

- [Namelist mcarWld_nml_job](#)
 - [Wld_nplcf -technical parameter-](#)
 - [Wld_rmlcf -technical parameter-](#)
 - [Wld_dtaumin -technical parameter-](#)
- [Namelist mcarAtm_nml_job](#)
 - [Atm_idread](#)
 - [Atm_dx, Atm_dy](#)
 - [Atm_zgrd0](#)
 - [Atm_wkd0](#)
 - [Atm_tmp1d](#)
 - [Atm_ext1d](#)
 - [Atm_omg1d](#)
 - [Atm_apf1d](#)
 - [Atm_abs1d](#)
 - [Scaling optical properties](#)
 - [Atm_fext1d](#)
 - [Atm_fext3d](#)
 - [Atm_fabs1d](#)
 - [Atm_fabs3d](#)
 - [Atm_tauamin -technical parameter-](#)
 - [Atm_fsupg -technical parameter-](#)
 - [Atm_fsupi -technical parameter-](#)
 - [Atm_nzzsup -technical parameter-](#)
- [Namelist mcarSfc_nml_job](#)
 - [Sfc_idread](#)
 - [Sfc_tmp](#)
 - [Sfc_mtype](#)
 - [Sfc_param](#)
 - [Sfc_nudsm, Sfc_nurpv, Sfc_nulsrt -technical parameter-](#)
 - [Sfc_nqpot -technical parameter-](#)
 - [Sfc_rrmax -technical parameter-](#)
 - [Sfc_rrexp -technical parameter-](#)
- [Namelist mcarSrc_nml_job](#)
 - [Src_mtype](#)
 - [Src_wlen](#)
 - [Src_dwlen](#)
 - [Src_fx](#)
 - [Src_qmax](#)
 - [Src_the, Src_phi](#)
 - [Src_mphi](#)
 - [Src_xpos, Src_ypos](#)
 - [Src_zloc](#)
 - [Src_apsize](#)
- [Namelist mcarRad_nml_job](#)
 - [Rad_mrproj](#)
 - [Rad_zloc](#)
 - [Rad_xpos, Rad_ypos](#)
 - [Rad_rmin0, Rad_rmax0](#)
 - [Rad_frmod](#)
 - [Rad_the, Rad_phi, Rad_psi](#)
 - [Camera coordinate system](#)

[Rad_umax, Rad_vmax](#)

-
- [Rad_qmax](#)
- [Rad_apsize](#)
- [Rad_wfunc0](#)
- [Rad_zetamin -technical parameter-](#)
- [Rad_difr0, Rad_difr1 -technical parameter-](#)
- [What variables are actually used?](#)

[Edit](#)

Namelist [mcarWld_nml_job](#)

Wld_nplcf -technical parameter-

```
integer, save :: Wld_nplcf = 0
```

Power exponent for local collision forcing (LCF), which forces collisions near the radiance detectors, by increasing extinction coefficients near the radiance detectors. The LCF is automatically deactivated if radiances are not target. The LCF is effective when the 1st kind of radiances (Rad_mrkind = 1) are calculated. The formula of the 1st kind of radiance contribution has a term proportional to $1/r^2$, where r is a distance between the collision point and radiance detector point. This type of local estimate has in general infinite variance. To reduce variance, LCF is useful. The power exponent should be 1 or 2 usually. Scaled extinction coefficients (B_ext1) will be

$$B_{\text{ext1}} = \max(B_{\text{min}}, B_{\text{ext}})$$

for original extinction coefficient B_{ext} , and the min extinction coefficient is

$$B_{\text{min}} = Wld_dtaumin / dz_lay * (zdif / Wld_rmlcf)**Wld_nplcf$$

where dz_lay is a layer geometrical thickness and $zdif$ is height (z-coordinate) difference from the camera location. See also [Wld_rmlcf](#) and [Wld_dtaumin](#).

Wld_rmlcf -technical parameter-

```
real(R_), save :: Wld_rmlcf = 100.0_R_
```

Scale length (m) for Local Collision-Forcing (LCF), which forces artificial collisions at locations near the radiance detectors. See also [Wld_nplcf](#).

Wld_dtaumin -technical parameter-

```
real(R_), save :: Wld_dtaumin = 0.0_R_
```

This has different effects on extinction coefficient scaling, depending on target radiative quantities.

When fluxes and heating rates are target or the local collision forcing (LCF) is not used ($Wld_nplcf = 0$), $Wld_dtaumin$ is a minimum layer optical thickness for collision forcing. If optical thickness of one layer is less than this threshold, then the collision-forcing method is used in that layer, and the scaled value of the optical thickness is set to $Wld_dtaumin$. Thus collision events are forced to occur with prescribed probability even in very optically thin layer.

When radiances are target and the LCF is used, then $Wld_dtaumin$ is used for the LCF. See the description of [Wld_nplcf](#) for how each parameter is used in the code.

[Edit](#)

Namelist [mcarAtm_nml_job](#)

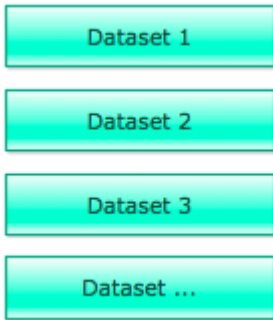
Several 1-D vertical profiles of some variables can be specified in this namelist. Data for 3-D distribution of properties for horizontally-inhomogeneous layers are usually very large. Therefore, they should be read in from the external data file. The 1-D and 3-D profile data are mixed and used in the code. Please work with care for how they are mixed, which could be different by variable. See [here](#) (in Chapter 3) for how homogeneous and inhomogeneous layers are defined.

Atm_idread

```
integer, save :: Atm_idread = 1
```

This controls how to import atmospheric dataset for the current job. Definition depends on the file format: binary or text. In both cases, if $Atm_idread = 0$, then namelist input for Atm_ext1d , Atm_omg1d , Atm_apf1d , and Atm_abs1d are all neglected. For the first job, Atm_idread should be > 0 .

This variable also controls what dataset is read in from the external data file, Atm_infile , for the current job. Note that it is assumed that multiple datasets are possibly stored in the file as in the figure below.



Binary format : Atm_mfmt = 1

Variable Atm_idread is location index of data set to be read, in the 3-D atmospheric data file. One of data sets in the file can be chosen for each simulation job. The user can specify explicitly the data set index by this parameter or leave it as default value, which is 1 for the first job, and 0 for the subsequent jobs.

If Atm_idread = 0, then no new data set is read in for the current job, and atmospheric properties are the same as previous job. This is possible for second or subsequent job.

