Numerical Project: Monte-Carlo modelling of supernovae spectra.

February 16, 2015

1 Introduction

In this report I present a Monte-Carlo (MC) based code for modelling supernovae spectra, as well as tests of the code and comparisons to other similar codes. The code consists of a generic and a specific part, the generic part being independent of the geometry and the methods used to calculate the state of the matter, which allow different geometries and methods to be implemented. The supernovae (SN) ejecta is located on a grid, subdivided into cells, holding the state of the matter and the radiation field. The radiation field is determined by (repeated) MC experiments, where indestructible and indivisible radiation (energy) packets are propagated on the grid following the methods described by Lucy (2005). The radiation packets interacts with the matter through electron scattering and line absorption/emission. The latter process may be treated as scattering or as fluorescence, in which case the macro-atom method by Lucy (2002, 2003) is used to ensure indestructibility and indivisibility of the radiation packets. The code assumes steady-state and a homologously expanding ejecta, and does not solve thermal equilibrium. Instead, the temperature is assumed to be controlled by the radiation field, parametrized as a dilute blackbody. The ionizated and excited states are assumed to be populated according to local thermal equilibrium (LTE), the nebular approximation (Abbott & Lucy 1985; Mazzali & Lucy 1993) or statistical equilibrium. Currently, the latter method is only available for the population of excited states, but the generic design of the code allows for full statistical equilibrium as well as thermal equilibrium to be implemented in the future.

2 Physical processes

2.1 λ-iterations

The basic principle used to determine the state of the matter and the radiation field is the so called λ-iterations, where the state of the matter and the radiation field are alternately and iteratively determined from each other. Such a scheme is known to be notoriously hard to converge because of the non-local coupling provided by the radiation field. However, as demonstrated by Lucy (2002), enforcing the constraint of energy conservation in the co-moving frame by the use of indestructible MC radiation (energy) packets, convergence is greatly improved and is routinely achieved in just a few iterations.

2.2 State of the matter

The state of the matter is given by the dynamical and thermal state of the gas. As steady state is assumed, the thermal time-scales are assumed to be much smaller than the dynamical time-scale, and the dynamical state only affects the calculation through the velocity field, which is assumed to be given by homologous expansion, where $v = r/t$. The thermal state is given by the density and temperature, and the populations of ionized and excited states, and is assumed to be controlled by the radiation field, parametrized as a dilute blackbody (Sect. 2.3). Following Mazzali & Lucy (1993), the gas and radiation temperatures are assumed to be related as $\mathrm{T_G} = 0.9\mathrm{T_R}$, and to determine the populations of ionized and excited states we use either local thermal equilibrium (LTE) at the radiation temperature, the nebular approximation (Abbott & Lucy 1985; Mazzali & Lucy 1993), or statistical equilibrium assuming that the radiative rates dominates. Note that the latter choice is only implemented for the population of excited states.

2.2.1 Statistical equilibrium

In general, the NLTE rate equations needs to be solved to determine the populations of ionized and excited states. Assuming steady state, these equations simplifies to the equations of statistical equilibrium, where the total rate of transitions in and out of each state is zero. However, this problem is still a quite demanding task to solve, and the code only provides the limited option to solve the statistical equilibrium equations for the excited states assuming that the radiative rates dominates. In this case the equations of statistical equilibrium simplifies to

$$\Sigma\Lambda_{l,i}n_l - (\Sigma\Lambda_{i,l} + \Sigma\Lambda_{i,u})n_i + \Sigma\Lambda_{u,i}n_u \tag{1}$$

where

$$\begin{aligned} \Lambda_{u,l} &= (A_{u,l} + B_{u,l}J_{u,l})\,\beta_{l,u} \\ \Lambda_{l,u} &= B_{l,u}\,\beta_{l,u}J_{u,l} \end{aligned} \tag{2}$$

and $A_{u,l}$, $B_{u,l}$ and $B_{l,u}$ are the Einstein coefficients for spontaneous emission, stimulated emission and absorption, respectively, β the Sobolev escape fraction and $J_{u,l}$ the mean intensity at the transition wavelength. Ignoring the coupling to the radiation field (which is handled through λ-iterations), and adding the number conservation equation, this equation system constitutes a non-linear (as the Sobolev escape fraction depends on the populations) equation system in the populations of the excited states, which has to be solved numerically (Sect. 4.7).

2.2.2 Local thermal equilibrium

In LTE the matter and the radiation field are assumed to be coupled and the radiation field to be that of a blackbody at the temperature of the gas. In this case the populations of ionized states is given by the Saha ionization equation

$$\frac{n_{\mathrm{i+1}} n_{\mathrm{e}}}{n_{\mathrm{i}}} = 2\frac{(2\pi m_{\mathrm{e}} k_{\mathrm{B}} T)^{3/2}}{h^3} \frac{U_{\mathrm{i+1}}}{U_{\mathrm{i}}} \exp\left(-\frac{\chi_{\mathrm{i}}}{k_{\mathrm{B}} T}\right) \tag{3}$$

where n_e is the electron number density and χ_i and U_i are ionization energy and partition function for ionization state i, respectively. Together with the equations for number and charge conservation, the Saha ionization equation give rise to a non-linear equation system in the populations of the ionized states, which has to be solved numerically (Sect. 4.7). For a particular ion, the populations of excited states are given by the Boltzman excitation equation

$$\frac{n_{\mathrm{i+1}}}{n_{\mathrm{i}}} = \frac{g_{\mathrm{i+1}}}{g_{\mathrm{i}}} \exp -\left(\frac{E_{\mathrm{i+1}} - E_{\mathrm{i}}}{k_{\mathrm{B}} T}\right) \tag{4}$$

where E_i and g_i are the energy and statistical weight for excitation state i. Note, however, as the code is intended for use in the optically thin regime, and as the LTE populations are evaluated at the radiation temperature of a diluted blackbody, LTE does not necessarily apply, and the actual level populations might be rather different (see Sect. 5.2 and Sect. 5.3).

