Planet_EMC
version 1.1.0
User Manual

November 29, 2009

V. Eymet, LAPLACE, Université Paul Sabatier, 31062 Toulouse, France

Contents

1 Introduction

This document is intended to be used by any new user of the code Planet_EMC . The basic linux commands are supposed to be known (file system commands, environment variables use, basic shell scripts).

The user is free to use and modify the sources of the code. It has been writen in fortran 77, which imposes some limitations, mostly for memory management.

Any question, remark, or improvement suggestion is welcome, and should be submitted by e-mail to the author (Vincent Eymet, eymet@laplace.univ-tlse.fr).

1.1 Purpose of the code

Planet_EMC is a computing code for simulation of radiative transfer into a three-dimensional inhomogeneous absorbing and scattering planetary atmosphere. It is based on the Monte-Carlo method. In the present stage of development (version 1.1.0), the code will actually only simulate solar radiative transfer.[1]

1.2 Geometric configuration

Basic configuration / cells: the atmosphere is devided into "cells": a spherical grid is imposed on the planet, i.e. the atmosphere is devided into a given number of columns (latitude/longitude grid). Then a column is devided into a number of cells. A given cell is therefore defined by two latitudes, two longitudes and two altitudes. Cells from adjacent columns can be defined at different altitudes. The number and position of cells can be different for each column, except for the altitude at the top of the last cell that **must be identical for every column**. The code will not work when this requirement is not satisfied.

Optical properties of the gas (absorption and scattering coefficients, for each narrowband spectral interval) as well as gas refractive indexes have to

[1] Actually, Planet_EMC is presently not taking into account any absorption by gas or particles ! In order to take absorption into account, line 360 of file "source/exchanges.for" has to be commented, and line 362 (in the same file) has to be uncommented. Then the executable has to be recompiled, of course.

be provided for each volumic cell. These optical properties are then homogeneous within the cell. Clouds optical properties also have to be provided, but separately from gas data.

Ground: the altitude of the ground has to be defined at the center of each column. If an indentical value is given for every column, then the ground is spherical. Altitude at a given location (latitude/longitude) is interpolated from values of altitude given at the center of adjacent columns. More precisely, this is how altitude at ground level is computed for a given (θ,ϕ) location:

- Coordinate r (radius) at a given (θ,ϕ) position is computed as: $r = a * \theta + b * \phi + c$, with a, b and c coefficients that belong to the column the (θ,ϕ) position is in.
- a, b and c coefficients are computed at initialization time for every column.

The value of the altitude within a given column must be **lower than** the altitude at the top of the first cell of adjacent columns. The computation will not start if this requirement is not satisfied.

Ground emissivity must be provided for each column, and for each narrowband spectral interval (of the gas). In addition, special zones can be defined over the ground. A special zone is defined by two values of the longitude, and two values of latitude. The emissivity that is specified for each special zone overrides the emissivity defined for each column. Special zones can not intersect.

Clouds: clouds are defined the same way gas cells are defined: using two values of latitude, two values of longitude, and two values of altitude. Clouds can intersect gas cells. Optical properties have to be defined for each clouds, for each narrowband spectral interval (gas and clouds spectral grids can be different), and will be homogeneous within the cloud.

Clouds can be defined everywhere (even below the ground / out of the atmosphere) and can intersect.

1.3 Physical hypothesis

Spectral model: the program is currently using a discrete k-distribution spectral model with the possibility to use any quadrature order. The spectral model is also using the CK hypothesis for taking into consideration the effects of inhomogeneities. Every optical property (ground and special zones emissivities, gas and clouds absorption and scattering coefficients, gas refraction indexes) have to be defined for each quadrature element, in each narrowband.

Two spectral grids are used in the code: one for gas optical properties, the other for clouds optical properties. Gas absorption and scattering coefficient, ground emissivity and special zones emissivities have to be provided for the gas spectral grid. Clouds optical parameters (absorption and scattering coefficient, asymetry parameter or real phase function) only have to be provided for the cloud spectral grid. Parameters that are defined over the cloud spectral grid are interpolated on the gas spectral grid during computation.

Refraction: the scattering algorithm implemented in the present code is basic: trajectories between two successive scattering positions are straight lines. However, refraction effects (modifications in refractive indexes that induce a curvature of trajectories) can be taken into account at the interface between gas cells. The refraction index is considered as constant in a given gas cell.

