User's manual

The 3D-PDR code

Thomas G. Bisbas

Serena Viti, Michael J. Barlow,
Jeremy Yates, Tom Bell

University College London

Contents

1 Introduction

The 3D-PDR code is the first fully three dimensional code dedicated for treating Photodissociation Regions (PDRs). It is a 3D version of the one-dimensional code UCL_PDR which have been implemented by T. Bell at University College London (UCL) (Bell et al., 2006). The full details of how 3D-PDR works can be found in Bisbas et al. (2012). The code has been implemented also at UCL during 2010-2012. It can treat any given density distribution in 1D, 2D and 3D, either from SPH-based or from grid-based data, or any other user-specified data set.

The code has been also integrated with the Adaptive Mesh Refinement hydrodynamical and radiative transfer code TORUS, named now TORUS-3DPDR (Bisbas et al., 2015), and which treats both the photoionized and the photodissociation region in full three-dimensions, making it the first ever code that can tackle complete astrochemical problems in diffuse HII regions of arbitrary density distribution. This can create three-dimensional synthetic observations which can be used in order to compare hyrdodynamical simulations with observational data cubes obtained by ALMA, JVLA, *Herschel* and other ground-based, air-borne, and space-borne instruments. Please contact the author for more information if needed.

2 Installing 3D-PDR in your machine

3D-PDR is a combined FORTRAN 90/95 and C++ `OpenMP` code. You will therefore need FORTRAN and C compilers. The preferred compilers for the code, however, are `gfortran` version 4.8.4 or `ifort` version 14.0.4 for FORTRAN and `gcc` for C++. Later versions of these compilers may require adjustments in the code and in particular in the connection with the SUNDIALS package.

2.1 The SUNDIALS package

The code uses the SUNDIALS package to solve the Ordinary Differential Equations of the given chemical network. Hence you will need to have SUNDIALS pre-installed on your system to link the corresponding libraries with the 3D-PDR abundance solver. SUNDIALS V.2.5.0 are included in the tarball, however you can also download the package from here:

http://computation.llnl.gov/casc/sundials/main.html

Later versions of SUNDIALS may require further adjustments in the code. You do not need superuser access to install SUNDIALS. To start the installation you will need to change directory to `~/3DPDR/sundials-2.5.0` and then type the command below:

- for `ifort` compiler:

```
./configure CC=gcc F77=ifort --with-cflags="-fopenmp"
      --with-ldflags="-fopenmp" --with-fflags="-openmp"
      --prefix="ABSOLUTE-HOME-PATH/3DPDR/sundials/"
```

- for `gfortran` compiler:

```
./configure CC=gcc F77=gfortran  --with-cflags="-fopenmp"
      --with-ldflags="-fopenmp" --with-fflags="-fopenmp"
      --prefix="ABSOLUTE-HOME-PATH/3DPDR/sundials/"
```

After the above configuration has been finished, you need to type (still in the `~/3DPDR/sundials-2.5.0` directory!):

```
make
```

and then

```
make install
```

This will proceed in installing SUNDIALS in the directory specified in `--prefix` flag. Once you have successfully installed SUNDIALS you will be able to proceed with the main 3D-PDR installation. Failure to install SUNDIALS will make impossible to proceed any further.

Please refer to the `INSTALL_NOTES` file found the `~/3DPDR/sundials-2.5.0` directory for full description of the installation process.

2.2 The `makefile` of the code

The `makefile` comes with the following default options:

```
#compiler options
F90               = ifort
CPPFLAGS          = -cpp
INCLUDES          = -I/ABSOLUTE-PATH-TO/3DPDR/sundials/include
LIBRARIES         = -L/ABSOLUTE-PATH-TO/3DPDR/sundials/lib
LIBS              = -lsundials_cvode -lsundials_nvecserial -lm
OPTIMISE          = 0
OPENMP            = 1
#main code
DIMENSIONS        = 1
NETWORK           = REDUCED
DUST              = 2
GUESS_TEMP        = 1
THERMALBALANCE    = 1
TEMP_FIX          = 1
CO_FIX            = 1
H2FORM            = 1
```

IMPORTANT: Do not leave even a single space at the end of each of the above lines!. You will need to replace the `INCLUDES` and `LIBRARIES` with the path in which SUNDIALS have been installed.

The user can specify the optimisation level (`OPTIMISE`); the parallel (`OPENMP = 1`) or the serial (`OPENMP = 0`) scheme; the number of dimensions to be considered in the simulation (`DIMENSIONS = 1,2,3`) which compiles the code as 1D, 2D, or 3D respectively; the chemical network that will be used (`NETWORK = REDUCED, FULL, MYNETWORK` – see §3 for further details). For `DUST = 2` dust treament in the simulation is included. The `GUESS_TEMP = 1` flag gives initial temperatures in the simulation using Eqn.(24) of Bisbas et al. (2012). Please refer to §2.8 of that article for further details. `THERMALBALANCE = 1` indicates iterations to obtain thermal balance by equating the heating and cooling functions. By using `THERMALBALANCE = 0` the user is able to run an isothermal simulation. In this case, you may set `GUESS_TEMP = 0` and specify the temperature in the `params.dat` file (see §4). `H2FORM = 1` enables detailed treatment of the H_2 formation rate (it is recommended to keep this flag value in all simulations). Setting it to 0, it will use the H_2 formation routines as used in the Roellig et al. (2007) benchmarking workshop.

Once you specify your preferences you simply need to type "`make`" or "`make -j`" to compile the code. The latter compiles the code in parallel, hence faster (not to be confused with parallelization of the code itself. This command simply uses all CPUs to *compile* the code).

As soon as the code has been successfully compiled and linked with SUNDIALS, the executable "`3DPDR`" will be generated. Please refer to §8 for the set of the default benchmaking tests. The code is ready to be used!

