1 General comment

In the program parts that are only called once at the start of the program, I changed all the in-code comment/uncomment blocks into input paramters instead (for example for the power law source: giving a BH mass or a total number of ionizing photons). However, the parts which are called for every cell (like for example doric: turning of/on the recombination photons in doric), I left as comment/uncomment block in the code. If one wants to use the simpler version, one does anyway use the "complicated" way of calculating things: for example, even if one does not want to use secondary ionizations, the fractions going into heat and secondary ionizations are first calculated but then everything is given to the heating. The reason is that I do not know a good way of both: avoiding double code (of course, one could just have two different routines) and at the same time not using if statements.

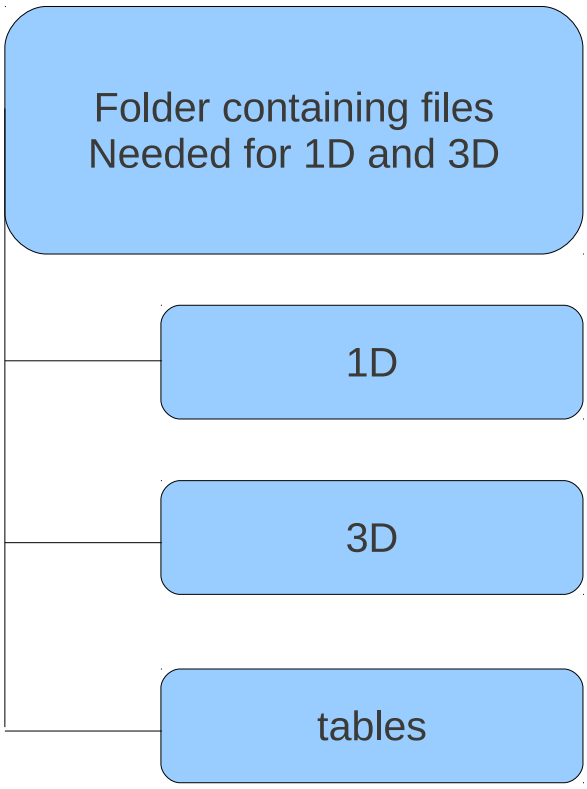

Figure 1: Structure of folders and files

2 Order of files

The files are structured as follows:

The main folder contains the files summarized in Table 1. Some modules are located in more than one file, of course, only one of them is used (specified in the Makefile). This folder contains the files that are identical for both, the 1D and the 3D version and those that are only used in one of them.

The folders `files_for_1D` and `files_for_3D` contain the set of files that are used in the 1D and 3D version, but which are not identical. This is summarized in Tables 2 and 3.

The folder `tables` contains the different cooling tables.

3 Cooling tables

We introduced more cooling points (compared with the version before, a factor 10) since the interpolation is done linearly in log temperature, thich leads to steps, or "ripples" in the temperature otherwise. This works, but perhaps it is not the best solution.

See matlab script `cooling_data.m` for a plot of the cooling data we are using (according to Garrelt the ones from Hummer, see comments in matlab script) against cooling data from other references.

Table 1: Files in the main directory.

module	**files**	
abundances	abundances.f90	
atomic	atomic.f90	
c2ray_parameters	c2ray_parameters.f90	c2ray_parameters_TEST4.f90
astroconstants	cgsastroconstants.f90	
cgsconstants	cgsconstants.f90	
cgsphotoconstants	cgsphotoconstants.f90	
clocks	clocks.f90	
radiative_cooling	cooling.f90	cooling_h.f90
cosmology	cosmology.f90	
cosmology_parameters	cosmoparms.f90	cosmoparms_EoRKP.f90
	cosmoparms_test4.f90	cosmoparms_WMAP1.f90
	cosmoparms_WMAP3.f90	cosmoparms_WMAP3plus.f90
	cosmoparms_WMAP5.f90	
m_ctrper	ctrper.f90	
nbody	cubep3m.F90	gadget.F90
	pmfast.F90	test4.F90
	test.F90	
doric_module	doric.f90	
file_admin	file_admin.f90	
mathconstants	mathconstants.f90	
my_mpi	mpi.F90	no_mpi.F90
m_mrgrnk	mrgrnk.f90	
precision	precision.f90	
radiation	radiation_including_pls_new.F90	radiation_monocromatic.F90
romberg	romberg.f90	
string_manipulation	string.f90	
thermalevolution	thermal.f90	
times	time.F90	time_ini.F90
tped	tped.f90	