Temperature field: temperature is currently not used by the code, since it simulates solar radiation only (photons emitted by a star, that reach the top of the planetary atmosphere). However, since the same algorithms can be used for thermal radiation simulation, interpolation of the temperature field within the atmosphere has already been implemented. Temperature has to be known at the mass-center of each gas cell. Temperature at a given location (latitude/longitude/altitude) will be interpolated from temperatures at mass-centers of adjacent cells, using a 3D interpolation.

3D interpolations performed by Planet_EMC use the following procedure:

- A set of n data points f_i is known for n positions x_i (for instance, n temperature values T_i are known for n positions x_i, in cartesian coordinates).
- Quantity f has to be interpolated at position x.

- Distances d_i between position x and the n positions x_i are computed.
- f is finally computed as: $f = \sum_{i=1}^{n} f_i \lambda_i$ with $\lambda_i = \frac{\frac{1}{d_i^p}}{\sum_{i=1}^{n} \frac{1}{d_i^p}}$
- in practice, $p = 1$ in Planet_EMC ; you can change its value in routine "gas_temp" that resides in file "source/interpolation.for".

1.4 Numerical method

Planet_EMC is based on the Monte-Carlo numerical method [1] and uses an algorithm that has been optimized for thick media [2, 3]. The main advantage of the Monte-Carlo method is the possibility to compute, in addition of each result, a statistical standard deviation (that can be interpreted in terms of numerical uncertainty) over each result. The method mainly consists in generating the optical path of a number of energy bundles (also denoted as "statistical events" in this text) from their entry point at the top of the atmosphere to their absorption point (when their energy has been completely depleted). These optical paths in a scattering medium are represented by a succession of non-curved segments. However, propagation directions can be modified at the interface between two gas cells, when refraction indexes in these cells have different values.

1.5 Analysis formalism

This code will compute spectrally integrated radiative fluxes densities (upward/downward/net) in W/m^2 at the bottom and top interfaces of each gas cell. This flux density is defined as the flux (in W) that crosses a given interface, devided by the actual surface (in m^2) of this interface. Surfaces at ground level are really computed (the surface at ground level might not be a portion of a sphere).

The code will also compute spectrally integrated radiative budgets in W/m^3 within each gas cell. The radiative budget (or source term) for a given cell is defined as the power (in W) absorbed within this cell, devided by the volume (in m^3) of this cell.

Finally, Planet_EMC will compute (if required, see the description of the "options.in" file in section 2) the angular distribution of upward fluxes at the top of the atmosphere (TOA), i.e. the energy that is backscattered by

the atmosphere and the planet, i.e. the energy that can be observed with a satellite. The frequential signal is kept for these results, which means these distributions can be latter used in order to retrieve the surfacic density flux signal observed by a satellite.

1.6 Informatics

Compiling the program Planet_EMC does not require external librairies: all source files are included in the package.

Planet_EMC is a project whose primary goal is **not** to produce fast results. Instead, it is a research tool, that is intended at learning and visiting all aspects of the coding a three-dimensional raytracing code, within the frame of an application of solar radiative transfer. The code spends most of its time computing intersections of optical paths with cells boundaries. A great amount of development efforts have been put into auto-debugging procedures so that no optical paths gets "lost" into the geometry, resulting in an error-free algorithm that consumes tremendous amounts of computing power.

Planet_EMC is a fully parallel program. As communication times between processes are negligible (compared to computation times), the total computation time will effectively be divided by the number of processors/cores Planet_EMC is running on (provided that all processors / machines in the cluster run at the same speed and are free of other time-consuming processes.) Parallelization requires using MPICH2.

2 Input data files

This section describes input files that are needed by the code. Example data files are provided, along with the program that generates them (when needed).

2.1 data/planet.in

This file contains basic information about the planet and its star:

- The equatorial radius of the planet (in km)
- The surface temperature of the star (in K)
- The star radius (in km)
- The mean distance from the star to the planet (in km)
- The latitude and the longitude of the point on the planet that is at the zenith (in degrees).