3 Choosing chemical network and abundances

3.1 Chemical network

3D-PDR comes with a set of two chemical networks: the 'reduced' chemical network (`species_reduced.d` containing 33 species including e^-) and the 'full' chemical network (`species_full.d` containing 215 species including e^-), both which are (sub-)sets of the UMIST2012 network (McElroy et al., 2013). In both cases there are the respective files `rates_reduced.d` and `rates_full.d`. Each of the network corresponds to different set of ordinary differential equations which are solved using the C++ `odes_reduced.c`, `jacobian_reduced.c` and `odes_full.c`, `jacobian_full.c` files respectively and which are linked with the SUNDIALS solvers.

The users are also welcome to use their own chemical networks. To do that, you need to create your own versions of the above files and name them as follows: `species_mynetwork.d`, `rates_mynetwork.d`, `odes_mynetwork.c`, `jacobian_mynetwork.c`. In order to build your own chemical network, you will need to use the self-consistent code MAKERATES kindly provided by T. Bell.

To choose any of the above chemical networks, you will need to compile the code according to your preference. In the `makefile`, you need to change the `NETWORK` flag to either `REDUCED`, `FULL`, or `MYNETWORK`. Make sure you "`make clean`" before executing "`make`".

3.2 Abundances

The default abundances of 3D-PDR in the `reduced` network are those described below. Users are free to use their own abundances in each network simply by editing the `species_NETWORK.d` file.

Species	Abundance
Mg	0.0
O	3.0×10^{-4}
C	1.0×10^{-4}
H	1.00
He	1.0×10^{-1}

4 The `params.dat` file

The `params.dat` file is the one in which the user specifies the input file, prefix of the output file, intensity of UV field etc.. In general this file is self-explanatory, however we list below the most important parameters that are frequently used from each block.

First block:

```
INPUT_FILE                !Input file (20 char. max.)
OUTPUT_PREFIX             !Output file (20 char. max)
0                         !HEALPix level of refinement
0.8                       !Theta critical (0<phi<pi/2)
8                         !First set of Chemical Iterations
6000                      !Total iterations
0.9                       !min number density (cm^-3)
1e12                      !max number density (cm^-3)
5.0E-17                   !Cosmic Rays (s^-1)
1.0                       !Turbulent velocity (km/s)
20.0                      !Dust temperature (K)
1e7                       !End time (yr)
```

```
100.0                    !Gas-to-dust
1.0                      !Metallicity
0.42                     !Omega
1.0E-7                   !Grain radius (m)
```

Here, in the INPUT_FILE and OUTPUT_PREFIX need to have up to 20 characters maximum.

The HEALPIX level of refinement has an effect *only* in 2D and 3D runs, whereas in 1D needs to be $\ell = 0$, otherwise the code won't proceed. For the `Theta critical` value of 0.8 please refer to §3.1 of Bisbas et al. (2012). For the 8 `First set of Chemical iterations` please refer to §2.7 of Bisbas et al. (2012).

The min and max values of number densities corresponds to the cut-off density region which the user defines as PDR. Densities below the min value are considered as 'ionized'. Densities above max are considered as 'molecular'. In both cases 3D-PDR *does not* consider these computational elements during the thermal balance iterations. The default values are $n_{\rm min} = 0.9\,{\rm cm}^{-3}$ and $n_{\rm max} = 1e12\,{\rm cm}^{-3}$.

The `Dust temperature` value of 20 is only effective when `DUST = 0` in the `makefile` (see §2.2). The `End time` defines the chemical time for which 3D-PDR calculates the reactions. In general equilibrium is achieved in any time above 100,000 years, however it is recommended to use the default value of 10 Myears.

The `gas-to-dust` value corresponds to the gas-to-dust ratio. The default value is 100. The default value for `Metallicity` is 1 (i.e. Solar metallicity). `Omega` and `Grain radius` correspond to dust properties.

Second block:

```
12co.dat                 !CO
12c+.dat                 !CII
12c.dat                  !CI
16o.dat                  !OI
```

The above files are used during the calculations of the cooling function. Unless the users have their own versions, these lines should not change.

Third block:

```
10.0                     !Initial temperature (used when TEMP_GUESS is OFF)
10.0                     !Tlow (used when TEMP_GUESS is OFF)
8000.0                   !Thigh (used when TEMP_GUESS is OFF)
10.0                     !Tmin (absolute lower bound)
30000.0                  !Tmax (absolute upper bound)
0.005                    !Fcrit (% Accuracy)
0.01                     !Tdiff (maximum temperature difference)
```

In the above block, the first three lines are used when the guess-temperature function (Eqn. 24 in Bisbas et al., 2012) is OFF (a `makefile` flag/option). The particular first line defines the temperature to be considered in an isothermal run (`THERMALBALANCE = 0` and `GUESS_TEMP = 0` simultaneously in the `makefile` – §2.2). `Tmin` and `Tmax` define the absolute minimum and maximum values that the gas temperature can have during the thermal balance iterations.

`Fcrit` defines the accurancy (percentage) of the thermal balance change, below which 3D-PDR considers the calculations as 'converged'. A secondary criterion is the `Tdiff` which is the absolute difference between two consecutive changes of gas-temperature and below which the code considers the calculations as 'converged'. All values in this third block can remain unchanged for almost all simulations performed.

The final, fourth, block is dedicated to the description of the UV radiation field. See the next section for details.

5 Choosing UV radiation field

In the code, the user is able to use different types of UV radiation fields. The units of the radiation field is the [Draine] unit (Draine, 1978). These are: the `UNI` for plane-parallel field of uniform-direction, and the `ISO` for an external isotropic radiation field. In 1D simulations only the `UNI` is applicable whereas all of the types are applicable in 3D simulations.

In the `params.dat` file, the fourth block of parameters are dedicated to the description of the radiation field. The first three lines read:

```
UNI                         !ISO-tropic, UNI-directional, or PoiNT source
10                          !G0 x-direction (in Draine field units)
```

In 1D simulations, only the `G0 x-direction` line is applicable. The radiation field is assumed to be impinging from the $-x$ side (i.e. its direction is from $-x$ to $+x$). The user simply needs to use a negative number in case a radiation impinging from the $+x$ side of the domain is required.