As described here, there are many ways to specify explicitly or implicitly the data set index for a job. For example,

- As default for all jobs

```
Job index      1 2 3 4 5 6
Atm_idread    - - - - -
Index of used data set 1 1 1 1 1 1
```

- Common single data set for all jobs

```
Job index      1 2 3 4 5 6
Atm_idread     5 - - - -
Index of used data set 5 5 5 5 5 5
```

- To give explicitly all the indexes

```
Job index      1 2 3 4 5 6
Atm_idread     5 2 1 3 5 4
Index of used data set 5 2 1 3 5 4
```

- A combined approach

```
Job index      1 2 3 4 5 6
Atm_idread     5 - - 3 0 4
Index of used data set 5 5 5 3 3 4
```

where "-" denotes that no value for Atm_idread is given in the namelist.

Text format : Atm_mfmt = 0

Variable Atm_idread is on/off flag for data import. If 0, no data is read in for the current job, and previously imported data are used again. If 1, then a new data set is read in from the file. For text format file, there is no possibility of random access as in binary format. Thus data sets are read in sequentially for each job.

Atm_dx, Atm_dy

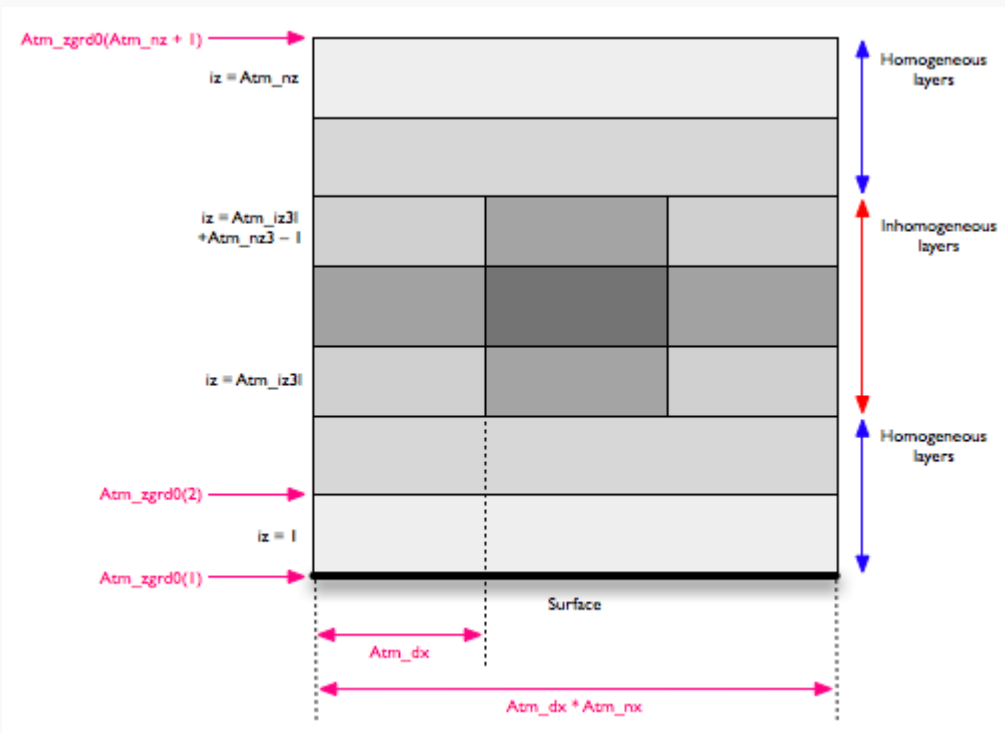
```
real(R_), save :: Atm_dx = 1.0e+4_R_ ! X size of cell
real(R_), save :: Atm_dy = 1.0e+4_R_ ! Y size of cell
```

Cell sizes (m) along x and y-axes, respectively.

Atm_zgrd0

```
real(R_), save :: Atm_zgrd0(1:KNZ+1) = ((10.0_R_*i, i = 1, KNZ+1)/)
```

Z-coordinate values (m) of layer boundaries. The bottom and top of a layer with index iz correspond to z=Atm_zgrd0(iz) and Atm_zgrd0(iz+1), respectively. The BOA (surface) corresponds to Atm_zgrd0(1), and TOA Atm_zgrd0(Atm_nz+1). The following illustration shows the definitions of several geometrical parameters.



Atm_wkd0

```
real(R_), save :: Atm_wkd0(KNKD) = (/1.0_R_, (0.0_R_, i = 1, KNKD - 1)/) ! k-distribution weights
```

Weight coefficients of the correlated k-distribution. Give $Atm_wkd0(1)=1$ for monochromatic computation.

Atm_tmp1d

```
real(R_), save :: Atm_tmp1d(1:KNZ+1) = (/300.0_R_, i = 1, KNZ+1)/)
```

Horizontally-averaged atmospheric temperatures (K) defined

- at layers [1:Atm_nz] if $Atm_mtprof=0$, or
- at layer interfaces [1:Atm_nz+1] if $Atm_mtprof=1$.

Used for thermal emission when $Src_mtype = 2$ or 3 .

When $Atm_mtprof=1$, an interface of $iz=1$ corresponds to BOA, and that of $iz=Atm_nz+1$ corresponds to TOA. Note that Atm_tmp1d is atmospheric temperature even when $iz=0$ (BOA). The surface temperature should be given by different variables. For horizontally inhomogeneous layer, data of perturbation in temperature could be specified in external data file.

Atm_ext1d

```
real(R_), save :: Atm_ext1d(KNZ, KNP1D)
```

Extinction coefficients (/m) for each layer and each scattering component, one of which can be gas, aerosol, hydrometeor, or their mixture. Extinction coefficient $Atm_ext1d(iz, ip1d)$ corresponds to a layer iz and a component $ip1d$. Note that optical properties specified by Atm_*1d variables are assumed be horizontally homogeneous in every atmospheric layers.

Atm_omg1d

```
real(R_), save :: Atm_omg1d(KNZ, KNP1D)
```

As Atm_exp1d , but for single scattering albedo.

