S3COM

0.9.0

Generated by Doxygen 1.9.6

Chapter 1

Source file repositories

Brief general description of the source file repositories:

Repository	Description
`s3com`	Main S3COM files
`io`	General input and Output
`oe`	Retrieval / Optimal estimation
`rttov`	Setup and run RTTOV
`config`	Parameters and structures
`utils`	Fortran, math and physics utilities

Chapter 2

Modules Index

2.1 Modules List

Here is a list of all documented modules with brief descriptions:

Chapter 3

Data Type Index

3.1 Data Types List

Here are the data types with brief descriptions:

Chapter 4

Module Documentation

4.1 mod_cld_legcoef Module Reference

Module dealing with computing and loading the coefficients of the Legendre polynomials.

Functions/Subroutines

- subroutine, public cld_legcoef_load (cld_mie)

  Export the coefficients of Legendre polynomials expansion of the phase function.
- subroutine cld_legcoef_compute (cld_mie)

  Compute the Legendre polynomials expansion coefficients from the phase function.
- subroutine cld_legcoef_read (cld_mie)

  Read the Legendre polynomials expansion coefficients from a netCDF file.
- subroutine cld_legcoef_write (cld_mie)

  Write the Legendre polynomials expansion coefficients to a netCDF file.
- character(len=:) function, allocatable fn_legcoef (fn, nmom)

  Finds the name of the netCDF file containing the Legendre coefficients.

4.1.1 Detailed Description

Module dealing with computing and loading the coefficients of the Legendre polynomials.

This module contains the following functions:

- `cld_legcoef_load` : Export the coefficients of Legendre polynomials expansion of the phase function
- `cld_legcoef_compute` : Compute the Legendre polynomials expansion coefficients from the phase function
- `cld_legcoef_read` : Read the Legendre polynomials expansion coefficients from a netCDF file
- `cld_legcoef_write` : Write the Legendre polynomials expansion coefficients in a netCDF file

4.1.2 Function/Subroutine Documentation

4.1.2.1 cld_legcoef_compute()

```
subroutine mod_cld_legcoef::cld_legcoef_compute (
            type(type_cld_mie), intent(inout) cld_mie )  [private]
```

Compute the Legendre polynomials expansion coefficients from the phase function.

The internal RTTOV function `rttov_legcoef_calc` is used to compute the coefficients from the phase function in `cld_mie`. The coefficients are then stored in `cld_mielegcoef`.

Parameters

`in,out`	*cld_mie*	Structure containing Mie optical properties

4.1.2.2 cld_legcoef_load()

```
subroutine, public mod_cld_legcoef::cld_legcoef_load (
            type(type_cld_mie), intent(inout) cld_mie )
```

Export the coefficients of Legendre polynomials expansion of the phase function.

This subroutine writes the coefficients of Legendre polynomials expansion of the phase function in the cld_mie structure. The coefficients are either computed from the phase function or read from a netCDF file. If that files does not exist, it is created. The new netcdf file has the same name as `fn_mie` but with the extension "_legcoef_↩nmom_.nc", with nmom the number of Legendre polynomials used.

Parameters

`in,out`	*cld_mie*	Structure containing Mie optical properties

4.1.2.3 cld_legcoef_read()

```
subroutine mod_cld_legcoef::cld_legcoef_read (
            type(type_cld_mie) cld_mie )  [private]
```

Read the Legendre polynomials expansion coefficients from a netCDF file.

Legendre expansion coefficients are read from the netCDF file `cld_miefn_legcoef` and stored in `cld_↩mielegcoef`.

Parameters

in,out	*cld_mie*	Structure containing Mie optical properties

4.1.2.4 cld_legcoef_write()

```
subroutine mod_cld_legcoef::cld_legcoef_write (
            type(type_cld_mie), intent(in) cld_mie )  [private]
```

Write the Legendre polynomials expansion coefficients to a netCDF file.

Legendre expansion coefficients created by `cld_legcoef_compute` are written to the netCDF file `cld_↩ miefn_legcoef`.

Parameters

in	*cld_mie*	Structure containing Mie optical properties

4.1.2.5 fn_legcoef()

```
character(len=:)  function, allocatable mod_cld_legcoef::fn_legcoef (
            character(len=*), intent(in) fn,
            integer, intent(in) nmom )  [private]
```

Finds the name of the netCDF file containing the Legendre coefficients.

Parameters

in	*fn*	Name of the netCDF file containing the Mie coefficients
in	*nmom*	Number of Legendre moments

Returns

Name of the netCDF file containing the Legendre coefficients

4.2 mod_cld_mie Module Reference

Allocate and load the user-defined Mie cloud properties.

