

- [5. Radiative Transfer \(RT\) Simulation](#)
 - [Examples](#)
 - [Usage of "mcarats" executables](#)
 - [Possible problems](#)
 - [Tips using tools](#)
 - [Quick visualization using GrADS](#)
- [6. Output data from the "mcarats" codes](#)
 - [Format](#)
 - [Header section](#)
 - [Data section](#)
 - [Reading the output](#)

[Edit](#)

5. Radiative Transfer (RT) Simulation

Examples

The user can easily learn about how to use the RT simulation tools from a shell script, `./examples/Job.csh`, and the example I/O files in `./examples`.

Usage of "mcarats" executables

The following executables will be installed in `./bin` directory after the installation:

```
mcarats3d: for a 3-D model atmosphere
mcarats2d: for a 2-D model atmosphere
mcarats1d: for a 1-D model atmosphere
```

The usage of the RT simulation tools is, for example, as follows:

```
% ./bin/mcarats3d ptot solver recomp configfile newoutfile (oldoutfile)
ptot   : Total number of incident photons. The format is integer or real (e.g., 1000000 or 1.0e+6)
solver : Radiative transfer solver
  =0: F3D, Fully-3-Dimensional radiative transfer
  =1: P3D, Partially-3-Dimensional radiative transfer
  =2: ICA, Independent Column (or pixel) Approximation
recomp : Recomputation switch
  =0: New experiment without previous results: Q = Qnew
  =1: Recomputation/average: Q = f*Qnew + (1-f)*Qold, where f=ptot_new/(ptot_new + ptot_old)
  =2: Integration/adding: Q = Qnew + Qold
configfile: Configuration file containing the namelist input
newoutfile: New output file
oldoutfile: Old file containing precomputed results (optional)
```

For example, when the user operates the following,

```
% ./bin/mcarats3d 1e5 0 0 ./examples/case3/conf_rs0067 out
```

then the output data file, `./out`, will be generated. Next, this simulation results can be improved by adding photons:

```
% ./bin/mcarats3d 1e5 0 1 ./examples/case3/conf_rs0067 out2 out
```

The new output file, `./out2`, contains results for total 200,000 incident photons. This recomputation process can be done recursively, and this function may be useful in some cases.

Possible problems

1) Invalid input format

The most frequent error is caused by invalid input format of the configuration file that contains namelist parameters. Format for namelist input may slightly differ by Fortran compiler. Please read carefully the compiler manual for valid namelist input format. In addition, if the code reported reading error of namelist input or external input files, then check the configuration file or the external input files is written in valid format as described here.

2) Invalid input values

The other common error is from invalid input values. In this case, diagnostics might be reported from the codes (in some cases). Change the values for the input variables in valid ranges. Especially, input values for a variable should be integer if the variable is declared as integer. In addition, some variables should be in specific ranges. Check also whether the arguments of each array variable are not out of bounds. See the descriptions of the input variables.

3) Output results are noisy

This may be due to small number of trajectories simulated. Increase the number `ptot`, in this case.

5) Bug?

In other cases, there is no advise from the developers. Please feel free to ask any question to developers, who hope to reply to all

questions as long as there is enough time. If you find any bug, please report to the developers.

Tips using tools

To simulate for several similar cases with small differences, there is a useful tool,

```
./shl/replconfig.awk
```

that is an awk script. This tool can replace one parameter value in the input configuration file. A limitation is that the variable to be replaced should be scalar (i.e., not array variable). For example, we can replace the value for variable fmax in

```
./examples/case3/conf_rs0067
```

by

```
% awk -f ./shl/replconfig.awk fmax 0.6 < ./examples/case3/conf_rs0067 > ./conf
```