Atm_apf1d

```
real(R_), save :: Atm_apf1d(KNZ, KNP1D)
```

As Atm_exp1d , but for phase function specification parameters. Phase functions are specified in the following way:

- $Atm_apf1d \leq -2$: isotropic scattering phase function
- $-2 < Atm_apf1d \leq -1$: Rayleigh scattering phase function
- $-1 < Atm_apf1d < 1$: Henyey-Greenstein phase function with asymmetry parameter of Atm_apf1d
- $1 \leq Atm_apf1d$: a tabulated phase function (given by the database file, see the namelist `mcaSca_nml_init`)

Note `Atm_apf1d` should be \leq `Sca_npf` (# of phase functions read in from the file). Even when `Atm_apf1d` \geq 1, this variable does not need to be integer. The phase function is assumed to follow an equation:

$$P = (1 - a) * P_tab(i) + a * P_tab(i+1)$$

where

$$i = \text{int}(\text{Atm_apf1d})$$
$$a = \text{Atm_apf1d} - i$$

That is, the phase function is linearly interpolated between `P_tab(i)` and `P_tab(i+1)`. Thus, for better accuracy, the parameter `Atm_apf1d` should be given, taking into account a relationship between the phase function and a physical property of particles. For example, note that the phase function is linear to the inverse of the effective radius (`Re`), as a first order approximation:

$$P \sim c + d / Re$$

Atm_abs1d

```
real(R_), save :: Atm_abs1d(KNZ, KNKD)
```

Horizontally-averaged absorption coefficients (/m) of purely-absorbing component, which can be gaseous mixture for instance. The correlated k-distribution is assumed. An element `Atm_abs1d(iz, ikd)` corresponds to a layer `iz` and a k-distribution index `ikd`. Note perturbations of the absorption coefficients in the horizontally inhomogeneous layers can be given by the data file (`Atm_inpfile`).

Scaling optical properties

Input extinction and absorption coefficients can be easily scaled by user-specified factors, as described below. The scaling is performed immediately after reading in the data (from the namelist file or from 3-D atmospheric data file). Variables, `Atm_fext1d`, `Atm_fext3d`, `Atm_fabs1d`, `Atm_fabs3d`, have been introduced in the namelist. These are useful for simple adjustments of input data. For example, someone could try two simulations easily: one for cloudy sky with 3-D cloud extinction, the other without the cloud (e. g., by setting `Atm_fext3d = 0` for cloud).

Note that these parameters are effective only when `Atm_idread` \geq 1. When `Atm_idread = 0`, no new dataset is read in, and no data scaling is applied.

Atm_fext1d

```
real(R_), save :: Atm_fext1d(KNP1D) = ((1.0_R_, i = 1, KNP1D)/)
```

Scaling factor for `Atm_ext1d(ip1d)`, where `ip1d` is component index. This is neglected when `Atm_idread = 0`: a new data set is not read in for the current job.

This provide a simple way to modify the original extinction coefficients given by `Atm_ext1d`. The data are modified just after reading in the namelist data, in the following way:

$$B_ext(iz, ip1d) = \text{Atm_ext1d}(iz, ip1d) * \text{Atm_fext1d}(ip1d)$$

for all `iz = [1, Atm_nz]`.

Note the values of `Atm_ext1d` are untouched by this scaling and modified data are saved in a different variable array. Therefore, values of `Atm_ext1d` will be saved in the subsequent jobs.

Atm_fext3d

```
real(R_), save :: Atm_fext3d(KNP3D) = ((1.0_R_, i = 1, KNP3D)/)
```

As `Atm_fext1d`, but for modifying `Atm_ext3d`, 3-D spatial distribution of extinction coefficients. This is neglected when `Atm_idread = 0`: a new data set is not read in for the current job.

The 3-D data are read in from the external data file (`Atm_inpfile`). The data are modified just after reading in the 3-D data, in the following way:

$$\text{Atm_ext3d}(ix, iy, iz3, ip3d) = \text{Atm_ext3d}(ix, iy, iz3, ip3d) * \text{Atm_fext3d}(ip3d)$$

for all `iz3 = [1, Atm_nz3]`.

Note that a new data set is not always read in for each job and that the modification by `Atm_fext3d` is applied only when a new data set is read in. See also `Atm_idread` for when a new data set is read in for a job.

Atm_fabs1d

```
real(R_), save :: Atm_fabs1d = 1.0_R_
```

Scaling factor for `Atm_abs1d`. This is neglected when `Atm_idread = 0`: a new data set is not read in for the current job.

This provide a simple way to modify the original absorption coefficients given by `Atm_abs1d`. The data are modified just after reading in the namelist data, in the following way:

```
B_abs(iz, ikd) = Atm_abs1d(iz, ikd) * Atm_fabs1d
```

for all $iz = [1, \text{Atm_nz}]$ and all $ikd = [1, \text{Atm_nkd}]$.

Note the values of `Atm_abs1d` are untouched by this scaling and modified data are saved in a different variable array. Therefore, values of `Atm_abs1d` will be saved in the subsequent jobs.

Atm_fabs3d

```
real(R_), save :: Atm_fabs3d = 1.0_R_
```

As `Atm_fabs1d`, but for modifying `Atm_abs3d`, 3-D spatial distribution of absorption coefficients. This is neglected when `Atm_idread = 0`: a new data set is not read in for the current job.

The 3-D data are read in from the external data file (`Atm_inpfile`). The data are modified just after reading in the 3-D data, in the following way:

```
Atm_abs3d(ix, iy, iz3, ikd) = Atm_abs3d(ix, iy, iz3, ikd) * Atm_fabs3d
```

for all $iz3 = [1, \text{Atm_nz3}]$ and all $ikd = [1, \text{Atm_nkd}]$.

Note that a new data set is not always read in for each job and that the modification by `Atm_fabs3d` is applied only when a new data set is read in. See also `Atm_idread` for when a new data set is read in for a job.

Atm_tauin -technical parameter-

```
real(R_), save :: Atm_tauin = 2.0_R_
```

Minimum column optical thickness used for collision forcing. If actual column optical thickness is less than this threshold, then the collision forcing method is used, and the scaled value of the total optical thickness is set to `Atm_tauin`. Recommended value is between 1 and 10.

Atm_fsupg -technical parameter-

```
real(R_), save :: Atm_fsupg = 2.0_R_
```

Geometrical threshold parameter for automatic, adaptive super-cell formation for maximum cross section method. In the preprocess, super-cells are formed by merging cells until one of conditions are met. Cells are merged until super-cell size (m) is larger, on average, than

```
Atm_fsupg / E_max
```

where `E_max` is the maximum extinction coefficient (/m).

This is one of three conditions for adaptive super-cell formation. See also `Atm_fsupi` and `Atm_nzzsup`.

