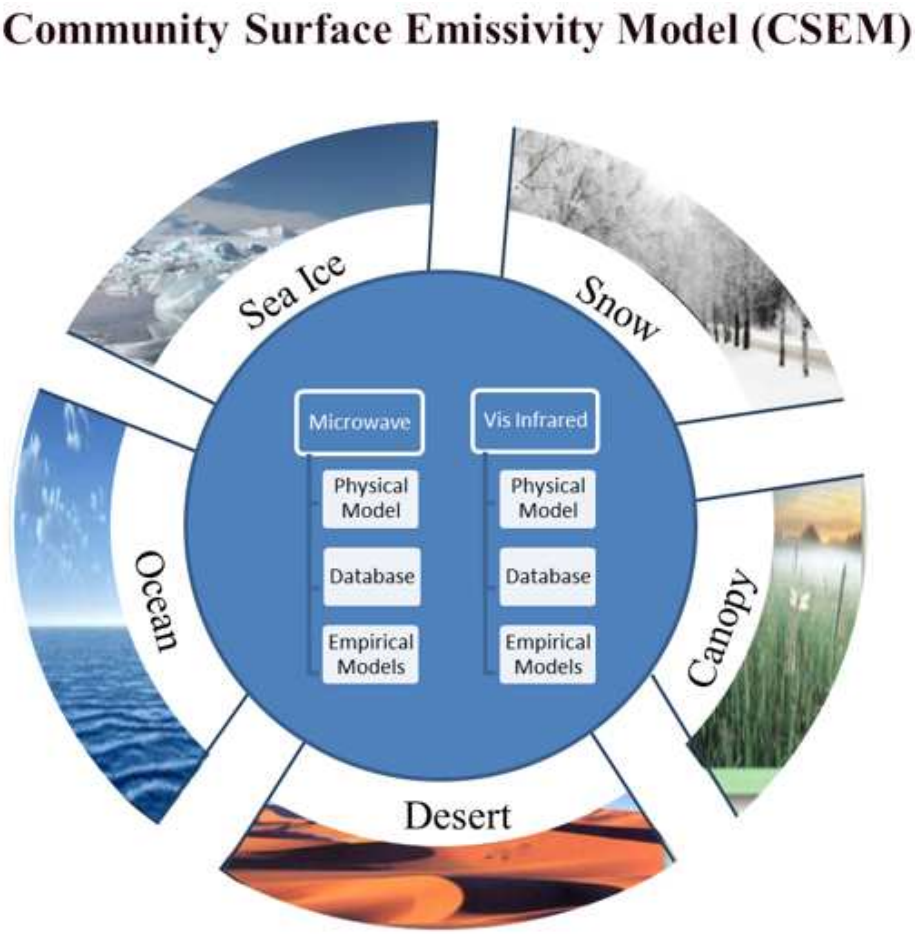
CSEM

REL 1.0.0

Generated by Doxygen 1.9.1

Chapter 1

CSEM Documentation



The Community Surface Emissivity Model (CSEM) is a highly modularized Earth's surface RT modeling system based on Object-Oriented Programming (OOP) design. It evolved from the surface modules of the Community Radiative Transfer Model (CRTM), but with completely redesigned model structure to facilitate the implementation of various surface RT models. CSEM provides the surface emissivity and reflectivity simulations of diverse surface types in the spectral range from the ultraviolet, visible to microwave bands.

Enclosed in CSEM are not only the physical models based on sound radiative transfer equations, but also a variety of empirical and semi-empirical models, type-based emissivity lookup tables (LUT), and global emissivity atlases from satellite retrievals. The object-oriented design provides very flexible software interfaces for implementing and

testing new model components. Multiple model options of the same kind (e.g., microwave land models) may be easily implemented and accommodated in the CSEM framework.

In practical applications, CSEM may be used as a standalone surface RT research tool, or used as a sub-system to provide the surface radiative conditions for CRTM, significantly leveraging the development and improvement efforts of CRTM.

Version

1.0.0

Author

STAR/NESDIS CRTM Team

Date

August 2022

Chapter 2

Model Setup

Version 1.0.0 is the current version.

Downloading:

CSEM is available from two GitHub sites:
NOAA-STAR domain: https://github.com/NOAA-STAR/CSEM1.0.0 JCSDA: https://github.com/JCSDA-i

The JCSDA CSEM repository is forked from the NOAA-STAR repository.

git clone https://github.com/NOAA-STAR/CSEM1.0.0, or

git clone https://github.com/JCSDA-internal/CSEM1.0.0

Installing:

1) Go to Build/env.setup and source the specific compiler configuration file.

e.g., source gfortran.setup

note: make sure the following three environment variables are already defined or included in the setup file.

B-shell (sh, bash)

export NETCDF_HOME=path to the netcdf

export HDF5_HOME=path to the hdf5

C-shell (csh)

setenv NETCDF_HOME path to the netcdf

setenv HDF5_HOME path to the hdf5

This step is needed for the first-time fresh installation.

2) Generate the file "configure" ./autogen.sh

This step is needed as long as the three ENV variables have been changed.

3) Generate the file "Makefile", you may specify where the CSEM library will be installed. The default is the current directory ./configure –prefix=path for the CSEM library to be installed 4) make

5) make install

See also

CSEM site https://github.com/NOAA-STAR/CSEM1.0.0/

Chapter 3

CRTM-CSEM Integration

The integration of CSEM with CRTM is only needed to perfrom one time. CSEM will replace the exisitng CRTM surface modules in the integrated CRTM-CSEM package, providing the exisitng default and the expanded surface functionality for the upper-tier RT solvers.

A shell script has been created to automate the integration. It is available at interfacing/CRTM/Setup_CSEM_↩ Library.sh. This script includes all the necessary steps to merge the CSEM codes into the general CRTM framework, and to build the integrated package. In short:

1) replace CSEM-related the source files and configuration files 2) set_CRTM_Environment.sh 3) export CRTM↩ _SOURCE_ROOT= 4) export PATH=∼/bin:$PATH 5) cd "src" and type make 6) cd "Build" 7) source ifort.setup 8) autogen.sh 9) CSEM_HOME=PATH-of-CSEM-library ./configure –prefix=path-to-install-CRTM-library 10) make 11) make install

The CSEM library needs to be built first, will be used as the external library for CRTM.

Chapter 4

Modules Index

4.1 Modules List

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

Chapter 5

Data Type Index

5.1 Data Types List

Here are the data types with brief descriptions:

Chapter 6

File Index

6.1 File List

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

Chapter 7

Module Documentation

7.1 azimuth_emissivity_f6_module Module Reference

Azimuthal functions of the FASTEM-6 model

Functions/Subroutines

- subroutine, public **azimuth_emissivity_f6** (AZCoeff, Wind_Speed, Azimuth_Angle, Frequency, Zenith_↩ Angle, e_Azimuth, iVar)
- subroutine, public azimuth_emissivity_f6_tl (AZCoeff, Wind_Speed_TL, Azimuth_Angle_TL, e_Azimuth_TL, iVar)
- subroutine, public azimuth_emissivity_f6_ad (AZCoeff, e_Azimuth_AD, Wind_Speed_AD, Azimuth_Angle_↩ AD, iVar)

7.1.1 Detailed Description

Azimuthal functions of the FASTEM-6 model

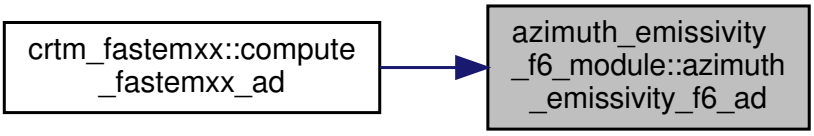
Helper module containing the azimuth-dependency routines for the CRTM FASTEM-6

7.1.2 Function/Subroutine Documentation

7.1.2.1 azimuth_emissivity_f6_ad()

```
subroutine, public azimuth_emissivity_f6_module::azimuth_emissivity_f6_ad (
             type(csem_fitcoeff_3d_type), intent(in) AZCoeff,
             real(fp), dimension(:), intent(inout) e_Azimuth_AD,
             real(fp), intent(inout) Wind_Speed_AD,
             real(fp), intent(inout) Azimuth_Angle_AD,
             type(ivar_type), intent(in) iVar )
```

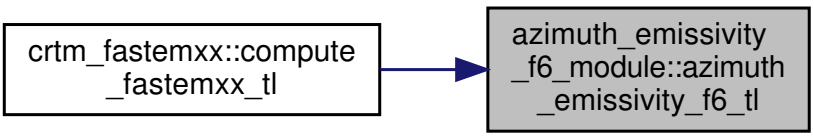
Adjoint model Here is the caller graph for this function:



7.1.2.2 azimuth_emissivity_f6_tl()

```
subroutine, public azimuth_emissivity_f6_module::azimuth_emissivity_f6_tl (
             type(csem_fitcoeff_3d_type), intent(in) AZCoeff,
             real(fp), intent(in) Wind_Speed_TL,
             real(fp), intent(in) Azimuth_Angle_TL,
             real(fp), dimension(:), intent(out) e_Azimuth_TL,
             type(ivar_type), intent(in) iVar )
```

Tangent-linear model Here is the caller graph for this function:



7.2 azimuth_emissivity_module Module Reference

Azimuthal emissivity subroutines of old FASTEM versons.

Functions/Subroutines

- subroutine, public azimuth_emissivity (AZCoeff, Wind_Speed, Azimuth_Angle, Frequency, cos_z, e_Azimuth, iVar)
- subroutine, public azimuth_emissivity_tl (AZCoeff, Wind_Speed_TL, Azimuth_Angle_TL, e_Azimuth_TL, i↩ Var)
- subroutine, public azimuth_emissivity_ad (AZCoeff, e_Azimuth_AD, Wind_Speed_AD, Azimuth_Angle_AD, iVar)

7.2.1 Detailed Description

Azimuthal emissivity subroutines of old FASTEM versons.

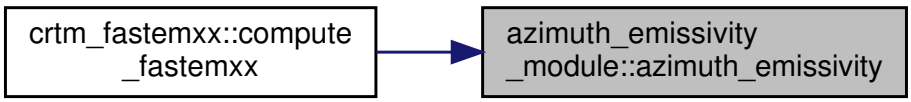
Helper module containing the azimuth emissivity routines for the CRTM implementation of FASTEM4 and FASTEM5

7.2.2 Function/Subroutine Documentation

7.2.2.1 azimuth_emissivity()

```
subroutine, public azimuth_emissivity_module::azimuth_emissivity (
             type(csem_fitcoeff_3d_type), intent(in) AZCoeff,
             real(fp), intent(in) Wind_Speed,
             real(fp), intent(in) Azimuth_Angle,
             real(fp), intent(in) Frequency,
             real(fp), intent(in) cos_z,
             real(fp), dimension(:), intent(out) e_Azimuth,
             type(ivar_type), intent(inout) iVar )
```

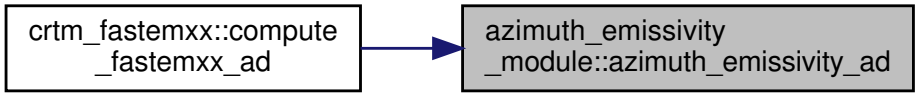
Compute emissivity as a function of relative azimuth angle. Here is the caller graph for this function:



7.2.2.2 azimuth_emissivity_ad()

```
subroutine, public azimuth_emissivity_module::azimuth_emissivity_ad (
             type(csem_fitcoeff_3d_type), intent(in) AZCoeff,
             real(fp), dimension(:), intent(inout) e_Azimuth_AD,
             real(fp), intent(inout) Wind_Speed_AD,
             real(fp), intent(inout) Azimuth_Angle_AD,
             type(ivar_type), intent(in) iVar )
```

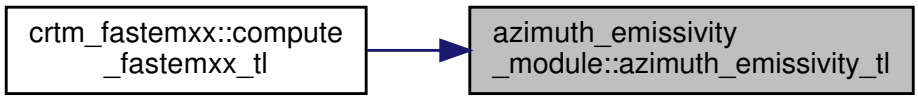
Adjoint model Here is the caller graph for this function:



7.2.2.3 azimuth_emissivity_tl()

```
subroutine, public azimuth_emissivity_module::azimuth_emissivity_tl (
             type(csem_fitcoeff_3d_type), intent(in) AZCoeff,
             real(fp), intent(in) Wind_Speed_TL,
             real(fp), intent(in) Azimuth_Angle_TL,
             real(fp), dimension(:), intent(out) e_Azimuth_TL,
             type(ivar_type), intent(in) iVar )
```

Tangent-linear model Here is the caller graph for this function:



7.3 cnrm_amsua_reader Module Reference

Module containing Data and routines for MW emissivity atlas METEO-FRANCE CNRM

Functions/Subroutines

- integer function, public **cnrm_amsua_setup** (path, imonth)
- integer function, public **cnrm_amsua_emiss** (latitude, longitude_in, frequency, zenangle, emissivity_↩ v, emissivity_h, pbats_veg)
- integer function, public **cnrm_amsua_emiss_multi** (latitude, longitude_in, frequency, zenangle, n_Channel, emissivity, pbats_veg)

Variables

- integer, public **cnrm_amsua_version** =200

7.3.1 Detailed Description

Module containing Data and routines for MW emissivity atlas METEO-FRANCE CNRM

7.4 cnrm_atlas_module Module Reference

Module for users to use CNRM land surface emissivity data sets by CSEM interfaces.

Functions/Subroutines

- integer function, public **cnrm_atlas_setup** (imonth, path, Atlas_ID, mw_atlas_ver)
- integer function, public **cnrm_atlas_emiss** (Frequency, Angle, Latitude, Longitude, imonth, Emissivity_H, Emissivity_V, stype)
- integer function, public **cnrm_atlas_emiss_nchannels** (Frequency, Angle, Latitude, Longitude, imonth, n↩ _Channel, emissivity, stype)
- logical function, public **cnrm_atlas_initialized** (imonth)
- subroutine, public **cnrm_atlas_close** ()

7.4.1 Detailed Description

Module for users to use CNRM land surface emissivity data sets by CSEM interfaces.

CNRM data includes the monthly land surface emissivity atlas retrieved from AMSU-A, AMSU-B, SSMI, SSMIS, TMI and AMSRE (http://www.cnrm.meteo.fr/gmap/mwemis/get_data.html). Only the interfaces for the monthly AMSU-A atlas are implemented in this module. Similar interfces may be implemented for the atlas retrieved from other sensors.

7.5 crtm_fastem1 Module Reference

Module with the old Fastem procedures.

Functions/Subroutines

- subroutine, public **fastem1** (Frequency, Sat_Zenith_Angle, SST, Wind_Speed, Emissivity, dEH_dWind↩ Speed, dEV_dWindSpeed)

7.5.1 Detailed Description

Module with the old Fastem procedures.

PURPOSE: This module computes ocean emissivity and its jacobian over water. The code is adopted from RTTOV Fastem version 1.

Method: FASTEM-1 English and Hewison 1998. http://www.metoffice.com/research/interproj/nwpsaf/rtm

7.6 crtm_fastem_module Module Reference

Container module with all the existing CRTM FASTEM versions.

Functions/Subroutines

- integer function, public crtm_fastem_emiss (Frequency, Angle, Water_Temperature, Salinity, Wind_Speed, Wind_Direction, Emissivity, Reflectivity, FASTEM_Version, Sensor_Azimuth_Angle, Transmittance)
- integer function, public compute_fastem_sfcoptics (Frequency, Angles, Water_Temperature, Salinity, Wind↩ _Speed, Wind_Direction, iVar, Emissivity, Reflectivity, FASTEM_Version, Sensor_Azimuth_Angle, Transmittance)
- integer function, public compute_fastem_sfcoptics_tl (Water_Temperature_TL, Salinity_TL, Wind_Speed_TL, Wind_Direction_TL, Transmittance_TL, iVar, Emissivity_TL, Reflectivity_TL, FASTEM_Version)
- integer function, public compute_fastem_sfcoptics_ad (Emissivity_AD, Reflectivity_AD, Water_↩ Temperature_AD, Salinity_AD, Wind_Speed_AD, Wind_Direction_AD, Transmittance_AD, iVar, FASTEM_↩ Version)
- integer function, public crtm_fastem_init (MWwaterCoeff_File, Version)
- integer function, public **crtm_fastem_destroy** ()

Variables

- logical, save, public **csem_mwwatercoeff_init** = .FALSE.

7.6.1 Detailed Description

Container module with all the existing CRTM FASTEM versions.

This module is provided to "wrap" all the existing CRTM FASTEM versions and provide a general interface to simplify integration into the main CRTM_SfcOptics module.

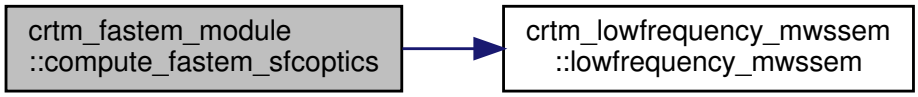
7.6.2 Function/Subroutine Documentation

7.6.2.1 compute_fastem_sfcoptics()

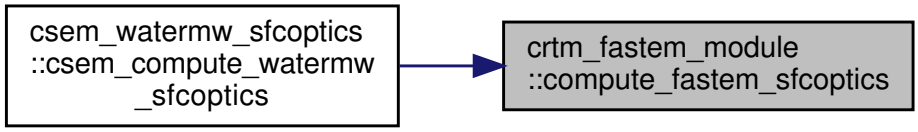
```
integer function, public crtm_fastem_module::compute_fastem_sfcoptics (
             real(fp), intent(in) Frequency,
             real(fp), dimension(:), intent(in) Angles,
             real(fp), intent(in) Water_Temperature,
             real(fp), intent(in) Salinity,
             real(fp), intent(in) Wind_Speed,
             real(fp), intent(in) Wind_Direction,
             type(ivar_type), intent(out) iVar,
             real(fp), dimension(:,:), intent(out) Emissivity,
             real(fp), dimension(:,:), intent(out) Reflectivity,
             integer, intent(in) FASTEM_Version,
             real(fp), intent(in) Sensor_Azimuth_Angle,
             real(fp), intent(in) Transmittance )
```

PURPOSE: Function to compute the surface emissivity and reflectivity at microwave frequencies over a water surface and at SINGLE frequency channel and MULTIPLE receiving angle

This function is a wrapper of different FASTEM versions Here is the call graph for this function:



Here is the caller graph for this function:

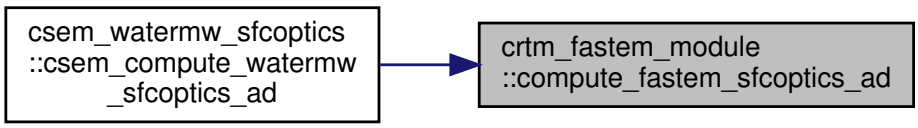


7.6.2.2 compute_fastem_sfcoptics_ad()

```
integer function, public crtm_fastem_module::compute_fastem_sfcoptics_ad (
             real(fp), dimension(:,:), intent(inout) Emissivity_AD,
             real(fp), dimension(:,:), intent(inout) Reflectivity_AD,
             real(fp), intent(out) Water_Temperature_AD,
             real(fp), intent(out) Salinity_AD,
             real(fp), intent(out) Wind_Speed_AD,
             real(fp), intent(out) Wind_Direction_AD,
             real(fp), intent(inout) Transmittance_AD,
             type(ivar_type), intent(in) iVar,
             integer, intent(in) FASTEM_Version )
```

PURPOSE: Function to compute the adjoint surface emissivity and reflectivity at microwave frequencies over a water surface.

This function is a wrapper of different FASTEM versions Here is the caller graph for this function:



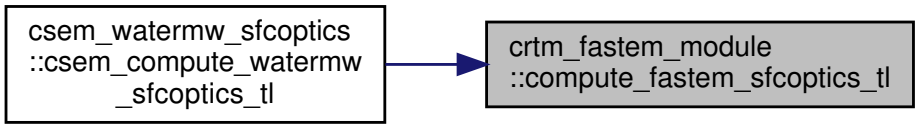
7.6.2.3 compute_fastem_sfcoptics_tl()

```
integer function, public crtm_fastem_module::compute_fastem_sfcoptics_tl (
             real(fp), intent(in) Water_Temperature_TL,
             real(fp), intent(in) Salinity_TL,
             real(fp), intent(in) Wind_Speed_TL,
             real(fp), intent(in) Wind_Direction_TL,
```

```
            real(fp), intent(in) Transmittance_TL,
            type(ivar_type), intent(in) iVar,
            real(fp), dimension(:,:), intent(out) Emissivity_TL,
            real(fp), dimension(:,:), intent(out) Reflectivity_TL,
            integer, intent(in) FASTEM_Version )
```

PURPOSE: Function to compute the tangent-linear surface emissivity and reflectivity at microwave frequencies over a water surface.

This function is a wrapper of different FASTEM versions Here is the caller graph for this function:

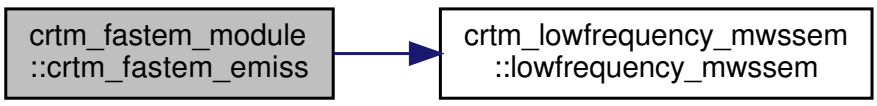


7.6.2.4 crtm_fastem_emiss()

```
integer function, public crtm_fastem_module::crtm_fastem_emiss (
            real(fp), intent(in) Frequency,
            real(fp), intent(in) Angle,
            real(fp), intent(in) Water_Temperature,
            real(fp), intent(in) Salinity,
            real(fp), intent(in) Wind_Speed,
            real(fp), intent(in) Wind_Direction,
            real(fp), dimension(n_stokes), intent(out) Emissivity,
            real(fp), dimension(n_stokes), intent(out) Reflectivity,
            integer, intent(in) FASTEM_Version,
            real(fp), intent(in) Sensor_Azimuth_Angle,
            real(fp), intent(in) Transmittance )
```

PURPOSE: Function to compute the surface emissivity and reflectivity at microwave frequencies over a water surface and at SINGLE frequency channel and SINGLE receiving angle

This function is a wrapper of different FASTEM versions Here is the call graph for this function:

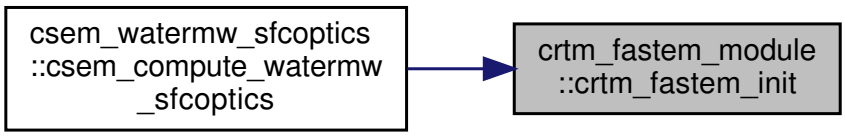


7.6.2.5 crtm_fastem_init()

```
integer function, public crtm_fastem_module::crtm_fastem_init (
             character(len=*) MWwaterCoeff_File,
             integer, intent(in) Version )
```

PURPOSE: Function to load FASTEM coefficient NETDCF files

This function must be called before calling other FASTEM functions Here is the caller graph for this function:



7.7 crtm_fastemxx Module Reference

Container Module for the Fastem4/5/6 models.

Functions/Subroutines

- subroutine, public compute_fastemxx (MWwaterCoeff, Frequency, n_Angles, Zenith_Angle, Temperature, Salinity, Wind_Speed, iVar, Emissivity, Reflectivity, Azimuth_Angle, Transmittance)
- subroutine, public compute_fastemxx_tl (MWwaterCoeff, Temperature_TL, Salinity_TL, Wind_Speed_TL, iVar, Emissivity_TL, Reflectivity_TL, Azimuth_Angle_TL, Transmittance_TL)
- subroutine, public compute_fastemxx_ad (MWwaterCoeff, Emissivity_AD, Reflectivity_AD, iVar, Temperature_AD, Salinity_AD, Wind_Speed_AD, Azimuth_Angle_AD, Transmittance_AD)

7.7.1 Detailed Description

Container Module for the Fastem4/5/6 models.

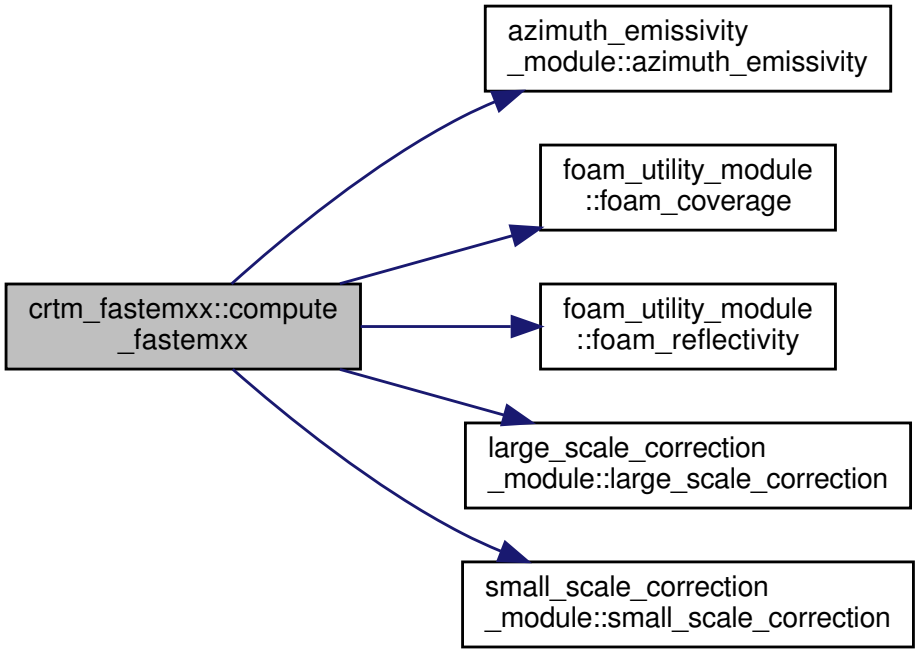
The difference between the Fastem4 and Fastem5 models is realised purely through the coefficients read during CRTM initialisation. For Fastem6, a different azimuth emissivity model is used.

7.7.2 Function/Subroutine Documentation

7.7.2.1 compute_fastemxx()

```
subroutine, public crtm_fastemxx::compute_fastemxx (
             type(mwwatercoeff_type), intent(in) MWwaterCoeff,
             real(fp), intent(in) Frequency,
             integer, intent(in) n_Angles,
             real(fp), intent(in) Zenith_Angle,
             real(fp), intent(in) Temperature,
             real(fp), intent(in) Salinity,
             real(fp), intent(in) Wind_Speed,
             type(ivar_type), intent(out) iVar,
             real(fp), dimension(:), intent(out) Emissivity,
             real(fp), dimension(:), intent(out) Reflectivity,
             real(fp), intent(in), optional Azimuth_Angle,
             real(fp), intent(in), optional Transmittance )
```

PURPOSE: Subroutine to compute the Fastem4 or Fastem5 microwave sea surface emissivity and reflectivity. Here is the call graph for this function:



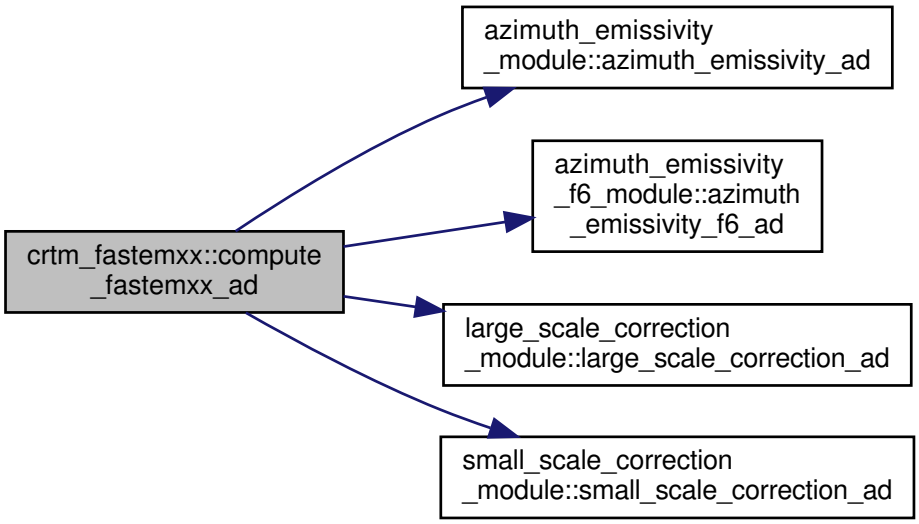
7.7.2.2 compute_fastemxx_ad()

```
subroutine, public crtm_fastemxx::compute_fastemxx_ad (
             type(mwwatercoeff_type), intent(in) MWwaterCoeff,
             real(fp), dimension(:), intent(inout) Emissivity_AD,
             real(fp), dimension(:), intent(inout) Reflectivity_AD,
             type(ivar_type), intent(in) iVar,
             real(fp), intent(inout) Temperature_AD,
             real(fp), intent(inout) Salinity_AD,
             real(fp), intent(inout) Wind_Speed_AD,
```

```
            real(fp), intent(inout), optional Azimuth_Angle_AD,
            real(fp), intent(inout), optional Transmittance_AD )
```

PURPOSE: Subroutine to compute the adjoint Fastem4 or Fastem5 microwave sea surface emissivity and reflectivity.

NOTE: The forward model must be called first to fill the internal variable argument with the intermediate forward calculations. Here is the call graph for this function:



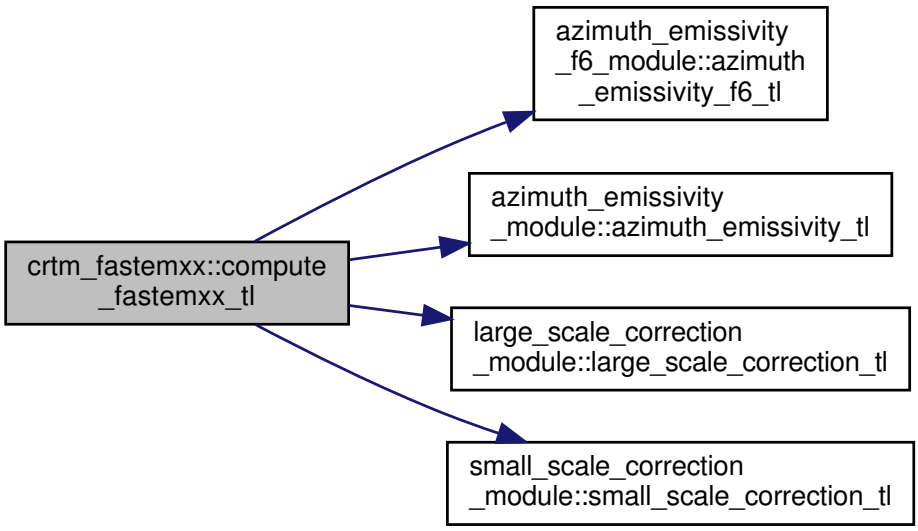
7.7.2.3 compute_fastemxx_tl()

```
subroutine, public crtm_fastemxx::compute_fastemxx_tl (
             type(mwwatercoeff_type), intent(in) MWwaterCoeff,
             real(fp), intent(in) Temperature_TL,
             real(fp), intent(in) Salinity_TL,
             real(fp), intent(in) Wind_Speed_TL,
             type(ivar_type), intent(in) iVar,
             real(fp), dimension(:), intent(out) Emissivity_TL,
             real(fp), dimension(:), intent(out) Reflectivity_TL,
             real(fp), intent(in), optional Azimuth_Angle_TL,
             real(fp), intent(in), optional Transmittance_TL )
```

PURPOSE: Subroutine to compute the tangent-linear Fastem4 or Fastem5 microwave sea surface emissivity and reflectivity.

NOTE: The forward model must be called first to fill the internal variable argument with the intermediate forward

calculations. Here is the call graph for this function:



7.8 crtm_lowfrequency_mwssem Module Reference

Module containg subroutines to compute microwave ocean emissivity components (FWD, TL, and AD) for low frequencies.

Functions/Subroutines

- subroutine, public lowfrequency_mwssem (Frequency, Zenith_Angle, Temperature, Salinity, Wind_Speed, Emissivity, iVar)
- subroutine, public **lowfrequency_mwssem_tl** (Temperature_TL, Salinity_TL, Wind_Speed_TL, Emissivity↩ _TL, iVar)
- subroutine, public **lowfrequency_mwssem_ad** (Emissivity_AD, Temperature_AD, Salinity_AD, Wind_↩ Speed_AD, iVar)

7.8.1 Detailed Description

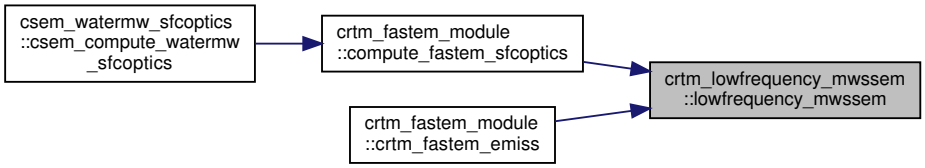
Module containg subroutines to compute microwave ocean emissivity components (FWD, TL, and AD) for low frequencies.

7.8.2 Function/Subroutine Documentation

7.8.2.1 lowfrequency_mwssem()

```
subroutine, public crtm_lowfrequency_mwssem::lowfrequency_mwssem (
             real(fp), intent(in) Frequency,
             real(fp), intent(in) Zenith_Angle,
             real(fp), intent(in) Temperature,
             real(fp), intent(in) Salinity,
             real(fp), intent(in) Wind_Speed,
             real(fp), dimension(:), intent(out) Emissivity,
             type(ivar_type), intent(inout) iVar )
```

PURPOSE: Subroutine to compute microwave sea surface emissivity for the frequency range 5GHz $<$ f $<$ 20GHz
Here is the caller graph for this function:



7.9 crtm_mwwatercoeff_define Module Reference

Module defining the MWwaterCoeff object.

Functions/Subroutines

- pure logical function, public crtm_mwwatercoeff_associated (self)
- pure subroutine, public **crtm_mwwatercoeff_destroy** (self)
- pure subroutine, public **crtm_mwwatercoeff_create** (self, ndim_subgrp, dims_subgrp)
- subroutine, public **crtm_mwwatercoeff_inspect** (self, pause)
- logical function, public **crtm_mwwatercoeff_validrelease** (self)
- subroutine, public **crtm_mwwatercoeff_info** (self, Info)
- subroutine, public **crtm_mwwatercoeff_defineversion** (Id)

7.9.1 Detailed Description

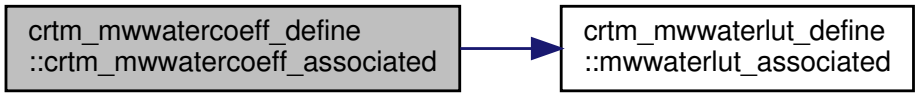
Module defining the MWwaterCoeff object.

7.9.2 Function/Subroutine Documentation

7.9.2.1 crtm_mwwatercoeff_associated()

```
pure logical function, public crtm_mwwatercoeff_define::crtm_mwwatercoeff_associated (
            type(crtm_mwwatercoeff_type), intent(in) self )
```

PURPOSE: Pure function to test the status of the allocatable components of the MWwaterCoeff structure. Here is the call graph for this function:



7.10 crtm_mwwaterlut_define Module Reference

Module defining the MWwaterLUT object containing the Look-Up Table (LUT) for the microWave (MW) sea surface emissivity model.

Functions/Subroutines

- pure logical function, public mwwaterlut_associated (self)
- pure subroutine, public **mwwaterlut_destroy** (self)
- pure subroutine, public **mwwaterlut_create** (self, n_Angles, n_Frequencies, n_Temperatures, n_Wind_↩ Speeds)
- subroutine, public **mwwaterlut_inspect** (self, pause)
- logical function, public **mwwaterlut_validrelease** (self)
- subroutine, public **mwwaterlut_info** (self, Info)
- subroutine, public **mwwaterlut_defineversion** (Id)

7.10.1 Detailed Description

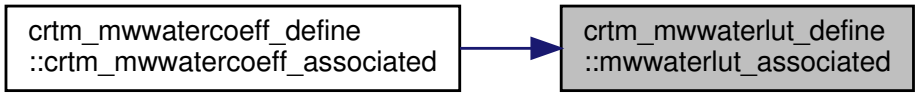
Module defining the MWwaterLUT object containing the Look-Up Table (LUT) for the microWave (MW) sea surface emissivity model.

7.10.2 Function/Subroutine Documentation

7.10.2.1 mwwaterlut_associated()

```
pure logical function, public crtm_mwwaterlut_define::mwwaterlut_associated (
            type(mwwaterlut_type), intent(in) self )
```

PURPOSE: Pure function to test the status of the allocatable components of the MWwaterLUT structure. Here is the caller graph for this function:



7.11 csem_define Module Reference

Module to define the general CSEM data structures.

Data Types

- type csem_land_surface
- type csem_water_surface
- type csem_snow_surface
- type csem_ice_surface
- type csem_sfcoptics_type
- type csem_sensorobs_struct
- type csem_geoinfo_struct
- type csem_atmosphere_parameters
- type csem_options_type

Functions/Subroutines

- subroutine **alloc_soil_profile** (land, n_Layers)
- elemental subroutine **clean_land** (land)
- subroutine **init_sfcoptics** (self, n_Angles)
- subroutine **alloc_sensorobs** (self, n_Channels)
- elemental subroutine **clean_sensorobs** (sensor)
- elemental subroutine **clean_sfcoptics** (sfcOptics)

7.11.1 Detailed Description

Module to define the general CSEM data structures.

7.12 csem_exception_handler Module Reference

Module currently used to define simple error/exit codes and output messages.