6 Constructing initial conditions

3D-PDR reads in ASCII files which have columns of `x,y,z,n`. The $x-, y-, z-$ axes are in [pc] and the n density is in [cm^{-3}]. In 1D simulations, the user has to include the position along $x-$ axis, while $y-$ and $z-$ are set to 0.

The 3D-PDR tarball comes with a set of 1D initial conditions of different densities from $n = 10^2 - 10^6\,\mathrm{cm}^{-3}$. The filenames are `1DnXX.dat`, where `XX` is the logarithm of the density. For example `1Dn34.dat` is the file with $n = 10^{3.4} = 2511.9\,\mathrm{cm}^{-3}$. All these files have minimum resolvable visual extinction $AV_{\mathrm{min}} = 10^{-6}$ and maximum $AV_{\mathrm{max}} = 20$, while the resolution is logarithmic and set to 40 elements per AV dex. These files can be considered as high-resolution 1D initial conditions.

7 Outputs

The outputs are written once the code has been fully converged and finished. The code produces outputs with different suffixes according to i) abundances of species (`.pdr.fin`), ii) line emission (`.line.fin`), iii) cooling functions (`.cool.fin`), iv) heating functions (`.heat.fin`), v) level populations (`.spop.fin`), and vi) optical depths (`.opdp.fin`). In every case the outputs are ASCII format and they are ready to be plotted using standard tools such as GNUPLOT or PYTHON.

In the following subsections we give the columns for each file when 3D-PDR has been compiled either as a 1D or as a 3D code. (I) indicates an integer number. All other values are real.

7.1 Compiled as 1D code

Columns for `OUTPUT.pdr.fin`

Column	Value	Comment
1	ID	(I) Identity of the point
2	x	x-Cartesian co-ordinate (pc)
3	AV	Visual extinction (mag)
4	$T_{\rm gas}$	Gas temperature (K)
5	$T_{\rm dust}$	Dust temperature (K)
6	$etype$	(I) computational element type
7	n	Local number density (cm^{-3})
8	UV	Local intensity of the radiation field (Draines)
9-end	[X]	Abundance of species, following `species_NETWORK.d` file

From column 9 onwards, the abundances of species for the chemical network in question are written. The sequence of the species follows the sequence of the `species_NETWORK.d` file i.e. column 9 is the first species of the .d file, column 10 is the second one etc..

Columns for `OUTPUT.cool.fin`

Column	Value	Comment
1	ID	(I) Identity of the point
2	x	x-Cartesian co-ordinate (pc)
3	AV	Visual extinction (mag)
4	CII	Cooling function for CII
5	CI	Cooling function for CI
6	OI	Cooling function for OI
7	CO	Cooling function for CO
8	Total	Total cooling function (sum of columns 4–7)

Columns for `OUTPUT.heat.fin`

Column	Value	Comment
1	ID	(I) Identity of the point
2	x	x-Cartesian co-ordinate (pc)
3	AV	Visual extinction (mag)
4	HR01	NOT CONTRIBUTING
5	HR02	Photoelectric heating
6	HR03	NOT CONTRIBUTING
7	HR04	Carbon ionization heating
8	HR05	H_2 formation heating
9	HR06	H_2 photodissociation heating
10	HR07	FUV pumbing heating
11	HR08	Cosmic-ray heating
12	HR09	Turbulent heating
13	HR10	Chemical heating
14	HR11	Gas-grain heating
15	Total	Total heating (sum of contributing heating functions)

Columns for `OUTPUT.line.fin` (local emissivities). Emissivities are in units of [erg/cm^{-3}/s].

Column	Value	Comment
1	ID	(I) Identity of the point
2	x	x-Cartesian co-ordinate (pc)
3	AV	Visual extinction (mag)
4	L01	CII 158μm emissivity
5	L02	CI 609μm emissivity
6	L03	CI forbidden emissivity
7	L04	CI 370μ emissivity
8	L05	OI 63μ emissivity
9	L06	OI forbidden emissivity
10	L07	OI 146μ emissivity
11	L08	CO(1-0) emissivity
12	L09	CO(2-1) emissivity
...	...	...
21	L18	CO(10-9) emissivity

To obtain the line emission from the above file, one needs to integrate along the line of sight. To convert the integrated emissivity units to [erg/cm^{-2}/s/sr], you need to divide the emission by 2π.

Columns for `OUTPUT.spop.fin` (level populations). Level populations are in units of [cm^{-3}].

Column	Value	Comment
1	ID	(I) Identity of the point
2	AV	Visual extinction (mag)
3–7	CII	CII 5 levels
8–12	CI	CI 5 levels
13–17	OI	OI 5 levels
18–59	CO	CO 41 levels

Columns for `OUTPUT.opdp.fin` (Optical depths). This is a dimensionless quantity.

Column	Value	Comment
1	ID	(I) Identity of the point
2	AV	Visual extinction (mag)
3	L01	CII 158μm
4	L02	CI 609μm
5	L03	CI forbidden
6	L04	CI 370μ
7	L05	OI 63μ
8	L06	OI forbidden
9	L07	OI 146μ
10	L08	CO(1-0)
11	L09	CO(2-1)
...	...	...
20	L18	CO(10-9)

7.2 Compiled as 3D code

The files follow similar format as in the 1D version, with the addition of y- and z- cartesian co-ordinates and the AVs along all HEALPIX directions. More specifically:

Columns for `OUTPUT.pdr.fin`

Column	Value	Comment
1	ID	(I) Identity of the point
2	x	x-Cartesian co-ordinate (pc)
3	y	y-Cartesian co-ordinate (pc)
4	z	z-Cartesian co-ordinate (pc)
5	$T_{\rm gas}$	Gas temperature (K)
6	$T_{\rm dust}$	Dust temperature (K)
7	$etype$	(I) computational element type
8	n	Local number density (cm^{-3})
9	UV	Local intensity of the radiation field (Draines)
10–(species+10)	[X]	Abundance of species, following `species_NETWORK.d` file
Species+11–end	AV	Visual extinction along each HEALPIX direction

The columns for AV have the same number as the number of the HEALPIX level of refinement specified by the user. At level ℓ there are $N_\ell = 12 \times 4^\ell$ HEALPIX rays, the sequence of which is followed as given by the `NESTED` scheme (see Górski et al., 2005, for more details). For convenience, the sequence at $\ell = 0$ is as follows: rays with ID=0–3 are in North pole ($z > 0$), ID=4–7 are in the Equator ($z = 0$), ID=8–11 are in the South pole ($z < 0$).