Atm_fsupi -technical parameter-

```
real(R_), save :: Atm_fsupi = 1.3_R_
```

Inhomogeneity threshold parameter for automatic, adaptive super-cell formation for maximum cross section method. Cells are merged when inhomogeneity of extinction coefficients within the super-cell is small enough, on average. If ratio of the maximum extinction coefficient (`E_max`) to super-cell average extinction coefficient (`E_ave`) is larger than `Atm_fsupi`, then cell merging will be stopped:

```
E_max / E_ave > Atm_fsupi
```

This is one of three conditions for adaptive super-cell formation. See also `Atm_fsupg` and `Atm_nzzsup`.

Atm_nzzsup -technical parameter-

```
integer, save :: Atm_nzzsup = 50
```

Possible max number of atmospheric layers merged into a single super-cell, in which the maximum cross section method is used.

[Edit](#)

Namelist [mcarSfc_nml_job](#)

Sfc_idread

```
integer, save :: Sfc_idread = 1
```

Location index of data set to be read, in the surface data file (`Sfc_inpfile`). Data set specification method is the same as `Atm_idread`. See `Atm_idread` for details.

Sfc_tmp

```
real(R_), save :: Sfc_tmp = 300.0_R_
```

Average surface temperature (K). Used for calculation of thermal emission when Src_mtype = 2 or 3. 2-D temperature perturbation can be specified by the external data file (Sfc_infile).

Sfc_mtype

```
integer, save :: Sfc_mtype = 1
```

Type index of globally constant surface BRDF. Used only when Sfc_infile = '', i.e. BRDF data are not given by the file, Sfc_infile. BRDF is assumed as follows:

```
0: black
1: Lambertian
2: Flx_diffuse-specular mixture (DSM) model
3: Rahman-Pinty-Verstraete (RPV) model
4: Li-Sparse-Ross-Thick (LSRT) model
```

Sfc_param

```
real(R_), save :: Sfc_param(Sfc_NPAR) = (/1.0_R_, 0.0_R_, 0.0_R_, 0.0_R_, 0.0_R_/)
```

BRDF parameters for globally constant surface properties. Used only when Sfc_infile = '', i.e. BRDF data are not given by the file, Sfc_infile. Definitions of Sfc_param(ipar) for respective ipar differs by the BRDF type, Sfc_mtype.

Sfc_mtype = 0 (black): No specification is required for Sfc_param, which is dummy parameter.

Sfc_mtype = 1 (Lambertian):

- Sfc_param(ipar=1): albedo

Sfc_mtype = 2 (DSM): The surface reflection is modeled by a microscopical mixture of diffuse and specular reflection.

- Sfc_param(ipar=1): diffuse albedo (A_d)
- Sfc_param(ipar=2): fraction of Flx_diffuse reflectors (f_d); Fraction of specular reflection is 1 - Sfc_param(ipar=2)
- Sfc_param(ipar=3): real part of refractive index of underlying material (e.g., water); Should be given as positive.
- Sfc_param(ipar=4): imaginary part of refractive index of underlying material (e.g., water); Should be given as positive.
- Sfc_param(ipar=5): variance of micro-scopical surface slope (tangent of the inclination angle). Give zero for flat surface. This is used for specular reflection. According to Cox and Munk's (1954) parameterization for ocean surface,

$$Sfc_param(ipar=5) = 0.00512 * V + 0.003.$$

where V is wind velocity (m/s) at 10 m above the ocean surface.

The DSM model albedo (A) and BRDF (R) follow

$$\begin{aligned} A &= f_d * A_d + (1 - f_d) * A_s, \\ R &= f_d * R_d + (1 - f_d) * R_s, \\ \pi * R_d &= \cos Q * A_d \end{aligned}$$

where A_s and R_s are albedo and BRDF for specular reflection.

Sfc_mtype = 3 (RPV): Rahman-Pinty-Verstraete (RPV) BRDF model

- Sfc_param(ipar=1): the first parameter, alpha
- Sfc_param(ipar=2): the exponent parameter, k
- Sfc_param(ipar=3): the asymmetry parameter, THETA
- Sfc_param(ipar=4): delta

The RPV BRDF, R, is given as

$$\pi * R = \alpha * [u_0 * u_1 / (u_0 - 1) / (u_1 - 1)]^{(k - 1)} * HG(THETA) * F(G)$$

where $\pi=3.141592653$, u_0 and u_1 are respectively cosines of incidence and reflection vectors, and HG is the Heyney-Greenstein function. The function F is as follows:

$$F(G) = 1 + (1 + G) / (\delta + G),$$

where G is a geometrical parameter.

Sfc_mtype = 4 (LSRT): Li-Sparse-Ross-Thick (LSRT) BRDF model

- Sfc_param(ipar=1): "k_L", weight of Lambertian reflection
- Sfc_param(ipar=2): "k_g", weight of geometrical optics reflection
- Sfc_param(ipar=3): "k_v", weight of volume scattering

The LSRT BRDF, R, is given by

$$\pi * R = k_L * \pi + k_g * K_{geo} + k_v * K_{vol}$$

where K_{geo} and K_{vol} are geometrical and volume scattering kernel functions.

Sfc_nudsm, Sfc_nurpv, Sfc_nulsrt -technical parameter-

```
integer, save :: Sfc_nudsm = 14
integer, save :: Sfc_nurpv = 8
integer, save :: Sfc_nulsrt = 14
```

Number of solar zenith angles for the look-up tables (LUTs) for DSM, RPV, and LSRT BRDF models, respectively. The LUTs contains parameters for albedo calculation and determination of reflection direction. In simulations, the parameters used are interpolated from the LUT. Thus larger Sfc_nudsm value achieves higher accuracy but requires larger memory for the LUTs.

Sfc_nqpot -technical parameter-

```
integer, save :: Sfc_nqpot = 24
```

Number of quadrature points for surface albedo pre-calculations for the BRDF models, DSM and RPV models. Used when creating the LUTs for albedo, which is calculated by integration of the BRDF over the hemisphere. Larger quadrature points can achieve accurate results but may take longer CPU time. Usually 10 is enough to achieve high accuracy as 0.5% (relative unit).

Sfc_rrmax -technical parameter-

```
real(R_), save :: Sfc_rrmax = 5.0_R_
```

Parameter for determination of random reflection direction by the BRDF models, DSM, RPV, and LSRT models. The rejection method is used for sampling. Smaller value needs less computer time, but the determined reflection direction is more approximate. If Sfc_rrmax is very large ($>1.0e+8$), then no such an approximation is used, and the simulation may require nearly infinite CPU time! Should be larger than or equal to 1. Default value is generally optimal.