Functions/Subroutines

- recursive subroutine, public **display_message** (Routine_Name, Message, Error_State, Message_Log)
- integer function, public **open_message_log** (Message_Log, File_ID)

Variables

- integer, parameter, public **success** = 0
- integer, parameter, public **information** = 1
- integer, parameter, public **warning** = 2
- integer, parameter, public **failure** = 3
- integer, parameter, public **eof** = 4
- integer, parameter, public **undefined** = 5

7.12.1 Detailed Description

Module currently used to define simple error/exit codes and output messages.

7.13 csem_fitcoeff_define Module Reference

Module defining the FitCoeff objects.

Functions/Subroutines

- subroutine, public **csem_fitcoeff_defineversion** (Id)

Variables

- integer, parameter, public **fitcoeff_max_n_dimensions** = 3

7.13.1 Detailed Description

Module defining the FitCoeff objects.

7.14 csem_fresnel Module Reference

Module containing several algorithms for the calculation of Fresnel Reflectance and transmittance.

Data Types

- interface fresnel_reflectance
- interface fresnel_reflectance_tl
- interface fresnel_reflectance_ad

Functions/Subroutines

- subroutine **fresnel_reflectance_1** (em1, em2, theta_i, theta_t, rv, rh)
- subroutine **fresnel_reflectance_tl_1** (em1, em2, theta_i, theta_t, em1_TL, em2_TL, rv_TL, rh_TL)
- subroutine **fresnel_reflectance_ad_1** (em1, em2, theta_i, theta_t, em1_AD, em2_AD, rv_AD, rh_AD)
- subroutine **fresnel_reflectance_2** (em1, em2, theta_i, rv, rh)
- subroutine **fresnel_reflectance_tl_2** (em1, em2, theta_i, em1_TL, em2_TL, rv_TL, rh_TL)
- subroutine **fresnel_reflectance_ad_2** (em1, em2, theta_i, em1_AD, em2_AD, rv_AD, rh_AD)
- subroutine **fresnel_transmittance** (em1, em2, theta_i, theta_t, tv, th)
- subroutine **fresnel_reflectance_liou** (theta_i, em1, em2, rv, rh)
- subroutine **dispersion** (theta_i, emc, k_real, k_img)

7.14.1 Detailed Description

Module containing several algorithms for the calculation of Fresnel Reflectance and transmittance.

7.15 csem_iceir_sfcoptics Module Reference

Container module with all the IR_ICE models available in the CSEM model repository.

Functions/Subroutines

- integer function, public csem_compute_iceir_sfcoptics (Surface, SfcOptics, Options)

  PURPOSE: Function to compute the ice surface emissivity and reflectivity at infrared wavelength.
- integer function, public csem_compute_iceir_sfcoptics_tl (SfcOptics_TL)

  PURPOSE: Function to compute the ice surface emissivity and reflectivity tangent-linear at infrared wavelength.
- integer function, public csem_compute_iceir_sfcoptics_ad (Surface_AD)

  PURPOSE: Function to compute the ice surface emissivity and reflectivity adjoint at infrared wavelength.

7.15.1 Detailed Description

Container module with all the IR_ICE models available in the CSEM model repository.

The surface emissivity and reflectivity are required to determine the surface radiative contribution to the overall atmosphere radiative transfer system. This module is designed as a container to implement different ice surface radiative transfer models of infrared bands in the CSEM package. It also provides a generic interface for the upper-level applications to access all the available IR_ICE models. Each individual model has the FWD(Forward), TL(Tangent-linear) and AD(Adjoint) functions for the variational data assimilation and the surface parameter retrieval applications.

It replaces the similar functionality of the original CRTM surface module "CRTM_IceIR_SfcOptics", which was written by Paul van Delst, 23-Jun-2005

7.15.2 Function/Subroutine Documentation

7.15.2.1 csem_compute_iceir_sfcoptics()

```
integer function, public csem_iceir_sfcoptics::csem_compute_iceir_sfcoptics (
            type(csem_ice_surface), intent(in) Surface,
            type(csem_sfcoptics_type), intent(inout) SfcOptics,
            type(csem_options_type), intent(in), optional Options )
```

PURPOSE: Function to compute the ice surface emissivity and reflectivity at infrared wavelength.

It encapsulates all available IR_ICE RT models in the CSEM package, and provides the genereic interface for the upper-level user applications.

Parameters

in	*Surface*	CSEM Ice surface derived-type input UNITS: N/A TYPE: CSEM_Ice_Surface DIMENSION: Scalar
in	*Options*	CSEM derived-type for optional inputs, e.g., Gelocation & Time metadata, sensor observations which are needed by emprirical and semi-empirical models. UNITS: N/A TYPE: CSEM_Options_Type DIMENSION: Scalar
in,out	*SfcOptics*	CSEM SfcOptics derived-type, containing the in&out surface optical property variables in the radiative transfer calculation, e.g., the wavelength input and the surface emissivity/reflectivity outputs. UNITS: N/A TYPE: CSEM_SfcOptics_Type DIMENSION: Scalar

Returns

IO_Status: The return value is an integer defining the error status.
The error codes are defined in the CSEM_Exception_Handler module.
== SUCCESS the computation was sucessful
== FAILURE an unrecoverable error occurred

7.15.2.2 csem_compute_iceir_sfcoptics_ad()

```
integer function, public csem_iceir_sfcoptics::csem_compute_iceir_sfcoptics_ad (
            type(csem_ice_surface), intent(inout) Surface_AD )
```

PURPOSE: Function to compute the ice surface emissivity and reflectivity adjoint at infrared wavelength.

Parameters

in,out	*Surface*	CSEM_Ice_Surface adjoint UNITS: N/A TYPE: CSEM_Ice_Surface DIMENSION: Scalar

Returns

IO_Status: The return value is an integer defining the error status.
The error codes are defined in the CSEM_Exception_Handler module.
== SUCCESS the computation was sucessful
== FAILURE an unrecoverable error occurred

7.15.2.3 csem_compute_iceir_sfcoptics_tl()

```
integer function, public csem_iceir_sfcoptics::csem_compute_iceir_sfcoptics_tl (
            type(csem_sfcoptics_type), intent(inout) SfcOptics_TL )
```

PURPOSE: Function to compute the ice surface emissivity and reflectivity tangent-linear at infrared wavelength.

Parameters

in,out	*SfcOptics_TL*	CSEM SfcOptics tangent-linear output UNITS: N/A TYPE: CSEM_SfcOptics_Type DIMENSION: Scalar

Returns

IO_Status: The return value is an integer defining the error status.
The error codes are defined in the CSEM_Exception_Handler module.
== SUCCESS the computation was sucessful
== FAILURE an unrecoverable error occurred

7.16 csem_icemw_sfcoptics Module Reference

Container module for all the MW_ICE models available in the CSEM model repository.

Functions/Subroutines

- integer function, public csem_compute_icemw_sfcoptics (Surface, SfcOptics, Options)

  PURPOSE: Function to compute the sea-ice surface emissivity and reflectivity at microwave frequencies.
- integer function, public csem_compute_icemw_sfcoptics_tl (CSEM_SfcOptics_TL)

  PURPOSE: Function to compute the ice surface emissivity and reflectivity tangent-linear at microwave frequencies.
- integer function, public csem_compute_icemw_sfcoptics_ad (CSEM_Surface_AD)

  PURPOSE: Function to compute the ice surface emissivity and reflectivity adjoint at microwave frequencies.

7.16.1 Detailed Description

Container module for all the MW_ICE models available in the CSEM model repository.

The surface emissivity and reflectivity are required to determine the surface radiative contribution to the overall atmosphere radiative transfer system. This module is designed as a container to implement different sea-ice surface radiative transfer models of microwave frequencies in the CSEM package. It also provides a generic interface for the upper-level applications to access all the available MW_ICE models. Each individual model has the FWD(Forward), TL(Tangent-linear) and AD(Adjoint) functions for the variational data assimilation and the surface parameter retrieval applications.

It replaces the similar functionality of the original CRTM surface module "CRTM_IceMW_SfcOptics", which was written by Paul van Delst, 23-Jun-2005

7.16.2 Function/Subroutine Documentation

7.16.2.1 csem_compute_icemw_sfcoptics()

```
integer function, public csem_icemw_sfcoptics::csem_compute_icemw_sfcoptics (
            type(csem_ice_surface), intent(in) Surface,
            type(csem_sfcoptics_type), intent(inout) SfcOptics,
            type(csem_options_type), intent(in) Options )
```

PURPOSE: Function to compute the sea-ice surface emissivity and reflectivity at microwave frequencies.

It encapsulates all available MW_ICE emissivity models in the CSEM package, and provides the genereic interface for the upper-level user applications.

Parameters

in	*Surface*	CSEM Ice surface derived-type input UNITS: N/A TYPE: CSEM_Ice_Surface DIMENSION: Scalar
in	*Options*	CSEM derived-type for optional inputs, e.g., Gelocation & Time metadata, sensor observations which are needed by emprirical and semi-empirical models. UNITS: N/A TYPE: CSEM_Options_Type DIMENSION: Scalar
in,out	*SfcOptics*	CSEM SfcOptics derived-type, containing the in&out surface optical property variables in the radiative transfer calculation, e.g., the wavelength input and the surface emissivity/reflectivity outputs. UNITS: N/A TYPE: CSEM_SfcOptics_Type DIMENSION: Scalar
out	*iVar*	Composite data structure containing internal variables required for the subsequent tangent-linear and adjoint model calls. The contents of this structure are NOT accessible outside of this module. UNITS: N/A TYPE: iVar_type DIMENSION: Scalar
out	*IO_Status*	The return value is an integer defining the error status. The error codes are defined in the CSEM_Exception_Handler module. == SUCCESS the computation was sucessful == FAILURE an unrecoverable error occurred UNITS: N/A TYPE: INTEGER DIMENSION: Scalar

Returns

IO_Status: The return value is an integer defining the error status.
The error codes are defined in the CSEM_Exception_Handler module.

== SUCCESS the computation was sucessful
== FAILURE an unrecoverable error occurred

7.16.2.2 csem_compute_icemw_sfcoptics_ad()

```
integer function, public csem_icemw_sfcoptics::csem_compute_icemw_sfcoptics_ad (
            type(csem_ice_surface), intent(inout) CSEM_Surface_AD )
```

PURPOSE: Function to compute the ice surface emissivity and reflectivity adjoint at microwave frequencies.

Parameters

in,out	*Surface*	CSEM_Ice_Surface adjoint UNITS: N/A TYPE: CSEM_Ice_Surface DIMENSION: Scalar
out	*IO_Status*	The return value is an integer defining the error status. The error codes are defined in the CSEM_Exception_Handler module. == SUCCESS the computation was sucessful == FAILURE an unrecoverable error occurred UNITS: N/A TYPE: INTEGER DIMENSION: Scalar

Returns

IO_Status: The return value is an integer defining the error status.
The error codes are defined in the CSEM_Exception_Handler module.
== SUCCESS the computation was sucessful
== FAILURE an unrecoverable error occurred

7.16.2.3 csem_compute_icemw_sfcoptics_tl()

```
integer function, public csem_icemw_sfcoptics::csem_compute_icemw_sfcoptics_tl (
            type(csem_sfcoptics_type), intent(inout) CSEM_SfcOptics_TL )
```

PURPOSE: Function to compute the ice surface emissivity and reflectivity tangent-linear at microwave frequencies.

Parameters

in,out	*SfcOptics_TL*	CSEM SfcOptics tangent-linear output UNITS: N/A TYPE: CSEM_SfcOptics_Type DIMENSION: Scalar

Parameters

out	*IO_Status*	The return value is an integer defining the error status. The error codes are defined in the CSEM_Exception_Handler module. == SUCCESS the computation was sucessful == FAILURE an unrecoverable error occurred UNITS: N/A TYPE: INTEGER DIMENSION: Scalar

Returns

IO_Status: The return value is an integer defining the error status.
The error codes are defined in the CSEM_Exception_Handler module.
== SUCCESS the computation was sucessful
== FAILURE an unrecoverable error occurred

7.17 csem_icevis_sfcoptics Module Reference

Container module with all the VIS_ICE models available in the CSEM model repository.

Functions/Subroutines

- integer function, public csem_compute_icevis_sfcoptics (Surface, SfcOptics, Options)

  PURPOSE: Function to compute the ice surface emissivity and reflectivity at visible wavelength.
- integer function, public csem_compute_icevis_sfcoptics_tl (SfcOptics_TL)

  PURPOSE: Function to compute the ice surface emissivity and reflectivity tangent-linear at visible wavelength.
- integer function, public csem_compute_icevis_sfcoptics_ad (Surface_AD)

  PURPOSE: Function to compute the ice surface emissivity and reflectivity adjoint at visible wavelength.

7.17.1 Detailed Description

Container module with all the VIS_ICE models available in the CSEM model repository.

The surface emissivity and reflectivity are required to determine the surface radiative contribution to the overall atmosphere radiative transfer system. This module is designed as a container to implement different ice surface radiative transfer models of visible bands in the CSEM package. It also provides a generic interface for the upper-level applications to access all the available VIS_ICE models. Each individual model has the FWD(Forward), TL(Tangent-linear) and AD(Adjoint) functions for the variational data assimilation and the surface parameter retrieval applications.

It replaces the similar functionality of the original CRTM surface module "CRTM_IceVIS_SfcOptics", which was written by Paul van Delst, 23-Jun-2005

7.17.2 Function/Subroutine Documentation

7.17.2.1 csem_compute_icevis_sfcoptics()

```
integer function, public csem_icevis_sfcoptics::csem_compute_icevis_sfcoptics (
            type(csem_ice_surface), intent(in) Surface,
            type(csem_sfcoptics_type), intent(inout) SfcOptics,
            type(csem_options_type), intent(in), optional Options )
```

PURPOSE: Function to compute the ice surface emissivity and reflectivity at visible wavelength.

It encapsulates all available VIS_ICE emissivity models in the CSEM package, and provides the genereic interface for the upper-level user applications.

Parameters

in	*Surface*	CSEM Ice surface derived-type input UNITS: N/A TYPE: CSEM_Ice_Surface DIMENSION: Scalar
in	*Options*	CSEM derived-type for optional inputs, e.g., Gelocation & Time metadata, sensor observations which are needed by emprirical and semi-empirical models. UNITS: N/A TYPE: CSEM_Options_Type DIMENSION: Scalar
in,out	*SfcOptics*	CSEM SfcOptics derived-type, containing the in&out surface optical property variables in the radiative transfer calculation, e.g., the wavelength input and the surface emissivity/reflectivity outputs. UNITS: N/A TYPE: CSEM_SfcOptics_Type DIMENSION: Scalar

Returns

IO_Status: The return value is an integer defining the error status.
The error codes are defined in the CSEM_Exception_Handler module.
== SUCCESS the computation was sucessful
== FAILURE an unrecoverable error occurred

7.17.2.2 csem_compute_icevis_sfcoptics_ad()

```
integer function, public csem_icevis_sfcoptics::csem_compute_icevis_sfcoptics_ad (
            type(csem_ice_surface), intent(inout) Surface_AD )
```

PURPOSE: Function to compute the ice surface emissivity and reflectivity adjoint at visible wavelength.

Parameters

in,out	*Surface*	CSEM_Ice_Surface adjoint UNITS: N/A TYPE: CSEM_Ice_Surface DIMENSION: Scalar

Returns

IO_Status: The return value is an integer defining the error status.
The error codes are defined in the CSEM_Exception_Handler module.
== SUCCESS the computation was sucessful
== FAILURE an unrecoverable error occurred

7.17.2.3 csem_compute_icevis_sfcoptics_tl()

```
integer function, public csem_icevis_sfcoptics::csem_compute_icevis_sfcoptics_tl (
            type(csem_sfcoptics_type), intent(inout) SfcOptics_TL )
```

PURPOSE: Function to compute the ice surface emissivity and reflectivity tangent-linear at visible wavelength.

Parameters

in,out	*SfcOptics_TL*	CSEM SfcOptics tangent-linear output UNITS: N/A TYPE: CSEM_SfcOptics_Type DIMENSION: Scalar

Returns

IO_Status: The return value is an integer defining the error status.
The error codes are defined in the CSEM_Exception_Handler module.
== SUCCESS the computation was sucessful
== FAILURE an unrecoverable error occurred

7.18 csem_landir_sfcoptics Module Reference

Container module with all the IR_LAND models available in the CSEM model repository.

Functions/Subroutines

- integer function, public csem_compute_landir_sfcoptics (Surface, SfcOptics, Options)

  PURPOSE: Function to compute the land surface emissivity and reflectivity at infrared wavelength.
- integer function, public csem_compute_landir_sfcoptics_tl (SfcOptics_TL)

  PURPOSE: Function to compute the land surface emissivity and reflectivity tangent-linear at infrared wavelength.
- integer function, public csem_compute_landir_sfcoptics_ad (Surface_AD)

  PURPOSE: Function to compute the Snowsurface emissivity and reflectivity adjoint at infrared wavelength.

7.18.1 Detailed Description

Container module with all the IR_LAND models available in the CSEM model repository.

The surface emissivity and reflectivity are required to determine the surface radiative contribution to the overall atmosphere radiative transfer system. This module is designed as a container to implement different land surface radiative transfer models of infrared bands in the CSEM package. It also provides a generic interface for the upper-level applications to access all the available IR_LAND models. Each individual model has the FWD(Forward), TL(Tangent-linear) and AD(Adjoint) functions for the variational data assimilation and the surface parameter retrieval applications.

It replaces the similar functionality of the original CRTM surface module "CRTM_LandIR_SfcOptics", which was written by Paul van Delst, 23-Jun-2005

7.18.2 Function/Subroutine Documentation

7.18.2.1 csem_compute_landir_sfcoptics()

```
integer function, public csem_landir_sfcoptics::csem_compute_landir_sfcoptics (
            type(csem_land_surface), intent(in) Surface,
            type(csem_sfcoptics_type), intent(inout) SfcOptics,
            type(csem_options_type), intent(in) Options )
```

PURPOSE: Function to compute the land surface emissivity and reflectivity at infrared wavelength.

This function encapsulates all the available CSEM IR_LAND RT models, and provides a genereic interface for the upper-level user applications.

Parameters

in	*Surface*	CSEM land surface derived-type input UNITS: N/A TYPE: CSEM_Land_Surface DIMENSION: Scalar
in	*Options*	CSEM derived-type for optional inputs, e.g., Gelocation & Time metadata, sensor observations which are needed by emprirical and semi-empirical models. UNITS: N/A TYPE: CSEM_Options_Type DIMENSION: Scalar
in,out	*SfcOptics*	CSEM SfcOptics derived-type, containing the in&out surface optical property variables in the radiative transfer calculation, e.g., the wavelength input and the surface emissivity/reflectivity outputs. UNITS: N/A TYPE: CSEM_SfcOptics_Type DIMENSION: Scalar

Returns

IO_Status: The return value is an integer defining the error status.
The error codes are defined in the CSEM_Exception_Handler module.
== SUCCESS the computation was sucessful
== FAILURE an unrecoverable error occurred

7.18.2.2 csem_compute_landir_sfcoptics_ad()

```
integer function, public csem_landir_sfcoptics::csem_compute_landir_sfcoptics_ad (
            type(csem_land_surface), intent(inout) Surface_AD )
```

PURPOSE: Function to compute the Snowsurface emissivity and reflectivity adjoint at infrared wavelength.

Parameters

in,out	*Surface*	CSEM_Land_Surface adjoint UNITS: N/A TYPE: CSEM_Land_Surface DIMENSION: Scalar

Returns

IO_Status: The return value is an integer defining the error status.
The error codes are defined in the CSEM_Exception_Handler module.
== SUCCESS the computation was sucessful
== FAILURE an unrecoverable error occurred

7.18.2.3 csem_compute_landir_sfcoptics_tl()

```
integer function, public csem_landir_sfcoptics::csem_compute_landir_sfcoptics_tl (
            type(csem_sfcoptics_type), intent(inout) SfcOptics_TL )
```

PURPOSE: Function to compute the land surface emissivity and reflectivity tangent-linear at infrared wavelength.

Parameters

in,out	*SfcOptics_TL*	CSEM SfcOptics tangent-linear output UNITS: N/A TYPE: CSEM_SfcOptics_Type DIMENSION: Scalar

Returns

IO_Status: The return value is an integer defining the error status.
The error codes are defined in the CSEM_Exception_Handler module.
== SUCCESS the computation was sucessful
== FAILURE an unrecoverable error occurred

7.19 csem_landmw_sfcoptics Module Reference

Container module with all the MW_LAND models available in the CSEM model repository.

Functions/Subroutines

- integer function, public csem_compute_landmw_sfcoptics (Surface, SfcOptics, Options, iVar)

  PURPOSE: Function to compute the land surface emissivity and reflectivity at microwave frequencies.
- subroutine, public **get_ref_index** (Frequency, Polarization, i_ref_h, i_ref_v)
- integer function, public csem_compute_landmw_sfcoptics_tl (Surface_TL, SfcOptics_TL, iVar)

  PURPOSE: Function to compute the land surface emissivity and reflectivity tangent-linear at microwave frequencies.
- integer function, public csem_compute_landmw_sfcoptics_ad (SfcOptics_AD, Surface_AD, iVar)

  PURPOSE: Function to compute the land surface emissivity and reflectivity adjoint at microwave frequencies.

7.19.1 Detailed Description

Container module with all the MW_LAND models available in the CSEM model repository.

The surface emissivity and reflectivity are required to determine the surface radiative contribution to the overall atmosphere radiative transfer system. This module is designed as a container to implement different land surface radiative transfer models of microwave frequencies in the CSEM package. It also provides a generic interface for the upper-level applications to access all the available MW_LAND models. Each individual model has the FWD(Forward), TL(Tangent-linear) and AD(Adjoint) functions for the variational data assimilation and the surface parameter retrieval applications.

It replaces the similar functionality of the original CRTM surface module "CRTM_LandMW_SfcOptics", which was written by Paul van Delst, 23-Jun-2005

7.19.2 Function/Subroutine Documentation

7.19.2.1 csem_compute_landmw_sfcoptics()

```
integer function, public csem_landmw_sfcoptics::csem_compute_landmw_sfcoptics (
             type(csem_land_surface), intent(in) Surface,
             type(csem_sfcoptics_type), intent(inout) SfcOptics,
             type(csem_options_type), intent(in) Options,
             type(ivar_type), intent(out) iVar )
```

PURPOSE: Function to compute the land surface emissivity and reflectivity at microwave frequencies.

It encapsulates all available MW_LAND RT models in the CSEM package, and provides the genereic interface for the upper-level user applications.

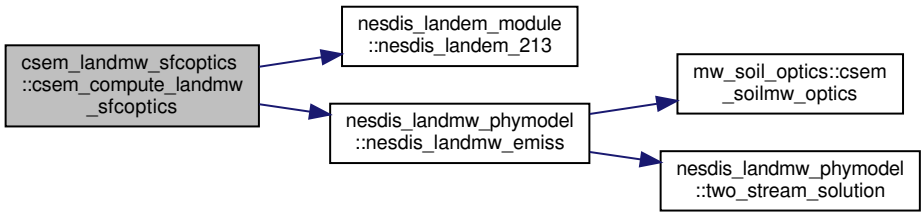
Parameters

in	*Surface*	CSEM land surface derived-type input UNITS: N/A TYPE: CSEM_Land_Surface DIMENSION: Scalar
in	*Options*	CSEM derived-type for optional inputs, e.g., Gelocation & Time metadata, sensor observations which are needed by emprirical and semi-empirical models. UNITS: N/A TYPE: CSEM_Options_Type DIMENSION: Scalar
in,out	*SfcOptics*	CSEM SfcOptics derived-type, containing the in&out surface optical property variables in the radiative transfer calculation, e.g., the wavelength input and the surface emissivity/reflectivity outputs. UNITS: N/A TYPE: CSEM_SfcOptics_Type DIMENSION: Scalar
out	*iVar*	Composite data structure containing internal variables required for the subsequent tangent-linear and adjoint model calls. The contents of this structure are NOT accessible outside of this module. UNITS: N/A TYPE: iVar_type DIMENSION: Scalar

Returns

IO_Status: The return value is an integer defining the error status.
The error codes are defined in the CSEM_Exception_Handler module.
== SUCCESS the computation was sucessful
== FAILURE an unrecoverable error occurred

Here is the call graph for this function:



7.19.2.2 csem_compute_landmw_sfcoptics_ad()

```
integer function, public csem_landmw_sfcoptics::csem_compute_landmw_sfcoptics_ad (
            type(csem_sfcoptics_type), intent(inout) SfcOptics_AD,
            type(csem_land_surface), intent(inout) Surface_AD,
            type(ivar_type), intent(in) iVar )
```

PURPOSE: Function to compute the land surface emissivity and reflectivity adjoint at microwave frequencies.

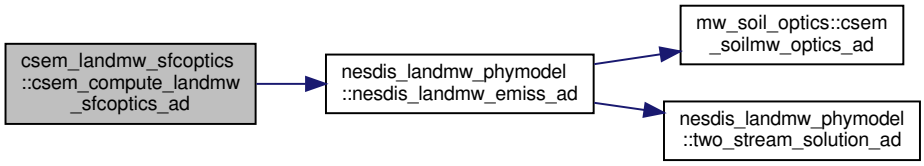
Parameters

in,out	*Surface_AD*	CSEM_Land_Surface adjoint outputs UNITS: N/A TYPE: CSEM_Land_Surface DIMENSION: Scalar
in,out	*SfcOptics_AD*	CSEM SfcOptics adjoint inputs UNITS: N/A TYPE: CSEM_SfcOptics_Type DIMENSION: Scalar
in	*iVar*	Composite data structure containing internal variables required for the subsequent tangent-linear and adjoint model calls. The contents of this structure are NOT accessible outside of this module. UNITS: N/A TYPE: iVar_type DIMENSION: Scalar

Returns

IO_Status: The return value is an integer defining the error status.
The error codes are defined in the CSEM_Exception_Handler module.
== SUCCESS the computation was sucessful
== FAILURE an unrecoverable error occurred

Here is the call graph for this function:



7.19.2.3 csem_compute_landmw_sfcoptics_tl()

```
integer function, public csem_landmw_sfcoptics::csem_compute_landmw_sfcoptics_tl (
            type(csem_land_surface), intent(in) Surface_TL,
            type(csem_sfcoptics_type), intent(inout) SfcOptics_TL,
            type(ivar_type), intent(in) iVar )
```

PURPOSE: Function to compute the land surface emissivity and reflectivity tangent-linear at microwave frequencies.

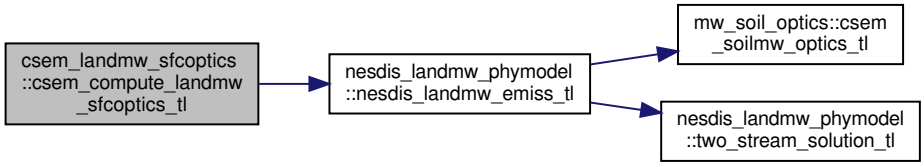
Parameters

in	*Surface_TL*	CSEM_Land_Surface tangent-linear inputs UNITS: N/A TYPE: CSEM_Land_Surface DIMENSION: Scalar
in,out	*SfcOptics_TL*	CSEM SfcOptics tangent-linear output UNITS: N/A TYPE: CSEM_SfcOptics_Type DIMENSION: Scalar
in	*iVar*	Composite data structure containing internal variables required for the subsequent tangent-linear and adjoint model calls. The contents of this structure are NOT accessible outside of this module. UNITS: N/A TYPE: iVar_type DIMENSION: Scalar

Returns

IO_Status: The return value is an integer defining the error status.
The error codes are defined in the CSEM_Exception_Handler module.
== SUCCESS the computation was sucessful
== FAILURE an unrecoverable error occurred

Here is the call graph for this function:



7.20 csem_landvis_sfcoptics Module Reference

Container module with all the VIS_LAND models available in the CSEM model repository.

Functions/Subroutines

- integer function, public csem_compute_landvis_sfcoptics (Surface, SfcOptics, Options)

  PURPOSE: Function to compute the land surface emissivity and reflectivity at visible wavelength.
- integer function, public csem_compute_landvis_sfcoptics_tl (SfcOptics_TL)

  PURPOSE: Function to compute the land surface emissivity and reflectivity tangent-linear at visible wavelength.
- integer function, public csem_compute_landvis_sfcoptics_ad (Surface_AD)

  PURPOSE: Function to compute the Snowsurface emissivity and reflectivity adjoint at visible wavelength.

7.20.1 Detailed Description

Container module with all the VIS_LAND models available in the CSEM model repository.

The surface emissivity and reflectivity are required to determine the surface radiative contribution to the overall atmosphere radiative transfer system. This module is designed as a container to implement different land surface radiative transfer models of visible bands in the CSEM package. It also provides a generic interface for the upper-level applications to access all the available VIS_LAND models. Each individual model has the FWD(Forward), TL(Tangent-linear) and AD(Adjoint) functions for the variational data assimilation and the surface parameter retrieval applications.

It replaces the similar functionality of the original CRTM surface module "CRTM_LandVIS_SfcOptics", which was written by Paul van Delst, 23-Jun-2005

7.20.2 Function/Subroutine Documentation

7.20.2.1 csem_compute_landvis_sfcoptics()

```
integer function, public csem_landvis_sfcoptics::csem_compute_landvis_sfcoptics (
            type(csem_land_surface), intent(in) Surface,
            type(csem_sfcoptics_type), intent(inout) SfcOptics,
            type(csem_options_type), intent(in), optional Options )
```

PURPOSE: Function to compute the land surface emissivity and reflectivity at visible wavelength.

This function encapsulates all the available CSEM VIS_LAND RT models, and provides a genereic interface for the upper-level user applications.

Parameters

in	*Surface*	CSEM Land surface derived-type input UNITS: N/A TYPE: CSEM_Land_Surface DIMENSION: Scalar
in	*Options*	CSEM derived-type for optional inputs, e.g., Gelocation & Time metadata, sensor observations which are needed by emprirical and semi-empirical models. UNITS: N/A TYPE: CSEM_Options_Type DIMENSION: Scalar
in,out	*SfcOptics*	CSEM SfcOptics derived-type, containing the in&out surface optical property variables in the radiative transfer calculation, e.g., the wavelength input and the surface emissivity/reflectivity outputs. UNITS: N/A TYPE: CSEM_SfcOptics_Type DIMENSION: Scalar
in,out	*iVar*	Composite data structure containing internal variables required for the subsequent tangent-linear and adjoint model calls. The contents of this structure are NOT accessible outside of this module. UNITS: N/A TYPE: iVar_type DIMENSION: Scalar

Returns

IO_Status: The return value is an integer defining the error status.
The error codes are defined in the CSEM_Exception_Handler module.
== SUCCESS the computation was sucessful
== FAILURE an unrecoverable error occurred

7.20.2.2 csem_compute_landvis_sfcoptics_ad()

```
integer function, public csem_landvis_sfcoptics::csem_compute_landvis_sfcoptics_ad (
            type(csem_land_surface), intent(inout) Surface_AD )
```

PURPOSE: Function to compute the Snowsurface emissivity and reflectivity adjoint at visible wavelength.

Parameters

in,out	*Surface*	CSEM_Land_Surface adjoint UNITS: N/A TYPE: CSEM_Land_Surface DIMENSION: Scalar

Returns

IO_Status: The return value is an integer defining the error status.
The error codes are defined in the CSEM_Exception_Handler module.
== SUCCESS the computation was sucessful
== FAILURE an unrecoverable error occurred

7.20.2.3 csem_compute_landvis_sfcoptics_tl()

```
integer function, public csem_landvis_sfcoptics::csem_compute_landvis_sfcoptics_tl (
            type(csem_sfcoptics_type), intent(inout) SfcOptics_TL )
```

PURPOSE: Function to compute the land surface emissivity and reflectivity tangent-linear at visible wavelength.

Parameters

in,out	*SfcOptics_TL*	CSEM SfcOptics adjoint output UNITS: N/A TYPE: CSEM_SfcOptics_Type DIMENSION: Scalar

Returns

IO_Status: The return value is an integer defining the error status.
The error codes are defined in the CSEM_Exception_Handler module.
== SUCCESS the computation was sucessful
== FAILURE an unrecoverable error occurred

7.21 csem_lifecycle Module Reference

Module with the CSEM life cycle functions to initialize and destroy the CSEM space.

Functions/Subroutines

- integer function, public **csem_init** (Model_Registor_File)
- subroutine, public **csem_destroy** ()

7.21.1 Detailed Description

Module with the CSEM life cycle functions to initialize and destroy the CSEM space.

7.22 csem_model_manager Module Reference

Module containing functions to manage the all model options already implemented in CSEM and registered in the Model_Registor_File.

Functions/Subroutines

- integer function, public **load_model_repo** (Model_Registor_File)
- subroutine, public **set_model_option** (Model, verbose)
- type(csem_model_id) function, public **inq_model_option** (ModelClass)
- character(len=256) function, public **get_data_path** (ModelClass, ModelName)

7.22.1 Detailed Description

Module containing functions to manage the all model options already implemented in CSEM and registered in the Model_Registor_File.

7.23 csem_snowir_sfcoptics Module Reference

Container module with all the IR_SNOW models available in the CSEM model repository.

Functions/Subroutines

- integer function, public csem_compute_snowir_sfcoptics (Surface, SfcOptics, Options)

  PURPOSE: Function to compute the snow surface emissivity and reflectivity at infrared wavelength.
- integer function, public csem_compute_snowir_sfcoptics_tl (SfcOptics_TL)

  PURPOSE: Function to compute the snow surface emissivity and reflectivity tangent-linear at infrared wavelength.
- integer function, public csem_compute_snowir_sfcoptics_ad (Surface_AD)

  PURPOSE: Function to compute the Snow surface emissivity and reflectivity adjoint at infrared wavelength.

7.23.1 Detailed Description

Container module with all the IR_SNOW models available in the CSEM model repository.

The surface emissivity and reflectivity are required to determine the surface radiative contribution to the overall atmosphere radiative transfer system. This module is designed as a container to implement different infrared snow surface radiative transfer models in the CSEM package. It also provides a generic interface for the upper-level applications to access all the available IR_SNOW models. Each individual model has the FWD(Forward), TL(Tangent-linear) and AD(Adjoint) functions for the variational data assimilation and the surface parameter retrieval applications.

It replaces the similar functionality of the original CRTM surface module "CRTM_SnowIR_SfcOptics", which was written by Paul van Delst, 23-Jun-2005

7.23.2 Function/Subroutine Documentation

7.23.2.1 csem_compute_snowir_sfcoptics()

```
integer function, public csem_snowir_sfcoptics::csem_compute_snowir_sfcoptics (
            type(csem_snow_surface), intent(in) Surface,
            type(csem_sfcoptics_type), intent(inout) SfcOptics,
            type(csem_options_type), intent(in), optional Options )
```

PURPOSE: Function to compute the snow surface emissivity and reflectivity at infrared wavelength.

It encapsulates all available IR_SNOW emissivity models in the CSEM package, and provides the genereic interface for the upper-level user applications.

Parameters

in	*Surface*	CSEM Snow surface derived-type input UNITS: N/A TYPE: CSEM_Snow_Surface DIMENSION: Scalar
in	*Options*	CSEM derived-type for optional inputs, e.g., Gelocation & Time metadata, sensor observations which are needed by emprirical and semi-empirical models. UNITS: N/A TYPE: CSEM_Options_Type DIMENSION: Scalar
in,out	*SfcOptics*	CSEM SfcOptics derived-type, containing the in&out surface optical property variables in the radiative transfer calculation, e.g., the wavelength input and the surface emissivity/reflectivity outputs. UNITS: N/A

Returns

IO_Status: The return value is an integer defining the error status.
The error codes are defined in the CSEM_Exception_Handler module.
== SUCCESS the computation was sucessful
== FAILURE an unrecoverable error occurred

7.23.2.2 csem_compute_snowir_sfcoptics_ad()

```
integer function, public csem_snowir_sfcoptics::csem_compute_snowir_sfcoptics_ad (
            type(csem_snow_surface), intent(inout) Surface_AD )
```

PURPOSE: Function to compute the Snow surface emissivity and reflectivity adjoint at infrared wavelength.

Parameters

in,out	*Surface*	CSEM_Snow_Surface adjoint UNITS: N/A TYPE: CSEM_Snow_Surface DIMENSION: Scalar

Returns

IO_Status: The return value is an integer defining the error status.
The error codes are defined in the CSEM_Exception_Handler module.
== SUCCESS the computation was sucessful
== FAILURE an unrecoverable error occurred

7.23.2.3 csem_compute_snowir_sfcoptics_tl()

```
integer function, public csem_snowir_sfcoptics::csem_compute_snowir_sfcoptics_tl (
            type(csem_sfcoptics_type), intent(inout) SfcOptics_TL )
```

PURPOSE: Function to compute the snow surface emissivity and reflectivity tangent-linear at infrared wavelength.

Parameters

in,out	*SfcOptics_TL*	CSEM SfcOptics tangent-linear output UNITS: N/A TYPE: CSEM_SfcOptics_Type DIMENSION: Scalar

Returns

IO_Status: The return value is an integer defining the error status.
The error codes are defined in the CSEM_Exception_Handler module.
== SUCCESS the computation was sucessful
== FAILURE an unrecoverable error occurred

7.24 csem_snowmw_sfcoptics Module Reference

This module provides a generic interface for the upper-level applications to access all the MW_SNOW models available in the CSEM model repository.