2.2.3 Nebular approximation

The nebular approximation is based on the assumption that the radiative rates dominates, and from this Mazzali & Lucy (1993) and Abbott & Lucy (1985) derive approximations for the populations of the ionized and excited states, respectively.

Ionization Basically, the populations are calculated from the Saha equation at the radiation temperature, but reduced by the dilution factor. However, recombinations to excited states also has to be taken into account and the nebular approximation becomes

$$\frac{n_{\mathrm{i+1}} n_{\mathrm{e}}}{n_{\mathrm{i}}} = W(\zeta + W(1-\zeta))\left(\frac{T_G}{T_R}\right)^{1/2} \left(\frac{n_{\mathrm{i+1}} n_{\mathrm{e}}}{n_{\mathrm{i}}}\right)^*_{T_R} \tag{5}$$

where W is the dilution factor, ζ the fraction of recombinations going into the ground state, and the asterisk refers to the populations given by LTE (Sect. 2.2.2). Mazzali & Lucy (1993) also describes how to handle optically thick ionization continua (e.g. at wavelengths below the Lyman break), where conditions are better described by LTE, and in that case an additional factor δ, described by Mazzali & Lucy (1993) enters the equation. This use of this method is included as a configurable choice in the code. As in the case of LTE, the nebular approximation give rise to a non-linear equation system in the populations of the ionized states, which has to be solved numerically (Sect. 4.7).

Excitation As argued by Mazzali & Lucy (1993) metastable levels are expected to be populated as in LTE at the radiation temperature if the radiative rates dominates. In addition, Mazzali & Lucy (1993) argues that states lying closely above metastable levels, can be assumed to be populated as in LTE at the radiation temperature, but reduced with the dilution factor, so for these levels the nebular approximation gives

$$\frac{n_{\mathrm{i,j+1}} n_{\mathrm{e}}}{n_{\mathrm{i,j}}} = W\left(\frac{T_G}{T_R}\right)^{1/2} \left(\frac{n_{\mathrm{i+1}} n_{\mathrm{e}}}{n_{\mathrm{i}}}\right)^*_{T_R} \tag{6}$$

where W is the dilution factor, n_e the electron number density and the asterisk refers to the populations given by LTE (Sect. 2.2.2).

2.3 Radiation field and matter interactions

The state of the radiation field is given by the intensity, i.e. the radiative energy flux per frequency and solid angle. In a radiative transfer code the intensity is determined by solving the radiative transfer equation, whereas in a MC based code the radiative transfer is treated explicitly by propagating radiation packets which interacts with the matter through absorption, emission and scattering. The code includes a treatment of electron scattering, and line absorption/emission, which is treated either as (resonance) scattering or fluorescence. The MC radiation field is parametrized as a diluted black body, the temperature and dilution factor given by

$$\begin{aligned} T_R &= \frac{h}{3.832\ k_B} \Sigma\ \nu J_\nu\ \Delta\nu \\ W &= \frac{\pi}{\sigma\ T_R^4} \Sigma\ J_\nu\ \Delta\nu \end{aligned} \tag{7}$$

where

$$J_\nu\ \Delta\nu = \frac{\Sigma\ \epsilon_\nu\ \Delta s}{4\pi\ V \Delta t} \tag{8}$$

is the MC estimator for the mean intensity (e.g. Lucy 2005), ϵ_ν the packet energy in the co-moving frame, δs the pathlength travelled, V the volume of the cell, δt the duration of the MC experiment and the summation is over all packets in the cell with frequency between ν and $\nu + \delta\nu$. The mean intensities at bound-bound transition wavelengths ($J_{u,l}$) are currently calculated from the diluted blackbody parametrization, but could be evaluated directly from the MC radiation field following Lucy (2005), and this choice will be added in the near future.

2.3.1 Electron scattering

Electron scattering is treated as coherent and isotropic in the co-moving frame, in which case the opacity is given by the Thomson cross-section as

$$\kappa = \sigma_T \frac{n_e}{\rho} \tag{9}$$

where ρ is the density and n_e the electron number density.

2.3.2 Line absorption

Line absorption is treated in the Sobolev approximation, appropriate when the expansion broadening dominates the thermal broadening. The Sobolev optical depth for absorption in a given transition from level j to level i is given by

$$\tau_{l,u} = (B_{l,u} n_l - B_{u,l} n_u) \frac{hc}{4\pi} \frac{dr}{dv} \tag{10}$$

where $B_{l,u}$ and $B_{u,l}$ are the Einstein coefficients for absorption and stimulated emission, respectively, and $\frac{dr}{dv} = t$ in a homologously expanding ejecta.