2.2 data/grid.in

This is the most difficult file to create. It must be generated by a program. An example program is provided into data/Example/make_grid.for; this program will generate a "grid.in" file into its current folder. The file must contain:

- N_θ the number of intervals latitude is discretized into (π rad), N_ϕ the number of intervals longitude is discretized into (2π), $N_{b,gas}$ the number of narrowband spectral intervals for the gas, N_{zones} the number of special zones.
- For each column (whose number is defined by N_θ and N_ϕ), N_z the number of cells on this column has to be provided, followed by the N_z+1 values of altitude (in meters) that define these cells. The first value is the altitude at ground level at the center of the column (can be different from the value of the planetary radius, but must be **lower** than the altitude at the top of the first layer for adjacent columns), and the N_z following values are the altitudes at the top of the N_z cells

of the column. The last value (altitude of space) must be identical for all columns.

- The N_θ+1 values of latitude (in rad) that define the N_θ latitude intervals. Latitude is defined between $-\pi/2$ for the south pole and $+\pi/2$ for the north pole.

- The N_ϕ+1 values of longitude (in rad) that define the N_ϕ longitude intervals. Longitude is defined between 0 and 2π.

- For each column, the value of altitude at mass center of each cell must be provided (N_z values, in meters).

- For each column, the temperature profile must be provided.[2] There are N_z+4 values for each column: the value of temperature for the ground, the value of temperature in the gas at ground level, N_z values of temperature, at the mass center of each one of the N_z gas cells, the value of temperature in the gas at space level, and the value of temperature for space. Values of temperature are considered at the longitude and latitude of column's center.

- The $N_{b,gas}$ lower limits (in μm) of the $N_{b,gas}$ gas narrowband spectral intervals.

- The $N_{b,gas}$ higher limits (in μm) of the $N_{b,gas}$ gas narrowband spectral intervals.

- The N_q quadrature weights.

- For each column, the $N_{b,gas}$ values of ground and space emissivities must be provided.

- For each column and for each gas cell in the column, the $N_{b,gas}$ values of gas absorption and scattering coefficients (in m^{-1}) must be provided.

- For each one of the N_{zones} special zones, two values of latitude (in rad) and of longitude (in rad) that define the spatial extension of the zone must be specified.

[2] In version 1.1.0, Planet_EMC does not use the temperature profile; any value can therefore be provided.

- For each one of the N_{zones} special zones, the value of the zone's emissivity must be provided for each one of the $N_{b,gas}$ gas narrowband spectral intervals.
- For each column and for each gas cell in the column, the value of the cell's refraction index must be provided.

2.3 data/clouds.in

This file gathers information about the clouds. It is produced by a program. An example program is provided for into data/Example/make_cloud_data.for; this program will generate a "clouds.in" file into its current folder. The file must contain:

- N_{clouds} the number of clouds in the atmosphere, $N_{b,clouds}$ the number of narrowband spectral intervals clouds optical parameters are defined over, N_a the number of angles the cloud phase function is defined for.
- The $N_{b,clouds}$ lower limits (in μm) of the $N_{b,clouds}$ clouds narrowband spectral intervals.
- The $N_{b,clouds}$ higher limits (in μm) of the $N_{b,clouds}$ clouds narrowband spectral intervals.
- For each cloud, two values of latitude (in rad), two values of longitude (in rad) and two values of altitude (in meters) that define the geometrical extension of the cloud have to be provided.
- For each cloud, the value of the absorption coefficient (in m^{-1}) has to be provided for each one of the $N_{b,clouds}$ clouds narrowband spectral intervals.
- For each cloud, the value of the scattering coefficient (in m^{-1}) has to be provided for each one of the $N_{b,clouds}$ clouds narrowband spectral intervals.
- For each cloud, the value of the asymetry parameter has to be provided for each one of the $N_{b,clouds}$ clouds narrowband spectral intervals.

- For each cloud, and for each one of the N_a angles, the value of the cosine of the angle and of the cloud real phase function has to be provided for each one of the $N_{b,clouds}$ clouds narrowband spectral intervals.

Note: It is possible to choose between using the clouds asymetry parameter or phase function at run time (see the description of file "options.in") but both have to be provided into the "data/clouds.in" file. Of course, if asymetry parameter only has to be used, N_a can be set to zero.