Functions/Subroutines

- integer function, public csem_compute_snowmw_sfcoptics (Surface, SfcOptics, Options)

  PURPOSE: Function to compute the snow surface emissivity and reflectivity at microwave frequencies.
- integer function, public csem_compute_snowmw_sfcoptics_tl (CSEM_SfcOptics_TL)

  PURPOSE: Function to compute the snow surface emissivity and reflectivity tangent-linear at microwave frequencies.
- integer function, public csem_compute_snowmw_sfcoptics_ad (CSEM_Surface_AD)

  PURPOSE: Function to compute the Snowsurface emissivity and reflectivity adjoint at microwave frequencies.

7.24.1 Detailed Description

This module provides a generic interface for the upper-level applications to access all the MW_SNOW models available in the CSEM model repository.

The surface emissivity and reflectivity are required to determine the surface radiative contribution to the overall atmosphere radiative transfer system. This module is designed as a container to implement different snow surface radiative transfer models of microwave frequencies in the CSEM package. It also provides a generic interface for the upper-level applications to access all the available MW_SNOW models. Each individual model has the FWD(Forward), TL(Tangent-linear) and AD(Adjoint) functions for the variational data assimilation and the surface parameter retrieval applications.

It replaces the similar functionality of the original CRTM surface module "CRTM_SnowMW_SfcOptics", which was written by Paul van Delst, 23-Jun-2005

7.24.2 Function/Subroutine Documentation

7.24.2.1 csem_compute_snowmw_sfcoptics()

```
integer function, public csem_snowmw_sfcoptics::csem_compute_snowmw_sfcoptics (
            type(csem_snow_surface), intent(in) Surface,
            type(csem_sfcoptics_type), intent(inout) SfcOptics,
            type(csem_options_type), intent(in) Options )
```

PURPOSE: Function to compute the snow surface emissivity and reflectivity at microwave frequencies.

It encapsulates all available MW_SNOW emissivity models in the CSEM package, and provides the genereic interface for the upper-level user applications.

Parameters

in	*Surface*	CSEM Snow surface derived-type input UNITS: N/A TYPE: CSEM_Snow_Surface DIMENSION: Scalar
in	*Options*	CSEM derived-type for optional inputs, e.g., Gelocation & Time metadata, sensor observations which are needed by emprirical and semi-empirical models. UNITS: N/A TYPE: CSEM_Options_Type DIMENSION: Scalar
in,out	*SfcOptics*	CSEM SfcOptics derived-type, containing the in&out surface optical property variables in the radiative transfer calculation, e.g., the wavelength input and the surface emissivity/reflectivity outputs. UNITS: N/A TYPE: CSEM_SfcOptics_Type DIMENSION: Scalar
out	*iVar*	Composite data structure containing internal variables required for the subsequent tangent-linear and adjoint model calls. The contents of this structure are NOT accessible outside of this module. UNITS: N/A TYPE: iVar_type DIMENSION: Scalar

Returns

IO_Status: The return value is an integer defining the error status.
The error codes are defined in the CSEM_Exception_Handler module.
== SUCCESS the computation was sucessful
== FAILURE an unrecoverable error occurred

7.24.2.2 csem_compute_snowmw_sfcoptics_ad()

```
integer function, public csem_snowmw_sfcoptics::csem_compute_snowmw_sfcoptics_ad (
            type(csem_snow_surface), intent(inout) CSEM_Surface_AD )
```

PURPOSE: Function to compute the Snowsurface emissivity and reflectivity adjoint at microwave frequencies.

Parameters

in,out	*Surface*	CSEM_Snow_Surface adjoint UNITS: N/A TYPE: CSEM_Ice_Surface DIMENSION: Scalar

Returns

IO_Status: The return value is an integer defining the error status.
The error codes are defined in the CSEM_Exception_Handler module.
== SUCCESS the computation was sucessful
== FAILURE an unrecoverable error occurred

7.24.2.3 csem_compute_snowmw_sfcoptics_tl()

```
integer function, public csem_snowmw_sfcoptics::csem_compute_snowmw_sfcoptics_tl (
            type(csem_sfcoptics_type), intent(inout) CSEM_SfcOptics_TL )
```

PURPOSE: Function to compute the snow surface emissivity and reflectivity tangent-linear at microwave frequencies.

Parameters

in,out	*SfcOptics_TL*	CSEM SfcOptics tangent-linear output UNITS: N/A TYPE: CSEM_SfcOptics_Type DIMENSION: Scalar

Returns

IO_Status: The return value is an integer defining the error status.
The error codes are defined in the CSEM_Exception_Handler module.
== SUCCESS the computation was sucessful
== FAILURE an unrecoverable error occurred

7.25 csem_snowvis_sfcoptics Module Reference

Container module of all the VIS_SNOW models available in the CSEM model repository.

Functions/Subroutines

- integer function, public csem_compute_snowvis_sfcoptics (Surface, SfcOptics, Options)

  PURPOSE: Function to compute the snow surface emissivity and reflectivity at visible wavelength.
- integer function, public csem_compute_snowvis_sfcoptics_tl (SfcOptics_TL)

  PURPOSE: Function to compute the snow surface emissivity and reflectivity tangent-linear at visible wavelength.
- integer function, public csem_compute_snowvis_sfcoptics_ad (Surface_AD)

  PURPOSE: Function to compute the Snowsurface emissivity and reflectivity adjoint at visible wavelength.

7.25.1 Detailed Description

Container module of all the VIS_SNOW models available in the CSEM model repository.

The surface emissivity and reflectivity are required to determine the surface radiative contribution to the overall atmosphere radiative transfer system. This module is designed as a container to implement different visible snow surface radiative transfer models in the CSEM package. It also provides a generic interface for the upper-level applications to access all the available VIS_SNOW models. Each individual model has the FWD(Forward), TL(Tangent-linear) and AD(Adjoint) functions for the variational data assimilation and the surface parameter retrieval applications.

It replaces the similar functionality of the original CRTM surface module "CRTM_SnowIR_SfcOptics", which was written by Paul van Delst, 23-Jun-2005

7.25.2 Function/Subroutine Documentation

7.25.2.1 csem_compute_snowvis_sfcoptics()

```
integer function, public csem_snowvis_sfcoptics::csem_compute_snowvis_sfcoptics (
            type(csem_snow_surface), intent(in) Surface,
            type(csem_sfcoptics_type), intent(inout) SfcOptics,
            type(csem_options_type), intent(in), optional Options )
```

PURPOSE: Function to compute the snow surface emissivity and reflectivity at visible wavelength.

It encapsulates all available IR_SNOW emissivity models in the CSEM package, and provides the genereic interface for the upper-level user applications.

Parameters

in	*Surface*	CSEM Snow surface derived-type input UNITS: N/A TYPE: CSEM_Snow_Surface DIMENSION: Scalar
in	*Options*	CSEM derived-type for optional inputs, e.g., Gelocation & Time metadata, sensor observations which are needed by emprirical and semi-empirical models. UNITS: N/A TYPE: CSEM_Options_Type DIMENSION: Scalar
in,out	*SfcOptics*	CSEM SfcOptics derived-type, containing the in&out surface optical property variables in the radiative transfer calculation, e.g., the wavelength input and the surface emissivity/reflectivity outputs. UNITS: N/A TYPE: CSEM_SfcOptics_Type DIMENSION: Scalar

Returns

IO_Status: The return value is an integer defining the error status.
The error codes are defined in the CSEM_Exception_Handler module.
== SUCCESS the computation was sucessful
== FAILURE an unrecoverable error occurred

7.25.2.2 csem_compute_snowvis_sfcoptics_ad()

```
integer function, public csem_snowvis_sfcoptics::csem_compute_snowvis_sfcoptics_ad (
            type(csem_snow_surface), intent(inout) Surface_AD )
```

PURPOSE: Function to compute the Snowsurface emissivity and reflectivity adjoint at visible wavelength.

Parameters

in,out	*Surface*	CSEM_Snow_Surface adjoint UNITS: N/A TYPE: CSEM_Ice_Surface DIMENSION: Scalar

Returns

IO_Status: The return value is an integer defining the error status.
The error codes are defined in the CSEM_Exception_Handler module.
== SUCCESS the computation was sucessful
== FAILURE an unrecoverable error occurred

7.25.2.3 csem_compute_snowvis_sfcoptics_tl()

```
integer function, public csem_snowvis_sfcoptics::csem_compute_snowvis_sfcoptics_tl (
            type(csem_sfcoptics_type), intent(inout) SfcOptics_TL )
```

PURPOSE: Function to compute the snow surface emissivity and reflectivity tangent-linear at visible wavelength.

Parameters

in,out	*SfcOptics_TL*	CSEM SfcOptics tangent-linear output UNITS: N/A TYPE: CSEM_SfcOptics_Type DIMENSION: Scalar

Returns

IO_Status: The return value is an integer defining the error status.
The error codes are defined in the CSEM_Exception_Handler module.
== SUCCESS the computation was sucessful
== FAILURE an unrecoverable error occurred

7.26 csem_waterir_sfcoptics Module Reference

Container module with all the IR_WATER models available in the CSEM model repository.

Functions/Subroutines

- integer function, public csem_compute_waterir_sfcoptics (Surface, SfcOptics, Options, iVar)

  PURPOSE: Function to compute the ocean surface emissivity and reflectivity at infrared wavelength.
- integer function, public csem_compute_waterir_sfcoptics_tl (Surface_TL, SfcOptics_TL, iVar)

  PURPOSE: Function to compute the ocean surface emissivity and reflectivity tangent-linear at infrared wavelength.
- integer function, public csem_compute_waterir_sfcoptics_ad (SfcOptics_AD, Surface_AD, iVar)

  PURPOSE: Function to compute the ocean surface emissivity and reflectivity adjoint at infrared wavelength.

7.26.1 Detailed Description

Container module with all the IR_WATER models available in the CSEM model repository.

The surface emissivity and reflectivity are required to determine the surface radiative contribution to the overall atmosphere radiative transfer system. This module is designed as a container to implement different water surface radiative transfer models of infrared bandsin the CSEM package. It also provides a generic interface for the upper-level applications to access all the available IR_WATER models. Each individual model has the FWD(Forward), TL(Tangent-linear) and AD(Adjoint) functions for the variational data assimilation and the surface parameter retrieval applications.

It replaces the similar functionality of the original CRTM surface module "CRTM_WaterIR_SfcOptics", which was written by Paul van Delst, 23-Jun-2005

7.26.2 Function/Subroutine Documentation

7.26.2.1 csem_compute_waterir_sfcoptics()

```
integer function, public csem_waterir_sfcoptics::csem_compute_waterir_sfcoptics (
            type(csem_water_surface), intent(in) Surface,
            type(csem_sfcoptics_type), intent(inout) SfcOptics,
            type(csem_options_type), intent(in) Options,
            type(ivar_type), intent(out) iVar )
```

PURPOSE: Function to compute the ocean surface emissivity and reflectivity at infrared wavelength.

This function encapsulates all the available CSEM IR_WATER RT models, and provides a genereic interface for the upper-level user applications.

Parameters

in	*Surface*	CSEM water surface derived-type input UNITS: N/A TYPE: CSEM_Water_Surface DIMENSION: Scalar
in	*Options*	CSEM derived-type for optional inputs, e.g., Gelocation & Time metadata, sensor observations which are needed by emprirical and semi-empirical models. UNITS: N/A TYPE: CSEM_Options_Type DIMENSION: Scalar
in	*Options*	CSEM derived-type for optional inputs, e.g., Gelocation & Time metadata, sensor observations which are needed by emprirical and semi-empirical models. UNITS: N/A TYPE: CSEM_Options_Type DIMENSION: Scalar
in,out	*SfcOptics*	CSEM SfcOptics derived-type, containing the in&out surface optical property variables in the radiative transfer calculation, e.g., the wavelength input and the surface emissivity/reflectivity outputs. UNITS: N/A TYPE: CSEM_SfcOptics_Type DIMENSION: Scalar
in,out	*iVar*	Composite data structure containing internal variables required for the subsequent tangent-linear and adjoint model calls. The contents of this structure are NOT accessible outside of this module. UNITS: N/A TYPE: iVar_type DIMENSION: Scalar

Returns

IO_Status: The return value is an integer defining the error status.
The error codes are defined in the CSEM_Exception_Handler module.
== SUCCESS the computation was sucessful
== FAILURE an unrecoverable error occurred

7.26.2.2 csem_compute_waterir_sfcoptics_ad()

```
integer function, public csem_waterir_sfcoptics::csem_compute_waterir_sfcoptics_ad (
            type(csem_sfcoptics_type), intent(inout) SfcOptics_AD,
            type(csem_water_surface), intent(inout) Surface_AD,
            type(ivar_type) iVar )
```

PURPOSE: Function to compute the ocean surface emissivity and reflectivity adjoint at infrared wavelength.

Parameters

in,out	*Surface_AD*	CSEM_Water_Surface adjoint outputs UNITS: N/A TYPE: CSEM_Water_Surface DIMENSION: Scalar
in,out	*SfcOptics_AD*	CSEM SfcOptics adjoint inputs UNITS: N/A TYPE: CSEM_SfcOptics_Type DIMENSION: Scalar
in,out	*iVar*	Composite data structure containing internal variables required for the subsequent tangent-linear and adjoint model calls. The contents of this structure are NOT accessible outside of this module. UNITS: N/A TYPE: iVar_type DIMENSION: Scalar

Returns

IO_Status: The return value is an integer defining the error status.
The error codes are defined in the CSEM_Exception_Handler module.
== SUCCESS the computation was sucessful
== FAILURE an unrecoverable error occurred

7.26.2.3 csem_compute_waterir_sfcoptics_tl()

```
integer function, public csem_waterir_sfcoptics::csem_compute_waterir_sfcoptics_tl (
            type(csem_water_surface), intent(in) Surface_TL,
            type(csem_sfcoptics_type), intent(inout) SfcOptics_TL,
            type(ivar_type), intent(in) iVar )
```

PURPOSE: Function to compute the ocean surface emissivity and reflectivity tangent-linear at infrared wavelength.

Parameters

in	*Surface_TL*	CSEM_Water_Surface tangent-linear inputs UNITS: N/A TYPE: CSEM_Water_Surface DIMENSION: Scalar
in,out	*SfcOptics_TL*	CSEM SfcOptics tangent-linear output UNITS: N/A TYPE: CSEM_SfcOptics_Type DIMENSION: Scalar
in	*iVar*	Composite data structure containing internal variables required for the subsequent tangent-linear and adjoint model calls. The contents of this structure are NOT accessible outside of this module. UNITS: N/A TYPE: iVar_type

Returns

IO_Status: The return value is an integer defining the error status.
The error codes are defined in the CSEM_Exception_Handler module.
== SUCCESS the computation was sucessful
== FAILURE an unrecoverable error occurred

7.27 csem_watermw_sfcoptics Module Reference

Container module with all the MWWater models available in the CSEM model repository.

Functions/Subroutines

- integer function, public csem_compute_watermw_sfcoptics (Surface, SfcOptics, Options, iVar)

  PURPOSE: Function to compute the ocean surface emissivity and reflectivity at microwave frequencies.
- integer function, public csem_compute_watermw_sfcoptics_tl (Surface_TL, Atmos_TL, SfcOptics_TL, iVar)

  PURPOSE: Function to compute the ocean surface emissivity and reflectivity tangent-linear at microwave frequencies.
- integer function, public csem_compute_watermw_sfcoptics_ad (SfcOptics_AD, Surface_AD, Atmos_AD, iVar)

  PURPOSE: Function to compute the ocean surface emissivity and reflectivity adjoint at microwave frequencies.

7.27.1 Detailed Description

Container module with all the MWWater models available in the CSEM model repository.

The surface emissivity and reflectivity are required to determine the surface radiative contribution to the overall atmosphere radiative transfer system. This module is designed as a container to implement different microwave ocean surface radiative transfer models in the CSEM package. It also provides a generic interface for the upper-level applications to access all the available MWWater models. Each individual model has the FWD(Forward), TL(Tangent-linear) and AD(Adjoint) functions for the variational data assimilation applications and the surface parameter retrieval applications.

It replaces the similar functionality of the original CRTM surface module "CRTM_WaterMW_SfcOptics", which was written by Paul van Delst, 23-Jun-2005

7.27.2 Function/Subroutine Documentation

7.27.2.1 csem_compute_watermw_sfcoptics()

```
integer function, public csem_watermw_sfcoptics::csem_compute_watermw_sfcoptics (
            type(csem_water_surface), intent(in) Surface,
            type(csem_sfcoptics_type), intent(inout) SfcOptics,
            type(csem_options_type), intent(in) Options,
            type(ivar_type), intent(out) iVar )
```

PURPOSE: Function to compute the ocean surface emissivity and reflectivity at microwave frequencies.

This function encapsulates all the available CSEM WaterMW emissivity models, and provides a genereic interface for the upper-level user applications.

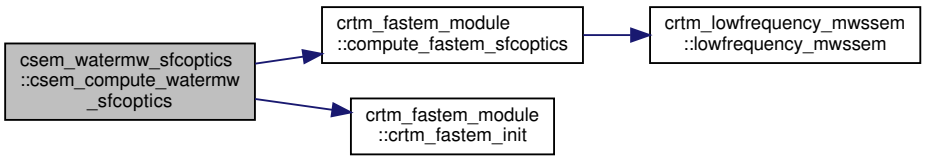
Parameters

in	*Surface*	CSEM water surface derived-type input UNITS: N/A TYPE: CSEM_Water_Surface DIMENSION: Scalar
in	*Options*	CSEM derived-type for optional inputs, e.g., Gelocation & Time metadata, sensor observations which are needed by emprirical and semi-empirical models. UNITS: N/A TYPE: CSEM_Options_Type DIMENSION: Scalar
in,out	*SfcOptics*	CSEM SfcOptics derived-type, containing the in&out surface optical property variables in the radiative transfer calculation, e.g., the wavelength input and the surface emissivity/reflectivity outputs. UNITS: N/A TYPE: CSEM_SfcOptics_Type DIMENSION: Scalar
in,out	*iVar*	Composite data structure containing internal variables required for the subsequent tangent-linear and adjoint model calls. The contents of this structure are NOT accessible outside of this module. UNITS: N/A TYPE: iVar_type DIMENSION: Scalar
out	*IO_Status*	The return value is an integer defining the error status. The error codes are defined in the CSEM_Exception_Handler module. == SUCCESS the computation was sucessful == FAILURE an unrecoverable error occurred UNITS: N/A TYPE: INTEGER DIMENSION: Scalar

Returns

IO_Status: The return value is an integer defining the error status.
The error codes are defined in the CSEM_Exception_Handler module.
== SUCCESS the computation was sucessful
== FAILURE an unrecoverable error occurred

Here is the call graph for this function:



7.27.2.2 csem_compute_watermw_sfcoptics_ad()

```
integer function, public csem_watermw_sfcoptics::csem_compute_watermw_sfcoptics_ad (
            type(csem_sfcoptics_type), intent(inout) SfcOptics_AD,
            type(csem_water_surface), intent(inout) Surface_AD,
            type(csem_atmosphere_parameters), intent(inout) Atmos_AD,
            type(ivar_type), intent(in) iVar )
```

PURPOSE: Function to compute the ocean surface emissivity and reflectivity adjoint at microwave frequencies.

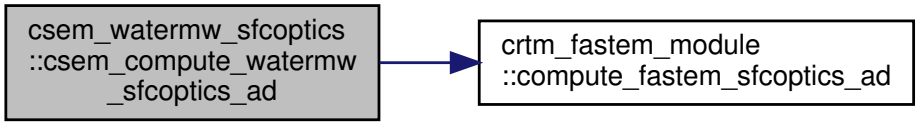
Parameters

in,out	*Surface_AD*	CSEM_Water_Surface adjoint outputs UNITS: N/A TYPE: CSEM_Water_Surface DIMENSION: Scalar
in,out	*Atmos_TL*	CSEM_Atmosphere_Parameters Adjoint output UNITS: N/A TYPE: CSEM_Atmosphere_Parameters DIMENSION: Scalar
in,out	*SfcOptics_AD*	CSEM SfcOptics adjoint inputs UNITS: N/A TYPE: CSEM_SfcOptics_Type DIMENSION: Scalar
in	*iVar*	Composite data structure containing internal variables required for the subsequent tangent-linear and adjoint model calls. The contents of this structure are NOT accessible outside of this module. UNITS: N/A TYPE: iVar_type DIMENSION: Scalar
out	*IO_Status*	The return value is an integer defining the error status. The error codes are defined in the CSEM_Exception_Handler module. == SUCCESS the computation was sucessful == FAILURE an unrecoverable error occurred UNITS: N/A TYPE: INTEGER DIMENSION: Scalar

Returns

IO_Status: The return value is an integer defining the error status.
The error codes are defined in the CSEM_Exception_Handler module.
== SUCCESS the computation was sucessful
== FAILURE an unrecoverable error occurred

Here is the call graph for this function:



7.27.2.3 csem_compute_watermw_sfcoptics_tl()

```
integer function, public csem_watermw_sfcoptics::csem_compute_watermw_sfcoptics_tl (
            type(csem_water_surface), intent(in) Surface_TL,
            type(csem_atmosphere_parameters), intent(in) Atmos_TL,
            type(csem_sfcoptics_type), intent(inout) SfcOptics_TL,
            type(ivar_type), intent(in) iVar )
```

PURPOSE: Function to compute the ocean surface emissivity and reflectivity tangent-linear at microwave frequencies.

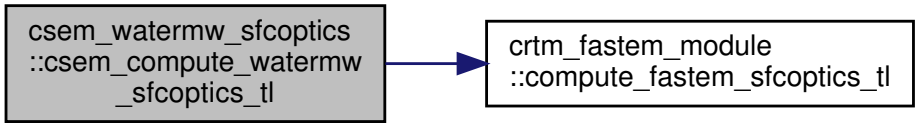
Parameters

in	*Surface_TL*	CSEM_Water_Surface tangent-linear inputs UNITS: N/A TYPE: CSEM_Water_Surface DIMENSION: Scalar
in	*Atmos_TL*	CSEM_Atmosphere_Parameters tangent-linear inputs UNITS: N/A TYPE: CSEM_Atmosphere_Parameters DIMENSION: Scalar
in,out	*SfcOptics_TL*	CSEM SfcOptics tangent-linear output UNITS: N/A TYPE: CSEM_SfcOptics_Type DIMENSION: Scalar
in,out	*iVar*	Composite data structure containing internal variables required for the subsequent tangent-linear and adjoint model calls. The contents of this structure are NOT accessible outside of this module. UNITS: N/A TYPE: iVar_type DIMENSION: Scalar
out	*IO_Status*	The return value is an integer defining the error status. The error codes are defined in the CSEM_Exception_Handler module. == SUCCESS the computation was sucessful == FAILURE an unrecoverable error occurred UNITS: N/A TYPE: INTEGER DIMENSION: Scalar

Returns

IO_Status: The return value is an integer defining the error status.
The error codes are defined in the CSEM_Exception_Handler module.
== SUCCESS the computation was sucessful
== FAILURE an unrecoverable error occurred

Here is the call graph for this function:



7.28 csem_watervis_sfcoptics Module Reference

Container module with all the VIS_WATER models available in the CSEM model repository.

Functions/Subroutines

- integer function, public csem_compute_watervis_sfcoptics (Surface, SfcOptics, Options, iVar)

  PURPOSE: Function to compute the ocean surface emissivity and reflectivity at visible wavelength.
- integer function, public csem_compute_watervis_sfcoptics_tl (SfcOptics_TL)

  PURPOSE: Function to compute the ocean surface emissivity and reflectivity tangent-linear at visible wavelength.
- integer function, public csem_compute_watervis_sfcoptics_ad (Surface_AD)

  PURPOSE: Function to compute the ocean surface emissivity and reflectivity adjoint at visible wavelength.

7.28.1 Detailed Description

Container module with all the VIS_WATER models available in the CSEM model repository.

The surface emissivity and reflectivity are required to determine the surface radiative contribution to the overall atmosphere radiative transfer system. This module is designed as a container to implement different water surface radiative transfer models of visible bands in the CSEM package. It also provides a generic interface for the upper-level applications to access all the available VIS_WATER models. Each individual model has the FWD(Forward), TL(Tangent-linear) and AD(Adjoint) functions for the variational data assimilation and the surface parameter retrieval applications.

It replaces the similar functionality of the original CRTM surface module "CRTM_WaterVIS_SfcOptics", which was written by Paul van Delst, 23-Jun-2005

7.28.2 Function/Subroutine Documentation

7.28.2.1 csem_compute_watervis_sfcoptics()

```
integer function, public csem_watervis_sfcoptics::csem_compute_watervis_sfcoptics (
             type(csem_water_surface), intent(in) Surface,
             type(csem_sfcoptics_type), intent(inout) SfcOptics,
             type(csem_options_type), intent(in), optional Options,
             type(ivar_type), optional iVar )
```

PURPOSE: Function to compute the ocean surface emissivity and reflectivity at visible wavelength.

This function encapsulates all the available CSEM VIS_WATER RT models, and provides a genereic interface for the upper-level user applications.

Parameters

in	*Surface*	CSEM water surface derived-type input UNITS: N/A TYPE: CSEM_Water_Surface DIMENSION: Scalar
in	*Options*	CSEM derived-type for optional inputs, e.g., Gelocation & Time metadata, sensor observations which are needed by emprirical and semi-empirical models. UNITS: N/A TYPE: CSEM_Options_Type DIMENSION: Scalar
in,out	*SfcOptics*	CSEM SfcOptics derived-type, containing the in&out surface optical property variables in the radiative transfer calculation, e.g., the wavelength input and the surface emissivity/reflectivity outputs. UNITS: N/A TYPE: CSEM_SfcOptics_Type DIMENSION: Scalar
in,out	*iVar*	Composite data structure containing internal variables required for the subsequent tangent-linear and adjoint model calls. The contents of this structure are NOT accessible outside of this module. UNITS: N/A TYPE: iVar_type DIMENSION: Scalar

Returns

IO_Status: The return value is an integer defining the error status.
The error codes are defined in the CSEM_Exception_Handler module.
== SUCCESS the computation was sucessful
== FAILURE an unrecoverable error occurred

7.28.2.2 csem_compute_watervis_sfcoptics_ad()

```
integer function, public csem_watervis_sfcoptics::csem_compute_watervis_sfcoptics_ad (
            type(csem_water_surface), intent(inout) Surface_AD )
```

PURPOSE: Function to compute the ocean surface emissivity and reflectivity adjoint at visible wavelength.

Parameters

in,out	*Surface_AD*	CSEM_Water_Surface adjoint outputs UNITS: N/A TYPE: CSEM_Water_Surface DIMENSION: Scalar

Returns

IO_Status: The return value is an integer defining the error status.
The error codes are defined in the CSEM_Exception_Handler module.
== SUCCESS the computation was sucessful
== FAILURE an unrecoverable error occurred

7.28.2.3 csem_compute_watervis_sfcoptics_tl()

```
integer function, public csem_watervis_sfcoptics::csem_compute_watervis_sfcoptics_tl (
            type(csem_sfcoptics_type), intent(inout) SfcOptics_TL )
```

PURPOSE: Function to compute the ocean surface emissivity and reflectivity tangent-linear at visible wavelength.

Parameters

in,out	*SfcOptics_TL*	CSEM SfcOptics tangent-linear output UNITS: N/A TYPE: CSEM_SfcOptics_Type DIMENSION: Scalar

Returns

IO_Status: The return value is an integer defining the error status.
The error codes are defined in the CSEM_Exception_Handler module.
== SUCCESS the computation was sucessful
== FAILURE an unrecoverable error occurred

7.29 ellison Module Reference

Ellison Ocean Permittivity module.

Functions/Subroutines

- subroutine, public **ellison_ocean_permittivity** (Temperature, Frequency, Permittivity, iVar)
- subroutine, public **ellison_ocean_permittivity_tl** (Temperature_TL, Permittivity_TL, iVar)
- subroutine, public **ellison_ocean_permittivity_ad** (Permittivity_AD, Temperature_AD, iVar)

7.29.1 Detailed Description

Ellison Ocean Permittivity module.

7.30 fastem_coeff_reader Module Reference

Module containing the load/destruction routines to handel the shared CSEM microwave water surface emissivity model data in NetCDF format.

Functions/Subroutines

- integer function, public **csem_mwwatercoeff_load** (Filename, File_Path, Quiet, Version, Process_ID, Output_Process_ID)
- integer function, public **csem_mwwatercoeff_cleanup** (Process_ID)
- logical function, public **csem_mwwatercoeff_isloaded** ()

Variables

- type(crtm_mwwatercoeff_type), save, public **csem_mwwaterc**

7.30.1 Detailed Description

Module containing the load/destruction routines to handel the shared CSEM microwave water surface emissivity model data in NetCDF format.

7.31 fastem_fresnel Module Reference

Module containing routines to compute Fresnel reflectivities.

Functions/Subroutines

- subroutine, public **fastem_fresnel_reflectivity** (permittivity, cos_i, Rv, Rh, iVar)
- subroutine, public **fastem_fresnel_reflectivity_tl** (permittivity_TL, cos_i, Rv_TL, Rh_TL, iVar)
- subroutine, public **fastem_fresnel_reflectivity_ad** (Rv_AD, Rh_AD, cos_i, permittivity_AD, iVar)

7.31.1 Detailed Description

Module containing routines to compute Fresnel reflectivities.

7.32 foam_utility_module Module Reference

Helper module containing the foam-related utility routines for the CRTM implementation of FASTEM4 and FASTEM5.

Functions/Subroutines

- subroutine, public foam_coverage (FCCoeff, wind_speed, coverage)
- subroutine, public **foam_coverage_tl** (FCCoeff, wind_speed, wind_speed_TL, coverage_TL)
- subroutine, public **foam_coverage_ad** (FCCoeff, wind_speed, coverage_AD, wind_speed_AD)
- subroutine, public foam_reflectivity (FRCoeff, Zenith_Angle, Frequency, Rv, Rh)

7.32.1 Detailed Description

Helper module containing the foam-related utility routines for the CRTM implementation of FASTEM4 and FASTEM5.

7.32.2 Function/Subroutine Documentation

7.32.2.1 foam_coverage()

```
subroutine, public foam_utility_module::foam_coverage (
             type(csem_fitcoeff_1d_type), intent(in) FCCoeff,
             real(fp), intent(in) wind_speed,
             real(fp), intent(out) coverage )
```

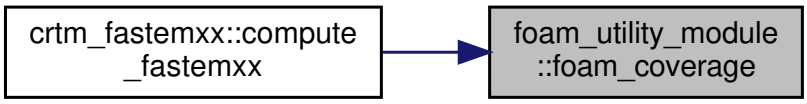
Foam coverage.

Monahan, E.C., and O'Muircheartaigh, I.G., (1986) Whitecaps and the passive remote sensing of the ocean surface, International Journal of Remote Sensing, 7, pp627-642.

The neutral stability condition is used here (i.e. the difference between the skin and air temperature is assumed to be zero) so that the form of the foam coverage equation is the same as in Tang (1974) and Liu et al. (1998)..

Liu, Q. et al. (1998) Monte Carlo simulations of the microwave emissivity of the sea surface. JGR, 103(C11), pp24983-24989

Tang, C. (1974) The effect of droplets in the air-sea transition zone on the sea brightness temperature. J. Phys. Oceanography, 4, pp579-593. Here is the caller graph for this function:



7.32.2.2 foam_reflectivity()

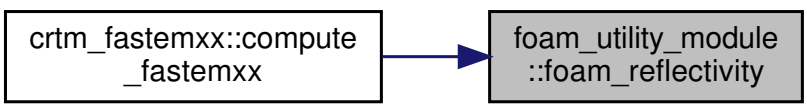
```
subroutine, public foam_utility_module::foam_reflectivity (
             type(csem_fitcoeff_1d_type), intent(in) FRCoeff,
             real(fp), intent(in) Zenith_Angle,
             real(fp), intent(in) Frequency,
             real(fp), intent(out) Rv,
             real(fp), intent(out) Rh )
```

Foam reflectivity

See section d in

Kazumori, M. et al. (2008) Impact Study of AMSR-E Radiances in the NCEP Global Data Assimilation System, Monthly Weather Review, 136, pp541-559

Function dependence is on zenith angle only so no TL or AD routine. Here is the caller graph for this function:



7.33 guillou Module Reference

Guillou Ocean Permittivity module.

Functions/Subroutines

- subroutine, public **guillou_ocean_permittivity** (Temperature, Salinity, Frequency, Permittivity, iVar)
- subroutine, public **guillou_ocean_permittivity_tl** (Temperature_TL, Salinity_TL, Frequency, Permittivity_TL, iVar)
- subroutine, public **guillou_ocean_permittivity_ad** (Permittivity_AD, Frequency, Temperature_AD, Salinity↩_AD, iVar)

7.33.1 Detailed Description

Guillou Ocean Permittivity module.

7.34 irssem_emiscoeff_define Module Reference

Module defining the EmisCoeff data structure and containing routines to manipulate it.

Functions/Subroutines

- logical function, public **associated_emiscoeff** (EmisCoeff, ANY_Test)
- integer function, public **destroy_emiscoeff** (EmisCoeff, No_Clear, RCS_Id, Message_Log)
- integer function, public **allocate_emiscoeff** (n_Angles, n_Frequencies, n_Wind_Speeds, EmisCoeff, RCS↩_Id, Message_Log)
- integer function, public **assign_emiscoeff** (EmisCoeff_in, EmisCoeff_out, RCS_Id, Message_Log)
- integer function, public **equal_emiscoeff** (EmisCoeff_LHS, EmisCoeff_RHS, ULP_Scale, Check_All, RCS↩_Id, Message_Log)
- integer function, public **check_emiscoeff_release** (EmisCoeff, RCS_Id, Message_Log)
- subroutine, public **info_emiscoeff** (EmisCoeff, Info, RCS_Id)

Variables

- integer(long), parameter, public **spectral_emiscoeff_type** = 1
- integer(long), parameter, public **sensor_emiscoeff_type** = 2
- integer(long), parameter, public **n_emiscoeff_items** = 4_Long
- integer(long), dimension(n_emiscoeff_items), parameter, public **emiscoeff_data_type** = (/ DOUBLE_TYPE, DOUBLE_TYPE, DOUBLE_TYPE, DOUBLE_TYPE /)
- character(∗), dimension(n_emiscoeff_items), parameter, public **emiscoeff_data_name** = (/ 'Angle ', 'Frequency ', 'Wind speed', 'Emissivity' /)

7.34.1 Detailed Description

Module defining the EmisCoeff data structure and containing routines to manipulate it.

7.35 irssem_emiscoeff_reader Module Reference

Module containing routines to read the netCDF format EmisCoeff files of the NESDIS physical Infrared ocean surface models.

Functions/Subroutines

- integer function, public **load_irssem_lut** (NC_Filename)
- integer function, public **close_irssem_lut** ()

Variables

- type(emiscoeff_type), save, public **irwaterc**

7.35.1 Detailed Description

Module containing routines to read the netCDF format EmisCoeff files of the NESDIS physical Infrared ocean surface models.

7.36 large_scale_correction_module Module Reference

Module containing the large-scale correction procedures for the CRTM implementations of FASTEM4 and FASTEM5.

Functions/Subroutines

- subroutine, public large_scale_correction (LSCCoeff, Frequency, cos_Z, Wind_Speed, Rv_Large, Rh_Large, iVar)
- subroutine, public large_scale_correction_tl (Wind_Speed_TL, Rv_Large_TL, Rh_Large_TL, iVar)
- subroutine, public large_scale_correction_ad (Rv_Large_AD, Rh_Large_AD, Wind_Speed_AD, iVar)

7.36.1 Detailed Description

Module containing the large-scale correction procedures for the CRTM implementations of FASTEM4 and FASTEM5.

Equations (A5a) and (A5b) of

Liu, Q. et al. (2011) An Improved Fast Microwave Water Emissivity Model, TGRSS, 49, pp1238-1250

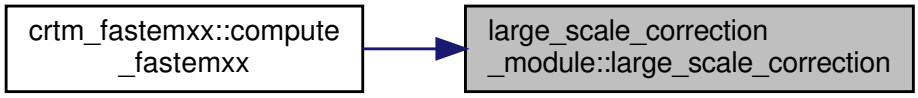
describes the fitting of the large-scale correction formulation. No explicit description of the data that was fitted is given.

7.36.2 Function/Subroutine Documentation

7.36.2.1 large_scale_correction()

```
subroutine, public large_scale_correction_module::large_scale_correction (
             type(csem_fitcoeff_3d_type), intent(in) LSCCoeff,
             real(fp), intent(in) Frequency,
             real(fp), intent(in) cos_Z,
             real(fp), intent(in) Wind_Speed,
             real(fp), intent(out) Rv_Large,
             real(fp), intent(out) Rh_Large,
             type(ivar_type), intent(inout) iVar )
```

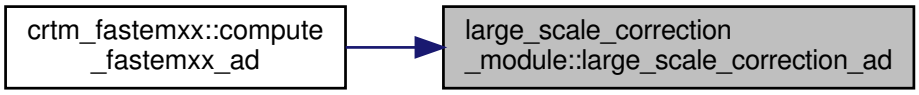
Procedures to compute the reflectivity large scale correction Here is the caller graph for this function:



7.36.2.2 large_scale_correction_ad()

```
subroutine, public large_scale_correction_module::large_scale_correction_ad (
             real(fp), intent(inout) Rv_Large_AD,
             real(fp), intent(inout) Rh_Large_AD,
             real(fp), intent(inout) Wind_Speed_AD,
             type(ivar_type), intent(in) iVar )
```

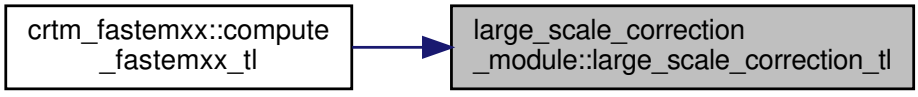
Adjoint model of Large_Scale_Correction Here is the caller graph for this function:



7.36.2.3 large_scale_correction_tl()

```
subroutine, public large_scale_correction_module::large_scale_correction_tl (
             real(fp), intent(in) Wind_Speed_TL,
             real(fp), intent(out) Rv_Large_TL,
             real(fp), intent(out) Rh_Large_TL,
             type(ivar_type), intent(in) iVar )
```

Tangent-linear model of Large_Scale_Correction Here is the caller graph for this function:



7.37 liu Module Reference

Liu Ocean Permittivity module.