In a 3D run, any densities that are lower than $n_{\rm min}$ are considered as ionized and any densities above $n_{\rm max}$ are considered as molecular. In both cases there are no PDR calculations involved. However, 3D-PDR writes out those grid cells in two different files, namely `OUTPUT.ion.fin` and `OUTPUT.mol.fin` respectively. The columns of those ASCII files are the same as above.

Columns for `OUTPUT.cool.fin`

Column	Value	Comment
1	ID	(I) Identity of the point
2	x	x-Cartesian co-ordinate (pc)
3	y	y-Cartesian co-ordinate (pc)
4	z	z-Cartesian co-ordinate (pc)
5	CII	Cooling function for CII
6	CI	Cooling function for CI
7	OI	Cooling function for OI
8	CO	Cooling function for CO
9	Total	Total cooling function (sum of columns 5–8)
10–end	AV	Visual extinction along each HEALPIX direction

Columns for `OUTPUT.heat.fin`

Column	Value	Comment
1	ID	(I) Identity of the point
2	x	x-Cartesian co-ordinate (pc)
3	y	y-Cartesian co-ordinate (pc)
4	z	z-Cartesian co-ordinate (pc)
5	HR01	NOT CONTRIBUTING
6	HR02	Photoelectric heating
7	HR03	NOT CONTRIBUTING
8	HR04	Carbon ionization heating
9	HR05	H_2 formation heating
10	HR06	H_2 photodissociation heating
11	HR07	FUV pumbing heating
12	HR08	Cosmic-ray heating
13	HR09	Turbulent heating
14	HR10	Chemical heating
15	HR11	Gas-grain heating
16	Total	Total heating (sum of contributing heating functions)
17–end	AV	Visual extinction along each HEALPIX direction

Columns for `OUTPUT.line.fin` (local emissivities)

Column	Value	Comment
1	ID	(I) Identity of the point
2	x	x-Cartesian co-ordinate (pc)
3	y	y-Cartesian co-ordinate (pc)
4	z	z-Cartesian co-ordinate (pc)
5	L01	CII 158μm emissivity
6	L02	CI 609μm emissivity
7	L03	CI forbidden emissivity
8	L04	CI 370μ emissivity
9	L05	OI 63μ emissivity
10	L06	OI forbidden emissivity
11	L07	OI 146μ emissivity
12	L08	CO(1-0) emissivity
13	L09	CO(2-1) emissivity
...	...	...
22	L18	CO(10-9) emissivity
23–end	AV	Visual extinction along each HEALPIX direction

Columns for `OUTPUT.spop.fin` (level populations). Level populations are in units of [cm^{-3}].

Column	Value	Comment
1	ID	(I) Identity of the point
2	x	x-Cartesian co-ordinate (pc)
3	y	y-Cartesian co-ordinate (pc)
4	z	z-Cartesian co-ordinate (pc)
5–9	CII	CII 5 levels
10–14	CI	CI 5 levels
15–19	OI	OI 5 levels
20–61	CO	CO 41 levels

Columns for `OUTPUT.opdp.fin` (Optical depths). This is a dimensionless quantity.

Column	Value	Comment
1	ID	(I) Identity of the point
2	x	x-Cartesian co-ordinate (pc)
3	y	y-Cartesian co-ordinate (pc)
4	z	z-Cartesian co-ordinate (pc)
5	L01	CII 158μm
6	L02	CI 609μm
7	L03	CI forbidden
8	L04	CI 370μ
9	L05	OI 63μ
10	L06	OI forbidden
11	L07	OI 146μ
12	L08	CO(1-0)
13	L09	CO(2-1)
...	...	...
22	L18	CO(10-9)

8 Benchmarking tests

In order to check whether or not the code has been properly installed, we provide below some default runs for 1D and 3D cases. These all use the `REDUCED` chemical network with the default abundances and the default parameters found in the `params.dat` file.

8.1 1D tests

We include four different benchmarking tests. The initial conditions and the parameters of each test are given in the table below. In order to perform these tests you need to compile the code as 1D, i.e. `DIMENSIONS = 1` in the `makefile` (see §2.2). Figures 1 – 5 show the results of those tests. The output of the tests can be found in the `~/3DPDR/Benchmarks/` directory.

Filename	$n_{\rm H}\,({\rm cm}^{-3})$	UV field (Draines)	$\zeta_{\rm CR}\,({\rm s}^{-1})$	Output prefix
`1Dn40.dat`	10^4	1	1.3×10^{-17}	`B01`
`1Dn20.dat`	10^2	1	1.3×10^{-17}	`B02`
`1Dn40.dat`	10^4	1	1.3×10^{-15}	`B03`
`1Dn40.dat`	10^4	10^3	1.3×10^{-17}	`B04`

8.2 3D test

We provide only one three-dimensional benchmaking test (prefix `3DB01`) in which we consider a sphere with radius $R = 0.6\,{\rm pc}$ and which has uniform density distribution with $n = 10^4\,{\rm cm}^{-3}$. The initial conditions of that sphere can be found in the `~/3DPDR/Code/` directory (file is called `sphere.dat`). To perform this test you will need to compile the code as 3D, i.e. `DIMENSIONS = 3` in the `makefile` (§2.2)

The sphere is interacting with an external radiation field emitting $\chi_0 = 10^3\,{\rm Draines}$. We can therefore compare the results of this run with the B04 test described previously. Figure 6 shows the results and the comparison with B04. Users are free to plot the results using GNUPLOT and color-code each computational element along with their preference[1].

Contact

Dr. Thomas G. Bisbas
University of Florida
tbisbas@ufl.edu

Acknowledgements

The development of the code has been funded by STFC grants ST/H001794/1 and ST/J001511/1.