2.3.3 Line emission

Following line absorption, the subsequent line emission is treated either as (resonance) scattering or fluorescence. In the case of scattering, absorption is assumed to be directly followed by emission through the same transition, whereas, in the case of fluorescence absorption is followed by a cascade of upward and downward transitions. Taking only radiative transitions into account, the upward and downward transition rates are

$$\begin{aligned} R_{i,l} &= A_{i,l}\, \beta_{l,i} n_i \\ R_{i,u} &= (B_{i,u} n_i - B_{u,i} n_u)\, \beta_{i,u} J_{u,i} \end{aligned} \tag{11}$$

where $A_{i,l}$, $B_{u,i}$ and $B_{i,u}$ are the Einstein coefficients for spontaneous emission, stimulated emission and absorption, respectibely, and

$$\beta_{j,i} = \frac{1}{\tau_{j,i}} (1 - \exp(-\tau_{j,i})) \tag{12}$$

is the Sobolev escape probability. In the code, this cascade is treated in a statistical sense using the macro-atom method described by Lucy (2002, 2003), and the end result is always emission through a radiative transition, the emissivity implicitly determined by the macro-atom machinery. As is physically justified (Lucy 2002), line emission, either through scattering or fluorescence, is treated as isotropic in the co-moving frame.

3 Monte-Carlo method

The radiation field is determined by the MC experiment, where a number of radiation packets are propagated on the grid and interacts with the matter. The method used is similar to the one outlined in Lucy (2005), where indivisible and indestructible radiation packets are propagated on a 3-D grid. Following Lucy (2005), here, and in the following, a radiation packet is called a r-packet, whereas a packet representing ionization or excitation energy, is called an i-packet. A r-packet is transformed into an i-packet during de-excitation using the macro-atom machinery (Sect. 3.5), and once re-emitted, it is immediately transformed back into a r-packet.

3.1 Event selection and packet propagation

After an interaction with the matter or a geometrical event the r-packet is ready to proceed its journey through the grid. If the previous event was an interaction a new random optical depth is drawn for the r-packet as

$$\tau = -\, ln\, z, \tag{13}$$

and in either case distances to continuum (electron scattering) and line interactions, as well as geometrical events (cell and grid crossings) are calculated. As the co-moving frequency is continously redshifted in a homologously expanding ejecta, all lines with a frequency lower (redder) than the r-packets co-moving frequency are checked sequentially, and if the accumulated optical depth (continuum plus line) at the line distance exceeds the drawn value a line interaction is chosen. If, at any other point, the accumulated optical depth exceeds the drawn value a continuum interaction is chosen and, if the accumulated optical depth never exceeds the drawn value within the cell, a geometrical event is chosen. When the next event has been chosen the r-packet is moved to that location, its frequency and energy in the co-moving frame updated, and the event treated as described in the following sections.

3.2 Emission

The MC experiment begins with the emission of r-packets from a surface corresponding to the photosphere, or rather the thermalization surface. The radiation is assumed to be blackbody emission at a specified temperature and a specified number of r-packets, initially with the same energy (see Mazzali & Lucy 1993), are drawn from this distribution.

3.3 Electron scattering

As mentioned, electron scattering is treated as coherent and isotropic in the co-moving frame, and a new direction is drawn as

$$cos\, \theta = 2z - 1 \tag{14}$$

$$\phi = 2\pi z \tag{15}$$

where θ and ϕ are the polar and equatorial angles, respectively. Finally, the rest-frame frequency and energy are updated and the packet journey proceeds as described in Sect. 3.1.

3.4 Line scattering

As mentioned, line scattering is treated as coherent and isotropic in the co-moving frame, and a new direction is drawn as described in Sect. 3.3, after which the journey proceeds as described in Sect. 3.1. Note, though, that the line has to be removed from the list of lines to check, as self-absorption is already included in the Sobolev approximation.

3.5 Line flourescence

Line fluorescence is treated using the macro-atom method described in Lucy (2002, 2003), where the r-packet is temporarily transformed into an i-packet, and either an radiative de-excitation, transforming the i-packet back to a r-packet, or internal transitions between the excited states are chosen iteratively. As explained by Lucy (2002), the macro-atom transition probabilities for radiative de-excitations and upward and downward transitions are

$$P^R_{i,l} = R_{i,l}(E_i - E_l)/D \tag{16}$$

$$P^U_{i,u} = R_{i,u}E_i/D \tag{17}$$

$$P^D_{i,l} = R_{i,l}E_l/D \tag{18}$$

where $R_{i,j}$ is the radiative transition rate from state i to state j, E_i the energy of state i and $D = \sum R_{i,j}E_i$ the total energy flow out of state i. Once the i-packet has de-excited, the packet is transformed back to a r-packet and the co-moving frequency is updated to that of the drawn transition. As mentioned, the re-emission of the r-packet is treated as isotropic in the co-moving frame, and a new direction is drawn as described in Sect. 3.3, after which the journey proceeds as described in Sect. 3.1. Note, though, as the co-moving frequency of the r-packet has changed, a new search in the list of lines to check has to be made.

3.6 Cell or grid crossing

If the r-packet cross the border of the grid its journey is terminated, and if the r-packet cross the border of the cell it proceeds its journey as described in Sect. 3.1.