Functions/Subroutines

- subroutine, public liu_ocean_permittivity (Temperature, Salinity, Frequency, Permittivity, iVar)

  PURPOSE: Subroutine to compute ocean permittivity according to the reference, Liu, Q. et al. (2010) An improved fast microwave water emissivity model. IEEE Trans. Geosci. Remote Sensing, accepted June 25, 2010.
- subroutine, public liu_ocean_permittivity_tl (Temperature_TL, Salinity_TL, Frequency, Permittivity_TL, iVar)

  PURPOSE: Subroutine to compute ocean permittivity according to the reference, Liu, Q. et al. (2010) An improved fast microwave water emissivity model. IEEE Trans. Geosci. Remote Sensing, accepted June 25, 2010.
- subroutine, public liu_ocean_permittivity_ad (Permittivity_AD, Frequency, Temperature_AD, Salinity_AD, iVar)

  PURPOSE: Subroutine to compute ocean permittivity according to the reference, Liu, Q. et al. (2010) An improved fast microwave water emissivity model. IEEE Trans. Geosci. Remote Sensing, accepted June 25, 2010.

7.37.1 Detailed Description

Liu Ocean Permittivity module.

Module containing routines to compute the complex permittivities for sea water based on

Liu, Q. et al. (2010) An improved fast microwave water emissivity model. IEEE Trans. Geosci. Remote Sensing, accepted June 25, 2010

7.37.2 Function/Subroutine Documentation

7.37.2.1 liu_ocean_permittivity()

```
subroutine, public liu::liu_ocean_permittivity (
             real(fp), intent(in) Temperature,
             real(fp), intent(in) Salinity,
             real(fp), intent(in) Frequency,
             complex(fp), intent(out) Permittivity,
             type(ivar_type), intent(inout) iVar )
```

PURPOSE: Subroutine to compute ocean permittivity according to the reference, Liu, Q. et al. (2010) An improved fast microwave water emissivity model. IEEE Trans. Geosci. Remote Sensing, accepted June 25, 2010.

Parameters

in	*Temperature*	Sea surface temperature UNITS: Kelvin (K) TYPE: REAL DIMENSION: Scalar

Parameters

in	*Salinity*	Ocean Water Salinity UNITS: ppt (parts per thousand) TYPE: REAL DIMENSION: Scalar
in	*Frequency*	Frequency UNITS: GHz TYPE: REAL DIMENSION: Scalar
out	*Permittivity*	Ocean permittivity UNITS: N/A TYPE: COMPLEX DIMENSION: Scalar
in	*iVar*	Composite data structure containing internal variables required for the subsequent tangent-linear and adjoint model calls. The contents of this structure are NOT accessible outside of this module. UNITS: N/A TYPE: iVar_type DIMENSION: Scalar

7.37.2.2 liu_ocean_permittivity_ad()

```
subroutine, public liu::liu_ocean_permittivity_ad (
            complex(fp), intent(inout) Permittivity_AD,
            real(fp), intent(in) Frequency,
            real(fp), intent(inout) Temperature_AD,
            real(fp), intent(inout) Salinity_AD,
            type(ivar_type), intent(in) iVar )
```

PURPOSE: Subroutine to compute ocean permittivity according to the reference, Liu, Q. et al. (2010) An improved fast microwave water emissivity model. IEEE Trans. Geosci. Remote Sensing, accepted June 25, 2010.

Parameters

in,out	*Temperature_AD*	Adjoint sea surface temperature UNITS: Kelvin(K) TYPE: REAL DIMENSION: Scalar
in,out	*Salinity_AD*	Adjoint water salinity UNITS: ppt (parts per thousand) TYPE: REAL DIMENSION: Scalar

Parameters

in	*Frequency*	Frequency UNITS: N/A TYPE: REAL DIMENSION: Scalar
in,out	*Permittivity_AD*	Adjoint permittivity UNITS: N/A TYPE: COMPLEX DIMENSION: Scalar
in	*iVar*	Composite data structure containing internal variables required for the subsequent tangent-linear and adjoint model calls. The contents of this structure are NOT accessible outside of this module. UNITS: N/A TYPE: iVar_type DIMENSION: Scalar

7.37.2.3 liu_ocean_permittivity_tl()

```
subroutine, public liu::liu_ocean_permittivity_tl (
            real(fp), intent(in) Temperature_TL,
            real(fp), intent(in) Salinity_TL,
            real(fp), intent(in) Frequency,
            complex(fp), intent(out) Permittivity_TL,
            type(ivar_type), intent(in) iVar )
```

PURPOSE: Subroutine to compute ocean permittivity according to the reference, Liu, Q. et al. (2010) An improved fast microwave water emissivity model. IEEE Trans. Geosci. Remote Sensing, accepted June 25, 2010.

Parameters

in	*Temperature_TL*	Tangent-linear sea surface temperature UNITS: Kelvin(K) TYPE: REAL DIMENSION: Scalar
in	*Salinity_TL*	Tangent-linear water salinity UNITS: ppt (parts per thousand) TYPE: REAL DIMENSION: Scalar
in	*Frequency*	Frequency UNITS: N/A TYPE: REAL DIMENSION: Scalar

Parameters

out	*Permittivity_TL*	Tangent-linear permittivity UNITS: N/A TYPE: COMPLEX DIMENSION: Scalar
in	*iVar*	Composite data structure containing internal variables required for the subsequent tangent-linear and adjoint model calls. The contents of this structure are NOT accessible outside of this module. UNITS: N/A TYPE: iVar_type DIMENSION: Scalar

7.38 mod_rttov_fastem5r1_coef Module Reference

Contains data for the FASTEM-4,5,6 MW sea surface emissivity models.

Variables

- real(fp), parameter, public **zero** = 0.0_fp
- real(fp), parameter, public **point_5** = 0.5_fp
- real(fp), parameter, public **one** = 1.0_fp
- real(fp), parameter, public **two** = 2.0_fp
- real(fp), parameter, public **three** = 3.0_fp
- real(fp), parameter, public **pi** = 3.14159265358979323846264338327 9_fp
- real(fp), parameter, public **degrees_to_radians** = PI/180.0_fp
- real(fp), parameter, public **transmittance_limit_lower** = 0.00001_fp
- real(fp), parameter, public **transmittance_limit_upper** = 0.9999_fp
- real(fp), parameter, public **e0_4** = 0.0088419_fp
- real(fp), parameter, public **e0_5** = 0.00885418781762_fp
- real(fp), parameter, public **min_f** = 1.4_fp
- real(fp), parameter, public **max_f** = 200.0_fp
- real(fp), parameter, public **min_wind** = 0.3_fp
- real(fp), parameter, public **max_wind** = 35.0_fp
- real(fp), dimension(0:38), parameter, public **a_coef** = (/ 3.8_fp, 0.0248033_fp, 87.9181727_fp, -0.↩ 4031592248_fp, 0.0009493088010_fp, -0.1930858348E-05_fp, -0.002697_fp, -7.3E-06_fp, -8.9E-06_fp, 5.↩ 723_fp, 0.022379_fp, -0.00071237_fp, -6.28908E-03_fp, 1.76032E-04_fp, -9.22144E-05_fp, 0.1124465_fp, -0.0039815727_fp, 0.00008113381_fp, -0.00000071824242_fp, -2.39357E-03_fp, 3.1353E-05_fp, -2.52477E-07_fp, 0.003049979018_fp, -3.010041629E-05_fp, 0.4811910733E-05_fp, -0.4259775841E-07_fp, 0.149_fp, -8.8E-04_fp, -1.05E-04_fp, 2.033E-02_fp, 1.266E-04_fp, 2.464E-06_fp, -1.849E-05_fp, 2.551E-07_fp, -2.↩ 551E-08_fp, 0.182521_fp, -1.46192E-03_fp, 2.09324E-05_fp, -1.28205E-07_fp/)
- real(fp), dimension(36), parameter, public **lcoef5** = (/ -5.994667E-02_fp, 9.341346E-04_fp,-9.566110E-07_↩ fp, 8.360313E-02_fp,-1.085991E-03_fp, 6.735338E-07_fp,-2.617296E-02_fp, 2.864495E-04_fp,-1.429979E-07_fp,-5.265879E-04_fp, 6.880275E-05_fp,-2.916657E-07_fp,-1.671574E-05_fp, 1.086405E-06_fp,-3.↩ 632227E-09_fp, 1.161940E-04_fp,-6.349418E-05_fp, 2.466556E-07_fp,-2.431811E-02_fp,-1.031810E-03_fp, 4.519513E-06_fp, 2.868236E-02_fp, 1.186478E-03_fp,-5.257096E-06_fp,-7.933390E-03_fp, -2.↩ 422303E-04_fp, 1.089605E-06_fp,-1.083452E-03_fp,-1.788509E-05_fp, 5.464239E-09_fp, -3.855673E-05↩ _fp, 9.360072E-07_fp,-2.639362E-09_fp, 1.101309E-03_fp, 3.599147E-05_fp, -1.043146E-07_fp /)

- real(fp), dimension(36), parameter, public **lcoef4** = (/ -9.197134E-02_fp, 8.310678E-04_fp,-6.065411E-07_↩ fp, 1.350073E-01_fp,-1.032096E-03_fp, 4.259935E-07_fp,-4.373322E-02_fp, 2.545863E-04_fp, 9.835554E-08_fp,-1.199751E-03_fp, 1.360423E-05_fp,-2.088404E-08_fp,-2.201640E-05_fp, 1.951581E-07_fp,-2.↩ 599185E-10_fp, 4.477322E-04_fp,-2.986217E-05_fp, 9.406466E-08_fp,-7.103127E-02_fp,-4.713113E-05_fp, 1.754742E-06_fp, 9.720859E-02_fp, 1.374668E-04_fp,-2.591771E-06_fp,-2.687455E-02_fp, -3.↩ 677779E-05_fp, 7.548377E-07_fp,-3.049506E-03_fp,-5.412826E-05_fp, 2.285387E-07_fp, -2.201640E-05↩ _fp, 1.951581E-07_fp,-2.599185E-10_fp, 2.297488E-03_fp, 3.787032E-05_fp, -1.553581E-07_fp /)
- real(fp), dimension(8), parameter, public **scoef** = (/ -5.0208480E-06_fp, 2.3297951E-08_fp, 4.6625726E-08_fp, -1.9765665E-09_fp, -7.0469823E-04_fp, 7.5061193E-04_fp, 9.8103876E-04_fp, 1.5489504E-04_fp /)
- real(fp), dimension(45), parameter, public **t_c5** = (/ 0.199277E+00_fp, 0.166155E+00_fp, 0.↩ 153272E-01_fp, 0.399234E+01_fp,-0.130968E+01_fp, -0.874716E+00_fp,-0.169403E+01_fp,-0.260998E-01_fp, 0.540443E+00_fp,-0.282483E+00_fp, -0.219994E+00_fp,-0.203438E-01_fp, 0.351731E+00_↩ fp, 0.208641E+01_fp,-0.693299E+00_fp, 0.867861E-01_fp, 0.619020E-01_fp, 0.595251E-02_fp,-0.↩ 475191E+01_fp,-0.430134E-01_fp, 0.248524E+01_fp, 0.388242E-01_fp, 0.194901E+00_fp,-0.425093E-01_fp, 0.607698E+01_fp, -0.313861E+01_fp,-0.103383E+01_fp,-0.377867E+01_fp, 0.180284E+01↩ _fp, 0.699556E+00_fp, -0.506455E-01_fp,-0.262822E+00_fp, 0.703056E-01_fp, 0.362055E+01_↩ fp,-0.120318E+01_fp, -0.124971E+01_fp, 0.154014E-01_fp, 0.759848E-01_fp,-0.268604E-01_fp,-0.↩ 802073E+01_fp, 0.324658E+01_fp, 0.304165E+01_fp, 0.100000E+01_fp, 0.200000E-01_fp, 0.↩ 300000E+00_fp /)
- real(fp), dimension(45), parameter, public **t_c4** = (/ -0.675700E-01_fp, 0.214600E+00_fp,-0.363000E-02_fp, 0.636730E+01_fp, 0.900610E+00_fp, -0.524880E+00_fp,-0.370920E+01_fp,-0.143310E+01↩ _fp, 0.397450E+00_fp, 0.823100E-01_fp, -0.255980E+00_fp, 0.552000E-02_fp, 0.208000E+01_↩ fp, 0.244920E+01_fp,-0.456420E+00_fp, -0.224900E-01_fp, 0.616900E-01_fp,-0.344000E-02_fp,-0.↩ 507570E+01_fp,-0.360670E+01_fp, 0.118750E+01_fp, 0.124950E+00_fp, 0.121270E+00_fp, 0.714000E-02_fp, 0.736620E+01_fp, -0.114060E+00_fp,-0.272910E+00_fp,-0.504350E+01_fp,-0.336450E+00↩ _fp, 0.161260E+00_fp, -0.154290E+00_fp,-0.141070E+00_fp,-0.809000E-02_fp, 0.395290E+01_↩ fp, 0.958580E+00_fp, -0.159080E+00_fp, 0.368500E-01_fp, 0.307100E-01_fp, 0.810000E-03_fp,-0.↩ 619960E+01_fp, -0.172580E+01_fp, 0.641360E+00_fp, 0.100000E+01_fp, 0.200000E-01_fp, 0.↩ 300000E+00_fp /)
- real(fp), dimension(120), parameter, public **b_coef** = (/ 3.307255E-04_fp,-2.901276E-06_fp,-1.475497E-04_fp, 1.288152E-06_fp, 1.004010E-04_fp, -2.671158E-07_fp, 4.363154E-06_fp,-9.817795E-09_fp,-4.↩ 777876E-05_fp, 3.051852E-08_fp, 1.369383E-03_fp,-2.215847E-05_fp,-8.099833E-04_fp, 1.767702E-05_fp,-5.977649E-06_fp, -1.784656E-07_fp,-9.355531E-07_fp, 5.495131E-08_fp,-3.479300E-05_fp,-3.↩ 751652E-07_fp, 2.673536E-04_fp,-1.378890E-06_fp,-8.660113E-05_fp, 2.871488E-07_fp, 1.361118E-05_fp, -1.622586E-08_fp,-1.232439E-07_fp,-3.067416E-09_fp,-1.835366E-06_fp, 8.098728E-09_fp, 1.↩ 255415E-04_fp,-5.145201E-07_fp,-8.832514E-06_fp,-5.105879E-09_fp, 2.734041E-05_fp, -3.398604E-07_fp, 3.417435E-06_fp,-7.043251E-09_fp, 1.497222E-05_fp,-6.832110E-09_fp, -2.315959E-03_fp,-1.↩ 023585E-06_fp, 5.154471E-05_fp, 9.534546E-06_fp,-6.306568E-05_fp, -4.378498E-07_fp,-2.132017E-06_fp, 1.612415E-08_fp,-1.929693E-06_fp,-6.217311E-09_fp, -1.656672E-04_fp, 6.385099E-07_fp, 2.↩ 290074E-06_fp, 1.103787E-07_fp,-5.548757E-06_fp, 5.275966E-08_fp,-4.653774E-07_fp, 1.427566E-09_fp,-3.197232E-06_fp,-4.048557E-09_fp, -1.909801E-04_fp,-3.387963E-07_fp, 4.641319E-05_fp, 4.↩ 502372E-07_fp,-5.055813E-05_fp, 2.104201E-07_fp,-4.121861E-06_fp,-1.633057E-08_fp,-2.469888E-05↩ _fp, 4.492103E-08_fp, -4.582853E-03_fp,-5.373940E-06_fp, 9.713047E-04_fp, 1.783009E-05_fp,-4.↩ 539091E-04_fp, 7.652954E-07_fp,-6.708905E-06_fp, 2.148401E-08_fp, 8.054350E-05_fp, 3.069258E-07_fp, -6.405746E-05_fp,-9.694284E-08_fp, 1.914498E-05_fp, 1.336975E-07_fp,-4.561696E-06_fp, 3.↩ 769169E-08_fp,-6.105244E-07_fp, 2.433761E-10_fp,-3.961735E-06_fp, 1.995636E-08_fp, 1.350148E-06_fp, 3.678149E-07_fp, 1.261701E-05_fp,-2.011440E-07_fp,-2.361347E-05_fp, 2.943147E-08_fp,-1.↩ 304551E-07_fp,-1.119368E-09_fp, 8.469458E-06_fp,-2.292171E-09_fp, 1.419156E-03_fp,-3.838338E-06_fp, 8.222562E-05_fp,-1.106098E-06_fp,-5.482327E-05_fp, 3.083137E-07_fp, 4.418828E-06_fp,-1.↩ 302562E-08_fp, 3.768883E-05_fp,-5.012753E-08_fp, -9.396649E-06_fp, 2.764698E-07_fp, 1.745336E-05_fp,-1.427031E-07_fp,-3.879930E-06_fp, -1.117458E-08_fp, 5.688281E-08_fp, 1.513582E-09_fp, 6.↩ 778764E-06_fp,-7.691286E-09_fp /)
- real(fp), dimension(9), parameter, public **x** = (/ 0.0_fp, 1.4_fp, 6.8_fp, 10.7_fp, 19.35_fp, 37._fp, 89._fp, 150._fp, 200._fp/)
- real(fp), dimension(9), parameter, public **y** = (/ 0.0_fp, 0.1_fp, 0.6_fp, 0.9_fp, 1._fp, 1.0_fp, 0.4_fp, 0.2_fp, 0.0_fp/)
- real(fp), dimension(6, 6, 2), parameter, public **coef_mk_azi** = RESHAPE((/ 4.401E-02, -1.636E+01, 1.↩ 478E+00, -4.800E-02, 3.202E-06, -6.002E-05, 4.379E-02, -1.633E+01, 1.453E+00, -4.176E-02, 5.561E-06, -

4.644E-05, 5.009E-02, -1.638E+01, 1.520E+00, -3.994E-02, 1.330E-05, 1.113E-05, 5.165E-02, -1.638E+01, 1.543E+00, -4.066E-02, 1.494E-05, 1.010E-05, 5.553E-02, -1.638E+01, 1.602E+00, -4.246E-02, 1.903E-05, 7.524E-06, -9.131E-05, 1.251E+00, 6.769E-01, -2.913E-02, 1.092E+00, -1.806E-04, -1.234E-07, -8.179E-03, -1.040E+01, 4.477E-01, 0.000E+00, 3.390E-05, -1.938E-05, -8.007E-03, -1.039E+01, 4.610E-01, 0.000E+00, 4.419E-05, 1.362E-04, -1.013E-03, -9.235E+00, 3.844E-01, 0.000E+00, 2.891E-04, 1.519E-04, -7.865E-04, -9.234E+00, 3.884E-01, 0.000E+00, 6.856E-04, 1.910E-04, -2.224E-04, -9.232E+00, 3.982E-01, 0.000E+00, 1.673E-03, 3.554E-04, 5.226E-04, 9.816E-01, -7.783E-03, 0.000E+00, 2.437E+01 /), (/6,6,2/))

- real(fp), dimension(5), parameter, public **fr_coeff** = (/ 0.07_fp, -1.748e-3_fp, -7.336e-5_fp, 1.044e-7_fp, -0.93_fp /)

7.38.1 Detailed Description

Contains data for the FASTEM-4,5,6 MW sea surface emissivity models.

7.39 mw_canopy_optics Module Reference

Container Module to compute the canopy optical properties at microwave frequencies.

Functions/Subroutines

- subroutine, public crtm_canopymw_optics (lai, leaf_refl, leaf_trans, g, ssalb, tau, iVar)

  PURPOSE: Subroutine to compute the canopy optical properties of land surface at microwave frequencies.
- subroutine, public crtm_canopymw_optics_tl (LAI_TL, ssalb_TL, tau_TL, iVar)

  PURPOSE: Tangent-linear mode of CRTM_CanopyMW_Optics.
- subroutine, public crtm_canopymw_optics_ad (LAI_AD, ssalb_AD, tau_AD, iVar)

  PURPOSE: Adjoint mode of CRTM_CanopyMW_Optics.

7.39.1 Detailed Description

Container Module to compute the canopy optical properties at microwave frequencies.

The canopy optical parametrs are used by the land surface physics-based radiative models to calculate the LAND surface emissivity and reflectivity.

This module is provided to faciliate the integration of canopy optical model codes into the model repository.

7.39.2 Function/Subroutine Documentation

7.39.2.1 crtm_canopymw_optics()

```
subroutine, public mw_canopy_optics::crtm_canopymw_optics (
            real(fp) lai,
            real(fp), dimension(2) leaf_refl,
            real(fp), dimension(2) leaf_trans,
            real(fp), dimension(2) g,
            real(fp), dimension(2) ssalb,
            real(fp), dimension(2) tau,
            type(ivar_type) iVar )
```

PURPOSE: Subroutine to compute the canopy optical properties of land surface at microwave frequencies.

This subroutine is the canopy optical model currently used in the NOAA CRTM

Parameters

in	*lai*	leaf area index UNITS: N/A TYPE: REAL DIMENSION: Scalar
in	*leaf_refl*	Leaf reflectance UNITS: N/A TYPE: REAL DIMENSION: Rank-1
in	*leaf_trans*	Leaf Transmittance UNITS: N/A TYPE: REAL DIMENSION: Rank-1
out	*ssalb*	Canopy Single scattering albedo UNITS: N/A TYPE: REAL DIMENSION: Rank-1
out	*tau*	Canopy optical depth UNITS: N/A TYPE: REAL DIMENSION: Rank-1
out	*g*	symetric parameter UNITS: N/A TYPE: REAL DIMENSION: Rank-1
in,out	*iVar*	Composite data structure containing internal variables required for the subsequent tangent-linear and adjoint model calls. The contents of this structure are NOT accessible outside of this module. UNITS: N/A TYPE: iVar_type DIMENSION: Scalar

7.39.2.2 crtm_canopymw_optics_ad()

```
subroutine, public mw_canopy_optics::crtm_canopymw_optics_ad (
            real(fp) LAI_AD,
            real(fp), dimension(2) ssalb_AD,
            real(fp), dimension(2) tau_AD,
            type(ivar_type) iVar )
```

PURPOSE: Adjoint mode of CRTM_CanopyMW_Optics.

Parameters

in,out	*lai_ad*	leaf area index adjoint UNITS: N/A TYPE: REAL DIMENSION: Scalar
in,out	*ssalb_ad*	Canopy Single scattering albedo adjoint UNITS: N/A TYPE: REAL DIMENSION: Rank-1
in,out	*tau_ad*	Canopy optical depth adjoint UNITS: N/A TYPE: REAL DIMENSION: Rank-1
in,out	*iVar*	Composite data structure containing internal variables required for the subsequent tangent-linear and adjoint model calls. The contents of this structure are NOT accessible outside of this module. UNITS: N/A TYPE: iVar_type DIMENSION: Scalar

7.39.2.3 crtm_canopymw_optics_tl()

```
subroutine, public mw_canopy_optics::crtm_canopymw_optics_tl (
            real(fp) LAI_TL,
            real(fp), dimension(2) ssalb_TL,
            real(fp), dimension(2) tau_TL,
            type(ivar_type) iVar )
```

PURPOSE: Tangent-linear mode of CRTM_CanopyMW_Optics.

Parameters

in	*lai_TL*	leaf area index tangent-linear UNITS: N/A TYPE: REAL DIMENSION: Scalar
out	*ssalb_TL*	Canopy Single scattering albedo tangent-linear UNITS: N/A TYPE: REAL DIMENSION: Rank-1
out	*tau_TL*	Canopy optical depth tangent-linear UNITS: N/A TYPE: REAL DIMENSION: Rank-1

Parameters

in,out	*iVar*	Composite data structure containing internal variables required for the subsequent tangent-linear and adjoint model calls. The contents of this structure are NOT accessible outside of this module. UNITS: N/A TYPE: iVar_type DIMENSION: Scalar

7.40 mw_leaf_optics Module Reference

Container Module to compute the leaf optical properties of LAND surfaces at microwave frequencies.

Functions/Subroutines

- subroutine, public **csem_leafmw_optics** (frequency, angle, mge, refl, trans, eveg, iVar)
- subroutine, public crtm_leafmw_optics (frequency, theta, esv, d, rh, rv, th, tv)

  PURPOSE: Function to calculate v-pol and h-pol refelectance and trasmittance of one single leaf at microwave frequency.
- subroutine, public mean_leafmw_optics (frequency, eveg, leaf_thick, rh, rv, th, tv)

  PURPOSE: Function to calculate averaged refelectance and trasmittance of one single leaf at microwave frequency. Leaves are taken as individual scatters of a canopy. The averaged refelectance and trasmittance is used by canopy-level scattering model.

7.40.1 Detailed Description

Container Module to compute the leaf optical properties of LAND surfaces at microwave frequencies.

The leaf optical parametrs are used by the canopy radiative models to calculate the canopy optical parameters.

This module is provided to faciliate the integration of leaf optical model codes into the model repository.

7.40.2 Function/Subroutine Documentation

7.40.2.1 crtm_leafmw_optics()

```
subroutine, public mw_leaf_optics::crtm_leafmw_optics (
             real(fp), intent(in) frequency,
             real(fp), intent(in) theta,
             complex(fp), intent(in) esv,
             real(fp), intent(in) d,
             real(fp), intent(out) rh,
             real(fp), intent(out) rv,
             real(fp), intent(out) th,
             real(fp), intent(out) tv )
```

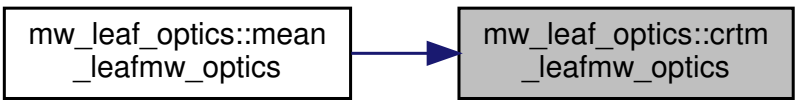
PURPOSE: Function to calculate v-pol and h-pol refelectance and trasmittance of one single leaf at microwave frequency.

This function is the default leaf optic model currently used by NOAA CRTM

Parameters

in	*frequency*	frequency UNITS: GHz TYPE: REAL DIMENSION: Scalar
in	*theta*	incident angle in degree UNITS: N/A TYPE: REAL DIMENSION: Scalar
in	*d*	leaf thicknesss UNITS: mm TYPE: REAL DIMENSION: Scalar
in	*esv*	leaf bulk dielectric constant UNITS: N/A TYPE: COMPLEX DIMENSION: Scalar
out	*rh*	leaf refelectance of h-pol UNITS: N/A TYPE: REAL DIMENSION: Scalar
out	*rv*	leaf refelectance of v-pol UNITS: N/A TYPE: REAL DIMENSION: Scalar
out	*th*	leaf trasmittance of h-pol UNITS: N/A TYPE: REAL DIMENSION: Scalar
out	*tv*	leaf trasmittance of v-pol UNITS: N/A TYPE: REAL DIMENSION: Scalar

Here is the caller graph for this function:



7.40.2.2 mean_leafmw_optics()

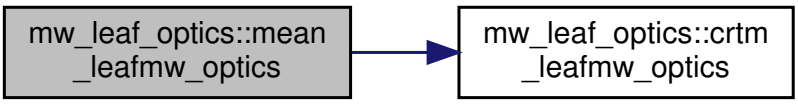
```
subroutine, public mw_leaf_optics::mean_leafmw_optics (
             real(fp), intent(in) frequency,
             complex(fp), intent(in) eveg,
             real(fp), intent(in) leaf_thick,
             real(fp), intent(out) rh,
             real(fp), intent(out) rv,
             real(fp), intent(out) th,
             real(fp), intent(out) tv )
```

PURPOSE: Function to calculate averaged refelectance and trasmittance of one single leaf at microwave frequency. Leaves are taken as individual scatters of a canopy. The averaged refelectance and trasmittance is used by canopy-level scattering model.

Parameters

in	*frequency*	frequency UNITS: GHz TYPE: REAL DIMENSION: Scalar
in	*leaf_thick*	leaf thicknesss UNITS: mm TYPE: REAL DIMENSION: Scalar
in	*eveg*	leaf bulk dielectric constant UNITS: N/A TYPE: COMPLEX DIMENSION: Scalar
out	*rh*	leaf refelectance of h-pol UNITS: N/A TYPE: REAL DIMENSION: Scalar
out	*rv*	leaf refelectance of v-pol UNITS: N/A TYPE: REAL DIMENSION: Scalar
out	*th*	leaf trasmittance of h-pol UNITS: N/A TYPE: REAL DIMENSION: Scalar
out	*tv*	leaf trasmittance of v-pol UNITS: N/A TYPE: REAL DIMENSION: Scalar

Here is the call graph for this function:



7.41 mw_soil_optics Module Reference

Container module with all the MW soil models available in the CSEM model repository.

Functions/Subroutines

- subroutine, public csem_soilmw_optics (frequency, theta, Tskin, Tsoil, smc, sand, clay, refl_smooth, teff, iVar)

  PURPOSE: Evaluation of the bare-soil optical parameters at microwave frequencies.
- subroutine, public csem_soilmw_optics_tl (Tskin_TL, Tsoil_TL, smc_TL, refl_h_TL, refl_v_TL, teff_TL, iVar)

  PURPOSE: Tangent-linear mode of CSEM_SoilMW_Optics.
- subroutine, public csem_soilmw_optics_ad (Tskin_AD, Tsoil_AD, smc_AD, refl_h_AD, refl_v_AD, teff_AD, i↩ Var)

  PURPOSE: Tangent-linear mode of CSEM_SoilMW_Optics.

Variables

- integer, parameter, public **max_soil_layers** = 1

7.41.1 Detailed Description

Container module with all the MW soil models available in the CSEM model repository.

The surface emissivity and reflectivity are required to determine the surface radiative contribution to the overall atmosphere radiative transfer system. This module is designed as a container to implement different microwave bare soil surface radiative transfer models in the CSEM package. It also provides a generic interface for the upper-level applications to access all the available MW soil models. Each individual model has the FWD(Forward), TL(Tangent-linear) and AD(Adjoint) functions for the variational data assimilation and the surface parameter retrieval applications.

7.41.2 Function/Subroutine Documentation

7.41.2.1 csem_soilmw_optics()

```
subroutine, public mw_soil_optics::csem_soilmw_optics (
             real(fp), intent(in) frequency,
             real(fp), intent(in) theta,
             real(fp), intent(in) Tskin,
             real(fp), dimension(:), intent(in) Tsoil,
             real(fp), dimension(:), intent(in) smc,
             real(fp), intent(in) sand,
             real(fp), intent(in) clay,
             real(fp), dimension(2), intent(out) refl_smooth,
             real(fp), intent(out) teff,
             type(ivar_type) iVar )
```

PURPOSE: Evaluation of the bare-soil optical parameters at microwave frequencies.

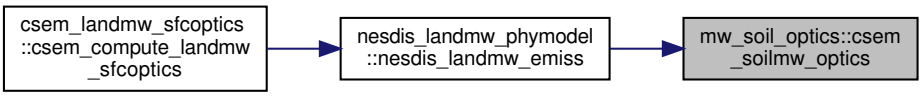
Parameters

in	*frequency*	frequency UNITS: GHz TYPE: REAL DIMENSION: Scalar
in	*theta*	Angle in degree UNITS: Degree TYPE: REAL DIMENSION: Scalar
in	*tskin*	soil surface temperature UNITS: Kelvin,K TYPE: REAL DIMENSION: Scalar
in	*tsoil*	soil temperature profile UNITS: Kelvin,K TYPE: REAL DIMENSION: Rank-1
in	*smc*	soil moisture content UNITS: N/A TYPE: REAL DIMENSION: Rank-1
in	*sand*	soil texture sand fraction UNITS: N/A TYPE: REAL DIMENSION: Scalar
in	*clay*	soil texture clay fraction UNITS: N/A TYPE: REAL DIMENSION: Scalar
out	*refl_smooth*	soil surface reflectance UNITS: N/A TYPE: REAL DIMENSION: Rank-1
out	*teff*	soil layer effective temperature UNITS: N/A TYPE: REAL DIMENSION: Scalar

Parameters

in,out	*iVar*	Composite data structure containing internal variables required for the subsequent tangent-linear and adjoint model calls. The contents of this structure are NOT accessible outside of this module. UNITS: N/A TYPE: iVar_type DIMENSION: Scalar

Here is the caller graph for this function:



7.41.2.2 csem_soilmw_optics_ad()

```
subroutine, public mw_soil_optics::csem_soilmw_optics_ad (
             real(fp), intent(inout) Tskin_AD,
             real(fp), intent(inout) Tsoil_AD,
             real(fp), intent(inout) smc_AD,
             real(fp), intent(inout) refl_h_AD,
             real(fp), intent(inout) refl_v_AD,
             real(fp), intent(inout) teff_AD,
             type(ivar_type) iVar )
```

PURPOSE: Tangent-linear mode of CSEM_SoilMW_Optics.

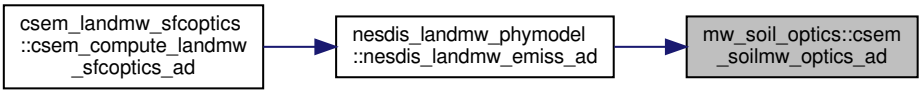
Parameters

in,out	*tskin_AD*	soil surface temperature adjoint UNITS: Kelvin,K TYPE: REAL DIMENSION: Scalar
in,out	*tsoil_AD*	soil temperature profile adjoint UNITS: Kelvin,K TYPE: REAL DIMENSION: Scalar
in,out	*smc_AD*	soil moisture content adjoint UNITS: N/A TYPE: REAL DIMENSION: Scalar
in,out	*refl_h_AD*	H-pol soil surface reflectance adjoint UNITS: N/A TYPE: REAL DIMENSION: Scalar

Parameters

in,out	*refl_v_AD*	V-pol soil surface reflectance adjoint UNITS: N/A TYPE: REAL DIMENSION: Scalar
in,out	*teff_AD*	Effective soil layer temperature adjoint UNITS: Kelvin,K TYPE: REAL DIMENSION: Scalar
in,out	*iVar*	Composite data structure containing internal variables required for the subsequent tangent-linear and adjoint model calls. The contents of this structure are NOT accessible outside of this module. UNITS: N/A TYPE: iVar_type DIMENSION: Scalar

Here is the caller graph for this function:



7.41.2.3 csem_soilmw_optics_tl()

```
subroutine, public mw_soil_optics::csem_soilmw_optics_tl (
            real(fp), intent(in) Tskin_TL,
            real(fp), intent(in) Tsoil_TL,
            real(fp), intent(in) smc_TL,
            real(fp), intent(out) refl_h_TL,
            real(fp), intent(out) refl_v_TL,
            real(fp), intent(out) teff_TL,
            type(ivar_type) iVar )
```

PURPOSE: Tangent-linear mode of CSEM_SoilMW_Optics.

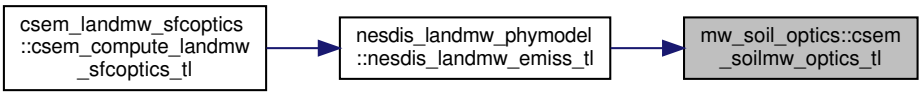
Parameters

in	*tskin_TL*	soil surface temperature tangent-linear UNITS: Kelvin,K TYPE: REAL DIMENSION: Scalar
in	*tsoil_TL*	soil temperature profile tangent-linear UNITS: Kelvin,K TYPE: REAL DIMENSION: Scalar

Parameters

in	*smc_TL*	soil moisture content tangent-linear UNITS: N/A TYPE: REAL DIMENSION: Scalar
out	*refl_h_TL*	H-pol soil surface reflectance tangent-linear UNITS: N/A TYPE: REAL DIMENSION: Scalar
out	*refl_v_TL*	V-pol soil surface reflectance tangent-linear UNITS: N/A TYPE: REAL DIMENSION: Scalar
out	*teff_TL*	Effective soil layer temperature tangent-linear UNITS: Kelvin,K TYPE: REAL DIMENSION: Scalar
in,out	*iVar*	Composite data structure containing internal variables required for the subsequent tangent-linear and adjoint model calls. The contents of this structure are NOT accessible outside of this module. UNITS: N/A TYPE: iVar_type DIMENSION: Scalar

Here is the caller graph for this function:



7.42 mw_soil_permittivity Module Reference

Module to compute the soil dielectric properties for LAND surfaces at microwave frequencies.

Functions/Subroutines

- subroutine, public **csem_soilmw_permittivity** (frequency, tsoil, smc, sand, clay, eps, ivar)
- subroutine, public **csem_soilmw_permittivity_tl** (tsoil_TL, smc_TL, eps_TL, iVar)
- subroutine, public **csem_soilmw_permittivity_ad** (tsoil_AD, smc_AD, eps_AD, iVar)

7.42.1 Detailed Description

Module to compute the soil dielectric properties for LAND surfaces at microwave frequencies.