Functions/Subroutines

- subroutine, public cld_mie_load (s3com, cld)

  General call to subroutines needed to load optical properties for liquid clouds from user-defined files.
- subroutine cld_mie_read (fn_mie, cld_mie)

  Initializes the cld_mie structure and loads the Mie scattering data from a netCDF file.

4.2.1 Detailed Description

Allocate and load the user-defined Mie cloud properties.

This module contains the subroutines needed to load the user-defined Mie cloud properties:

- `cld_mie_load`: general call to subroutines needed to load optical properties for liquid clouds from user-defined files
- `cld_mie_read`: initializes the cld_mie structure and loads the Mie scattering data from a netCDF file

4.2.2 Function/Subroutine Documentation

4.2.2.1 cld_mie_load()

```
subroutine, public mod_cld_mie::cld_mie_load (
            type(type_s3com), intent(in) s3com,
            type(type_cld), intent(out) cld )
```

General call to subroutines needed to load optical properties for liquid clouds from user-defined files.

This initializes and sets the `cldmie` structure, which contains user-defined liquid cloud optical properties needed by RTTOV.

These properties are read from netCDF files, currently expected to be located in $PATH/data/opt_prop. Current properties are generated from a Mie code. They are loaded for a given instrument. S3COM stops if the property files are not found.

These stored Mie optical properties later required by RTTOV are:

- the absorption cross-section (in um^2)
- the scattering cross-section (in um^2)
- the phase function for defined angles
- the legendre coefficients of the phase function

Note

Note that all cloud properties are defined for a normalized droplet size distribution n(r)! This means that the integral of n(r) is here 1, and converting the absorption and scattering cross-sections to the total absorption and scattering coefficients (typically in km^-1) requires multiplying by the droplet number concentration.

Parameters

`out`	*cld*	cld structure (currently only contains mie)
`in`	*s3com*	s3com structure

4.2.2.2 cld_mie_read()

```
subroutine mod_cld_mie::cld_mie_read (
             character(len = 128), intent(in) fn_mie,
             type(type_cld_mie), intent(out) cld_mie )  [private]
```

Initializes the cld_mie structure and loads the Mie scattering data from a netCDF file.

This reads Mie optical properties from a user-defined NetCDF file and stores them in the `cld_mie` structure. The following variables are set in `cld_mie`:

- nchan: number of wavelengths for the given instrument
- nrad: number of radius on which Mie properties are defined
- nang: number of angles on which the phase function is computed
- chan_id: ID of the instrument channel in RTTOV
- radius: effective radii (in um)
- angle: phase function angles (in degrees)
- Csca: the scattering cross-section coefficient (in um^2). Computed as Cext $*$ w0, with Cext the extinction cross-section and w0 the single-scattering albedo
- Cabs: the absorption cross-section coefficient (in um^2). Computed as Cext $*$ (1-w0).
- pha: the phase function

Parameters

`in`	*fn_mie*	name of the netCDF file containing the Mie scattering data
`out`	*cld_mie*	cld_mie structure

Chapter 5

Data Type Documentation

5.1 s3com_types::type_cld Type Reference

Structure containing all cloud optical properties.

Public Attributes

- type(type_cld_mie) **mie**
  Cloud optical properties from Mie calculations.

5.1.1 Detailed Description

Structure containing all cloud optical properties.

The documentation for this type was generated from the following file:

- src/conf/types.f90

5.2 s3com_types::type_cld_mie Type Reference

Structure containing cloud optical properties from Mie calculations.

Public Attributes

- integer(wpi) **nang**

  Number of scattering angles for which Mie calculations were performed.
- integer(wpi) **nchan**

  Number of instrumental channels.
- integer(wpi) **nrad**

  Number of available effective radii on which Mie calculations were performed.
- integer(wpi) **nmom**

  Number of coefficients for Legendre polynomial decomposition.
- character(len=128) **fn_mie**

  Name of the file containing the Mie optical properties for a given instrument.
- character(len=128) **fn_legcoef**

  Name of the file containing the corresponding Legendre polynomial coefficients.
- integer(wpi), dimension(:), allocatable **chan_id**

  ID of the instrument channel in RTTOV.
- integer(wpi), dimension(:), allocatable **mom**

  Moments number of the Legendre polynomial decomposition.
- real(kind=wp), dimension(:), allocatable **chan_wl**

  Wavelength of the instrument channel um
- real(kind=wp), dimension(:), allocatable **radius**

  Effective radius of the spherical cloud particles um
- real(kind=wp), dimension(:), allocatable **angle**