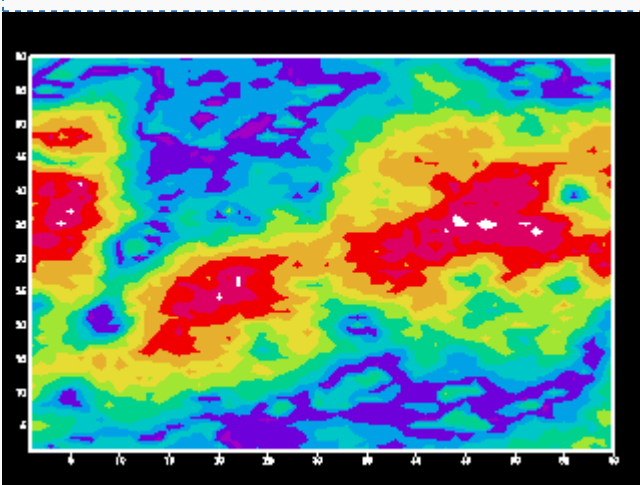
Then the new ./conf file contains a line as "fmax = 0.6". This is useful to perform many simulations, as follows:

```
% awk -f ./shl/replconfig.awk fmax 0.0 < ./examples/case3/conf_rs0067 > ./conf
% ./bin/mcarats3d 1e5 0 0 ./conf ./out-fmax0.0
% awk -f ./shl/replconfig.awk fmax 0.2 < ./examples/case3/conf_rs0067 > ./conf
% ./bin/mcarats3d 1e5 0 0 ./conf ./out-fmax0.2
% awk -f ./shl/replconfig.awk fmax 0.4 < ./examples/case3/conf_rs0067 > ./conf
% ./bin/mcarats3d 1e5 0 0 ./conf ./out-fmax0.4
% awk -f ./shl/replconfig.awk fmax 0.6 < ./examples/case3/conf_rs0067 > ./conf
% ./bin/mcarats3d 1e5 0 0 ./conf ./out-fmax0.6
% awk -f ./shl/replconfig.awk fmax 0.8 < ./examples/case3/conf_rs0067 > ./conf
% ./bin/mcarats3d 1e5 0 0 ./conf ./out-fmax0.8
% rm -f ./conf
```

Quick visualization using GrADS

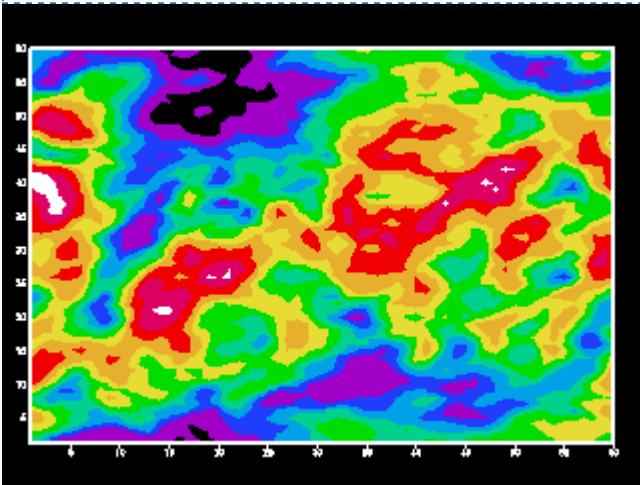
To quickly visualize output data from the RT codes ("mcarats*"), a tool, pgrads, is provided. This converts the output data file to GrADSformat files. If GrADS is installed in your system, you can try the followings:

```
% ./bin/mcarats3d 1e5 0 0 ./examples/case3/conf_rs0067 out
% ./bin/pgrads 1 3 0 ./out
% grads -l
...
ga-> open out.gdctl
Scanning description file: out.gdctl
Data file out.gd is open as file 1
LON set to 1 64
LAT set to 1 64
LEV set to 1 1
Time values set: 2000:1:0:0 2000:1:0:0
ga-> set gxout shaded
ga-> set mproj off
ga-> set clevs 0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1
Number of clevs = 11
ga-> d rad
Contouring at clevs = 0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1
```



```
ga-> set t 2
Time values set: 2001:1:0:0 2001:1:0:0
ga-> set clevs 0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1
```

```
Number of clevs = 11
ga-> d rad
Contouring at clevs = 0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1
```



```
ga-> quit
No hardcopy metafile open
GX package terminated
```

When "d rad" command is entered, the window displays images. Details of pgrads command are written [here](#).