3.7 Frame transformations

Transformations between the rest and co-moving frames are calculated keeping terms of order v/c as follows. The energy and the frequency are transformed as

$$E' = E(1 - \mu v/c) \tag{19}$$

$$\nu' = \nu(1 - \mu v/c) \tag{20}$$

where primed quantities are evaluated in the co-moving frame and μ is the cosine of the polar angle about the radius vector (pointing in the direction of the velocity field in a homologously expanding ejecta). After a scattering, which is treated as isotropic in the co-moving frame, the flight direction is transformed to the rest frame as

$$\mu = \frac{\mu' + v/c}{1 + \mu' c/v} \tag{21}$$

3.8 Virtual cells

In addition to the methods described by Lucy (2002, 2003, 2005) the code implements the concept of virtual cells introduced by Jerkstrand et al. (2011). This method allows macroscopic mixing or clumping to be treated in a statistical sense in a otherwise spherically symmetric grid, which may speed up such a calculation significantly as compared to a 3-D Cartesian grid (which is nevertheless not supported yet). When using this method, each geometrical cell may contain a number of virtual cell types with different composition and density. Each virtual cell type is assigned a filling factor and a size (determining the number of virtual cells), and at each virtual cell crossing a new virtual cell is drawn, the type determined by the filling factors, and the impact parameter from an isotropic distribution.

4 Code design

The code is written in C++ and below follows a simplified description of the objects, the data they hold and the functions they provide. The purpose is not to give all the details, but rather to describe the general ideas the code is built upon. The two main objects are the *Grid* and the *MCExperiment*, representing the SN ejecta model and the MC experiment performed on the grid. A typical code flow would look something like:

```
// Create and load grid from file
grid = new Grid(...);
grid.load("grid.dat");

// While temperature not converged
while (convergence criteria) {
  // Create and run MC experiment
  mCExperiment = new MCExperiment(grid,...);
  mCExperiment.run();
  // Update raditation state
  grid.updateRadiationState(mCExperiment);
  // Update matter state
  grid.upddateMatterState();
}

// Export spectrum
mCExperiment.exportSpectrum("spectrum.dat",...);
```

In the example, a *Grid* object is first created and the atomic data and the ejecta model loaded using the grid configuration file. In a number of λ-iterations *MCExperiment* objects are then used to perform MC experiments on the grid. These determines the radiation field and are continuously used to update matter state of the grid. Once the temperature of the matter has converged the spectrum is saved to file.

4.1 Grid

```
class Grid {
  // List of cells
  vector<Cell*> cell;
  // Atomic data
  AtomDataList atomData;
  // Update the radiation state
  void updateRadiatonState(MCExperiment&
  mCExperiment);
  // Update the matter state
  void updateMatterState();
  // Load grid data from file
  void load(string dataFile);
  // Export statistics to file
  void exportStatistics(string dataFile,...);
  ...
}
```

The *Grid* object represents the SN ejecta and is primarily a container for the *Cell* objects, representing some subdivision of the ejecta into cells. The *Grid* object also holds the

AtomData object and provides top-level functions to update the matter state and the radiation field. In addition, the *Grid* object provides the top-level functions to load the atomic data and the ejecta model using the grid configuration file and to calculate and write grid statistics to file.

4.2 Cell

```
class Cell {
  // Matter state
  // List of atoms
  vector<Atom> atom;
  // Density
  double rho;
  // Temperature
  double rho
  // Update the matter state
  void updateMatterState();
  // Calls the following:
  // Update Temperature
  void updateTemperature();
  // Update ionization state
  void updateIonState();
  // Update bound state
  void updateBoundState();
  ...
  // Radiation state
  // Frequency binned mean intensity
  vector<double> Jnu;
  // Radiation temperature
  double TR
  // Dilution factor
  doulbe W;
  // Update the radiation state
  void updateRadiatonState(MCExperiment&
  mCExpermiment);
  ...
  // Packet interface
  // Frequency sorted line list
  vector<Trans*> lineList;
  // Get border distance
  double borderDistance(Vector3D position,
  Vector3D direction);
  // Get electron scatter opacity
  double electronScatterKappa();
  // Get (Sobolev) optical depth
  double lineTau(Trans* trans);
  ...
  // Administration
  // Load cell data
  void load(pstring& pData,...);
  // Export cell statistics
  void exportStatistics(ofstream& file,...);
...
}
```

The *Cell* object holds the state of the matter and the radiation field in the cell. The state of the matter is represented by the density and the gas temperature, hold by the cell, as well as the populations of ionized and excited states, hold by a list of *Atom* objects (Sect. 4.4). The state of the radiation field is represented by the frequency binned mean intensity, as determined from the MC experiment, from which the radiation temperature and dilution factor for the diluted blackbody parametrization is derived (Sect. 2.3). The *Cell* object provides functions to update the state of the matter and the radiation field, but is independent of the implementation of these. Currently, the LTE and nebular methods for the ionized and excited states (Sects. 2.2.2 and 2.2.3), as well as statistical equilibrium for the excited states (Sect. 2.2.1) have been implemented, but other methods (e.g. full statistical equilibrium and thermal equilibrium) may be implemented in the future.