  Scattering angle degrees
- real(kind=wp), dimension(:,:), allocatable **cext**

  Extinction cross-section coefficient um^2
- real(kind=wp), dimension(:,:), allocatable **csca**

  Scattering cross-section coefficient um^2
- real(kind=wp), dimension(:,:), allocatable **cabs**

  Absorption cross-section coefficient um^2
- real(kind=wp), dimension(:,:), allocatable **w0**

  Single-scattering albedo.
- real(kind=wp), dimension(:,:,:), allocatable **pha**

  Azimuthally-averaged phase function.
- real(kind=wp), dimension(:,:,:), allocatable **legcoef**

  Coefficients of the Legendre polynomial decomposition of the phase function.

5.2.1 Detailed Description

Structure containing cloud optical properties from Mie calculations.

The documentation for this type was generated from the following file:

- src/conf/types.f90

5.3 s3com_types::type_icon Type Reference

Structure for model outputs from ICON simulations.

Public Attributes

- integer(wpi) **npoints**

  Total number of grid points.
- integer(wpi) **nlevels**

  Number of vertical levels.
- integer(wpi) **nlayers**

  Number of vertical layers (typically, nlevels - 1)
- integer(wpi) **nlat**

  Number of latitude points in the grid.
- integer(wpi) **nlon**

  Number of longitude points in the grid.
- integer(wpi) **mode**

  Model grid type (1: track, 2: lon-lat, 3: lat-lon)
- real(dp) **time**

  Time of the simulation (format is %Y%m%d.%f UTC) input
- integer(wpi), dimension(:), allocatable **height**

  Index of vertical layers input
- integer(wpi), dimension(:), allocatable **height_2**

  Index of vertical levels input
- real(wp), dimension(:), allocatable **lon**

  Longitude degrees East input
- real(wp), dimension(:), allocatable **lat**

  Latitude degrees North input
- real(wp), dimension(:), allocatable **lon_orig**

  Longitude that won't be regridded degrees East
- real(wp), dimension(:), allocatable **lat_orig**

  Latitude that won't be regridded degrees North
- real(wp), dimension(:), allocatable **topography**

  Surface height m input
- real(wp), dimension(:), allocatable **landmask**

  Land/sea mask (0/1) input
- real(wp), dimension(:), allocatable **ps**

  Surface pressure Pa input
- real(wp), dimension(:), allocatable **ts**

  Skin temperature K input
- real(wp), dimension(:), allocatable **t_2m**

  2-m temperature K input
- real(wp), dimension(:), allocatable **q_2m**

  2-m specific humidity kg/kg input
- real(wp), dimension(:), allocatable **u_10m**

  Zonal 10-m wind m/s input
- real(wp), dimension(:), allocatable **v_10m**

  Meridional 10-m wind m/s input
- real(wp), dimension(:,:), allocatable **p**

  Layer pressure Pa input
- real(wp), dimension(:,:), allocatable **p_ifc**

  Pressure at half-level center Pa input
- real(wp), dimension(:,:), allocatable **z**

  Layer height m input
- real(wp), dimension(:,:), allocatable **z_ifc**

Height at half-levels center m input

- real(wp), dimension(:,:), allocatable **t**

  Temperature K input
- real(wp), dimension(:,:), allocatable **t_ifc**

  Temperature at half-levels center K input
- real(wp), dimension(:,:), allocatable **q**

  Specific humidity kg/kg input
- real(wp), dimension(:,:), allocatable **q_ifc**

  Specific humidity at half level center kg/kg input
- real(wp), dimension(:,:), allocatable **clc**

  Total cloud fraction (0-1) input
- real(wp), dimension(:,:), allocatable **clw**

  Specific cloud water content kg/kg input
- real(wp), dimension(:,:), allocatable **cli**

  Specific cloud ice content kg/kg input
- real(wp), dimension(:,:), allocatable **qnc**

  Cloud droplet number concentration # m^-3 input
- real(wp), dimension(:,:), allocatable **qr**

  Rain mixing ratio kg/kg input
- real(wp), dimension(:,:), allocatable **qs**

  Snow mixing ratio kg/kg input
- real(wp), dimension(:,:), allocatable **dz**

  Layer thickness m
- real(wp), dimension(:,:), allocatable **rho**

  Air density used for liquid clouds kg m^-3
- real(wp), dimension(:,:), allocatable **tv**

  Virtual temperature K
- real(wp), dimension(:,:), allocatable **lwc**

  Liquid water content kg m^-3
- real(wp), dimension(:,:), allocatable **iwc**

  Ice water content kg m^-3
- real(wp), dimension(:,:), allocatable **cdnc**

  Cloud droplet number concentration # m^-3
- real(wp), dimension(:,:), allocatable **reff**

  Cloud liquid water effective radius m

5.3.1 Detailed Description

Structure for model outputs from ICON simulations.