[Edit](#)

6. Output data from the "mcarats" codes

Format

The general format of "mcarats" output file is as follows:

```
header
data
header
data
header
data
...
```

For example, see `./examples/case3/out_rs0067-f3d`. A set of one header section and one data section corresponds to one definition of radiation source in the input configuration file (see `./examples/case3/conf_rs0067`). Thus the number of sets of the header and data sections is equal to "nsrc" in the configuration file.

Header section

At the beginning of the output file, header information are recorded. The following is an example:

```
%mcarats-0.9
 1 : isrc
 0 : isolver
1.00000E+09 : ptot
 64  64  0  0 : nx, ny, nzf, nzh
 64  64  1 : nxr, nyr, nrdc
600 450  0 : nxc, nyc, ncam
513 257  0 : nxp, nyp, npot
```

Notations are as follows:

- solver: radiative transfer solver; 0 for Fully 3-D (F3D) scheme, 1 for Partially-3-D (P3D) scheme, and 2 for ICA
- ptot: total number of source photons.
- nx, ny: numbers of x and y-grids, respectively, for fluxes and heating rates
- nzf: number of z-grids for fluxes
- nzh: number of z-grids for heating rates
- nxr, nyr: numbers of x and y-grids, respectively, for radiances
- nrdc: number of definitions of radiances
- nxc, nyc: numbers of x and y-grids, respectively, for camera image radiances
- ncam: number of cameras

nppot, nqpot: numbers of pathlength bins and FOV angle bins, respectively, for the local fluxes

- nptot: number of specifications of local flux

Fortran 77 source codes to read/write the header are provided in the package:

```
./src/libmcarats/getmchdr.f and ./src/libmcarats/putmchdr.f.
```

Data section

The data section contains five subsections for fluxes, heating rates, radiances, camera radiances, and local fluxes. Each subsections are written by codes such like the followings:

- Flux subsection:

```
if (nzf .gt. 0) then
  do iflx = 1, 3
    write (iuo, '(a, i6)', err=101) '# iflx= ', iflx
    do izf = 1, nzf
      write (iuo, *) ((pflx(ix, iy, izf, iflx), ix = 1, nx), iy = 1, ny)
    end do
  end do
end if
```

- Heating rate subsection:

```
if (nzh .gt. 0) then
  write (iuo, '(a)', err=101) '# heating rate'
  do izh = 1, nzh
    write (iuo, *) ((phrt(ix, iy, izh), ix = 1, nx), iy = 1, ny)
  end do
end if
```

- Radiance & air mass factor) subsection:

```
do imom = 1, 2
  do irdc = 1, nrdc
    write (iuo, '(a, 2i6)', err=101) '# irdc, imom: ', irdc, imom
    write (iuo, *) ((prdc(ixr, iyr, imom, irdc), ixr = 1, nxr), iyr = 1, nyr)
  end do
end do
```

Note that prdc is radiance (imom=1) or AMF (imom=2).

- Camera radiance subsection:

```
do icam = 1, ncam
  write (iuo, '(a, i6)', err=101) '# icam= ', icam
  write (iuo, *) ((pcam(ixc, iyc, icam), ixc = 1, nxc), iyc = 1, nyc)
end do
```

- Local flux subsection:

```
do ipot = 1, npot
  write (iuo, '(a, 2i6)', err=101) '# iopot, ipot= ', 1, ipot
  write (iuo, *) ((ppot(ippot, iqpot, 1, ipot), ippot = 1, nppot), iqpot = 1, nqpot)
  !// contributions from all events
  write (iuo, '(a, 2i6)', err=101) '# iopot, ipot= ', 2, ipot
  write (iuo, *) ((ppot(ippot, iqpot, 2, ipot), ippot = 1, nppot), iqpot = 1, nqpot)
  !// contributions from source emission plus single scattering
end do
```

See, for example,

```
./examples/case3/out*.
```

Reading the output

The output files can be read in the same way in ./src/process/pconvert.f, for example.