7.43 nesdis_amsre_iceem_module Module Reference

Module containing the AMSR-E microwave sea ice emissivity model.

Functions/Subroutines

- subroutine, public **nesdis_amsre_iceem** (frequency, User_Angle, tv, th, Ts, Tice, Emissivity_H, Emissivity_V)

Variables

- integer, parameter, public **n_freq** = 7
- real(fp), dimension(n_freq), parameter, public **frequency_amsrealg** = (/ 6.925_fp, 10.65_fp, 18.7_fp,23.8_fp, 36.5_fp, 89.0_fp,157._fp/)
- real(fp), dimension(n_freq), parameter, public **rs_ice_a_emiss** =(/0.93_fp, 0.94_fp, 0.96_fp, 0.97_fp, 0.97_fp, 0.94_fp, 0.93_fp/)
- real(fp), dimension(n_freq), parameter, public **rs_ice_b_emiss** =(/0.86_fp, 0.87_fp, 0.90_fp, 0.91_fp, 0.90_fp, 0.90_fp, 0.89_fp/)
- real(fp), dimension(n_freq), parameter, public **mixed_newice_snow_emiss** =(/0.88_fp, 0.88_fp, 0.89_fp, 0.88_fp, 0.87_fp, 0.84_fp, 0.82_fp/)
- real(fp), dimension(n_freq), parameter, public **nare_newice_emiss** =(/0.80_fp, 0.81_fp, 0.81_fp, 0.81_fp, 0.80_fp, 0.79_fp, 0.79_fp/)
- real(fp), dimension(n_freq), parameter, public **broken_ice_emiss** =(/0.75_fp, 0.78_fp, 0.80_fp, 0.81_fp, 0.80_fp, 0.77_fp, 0.74_fp/)
- real(fp), dimension(n_freq), parameter, public **first_year_ice_emiss** =(/0.93_fp, 0.93_fp, 0.92_fp, 0.92_fp, 0.89_fp, 0.78_fp, 0.69_fp/)
- real(fp), dimension(n_freq), parameter, public **composite_pack_ice_emiss** =(/0.89_fp, 0.88_fp, 0.87_fp, 0.85_fp, 0.82_fp, 0.69_fp, 0.59_fp/)
- real(fp), dimension(n_freq), parameter, public **rs_ice_c_emiss** =(/0.92_fp, 0.90_fp, 0.83_fp, 0.78_fp, 0.73_fp, 0.62_fp, 0.58_fp/)
- real(fp), dimension(n_freq), parameter, public **fast_ice_emiss** =(/0.85_fp, 0.85_fp, 0.84_fp, 0.81_fp, 0.78_fp, 0.63_fp, 0.56_fp/)
- real(fp), dimension(n_freq), parameter, public **rs_ice_d_emiss** =(/0.76_fp, 0.76_fp, 0.76_fp, 0.76_fp, 0.74_fp, 0.65_fp, 0.60_fp/)
- real(fp), dimension(n_freq), parameter, public **rs_ice_e_emiss** =(/0.63_fp, 0.65_fp, 0.67_fp, 0.68_fp, 0.70_fp, 0.74_fp, 0.75_fp/)
- real(fp), dimension(n_freq), parameter, public **rs_ice_f_emiss** =(/0.54_fp, 0.60_fp, 0.64_fp, 0.67_fp, 0.70_fp, 0.71_fp, 0.72_fp/)
- real(fp), dimension(n_freq), parameter, public **grease_ice_emiss** =(/0.49_fp, 0.51_fp, 0.53_fp, 0.55_fp, 0.58_fp, 0.65_fp, 0.67_fp/)
- real(fp), dimension(n_freq), parameter, public **rs_ice_a_ev** =(/ 0.96_fp, 0.97_fp, 0.99_fp, 0.99_fp, 0.99_fp, 0.98_fp, 0.97_fp/)
- real(fp), dimension(n_freq), parameter, public **rs_ice_b_ev** =(/0.95_fp, 0.96_fp, 0.99_fp, 0.98_fp, 0.97_fp, 0.94_fp, 0.93_fp/)
- real(fp), dimension(n_freq), parameter, public **mixed_newice_snow_ev** =(/0.96_fp, 0.96_fp, 0.95_fp, 0.94_fp, 0.93_fp, 0.88_fp, 0.86_fp/)
- real(fp), dimension(n_freq), parameter, public **nare_newice_ev** =(/0.88_fp, 0.89_fp, 0.91_fp, 0.91_fp, 0.91_fp, 0.88_fp, 0.88_fp/)
- real(fp), dimension(n_freq), parameter, public **broken_ice_ev** =(/0.85_fp, 0.87_fp, 0.91_fp, 0.91_fp, 0.91_fp, 0.87_fp, 0.84_fp/)
- real(fp), dimension(n_freq), parameter, public **first_year_ice_ev** =(/0.98_fp, 0.98_fp, 0.98_fp, 0.97_fp, 0.95_fp, 0.84_fp, 0.75_fp/)

- real(fp), dimension(n_freq), parameter, public **composite_pack_ice_ev** =(/0.98_fp, 0.97_fp, 0.95_fp, 0.93↩ _fp, 0.89_fp, 0.72_fp, 0.62_fp/)
- real(fp), dimension(n_freq), parameter, public **rs_ice_c_ev** =(/0.99_fp, 0.96_fp, 0.90_fp, 0.86_fp, 0.75_fp, 0.66_fp, 0.62_fp/)
- real(fp), dimension(n_freq), parameter, public **fast_ice_ev** =(/0.95_fp, 0.95_fp, 0.94_fp, 0.91_fp, 0.85_fp, 0.69_fp, 0.62_fp/)
- real(fp), dimension(n_freq), parameter, public **rs_ice_d_ev** =(/0.87_fp, 0.87_fp, 0.88_fp, 0.88_fp, 0.88_fp, 0.77_fp, 0.72_fp/)
- real(fp), dimension(n_freq), parameter, public **rs_ice_e_ev** =(/0.77_fp, 0.78_fp, 0.81_fp, 0.82_fp, 0.84_fp, 0.86_fp, 0.88_fp/)
- real(fp), dimension(n_freq), parameter, public **rs_ice_f_ev** =(/0.71_fp, 0.73_fp, 0.77_fp, 0.78_fp, 0.81_fp, 0.86_fp, 0.87_fp/)
- real(fp), dimension(n_freq), parameter, public **grease_ice_ev** =(/0.66_fp, 0.67_fp, 0.70_fp, 0.72_fp, 0.76_fp, 0.82_fp, 0.84_fp/)
- real(fp), dimension(n_freq), parameter, public **rs_ice_a_eh** =(/ 0.88_fp, 0.92_fp, 0.94_fp, 0.94_fp, 0.95_fp, 0.92_fp, 0.91_fp/)
- real(fp), dimension(n_freq), parameter, public **rs_ice_b_eh** =(/0.81_fp, 0.82_fp, 0.85_fp, 0.86_fp, 0.87_fp, 0.88_fp, 0.87_fp/)
- real(fp), dimension(n_freq), parameter, public **mixed_newice_snow_eh** =(/0.83_fp, 0.84_fp, 0.86_fp, 0.↩ 85_fp, 0.84_fp, 0.82_fp, 0.80_fp/)
- real(fp), dimension(n_freq), parameter, public **nare_newice_eh** =(/0.74_fp, 0.75_fp, 0.76_fp, 0.76_fp, 0.↩ 77_fp, 0.73_fp, 0.73_fp/)
- real(fp), dimension(n_freq), parameter, public **broken_ice_eh** =(/0.71_fp, 0.73_fp, 0.76_fp, 0.77_fp, 0.80_fp, 0.72_fp, 0.69_fp/)
- real(fp), dimension(n_freq), parameter, public **first_year_ice_eh** =(/0.91_fp, 0.90_fp, 0.89_fp, 0.88_fp, 0.↩ 86_fp, 0.76_fp, 0.67_fp/)
- real(fp), dimension(n_freq), parameter, public **composite_pack_ice_eh** =(/0.85_fp, 0.84_fp, 0.83_fp, 0.82↩ _fp, 0.79_fp, 0.67_fp, 0.57_fp/)
- real(fp), dimension(n_freq), parameter, public **rs_ice_c_eh** =(/0.90_fp, 0.87_fp, 0.81_fp, 0.78_fp, 0.69_fp, 0.60_fp, 0.56_fp/)
- real(fp), dimension(n_freq), parameter, public **fast_ice_eh** =(/0.80_fp, 0.80_fp, 0.78_fp, 0.76_fp, 0.72_fp, 0.60_fp, 0.53_fp/)
- real(fp), dimension(n_freq), parameter, public **rs_ice_d_eh** =(/0.71_fp, 0.71_fp, 0.70_fp, 0.70_fp, 0.70_fp, 0.59_fp, 0.54_fp/)
- real(fp), dimension(n_freq), parameter, public **rs_ice_e_eh** =(/0.55_fp, 0.59_fp, 0.60_fp, 0.61_fp, 0.62_fp, 0.67_fp, 0.69_fp/)
- real(fp), dimension(n_freq), parameter, public **rs_ice_f_eh** =(/0.48_fp, 0.51_fp, 0.56_fp, 0.57_fp, 0.60_fp, 0.64_fp, 0.65_fp/)
- real(fp), dimension(n_freq), parameter, public **grease_ice_eh** =(/0.42_fp, 0.42_fp, 0.43_fp, 0.45_fp, 0.49_fp, 0.54_fp, 0.56_fp/)

7.43.1 Detailed Description

Module containing the AMSR-E microwave sea ice emissivity model.

7.44 nesdis_amsre_snowem_module Module Reference

Module containing the AMSR-E microwave snow emissivity model.

Functions/Subroutines

- subroutine, public **nesdis_amsre_snowem** (Frequency, User_Angle, tv, th, Ts, Tsnow, Emissivity_H, Emissivity_V)

7.44.1 Detailed Description

Module containing the AMSR-E microwave snow emissivity model.

7.45 nesdis_amsu_iceem_module Module Reference

Module containing the AMSU microwave sea ice emissivity model.

Functions/Subroutines

- subroutine, public **nesdis_amsu_iceem** (Satellite_Angle, User_Angle, frequency, Ts, tba, tbb, Emissivity_H, Emissivity_V)

7.45.1 Detailed Description

Module containing the AMSU microwave sea ice emissivity model.

7.46 nesdis_amsu_snowem_module Module Reference

Module containing the AMSU microwave snow emissivity model.

Functions/Subroutines

- subroutine, public **nesdis_amsu_snowem** (Satellite_Angle, User_Angle, frequency, Snow_Depth, Ts, tba, tbb, Emissivity_H, Emissivity_V)

7.46.1 Detailed Description

Module containing the AMSU microwave snow emissivity model.

7.47 nesdis_atms_iceem_module Module Reference

NESDIS_ATMS_SeaICE_LIB Module to implement the library-based seaIce emissivity model.

Functions/Subroutines

- subroutine, public **nesdis_atms_seaice** (Satellite_Angle, User_Angle, frequency, Ts, Tbs, Emissivity_H, Emissivity_V)

7.47.1 Detailed Description

NESDIS_ATMS_SeaICE_LIB Module to implement the library-based seaIce emissivity model.

7.48 nesdis_atms_seaice_lib Module Reference

Module containing the snow emissivity library ATMS channels.

Functions/Subroutines

- integer function **seaicetype_name2index** (sname)
- character(len=100) function **seaicetype_index2name** (sindex)

Variables

- integer(ip), parameter, public **n_freq_atms** = 13
- integer(ip), parameter, public **n_seaice_types** = 13
- integer(ip), parameter, public **n_freq_amsre** = 7
- integer(ip), parameter, public **invalid_seaice_type** = -1
- character(len=20), dimension(n_seaice_types), parameter, public **seaice_type_names** =(/ 'RS_ICE_A↩ _EMISS ', 'RS_ICE_B_EMISS ', 'MIXED_NEWICE_SNOW_EM', 'NARE_NEWICE_EMISS ', 'BROKEN_↩ ICE_EMISS ', 'FIRST_YEAR_ICE_EMISS', 'COMPOSITE_PACK_ICE ', 'RS_ICE_C_EMISS ', 'FAST_ICE↩ _EMISS ', 'RS_ICE_D_EMISS ', 'RS_ICE_E_EMISS ', 'RS_ICE_F_EMISS ', 'GREASE_ICE_EMISS '/)
- real(fp), dimension(n_freq_atms), parameter, public **frequency_atms** = (/23.80_fp,31.40_fp,50.30_fp,51.↩ 76_fp,52.80_fp,53.60_fp,54.40_fp, 54.90_fp,55.50_fp,57.30_fp,88.20_fp,165.50_fp,183.30_fp/)
- real(fp), dimension(n_freq_atms, n_seaice_types), parameter, public **seaice_emiss_atms_h** = RESHAPE((/ 0.94_fp,0.95_fp,0.94_fp,0.94_fp,0.94_fp,0.94_fp,0.94_fp,0.94_fp,0.94_fp,0.92_fp,0.91_fp,0.91_fp, 0.86_fp,0.87_fp,0.87_fp,0.87_fp,0.87_fp,0.87_fp,0.87_fp,0.87_fp,0.87_fp,0.87_fp,0.88_fp,0.87_fp,0.87_fp, 0.85_fp,0.84_fp,0.83_fp,0.83_fp,0.83_fp,0.83_fp,0.83_fp,0.83_fp,0.83_fp,0.83_fp,0.82_fp,0.80_fp,0.79_fp, 0.76_fp,0.77_fp,0.76_fp,0.76_fp,0.76_fp,0.76_fp,0.76_fp,0.76_fp,0.76_fp,0.75_fp,0.73_fp,0.73_fp,0.73_fp, 0.77_fp,0.79_fp,0.78_fp,0.78_fp,0.78_fp,0.77_fp,0.77_fp,0.77_fp,0.77_fp,0.77_fp,0.72_fp,0.69_fp,0.68_fp, 0.88_fp,0.87_fp,0.83_fp,0.83_fp,0.83_fp,0.83_fp,0.83_fp,0.82_fp,0.82_fp,0.82_fp,0.76_fp,0.66_fp,0.64_fp, 0.82_fp,0.80_fp,0.76_fp,0.76_fp,0.75_fp,0.75_fp,0.75_fp,0.75_fp,0.75_fp,0.74_fp,0.67_fp,0.56_fp,0.53_fp, 0.78_fp,0.73_fp,0.67_fp,0.66_fp,0.66_fp,0.66_fp,0.66_fp,0.66_fp,0.66_fp,0.65_fp,0.60_fp,0.56_fp,0.54_fp, 0.76_fp,0.74_fp,0.69_fp,0.69_fp,0.68_fp,0.68_fp,0.68_fp,0.68_fp,0.68_fp,0.67_fp,0.60_fp,0.52_fp,0.50_fp, 0.70_fp,0.70_fp,0.67_fp,0.67_fp,0.67_fp,0.66_fp,0.66_fp,0.66_fp,0.66_fp,0.66_fp,0.59_fp,0.53_fp,0.52_fp, 0.61_fp,0.62_fp,0.63_fp,0.63_fp,0.64_fp,0.64_fp,0.64_fp,0.64_fp,0.64_fp,0.64_fp,0.67_fp,0.69_fp,0.70_fp, 0.57_fp,0.59_fp,0.61_fp,0.61_fp,0.61_fp,0.61_fp,0.61_fp,0.61_fp,0.61_fp,0.62_fp,0.64_fp,0.65_fp,0.65_fp, 0.45_fp,0.47_fp,0.50_fp,0.50_fp,0.51_fp,0.51_fp,0.51_fp,0.51_fp,0.51_fp,0.51_fp,0.54_fp,0.56_fp,0.57_fp /), (/N_FREQ_ATMS,N_SEAICE_TYPES/))
- real(fp), dimension(n_freq_atms, n_seaice_types), parameter, public **seaice_emiss_atms_v** = RESHAPE((/ 0.99_fp,0.99_fp,0.99_fp,0.99_fp,0.99_fp,0.99_fp,0.99_fp,0.99_fp,0.99_fp,0.98_fp,0.97_fp,0.97_fp, 0.98_fp,0.97_fp,0.96_fp,0.96_fp,0.96_fp,0.96_fp,0.96_fp,0.96_fp,0.96_fp,0.96_fp,0.94_fp,0.93_fp,0.93_fp, 0.94_fp,0.93_fp,0.92_fp,0.92_fp,0.91_fp,0.91_fp,0.91_fp,0.91_fp,0.91_fp,0.91_fp,0.88_fp,0.86_fp,0.85_fp, 0.91_fp,0.91_fp,0.90_fp,0.90_fp,0.90_fp,0.90_fp,0.90_fp,0.90_fp,0.90_fp,0.90_fp,0.88_fp,0.88_fp,0.88_fp, 0.91_fp,0.91_fp,0.90_fp,0.90_fp,0.90_fp,0.90_fp,0.90_fp,0.90_fp,0.90_fp,0.89_fp,0.87_fp,0.84_fp,0.83_fp, 0.97_fp,0.96_fp,0.92_fp,0.92_fp,0.92_fp,0.91_fp,0.91_fp,0.91_fp,0.91_fp,0.91_fp,0.84_fp,0.74_fp,0.72_fp, 0.93_fp,0.91_fp,0.85_fp,0.84_fp,0.84_fp,0.83_fp,0.83_fp,0.83_fp,0.83_fp,0.82_fp,0.72_fp,0.61_fp,0.58_fp, 0.86_fp,0.79_fp,0.73_fp,0.72_fp,0.72_fp,0.72_fp,0.72_fp,0.72_fp,0.72_fp,0.71_fp,0.66_fp,0.62_fp,0.60_fp, 0.91_fp,0.87_fp,0.81_fp,0.80_fp,0.80_fp,0.80_fp,0.80_fp,0.79_fp,0.79_fp,0.79_fp,0.69_fp,0.61_fp,0.59_fp, 0.88_fp,0.88_fp,0.85_fp,0.85_fp,0.85_fp,0.84_fp,0.84_fp,0.84_fp,0.84_fp,0.84_fp,0.77_fp,0.71_fp,0.70_fp, 0.82_fp,0.83_fp,0.85_fp,0.85_fp,0.85_fp,0.85_fp,0.85_fp,0.85_fp,0.85_fp,0.85_fp,0.86_fp,0.88_fp,0.89_fp, 0.78_fp,0.80_fp,0.82_fp,0.82_fp,0.83_fp,0.83_fp,0.83_fp,0.83_fp,0.83_fp,0.83_fp,0.86_fp,0.87_fp,0.87_fp, 0.72_fp,0.74_fp,0.78_fp,0.78_fp,0.78_fp,0.78_fp,0.78_fp,0.78_fp,0.78_fp,0.78_fp,0.82_fp,0.84_fp,0.85_fp /), (/N_FREQ_ATMS,N_SEAICE_TYPES/))

- real(fp), dimension(n_freq_atms, n_seaice_types), parameter, public **seaice_emiss_atms_lib** = RESHAPE((/ 0.97_fp,0.97_fp,0.96_fp,0.96_fp,0.96_fp,0.96_fp,0.96_fp,0.96_fp,0.96_fp,0.95_fp,0.94_↩ fp,0.94_fp, 0.92_fp,0.92_fp,0.92_fp,0.92_fp,0.92_fp,0.92_fp,0.92_fp,0.92_fp,0.92_fp,0.91_fp,0.90_↩ fp,0.90_fp, 0.90_fp,0.89_fp,0.88_fp,0.87_fp,0.87_fp,0.87_fp,0.87_fp,0.87_fp,0.87_fp,0.85_fp,0.83_↩ fp,0.82_fp, 0.84_fp,0.84_fp,0.83_fp,0.83_fp,0.83_fp,0.83_fp,0.83_fp,0.83_fp,0.83_fp,0.81_fp,0.80_↩ fp,0.80_fp, 0.84_fp,0.85_fp,0.84_fp,0.84_fp,0.84_fp,0.84_fp,0.83_fp,0.83_fp,0.83_fp,0.83_fp,0.80_fp,0.76_↩ fp,0.75_fp, 0.93_fp,0.91_fp,0.88_fp,0.87_fp,0.87_fp,0.87_fp,0.87_fp,0.87_fp,0.87_fp,0.86_fp,0.80_fp,0.70_↩ fp,0.68_fp, 0.88_fp,0.85_fp,0.80_fp,0.80_fp,0.79_fp,0.79_fp,0.79_fp,0.79_fp,0.79_fp,0.78_fp,0.70_fp,0.58_↩ fp,0.56_fp, 0.82_fp,0.76_fp,0.70_fp,0.69_fp,0.69_fp,0.69_fp,0.69_fp,0.69_fp,0.69_fp,0.68_fp,0.63_fp,0.59_↩ fp,0.57_fp, 0.84_fp,0.81_fp,0.75_fp,0.74_fp,0.74_fp,0.74_fp,0.74_fp,0.74_fp,0.73_fp,0.73_fp,0.65_fp,0.57_↩ fp,0.55_fp, 0.79_fp,0.79_fp,0.76_fp,0.76_fp,0.76_fp,0.75_fp,0.75_fp,0.75_fp,0.75_fp,0.75_fp,0.68_fp,0.62_↩ fp,0.61_fp, 0.72_fp,0.72_fp,0.74_fp,0.74_fp,0.74_fp,0.74_fp,0.74_fp,0.74_fp,0.74_fp,0.74_fp,0.76_fp,0.79_↩ fp,0.79_fp, 0.68_fp,0.69_fp,0.72_fp,0.72_fp,0.72_fp,0.72_fp,0.72_fp,0.72_fp,0.72_fp,0.72_fp,0.75_fp,0.76_↩ fp,0.76_fp, 0.59_fp,0.61_fp,0.64_fp,0.64_fp,0.64_fp,0.64_fp,0.64_fp,0.64_fp,0.64_fp,0.65_fp,0.68_fp,0.70_↩ fp,0.71_fp /), (/N_FREQ_ATMS,N_SEAICE_TYPES/))

7.48.1 Detailed Description

Module containing the snow emissivity library ATMS channels.

7.49 nesdis_atms_snowem_module Module Reference

NESDIS_SnowEM_ATMS_Parameters Module to implement the library-based snow emissivity model.

Functions/Subroutines

- subroutine, public **nesdis_atms_snowem** (Satellite_Angle, User_Angle, Frequency, Tbs, Tss, Snow_Depth, Emissivity_H, Emissivity_V)

7.49.1 Detailed Description

NESDIS_SnowEM_ATMS_Parameters Module to implement the library-based snow emissivity model.

7.50 nesdis_iceir_phymodel Module Reference

Module containing the NESDIS physical Ice emissivity model of infrared Channels.

Functions/Subroutines

- integer function, public **nesdis_iceir_emiss** (Frequency, Angle, Ice_Temperature, Emissivity_H, Emissivity↩ _V)

7.50.1 Detailed Description

Module containing the NESDIS physical Ice emissivity model of infrared Channels.

7.51 nesdis_icemw_phymodel Module Reference

Module containing the NESDIS physical Ice emissivity model of microwave Channels.

Functions/Subroutines

- integer function, public **nesdis_icemw_emiss** (Frequency, Angle, Ice_Temperature, Salinity, Emissivity_H, Emissivity_V)

7.51.1 Detailed Description

Module containing the NESDIS physical Ice emissivity model of microwave Channels.

7.52 nesdis_icevis_phymodel Module Reference

Module containing the NESDIS physical Ice emissivity model of visible channels.

Functions/Subroutines

- integer function, public **nesdis_icevis_emiss** (Frequency, Angle, Ice_Temperature, Emissivity_H, Emissivity_V)

7.52.1 Detailed Description

Module containing the NESDIS physical Ice emissivity model of visible channels.

7.53 nesdis_landem_module Module Reference

Module containing the old-version NESDIS microwave land emissivity model.

Functions/Subroutines

- subroutine, public nesdis_landem_213 (Angle, Frequency, Soil_Moisture_Content, Vegetation_Fraction, Soil_Temperature, t_skin, Lai, Soil_Type, Vegetation_Type, Emissivity_H, Emissivity_V)
- subroutine, public **nesdis_landem_old** (Angle, Frequency, Soil_Moisture_Content, Vegetation_Fraction, Soil_Temperature, t_skin, Lai, Soil_Type, Vegetation_Type, Emissivity_H, Emissivity_V)

7.53.1 Detailed Description

Module containing the old-version NESDIS microwave land emissivity model.

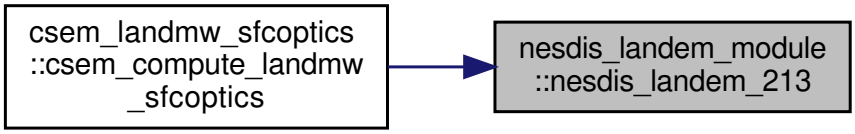
7.53.2 Function/Subroutine Documentation

7.53.2.1 nesdis_landem_213()

```
subroutine, public nesdis_landem_module::nesdis_landem_213 (
             real(fp), intent(in) Angle,
             real(fp), intent(in) Frequency,
             real(fp), intent(in) Soil_Moisture_Content,
             real(fp), intent(in) Vegetation_Fraction,
             real(fp), intent(in) Soil_Temperature,
             real(fp), intent(in) t_skin,
             real(fp), intent(in) Lai,
             integer, intent(in) Soil_Type,
             integer, intent(in) Vegetation_Type,
             real(fp), intent(out) Emissivity_H,
             real(fp), intent(out) Emissivity_V )
```

PURPOSE: Subroutine to simulate microwave emissivity over land conditions.

REFERENCES: Weng, F., B. Yan, and N. Grody, 2001: "A microwave land emissivity model", J. Geophys. Res., 106, 20, 115-20, 123 Here is the caller graph for this function:



7.54 nesdis_landir_phymodel Module Reference

Module containing the NESDIS infrared non-snow land emissivity model.

Functions/Subroutines

- integer function, public **nesdis_landir_emiss** (Wavenumber, Angle, Land_Skin_Temperature, Soil_↩ Temperature, Soil_Moisture_Content, Vegetation_Fraction, LAI, Vegetation_Type, Soil_Type, Emissivity_H, Emissivity_V)

7.54.1 Detailed Description

Module containing the NESDIS infrared non-snow land emissivity model.

7.55 nesdis_landmw_phymodel Module Reference

Module of the physics-based microwave land surface emissivity model.

Functions/Subroutines

- integer function, public nesdis_landmw_emiss (Frequency, Angle, Land_Skin_Temperature, Soil_↩ Temperature, Soil_Moisture_Content, Vegetation_Fraction, LAI, Vegetation_Type, Soil_Type, Emissivity_H, Emissivity_V, iVar)
- integer function, public nesdis_landmw_emiss_tl (Land_Skin_Temperature_TL, Soil_Temperature_TL, Soil↩ _Moisture_Content_TL, Vegetation_Fraction_TL, Emissivity_H_TL, Emissivity_V_TL, iVar)

  PURPOSE: Tangent-linear mode of NESDIS_LandMW_Emiss.
- integer function, public nesdis_landmw_emiss_ad (Land_Skin_Temperature_AD, Soil_Temperature_AD, Soil_Moisture_Content_AD, Vegetation_Fraction_AD, Emissivity_H_AD, Emissivity_V_AD, iVar)

  PURPOSE: Adjoint mode of NESDIS_LandMW_Emiss.
- subroutine, public two_stream_solution (emiss, iVar)

  Two stream RT solver of three-layer MW land surface physical model.
- subroutine, public two_stream_solution_tl (ssalb_TL, tau_TL, r23_TL, Tskin_TL, Tsoil_TL, emiss_TL, iVar)

  PURPOSE: Tangent-linear mode of the Two_Stream_Solution.
- subroutine, public two_stream_solution_ad (ssalb_AD, tau_AD, r23_AD, Tskin_AD, Tsoil_AD, emiss_AD, i↩ Var)

  PURPOSE: Adjoint mode of the Two_Stream_Solution.

7.55.1 Detailed Description

Module of the physics-based microwave land surface emissivity model.

This module contains the NON-SNOW (bare soil, desert, vegetation-covered) physics-based land surface emissivity model of microwave channels. It wraps all the available versions, and provide a general interface for the upper-level applications.
Unlike its counterpart in the ealier CRTM releases, the emissivity and reflectivity models of Snow and Sea ice are not enclosed in this module. Dedicated modules are created for the Snow and sea ice models, separately. Tangent-linear and adjoint modes are implemented to support the variational data assimilation applications.

Soil and canopy models are also implemented in their respective individual modules. The soil and canopy modules provide the soil and canopy optical parameters. Since different model options are available in the soil module (MW↩ _Soil_Optics) and the canopy module (MW_Canopy_Optics), users need to specify the soil model and the canopy model to be used.

Non-isothermal two-stream model is enclosed in this module to account for the temperature difference between the canopy and the underlying soil.

Tangent-linear and adjoint functions are developed for the applications in data assimilation and surface property retrieval systems.

References:

```
Weng, F., B. Yan, and N. Grody, 2001: "A microwave land emissivity model",
  J. Geophys. Res., 106, 20, 115-20, 123
```

7.55.2 Function/Subroutine Documentation

7.55.2.1 nesdis_landmw_emiss()

```
integer function, public nesdis_landmw_phymodel::nesdis_landmw_emiss (
            real(fp), intent(in) Frequency,
            real(fp), intent(in) Angle,
            real(fp), intent(in) Land_Skin_Temperature,
            real(fp), intent(in) Soil_Temperature,
            real(fp), intent(in) Soil_Moisture_Content,
            real(fp), intent(in) Vegetation_Fraction,
            real(fp), intent(in) LAI,
            integer, intent(in) Vegetation_Type,
            integer, intent(in) Soil_Type,
            real(fp), intent(out) Emissivity_H,
            real(fp), intent(out) Emissivity_V,
            type(ivar_type) iVar )
```

PURPOSE: Physical simulation of the microwave emissivity over non-snow land surface using non-isothermal two-stream radiative transfer model. This is based on the version in CRTM-REL2.1.3

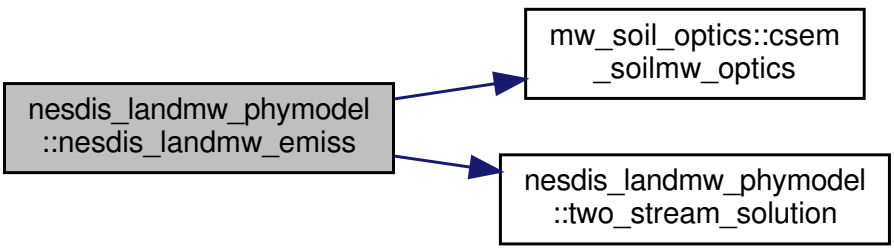
Parameters

in	*Frequency*	Frequency UNITS: GHz TYPE: REAL DIMENSION: Scalar
in	*Angle*	View angle value UNITS: Degree TYPE: REAL DIMENSION: Scalar
in	*Soil_Moisture_Content*	The volumetric water content of the top soil layer (0:1) UNITS: cm-3/cm-3 TYPE: REAL DIMENSION: Scalar
in	*Vegetation_Fraction*	Surface vegetation cover fraction (0:1) UNITS: N/A TYPE: REAL DIMENSION: Scalar
in	*Soil_Temperature*	Top-layer soil temperature UNITS: Kevin, K TYPE: REAL DIMENSION: Scalar
in	*Land_Skin_Temperature*	Land surface skin temperature UNITS: Kevin, K TYPE: REAL DIMENSION: Scalar

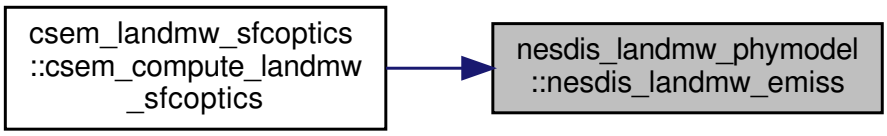
Parameters

in	*LAI*	Leaf area index UNITS: N/A TYPE: REAL DIMENSION: Scalar
in	*Soil_Type*	Soil type (1-9) UNITS: N/A TYPE: INTEGER DIMENSION: Scalar
in	*Vegetation_Type*	Land surface vegetation cover type (1-13) UNITS: N/A TYPE: INTEGER DIMENSION: Scalar
out	*Emissivity_V*	Surface emissivity of vertical polarization UNITS: N/A TYPE: REAL DIMENSION: Scalar
out	*Emissivity_H*	Surface emissivity of horizontal polarization UNITS: N/A TYPE: REAL DIMENSION: Scalar
in,out	*iVar*	Composite data structure containing internal variables required for the subsequent tangent-linear and adjoint model calls. The contents of this structure are NOT accessible outside of this module. UNITS: N/A TYPE: iVar_type DIMENSION: Scalar
out	*IO_Status*	The return value is an integer defining the error status. The error codes are defined in the CSEM_Exception_Handler module. == SUCCESS the computation was sucessful == FAILURE an unrecoverable error occurred UNITS: N/A TYPE: INTEGER DIMENSION: Scalar

Here is the call graph for this function:



Here is the caller graph for this function:



7.55.2.2 nesdis_landmw_emiss_ad()

```
integer function, public nesdis_landmw_phymodel::nesdis_landmw_emiss_ad (
            real(fp), intent(inout) Land_Skin_Temperature_AD,
            real(fp), intent(inout) Soil_Temperature_AD,
            real(fp), intent(inout) Soil_Moisture_Content_AD,
            real(fp), intent(inout) Vegetation_Fraction_AD,
            real(fp), intent(inout) Emissivity_H_AD,
            real(fp), intent(inout) Emissivity_V_AD,
            type(ivar_type) iVar )
```

PURPOSE: Adjoint mode of NESDIS_LandMW_Emiss.

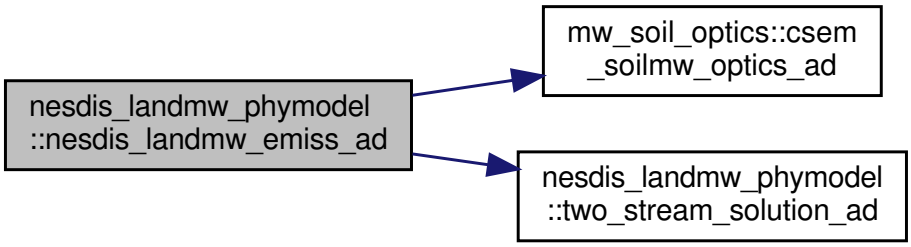
Parameters

in,out	*Soil_Moisture_Content_AD*	Soil Moisture Content Adjoint UNITS: cm-3/cm-3 TYPE: REAL DIMENSION: Scalar
in,out	*Vegetation_Fraction_AD*	Vegetation Fraction Adjoint UNITS: N/A TYPE: REAL DIMENSION: Scalar
in,out	*Soil_Temperature_AD*	Top-layer Soil Temperature Adjoint UNITS: Kelvin,K TYPE: REAL DIMENSION: Scalar
in,out	*Land_Skin_Temperature_AD*	Land Skin Temperature Adjoint UNITS: Kelvin,K TYPE: REAL DIMENSION: Scalar
in,out	*Emissivity_V_AD*	Surface Emissivity of V-pol Adjoint UNITS: N/A TYPE: REAL DIMENSION: Scalar

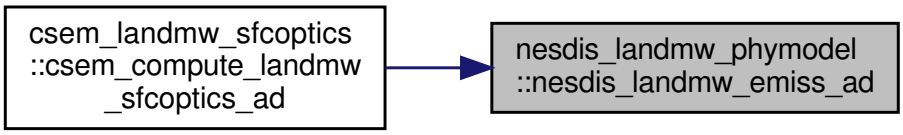
Parameters

in,out	*Emissivity_H_AD*	Surface Emissivity of H-pol Adjoint UNITS: N/A TYPE: REAL DIMENSION: Scalar
in,out	*iVar*	Composite data structure containing internal variables required for the subsequent tangent-linear and adjoint model calls. The contents of this structure are NOT accessible outside of this module. UNITS: N/A TYPE: iVar_type DIMENSION: Scalar
out	*IO_Status*	The return value is an integer defining the error status. The error codes are defined in the CSEM_Exception_Handler module. == SUCCESS the computation was sucessful == FAILURE an unrecoverable error occurred UNITS: N/A TYPE: INTEGER DIMENSION: Scalar

Here is the call graph for this function:



Here is the caller graph for this function:



7.55.2.3 nesdis_landmw_emiss_tl()

```
integer function, public nesdis_landmw_phymodel::nesdis_landmw_emiss_tl (
            real(fp), intent(in) Land_Skin_Temperature_TL,
            real(fp), intent(in) Soil_Temperature_TL,
            real(fp), intent(in) Soil_Moisture_Content_TL,
```

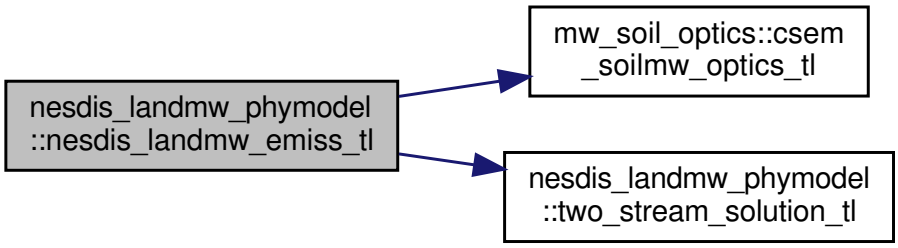
```
real(fp), intent(in) Vegetation_Fraction_TL,
real(fp), intent(out) Emissivity_H_TL,
real(fp), intent(out) Emissivity_V_TL,
type(ivar_type) iVar )
```

PURPOSE: Tangent-linear mode of NESDIS_LandMW_Emiss.