The *Cell* object also provides geometrical functions (e.g. volume and border distance), which are implemented by geometry specific *Cell* objects. Currently, a spherical symmetric geometry is implemented by the *CellSS* object, but other geometries (e.g. a Cartesian grid) may be implemented in the future. Finally, the *Cell object* provides functions to calculate the continuum (electron scattering) and line opacities and holds a wavelength sorted list of transitions. This list only includes transitions with optical depth above a configurable minimum value, and is updated each time the matter state have been updated. Together with the geometrical functions, these functions provide the interface used by the *Packet* objects (Sect. 4.6) during the MC experiment.

4.3 Atomic data

The *AtomData* object holds the atomic data for a particular species, and is the root of a hierarchical structure build upon the *IsotopeData*, *IStateData*, *BStateData* and *TransData* objects, holding the atom (e.g. atomic number), isotope (e.g. number of neutrons), ionization state (e.g. energy), bound state (e.g. energy) and transition (e.g. wavelength) data, respectively. The *AtomDataList* object is a list of *AtomData* objects and provides functions to load the atomic data from file. Files with the atomic data used by Jerkstrand et al. (2015), as well as the atomic data used by TARDIS (Kerzendorf & Sim 2014) are currently available. As a generic file format is used, once converted to the this format, other sets of atomic data could be loaded as well.

4.4 Atom

```
class Atom {
  // List of isotopes
  vector<Isotope> isotope;
  // List of ionization states
  vector<IState> iState;
  // Number of protons
  double Z();
  ...
}

class Isotope {
  // Mass fraction
  double X;
  // Number of neutrons
  // (Hold by IsotopeData)
  double N();
  // Atomic mass
  // (Hold by IsotopeData)
  double m()
  ...
}

class IState {
  // List of bound states
  vector<BState> bState;
  // Number fraction
  double X;
  // Ionization energy
  // Hold by (IStateData)
  double Xi();
  // Partition function
  doube U();
  ...
}

class BState {
  // List of upward transitions
  vector<Trans*> upTrans;
  // List of downward transitions
  vector<Trans*> downTrans;
  // Number fraction
  double X;
  // Energy
  // (Hold by BStateData)
  double E();
  // Statistical weight
  // (Hold by BStateData)
  double g();
  ...
  // Macro-atom machinery
  // De-excite using macro-atom machinery
  Trans* deExcite(TransType& transType):
  // Downward transitional energy outflow
  double ED();
  // Upward transitional energy outflow
  double EU();
  // Radiative energy outflow
  double ER();
  ...
}

class Trans {
  // Transition wavelength
  double nu();
  // Spontaneus emission coefficient
  // (Hold by TransData)
  double A();
  // Downward radiative transition rate
  double RD();
  // Upward radiative transition rate
  double RU();
  // (Sobolev) optical depth
  double tau();
  // (Sobolev) escape fraction
  double beta();
  ...
}
```

The *Atom* object is the root of a hierarchical structure that mirrors a *AtomData* object, from which the atomic data is obtained, but only contains the isotopes, ionization states, bound states and transitions specified in the model configuration file or restricted by selection rules. For convenience, each object in the hierarchy holds a back references to its parent object, and the root object holds a back reference to a *Cell* object. This allows functions for e.g. the number density to be implemented at each level in the hierarchy without holding redundant information. The *Atom* object holds lists of isotopes and ionization states, and provides functions to get the number of protons from the *AtomData* object, as well as to calculate derived quantities (e.g. number density).

Isotope The *Isotope* object holds the mass fraction of the isotope, and provides functions to get the number of neutrons and atomic mass from the *IsotopeData* object, as well as to calculate derived quantities (e.g. number density).

Ionization state The *IState* object holds the number fraction in the state and a list of bound states, and provides functions to get the ionization energy and ion charge from the *IStateData* object, as well as to calculate derived quantities (e.g. number density and partition function).

Bound state The *Bstate* object holds the number fraction in the state and lists of upward and downward transitions, and provides functions to get the bound state energy and statistical weight from the *BStateData object*, as well to calculate derived quantities (e.g. number density). The *BState* object also provides functions to get the macro-atom transition rates and to de-excite from the state using the macro-atom machinery (Sect. 3.5).

Transition The *Trans* object provides functions to get the transition wavelength and the spontaneous emission coefficient from the *TransData* object, as well as functions to calculate radiative transition probabilities and the Sobolev optical depth and escape fraction. Note that the *Trans* object holds back references to both upper and lower *BState* objects, which therefore shares the parenthood.

4.5 Monte-Carlo experiment

```
class MCExperiment {
  // List of packets
  vector<Packet> packet;
  // Run the MC experiment
  double run();
  // Export spectrum
  void exportSpectrum(string dataFile, ...);
  // Export statistic
  void exportStatistics(string dataFile, ...);
  ...
}
```

The *MCExperiment* object is basically a container for a specified number of *Packet* objects to be propagated on the grid, and provides the top-level function to run the MC experiment, as well as functions to export the spectrum and calculate and export statistics to file.