References

Asplund, M., Grevesse, N., Sauval, A. J., & Scott, P. 2009, ARA&A, 47, 481

Bell, T. A., Roueff, E., Viti, S., & Williams, D. A. 2006, MNRAS, 371, 1865

Bisbas, T. G., Bell, T. A., Viti, S., Yates, J., & Barlow, M. J. 2012, MNRAS, 427, 2100

Bisbas, T. G., Bell, T. A., Viti, S., et al. 2014, MNRAS, 443, 111

Draine, B. T. 1978, ApJS, 36, 595

Górski, K. M., Hivon, E., Banday, A. J., et al. 2005, ApJ, 622, 759

McElroy, D., Walsh, C., Markwick, A. J., et al. 2013, A&A, 550, AA36

[1] i.e. to plot the sphere in 3D and color-code according to the temperature, you will need to type in GNUPLOT: `splot '3DB01.pdr.fin' u 2:3:4:5 palette`

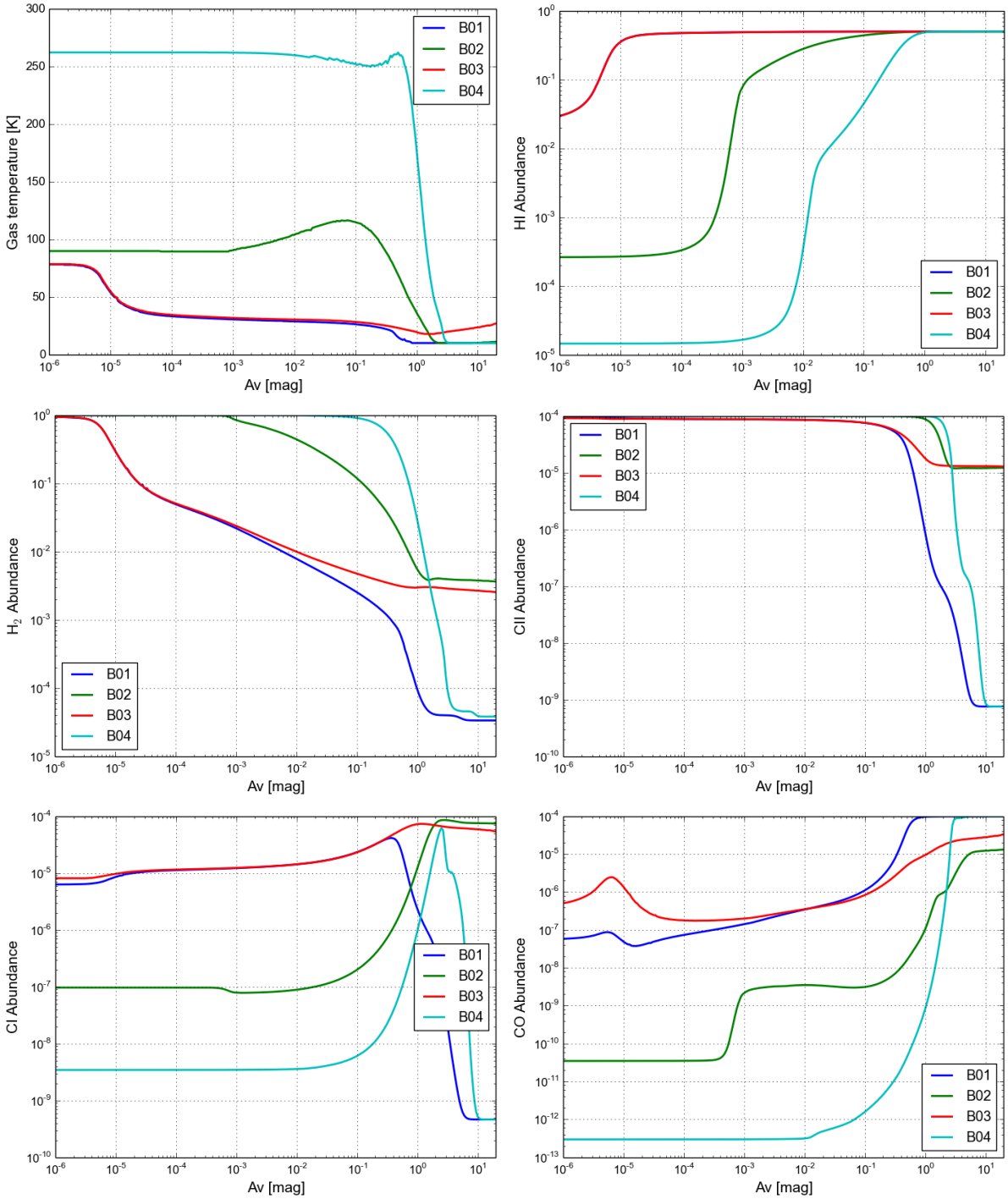

Figure 1: Results plotted using the `OUTPUT.pdr.fin` file. In all cases the x-axis plots the visual extinction AV. From top left to bottom right: Gas temperature, HI abundance, H_2 abundance, CII abundance, CI abundance, and CO abundance.