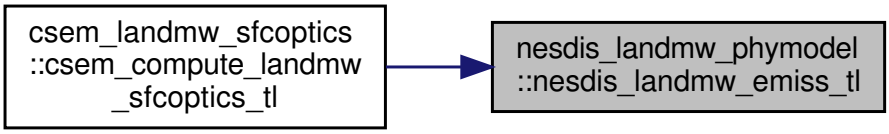
Parameters

in	*Soil_Moisture_Content_TL*	Soil Moisture Content Tangent-linear UNITS: cm-3/cm-3 TYPE: REAL DIMENSION: Scalar
in	*Vegetation_Fraction_TL*	Vegetation Fraction Tangent-linear UNITS: N/A TYPE: REAL DIMENSION: Scalar
in	*Soil_Temperature_TL*	Top-layer Soil Temperature Tangent-linear UNITS: Kelvin,K TYPE: REAL DIMENSION: Scalar
in	*Land_Skin_Temperature_TL*	Land Skin Temperature Tangent-linear UNITS: Kelvin,K TYPE: REAL DIMENSION: Scalar
out	*Emissivity_V_TL*	Surface Emissivity of V-pol Tangent-linear UNITS: N/A TYPE: REAL DIMENSION: Scalar
out	*Emissivity_H_TL*	Surface Emissivity of H-pol Tangent-linear UNITS: N/A TYPE: REAL DIMENSION: Scalar
in,out	*iVar*	Composite data structure containing internal variables required for the subsequent tangent-linear and adjoint model calls. The contents of this structure are NOT accessible outside of this module. UNITS: N/A TYPE: iVar_type DIMENSION: Scalar
out	*IO_Status*	The return value is an integer defining the error status. The error codes are defined in the CSEM_Exception_Handler module. == SUCCESS the computation was sucessful == FAILURE an unrecoverable error occurred UNITS: N/A TYPE: INTEGER DIMENSION: Scalar

Here is the call graph for this function:



Here is the caller graph for this function:



7.55.2.4 two_stream_solution()

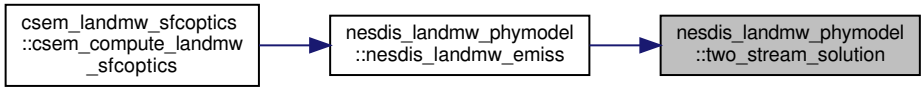
```
subroutine, public nesdis_landmw_phymodel::two_stream_solution (
             real(fp), dimension(2), intent(out) emiss,
             type(ivar_type) iVar )
```

Two stream RT solver of three-layer MW land surface physical model.

Parameters

out	*Emiss*	The surface emissivity at a vertical and horizontal polarizations UNITS: N/A DIMENSION: Array(2)
in,out	*iVar*	Structure containing internal variables required for subsequent tangent-linear or adjoint model calls. The contents of this structure are NOT accessible outside of this module.

Here is the caller graph for this function:



7.55.2.5 two_stream_solution_ad()

```
subroutine, public nesdis_landmw_phymodel::two_stream_solution_ad (
             real(fp), dimension(2), intent(inout) ssalb_AD,
```

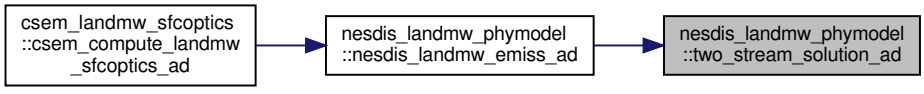
```
real(fp), dimension(2), intent(inout) tau_AD,
real(fp), dimension(2), intent(inout) r23_AD,
real(fp), intent(inout) Tskin_AD,
real(fp), intent(inout) Tsoil_AD,
real(fp), dimension(2), intent(inout) emiss_AD,
type(ivar_type) iVar )
```

PURPOSE: Adjoint mode of the Two_Stream_Solution.

Parameters

in,out	*ssalb_AD*	Single scatering albedo of canopy layer AD value (Vertical and horizontal polarizations) UNITS: N/A TYPE: REAL DIMENSION: Rank-1
in,out	*tau_AD*	Transmittance of the canopy layer AD value (Vertical and horizontal polarizations) UNITS: N/A TYPE: REAL DIMENSION: Rank-1
in,out	*r23_AD*	Intersurface reflectivity between the canopy and soil AD value (Vertical and horizontal polarizations) UNITS: N/A TYPE: REAL DIMENSION: Rank-1
in,out	*Tskin_AD*	Surface skin temperature AD value UNITS: Kelvin,K TYPE: REAL DIMENSION: Scalar
in,out	*Tsoil_AD*	Top-layer Soil temperature AD value (Vertical and horizontal polarizations) UNITS: Kelvin,K TYPE: REAL DIMENSION: Scalar
in,out	*emiss_AD*	Surface Emissivity AD (Vertical and horizontal polarizations) UNITS: N/A TYPE: REAL DIMENSION: Rank-1
in,out	*iVar*	Composite data structure containing internal variables required for the subsequent tangent-linear and adjoint model calls. The contents of this structure are NOT accessible outside of this module. UNITS: N/A TYPE: iVar_type DIMENSION: Scalar

Here is the caller graph for this function:



7.55.2.6 two_stream_solution_tl()

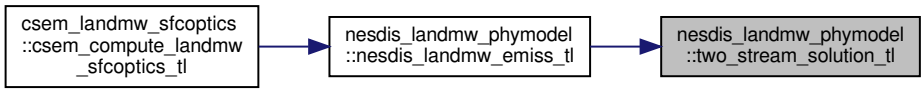
```
subroutine, public nesdis_landmw_phymodel::two_stream_solution_tl (
             real(fp), dimension(2), intent(in) ssalb_TL,
             real(fp), dimension(2), intent(in) tau_TL,
             real(fp), dimension(2), intent(in) r23_TL,
             real(fp), intent(in) Tskin_TL,
             real(fp), intent(in) Tsoil_TL,
             real(fp), dimension(2), intent(out) emiss_TL,
             type(ivar_type) iVar )
```

PURPOSE: Tangent-linear mode of the Two_Stream_Solution.

Parameters

in,out	*ssalb_TL*	Single scatering albedo of canopy layer AD value (Vertical and horizontal polarizations) UNITS: N/A TYPE: REAL DIMENSION: Rank-1
in,out	*tau_TL*	Transmittance of the canopy layer AD value (Vertical and horizontal polarizations) UNITS: N/A TYPE: REAL DIMENSION: Rank-1
in,out	*r23_TL*	Intersurface reflectivity between the canopy and soil AD value (Vertical and horizontal polarizations) UNITS: N/A TYPE: REAL DIMENSION: Rank-1
in,out	*Tskin_TL*	Surface skin temperature AD value UNITS: Kelvin,K TYPE: REAL DIMENSION: Scalar
in,out	*Tsoil_TL*	Top-layer Soil temperature AD value (Vertical and horizontal polarizations) UNITS: Kelvin,K TYPE: REAL DIMENSION: Scalar
in,out	*emiss_TL*	Surface Emissivity AD (Vertical and horizontal polarizations) UNITS: N/A TYPE: REAL DIMENSION: Rank-1
in,out	*iVar*	Composite data structure containing internal variables required for the subsequent tangent-linear and adjoint model calls. The contents of this structure are NOT accessible outside of this module. UNITS: N/A TYPE: iVar_type DIMENSION: Scalar

Here is the caller graph for this function:



7.56 nesdis_landvis_phymodel Module Reference

Module containing the NESDIS visible non-snow land emissivity model.

Functions/Subroutines

- integer function, public **nesdis_landvis_emiss** (Frequency, Angle, Land_Skin_Temperature, Soil_↩ Temperature, Soil_Moisture_Content, Vegetation_Fraction, LAI, Vegetation_Type, Soil_Type, Emissivity_H, Emissivity_V)

7.56.1 Detailed Description

Module containing the NESDIS visible non-snow land emissivity model.

7.57 nesdis_mhs_iceem_module Module Reference

Module containing the MHS microwave sea ice emissivity model.

Functions/Subroutines

- subroutine, public **nesdis_mhs_iceem** (Satellite_Angle, User_Angle, frequency, Ts, tbb, Emissivity_H, Emissivity_V)

7.57.1 Detailed Description

Module containing the MHS microwave sea ice emissivity model.

7.58 nesdis_mhs_snowem_module Module Reference

Module containing the MHS microwave snow emissivity model.

Functions/Subroutines

- subroutine, public **nesdis_mhs_snowem** (Satellite_Angle, User_Angle, frequency, Ts, tbb, Emissivity_H, Emissivity_V)

7.58.1 Detailed Description

Module containing the MHS microwave snow emissivity model.

7.59 nesdis_mw_iceem_lut Module Reference

Module containing the parameters related to microwave Ice emissivity model.

Variables

- integer, parameter, public **n_mwice_types** = 13
- character(len=20), dimension(n_mwice_types), parameter, public **nesdis_ice_type_list** = (/ 'RS_ICE_A ', 'RS_ICE_B ', 'MIXED_NEWICE_SNOW ', 'NARE_NEWICE_ ', 'BROKEN_ICE_ ', 'FIRST_YEAR_ICE_ ', 'COMPOSITE_PACK_ICE ', 'RS_ICE_C ', 'FAST_ICE ', 'RS_ICE_D ', 'RS_ICE_E ', 'RS_ICE_F ', 'GREASE_ICE ' /)
- integer, parameter, public **n_mwice_frequency** = 7

7.59.1 Detailed Description

Module containing the parameters related to microwave Ice emissivity model.

7.60 nesdis_mw_iceemiss_util Module Reference

Module containing a simplfied NESDIS physical microwave emissivity model to be used by empirical models in angle dependence estimation.

Functions/Subroutines

- subroutine, public **nesdis_mwemiss_ice** (Angle, Frequency, Soil_Moisture_Content, Vegetation_Fraction, Soil_Temperature, t_skin, Lai, Soil_Type, Vegetation_Type, Snow_depth, Emissivity_H, Emissivity_V)

7.60.1 Detailed Description

Module containing a simplfied NESDIS physical microwave emissivity model to be used by empirical models in angle dependence estimation.

7.61 nesdis_mw_snowem_lut Module Reference

Module containing the parameters related to microwave snow emissivity model.

Variables

- integer, parameter, public **n_mwsnow_types** = 16
- integer, parameter, public **invalid_snow_type** = -999
- integer, parameter, public **wet_snow** = 1
- integer, parameter, public **grass_after_snow** = 2
- integer, parameter, public **rs_snow_a** = 3
- integer, parameter, public **powder_snow** = 4
- integer, parameter, public **rs_snow_b** = 5
- integer, parameter, public **rs_snow_c** = 6
- integer, parameter, public **rs_snow_d** = 7
- integer, parameter, public **thin_crust_snow** = 8
- integer, parameter, public **rs_snow_e** = 9
- integer, parameter, public **bottom_crust_snow_a** = 10
- integer, parameter, public **shallow_snow** = 11
- integer, parameter, public **deep_snow** = 12
- integer, parameter, public **crust_snow** = 13
- integer, parameter, public **medium_snow** = 14
- integer, parameter, public **bottom_crust_snow_b** = 15
- integer, parameter, public **thick_crust_snow** = 16
- character(len=20), dimension(n_mwsnow_types), parameter, public **nesdis_snow_type_list** = (/ 'WET_↩ SNOW ', 'GRASS_AFTER_SNOW ', 'RS_SNOW_A ', 'POWDER_SNOW ', 'RS_SNOW_B ', 'RS_SNOW_C ', 'RS_SNOW_D ', 'THIN_CRUST_SNOW ', 'RS_SNOW_E ', 'BOTTOM_CRUST_SNOW_A ', 'SHALLOW_↩ SNOW ', 'DEEP_SNOW ', 'CRUST_SNOW ', 'MEDIUM_SNOW ', 'BOTTOM_CRUST_SNOW_B ', 'THICK_↩ CRUST_SNOW ' /)
- integer, parameter, public **n_mwsnow_frequency** = 10
- integer, parameter, public **n_amsre_snow_freq** = 7

7.61.1 Detailed Description

Module containing the parameters related to microwave snow emissivity model.

7.62 nesdis_mw_snowemiss_util Module Reference

Module containing a simplfied NESDIS physical microwave emissivity model to be used by empirical models in angle dependence estimation.

Functions/Subroutines

- subroutine, public **nesdis_mwemiss_snow** (Angle, Frequency, Soil_Moisture_Content, Vegetation_↩ Fraction, Soil_Temperature, t_skin, Lai, Soil_Type, Vegetation_Type, Snow_depth, Emissivity_H, Emissivity↩ _V)

7.62.1 Detailed Description

Module containing a simplfied NESDIS physical microwave emissivity model to be used by empirical models in angle dependence estimation.

7.63 nesdis_sensors_icemw_modules Module Reference

Container Module to wrap all the microwave sensor-based ice surface emissivity regression models with a generic interface.

Functions/Subroutines

- integer function, public **crtm_sensors_icemw_emiss** (Surface, SensorObs, SfcOptics)
- character(len(sensor_id)) function, public **icemw_sensorname** (sensor_id)

7.63.1 Detailed Description

Container Module to wrap all the microwave sensor-based ice surface emissivity regression models with a generic interface.

7.64 nesdis_sensors_snowmw_modules Module Reference

Module to wrap the microwave sensor-based snow-surface regression models with a generic interface.

Functions/Subroutines

- integer function, public **crtm_sensors_snowmw_emiss** (Surface, SensorObs, SfcOptics)
- character(len(sensor_id)) function, public **snowmw_sensorname** (sensor_id)

7.64.1 Detailed Description

Module to wrap the microwave sensor-based snow-surface regression models with a generic interface.

7.65 nesdis_snowem_atms_parameters Module Reference

Module containing the snow emissivity library ATMS channels. The library contain 16.

Functions/Subroutines

- integer function **snowtype_name2index** (sname)
- character(len=100) function **snowtype_index2name** (sindex)

Variables

- integer, parameter, public **n_freq_atms** = 13
- integer, parameter, public **n_snow_types** = 16
- character(len=20), dimension(n_snow_types), parameter, public **snow_type_names** =(/ 'WET_SNOW ', 'GRASS_AFTER_SNOW ', 'RS_SNOW_A ', 'POWDER_SNOW ', 'RS_SNOW_B ', 'RS_SNOW_C ', 'RS↩ _SNOW_D ', 'THIN_CRUST_SNOW ', 'RS_SNOW_E ', 'BOTTOM_CRUST_SNOW_A ', 'SHALLOW_SNOW ', 'DEEP_SNOW ', 'CRUST_SNOW ', 'MEDIUM_SNOW ', 'BOTTOM_CRUST_SNOW_B ', 'THICK_CRUST↩ _SNOW '/)
- integer, dimension(n_snow_types), parameter, public **code2excel_idx** = (/2, 1, 4, 11, 14, 12, 8, 16, 10, 15, 13, 5, 6, 3, 7, 9/)
- integer, dimension(n_snow_types), parameter, public **excel2code_idx** = (/2, 1, 14, 3, 12, 13, 15, 7, 16, 9, 4, 6, 11, 5, 10, 8/)
- real(fp), dimension(n_freq_atms), parameter, public **frequency_atms** = (/23.80_fp,31.40_fp,50.30_fp,51.↩ 76_fp,52.80_fp,53.60_fp,54.40_fp, 54.90_fp,55.50_fp,57.30_fp,88.20_fp,165.50_fp,183.30_fp/)
- real(fp), dimension(n_freq_atms, n_snow_types), parameter, public **snow_emiss_atms_h** = RESHAPE((/ 0.94_fp,0.93_fp,0.93_fp,0.93_fp,0.93_fp,0.93_fp,0.93_fp,0.93_fp,0.93_fp,0.93_fp,0.89_fp,0.88_fp, 0.90_fp,0.91_fp,0.91_fp,0.91_fp,0.91_fp,0.91_fp,0.91_fp,0.91_fp,0.91_fp,0.91_fp,0.90_fp,0.84_fp,0.82_fp, 0.84_fp,0.83_fp,0.81_fp,0.81_fp,0.81_fp,0.81_fp,0.81_fp,0.81_fp,0.81_fp,0.81_fp,0.80_fp,0.80_fp,0.80_fp, 0.92_fp,0.91_fp,0.87_fp,0.87_fp,0.87_fp,0.87_fp,0.87_fp,0.86_fp,0.86_fp,0.86_fp,0.80_fp,0.79_fp,0.78_fp, 0.76_fp,0.75_fp,0.74_fp,0.74_fp,0.74_fp,0.74_fp,0.74_fp,0.74_fp,0.74_fp,0.74_fp,0.74_fp,0.74_fp,0.74_fp, 0.78_fp,0.77_fp,0.76_fp,0.76_fp,0.76_fp,0.76_fp,0.76_fp,0.76_fp,0.76_fp,0.76_fp,0.75_fp,0.67_fp,0.66_fp, 0.75_fp,0.74_fp,0.72_fp,0.72_fp,0.72_fp,0.72_fp,0.72_fp,0.72_fp,0.72_fp,0.72_fp,0.71_fp,0.71_fp,0.71_fp, 0.94_fp,0.91_fp,0.85_fp,0.85_fp,0.85_fp,0.84_fp,0.84_fp,0.84_fp,0.84_fp,0.83_fp,0.75_fp,0.62_fp,0.60_fp, 0.72_fp,0.71_fp,0.70_fp,0.70_fp,0.70_fp,0.70_fp,0.70_fp,0.70_fp,0.70_fp,0.70_fp,0.68_fp,0.67_fp,0.66_fp, 0.85_fp,0.82_fp,0.77_fp,0.77_fp,0.76_fp,0.76_fp,0.76_fp,0.76_fp,0.76_fp,0.75_fp,0.68_fp,0.62_fp,0.60_fp, 0.78_fp,0.74_fp,0.69_fp,0.69_fp,0.69_fp,0.69_fp,0.69_fp,0.68_fp,0.68_fp,0.68_fp,0.62_fp,0.56_fp,0.54_fp, 0.80_fp,0.78_fp,0.75_fp,0.74_fp,0.74_fp,0.74_fp,0.74_fp,0.74_fp,0.74_fp,0.73_fp,0.68_fp,0.60_fp,0.59_fp, 0.71_fp,0.69_fp,0.66_fp,0.66_fp,0.66_fp,0.66_fp,0.66_fp,0.66_fp,0.66_fp,0.66_fp,0.64_fp,0.64_fp,0.64_fp, 0.88_fp,0.85_fp,0.75_fp,0.74_fp,0.74_fp,0.73_fp,0.73_fp,0.72_fp,0.72_fp,0.71_fp,0.53_fp,0.47_fp,0.45_fp, 0.82_fp,0.77_fp,0.68_fp,0.68_fp,0.67_fp,0.67_fp,0.67_fp,0.67_fp,0.66_fp,0.66_fp,0.53_fp,0.48_fp,0.47_fp, 0.81_fp,0.80_fp,0.72_fp,0.71_fp,0.70_fp,0.70_fp,0.69_fp,0.69_fp,0.69_fp,0.68_fp,0.51_fp,0.45_fp,0.43_fp /), (/N_FREQ_ATMS,N_SNOW_TYPES/))
- real(fp), dimension(n_freq_atms, n_snow_types), parameter, public **snow_emiss_atms_v** = RESHAPE((/ 0.95_fp,0.94_fp,0.94_fp,0.94_fp,0.94_fp,0.94_fp,0.94_fp,0.94_fp,0.94_fp,0.94_fp,0.90_fp,0.89_fp, 0.96_fp,0.96_fp,0.95_fp,0.95_fp,0.95_fp,0.95_fp,0.95_fp,0.95_fp,0.95_fp,0.94_fp,0.92_fp,0.86_fp,0.84_fp, 0.96_fp,0.94_fp,0.91_fp,0.91_fp,0.91_fp,0.91_fp,0.91_fp,0.91_fp,0.91_fp,0.91_fp,0.87_fp,0.87_fp,0.87_fp, 0.98_fp,0.97_fp,0.93_fp,0.93_fp,0.92_fp,0.92_fp,0.92_fp,0.92_fp,0.92_fp,0.91_fp,0.84_fp,0.83_fp,0.82_fp, 0.92_fp,0.90_fp,0.88_fp,0.88_fp,0.87_fp,0.87_fp,0.87_fp,0.87_fp,0.87_fp,0.87_fp,0.84_fp,0.84_fp,0.84_fp, 0.94_fp,0.92_fp,0.89_fp,0.89_fp,0.89_fp,0.89_fp,0.89_fp,0.89_fp,0.88_fp,0.88_fp,0.84_fp,0.76_fp,0.75_fp, 0.90_fp,0.88_fp,0.84_fp,0.84_fp,0.84_fp,0.84_fp,0.84_fp,0.84_fp,0.84_fp,0.84_fp,0.80_fp,0.80_fp,0.80_fp, 0.97_fp,0.94_fp,0.88_fp,0.88_fp,0.87_fp,0.87_fp,0.87_fp,0.87_fp,0.87_fp,0.86_fp,0.77_fp,0.64_fp,0.62_fp, 0.86_fp,0.84_fp,0.80_fp,0.80_fp,0.80_fp,0.79_fp,0.79_fp,0.79_fp,0.79_fp,0.79_fp,0.74_fp,0.73_fp,0.72_fp, 0.93_fp,0.89_fp,0.83_fp,0.82_fp,0.82_fp,0.82_fp,0.82_fp,0.81_fp,0.81_fp,0.81_fp,0.71_fp,0.65_fp,0.63_fp, 0.90_fp,0.86_fp,0.80_fp,0.79_fp,0.79_fp,0.79_fp,0.79_fp,0.78_fp,0.78_fp,0.78_fp,0.68_fp,0.62_fp,0.60_fp, 0.90_fp,0.87_fp,0.83_fp,0.83_fp,0.83_fp,0.82_fp,0.82_fp,0.82_fp,0.82_fp,0.82_fp,0.77_fp,0.69_fp,0.68_fp, 0.90_fp,0.85_fp,0.78_fp,0.78_fp,0.78_fp,0.78_fp,0.78_fp,0.77_fp,0.77_fp,0.77_fp,0.71_fp,0.71_fp,0.71_fp, 0.96_fp,0.94_fp,0.83_fp,0.82_fp,0.81_fp,0.81_fp,0.80_fp,0.80_fp,0.79_fp,0.78_fp,0.58_fp,0.51_fp,0.49_fp, 0.95_fp,0.90_fp,0.79_fp,0.78_fp,0.77_fp,0.77_fp,0.76_fp,0.76_fp,0.76_fp,0.75_fp,0.58_fp,0.53_fp,0.52_fp, 0.94_fp,0.91_fp,0.80_fp,0.79_fp,0.79_fp,0.78_fp,0.78_fp,0.77_fp,0.77_fp,0.76_fp,0.57_fp,0.50_fp,0.48_fp /), (/N_FREQ_ATMS,N_SNOW_TYPES/))
- real(fp), dimension(n_freq_atms, n_snow_types), parameter, public **snow_emiss_atms_lib** = RESHAPE((/ 0.94_fp,0.94_fp,0.94_fp,0.93_fp,0.93_fp,0.93_fp,0.93_fp,0.93_fp,0.93_fp,0.92_fp,0.89_fp,0.89_fp, 0.90_fp,0.90_fp,0.91_fp,0.91_fp,0.91_fp,0.91_fp,0.91_fp,0.91_fp,0.91_fp,0.91_fp,0.85_fp,0.83_fp, 0.86_fp,0.86_fp,0.85_fp,0.85_fp,0.85_fp,0.85_fp,0.85_fp,0.85_fp,0.85_fp,0.82_fp,0.82_fp,0.82_fp, 0.93_fp,0.93_fp,0.89_fp,0.88_fp,0.88_fp,0.88_fp,0.88_fp,0.87_fp,0.87_fp,0.87_fp,0.79_fp,0.79_fp,0.79_fp, 0.84_fp,0.83_fp,0.83_fp,0.82_fp,0.82_fp,0.82_fp,0.82_fp,0.82_fp,0.82_fp,0.82_fp,0.79_fp,0.71_fp,0.70_fp, 0.80_fp,0.79_fp,0.78_fp,0.78_fp,0.78_fp,0.78_fp,0.78_fp,0.78_fp,0.78_fp,0.78_fp,0.77_fp,0.77_fp,0.77_fp,

0.78_fp,0.77_fp,0.77_fp,0.76_fp,0.76_fp,0.76_fp,0.76_fp,0.76_fp,0.76_fp,0.75_fp,0.72_fp,0.72_fp,0.72_fp,
0.95_fp,0.93_fp,0.87_fp,0.86_fp,0.86_fp,0.86_fp,0.85_fp,0.85_fp,0.85_fp,0.84_fp,0.74_fp,0.63_fp,0.60_fp,
0.76_fp,0.76_fp,0.75_fp,0.75_fp,0.75_fp,0.75_fp,0.75_fp,0.75_fp,0.75_fp,0.74_fp,0.70_fp,0.69_fp,0.68_fp,
0.73_fp,0.68_fp,0.66_fp,0.66_fp,0.66_fp,0.66_fp,0.66_fp,0.66_fp,0.66_fp,0.66_fp,0.68_fp,0.67_fp,0.66_fp,
0.86_fp,0.82_fp,0.77_fp,0.76_fp,0.76_fp,0.76_fp,0.76_fp,0.76_fp,0.75_fp,0.75_fp,0.69_fp,0.63_fp,0.61_fp,
0.81_fp,0.77_fp,0.74_fp,0.73_fp,0.73_fp,0.73_fp,0.73_fp,0.73_fp,0.73_fp,0.72_fp,0.69_fp,0.63_fp,0.61_fp,
0.82_fp,0.78_fp,0.69_fp,0.68_fp,0.68_fp,0.67_fp,0.67_fp,0.67_fp,0.67_fp,0.66_fp,0.51_fp,0.46_fp,0.45_fp,
0.80_fp,0.75_fp,0.70_fp,0.70_fp,0.70_fp,0.70_fp,0.70_fp,0.70_fp,0.70_fp,0.69_fp,0.64_fp,0.59_fp,0.58_fp,
0.84_fp,0.76_fp,0.66_fp,0.65_fp,0.64_fp,0.64_fp,0.63_fp,0.63_fp,0.63_fp,0.62_fp,0.48_fp,0.43_fp,0.42_fp,
0.86_fp,0.74_fp,0.63_fp,0.63_fp,0.63_fp,0.63_fp,0.62_fp,0.62_fp,0.62_fp,0.61_fp,0.50_fp,0.44_fp,0.42_fp /),
(/N_FREQ_ATMS,N_SNOW_TYPES/))

- real(fp), dimension(n_frequency, n_snow_types), parameter, public **snow_emiss_default_lib** = RESHAPE((/ 0.87_fp,0.89_fp,0.91_fp,0.93_fp,0.94_fp,0.94_fp,0.94_fp,0.93_fp,0.92_fp,0.90_fp, 0.91_fp,0.↩
91_fp,0.92_fp,0.91_fp,0.90_fp,0.90_fp,0.91_fp,0.91_fp,0.91_fp,0.86_fp, 0.90_fp,0.89_fp,0.88_fp,0.87_fp,0.↩
86_fp,0.86_fp,0.85_fp,0.85_fp,0.82_fp,0.82_fp, 0.91_fp,0.91_fp,0.93_fp,0.93_fp,0.93_fp,0.93_fp,0.89_fp,0.↩
88_fp,0.79_fp,0.79_fp, 0.90_fp,0.89_fp,0.88_fp,0.85_fp,0.84_fp,0.83_fp,0.83_fp,0.82_fp,0.79_fp,0.73_fp,
0.90_fp,0.89_fp,0.86_fp,0.82_fp,0.80_fp,0.79_fp,0.78_fp,0.78_fp,0.77_fp,0.77_fp, 0.88_fp,0.86_fp,0.85_↩
fp,0.80_fp,0.78_fp,0.77_fp,0.77_fp,0.76_fp,0.72_fp,0.72_fp, 0.93_fp,0.94_fp,0.96_fp,0.96_fp,0.95_fp,0.93_↩
fp,0.87_fp,0.86_fp,0.74_fp,0.65_fp, 0.87_fp,0.86_fp,0.84_fp,0.80_fp,0.76_fp,0.76_fp,0.75_fp,0.75_fp,0.70↩
_fp,0.69_fp, 0.87_fp,0.86_fp,0.83_fp,0.77_fp,0.73_fp,0.68_fp,0.66_fp,0.66_fp,0.68_fp,0.67_fp, 0.89_fp,0.↩
89_fp,0.88_fp,0.87_fp,0.86_fp,0.82_fp,0.77_fp,0.76_fp,0.69_fp,0.64_fp, 0.88_fp,0.87_fp,0.86_fp,0.83_fp,0.↩
81_fp,0.77_fp,0.74_fp,0.73_fp,0.69_fp,0.64_fp, 0.86_fp,0.86_fp,0.86_fp,0.85_fp,0.82_fp,0.78_fp,0.69_fp,0.↩
68_fp,0.51_fp,0.47_fp, 0.89_fp,0.88_fp,0.87_fp,0.83_fp,0.80_fp,0.75_fp,0.70_fp,0.70_fp,0.64_fp,0.60_fp,
0.91_fp,0.92_fp,0.93_fp,0.88_fp,0.84_fp,0.76_fp,0.66_fp,0.64_fp,0.48_fp,0.44_fp, 0.94_fp,0.95_fp,0.97_↩
fp,0.91_fp,0.86_fp,0.74_fp,0.63_fp,0.63_fp,0.50_fp,0.45_fp /), (/N_FREQUENCY,N_SNOW_TYPES/))
- real(fp), dimension(n_freq_atms, n_snow_types), parameter, public **snow_emiss_atms_lib_2** = RESHAPE((/ 0.945_fp,0.935_fp,0.935_fp,0.935_fp,0.935_fp,0.935_fp,0.935_fp,0.935_fp,0.935_fp,0.935_↩
fp,0.935_fp,0.895_fp,0.885_fp, 0.930_fp,0.935_fp,0.930_fp,0.930_fp,0.930_fp,0.930_fp,0.930_fp,0.930_↩
fp,0.930_fp,0.925_fp,0.910_fp,0.850_fp,0.830_fp, 0.900_fp,0.885_fp,0.860_fp,0.860_fp,0.860_fp,0.860_↩
fp,0.860_fp,0.860_fp,0.860_fp,0.860_fp,0.835_fp,0.835_fp,0.835_fp, 0.950_fp,0.940_fp,0.900_fp,0.900_↩
fp,0.895_fp,0.895_fp,0.895_fp,0.890_fp,0.890_fp,0.885_fp,0.820_fp,0.810_fp,0.800_fp, 0.840_fp,0.825_↩
fp,0.810_fp,0.810_fp,0.805_fp,0.805_fp,0.805_fp,0.805_fp,0.805_fp,0.805_fp,0.790_fp,0.790_fp,0.790_↩
fp, 0.860_fp,0.845_fp,0.825_fp,0.825_fp,0.825_fp,0.825_fp,0.825_fp,0.825_fp,0.820_fp,0.820_fp,0.795_↩
fp,0.715_fp,0.705_fp, 0.825_fp,0.810_fp,0.780_fp,0.780_fp,0.780_fp,0.780_fp,0.780_fp,0.780_fp,0.780_↩
fp,0.780_fp,0.755_fp,0.755_fp,0.755_fp, 0.955_fp,0.925_fp,0.865_fp,0.865_fp,0.860_fp,0.855_fp,0.855_↩
fp,0.855_fp,0.855_fp,0.845_fp,0.760_fp,0.630_fp,0.610_fp, 0.790_fp,0.775_fp,0.750_fp,0.750_fp,0.750_↩
fp,0.745_fp,0.745_fp,0.745_fp,0.745_fp,0.745_fp,0.710_fp,0.700_fp,0.690_fp, 0.890_fp,0.855_fp,0.800_↩
fp,0.795_fp,0.790_fp,0.790_fp,0.790_fp,0.785_fp,0.785_fp,0.780_fp,0.695_fp,0.635_fp,0.615_fp, 0.840_↩
fp,0.800_fp,0.745_fp,0.740_fp,0.740_fp,0.740_fp,0.740_fp,0.730_fp,0.730_fp,0.730_fp,0.650_fp,0.590_↩
fp,0.570_fp, 0.850_fp,0.825_fp,0.790_fp,0.785_fp,0.785_fp,0.780_fp,0.780_fp,0.780_fp,0.780_fp,0.775_↩
fp,0.725_fp,0.645_fp,0.635_fp, 0.805_fp,0.770_fp,0.720_fp,0.720_fp,0.720_fp,0.720_fp,0.720_fp,0.715_↩
fp,0.715_fp,0.715_fp,0.675_fp,0.675_fp,0.675_fp, 0.920_fp,0.895_fp,0.790_fp,0.780_fp,0.775_fp,0.770_↩
fp,0.765_fp,0.760_fp,0.755_fp,0.745_fp,0.555_fp,0.490_fp,0.470_fp, 0.885_fp,0.835_fp,0.735_fp,0.730_↩
fp,0.720_fp,0.720_fp,0.715_fp,0.715_fp,0.710_fp,0.705_fp,0.555_fp,0.505_fp,0.495_fp, 0.875_fp,0.855_↩
fp,0.760_fp,0.750_fp,0.745_fp,0.740_fp,0.735_fp,0.730_fp,0.730_fp,0.720_fp,0.540_fp,0.475_fp,0.455_fp /),
(/N_FREQ_ATMS,N_SNOW_TYPES/))

7.65.1 Detailed Description

Module containing the snow emissivity library ATMS channels. The library contain 16.

7.66 nesdis_snowem_parameters Module Reference

Module containing the parameters related to microwave snow emissivity model.