4.6 Radiation packet

```
// Energy packet
class Packet {
  // Frequency (in rest frame)
  double nu;
  // Energy (in rest frame)
  double E;
  // Position
  Vector3D position;
  // Direction
  Vector3D direction;
  // Cell pointer
  Cell* cell;
  // Handle electron scattering
  void electronScatter();
  // Handle line scattering/flourescence
  void lineScatter();
  // Enter new cell
  void newCell();
  // Process i-packet
  Trans* processIPacket();
  // Propagate r-packet
  void propagate();
  // Packet history
  vector<PacketEvent> history;
  ...
}
```

The *Packet* object represents a r-packet, although it is temporarily converted to a i-packet in the case of line fluorescence, and implements the MC machinery (Sect. 3). It provides functions to propagate the packet, and to handle interactions with the matter and geometrical events, and it holds the energy and frequency as well as the position and flight direction. Packet propagation is implemented in 3D, but in a coordinate system independent way using *Vector3D* objects. The *Packet* object also holds a pointer to a *Cell* object, which is updated at each cell crossing, from which the optical depths and geometrical quantities needed for the calculations are obtained. The default behaviour is to record each event in a list hold by the *Packet* object, which is later used to update the radiation field of the grid/cells as well as for statistics. However, to save memory when a large number of packets are propagated, the radiation field of the grid/cells may also be updated on-the-fly, avoiding the need to record each event.

4.7 Numerical Methods

Most calculations in the code are quite straightforward, but, as mentioned in Sects. 2.2.1, 2.2.2 and 2.2.3, the Saha-like ionization equation used for the population of ionized states and the statistical equilibrium equation for the population of excited states give rise to non-linear equation system that needs to be solved numerically.

Ionization The populations of the ionized states are calculated using Saha-like ionization equations and requires solving non-linear equation systems in the populations and the electron number density (Sect. 2.2.2 and 2.2.3). Such a system can be solved using the Newton-Raphson method, resulting in a linearised equation system that can be solved with standard methods for matrix inversion. The use of this method is implemented as a configurable choice using routines provided by Numerical Recipes, but the default method is to iteratively and alternately calculate the electron number density and the level populations using a species-decoupled version of the equations as an initial guess. As the electron number density provides the coupling between the different atomic species there is no need to solve the equation system when using this method. The default convergence criteria is 0.1 percent error in the electron number density, but this choice is configurable.

Excitation Statistical equilibrium for the excited states results in a non-linear equation system in the level populations (Sect. 2.2.1), which might be solved by the Newton-Raphson method as discussed above. However, as for the ionized states, an iterative method is instead used, where the non-linearity of the problem is avoided by computing the radiative rates from the populations of a previous iteration, using a LTE solution as initial guess. The resulting linear equation system is solved using the LU-decomposition method for matrix inversion as provided by Numerical Recipes. The default convergence criteria is a 1 percent error in the populations of states with at least one transition with an optical depth larger than 10^{-3}, but these choices are configurable.

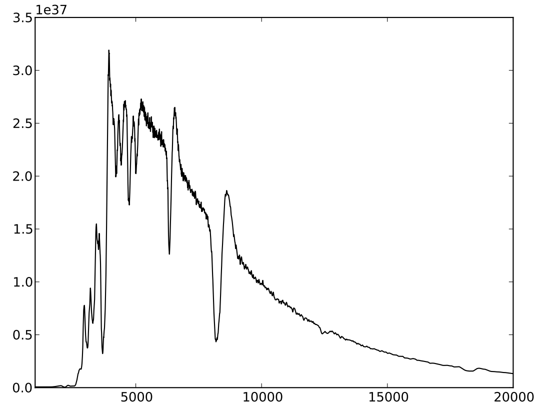

Figure 1: Spectrum calculated using the nebular approximation for ionization, NLTE for excitation and fluorescence for ejecta with LMC metallicity and a density profile with power law of -9. The blackbody temperature was iterated to reproduce the observed luminosity of SN 1987A at 5.78 days using the observed blackbody radius.

5 Tests and comparisons

In the following the code is tested using each of the implemented methods for ionization, excitation and line interactions. Inter-comparisons of the results are made, as well as comparisons to results obtained with the open source code TARDIS (Kerzendorf & Sim 2014), which have very similar capabilities. For the comparisons with TARDIS we use a pure hydrogen composition and the TARDIS atomic data, whereas for the method inter-comparisons we use a LMC composition and the Jerkstrand et al. (2015) atomic data. Note that the TARDIS atomic data only includes the α-γ and α-β lines for the Balmer and Pashen series.

5.1 Detailed treatment

First, spectra were calculated with MCE using the detailed treatment (nebular approximation for the ionization, statistical equilibrium for the excitation and fluorescence for the line interactions) for LMC and pure hydrogen compositions. A density profile with a power law of -9 was used, and the blackbody temperature was iterated to reproduce the observed luminosity of SN 1987A at 5.78 days using the observed blackbody radius. The calculation for pure hydrogen composition was then repeated with TARDIS using the same blackbody temperature and setup. Fig. 1 shows the MCE spectrum for LMC composition whereas Fig. 2 shows the MCE and TARDIS spectra for pure hydrogen composition, and as is clear from the latter, the agreement between MCE and TARDIS is excellent.

In the following sections we repeat the procedure using the same blackbody temperature and setup, changing one method at a time, thus testing the effect of the LTE approximation for the ionization, the LTE and nebular approximation for the excitation and the scattering approximation for the line interactions.