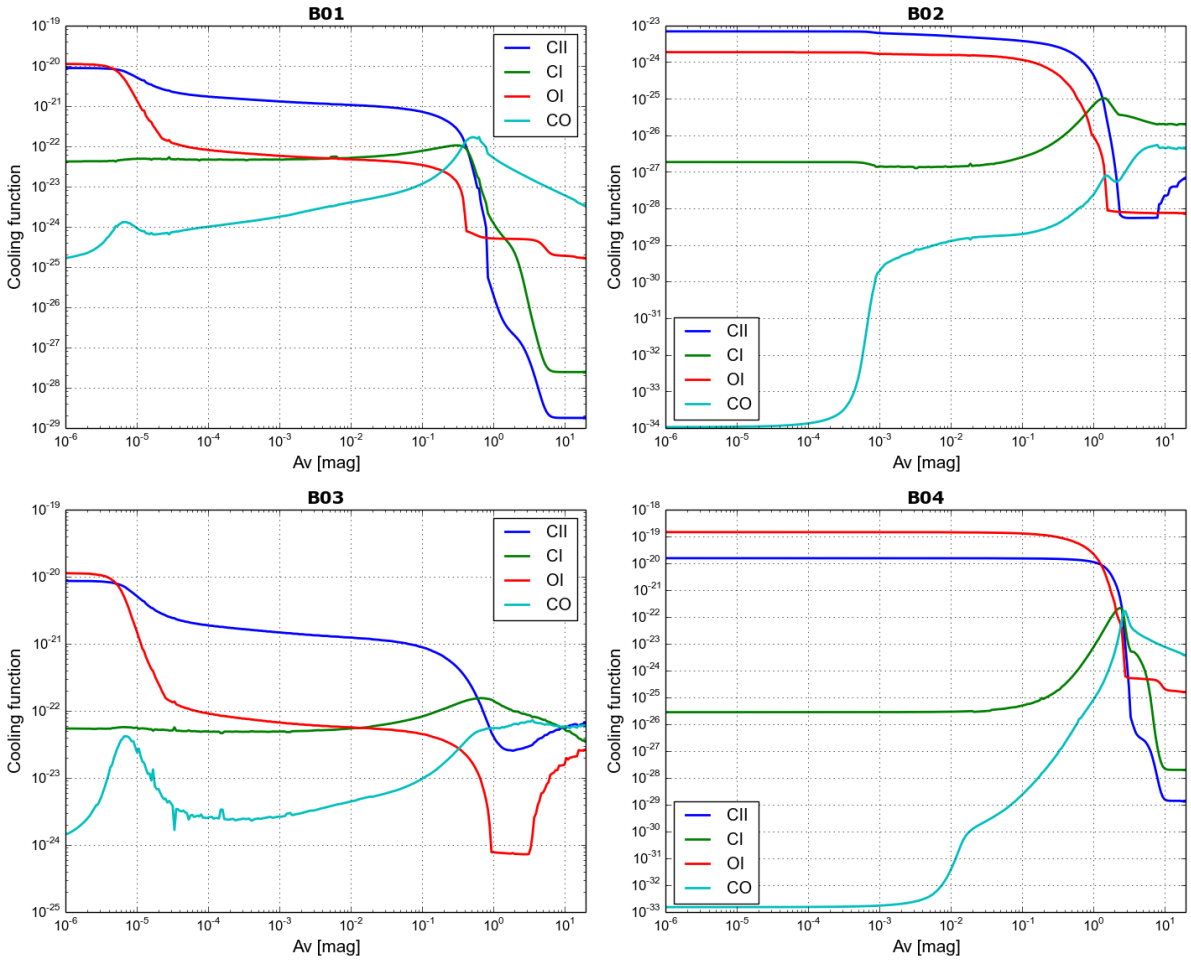

Figure 2: Results plotted using the `OUTPUT.cool.fin` file. In all cases the x-axis plots the visual extinction AV. From top left to bottom right: test B01, B02, B03, B04.