Sfc_rrexp -technical parameter-

```
real(R_), save :: Sfc_rrexp = 0.5_R_
```

Parameter for determination of random reflection direction by the BRDF models, DSM and RPV models. Smaller value needs less computer time but the determined reflection direction is more approximate. If Sfc_rrexp=1, then no such an approximation is used, and the simulation may require nearly infinite CPU time! Should be between 0 and 1. Default value is generally optimal.

[Edit](#)

Namelist [mcarSrc_nml_job](#)

Src_mtype

```
integer, save :: Src_mtype(KNSRC) = ((1, i = 1, KNSRC)/)
```

Flag for source type (0,1,2,3). Defines external and/or internal sources, as follows:

- 0: localized sources (the location, direction, and angular width should be specified)
- 1: solar sources (incidence from the top of atmosphere)
- 2: solar plus thermal emission sources
- 3: thermal emission sources

Src_wlen

```
real(R_), save :: Src_wlen(KNSRC) = ((10.0_R_, i = 1, KNSRC)/)
```

Wavelength (micrometer) of light for thermal emission. This is used only when Src_mtype = 2 or 3. If the wavelength band width (Src_dwlen) is not zero, then Src_wlen is the lower limit of wavelength. See also Src_dwlen.

Src_dwlen

```
real(R_), save :: Src_dwlen(KNSRC) = ((0.1_R_, i = 1, KNSRC)/)
```

Wavelength band width (micrometer). Zero is acceptable. When Src_dwlen=0, the simulation is completely monochromatic, and output quantities are spectral ones. Otherwise (Src_dwlen > 0), output quantities are spectrally averaged ones, and spectrally averaged thermal emission are calculated when Src_mtype = 2 or 3.

Note on output quantities and physical units:

a) Src_dwlen=0: spectral, monochromatic calculations, as the followings

- area-averaged spectral flux densities (e.g., W/m²/micrometer),
- volume-averaged spectral heating rates (e.g., convergences, W/m³/micrometer),
- spectral radiances (e.g., W/m²/steradian/micrometer),

b) Src_dwlen > 0: spectrally-averaged calculations; Units are the same as above.

Src_flg

```
real(R_), save :: Src_flg(KNSRC) = ((1.0_R_, i = 1, KNSRC)/)
```

Spectral or spectrally-averaged source flux density or power.

- When the solar source is present (Src_mtype = 1 or 2), Src_flg is the solar source spectral irradiance (W/m²/micrometer) at the top of atmosphere. The Src_flg is defined as integrated on the plane normal to the center direction of the source emission. The top-of-atmosphere solar source irradiance on the horizontal plane should be, for collimated solar incidence (Src_qmax=0),

```
Src_flg * abs(cos(180 - Src_the))
```

- When the source is localized (Src_mtype = 0), Src_flg is the emitted spectral power (W/micrometer).

Even if Src_dwlen > 0, Src_flg should be spectrally-averaged quantity, averaged over the wavelength band. Then spectrally integrated solar source irradiance should be Src_dwlen*Src_flg, for example.

Src_qmax

```
real(R_), save :: Src_qmax(KNSRC) = ((0.0_R_, i = 1, KNSRC)/)
```

Full cone angle (degrees) that determines angular width of solar or localized source, used only when Src_mtype = 0, 1 or 2. This should be 0.533133 (degree) for solar incidence at an annual-mean state of the sun-earth system. Give zero for cases in which sources are negligibly small or infinitely far.

Src_the, Src_phi

```
real(R_), save :: Src_the(KNSRC) = ((120.0_R_, i = 1, KNSRC)/) ! zenith angle  
real(R_), save :: Src_phi(KNSRC) = ((0.0_R_, i = 1, KNSRC)/) ! azimuth angle
```

Zenith and azimuth angles (degrees), respectively, of the source emission direction vector. Used only when Src_mtype = 0, 1 or 2. Note the source direction is defined as transport direction of source light. If the source is solar light, then the transport direction corresponds to the sun-to-surface vector. Thus, the solar zenith angle should be (180 - Src_the).

Src_mphi

```
integer, save :: Src_mphi(KNSRC) = ((0, i = 1, KNSRC)/)
```

Flag for random azimuth of source emission (0:no, 1:yes). If Src_mphi = 0, then the azimuth angles of the source vectors are set as Src_phi. Else if Src_mphi = 1, then the azimuth angles are set as random between 0 to 360 degrees.

Src_xpos, Src_ypos

```
real(R_), save :: Src_xpos(KNSRC) = ((0.5_R_, i = 1, KNSRC)/) ! X relative position  
real(R_), save :: Src_ypos(KNSRC) = ((0.5_R_, i = 1, KNSRC)/) ! Y relative position
```

Coordinates that specifies the location of the localized source, used when Src_mtype=0.

Src_zloc

```
real(R_), save :: Src_zloc(KNSRC) = ((0.0_R_, i = 1, KNSRC)/)
```

Z-coordinate that specifies the location of the localized source, used when Src_mtype=0.

Src_apsize

```
real(R_), save :: Src_apsize(KNSRC) = ((0.0_R_, i = 1, KNSRC)/)
```

Aperture size, diameter (m) of the local source surface (or the surface of the outmost lens of the artificial source), when Src_mtype = 0. The localized source should be very small compared to the model domain, but non-zero finite size is allowed. If Src_apsize = 0, then it means the source is from a point. Otherwise, the source surface is assumed to be circular. Emission surface is assumed to be on a plane perpendicular to the emission center direction (with Src_the & Src_phi).

[Edit](#)

Namelist [mcarRad_nml_job](#)

Rad_mrproj

```
integer, save :: Rad_mrproj = 0
```

Flag for angular weighting, when radiances are averaged over finite solid angle (Rad_mrkind = 1 or 3). Options are as follows:

- = 0 : w = 1, results are radiances simply averaged over solid angle
- = 1 : w = cosQ for Q = angle from the camera center direction, results are weighted average

where w is a weighting function. Output radiance will be

$$R_{\text{out}} = \frac{\int R(s) * w(s) * dS}{\int w(s) * dS}$$

where R(s) is radiance in a direction s, and S is solid angle within the FOV. Here are two examples. When FOV is a hemisphere (Rad_umax = 180 deg.),

- Rad_mrproj = 0: R_out = hemispherical-mean radiance
- Rad_mrproj = 1: R_out * pi = irradiance (radiative flux density)

Rad_zloc

```
real(R_), save :: Rad_zloc(KNRAD) = (( 0.0_R_, i = 1, KNRAD)/)
```

Z coordinate, altitude (m) of the detector location.