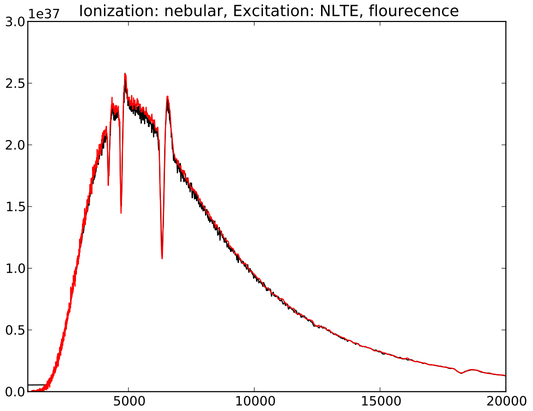

Figure 2: Comparison of spectra using MCE (black) and TARDIS (red) for pure hydrogen composition and otherwise as specified in Fig. 1.

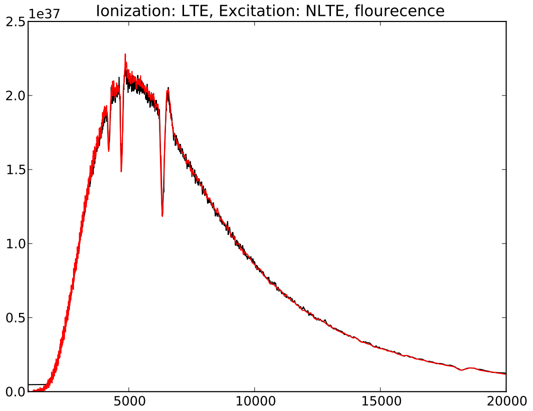

Figure 4: Comparison between MCE (black) and TARDIS (red) spectra for pure hydrogen composition using LTE approximation for the ionization.

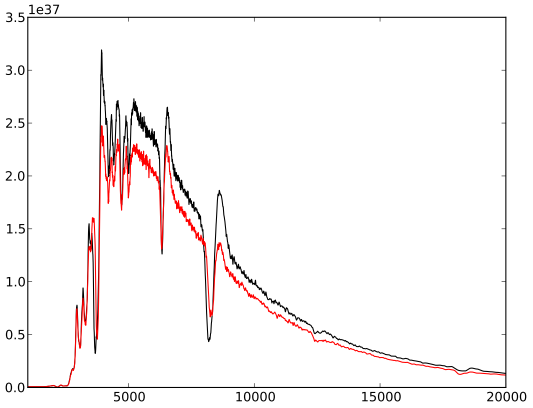

Figure 3: Comparison of spectra for LMC composition using LTE (red) and nebular (black) approximations for the ionization.

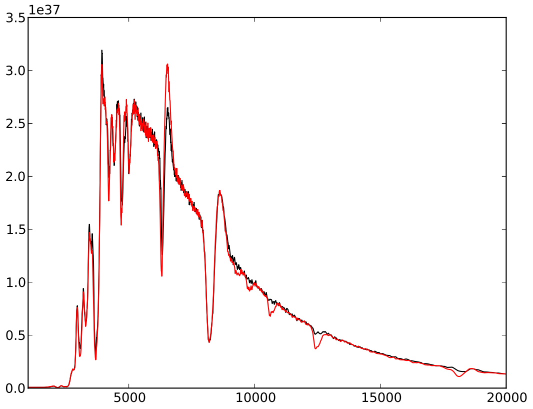

Figure 5: Comparison of spectra for LMC composition using LTE (red) approximation and NLTE (black) for the excitation.

5.2 LTE ionization

Fig. 3 shows a comparison of spectra for LMC composition calculated using the detailed treatment and the LTE approximation for the ionization. The latter spectrum is significantly fainter, caused by a higher degree of ionization, which leads to higher electron density and more backscattered photons. Fig. 4 shows a comparison of MCE and TARDIS spectra for pure hydrogen composition calculated using the LTE approximation for the ionization. The agreement is excellent and, comparing to Fig. 2, as for the LMC composition, the spectra are significantly fainter than when the LTE approximation is used.

5.3 LTE excitation

Fig. 5 shows a comparison of spectra for LMC composition calculated using the detailed treatment and the LTE approximation for the excitation. The latter spectrum shows significantly stronger Balmer and Pashen series emission/absorption, caused by a the higher degree of excitation for the non-metastable states. Fig. 6 shows a comparison of MCE and TARDIS spectra for pure hydrogen composition calculated using the LTE approximation for the excitation. The agreement is excellent and, comparing to Fig. 2, as for the LMC composition, the Balmer and Pashen series shows significantly stronger emission/absorption when the LTE approximation is used.

5.4 Nebular excitation

Fig. 7 shows a comparison of the spectra for LMC composition calculated using the detailed treatment and the nebular approximation for the excitation. The agreement for emission/absorption in the Balmer and Pashen series with the detailed treatment is improved as compared to the LTE approximation, but still overproduce absorption in Pashen β-γ, which suggests that the nebular approximation becomes increasingly worse for higher lying states (see Abbott & Lucy 1985). Fig. 8 shows a comparison of MCE and TARDIS spectra for pure hydrogen composition calculated using the nebular approximation for the excitations. The agreement is excellent and, comparing to Fig. 2, as for the LMC com-

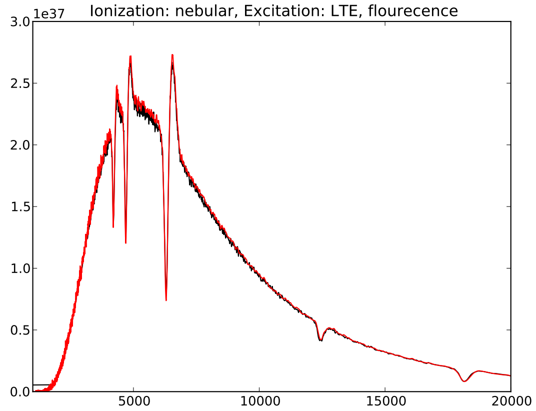

Figure 6: Comparison between MCE (black) and TARDIS (red) spectra for pure hydrogen composition using LTE approximation for the excitation.