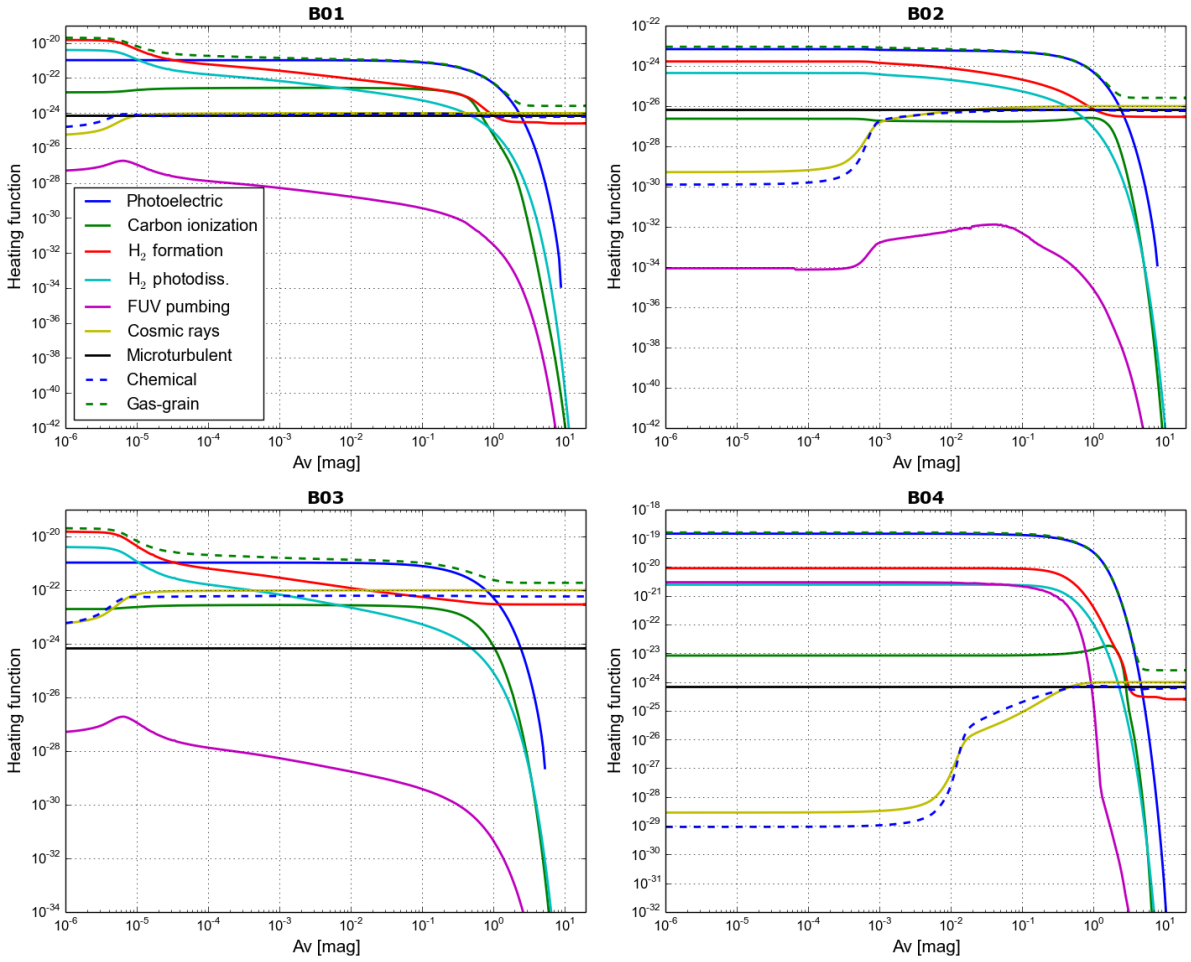

Figure 3: Results plotted using the `OUTPUT.heat.fin` file. In all cases the x-axis plots the visual extinction AV. From top left to bottom right: test B01, B02, B03, B04.

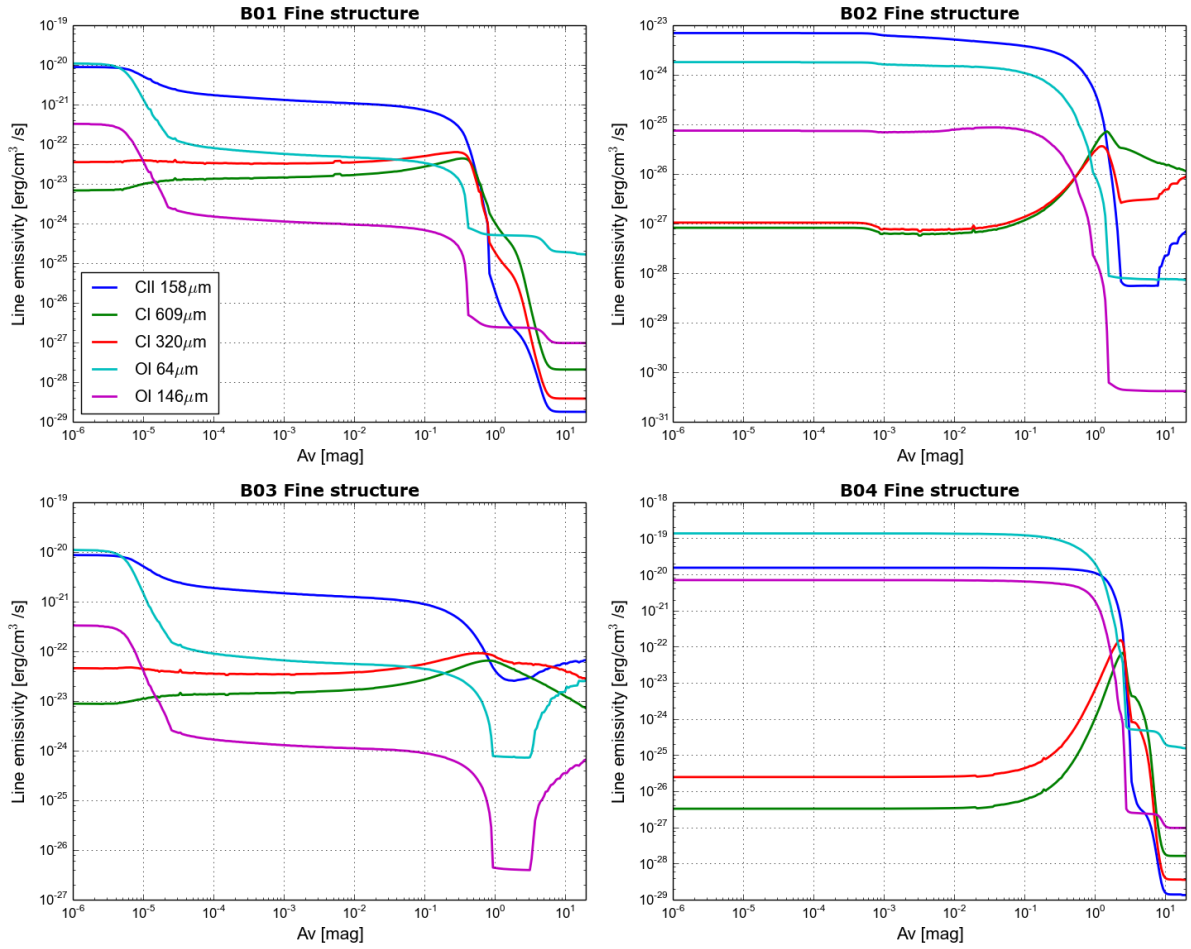

Figure 4: Results plotted using the `OUTPUT.line.fin` file. Here we plot only the fine-structure lines excluding the forbidden lines. In all cases the x-axis plots the visual extinction AV. From top left to bottom right: test B01, B02, B03, B04.