They are either read form the NetCDF file (input flag) or calculated from these model output

The documentation for this type was generated from the following file:

- src/conf/types.f90

5.4 s3com_types::type_model Type Reference

General structure for all atmospheric data from model outputs.

Public Attributes

- integer(wpi) **npoints**

  Total number of grid points.
- integer(wpi) **nlevels**

  Number of vertical levels.
- integer(wpi) **nlayers**

  Number of vertical layers (typically, nlevels - 1)
- integer(wpi) **nlat**

  Number of latitude points in the grid.
- integer(wpi) **nlon**

  Number of longitude points in the grid.
- integer(wpi) **mode**

  Model grid type (1: track, 2: lon-lat, 3: lat-lon)
- integer(wpi) **idx_start**

  Starting point index for the subset grid.
- integer(wpi) **idx_end**

  Ending point index for the subset grid.
- integer(wpi), dimension(:), allocatable **height**

  Index of vertical layers.
- integer(wpi), dimension(:), allocatable **height_2**

  Index of vertical levels.
- integer(wpi), dimension(3) **time**

  Time of the day, UTC /hour, minute, second/
- integer(wpi), dimension(3) **date**

  Day of the year /day, month, year/
- integer(wpi), dimension(:), allocatable **point**

  Point index.
- real(wp), dimension(:), allocatable **lon**

  Longitude degrees East
- real(wp), dimension(:), allocatable **lat**

  Latitude degrees North
- real(wp), dimension(:), allocatable **lon_orig**

  Longitude that won't be regridded degrees East
- real(wp), dimension(:), allocatable **lat_orig**

  Latitude that won't be regridded degrees North
- real(wp), dimension(:), allocatable **topography**

  Surface height m
- real(wp), dimension(:), allocatable **landmask**

  Land/sea mask (0/1)
- real(wp), dimension(:), allocatable **ps**

  Surface pressure Pa
- real(wp), dimension(:), allocatable **ts**

  Skin temperature K
- real(wp), dimension(:), allocatable **t_2m**

  2-m temperature K
- real(wp), dimension(:), allocatable **q_2m**

  2-m specific humidity kg/kg
- real(wp), dimension(:), allocatable **u_10m**

  Zonal 10-m wind m/s
- real(wp), dimension(:), allocatable **v_10m**

Meridional 10-m wind m/s

- real(wp), dimension(:), allocatable **sunzenangle**

  Solar zenith angle degrees
- real(wp), dimension(:), allocatable **sunazangle**

  Solar azimuth angle degrees
- real(wp), dimension(:,:), allocatable **o3**

  Ozone concentrations on model levels for user-defined gas profiles, see namelist configuration gas_unit
- real(wp), dimension(:,:), allocatable **co2**

  CO2 concentrations, similarly to o3.
- real(wp), dimension(:,:), allocatable **ch4**

  CH4 concentrations, similarly to o3.
- real(wp), dimension(:,:), allocatable **n2o**

  N2O concentrations, similarly to o3.
- real(wp), dimension(:,:), allocatable **s2o**

  S2O concentrations, similarly to o3.
- real(wp), dimension(:,:), allocatable **co**

  CO concentrations, similarly to o3.
- real(wp), dimension(:,:), allocatable **p**

  Pressure on model levels Pa
- real(wp), dimension(:,:), allocatable **z**

  Altitude on model levels m
- real(wp), dimension(:,:), allocatable **dz**

  Geometrical thickness of model layer m
- real(wp), dimension(:,:), allocatable **t**

  Temperature on model levels K
- real(wp), dimension(:,:), allocatable **q**

  Specific humidity on model levels kg/kg
- real(wp), dimension(:,:), allocatable **clc**

  Total cloud fraction in model layer (0-1)
- real(wp), dimension(:,:), allocatable **lwc**

  Liquid water content in model layer kg/m3
- real(wp), dimension(:,:), allocatable **iwc**

  Ice water content in model layer kg/m3
- real(wp), dimension(:,:), allocatable **cdnc**

  Cloud droplet number concentration in model layer # m^ -3
- real(wp), dimension(:,:), allocatable **reff**

  Liquid water cloud effective radius in model layer m

5.4.1 Detailed Description

General structure for all atmospheric data from model outputs.