Variables

- integer, parameter, public **invalid_snow_type** = -999
- integer, parameter, public **wet_snow** = 1
- integer, parameter, public **grass_after_snow** = 2
- integer, parameter, public **rs_snow_a** = 3
- integer, parameter, public **powder_snow** = 4
- integer, parameter, public **rs_snow_b** = 5
- integer, parameter, public **rs_snow_c** = 6
- integer, parameter, public **rs_snow_d** = 7
- integer, parameter, public **thin_crust_snow** = 8
- integer, parameter, public **rs_snow_e** = 9
- integer, parameter, public **bottom_crust_snow_a** = 10
- integer, parameter, public **shallow_snow** = 11
- integer, parameter, public **deep_snow** = 12
- integer, parameter, public **crust_snow** = 13
- integer, parameter, public **medium_snow** = 14
- integer, parameter, public **bottom_crust_snow_b** = 15
- integer, parameter, public **thick_crust_snow** = 16
- integer, parameter, public **n_frequency** = 10
- integer, parameter, public **n_freq_amsre** = 7
- real(fp), dimension(n_frequency), parameter, public **frequency_default** = (/ 4.9_fp, 6.93_fp, 10.65_fp, 18.↩ 7_fp,23.8_fp, 31.4_fp, 50.3_fp, 52.5_fp, 89.0_fp,150._fp/)
- real(fp), dimension(n_frequency), parameter, public **wet_snow_emiss** = (/0.87_fp,0.89_fp,0.91_fp,0.93_↩ fp,0.94_fp, 0.94_fp,0.94_fp,0.93_fp,0.92_fp,0.90_fp/)
- real(fp), dimension(n_frequency), parameter, public **grass_after_snow_emiss** = (/0.91_fp,0.91_fp,0.92_↩ fp,0.91_fp, 0.90_fp,0.90_fp,0.91_fp,0.91_fp, 0.91_fp,0.86_fp/)
- real(fp), dimension(n_frequency), parameter, public **rs_snow_a_emiss** = (/0.90_fp,0.89_fp,0.88_fp,0.87_fp, 0.86_fp, 0.86_fp,0.85_fp,0.85_fp,0.82_fp,0.82_fp/)
- real(fp), dimension(n_frequency), parameter, public **powder_snow_emiss** = (/0.91_fp,0.91_fp,0.93_fp,0.↩ 93_fp,0.93_fp, 0.93_fp,0.89_fp,0.88_fp,0.79_fp,0.79_fp/)
- real(fp), dimension(n_frequency), parameter, public **rs_snow_b_emiss** = (/0.90_fp,0.89_fp,0.88_fp,0.85_↩ fp,0.84_fp, 0.83_fp,0.83_fp,0.82_fp,0.79_fp,0.73_fp/)
- real(fp), dimension(n_frequency), parameter, public **rs_snow_c_emiss** = (/0.90_fp,0.89_fp,0.86_fp,0.82_↩ fp,0.80_fp, 0.79_fp,0.78_fp,0.78_fp,0.77_fp,0.77_fp/)
- real(fp), dimension(n_frequency), parameter, public **rs_snow_d_emiss** = (/0.88_fp,0.86_fp,0.85_fp,0.80_↩ fp,0.78_fp, 0.77_fp,0.77_fp,0.76_fp,0.72_fp,0.72_fp/)
- real(fp), dimension(n_frequency), parameter, public **thin_crust_snow_emiss** = (/0.93_fp,0.94_fp,0.96_↩ fp,0.96_fp,0.95_fp, 0.93_fp,0.87_fp,0.86_fp,0.74_fp,0.65_fp/)
- real(fp), dimension(n_frequency), parameter, public **rs_snow_e_emiss** = (/0.87_fp,0.86_fp,0.84_fp,0.80_↩ fp,0.76_fp, 0.76_fp,0.75_fp,0.75_fp,0.70_fp,0.69_fp/)
- real(fp), dimension(n_frequency), parameter, public **bottom_crust_snow_a_emiss** = (/0.87_fp,0.86_fp,0.↩ 83_fp,0.77_fp, 0.73_fp,0.68_fp,0.66_fp,0.66_fp, 0.68_fp,0.67_fp/)
- real(fp), dimension(n_frequency), parameter, public **shallow_snow_emiss** = (/0.89_fp,0.89_fp,0.88_fp,0.↩ 87_fp, 0.86_fp, 0.82_fp,0.77_fp,0.76_fp,0.69_fp,0.64_fp/)
- real(fp), dimension(n_frequency), parameter, public **deep_snow_emiss** = (/0.88_fp,0.87_fp,0.86_fp,0.83↩ _fp,0.81_fp, 0.77_fp,0.74_fp,0.73_fp,0.69_fp,0.64_fp/)
- real(fp), dimension(n_frequency), parameter, public **crust_snow_emiss** = (/0.86_fp,0.86_fp,0.86_fp,0.85↩ _fp,0.82_fp, 0.78_fp,0.69_fp,0.68_fp,0.51_fp,0.47_fp/)
- real(fp), dimension(n_frequency), parameter, public **medium_snow_emiss** = (/0.89_fp,0.88_fp,0.87_fp,0.↩ 83_fp,0.80_fp, 0.75_fp,0.70_fp,0.70_fp,0.64_fp,0.60_fp/)
- real(fp), dimension(n_frequency), parameter, public **bottom_crust_snow_b_emiss** = (/0.91_fp,0.92_fp,0.↩ 93_fp,0.88_fp, 0.84_fp,0.76_fp,0.66_fp,0.64_fp, 0.48_fp,0.44_fp/)
- real(fp), dimension(n_frequency), parameter, public **thick_crust_snow_emiss** = (/0.94_fp,0.95_fp,0.97_↩ fp,0.91_fp, 0.86_fp,0.74_fp,0.63_fp,0.63_fp, 0.50_fp,0.45_fp/)

- real(fp), dimension(n_freq_amsre), parameter, public **frequency_amsre** = (/ 6.925_fp, 10.65_fp, 18.7_↩ fp,23.8_fp, 36.5_fp, 89.0_fp,150._fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **wet_snow_em_amsre** = (/0.91_fp, 0.93_fp, 0.94_fp, 0.95_fp, 0.95_fp, 0.93_fp, 0.93_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **grass_after_snow_em_amsre** = (/0.91_fp, 0.92_fp, 0.91_fp, 0.90_fp, 0.91_fp, 0.91_fp, 0.91_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **rs_snow_a_em_amsre** = (/0.90_fp, 0.89_fp, 0.88_fp, 0.87_fp, 0.86_fp, 0.82_fp, 0.82_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **powder_snow_em_amsre** = (/0.92_fp, 0.93_fp, 0.↩ 94_fp, 0.94_fp, 0.92_fp, 0.80_fp, 0.80_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **rs_snow_b_em_amsre** = (/0.87_fp, 0.86_fp, 0.83_fp, 0.80_fp, 0.79_fp, 0.77_fp, 0.77_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **rs_snow_c_em_amsre** = (/0.89_fp, 0.88_fp, 0.85_fp, 0.84_fp, 0.83_fp, 0.79_fp, 0.79_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **rs_snow_d_em_amsre** = (/0.84_fp, 0.83_fp, 0.82_fp, 0.80_fp, 0.78_fp, 0.72_fp, 0.72_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **thin_crust_snow_em_amsre** = (/0.95_fp, 0.96_fp, 0.96_fp, 0.95_fp, 0.91_fp, 0.75_fp, 0.75_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **rs_snow_e_em_amsre** = (/0.80_fp, 0.80_fp, 0.80_fp, 0.79_fp, 0.75_fp, 0.70_fp, 0.70_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **bottom_crust_snow_a_em_amsre** = (/0.91_fp, 0.↩ 90_fp, 0.89_fp, 0.87_fp, 0.82_fp, 0.69_fp, 0.69_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **shallow_snow_em_amsre** = (/0.90_fp, 0.89_fp, 0.↩ 85_fp,0.82_fp, 0.76_fp, 0.65_fp, 0.65_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **deep_snow_em_amsre** = (/0.89_fp, 0.88_fp, 0.86_fp, 0.83_fp, 0.78_fp, 0.70_fp, 0.70_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **crust_snow_em_amsre** = (/0.88_fp, 0.86_fp, 0.80_fp, 0.75_fp, 0.69_fp, 0.67_fp, 0.67_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **medium_snow_em_amsre** = (/0.96_fp, 0.97_fp, 0.↩ 92_fp, 0.87_fp, 0.72_fp, 0.50_fp, 0.50_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **bottom_crust_snow_b_em_amsre** = (/0.93_fp, 0.↩ 94_fp, 0.89_fp, 0.85_fp, 0.74_fp, 0.48_fp, 0.48_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **thick_crust_snow_em_amsre** = (/0.88_fp, 0.88_fp, 0.87_fp, 0.85_fp, 0.77_fp, 0.52_fp, 0.52_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **wet_snow_eh_amsre** = (/0.93_fp, 0.92_fp, 0.93_fp, 0.94_fp, 0.93_fp, 0.93_fp, 0.90_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **grass_after_snow_eh_amsre** = (/0.91_fp, 0.90_fp, 0.90_fp, 0.90_fp, 0.91_fp, 0.90_fp, 0.85_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **rs_snow_a_eh_amsre** = (/0.85_fp, 0.85_fp, 0.84_fp, 0.84_fp, 0.82_fp, 0.80_fp, 0.80_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **powder_snow_eh_amsre** = (/0.90_fp, 0.90_fp, 0.↩ 92_fp, 0.92_fp, 0.90_fp, 0.80_fp, 0.79_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **rs_snow_b_eh_amsre** = (/0.82_fp, 0.81_fp, 0.77_fp, 0.76_fp, 0.74_fp, 0.74_fp, 0.74_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **rs_snow_c_eh_amsre** = (/0.84_fp, 0.83_fp, 0.80_fp, 0.78_fp, 0.77_fp, 0.75_fp, 0.69_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **rs_snow_d_eh_amsre** = (/0.77_fp, 0.77_fp, 0.76_fp, 0.75_fp, 0.73_fp, 0.71_fp, 0.71_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **thin_crust_snow_eh_amsre** = (/0.95_fp, 0.94_fp, 0.↩ 95_fp, 0.94_fp, 0.89_fp, 0.75_fp, 0.65_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **rs_snow_e_eh_amsre** = (/0.73_fp, 0.73_fp, 0.74_fp, 0.72_fp, 0.71_fp, 0.68_fp, 0.67_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **bottom_crust_snow_a_eh_amsre** = (/0.88_fp, 0.↩ 87_fp, 0.86_fp, 0.85_fp, 0.80_fp, 0.68_fp, 0.63_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **shallow_snow_eh_amsre** = (/0.86_fp, 0.84_fp, 0.↩ 80_fp, 0.78_fp, 0.72_fp, 0.62_fp, 0.57_fp/)

- real(fp), dimension(n_freq_amsre), parameter, public **deep_snow_eh_amsre** = (/0.87_fp, 0.85_fp, 0.83_fp, 0.80_fp, 0.77_fp, 0.68_fp, 0.62_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **crust_snow_eh_amsre** = (/0.82_fp, 0.78_fp, 0.74_fp, 0.71_fp, 0.67_fp, 0.64_fp, 0.64_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **medium_snow_eh_amsre** = (/0.90_fp, 0.90_fp, 0.↩ 89_fp, 0.88_fp, 0.83_fp, 0.53_fp, 0.48_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **bottom_crust_snow_b_eh_amsre** = (/0.87_fp, 0.↩ 85_fp, 0.84_fp, 0.82_fp, 0.74_fp, 0.53_fp, 0.49_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **thick_crust_snow_eh_amsre** = (/0.85_fp, 0.84_fp, 0.83_fp, 0.81_fp, 0.79_fp, 0.51_fp, 0.46_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **wet_snow_ev_amsre** = (/0.96_fp, 0.94_fp, 0.96_fp, 0.95_fp, 0.94_fp, 0.94_fp, 0.91_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **grass_after_snow_ev_amsre** = (/0.96_fp, 0.94_fp, 0.95_fp, 0.96_fp, 0.96_fp, 0.92_fp, 0.87_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **rs_snow_a_ev_amsre** = (/0.99_fp, 0.97_fp, 0.96_fp, 0.96_fp, 0.93_fp, 0.87_fp, 0.87_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **powder_snow_ev_amsre** = (/0.98_fp, 0.97_fp, 0.↩ 99_fp, 0.98_fp, 0.96_fp, 0.84_fp, 0.83_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **rs_snow_b_ev_amsre** = (/0.97_fp, 0.95_fp, 0.93_fp, 0.92_fp, 0.89_fp, 0.84_fp,0.84_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **rs_snow_c_ev_amsre** = (/1.00_fp, 0.97_fp, 0.96_fp, 0.94_fp, 0.91_fp, 0.84_fp, 0.78_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **rs_snow_d_ev_amsre** = (/0.99_fp, 0.96_fp, 0.93_fp, 0.90_fp, 0.86_fp, 0.80_fp, 0.80_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **thin_crust_snow_ev_amsre** = (/0.98_fp, 0.97_fp, 0.↩ 98_fp, 0.97_fp, 0.92_fp, 0.77_fp, 0.67_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **rs_snow_e_ev_amsre** = (/0.98_fp, 0.95_fp, 0.90_fp, 0.86_fp, 0.82_fp, 0.74_fp, 0.73_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **bottom_crust_snow_a_ev_amsre** = (/0.96_fp, 0.↩ 95_fp, 0.95_fp, 0.93_fp, 0.87_fp, 0.71_fp, 0.66_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **shallow_snow_ev_amsre** = (/0.97_fp, 0.95_fp, 0.↩ 94_fp, 0.90_fp, 0.84_fp, 0.68_fp, 0.63_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **deep_snow_ev_amsre** = (/0.96_fp, 0.94_fp, 0.92_fp, 0.90_fp, 0.85_fp, 0.77_fp, 0.71_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **crust_snow_ev_amsre** = (/0.98_fp, 0.96_fp, 0.93_fp, 0.90_fp, 0.81_fp, 0.71_fp, 0.71_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **medium_snow_ev_amsre** = (/0.99_fp, 0.97_fp, 0.↩ 98_fp, 0.96_fp, 0.92_fp, 0.57_fp, 0.52_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **bottom_crust_snow_b_ev_amsre** = (/1.00_fp, 0.↩ 97_fp, 0.97_fp, 0.95_fp, 0.86_fp, 0.58_fp, 0.54_fp/)
- real(fp), dimension(n_freq_amsre), parameter, public **thick_crust_snow_ev_amsre** = (/0.98_fp, 0.96_fp, 0.96_fp, 0.94_fp, 0.89_fp, 0.56_fp, 0.51_fp/)

7.66.1 Detailed Description

Module containing the parameters related to microwave snow emissivity model.

7.67 nesdis_snowir_phymodel Module Reference

Module containing the NESDIS Snow emissivity model of infrared bands.

Functions/Subroutines

- integer function, public **nesdis_snowir_emiss** (Frequency, Angle, Snow_Temperature, Soil_Temperature, Soil_Moisture_Content, Soil_Type, Emissivity_H, Emissivity_V)

7.67.1 Detailed Description

Module containing the NESDIS Snow emissivity model of infrared bands.

7.68 nesdis_snowmw_phymodel Module Reference

Module containing the NESDIS physical snow emissivity model of microwave Channels.

Functions/Subroutines

- subroutine, public **nesdis_snowmw_emiss** (Angle, Frequency, Soil_Moisture_Content, Vegetation_↩ Fraction, Soil_Temperature, t_skin, Lai, Soil_Type, Vegetation_Type, Snow_depth, Emissivity_H, Emissivity↩ _V)

7.68.1 Detailed Description

Module containing the NESDIS physical snow emissivity model of microwave Channels.

7.69 nesdis_snowvis_phymodel Module Reference

Module containing the NESDIS Snow emissivity model of visible bands.

Functions/Subroutines

- integer function, public **nesdis_snowvis_emiss** (Frequency, Angle, Snow_Temperature, Soil_Temperature, Soil_Moisture_Content, Soil_Type, Emissivity_H, Emissivity_V)

7.69.1 Detailed Description

Module containing the NESDIS Snow emissivity model of visible bands.

7.70 nesdis_ssmi_iceem_module Module Reference

Module containing the SSM/Imicrowave sea ice emissivity model.

Functions/Subroutines

- subroutine, public **nesdis_ssmi_iceem** (frequency, Angle, Ts, tb, Depth, Emissivity_H, Emissivity_V)

7.70.1 Detailed Description

Module containing the SSM/Imicrowave sea ice emissivity model.

7.71 nesdis_ssmi_snowem_module Module Reference

Module containing the SSM/I microwave snow emissivity model.

Functions/Subroutines

- subroutine, public **nesdis_ssmi_snowem** (frequency, Angle, Ts, tb, Depth, Emissivity_H, Emissivity_V)

7.71.1 Detailed Description

Module containing the SSM/I microwave snow emissivity model.

7.72 nesdis_ssmis_iceem_module Module Reference

Module containing the SSMIS microwave sea ice emissivity model.

Functions/Subroutines

- subroutine, public **nesdis_ssmis_iceem** (frequency, Angle, Ts, tb, Depth, Emissivity_H, Emissivity_V)

7.72.1 Detailed Description

Module containing the SSMIS microwave sea ice emissivity model.

7.73 nesdis_waterir_brdf_module Module Reference

Module to compute the ocean surface BRDF at near-infrared channels.

Functions/Subroutines

- integer function, public **nesdis_irwater_brdf** (Wavenumber, Wind_Speed, Sensor_Zenith_Radian, Sensor↩ _Azimuth_Radian, Source_Zenith_Radian, Source_Azimuth_Radian, Direct_Reflectivity, iVar)
- integer function, public **nesdis_irwater_brdf_tl** (Wind_Speed_TL, Direct_Reflectivity_TL, iVar)
- integer function, public **nesdis_irwater_brdf_ad** (Direct_Reflectivity_AD, Wind_Speed_AD, iVar)

7.73.1 Detailed Description

Module to compute the ocean surface BRDF at near-infrared channels.

7.74 nesdis_waterir_emiss_module Module Reference

Module containing function to invoke the CSEM Infrared Sea Surface Emissivity Model (IRSSEM).

Functions/Subroutines

- integer function, public **nesdis_waterir_emiss** (Wind_Speed, Frequency, Angle, Emissivity, iVar)
- integer function, public **nesdis_waterir_emiss_tl** (Wind_Speed_TL, Emissivity_TL, iVar)
- integer function, public **nesdis_waterir_emiss_ad** (Emissivity_AD, Wind_Speed_AD, iVar)
- integer function, public **irssem_setup** (Coeff_File_Name)
- logical function, public **irssem_initialized** ()
- integer function, public **irssem_cleanup** ()

7.74.1 Detailed Description

Module containing function to invoke the CSEM Infrared Sea Surface Emissivity Model (IRSSEM).

7.75 nesdis_waterir_emiss_v2_module Module Reference

Module containing function to invoke the Ver-2 CSEM Infrared Sea Surface Emissivity Model (IRSSEM) .

Functions/Subroutines

- integer function, public **nesdis_waterir_emiss_v2** (Wind_Speed, Temperature, Frequency, Angle, Emissivity, iVar)
- integer function, public **nesdis_waterir_emiss_v2_tl** (Wind_Speed_TL, Temperature_TL, Emissivity_TL, iVar)
- integer function, public **nesdis_waterir_emiss_v2_ad** (Emissivity_AD, Wind_Speed_AD, Temperature_AD, iVar)
- integer function, public **irssem_v2_setup** (Coeff_File_Name)
- logical function, public **irssem_v2_initialized** ()
- integer function, public **irssem_v2_cleanup** ()

7.75.1 Detailed Description

Module containing function to invoke the Ver-2 CSEM Infrared Sea Surface Emissivity Model (IRSSEM) .

7.76 nesdis_waterir_phymodel Module Reference

Module containing the NESDIS water emissivity model of infrared channels.

Functions/Subroutines

- integer function, public **nesdis_irssem_brdf_tl** (Wind_Speed_TL, Emissivity_TL, Reflectivity_TL, Direct_↩ Reflectivity_TL, iVar)
- integer function, public **nesdis_irssem_brdf_ad** (Emissivity_AD, Reflectivity_AD, Direct_Reflectivity_AD, Wind_Speed_AD, iVar)
- integer function, public **nesdis_irssem_setup** (File_Name)
- subroutine, public **nesdis_irssem_close** ()
- logical function, public **nesdis_irssem_initialized** ()

7.76.1 Detailed Description

Module containing the NESDIS water emissivity model of infrared channels.

7.77 nesdis_waterir_phymodel_v2 Module Reference

Module containing the NESDIS water emissivity model of infrared channels.

Functions/Subroutines

- integer function, public **nesdis_irssem_brdf_v2_tl** (Wind_Speed_TL, Temperature_TL, Emissivity_TL, Reflectivity_TL, Direct_Reflectivity_TL, iVar)
- integer function, public **nesdis_irssem_brdf_v2_ad** (Emissivity_AD, Reflectivity_AD, Direct_Reflectivity_↩ AD, Wind_Speed_AD, Temperature_AD, iVar)
- integer function, public **nesdis_irssem_v2_setup** (File_Name)
- subroutine, public **nesdis_irssem_v2_close** ()
- logical function, public **nesdis_irssem_v2_initialized** ()

7.77.1 Detailed Description

Module containing the NESDIS water emissivity model of infrared channels.

7.78 nesdis_watervis_brdf_module Module Reference

Module to compute the ocean surface BRDF at visible wavelength.

Functions/Subroutines

- integer function, public **nesdis_viswater_brdf** (Wavenumber, Wind_Speed, Sensor_Zenith_Radian, Sensor_Azimuth_Radian, Source_Zenith_Radian, Source_Azimuth_Radian, Direct_Reflectivity, iVar)
- integer function, public **nesdis_viswater_brdf_tl** (Wind_Speed_TL, Direct_Reflectivity_TL, iVar)
- integer function, public **nesdis_viswater_brdf_ad** (Direct_Reflectivity_AD, Wind_Speed_AD, iVar)

7.78.1 Detailed Description

Module to compute the ocean surface BRDF at visible wavelength.

7.79 nesdis_watervis_phymodel Module Reference

Module containing the NESDIS physical Water emissivity model of visibal bands.

Functions/Subroutines

- integer function, public **nesdis_watervis_emiss** (Frequency, Angle, Water_Temperature, Salinity, Wind_↩ Speed, Wind_Direction, Emissivity_H, Emissivity_V)

7.79.1 Detailed Description

Module containing the NESDIS physical Water emissivity model of visibal bands.

7.80 npoess_lut_module Module Reference

Module for users to use the LUT of the land IR-VIS surface emissivity/reflectivity spectrum with respect to NPOESS surface types.

7.80.1 Detailed Description

Module for users to use the LUT of the land IR-VIS surface emissivity/reflectivity spectrum with respect to NPOESS surface types.

7.81 npoess_lut_reader Module Reference

Module containing the load/destruction routines to handel the shared NPOESS LUT.

Functions/Subroutines

- integer function, public **read_npoess_lut** (wavenumber, emissivity, surface_type)
- integer function, public **read_npoess_lut_0** (wavelength, emissivity, surface_type)
- subroutine, public **read_stype_map** (alat, alon, stype)
- subroutine, public **load_npoess_lut** ()

7.81.1 Detailed Description

Module containing the load/destruction routines to handel the shared NPOESS LUT.

7.82 ocean_permittivity Module Reference

Container module for the sea water complex permittivity model collections.

7.82.1 Detailed Description

Container module for the sea water complex permittivity model collections.

Three models are included in this module; that of

Guillou, C. et al. (1998) Impact of new permittivity measurements on sea surface emissivity modeling in microwaves. Radio Science, Volume 33, Number 3, Pages 649-667

and of

Ellison, W.J. et al. (2003) A comparison of ocean emissivity models using the Advanced Microwave Sounding Unit, the Special Sensor Microwave Imager, the TRMM Microwave Imager, and airborne radiometer observations. Journal of Geophysical Research, v108, D21, Pages ACL 1,1-14 doi:10.1029/2002JD0032132

and of

Liu, Q. et al. (2010) An improved fast microwave water emissivity model. IEEE Trans. Geosci. Remote Sensing, accepted June 25, 2010

7.83 reflection_correction_module Module Reference

Helper module conmtaining the reflection correction routines for the CRTM implementation of FASTEM4 and FASTEM5.

Functions/Subroutines

- subroutine, public **reflection_correction** (RCCoeff, Frequency, cos_z, Wind_Speed, Transmittance, Rv_↩ Mod, Rh_Mod, iVar)
- subroutine, public **reflection_correction_tl** (RCCoeff, Wind_Speed_TL, Transmittance_TL, Rv_Mod_TL, Rh_Mod_TL, iVar)
- subroutine, public **reflection_correction_ad** (RCCoeff, Rv_Mod_AD, Rh_Mod_AD, Wind_Speed_AD, Transmittance_AD, iVar)

7.83.1 Detailed Description

Helper module conmtaining the reflection correction routines for the CRTM implementation of FASTEM4 and FASTEM5.

7.84 rttov_fastem5r1_ad_module Module Reference

AD of RTTOV FASTEM-4,5,6 emissivity and reflectance calculation.

Functions/Subroutines

- subroutine **rttov_fastem5r1_ad** (fastem_version, Frequency, Zenith_Angle, Temperature, Salinity, Wind↩ _Speed, Emissivity_ad, Reflectivity_ad, Temperature_ad, Salinity_ad, Wind_Speed_ad, Emissivity, Reflectivity, Transmittance, Rel_Azimuth, Transmittance_ad, Rel_Azimuth_ad, Supply_Foam_Fraction, Foam_↩ Fraction, Foam_Fraction_ad)

7.84.1 Detailed Description

AD of RTTOV FASTEM-4,5,6 emissivity and reflectance calculation.

7.85 rttov_fastem5r1_module Module Reference

Compute RTTOV FASTEM-4,5,6 emissivity and reflectance for a single channel.

Functions/Subroutines

- subroutine rttov_fastem5r1 (fastem_version, Frequency, Zenith_Angle, Temperature, Salinity, Wind_Speed, Emissivity, Reflectivity, Transmittance, Rel_Azimuth, Supply_Foam_Fraction, Foam_Fraction)

  Compute FASTEM-4,5,6 emissivity and reflectance for a single channel.

7.85.1 Detailed Description

Compute RTTOV FASTEM-4,5,6 emissivity and reflectance for a single channel.

7.85.2 Function/Subroutine Documentation

7.85.2.1 rttov_fastem5r1()

```
subroutine rttov_fastem5r1_module::rttov_fastem5r1 (
             integer, intent(in) fastem_version,
             real(fp), intent(in) Frequency,
             real(fp), intent(in) Zenith_Angle,
             real(fp), intent(in) Temperature,
             real(fp), intent(in) Salinity,
             real(fp), intent(in) Wind_Speed,
             real(fp), dimension(4), intent(out) Emissivity,
             real(fp), dimension(4), intent(out) Reflectivity,
             real(fp), intent(in) Transmittance,
             real(fp), intent(in) Rel_Azimuth,
             logical, intent(in), optional Supply_Foam_Fraction,
             real(fp), intent(in), optional Foam_Fraction )
```

Compute FASTEM-4,5,6 emissivity and reflectance for a single channel.

References for FASTEM are given in the user guide.

Parameters

in	*fastem_version*	FASTEM version to compute (4, 5 or 6)
in	*frequency*	channel frequency (GHz)
in	*zenith_angle*	profile zenith angle (degrees)
in	*temperature*	profile skin temperature (K)
in	*salinity*	profile salinity (practical salinity units)
in	*wind_speed*	profile wind speed (m/s)
out	*emissivity*	calculated emissivity (4 Stokes components)
out	*reflectivity*	calculated reflectivity (4 Stokes components)
in	*transmittance*	surface-to-space transmittance
in	*rel_azimuth*	relative azimuth angle
in	*supply_foam_fraction*	flag to indicate user is supplying foam fraction, optional
in	*foam_fraction*	user supplied foam fraction, optional

7.86 rttov_fastem5r1_tl_module Module Reference

TL of RTTOV FASTEM-4,5,6 emissivity and reflectance calculation.

Functions/Subroutines

- subroutine rttov_fastem5r1_tl (fastem_version, Frequency, Zenith_Angle, Temperature, Salinity, Wind_↩ Speed, Temperature_tl, Salinity_tl, Wind_Speed_tl, Emissivity, Reflectivity, Emissivity_tl, Reflectivity_↩ tl, Transmittance, Rel_Azimuth, Transmittance_tl, Rel_Azimuth_tl, Supply_Foam_Fraction, Foam_Fraction, Foam_Fraction_tl)

  TL of FASTEM-4,5,6 emissivity and reflectance calculation.

7.86.1 Detailed Description

TL of RTTOV FASTEM-4,5,6 emissivity and reflectance calculation.

7.86.2 Function/Subroutine Documentation

7.86.2.1 rttov_fastem5r1_tl()

```
subroutine rttov_fastem5r1_tl_module::rttov_fastem5r1_tl (
             integer, intent(in) fastem_version,
             real(fp), intent(in) Frequency,
             real(fp), intent(in) Zenith_Angle,
             real(fp), intent(in) Temperature,
             real(fp), intent(in) Salinity,
             real(fp), intent(in) Wind_Speed,
             real(fp), intent(in) Temperature_tl,
```

```
real(fp), intent(in) Salinity_tl,
real(fp), intent(in) Wind_Speed_tl,
real(fp), dimension(4), intent(out) Emissivity,
real(fp), dimension(4), intent(out) Reflectivity,
real(fp), dimension(4), intent(inout) Emissivity_tl,
real(fp), dimension(4), intent(inout) Reflectivity_tl,
real(fp), intent(in) Transmittance,
real(fp), intent(in) Rel_Azimuth,
real(fp), intent(in), optional Transmittance_tl,
real(fp), intent(in), optional Rel_Azimuth_tl,
logical, intent(in), optional Supply_Foam_Fraction,
real(fp), intent(in), optional Foam_Fraction,
real(fp), intent(in), optional Foam_Fraction_tl )
```

TL of FASTEM-4,5,6 emissivity and reflectance calculation.

Parameters

in	*fastem_version*	FASTEM version to compute (4, 5 or 6)
in	*frequency*	channel frequency (GHz)
in	*zenith_angle*	profile zenith angle (degrees)
in	*temperature*	profile skin temperature (K)
in	*salinity*	profile salinity (practical salinity units)
in	*wind_speed*	profile wind speed (m/s)
in,out	*emissivity_tl*	emissivity perturbation (4 Stokes components)
in,out	*reflectivity_tl*	reflectivity perturbation (4 Stokes components)
in	*temperature_tl*	profile skin temperature perturbation
in	*salinity_tl*	profile salinity perturbation
in	*wind_speed_tl*	profile wind speed perturbation
out	*emissivity*	calculated emissivity (4 Stokes components)
out	*reflectivity*	calculated reflectivity (4 Stokes components)
in	*transmittance*	surface-to-space transmittance
in	*rel_azimuth*	relative azimuth angle
in	*transmittance_tl*	surface-to-space transmittance perturbation
in	*rel_azimuth_tl*	relative azimuth angle perturbation
in	*supply_foam_fraction*	flag to indicate user is supplying foam fraction, optional
in	*foam_fraction*	user supplied foam fraction, optional
in	*foam_fraction_tl*	user foam fraction perturbation, optional

7.87 rttov_fastem_module Module Reference

Module to provide a general interface to RTTOV FASTEM modules.

Functions/Subroutines

- subroutine, public **compute_rttov_fastem** (fastem_version, Frequency, Zenith_Angle, Temperature, Salinity, Wind_Speed, Emissivity, Reflectivity, Transmittance, Rel_Azimuth, Supply_Foam_Fraction, Foam_Fraction, iVar)
- subroutine, public **compute_rttov_fastem_tl** (Temperature_tl, Salinity_tl, Wind_Speed_tl, Emissivity_↩ tl, Reflectivity_tl, Transmittance_tl, Rel_Azimuth_tl, Foam_Fraction_tl, iVar)
- subroutine, public **compute_rttov_fastem_ad** (Emissivity_ad, Reflectivity_ad, Temperature_ad, Salinity_ad, Wind_Speed_ad, Transmittance_ad, Rel_Azimuth_ad, Foam_Fraction_ad, iVar)

7.87.1 Detailed Description

Module to provide a general interface to RTTOV FASTEM modules.

7.88 rttov_tessem_mod Module Reference

Subroutines for TESSEM2 MW sea surface emissivity model.

Data Types

- type tessem_net

Functions/Subroutines

- subroutine **prop_neuralnet** (net, x, y)
- subroutine **rttov_tessem** (freq, theta, windspeed, tskin, salinity, emis_h, emis_v)
- subroutine **prop_neuralnet_tl** (net, x, x_tl, y_tl)
- subroutine **rttov_tessem_tl** (freq, theta, windspeed, tskin, salinity, windspeed_tl, tskin_tl, salinity_tl, emis_h_tl, emis_v_tl)
- subroutine **prop_neuralnet_ad** (net, x, x_ad, y_ad)
- subroutine **rttov_tessem_ad** (freq, theta, windspeed, tskin, salinity, windspeed_ad, tskin_ad, salinity_ad, emis_h_ad, emis_v_ad)

Variables

- integer(jpim), parameter **tessem_nin** = 5
- integer(jpim), parameter **tessem_nout** = 1
- integer(jpim), parameter **tessem_ncache** = 15
- type(tessem_net) **net_h**
- type(tessem_net) **net_v**

7.88.1 Detailed Description

Subroutines for TESSEM2 MW sea surface emissivity model.

This contains the code which implements TESSEM2 for the direct, TL, and AD/K models.

TESSEM2 is a neural network-based emissivity model applicable between 10 and 700GHz.

It is recommmended to use TESSEM2 for channels above 200GHz.

For frequencies below 200GHz TESSEM2 is based on FASTEM-6, but there is no azimuthal dependence.

Reference: Prigent, C., Aires, F., Wang, D., Fox, S. and Harlow, C. (2016) Sea surface emissivity parameterization from microwaves to millimeter waves. Q.J.R. Meteorol. Soc. Accepted Author Manuscript. doi:10.1002/qj.2953

7.89 slope_variance Module Reference

Helper module containing the slope variance routines for the CRTM implementation of FASTEM4.

Functions/Subroutines

- subroutine, public **compute_slope_variance** (Frequency, Wind_Speed, iVar, Variance)
- subroutine, public **compute_slope_variance_tl** (Wind_Speed_TL, iVar, Variance_TL)
- subroutine, public **compute_slope_variance_ad** (Variance_AD, iVar, Wind_Speed_AD)

7.89.1 Detailed Description

Helper module containing the slope variance routines for the CRTM implementation of FASTEM4.

7.90 small_scale_correction_module Module Reference

Module containing the small-scale correction procedures for the CRTM implementations of FASTEM4 and FASTEM5.

Functions/Subroutines

- subroutine, public small_scale_correction (SSCCoeff, Frequency, cos_Z, Wind_Speed, Correction, iVar)
- subroutine, public small_scale_correction_tl (SSCCoeff, Wind_Speed_TL, Correction_TL, iVar)
- subroutine, public small_scale_correction_ad (SSCCoeff, Correction_AD, Wind_Speed_AD, iVar)

7.90.1 Detailed Description

Module containing the small-scale correction procedures for the CRTM implementations of FASTEM4 and FASTEM5.

Equation (A4) of

Liu, Q. et al. (2011) An Improved Fast Microwave Water Emissivity Model, TGRSS, 49, pp1238-1250

describes the fitting of the small-scale correction formulation given in equation (17a,b) of

Liu, Q. et al. (1998) Monte Carlo simulations of the microwave emissivity of the sea surface, JGR, 103, pp24983-24989

and originally in equation (30) of

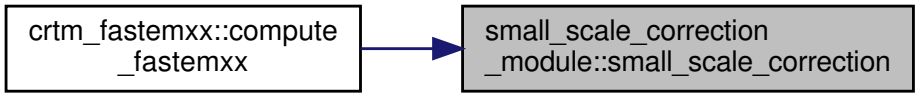
Guissard,A. and P.Sobieski (1987) An approximate model for the microwave brightness temperature of the sea, Int.J.Rem.Sens., 8, pp1607-1627.

7.90.2 Function/Subroutine Documentation

7.90.2.1 small_scale_correction()

```
subroutine, public small_scale_correction_module::small_scale_correction (
             type(csem_fitcoeff_1d_type), intent(in) SSCCoeff,
             real(fp), intent(in) Frequency,
             real(fp), intent(in) cos_Z,
             real(fp), intent(in) Wind_Speed,
             real(fp), intent(out) Correction,
             type(ivar_type), intent(inout) iVar )
```

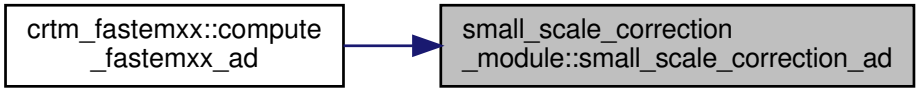
Procedures to compute the reflectivity small scale correction Here is the caller graph for this function:



7.90.2.2 small_scale_correction_ad()

```
subroutine, public small_scale_correction_module::small_scale_correction_ad (
             type(csem_fitcoeff_1d_type), intent(in) SSCCoeff,
             real(fp), intent(inout) Correction_AD,
             real(fp), intent(inout) Wind_Speed_AD,
             type(ivar_type), intent(in) iVar )
```

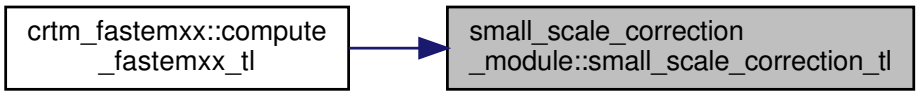
Adjoint model of Small_Scale_Correction Here is the caller graph for this function:



7.90.2.3 small_scale_correction_tl()

```
subroutine, public small_scale_correction_module::small_scale_correction_tl (
             type(csem_fitcoeff_1d_type), intent(in) SSCCoeff,
             real(fp), intent(in) Wind_Speed_TL,
             real(fp), intent(out) Correction_TL,
             type(ivar_type), intent(in) iVar )
```

Tangent-linear model of Small_Scale_Correction Here is the caller graph for this function:



7.91 snowmw_optical_model Module Reference

Module containing functions to simuate snow optics properties.

Functions/Subroutines

- subroutine **snow_diel** (frequency, ep_real, ep_imag, rad, frac, ep_eff)
- subroutine **snow_optic** (frequency, a, h, f, ep_real, ep_imag, gv, gh, ssalb_v, ssalb_h, tau_v, tau_h)

7.91.1 Detailed Description

Module containing functions to simuate snow optics properties.

7.92 telsem2_atlas_module Module Reference

Module for users to use TELSEM2 land surface emissivity data sets by CSEM interfaces.

Functions/Subroutines

- integer function, public **telsem2_atlas_setup** (imonth, path, mw_atlas_ver)
- integer function, public **telsem2_atlas_emiss** (Frequency, Angle, Latitude, Longitude, imonth, Emissivity_H, Emissivity_V, resolution, emis_std_v, emis_std_h, emis_cov, stype)
- integer function, public **telsem2_atlas_emiss_nchannels** (Frequency, Angle, Latitude, Longitude, imonth, crtm_polar_idx, n_Channels, emissivity, resolution, emis_std, emis_cov, stype)
- subroutine, public **telsem2_atlas_close** ()
- logical function, public **telsem2_atlas_initialized** (imonth)

7.92.1 Detailed Description

Module for users to use TELSEM2 land surface emissivity data sets by CSEM interfaces.

7.93 telsem2_atlas_reader Module Reference

Subroutines for TELSEM2 MW emissivity atlas and interpolator.

Functions/Subroutines

- subroutine, public test_inputs (month, lat, lon, theta, freq)

  Subroutine to check input variables: not used by RTTOV.
- subroutine, public rttov_readmw_atlas (dir, month, atlas, verbose, err, lat1, lat2, lon1, lon2)

  Initialise a TELSEM2 atlas data structure. Atlas data may be initialised for a region of the globe defined by the lower and upper lat/lon limits, though this feature is not used by RTTOV.
- integer function, public **load_telsem2_atlas** (dir, month, lat1, lat2, lon1, lon2)
- subroutine, public rttov_closemw_atlas ()

  Deallocate data in TELSEM2 atlas data structure.
- subroutine, public emis_interp_ind_sing (lat, lon, theta, freq, ev, eh, stdv, stdh, covvh, verb)

  Return emissivities for a single channel at the native atlas resolution.
- subroutine, public emis_interp_ind_mult (lat, lon, theta, freq, n_chan, ev, eh, std, verb)

  Return emissivities for multiple channels at the native atlas resolution. Each dimension of the covariance matrix std(:,:) has V-pol values for all channels followed by H-pol values for all channels.
- subroutine, public emis_interp_int_sing (lat, lon, resol, theta, freq, ev, eh, stdv, stdh, covvh, verb)

  Return emissivities for a single channel at the user-specified resolution.
- subroutine, public emis_interp_int_mult (lat, lon, resol, theta, freq, n_chan, ev, eh, std, verb)

  Return emissivities for multiple channels at the user-specified resolution. Each dimension of the covariance matrix std(:,:) has V-pol values for all channels followed by H-pol values for all channels.