2.4 data.in

This file resides into the Planet_EMC main folder. It is used for input data the algorithms may need during execution. It must contain:

- The percentage of energy that is neglected when the spectral range is computed: during initialization, Planet_EMC computes the spectral range radiative transfer has to be simulated for. Only solar radiative transfer is involved in version 1.1.0, so the total spectral range is based on the Planck curve for T_{star}, the star emission temperature (this data is found by the code into file "data/planet.in"). Theoretically, the whole spectrum should be considered (from $\lambda = 0$ to $\lambda = +\infty$) but it is not possible in practice. The effective spectral range that will be considered is the spectral interval that contains $x * \sigma * T_{star}^4$, with x the fraction of energy taken into account. The percentage of energy required in this file is $(1 - x)$, and its value should be between 0 and 1. Default value is 2.10^{-3} (0.2%).

- Number of statistical events: increase this value to get a better accuracy over the results. Keep in mind that the uncertainty over a result (its statistical standard deviation) will evolve as $\frac{1}{\sqrt{N}}$ with N the number of statistical events. If uncertainty as to be devided by a factor 10, N has to be increased by a factor 100.

- Number of computational chunks: the code will be run over n_p cores. Each core will be sent a computational chunk, until no chunk is left. A value of -1 indicates the number of computational chunks has to be equal to the number of child processes (the number of cores).

2.5 options.in

This file resides into the Planet_EMC main folder. It is used for input options the algorithms may need during execution. This file must contain:

2.5.1 Physics

- Whether reflexion at ground level has to be specular of diffuse.
- Whether Planck intensity has to be integrated for each narrowband spectral interval or not. If not, the value of the Planck intensity at the center of each narrowband interval is used.
- Whether the asymetry parameter or the user-defined phase function has to be used for clouds. In the case asymetry parameters have to be used, the phase function is a Henyey-Greenstein function.
- Whether or not refraction effects have to be taken into account at gas cells interfaces.

2.5.2 Algorithms

- Whether TOA fluxes angular distribution have to be computed or not.

3 Compiling and using the code

As mentionned in section 1, Planet_EMC requires MPICH version 2 to be installed, because the code has been parallelized.

3.1 Installation of MPICH

If you do not already have MPICH installed on the machine / group of machines you want to run Planet_EMC on, you will first have to download mpich2 from http://www.mcs.anl.gov/research/projects/mpich2 ; make sure you download version 1.0.7. or newer. Next, untar the downloaded archive, and install it on every system that will be part of your cluster:

First, you may have to set environment variables CFLAGS, FFLAGS, F90FLAGS, CXXFLAGS, F77 and F90 according to your architecture and the compilers that are installed on your system.

> export CFLAGS="-m32" (use "-m64" on 64 bits systems)
> export FFLAGS="-m32" (idem)
> export CXXFLAGS="-m32" (idem)
> export F77="ifort" (use any other compiler)
> export F90="ifort" (idem)
> ./configure –prefix=/path/to/installation/directory
> make
> make install

Before running the MPD daemon, you must create a ".mpd.conf" file in your home folder:

> echo secretword=[secretword] » /.mpd.conf
> chmod 600 .mpd.conf

using any "secretword".

Next, you will need to be able to connect via ssh to every other machine of your cluster, with no password request. For this, you must first create a DSA key on the machine you will run the code from :

> ssh-keygen -t dsa

leaving all fields blank (use the "enter" key to answer each question). Then you will have to add this DSA key to the list of authorized keys of every machine that will need to be accessed for computation :

> cd .ssh
> cat id_dsa.pub » authorized_keys

Finally, create the list of machines that belong to your cluster. This list must reside within the "mpd.host" file on your home folder. Each line must contain the name of the machine, by order of availability:

[host1].[domain]
[host2].[domain]
[host3].[domain]
etc

You can then try to run the MPD daemon:

> mpdboot -n [#]

with [#] the number of hosts you want to run MPD on (typically, the number of machines in your cluster). If you encounter no error, you can use command "mpdtrace" to check the number of hosts the MPD daemon is running on. This should give you the list of machines in your cluster.

3.2 Tweaking the "Makefile"

Before compiling, you will have to find out what compilation options are right for your compiler, and your machine. Open the "Makefile" file, and look at variables "FOR", "ARCH" and "OPTI". Variable "FOR" is used to specify your fortran 77 compiler. As Planet_EMC uses MPICH, you will most likely use the "mpif77" compilation command, that has been installed along with MPICH.

Variable "ARCH" is used to specify machine architecture. "-m486" is probably a good choice for a PC running a 32bits linux. On recent Mac computers, "i686 -m64" works. Use the documentation of your fortran compiler to find out what architecture option you can use.