The pixel mapping for radiance image is performed at the detector altitude. Note that this differs from the method that is used for image projection of satellite observation image pixels, which are mapped at the ground level (not at the satellite altitude). In the next version, more useful option will be added.

Rad_xpos, Rad_ypos

```
real(R_), save :: Rad_xpos(KNRAD) = (( 0.5_R_, i = 1, KNRAD)/) ! X relative position  
real(R_), save :: Rad_ypos(KNRAD) = (( 0.5_R_, i = 1, KNRAD)/) ! Y relative position
```

Relative horizontal location along x and y-axes, respectively, for the detector. X and Y coordinates at the detector location are respectively

$$X_{\text{cam}} = \text{Rad_xpos} * \text{Atm_dx} * \text{Atm_nx}$$
$$Y_{\text{cam}} = \text{Rad_ypos} * \text{Atm_dy} * \text{Atm_ny}$$

Thus, Rad_xpos=0 corresponds to the X coordinate of 0, and Rad_xpos=1 corresponds to the uppermost X coordinate of the domain. If Rad_xpos or Rad_ypos is out of a range [0, 1], then they are reset in the code under the cyclic boundary condition. For example, Rad_xpos = 2.345 will be reset as 0.345, and -2.345 will be reset as 0.655.

Rad_rmin0, Rad_rmax0

```
real(R_), save :: Rad_rmin0(KNRAD) = ((1.0e-17_R_, i = 1, KNRAD)/)  
real(R_), save :: Rad_rmax0(KNRAD) = ((1.0e+17_R_, i = 1, KNRAD)/)
```

Minimum and maximum distances (m), respectively, between emission point and the detector. Used only when Rad_mrkind = 1 or 2, in order to limit the distance between emission/scattering point and the detector. A view region is defined for each detector, and the distance R between the emission point and the detector should be

$$\text{Rad_rmin0} < R < \text{Rad_rmax0}$$

Emissions from out of the view region are all excluded from sampling. Transmittance between the line of sight segment for $0 < R < \text{Rad_rmin0}$ is artificially set at 1, which may result in bias of computed radiances.

Note current code has an intelligent way to automatically set the view region: Rad_rmin0, Rad_rmax0, and Rad_frmod are automatically set taking into account the domain size, detector location, and detector FOV. When Rad_mrkind = 2, default values are Rad_rmin0 = 0.0 and Rad_rmax0 = (a large enough value). Therefore, the user can leave the default setting as is. Alternatively

set an explicit view region.

Basically, emission contributions from source emission, scattering, and reflection should be all sampled for calculating radiances. However, if the first kind of radiances are calculated ($Rad_mrkind = 1$), simulated images can be very noisy when Rad_rmin0 is too small, and scattering particles are present near the camera. This is because contribution function is proportional to $1/R^2$. For that reason, emission near the camera should be excluded from sampling.

For simulation of physically-correct results, Rad_rmax0 should be infinity. However, such a simulation could be highly time consuming, so that an upper limit should be set.

The figure below schematically illustrates the geometry of the view region and the detector.

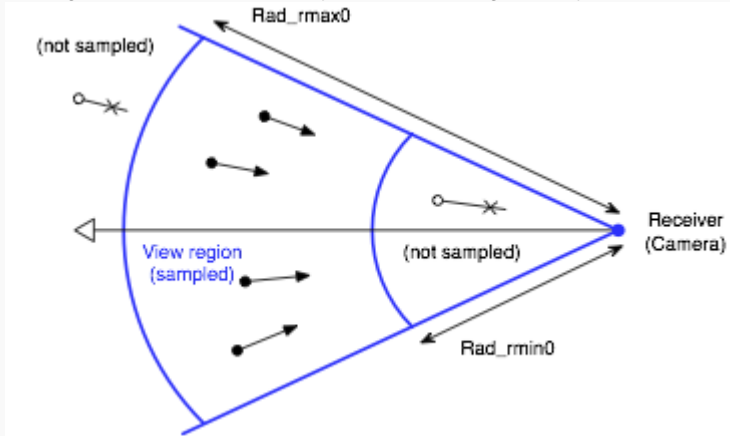


Figure. The geometry of the view region and the detector.

Rad_frmod

```
real(R_), save :: Rad_frmod = 0.3_R_
```

Factor for mode distance between emission point and the detector. Used only when the first kind of radiances are calculated ($Rad_mrkind = 1$). The mode distance (R_mod) will be

$$R_mod = [(1 - Rad_frmod) * \sqrt{Rad_rmin0} + Rad_frmod * \sqrt{Rad_rmax0}] ** 2$$

Contribution function for the first kind of radiance is proportional to $1/R^2$. This mode distance is used for variance reduction. Emission contribution from a point far enough from the detector ($R > R_mod$), is scaled by the Russian roulette:

$$\begin{aligned} Zeta_new &= Zeta_old * (R / R_mod), \text{ if } Xi < R_mod / R \\ Zeta_new &= 0, \text{ otherwise} \end{aligned}$$

where Xi is a random number. This reduces populations of sampling from points far from the detector, reducing calculation time.

Rad_the, Rad_phi, Rad_psi

```
real(R_), save :: Rad_phi(KNRAD) = ((0.0_R_, i = 1, KNRAD)/) ! phi, angle around Z0
real(R_), save :: Rad_the(KNRAD) = ((0.0_R_, i = 1, KNRAD)/) ! theta, angle around Y1
real(R_), save :: Rad_psi(KNRAD) = ((270.0_R_, i = 1, KNRAD)/) ! psi, angle around Z2
```

Rotation angles (degrees) for the camera (detector) coordinates, which are defined by rotating the original X-Y-Z world coordinates, following Z-Y-Z rotation rule. Values can be in a range (-Inf, +Inf). Three rotations are operated in the correct order:

- 1: rotation about Z (Z-axis in (X,Y,Z) world coordinates) by Rad_phi
- 2: rotation about Y1 (Y-axis in (X1,Y1,Z1) coordinates) by Rad_theta
- 3: rotation about Z2 (Z-axis in (X2,Y2,Z2) coordinates) by Rad_psi

Finally camera coordinates are (Xc, Yc, Zc) , and the Zc axis corresponds to the reverse radiance direction (when $Rad_mrkind = 2$) or the center direction of radiance image projection (when $Rad_mrkind = 1$ or 3). Angle between Zc axis direction and the original Z axis direction should be Rad_theta , and azimuth angle of Zc axis direction in the original world coordinate system (X, Y, Z) should be Rad_phi . Therefore, Rad_theta and Rad_phi denotes the zenith and azimuth angles, respectively, in degrees, of the radiance reverse direction or the center direction of radiance image projection. The direction is here defined as the reverse of light transport, i.e. camera to target.