It is used to store model outputs consistently and completed with other variables. This is the structure that is used in the main program and passed for radiation calculations.

The documentation for this type was generated from the following file:

- src/conf/types.f90

5.5 s3com_types::type_nml Type Reference

Structure containing namelist variables.

Public Attributes

- character(len=256) **path_s3com**

  Path to S3COM directory.
- character(len=256) **path_rttov**

  Path to RTTOV directory.
- character(len=256) **fname_in**

  Name of the input model file (e.g. ICON or NWPSAF)
- character(len=256) **path_out**

  Path to the repository containing where the output files will be written.
- character(len=256) **suffix_out**

  Suffix added to the output filenames.
- character(len=256) **model_name**

  Name of the physical model. Currently supported: ICON, NWPSAF.
- integer(wpi) **npoints_it**

  Number of subset data points (chunks) to process in each iteration (only relevant to optimize memory usage)
- integer(wpi) **platform**

  Platform ID for RTTOV.
- integer(wpi) **satellite**

  Satellite ID for RTTOV.
- integer(wpi) **instrument**

  Instrument ID for RTTOV.
- integer(wpi) **nchannels**

  Number of instrument channels to simulate.
- integer(wpi) **ir_scatt_model**

  Scattering model for IR source term: 1=DOM; 2=Chou-scaling.
- integer(wpi) **vis_scatt_model**

  Scattering model for solar source term: 1=DOM; 2=single-scattering; 3=MFASIS.
- integer(wpi) **dom_nstreams**

  Number of streams for DOM scattering model.
- integer(wpi) **dom_nmoments**

  Number of moments for discrete ordinate method.
- integer(wpi) **rttov_nthreads**

  Number of threads for RTTOV calculations (if openMP is enabled)
- integer(wpi) **gas_unit**

  Gas units for atmospheric profiles (1: kg/kg; 2: ppmv over moist air; 0: ppmv over dry air)
- integer(wpi) **ice_scheme**

  Scheme used for ice cloud optical properties: 1=Baum; 2=Baran 2014; 3=Baran 2018. Not relevant if `user_cld_↩ opt_param` is true.
- integer(wpi) **clw_scheme**

  Scheme used for liquid cloud optical properties: 1=OPAC; 2=Deff. Not relevant if `user_cld_opt_param` *is true.*
- integer(wpi), dimension(:), allocatable **channel_list**

  List of satellite channels to simulate (should be of dimension nchannels)
- logical **user_cld_opt_param**

  If true, users can supply their own cloud optical properties (`ice_scheme` and `clw_scheme` are then not used)
- logical **flag_retrievals**

Flag to specify if retrievals should be performed.

- logical **flag_output_atm**

  Flag to specify if atmospheric profiles should be output.
- logical **flag_output_jac**

  Flag to specify if Jacobian matrices should be output.
- logical **flag_output_k_tl**

  Flag to specify if K matrices should be output.
- logical **do_jacobian_calc**

  Flag to specify if Jacobian matrices should be calculated.
- logical **do_k_tl_calc**

  Flag to specify if K matrices should be calculated via TL.
- logical **do_opdep_calc**

  If false, disables the RTTOV gas optical depth calculation (default = true)
- logical **dom_rayleigh**

  If true, Rayleigh scattering is included in the DOM model.
- logical **mmr_cldaer**

  Cloud and aerosol units: true => kg/kg (cld+aer); false => g/m3 (cld), cm-3 (aer)
- logical **ozone_data**

  Set to true when supplying a profile of ozone, false to use climatology from RTTOV.
- logical **add_refrac**

  If true accounts for atmospheric refraction.
- logical **add_clouds**

  If true, clouds are included in the RTTOV model.
- logical **add_aerosols**

  If true, aerosols are included in the RTTOV model.

5.5.1 Detailed Description

Structure containing namelist variables.

These variables are directly read from the namelist file that is provided as argument to the S3COM executable

The documentation for this type was generated from the following file:

- src/conf/types.f90

5.6 s3com_types::type_nwpsaf Type Reference

Structure for model outputs from NWPSAF simulations.