Variable "OPTI" is used to specify code optimization options. The default options should be enough. Please note that you definitely must use option "-Wno-globals" for compiling parallel code.

You might also want to set variable "DEBUG" (look for its definition in the file). You can expect faster execution times if you leave it empty.

3.3 Setting up array sizes

One limitation of fortran 77 code is that you must define array sizes before compilation. Arrays sizes used by the present code are defined within the "includes/max.inc" file. You should at least look at it before compiling, and more precisely at the value of variables Ntheta_mx, Nphi_mx, Nz_mx,

Nb_clouds_mx, Nb_gas_mx, and Nq_mx. Please note you should never modify the value of variable Nmat_mx (3).

When the angular distribution of TOA fluxes is computed, remember that the code will have to evaluate $N_{b,gas}$*(N_θ*N_ϕ)^2 variables with N_θ*N_ϕ and $N_{b,gas}$ the actual number of columns and spectral narrowbands that have been specified within the "data/grid.in" input data file. This number can easily grow to huge values. In this case, N_θ, N_ϕ and $N_{b,gas}$ should have small values. In theory, the corresponding definition variables Ntheta_mx, Nphi_mx and Nb_gas_mx can be as large as required, with two small details to keep in mind:

- The executable will not be compiled for large values (arrays definition is possible only in the limit of the RAM amount the system manages).
- Communication times between processes will increase with array sizes, which results in slower computations.

As a consequence, when angular distribution of TOA fluxes is computed, definition variables Ntheta_mx, Nphi_mx and Nb_gas_mx should be set to their minimum values, i.e. the values of variables N_θ, N_ϕ and $N_{b,gas}$ that will be used during the computation.

3.4 Compilation

Once you checked compilation options and array size definitions, you can use the following command in order to compile the executable file:

> make all

If compilation fails, use the compiler error message to determine what went wrong. The most probable error causes are: a bad definition of architecture compilation option, or an inappropriate value in code optimization options.

If you ever need to modify the source files (in directory "source"), you can quickly recompile the code using "make all" again. This will only recompile the modified source files, and link objects files in order to produce the new executable file.

If you have to modify the value of any variable defined in includes files (directory "includes"), you will have to recompile the whole code from scratch. Use the following command to erase all objects files:

> make clean all

and then recompile them properly with "make all".

Odd errors may happen if you modify an include file and then recompile using only the "make all" command (old value of the modified variable will remain in the unchanged object files).

3.5 Running the code using MPICH

Once everything is installed and the executable file "planet.exe" file has been compiled, you can try to run a computation. I would recommend that, for the first time, you run Planet_EMC using the provided example data files.

Use the following command to run the code:

> mpirun -np [#] planet.exe

with [#] the number of processes that have to run.

Because communication times are small compared to computation times in Planet_EMC , it is a good idea to chose a number of processes equal to the number of (physical) processors of your cluster, plus one. One process, the master process, is dispatching computational loads to every other processes (slave processes), and gathering results from them. It does not require any significant CPU time, therefore it is OK to have a number of slave processes equal to the number of processors, so that each slave process can use a processor (or each processor will have only one slave process running on it).

In practice, if your cluster is composed of n processors, you can use:

> mpirun -np n+1 planet.exe

4 Results

4.1 radiative transfer simulation results

Results from Planet_EMC for the current version are:

- Integrated solar upward fluxes (W/m^2).
- Integrated solar downward fluxes (W/m^2).
- Integrated solar net fluxes (W/m^2).
- Integrated solar radiative budgets (W/m^3).
- The angular distribution of solar upward fluxes at the top of the atmosphere (W/m^2), for each spectral interval. For every column, the angular distribution of outgoing TOA fluxes is kept: a 3D angular discretization is attached to every column; during the raytracing algorithm, each particle that reaches space level contributes to the TOA flux of the column it is exiting, in the angular sector its exit direction belongs to, in the spectral narrowband its wavenumber belongs to. See figure 4.1.

The file that contains the angular distribution of upward TOA fluxes ("results/integrated_solar_TOA_fluxes.txt") is by far the most heavy file. Results that are recorded into this file can be used in order to compute the solar flux density received by a satellite detector, in each spectral interval (this is the purpose of the "satellite" code, that uses this file as an input).