Rad_psi is dummy when $Rad_mrkind = 2$, i.e., detector FOV is infinitesimal. Rad_psi should be appropriately set when $Rad_mrkind = 1$ or 3 , because the angular distribution of radiance will be mapped in an image, definition of which requires the rotation angle Rad_psi .

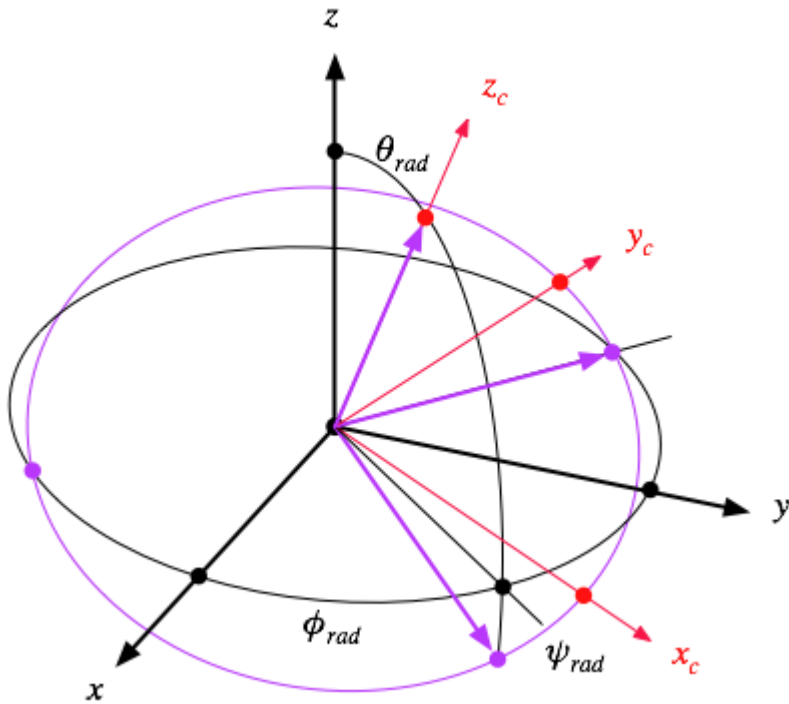


Figure. The camera coordinates and world coordinates, where theta_rad = Rad_theta, phi_rad = Rad_phi, psi_rad = Rad_psi.

Camera coordinate system

The camera coordinates (Xc,Yc,Zc) are defined as described above, by the three angles (Rad_the, Rad_phi, Rad_psi). When Rad_mrkind = 2 (horizontal distribution of area-averaged radiances in specific direction), Zc axis is the single, specific direction. When Rad_mrkind = 1 or 3 (angular distribution of angle-averaged radiances), view directions are mapped in an image, which is defined for each camera. In the camera coordinate system, an arbitrary view direction (view point-to-target direction) is defined by the polar angle (theta_c) and the azimuth angle (phi_c), as illustrated in the figure below. The projection image coordinates are defined by U and V, as described below.

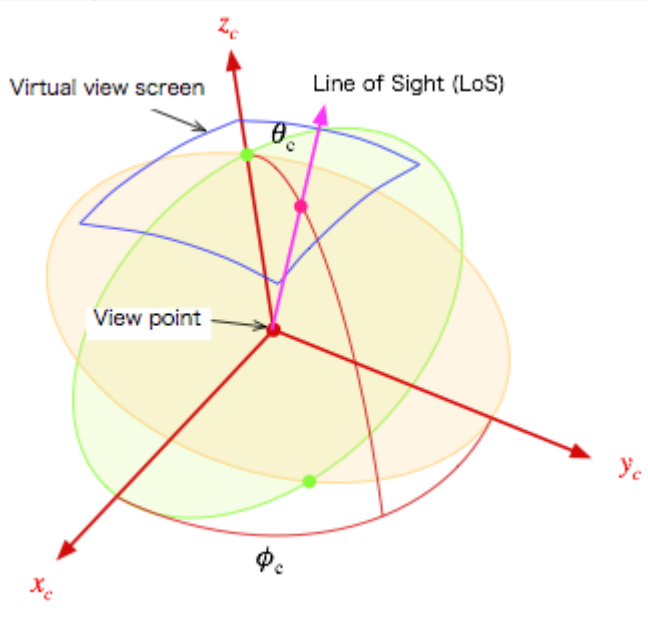


Figure. Camera coordinates and a view direction.

Rad_umax , Rad_vmax

```
real(R_) , save :: Rad_umax(KNRAD) = (/ (180.0_R_ , i = 1, KNRAD) /) ! max angle along U coordinate
real(R_) , save :: Rad_vmax(KNRAD) = (/ (180.0_R_ , i = 1, KNRAD) /) ! max angle along V coordinate
```

Maximum angles (degrees) of projection image coordinates, U and V, respectively, for each camera. Used only when Rad_mrkind = 1 or 3. Both should be <= 180 degrees. The pixel mapping method differs by Rad_mpmmap.

Rad_mpmmap = 1 (rectangular mapping)

```
U = theta_c * cos(phi_c) + Rad_umax / 2, in [0, Rad_umax]
V = theta_c * sin(phi_c) + Rad_vmax / 2, in [0, Rad_vmax]
```

Rad_mpmmap = 2 (polar mapping)

$$U = \theta_c * 2, \text{ in } [0, \text{Rad_umax}]$$

$$V = \phi_c, \text{ in } [0, 360]$$

where θ_c and ϕ_c are the polar and azimuth angles in the camera coordinate system, of the view direction. Figures below show the image geometry schematically. See also Rad_qmax and [a page](#) for example applications. When Rad_mpmmap = 2, Rad_vmax is not used.