Public Attributes

- integer(wpi) **npoints**

  Total number of grid points.
- integer(wpi) **nlevels**

  Number of vertical levels.
- integer(wpi) **nlayers**

  Number of vertical layers (typically, nlevels - 1)
- integer(wpi) **nlat**

  Number of latitude points in the grid.
- integer(wpi) **nlon**

  Number of longitude points in the grid.
- integer(wpi) **mode**

  Model grid type (1: track, 2: lon-lat, 3: lat-lon)
- integer(wpi), dimension(:), allocatable **height**

  Index of vertical layers.
- integer(wpi), dimension(:), allocatable **height_2**

  Index of vertical levels.
- integer(wpi), dimension(:), allocatable **point**

  Index of grid points input
- integer(wpi), dimension(:), allocatable **day**

  Day of the simulation input
- integer(wpi), dimension(:), allocatable **month**

  Month of the simulation input
- integer(wpi), dimension(:), allocatable **year**

  Year of the simulation input
- real(wp), dimension(:), allocatable **lon**

  Longitude degrees East
- real(wp), dimension(:), allocatable **lat**

  Latitude degrees North
- real(wp), dimension(:), allocatable **lon_orig**

  Longitude that won't be regridded degrees East
- real(wp), dimension(:), allocatable **lat_orig**

  Latitude that won't be regridded degrees North
- real(wp), dimension(:), allocatable **elevation**

  Surface height m input
- real(wp), dimension(:), allocatable **lsm**

  Land/sea mask (0/1) input
- real(wp), dimension(:), allocatable **psurf**

  Surface pressure Pa input
- real(wp), dimension(:), allocatable **tsurf**

  Skin temperature K input
- real(wp), dimension(:), allocatable **t2m**

  2-m temperature K input
- real(wp), dimension(:), allocatable **q2m**

  2-m specific humidity kg/kg
- real(wp), dimension(:), allocatable **u10**

  Zonal 10-m wind m/s input
- real(wp), dimension(:), allocatable **v10**

  Meridional 10-m wind m/s input
- real(wp), dimension(:,:), allocatable **pap**

Layer pressure Pa input

- real(wp), dimension(:,:), allocatable **paph**
  Pressure at half-level center Pa input
- real(wp), dimension(:,:), allocatable **altitude**
  Layer height m input
- real(wp), dimension(:,:), allocatable **altitudeh**
  Height at half-levels center m input
- real(wp), dimension(:,:), allocatable **temp**
  Air temperature K input
- real(wp), dimension(:,:), allocatable **temph**
  Air temperature at half-levels center K input
- real(wp), dimension(:,:), allocatable **hum**
  Specific humidity kg/kg input
- real(wp), dimension(:,:), allocatable **humh**
  Specific humidity at half level center kg/kg input
- real(wp), dimension(:,:), allocatable **cc**
  Total cloud fraction (0-1) input
- real(wp), dimension(:,:), allocatable **dz**
  Geometrical thickness of layer m
- real(wp), dimension(:,:), allocatable **rho**
  Air density used for liquid clouds kg m^ -3
- real(wp), dimension(:,:), allocatable **tv**
  Virtual temperature K
- real(wp), dimension(:,:), allocatable **lwc**
  Liquid water content kg m^ -3 input
- real(wp), dimension(:,:), allocatable **iwc**
  Ice water content kg m^ -3 input
- real(wp), dimension(:,:), allocatable **cdnc**
  Cloud droplet number concentration # m^ -3
- real(wp), dimension(:,:), allocatable **reff**
  Cloud liquid water effective radius m

5.6.1 Detailed Description

Structure for model outputs from NWPSAF simulations.

They are either read form the NetCDF file (input flag) or calculated from these model output

Note

This time is missing for NWP-SAF simulations, later set to 12:00:00 UTC for all data points

The documentation for this type was generated from the following file:

- src/conf/types.f90

5.7 s3com_types::type_rttov_opt Type Reference

Structure containing main RTTOV options.

Public Attributes

- integer(wpi) **dosolar**

  Activation of solar radiation calculations (0: false, 1: true)
- integer(wpi) **nchannels**

  Number of instrument channels to simulate input
- integer(wpi) **nthreads**

  Number of threads for RTTOV calculations (if openMP is available) input
- integer(wpi) **platform**

  Platform ID for RTTOV input
- integer(wpi) **satellite**

  Satellite ID for RTTOV input
- integer(wpi) **instrument**

  Instrument ID for RTTOV input
- integer(wpi) **month**

  Month of the year, used to load surface brdf and emissivity data.
- integer(wpi) **gas_units**

  Gas units for atmospheric profiles (1: kg/kg; 2: ppmv over moist air; 0: ppmv over dry air)
- integer(wpi) **ice_scheme**

  Scheme used for ice cloud optical properties: 1=Baum; 2=Baran 2014; 3=Baran 2018. Not relevant if `user_cld_↩ opt_param` is true. input
- integer(wpi) **clw_scheme**

  Scheme used for liquid cloud optical properties: 1=OPAC; 2=Deff input
- integer(wpi) **ir_scatt_model**