Results files can be found in the "results" folder. You may want to look at routine "solar_postcalc" into file "source/postcalc.for" to see exactly how result files are generated.

4.2 log files

In addition to radiative transfer simulation results, Planet_EMC will keep a record of what happened during the execution into log files. These files can be found into the "logs" folder. Also, a directory is created for each run into the "logs/archive" folder, using the name of the machine, the number of cores the code ran on, the date of record and the number of statistical realizations that where required. Into this directory, you may find several files named

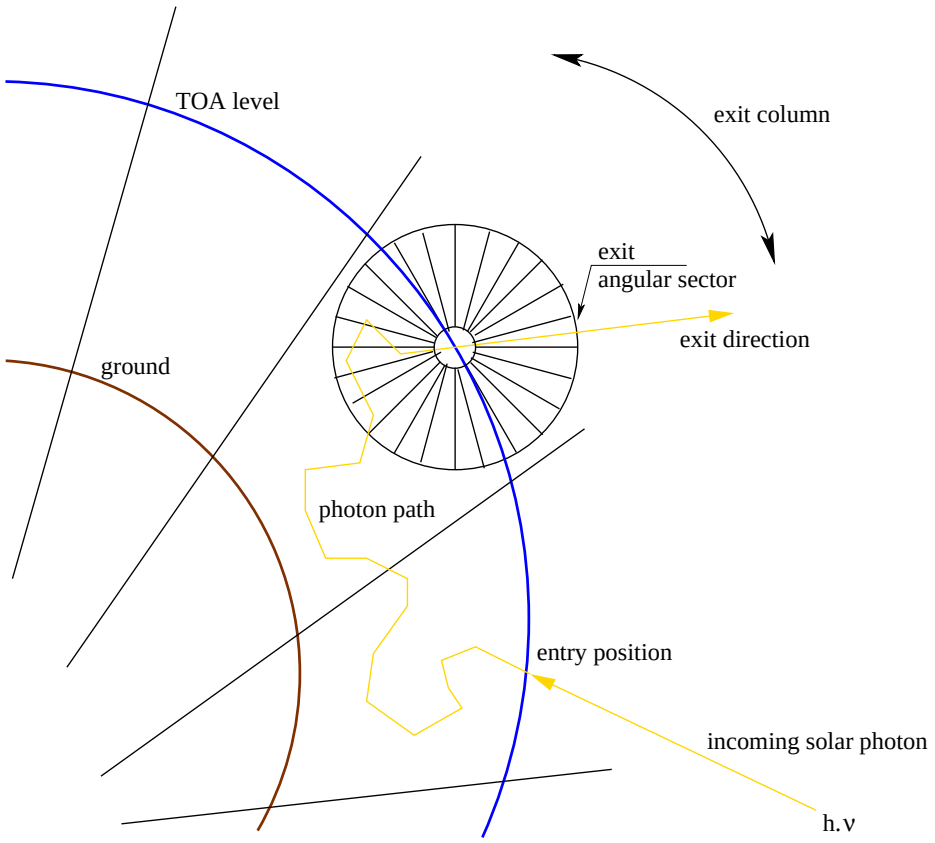

Figure 1: In this example, a particle exits the atmosphere (in a different column than the entry column). The energy it carries is added to the TOA flux distribution of the exit column, in the exit angular sector, in the spectral narrowband frequency ν belongs to.

"error_process***_event***.log". These files are generated when the code could not find an intersection for a given propagation direction. You may want to check the automatic debugging procedure solved the problem by looking into these files.[3]

[3]There may be a lot of information into these files. However, self-explanatory messages should be found at the very end of each file, indicating whether or not the problem could be solved.

References

[1] J.M. Hammersley and D.C. Handscomb. *Monte-Carlo methods*. John Wiley, New York, 1964.

[2] V. Eymet, J.L. Dufresne, R. Fournier, and S. Blanco. A boundary-based net exchange Monte-Carlo Method for absorbing and scattering thick medium. *Journal of Quantitative Spectroscopy and Radiative Transfer*, 95:27–46, 2005.

[3] A De Lataillade, J. L. Dufresne, M. El Hafi, V. Eymet, and R. Fournier. A net exchange Monte-Carlo approach to radiation in optically thick systems. *Journal of Quantitative Spectroscopy and Radiative Transfer*, 74:563–584, 2002.