# Monte Carlo Photoionization Code

## 1 General stuff

Photoionization code described in Wood, Mathis, & Ercolano (2003) for low density HII regions. Code calculates the ionization structure of a 3D linear Cartesian grid which is illuminated by point sources within the grid. Outputs are the ionization and temperature structure and an array containing the emissivities for many optical and infrared lines of the ions tracked in the code. The code tracks the ionization structure of H, He, C, N, O, Ne, and S - the most important coolants in low density HII regions.

The code only follows photons with energies between 13.6eV and 54eV (ionization potential of $He^+$), since in general there is not observed to be any $He^{+2}$ in low density HII regions and there is little ionizing radiation above 54eV from the atmospheres of normal O and B stars. Assumes lowest ionization stages of C and S are $C^+$ and $S^+$, i.e., C and S are singly ionized by interstellar radiation field.

Atomic data, recombination rates, and emissivities are as described in the WME paper.

The code iterates to convergence, initially assuming the grid is highly ionized (neutral fraction around 1e-6). After the first few iterations the cells farthest from the sources receive the lowest mean intensity, so become more neutral. The code uses constant temperature for first four iterations and then computes temperature structure at each iteration after that. First seven iterations are done with $10^6$ photons and this number increases at further iterations to give better signal to noise. No convergence checks are performed, rather code is run for many iterations and successive iterations compared by eye. Typically a dozen iterations is sufficient for convergence... but not guaranteed!

## 2 Inputs: input.params

- nphotons - number of photons for first seven iterations. Increased by factors of ten for later iterations.
- iseed - initial seed for random number generator (ran2.f)

- niter - maximum number of iterations
- length - length unit in cm, e.g., if code works in pc then length = 3.086e18
- rgrid - grid is a cube of side 2*rgrid*length
- Tconst - uniform temperature in all cells for initial iterations
- AHe, AC, AN, AO, AS, ANe - abundances of He, C, N, O, Ne, S
- Tstar - temperature if assuming a blackbody ionizing spectrum
- atmosfile - stellar atmosphere file for the ionizing spectrum
- Q49 - total ionizaing luminosity of all sources in units of $10^{49}$ photons per second.
- bbody - set this integer to 1 for blackbody and 2 for model atmosphere for the ionizing spectrum.
- pahfac - factor that accounts for gas heating by PAHs

# 3 Code outputs

- input.spectrum.dat - input ionizing spectrum (photons/sec/Q49)
- ntot.dat - number density grid ($cm^{-3}$)
- rr.dat - grid that contains the physical coordinates of the centre of each grid cell. I use this in the ionfracs.pro IDL program to plot ionization fractions and temperature.
- temp.dat - grid of electron temperature
- specout.dat - file containing the spectrum leaking out of the model HII region.
- emiss.dat - 4D array (nx, ny, nz, nlines) containing emissivity for various emission lines. Look in the subroutine lines.f and you'll see the list of emissivities that are produced.

- nfracH.dat, nfracHe.dat - 3D grids (nx, ny, nz) of neutral fractions of H and He
- fracC.dat - 4D grid (nx, ny, nz, nions) of ionization fractions of $C^{+}$, $C^{+2}$, $C^{+3}$.

# 4 Setting up and running code

Type 'make' and this produces the executable 'ionize'. Then type 'ionize' and the code should start running. After each iteration the code outputs the files listed above. You can then use the IDL programs to view the ionization fractions (ionfracs.pro) and an example of plotting diagnostic diagrams (ratios.scatter.pro). The simplest way to make spatially resolved intensity maps is to sum the emissivity grid along one axis.

Start off running the code using a blackbody spectrum. The parameters file (params.par) is set up for the HII40 benchmark described in Wood et al. (2004) and references therein. I'd run it as is and have a look at the output using the IDL programs and watch it converge. You'll get OK looking results in a few minutes (after eight iterations). Running for ten or more iterations gives better S/N and will take about an hour... on my laptop.

# 5 Subroutines

Subroutines that can be modified relatively safely:

- sources.f & sources.txt - sets up number, locations, and relative luminosities of point sources.
- gridset.f & density.f - sets up grid, density structure within grid, and initializes ionization fractions and temperatures.