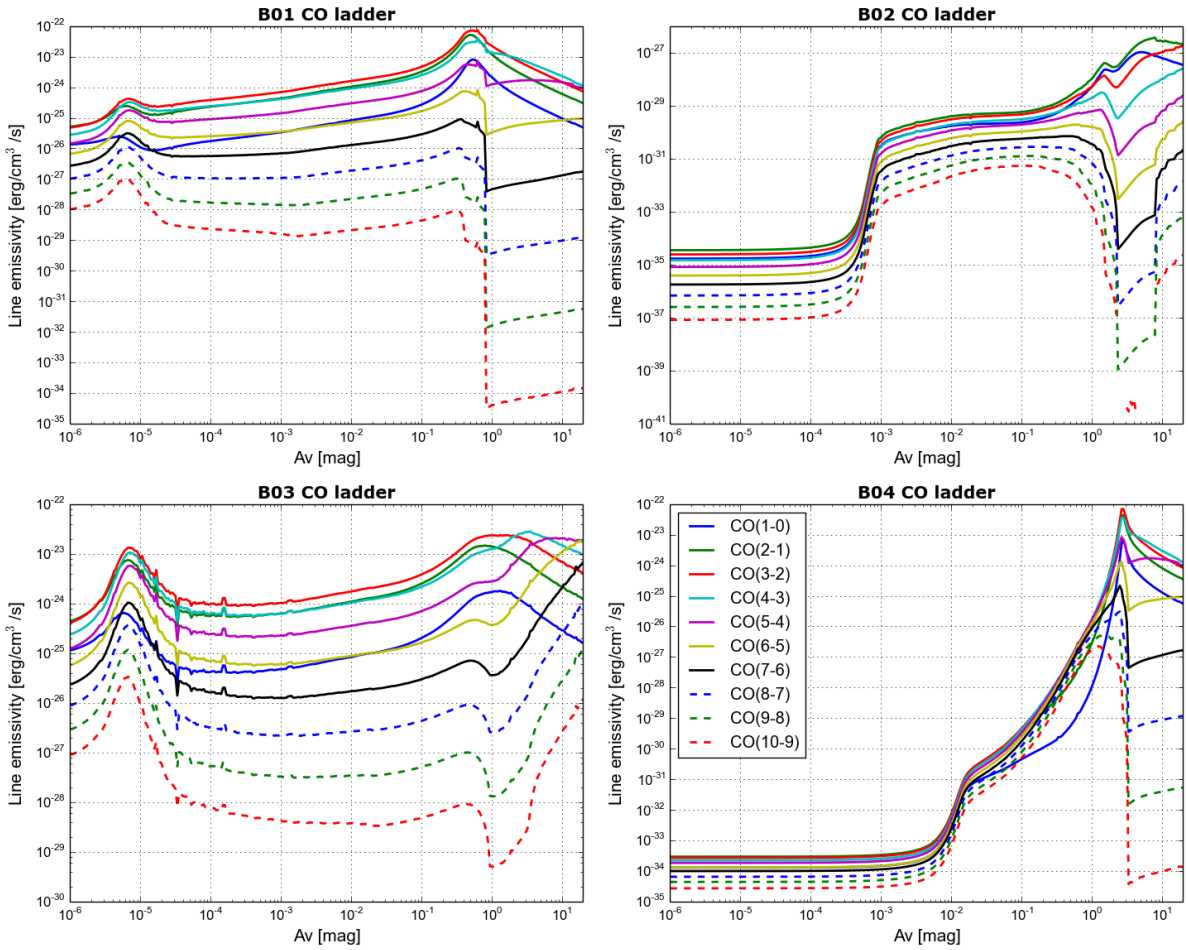

Figure 5: Results plotted using the `OUTPUT.line.fin` file. Here we plot only the CO ladder. In all cases the x-axis plots the visual extinction AV. From top left to bottom right: test B01, B02, B03, B04.

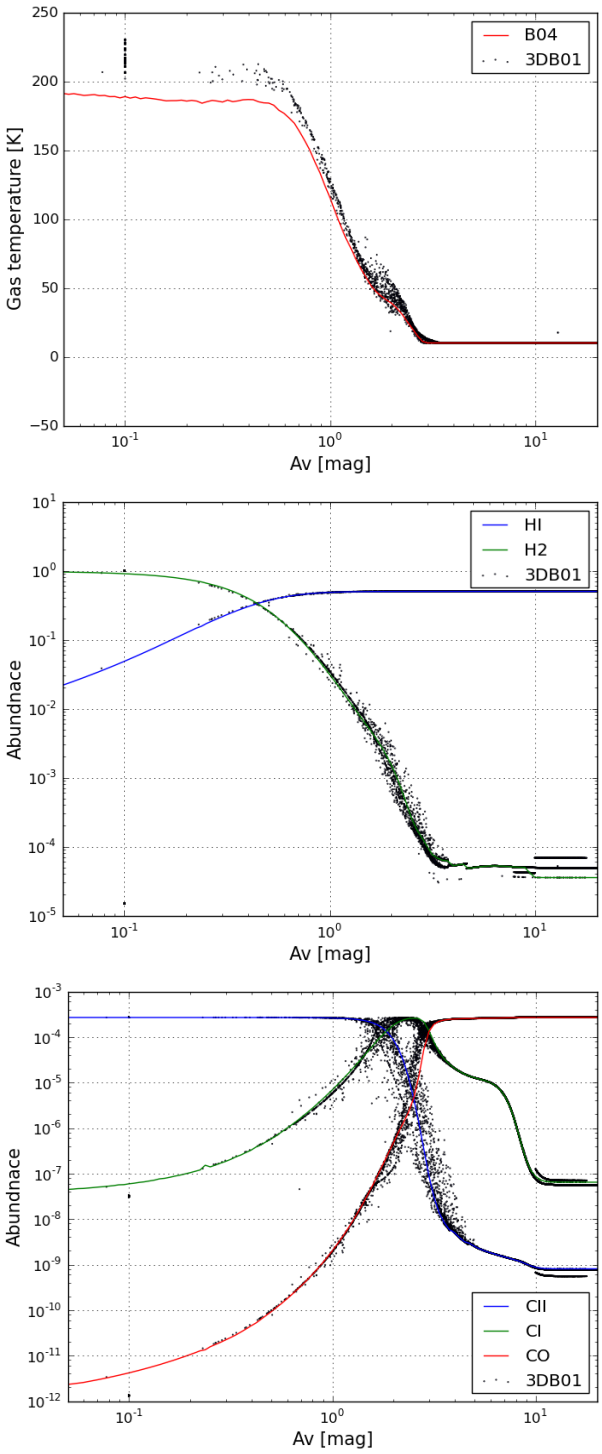

Figure 6: Comparison of the 3DB01 run with B04. Since the UV source is located in the $-x$ axis, we need to plot the AV along that direction. This corresponds to column 49 of the given `3DB01.pdr.fin` file. From top to bottom: gas temperature, HI-to-H_2 transition, CII/CI/CO transition.

Offner, S. S. R., Bisbas, T. G., Viti, S., & Bell, T. A. 2013, ApJ, 770, 49

Offner, S. S. R., Bisbas, T. G., Bell, T. A., & Viti, S. 2014, MNRAS, 440, L81