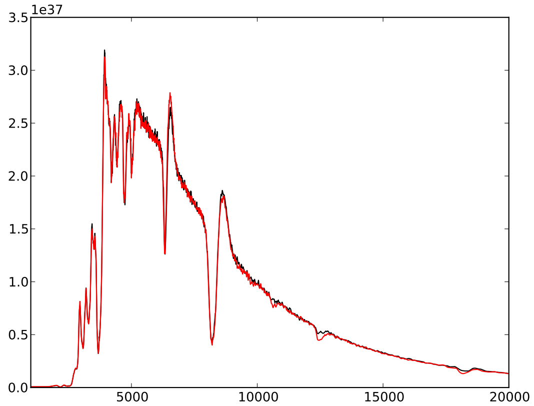

Figure 7: Comparison of spectra for LMC composition using nebular (red) approximation and NLTE (black) for the excitation.

position, the agreement with the most detailed treatment is improved as compared to the LTE approximation, but emission/absorption, in particular for the Paschen series, is stronger than when using statistical equilibrium.

5.5 Scattering

Fig. 9 shows a comparison of the spectra for LMC composition calculated using the detailed treatment and scattering for the line interactions. The two spectra agree surprisingly well and the most noticeable difference is a slightly stronger emission in the Hα line in the case of fluorescence. Fig. 10 shows a comparison of MCE and TARDIS spectra for pure hydrogen composition using scattering for the line interactions. The agreement is excellent and, comparing to Fig. 2, it is again evident that the difference between the fluorescence and scattering calculations is small.

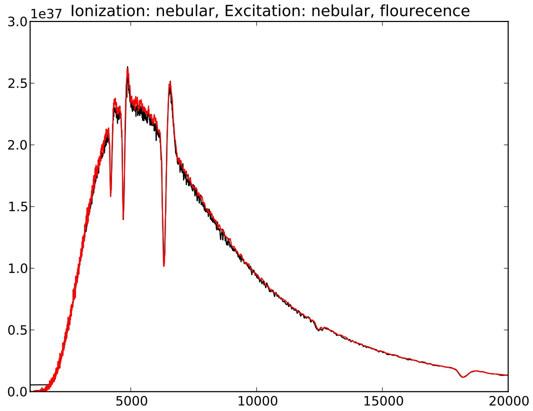

Figure 8: Comparison between MCE (black) and TARDIS (red) spectra for pure hydrogen composition using nebular approximation for the excitation.

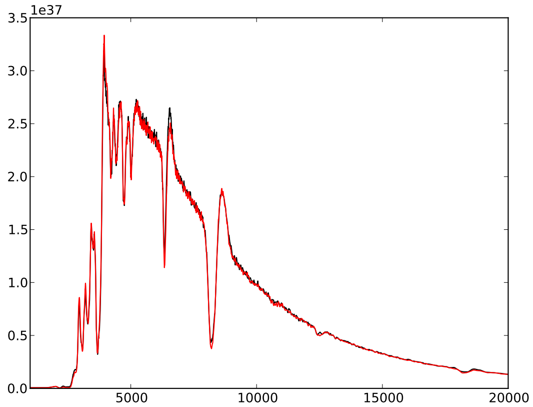

Figure 9: Comparison of spectra for LMC composition using scattering (red) and fluorescence (black) for the line interactions.

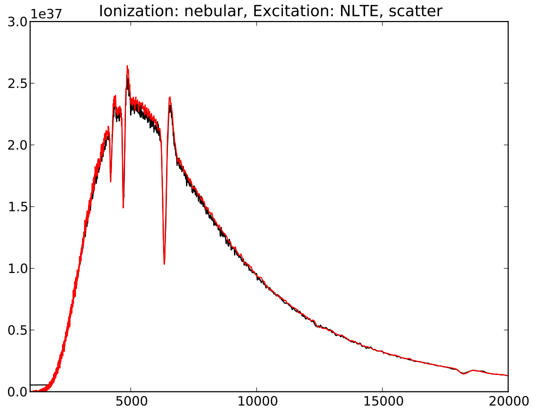

Figure 10: Comparison between MCE (black) and TARDIS (red) spectra for pure hydrogen composition using scattering for the line interactions.

6 Summary

References

Abbott, D. C. & Lucy, L. B. 1985, ApJ, 288, 679

Jerkstrand, A., Ergon, M., Smartt, S. J., et al. 2015, A&A, 573, A12

Jerkstrand, A., Fransson, C., & Kozma, C. 2011, A&A, 530, A45

Kerzendorf, W. E. & Sim, S. A. 2014, TARDIS: Temperature And Radiative Diffusion In Supernovae, Astrophysics Source Code Library

Lucy, L. B. 2002, A&A, 384, 725

Lucy, L. B. 2003, A&A, 403, 261

Lucy, L. B. 2005, A&A, 429, 19

Mazzali, P. A. & Lucy, L. B. 1993, A&A, 279, 447