  Scattering model for IR source term: 1=DOM; 2=Chou-scaling input
- integer(wpi) **vis_scatt_model**

  Scattering model for solar source term: 1=DOM; 2=single-scattering; 3=MFASIS input
- integer(wpi) **dom_nstreams**

  Number of streams for discrete ordinate method input
- integer(wpi) **dom_nmoments**

  Number of moments for discrete ordinate method input
- logical **mmr_cldaer**

  Cloud and aerosol units: true => kg/kg (cld+aer); false => g/m3 (cld), cm-3 (aer) input
- logical **ozone_data**

  True when supplying a profile of ozone, false to use RTTOV climatologies input
- logical **co2_data**

  True when supplying a profile of CO2, false to use RTTOV climatologies.
- logical **n2o_data**

  True when supplying a profile of N2O, false to use RTTOV climatologies.
- logical **ch4_data**

  True when supplying a profile of CH4, false to use RTTOV climatologies.
- logical **co_data**

  True when supplying a profile of CO, false to use RTTOV climatologies.
- logical **so2_data**

  True when supplying a profile of SO2, false to use RTTOV climatologies.
- logical **add_clouds**

  True to add clouds to the RTTOV calculations input
- logical **add_aerosols**

  True to add aerosols to the RTTOV calculations input
- logical **add_refrac**

  True to add atmospheric refraction input

- logical **do_opdep_calc**

  If false, disables the RTTOV gas optical depth calculation input
- logical **dom_rayleigh**

  If false, disables the RTTOV Rayleigh scattering calculation input
- logical **user_cld_opt_param**

  If true, users can supply their own cloud optical properties (`ice_scheme` and `clw_scheme` are then not used) input
- integer, dimension(:), allocatable **channel_list**

  List of channels to simulate input
- character(len=32) **platform_name**

  Platform name.
- character(len=32) **inst_name**

  Instrument name.
- character(len=32) **sat_name**

  Satellite name.
- real(wp) **zenangle**

  Satellite zenith angle degrees
- real(wp) **azangle**

  Satellite azimuth angle degrees

5.7.1 Detailed Description

Structure containing main RTTOV options.

Data from namelist are indicated with input. Others are set in `rttov_setup_opt`

The documentation for this type was generated from the following file:

- src/conf/types.f90

5.8 s3com_types::type_s3com Type Reference

Overall S3COM structure.

Public Attributes

- integer(kind=4), dimension(3) **time**

  Time of the day, UTC /hour, minute, second/
- integer(kind=4), dimension(3) **date**

  Day of the year /day, month, year/
- integer(kind=4) **npoints**

  Total number of grid points.
- integer(kind=4) **nlevels**

  Number of vertical levels.
- integer(kind=4) **nlayers**

  Number of vertical layers (typically, nlevels - 1)
- integer(kind=4) **nlat**

Number of latitude points in the grid.

- integer(kind=4) **nlon**

  Number of longitude points in the grid.
- integer(kind=4) **mode**

  Model grid type (1: track, 2: lon-lat, 3: lat-lon)
- integer(kind=4) **nmeas**

  Size of the measurement vector.
- integer(kind=4) **nstates**

  Size of the state vector.
- integer(kind=4) **idx_start**

  Starting point index for the subset grid.
- integer(kind=4) **idx_end**

  Ending point index for the subset grid.
- logical, dimension(:), allocatable **flag_rttov**

  Flag indicating if RTTOV should be called for a given point.
- type(type_nml) **nml**

  Contains all the S3COM namelist options.
- type(type_s3com_rad) **rad**

  Contains all the S3COM radiative transfer output variables.
- type(type_s3com_atm) **atm**

  Contains all the S3COM atmospheric profiles.
- type(type_s3com_jac) **jac**

  Contains all the S3COM Jacobian output variables.
- type(type_s3com_k_tl) **k_tl**

  Contains all the S3COM tangent linear output variables.
- type(type_s3com_opt) **opt**

  Contains all the S3COM radiative transfer options.

5.8.1 Detailed Description

Overall S3COM structure.

This structure contains all the variables used by S3COM for forward model simulations and retrievals It also stores all relevant output variables

The documentation for this type was generated from the following file:

- src/conf/types.f90

5.9 s3com_types::type_s3com_atm Type Reference

S3COM structure for atmospheric profiles.