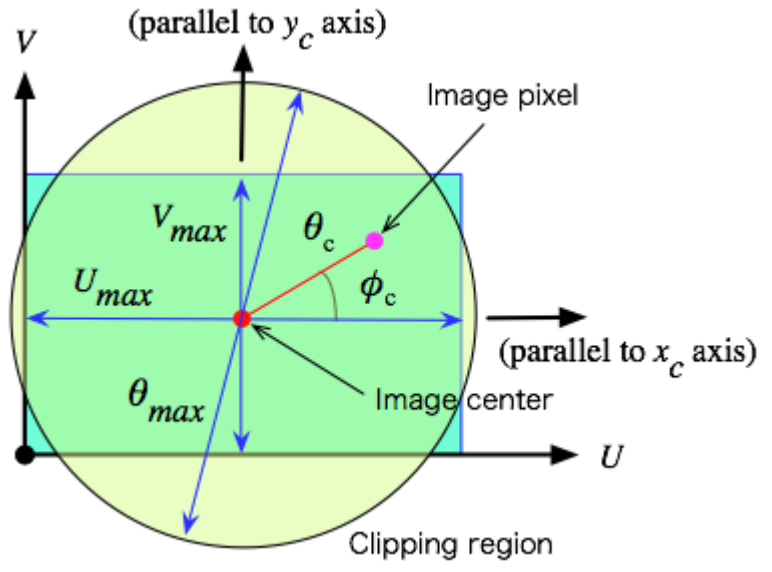


Figure. Rectangular image mapping (Rad_mpmmap=1).

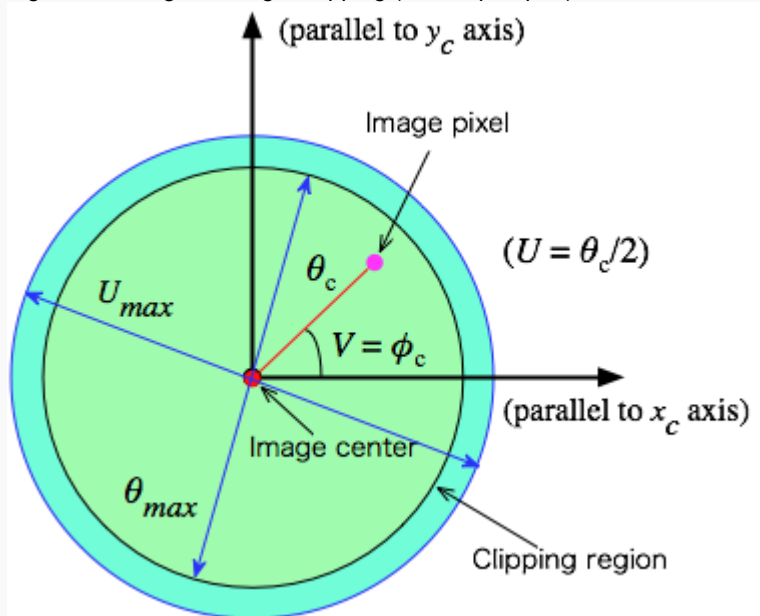
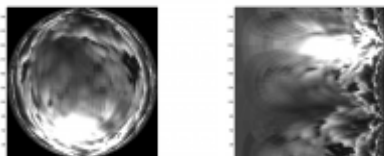


Figure. Polar image mapping (Rad_mpmmap=2).

Here is an example: Both show radiance angular distribution for an hemisphere, looking up a stratocumulus cloud deck. In both cases, Rad_umax = 180, and Rad_vmax = 180 (Rad_vmax is dummy when Rad_mpmmap = 2).

Rad_mpmmap=1 2



Rad_qmax

$$\text{real}(R_), \text{ save } :: \text{Rad_qmax}(\text{KNRAD}) = (/ (180.0_R_ , i = 1, \text{KNRAD}) /)$$

Maximum angle (degrees), full cone angle of the field of view (FOV) of the detector. Should be ≤ 180 degrees. If Rad_qmax = 180 degrees, the the FOV corresponds to a hemisphere, the normal of which corresponds to the direction with Rad_theta and Rad_phi.

Rad_apsize

```
real(R_), save :: Rad_apsize(KNRAD) = (/ (0.0_R_, i = 1, KNRAD) /)
```

Aperture size, diameter (m) of the detector surface (or the surface of the outmost lens of the detector). Used only when the first kind of radiances are calculated (Rad_mrkind = 1).

The detector size should be very small compared to the model domain, but non-zero finite size is allowed. If Rad_apsize = 0, then it means an infinitesimally small detector. Otherwise, the detector surface is assumed to be circular. Detector surface is assumed to be on a plane perpendicular to the FOV center direction.

Rad_wfunc0

```
real(R_), save :: Rad_wfunc0(KNZ, KNWF)
```

Weighting functions used when column averaged pathlengths are calculated (Rad_mplen=2). Unit is arbitrary. An element Rad_wfunc0(iz, iwf) corresponds to a layer index iz and a weighting function index iwf. The weighting function is used in the following formula: air mass factor (AMF),

$$\text{AMF}(iwf) = \frac{\sum \{ L(iz) * \text{Rad_wfunc0}(iz, iwf) \}}{\sum \{ \text{Rad_wfunc0}(iz, iwf) \}} / \text{Dz}(iz)$$

where the sum is for all layers, iz = [1, Atm_nz], and L is pathlength and Dz is layer depth.

Multiple weighting functions are useful for calculating multiple air mass factors for multiple gaseous species with different vertical profiles.

Rad_zetamin -technical parameter-

```
real(R_), save :: Rad_zetamin = 0.3_R_
```

Threshold for each radiance contribution function sampled by the local estimation method. Radiance contribution from each emission/scattering/reflection event is calculated as the normalized angular distribution function multiplied by transmittance between the event point and the detector.

Rad_difr0, Rad_difr1 -technical parameter-

```
real(R_), save :: Rad_difr0 = 30.0_R_  
real(R_), save :: Rad_difr1 = 0.01_R_
```

Artificial diffusion coefficients for radiances. Default values are generally optimal.

What variables are actually used?

Different sets of variables are used depending on the kind of radiance (Rad_mrkind). The following table summarizes which variables are used for a given Rad_mrkind.

Here, "y" denotes that the variable is used, and "-" not used.

Rad_mrkind	1	2	3	Only when	Rad_mrkind	1	2	3	Only when
Rad_zloc	y	y	y	*	Rad_rmin0	y	y	-	*
Rad_wfunc0	y	y	y	Rad_mplen>=1	Rad_rmax0	y	y	-	*
Rad_the	y	y	y	*	Rad_frmod	y	-	-	*
Rad_phi	y	y	y	*	Rad_apsize	y	-	-	*
Rad_psi	y	-	y	*	Rad_xpos	y	-	-	*
Rad_mrproj	y	-	y	*	Rad_ypos	y	-	-	*
Rad_umax	y	-	y	*					
Rad_vmax	y	-	y	Rad_mpmmap=2					
Rad_qmax	y	-	y	*					

[Edit](#)

[Next](#) / [Return](#)