Table 2: Files in the directory `files_for_1D`.

module	**files**
< program file >	C2Ray.F90
< make file >	Makefile
evolve	evolve_new.F90
grid	grid.F90
material	mat_ini.F90
output_module	output.f90
photonstatistics	photonstatistics.f90
sizes	sizes.f90
sourceprops	sourceprops_test_one_source.F90

Table 3: Files in the directory `files_for_3D`.

module	**files**	
< program file >	C2Ray.F90	
< make file >	Makefile	
evolve	evolve8.F90	
grid	grid.F90	
material	mat_ini_cubep3m_compr.F90	mat_ini_cubep3m.F90
	mat_ini_Gadget.F90	mat_ini_LG.F90
	mat_ini_pmfast_compr.F90	mat_ini_pmfast.F90
	mat_ini_test4.F90	mat_ini_test.F90
output_module	output_compr.F90	output.F90
photonstatistics	photonstatistics_compr.f90	photonstatistics.f90
sizes	sizes.f90	
sourceprops	sourceprops_cubep3m_compr.F90	sourceprops_cubep3m.F90
	sourceprops_gadget.F90	sourceprops_LG.F90
	sourceprops_pmfast_compr.F90	sourceprops_pmfast.F90
	sourceprops_test2.F90	sourceprops_test4.F90
	sourceprops_test.F90	

4 Radiation

There are three main bins in radiation. Currently, radiation is set up in a way that it can handle (1,2,3,6,26) sub-bins in bin 2 and (1,4,9,11,16,20) sub-bins in bin 3. Of course, one can chose whatever number, the only things one has to do is to specify the frequency bin boundaries and the fit-parameters to the optical depth fitting function to the data from Verner. See matlab script `my_cross_section2.m` that gives not only the cross section fits but also the parameters for the secondary ionizations, see comment below. This script writes out all parameters that need to be specified in the code, in an already formatted way, so one just needs to copy those lines at the appropriate place in radiation. It also plots the janev data and the fits to it used in each sub-bin in three different plots, one for each bin. The parameters in the radiation module are set in a subroutine called `setup_scalingfactors`, which is located at the very end of the file. The frequency boundaries are also given there, may be one should use the test that Chael Kruip (?) did, on the optimal frequency bin distribution?

At the moment, the parameters for the secondary ionizations have not been implemented for all frequency intervals. However, see Ricotti et al 2002 for the determination of the parameters and comments in the code 1599ff how to implement them or use the matlab script `my_cross_section2.m` which gives directly the desired parameter values when the frequency bin boundaries are given. The only thing you need to do is to comment and uncomment the desired frequency bin boundaries in the "speicification section" at the beginning of the script. For non-isothermal, only the (bin 2, bin 3) pair (26,20) is currently implemented. If you do not want to use secondary ionizations, see the blocks 1400-1406 and 1464-1470 for information.

You specify the number of sub-bins at the beginning of *radiation* where also the number of table points and frequency integration points is given.

5 Doric

For tests with only A or B recombination coefficients, uncomment the appropriate block in lines 145-161. The different y and z fractions are calculated outside doric in evolve (or evolve8 in the 3D case) and passed as a routine parameter to doric. The other fractions (w,p,..) are calculated or given in doric, however, they should be perhaps moved to one of the "constants" modules, may be to the same that contains ini_colion_factors (cgsconstants).

6 Recombination rates and other temperature dependent parameters

Those are collected in the routine `ini_rec_colion_factors`. See there or in Friedrich et al. 2012 for references of the fitting functions. This routine is called first in `mat_ini` to initialize those values corresponding to the initial temperature from node 0.

For the moment, `evolve0D_global` is only called by node 0 (the call to `do_chemistry` on the first iteration is turned off). However, if in the future, this part of the calculation is also parallized, here is

most probably a problem with the routine `ini_rec_colion_factors`: Probably, one should change the `ini_rec_colion_factors` routine in such a way that it has the desired parameters as output parameters instead. This might bring the problem that vfrac might have be included in the input paramters of doric.

7 Material, nbody and sourceprops

Since I haven't worked with the corresponding files for gadget, pmfast, LG, (and pmfast_comp, cubep3m_compr), they might lack adjustments.

8 Comments on LLS

Since they were included in C^2-Ray somewhat in parallel to helium, I added the changes afterwards in my version. It might not work, haven't tested it.