, the hydrogen density, the heating rate, and the ionization distribution for H, He, C, and O. Figure 2 shows the hydrogen ionization 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.

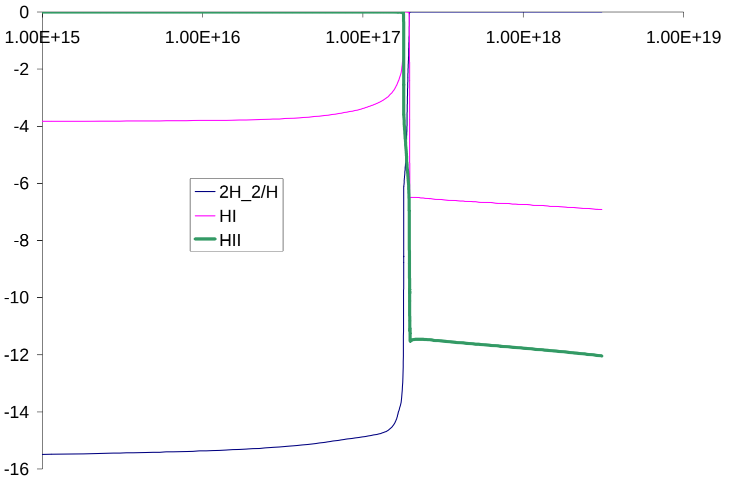

Figure 2 – the hydrogen ionization structure. The x-axis is the depth in cm and the y-axis gives the log of the fraction of H in H^+, H^0, and H_2.

orion_hii_pdr_pp.grntem; This 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. It 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.

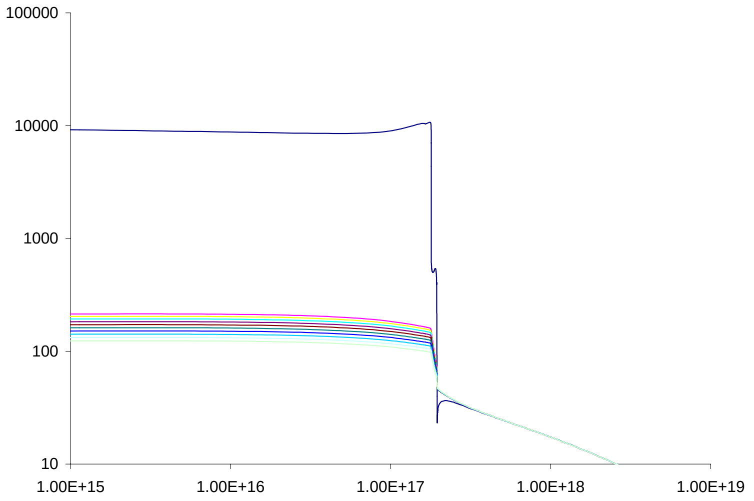

Figure 3 – temperatures of grains (the lower cluster of curves) and electrons (the higher curve). They become nearly equal in the molecular cloud.

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

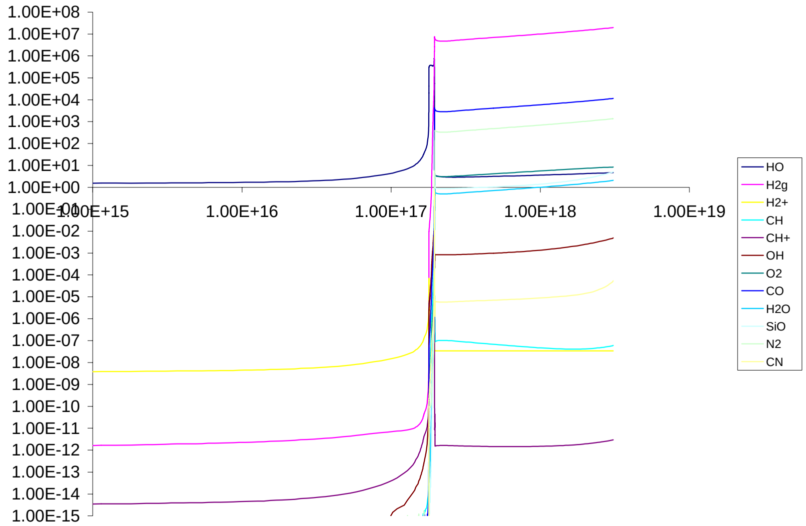

Figure 4 – densities of the molecules across the cloud.

several measures of extinction due to dust.

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

7.3 Heads ups for classes of objects

The simulations in the test suite fall into several broad categories of objects. Following subsections describe important considerations for setting up these simulations.

7.3.1 BLR

Their density is high enough for free-free absorption 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 (1999; ASP Conf Series 162, astro-ph 0307450, available here).

7.3.2 NLR

Does dust exist in the ionized gas? Depletion patterns are not clear, as discussed by Ferguson et al. (1997, ApJ, 487, 122, available here).

7.3.3 PDRs

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 since these molecules tend to have much smaller activation barriers than interactions between neutral particles. 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 11 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 15 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 populates excited levels of hydrogen. This is called "Case C" in the ionized cloud community (see Ferland 1999, PASP, 111, 1528, available here). Excited levels are then photoionized by relatively soft radiation. Classical PDR calculations do not consider this physics. It can be disabled by telling the code to assume that the Lyman lines are very optically thick so that these lines are self-shielded. Do this with the **case b** command (page 15 above). The **case b** command should not be included in any realistic simulation (which should start with the H II region and extend into the PDR).

7.3.4 Galactic nebulae

Cosmic rays and the ISM background should probably be included. Dust is almost certainly present in the ionized gas, and should be included, along with depleted abundances of the refractory elements.

8 References

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, 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

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