Public Attributes

- real(kind=wp), dimension(:,:), allocatable **t**

  Temperature on levels K
- real(kind=wp), dimension(:,:), allocatable **z**

  Altitude on levels m
- real(kind=wp), dimension(:,:), allocatable **clc**

  Cloud fraction in layers -
- real(kind=wp), dimension(:,:), allocatable **cdnc**

  Cloud droplet number concentration in layers # m-3
- real(kind=wp), dimension(:,:), allocatable **reff**

  Cloud droplet effective radius in layers um
- real(kind=wp), dimension(:,:), allocatable **lwc**

  Cloud liquid water content in layers kg m-2

5.9.1 Detailed Description

S3COM structure for atmospheric profiles.

The atmospheric profiles are stored exactly as they have been used for forward model calculations When retrievals are activated, these can deviate from model outputs as the atmospheric model can be modified. When retrievals are not activated, these are identical to model outputs.

The documentation for this type was generated from the following file:

- src/conf/types.f90

5.10 s3com_types::type_s3com_jac Type Reference

S3COM structure for Jacobian calculations.

Public Attributes

- real(kind=wp), dimension(:), allocatable **wavelength**

  Wavelength um
- real(kind=wp), dimension(:,:,:), allocatable **p**

  Jacobian of the forward model with respect to the pressure W m-2 sr-1 um-1 Pa-1
- real(kind=wp), dimension(:,:,:), allocatable **t**

  Jacobian of the forward model with respect to the temperature W m-2 sr-1 um-1 K-1
- real(kind=wp), dimension(:,:,:), allocatable **cfrac**

  Jacobian of the forward model with respect to the cloud fraction W m-2 sr-1 um-1
- real(kind=wp), dimension(:,:,:), allocatable **clwde**

  Jacobian of the forward model with respect to the cloud droplet effective diameter W m-2 sr-1 um-1 um-1
- logical **do_jacobian_calc**

  Flag to specify if Jacobian matrices should be calculated.

5.10.1 Detailed Description

S3COM structure for Jacobian calculations.

Note

RTTOV outputs the gradient of each forward model radiance with respect to each input profile variable evaluated for a given input profile. The input perturbation is here set to a unit radiance (BT are also possible).

The documentation for this type was generated from the following file:

- src/conf/types.f90

5.11 s3com_types::type_s3com_k_tl Type Reference

S3COM structure for Tangent Linear calculations.

Public Attributes

- real(kind=wp), dimension(:), allocatable **wavelength**

  Wavelength um
- real(kind=wp), dimension(:,:,:), allocatable **t**

  Jacobian of the forward model with respect to the pressure W m-2 sr-1 um-1 Pa-1
- logical **do_k_tl_calc**

  Flag to specify if K matrices should be calculated via TL.

5.11.1 Detailed Description

S3COM structure for Tangent Linear calculations.

This structure contains Jacobians that are computed using the TL model of RTTOV.

Warning

This structure is not yet fully tested for all configurations. Use with caution.

The documentation for this type was generated from the following file:

- src/conf/types.f90

5.12 s3com_types::type_s3com_opt Type Reference

Structure containing all S3COM options.

Public Attributes

- type(type_rttov_opt) **rttov**
  RTTOV options.

5.12.1 Detailed Description

Structure containing all S3COM options.

This stores the radiative transfer options relevant to the S3COM code

The documentation for this type was generated from the following file:

- src/conf/types.f90

5.13 s3com_types::type_s3com_rad Type Reference

S3COM structure for direct radiative transfer simulations and measurements.

Public Attributes

- real(kind=wp), dimension(:), allocatable **wavelength**
  Wavelength um
- real(kind=wp), dimension(:,:), allocatable **y**
  Satellite measurements. Units will depend on type.
- real(kind=wp), dimension(:,:), allocatable **f**
  Forward model simulations with same type and units as y.
- real(kind=wp), dimension(:,:), allocatable **f_ref_total**
  Total top-of-atmosphere outgoing reflectance -
- real(kind=wp), dimension(:,:), allocatable **f_ref_clear**
  Clear-sky top-of-atmosphere outgoing reflectance -
- real(kind=wp), dimension(:,:), allocatable **f_bt_total**
  Total top-of-atmosphere brightness temperature K
- real(kind=wp), dimension(:,:), allocatable **f_bt_clear**
  Clear-sky top-of-atmosphere brightness temperature K
- real(kind=wp), dimension(:,:), allocatable **f_rad_total**
  Total top-of-atmosphere radiance W m-2 sr-1 um-1
- real(kind=wp), dimension(:,:), allocatable **f_rad_clear**
  Total top-of-atmosphere radiance W m-2 sr-1 um-1

5.13.1 Detailed Description

S3COM structure for direct radiative transfer simulations and measurements.

The documentation for this type was generated from the following file:

- src/conf/types.f90

Index