Variables

- type(telsem2_atlas_data), save, public **atlas2**

7.93.1 Detailed Description

Subroutines for TELSEM2 MW emissivity atlas and interpolator.

It is intended that this atlas be used via the RTTOV interface rather than by calling these subroutines directly.

Surface emissivity at microwaves to millimeter waves over Polar Regions: parameterization and evaluation with aircraft experiments D. Wang, C. Prigent, L. Kilic, S. Fox, R. C. Harlow, C. Jimenez, F. Aires, C. Grassotti, and F. Karbou Submitted to QJRMS

7.93.2 Function/Subroutine Documentation

7.93.2.1 emis_interp_ind_mult()

```
subroutine, public telsem2_atlas_reader::emis_interp_ind_mult (
            real(jprb), intent(in) lat,
            real(jprb), intent(in) lon,
            real(jprb), intent(in) theta,
            real(jprb), dimension(:), intent(in) freq,
            integer, intent(in) n_chan,
            real(jprb), dimension(:), intent(out) ev,
            real(jprb), dimension(:), intent(out) eh,
            real(jprb), dimension(:,:), intent(out), optional std,
            integer, intent(in) verb )
```

Return emissivities for multiple channels at the native atlas resolution. Each dimension of the covariance matrix std(:,:) has V-pol values for all channels followed by H-pol values for all channels.

Parameters

in	*lat*	latitude
in	*lon*	longitude
in	*theta*	zenith angle
in	*freq*	frequencies
in	*n_chan*	number of channels
in	*atlas*	TELSEM2 atlas data
out	*ev*	V-pol emissivities
out	*eh*	H-pol emissivities
out	*std*	Covariance matrix, optional
in	*verb*	switch for verbose output

7.93.2.2 emis_interp_ind_sing()

```
subroutine, public telsem2_atlas_reader::emis_interp_ind_sing (
             real(jprb), intent(in) lat,
             real(jprb), intent(in) lon,
             real(jprb), intent(in) theta,
             real(jprb), intent(in) freq,
             real(jprb), intent(out) ev,
             real(jprb), intent(out) eh,
             real(jprb), intent(out), optional stdv,
             real(jprb), intent(out), optional stdh,
             real(jprb), intent(out), optional covvh,
             integer, intent(in) verb )
```

Return emissivities for a single channel at the native atlas resolution.

Parameters

in	*lat*	latitude
in	*lon*	longitude
in	*theta*	zenith angle
in	*freq*	frequency
in	*atlas*	TELSEM2 atlas data
out	*ev*	V-pol emissivity
out	*eh*	H-pol emissivity
out	*stdv*	V-pol emissivity standard deviation, optional
out	*stdh*	H-pol emissivity standard deviation, optional
out	*covvh*	H-/V-pol emissivity covariance, optional
in	*verb*	switch for verbose output

7.93.2.3 emis_interp_int_mult()

```
subroutine, public telsem2_atlas_reader::emis_interp_int_mult (
```

```
            real(jprb), intent(in) lat,
            real(jprb), intent(in) lon,
            real(jprb), intent(in) resol,
            real(jprb), intent(in) theta,
            real(jprb), dimension(:), intent(in) freq,
            integer, intent(in) n_chan,
            real(jprb), dimension(:), intent(out) ev,
            real(jprb), dimension(:), intent(out) eh,
            real(jprb), dimension(:,:), intent(out), optional std,
            integer, intent(in) verb )
```

Return emissivities for multiple channels at the user-specified resolution. Each dimension of the covariance matrix std(:,:) has V-pol values for all channels followed by H-pol values for all channels.

Parameters

in	*lat*	latitude
in	*lon*	longitude
in	*resol*	resolution
in	*theta*	zenith angle
in	*freq*	frequencies
in	*n_chan*	number of channels
in	*atlas*	TELSEM2 atlas data
out	*ev*	V-pol emissivities
out	*eh*	H-pol emissivities
out	*std*	Covariance matrix, optional
in	*verb*	switch for verbose output

7.93.2.4 emis_interp_int_sing()

```
subroutine, public telsem2_atlas_reader::emis_interp_int_sing (
            real(jprb), intent(in) lat,
            real(jprb), intent(in) lon,
            real(jprb), intent(in) resol,
            real(jprb), intent(in) theta,
            real(jprb), intent(in) freq,
            real(jprb), intent(out) ev,
            real(jprb), intent(out) eh,
            real(jprb), intent(out), optional stdv,
            real(jprb), intent(out), optional stdh,
            real(jprb), intent(out), optional covvh,
            integer, intent(in) verb )
```

Return emissivities for a single channel at the user-specified resolution.

Parameters

in	*lat*	latitude
in	*lon*	longitude
in	*resol*	resolution
in	*theta*	zenith angle

Parameters

in	*freq*	frequency
in	*atlas*	TELSEM2 atlas data
out	*ev*	V-pol emissivity
out	*eh*	H-pol emissivity
out	*stdv*	V-pol emissivity standard deviation, optional
out	*stdh*	H-pol emissivity standard deviation, optional
out	*covvh*	H-/V-pol emissivity covariance, optional
in	*verb*	switch for verbose output

7.93.2.5 rttov_closemw_atlas()

```
subroutine, public telsem2_atlas_reader::rttov_closemw_atlas
```

Deallocate data in TELSEM2 atlas data structure.

Parameters

in,out	*atlas*	TELSEM2 atlas data structure to deallocate

7.93.2.6 rttov_readmw_atlas()

```
subroutine, public telsem2_atlas_reader::rttov_readmw_atlas (
             character(len=*), intent(in) dir,
             integer, intent(in) month,
             type(telsem2_atlas_data), intent(inout) atlas,
             logical, intent(in) verbose,
             integer, intent(inout) err,
             real(jprb), intent(in), optional lat1,
             real(jprb), intent(in), optional lat2,
             real(jprb), intent(in), optional lon1,
             real(jprb), intent(in), optional lon2 )
```

Initialise a TELSEM2 atlas data structure. Atlas data may be initialised for a region of the globe defined by the lower and upper lat/lon limits, though this feature is not used by RTTOV.

Parameters

in	*dir*	path to atlas data files
in	*month*	month of data to read (1-12)
in,out	*atlas*	TELSEM2 atlas data structure to initialise
in	*verbose*	flag to turn verbose output on/off
in,out	*err*	status on exit
in	*lat1*	latitude lower bound, optional
in	*lat2*	latitude upper bound, optional
in	*lon1*	longitude lower bound, optional
in	*lon2*	longitude upper bound, optional

7.93.2.7 test_inputs()

```
subroutine, public telsem2_atlas_reader::test_inputs (
            integer, intent(in) month,
            real(jprb), intent(in) lat,
            real(jprb), intent(in) lon,
            real(jprb), intent(in) theta,
            real(jprb), intent(in) freq )
```

Subroutine to check input variables: not used by RTTOV.

Parameters

in	*month*	month (1-12)
in	*lat*	latitude
in	*lon*	longitude
in	*theta*	zenith angle
in	*freq*	channel frequency (GHz)

7.94 telsem_atlas_module Module Reference

Module for users to use TELSEM land surface emissivity data sets by CSEM interfaces.

Functions/Subroutines

- integer function, public **telsem_atlas_setup** (imonth, path, mw_atlas_ver)
- integer function, public **telsem_atlas_emiss** (Frequency, Angle, Latitude, Longitude, imonth, Emissivity_H, Emissivity_V, resolution, emis_std_v, emis_std_h, emis_cov, stype)
- integer function, public **telsem_atlas_emiss_nchannels** (Frequency, Angle, Latitude, Longitude, imonth, crtm_polar_idx, n_Channels, emissivity, resolution, emis_std, emis_cov, stype)
- subroutine, public **telsem_atlas_close** ()
- logical function, public **telsem_atlas_initialized** (imonth)

7.94.1 Detailed Description

Module for users to use TELSEM land surface emissivity data sets by CSEM interfaces.

TELSEM includes the monthly land surface emissivity atlas based on the multiple-year retrievals from SSMI and some trievals from other sensors. TELSEM is a generalized atlas which means it may be applicable for different sensors besides SSMI.

The interfacing follows the general CSEM design where each emissivity model is required to implement two interfaces with one to provide the h-pol and v-pol emissivity values of a single frequecy and the other to provide the emissivity values of all the channels of a specific sensor.

7.95 telsem_atlas_reader Module Reference

Data and routines for MW emissivity atlas

Functions/Subroutines

- integer function, public **load_telsem_atlas** (dir, month, lat1, lat2, lon1, lon2)
- subroutine, public **close_telsem_atlas**
- subroutine, public **emis_interp_ind_sing** (lat, lon, theta, freq, ev, eh, stdv, stdh, covvh, verb)
- subroutine, public **emis_interp_ind_mult** (lat, lon, theta, freq, n_chan, ev, eh, std, verb, stype)
- subroutine, public **emis_interp_int_sing** (lat, lon, resol, theta, freq, ev, eh, stdv, stdh, covvh, verb)
- subroutine, public **emis_interp_int_mult** (lat, lon, resol, theta, freq, n_chan, ev, eh, std, verb, stype)

Variables

- type(telsem_atlas), save, public **atlas**
- integer, public **telsem_atlas_version** =100

7.95.1 Detailed Description

Data and routines for MW emissivity atlas

7.96 uwir_atlas_module Module Reference

Module for users to use UWIR land surface emissivity data sets by CSEM interfaces.

Functions/Subroutines

- integer function, public **uwir_atlas_setup** (imonth, path, mw_atlas_ver)
- integer function, public **uwir_atlas_emiss** (Wavenumber, Latitude, Longitude, imonth, Emissivity, emis_cov, stype)
- integer function, public **uwir_atlas_emiss_nchannels** (Wavenumber, Latitude, Longitude, imonth, n_↩ Channels, emissivity, emis_cov, stype)
- subroutine, public **uwir_atlas_close** ()
- logical function, public **uwir_atlas_initialized** (imonth)

7.96.1 Detailed Description

Module for users to use UWIR land surface emissivity data sets by CSEM interfaces.

7.97 uwir_atlas_reader Module Reference

Data and routines for UWIR emissivity atlas.

Functions/Subroutines

- integer function, public **crtm_uwiremis_init** (path, imonth)
- subroutine, public **crtm_uwiremis** (nchs, lat, lon, surfacetype, snowfrac, instr_wavenum, instr_emis, instr↩ _emis_cov, instr_emis_flag)
- subroutine, public **csem_uwiremis_multi** (instr_wavenum, lat, lon, surfacetype, nchs, instr_emis, instr_↩ emis_cov, instr_emis_flag)
- subroutine, public **csem_uwiremis_single** (instr_wavenum, lat, lon, surfacetype, instr_emis, instr_emis_cov, instr_emis_flag)
- subroutine, public **crtm_uwiremis_close_atlas** ()

Variables

- integer, parameter, public **surftype_land** = 0
- integer, parameter, public **surftype_sea** = 1
- integer, parameter, public **surftype_seaice** = 2
- integer, parameter, public **surftype_snow** = 4
- integer, public **ir_atlas_version** =100

7.97.1 Detailed Description

Data and routines for UWIR emissivity atlas.

Chapter 8

Data Type Documentation

8.1 csem_define::csem_atmosphere_parameters Type Reference

Public Attributes

- real(fp) **downward_atm_radiance** = 0.0_fp
- real(fp) **transmittance** = 0.0_fp
- real(fp) **downward_solar_irradiance** = 0.0_fp

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

- src/CSEM_Define/CSEM_Struct_Define.f90

8.2 csem_define::csem_geoinfo_struct Type Reference

Public Attributes

- real(fp) **latitude** = 0.0_fp
- real(fp) **longitude** = 0.0_fp
- integer **year** = 2001
- integer **month** = 1
- integer **day** = 1
- integer **hour** = 1

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

- src/CSEM_Define/CSEM_Struct_Define.f90

8.3 csem_define::csem_ice_surface Type Reference

Public Attributes

- integer **ice_type** = 1
- real(fp) **ice_temperature** = 263.0_fp
- real(fp) **ice_thickness** = 10.0_fp
- real(fp) **ice_density** = 0.9_fp
- real(fp) **ice_roughness** = 0.0_fp
- real(fp) **salinity** = 33.0_fp

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

- src/CSEM_Define/CSEM_Struct_Define.f90

8.4 csem_define::csem_land_surface Type Reference

Public Member Functions

- PROCEDURE **init** =>alloc_soil_profile
- FINAL **clean_land**

Public Attributes

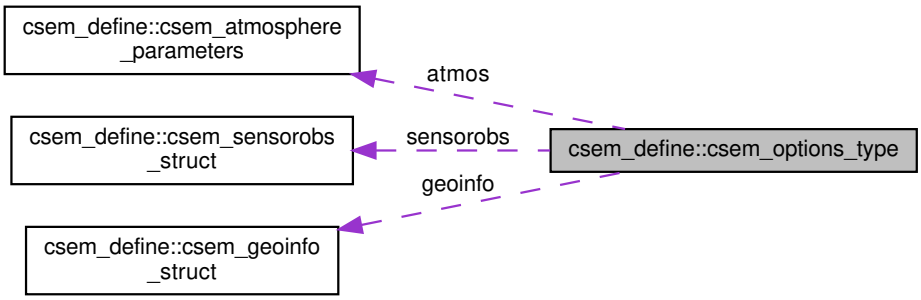
- integer **land_cover_type** = 1
- integer **vegetation_type** = 1
- integer **soil_type** = 1
- real(fp) **vegetation_fraction** = 0.3_fp
- real(fp) **land_skin_temperature** = 283.0_fp
- real(fp) **top_soil_temperature** = 283.0_fp
- real(fp) **top_soil_moisture** = 0.05_fp
- real(fp) **lai** = 3.5
- real(fp) **canopy_water_content** = 0.05_fp
- integer **n_soil_layers** = 0
- logical **is_allocated** = .FALSE.
- real(fp), dimension(:), allocatable **temperature_profile**
- real(fp), dimension(:), allocatable **moisture_profile**
- real(fp), dimension(:), allocatable **soil_depth**

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

- src/CSEM_Define/CSEM_Struct_Define.f90

8.5 csem_define::csem_options_type Type Reference

Collaboration diagram for csem_define::csem_options_type:



Public Attributes

- type(csem_sensorobs_struct) **sensorobs**
- type(csem_atmosphere_parameters) **atmos**
- type(csem_geoinfo_struct) **geoinfo**

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

- src/CSEM_Define/CSEM_Struct_Define.f90

8.6 csem_define::csem_sensorobs_struct Type Reference

Public Member Functions

- PROCEDURE, pass(self) **init** => alloc_sensorobs
- FINAL **clean_sensorobs**

Public Attributes

- character(len=100) **sensor_id** = ' '
- logical **is_allocated** = .FALSE.
- integer **n_channels** = 0
- real(fp), dimension(:), allocatable **channel_frequency**
- integer, dimension(:), allocatable **channel_polarization**
- real(fp), dimension(:), allocatable **tb**

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

- src/CSEM_Define/CSEM_Struct_Define.f90

8.7 csem_define::csem_sfcoptics_type Type Reference

Public Member Functions

- PROCEDURE, pass(self) **init** => init_sfcoptics
- FINAL **clean_sfcoptics**

Public Attributes

- logical **is_allocated** = .FALSE.
- logical **is_solar** = .FALSE.
- logical **is_spectral** = .FALSE.
- real(fp) **frequency**
- real(fp) **wavenumber**
- real(fp) **source_zenith_angle** = 0.0_fp
- real(fp) **source_azimuth_angle** = 0.0_fp
- real(fp) **sensor_zenith_angle** = 0.0_fp
- real(fp) **sensor_scan_angle** = 0.0_fp
- real(fp) **sensor_azimuth_angle** = 0.0_fp
- real(fp) **relative_azimuth_angle** = 0.0_fp
- integer **n_angles** = 1
- integer **n_stokes** = 4
- integer **mth_azi** = 0
- real(fp), dimension(:), allocatable **angle**
- real(fp), dimension(:), allocatable **weight**
- real(fp), dimension(:,:), allocatable **emissivity**
- real(fp), dimension(:,:), allocatable **direct_reflectivity**
- real(fp), dimension(:,:,:,:), allocatable **reflectivity**

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

- src/CSEM_Define/CSEM_Struct_Define.f90

8.8 csem_define::csem_snow_surface Type Reference

Public Attributes

- integer **snow_type** = 1
- real(fp) **snow_temperature** = 263.0_fp
- real(fp) **snow_depth** = 50.0_fp
- real(fp) **snow_density** = 0.2_fp
- real(fp) **snow_grain_size** = 2.0_fp
- integer **soil_type** = 1
- real(fp) **top_soil_temperature** = 283.0_fp
- real(fp) **top_soil_moisture_content** = 0.05_fp
- integer **vegetation_type** = 1
- real(fp) **lai** = 3.5

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

- src/CSEM_Define/CSEM_Struct_Define.f90

8.9 csem_define::csem_water_surface Type Reference

Public Attributes

- integer **water_type** = 1
- real(fp) **water_temperature** = 283.0_fp
- real(fp) **wind_speed** = 5.0_fp
- real(fp) **wind_direction** = 0.0_fp
- real(fp) **salinity** = 33.0_fp
- real(fp) **foam_fraction** = 0.0_fp

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

- src/CSEM_Define/CSEM_Struct_Define.f90

8.10 csem_fresnel::fresnel_reflectance Interface Reference

Public Member Functions

- subroutine **fresnel_reflectance_1** (em1, em2, theta_i, theta_t, rv, rh)
- subroutine **fresnel_reflectance_2** (em1, em2, theta_i, rv, rh)

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

- src/CSEM_Utility/CSEM_Fresnel.f90

8.11 csem_fresnel::fresnel_reflectance_ad Interface Reference

Public Member Functions

- subroutine **fresnel_reflectance_ad_1** (em1, em2, theta_i, theta_t, em1_AD, em2_AD, rv_AD, rh_AD)
- subroutine **fresnel_reflectance_ad_2** (em1, em2, theta_i, em1_AD, em2_AD, rv_AD, rh_AD)

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

- src/CSEM_Utility/CSEM_Fresnel.f90

8.12 csem_fresnel::fresnel_reflectance_tl Interface Reference

Public Member Functions

- subroutine **fresnel_reflectance_tl_1** (em1, em2, theta_i, theta_t, em1_TL, em2_TL, rv_TL, rh_TL)
- subroutine **fresnel_reflectance_tl_2** (em1, em2, theta_i, em1_TL, em2_TL, rv_TL, rh_TL)

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

- src/CSEM_Utility/CSEM_Fresnel.f90

8.13 rttov_tessem_mod::tessem_net Type Reference

Public Attributes

- real(jprb), dimension(tessem_ncache) **b1**
- real(jprb), dimension(tessem_nout) **b2**
- real(jprb), dimension(tessem_ncache, tessem_nin) **w1**
- real(jprb), dimension(tessem_nout, tessem_ncache) **w2**
- real(jprb), dimension(tessem_nin) **x_min**
- real(jprb), dimension(tessem_nin) **x_max**
- real(jprb), dimension(tessem_nout) **y_min**
- real(jprb), dimension(tessem_nout) **y_max**

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

- src/MW/Water/RTTOV_FASTEM/rttov_tessem_mod.F90

Chapter 9

File Documentation

9.1 src/MW/Ice/CSEM_IceMW_SfcOptics.f90 File Reference

CSEM_IceMW_SfcOptics.f90.

Modules

- module csem_icemw_sfcoptics

  Container module for all the MW_ICE models available in the CSEM model repository.

Functions/Subroutines

- integer function, public csem_icemw_sfcoptics::csem_compute_icemw_sfcoptics (Surface, SfcOptics, Options)

  PURPOSE: Function to compute the sea-ice surface emissivity and reflectivity at microwave frequencies.
- integer function, public csem_icemw_sfcoptics::csem_compute_icemw_sfcoptics_tl (CSEM_SfcOptics_TL)

  PURPOSE: Function to compute the ice surface emissivity and reflectivity tangent-linear at microwave frequencies.
- integer function, public csem_icemw_sfcoptics::csem_compute_icemw_sfcoptics_ad (CSEM_Surface_AD)

  PURPOSE: Function to compute the ice surface emissivity and reflectivity adjoint at microwave frequencies.

9.1.1 Detailed Description

CSEM_IceMW_SfcOptics.f90.

9.2 src/MW/Land/CSEM_LandMW_SfcOptics.f90 File Reference

CSEM_LandMW_SfcOptics.f90.

Modules

- module csem_landmw_sfcoptics

  Container module with all the MW_LAND models available in the CSEM model repository.

Functions/Subroutines

- integer function, public csem_landmw_sfcoptics::csem_compute_landmw_sfcoptics (Surface, SfcOptics, Options, iVar)

  PURPOSE: Function to compute the land surface emissivity and reflectivity at microwave frequencies.
- subroutine, public **csem_landmw_sfcoptics::get_ref_index** (Frequency, Polarization, i_ref_h, i_ref_v)
- integer function, public csem_landmw_sfcoptics::csem_compute_landmw_sfcoptics_tl (Surface_TL, Sfc↩ Optics_TL, iVar)

  PURPOSE: Function to compute the land surface emissivity and reflectivity tangent-linear at microwave frequencies.
- integer function, public csem_landmw_sfcoptics::csem_compute_landmw_sfcoptics_ad (SfcOptics_AD, Surface_AD, iVar)

  PURPOSE: Function to compute the land surface emissivity and reflectivity adjoint at microwave frequencies.

9.2.1 Detailed Description

CSEM_LandMW_SfcOptics.f90.

9.3 src/MW/Land/MW_Canopy_Optics.f90 File Reference

MW_Canopy_Optics.f90.

Modules

- module mw_canopy_optics

  Container Module to compute the canopy optical properties at microwave frequencies.

Functions/Subroutines

- subroutine, public mw_canopy_optics::crtm_canopymw_optics (lai, leaf_refl, leaf_trans, g, ssalb, tau, iVar)

  PURPOSE: Subroutine to compute the canopy optical properties of land surface at microwave frequencies.
- subroutine, public mw_canopy_optics::crtm_canopymw_optics_tl (LAI_TL, ssalb_TL, tau_TL, iVar)

  PURPOSE: Tangent-linear mode of CRTM_CanopyMW_Optics.
- subroutine, public mw_canopy_optics::crtm_canopymw_optics_ad (LAI_AD, ssalb_AD, tau_AD, iVar)

  PURPOSE: Adjoint mode of CRTM_CanopyMW_Optics.

9.3.1 Detailed Description

MW_Canopy_Optics.f90.

9.4 src/MW/Land/MW_Leaf_Optics.f90 File Reference

MW_Leaf_Optics.f90.

Modules

- module mw_leaf_optics

  Container Module to compute the leaf optical properties of LAND surfaces at microwave frequencies.

Functions/Subroutines

- subroutine, public **mw_leaf_optics::csem_leafmw_optics** (frequency, angle, mge, refl, trans, eveg, iVar)
- subroutine, public mw_leaf_optics::crtm_leafmw_optics (frequency, theta, esv, d, rh, rv, th, tv)

  PURPOSE: Function to calculate v-pol and h-pol refelectance and trasmittance of one single leaf at microwave frequency.
- subroutine, public mw_leaf_optics::mean_leafmw_optics (frequency, eveg, leaf_thick, rh, rv, th, tv)

  PURPOSE: Function to calculate averaged refelectance and trasmittance of one single leaf at microwave frequency. Leaves are taken as individual scatters of a canopy. The averaged refelectance and trasmittance is used by canopy-level scattering model.

9.4.1 Detailed Description

MW_Leaf_Optics.f90.

9.5 src/MW/Land/NESDIS_LandEM_Module.f90 File Reference

NESDIS_LandEM_Module.f90.

Modules

- module nesdis_landem_module

  Module containing the old-version NESDIS microwave land emissivity model.

Functions/Subroutines

- subroutine, public nesdis_landem_module::nesdis_landem_213 (Angle, Frequency, Soil_Moisture_Content, Vegetation_Fraction, Soil_Temperature, t_skin, Lai, Soil_Type, Vegetation_Type, Emissivity_H, Emissivity_V)
- subroutine, public **nesdis_landem_module::nesdis_landem_old** (Angle, Frequency, Soil_Moisture_↩ Content, Vegetation_Fraction, Soil_Temperature, t_skin, Lai, Soil_Type, Vegetation_Type, Emissivity_H, Emissivity_V)

9.5.1 Detailed Description

NESDIS_LandEM_Module.f90.

9.6 src/MW/Land/NESDIS_LandMW_PhyModel.f90 File Reference

NESDIS_LandMW_PhyModel.f90.

Modules

- module nesdis_landmw_phymodel

  Module of the physics-based microwave land surface emissivity model.

Functions/Subroutines

- integer function, public nesdis_landmw_phymodel::nesdis_landmw_emiss (Frequency, Angle, Land_Skin_↩ Temperature, Soil_Temperature, Soil_Moisture_Content, Vegetation_Fraction, LAI, Vegetation_Type, Soil_↩ Type, Emissivity_H, Emissivity_V, iVar)
- integer function, public nesdis_landmw_phymodel::nesdis_landmw_emiss_tl (Land_Skin_Temperature_TL, Soil_Temperature_TL, Soil_Moisture_Content_TL, Vegetation_Fraction_TL, Emissivity_H_TL, Emissivity_↩ V_TL, iVar)

  PURPOSE: Tangent-linear mode of NESDIS_LandMW_Emiss.
- integer function, public nesdis_landmw_phymodel::nesdis_landmw_emiss_ad (Land_Skin_Temperature_AD, Soil_Temperature_AD, Soil_Moisture_Content_AD, Vegetation_Fraction_AD, Emissivity_H_AD, Emissivity↩ _V_AD, iVar)

  PURPOSE: Adjoint mode of NESDIS_LandMW_Emiss.
- subroutine, public nesdis_landmw_phymodel::two_stream_solution (emiss, iVar)

  Two stream RT solver of three-layer MW land surface physical model.
- subroutine, public nesdis_landmw_phymodel::two_stream_solution_tl (ssalb_TL, tau_TL, r23_TL, Tskin_TL, Tsoil_TL, emiss_TL, iVar)

  PURPOSE: Tangent-linear mode of the Two_Stream_Solution.
- subroutine, public nesdis_landmw_phymodel::two_stream_solution_ad (ssalb_AD, tau_AD, r23_AD, Tskin↩ _AD, Tsoil_AD, emiss_AD, iVar)

  PURPOSE: Adjoint mode of the Two_Stream_Solution.

9.6.1 Detailed Description

NESDIS_LandMW_PhyModel.f90.

9.7 src/MW/LUT_Atlas/CNRM_Atlas_Module.f90 File Reference

CSEM_CNRM_Atlas.f90.

Modules

- module cnrm_atlas_module

  Module for users to use CNRM land surface emissivity data sets by CSEM interfaces.

Functions/Subroutines

- integer function, public **cnrm_atlas_module::cnrm_atlas_setup** (imonth, path, Atlas_ID, mw_atlas_ver)
- integer function, public **cnrm_atlas_module::cnrm_atlas_emiss** (Frequency, Angle, Latitude, Longitude, imonth, Emissivity_H, Emissivity_V, stype)
- integer function, public **cnrm_atlas_module::cnrm_atlas_emiss_nchannels** (Frequency, Angle, Latitude, Longitude, imonth, n_Channel, emissivity, stype)
- logical function, public **cnrm_atlas_module::cnrm_atlas_initialized** (imonth)
- subroutine, public **cnrm_atlas_module::cnrm_atlas_close** ()

9.7.1 Detailed Description

CSEM_CNRM_Atlas.f90.

9.8 src/MW/LUT_Atlas/CNRM_Atlas_Reader.f90 File Reference

CNRM_Atlas_Reader.f90.

Modules

- module cnrm_amsua_reader

  Module containing Data and routines for MW emissivity atlas METEO-FRANCE CNRM

Functions/Subroutines

- integer function, public **cnrm_amsua_reader::cnrm_amsua_setup** (path, imonth)
- integer function, public **cnrm_amsua_reader::cnrm_amsua_emiss** (latitude, longitude_in, frequency, zenangle, emissivity_v, emissivity_h, pbats_veg)
- integer function, public **cnrm_amsua_reader::cnrm_amsua_emiss_multi** (latitude, longitude_in, frequency, zenangle, n_Channel, emissivity, pbats_veg)

Variables

- integer, public **cnrm_amsua_reader::cnrm_amsua_version** =200

9.8.1 Detailed Description

CNRM_Atlas_Reader.f90.

9.9 src/MW/LUT_Atlas/TELSEM2_Atlas_Reader.f90 File Reference

Subroutines for TELSEM2 MW emissivity atlas and interpolator.

Modules

- module telsem2_atlas_reader

  Subroutines for TELSEM2 MW emissivity atlas and interpolator.

Functions/Subroutines

- subroutine, public telsem2_atlas_reader::test_inputs (month, lat, lon, theta, freq)

  Subroutine to check input variables: not used by RTTOV.
- subroutine, public telsem2_atlas_reader::rttov_readmw_atlas (dir, month, atlas, verbose, err, lat1, lat2, lon1, lon2)

  Initialise a TELSEM2 atlas data structure. Atlas data may be initialised for a region of the globe defined by the lower and upper lat/lon limits, though this feature is not used by RTTOV.
- integer function, public **telsem2_atlas_reader::load_telsem2_atlas** (dir, month, lat1, lat2, lon1, lon2)
- subroutine, public telsem2_atlas_reader::rttov_closemw_atlas ()

  Deallocate data in TELSEM2 atlas data structure.
- subroutine, public telsem2_atlas_reader::emis_interp_ind_sing (lat, lon, theta, freq, ev, eh, stdv, stdh, covvh, verb)

  Return emissivities for a single channel at the native atlas resolution.
- subroutine, public telsem2_atlas_reader::emis_interp_ind_mult (lat, lon, theta, freq, n_chan, ev, eh, std, verb)

  Return emissivities for multiple channels at the native atlas resolution. Each dimension of the covariance matrix std(:,:) has V-pol values for all channels followed by H-pol values for all channels.
- subroutine, public telsem2_atlas_reader::emis_interp_int_sing (lat, lon, resol, theta, freq, ev, eh, stdv, stdh, covvh, verb)

  Return emissivities for a single channel at the user-specified resolution.
- subroutine, public telsem2_atlas_reader::emis_interp_int_mult (lat, lon, resol, theta, freq, n_chan, ev, eh, std, verb)

  Return emissivities for multiple channels at the user-specified resolution. Each dimension of the covariance matrix std(:,:) has V-pol values for all channels followed by H-pol values for all channels.

Variables

- type(telsem2_atlas_data), save, public **telsem2_atlas_reader::atlas2**

9.9.1 Detailed Description

Subroutines for TELSEM2 MW emissivity atlas and interpolator.

9.10 src/MW/LUT_Atlas/TELSEM_Atlas_Module.f90 File Reference

TELSEM_Atlas_Module.f90.

Modules

- module telsem_atlas_module

  Module for users to use TELSEM land surface emissivity data sets by CSEM interfaces.

Functions/Subroutines

- integer function, public **telsem_atlas_module::telsem_atlas_setup** (imonth, path, mw_atlas_ver)
- integer function, public **telsem_atlas_module::telsem_atlas_emiss** (Frequency, Angle, Latitude, Longitude, imonth, Emissivity_H, Emissivity_V, resolution, emis_std_v, emis_std_h, emis_cov, stype)
- integer function, public **telsem_atlas_module::telsem_atlas_emiss_nchannels** (Frequency, Angle, Latitude, Longitude, imonth, crtm_polar_idx, n_Channels, emissivity, resolution, emis_std, emis_cov, stype)
- subroutine, public **telsem_atlas_module::telsem_atlas_close** ()
- logical function, public **telsem_atlas_module::telsem_atlas_initialized** (imonth)

9.10.1 Detailed Description

TELSEM_Atlas_Module.f90.

9.11 src/MW/LUT_Atlas/TELSEM_Atlas_Reader.f90 File Reference

TELSEM_Atlas_Reader.f90.

Modules

- module telsem_atlas_reader

  Data and routines for MW emissivity atlas

Functions/Subroutines

- integer function, public **telsem_atlas_reader::load_telsem_atlas** (dir, month, lat1, lat2, lon1, lon2)
- subroutine, public **telsem_atlas_reader::close_telsem_atlas**
- subroutine, public **telsem_atlas_reader::emis_interp_ind_sing** (lat, lon, theta, freq, ev, eh, stdv, stdh, covvh, verb)
- subroutine, public **telsem_atlas_reader::emis_interp_ind_mult** (lat, lon, theta, freq, n_chan, ev, eh, std, verb, stype)
- subroutine, public **telsem_atlas_reader::emis_interp_int_sing** (lat, lon, resol, theta, freq, ev, eh, stdv, stdh, covvh, verb)
- subroutine, public **telsem_atlas_reader::emis_interp_int_mult** (lat, lon, resol, theta, freq, n_chan, ev, eh, std, verb, stype)

Variables

- type(telsem_atlas), save, public **telsem_atlas_reader::atlas**
- integer, public **telsem_atlas_reader::telsem_atlas_version** =100

9.11.1 Detailed Description

TELSEM_Atlas_Reader.f90.

9.12 src/MW/Snow/CSEM_SnowMW_SfcOptics.f90 File Reference

CSEM_SnowMW_SfcOptics.f90.

Modules

- module csem_snowmw_sfcoptics

  This module provides a generic interface for the upper-level applications to access all the MW_SNOW models available in the CSEM model repository.

Functions/Subroutines

- integer function, public csem_snowmw_sfcoptics::csem_compute_snowmw_sfcoptics (Surface, SfcOptics, Options)

  PURPOSE: Function to compute the snow surface emissivity and reflectivity at microwave frequencies.
- integer function, public csem_snowmw_sfcoptics::csem_compute_snowmw_sfcoptics_tl (CSEM_Sfc↩ Optics_TL)

  PURPOSE: Function to compute the snow surface emissivity and reflectivity tangent-linear at microwave frequencies.
- integer function, public csem_snowmw_sfcoptics::csem_compute_snowmw_sfcoptics_ad (CSEM_Surface↩ _AD)

  PURPOSE: Function to compute the Snowsurface emissivity and reflectivity adjoint at microwave frequencies.

9.12.1 Detailed Description

CSEM_SnowMW_SfcOptics.f90.

9.13 src/MW/Soil/MW_Soil_Optics.f90 File Reference

MW_Soil_Optics.f90.

Modules

- module mw_soil_optics

  Container module with all the MW soil models available in the CSEM model repository.

Functions/Subroutines

- subroutine, public mw_soil_optics::csem_soilmw_optics (frequency, theta, Tskin, Tsoil, smc, sand, clay, refl↩ _smooth, teff, iVar)

  PURPOSE: Evaluation of the bare-soil optical parameters at microwave frequencies.
- subroutine, public mw_soil_optics::csem_soilmw_optics_tl (Tskin_TL, Tsoil_TL, smc_TL, refl_h_TL, refl_v↩ _TL, teff_TL, iVar)

  PURPOSE: Tangent-linear mode of CSEM_SoilMW_Optics.
- subroutine, public mw_soil_optics::csem_soilmw_optics_ad (Tskin_AD, Tsoil_AD, smc_AD, refl_h_AD, refl↩ _v_AD, teff_AD, iVar)

  PURPOSE: Tangent-linear mode of CSEM_SoilMW_Optics.

Variables

- integer, parameter, public **mw_soil_optics::max_soil_layers** = 1

9.13.1 Detailed Description

MW_Soil_Optics.f90.

9.14 src/MW/Water/CRTM_FASTEM/Azimuth_Emissivity_F6_Module.f90 File Reference

Azimuth_Emissivity_F6_Module.f90.

Modules

- module azimuth_emissivity_f6_module

  Azimuthal functions of the FASTEM-6 model

Functions/Subroutines

- subroutine, public **azimuth_emissivity_f6_module::azimuth_emissivity_f6** (AZCoeff, Wind_Speed, Azimuth_Angle, Frequency, Zenith_Angle, e_Azimuth, iVar)
- subroutine, public azimuth_emissivity_f6_module::azimuth_emissivity_f6_tl (AZCoeff, Wind_Speed_TL, Azimuth_Angle_TL, e_Azimuth_TL, iVar)
- subroutine, public azimuth_emissivity_f6_module::azimuth_emissivity_f6_ad (AZCoeff, e_Azimuth_AD, Wind_Speed_AD, Azimuth_Angle_AD, iVar)

9.14.1 Detailed Description

Azimuth_Emissivity_F6_Module.f90.

9.15 src/MW/Water/CRTM_FASTEM/Azimuth_Emissivity_Module.f90 File Reference

Azimuth_Emissivity_Module.f90.

Modules

- module azimuth_emissivity_module

  Azimuthal emissivity subroutines of old FASTEM versons.

Functions/Subroutines

- subroutine, public azimuth_emissivity_module::azimuth_emissivity (AZCoeff, Wind_Speed, Azimuth_Angle, Frequency, cos_z, e_Azimuth, iVar)
- subroutine, public azimuth_emissivity_module::azimuth_emissivity_tl (AZCoeff, Wind_Speed_TL, Azimuth↩ _Angle_TL, e_Azimuth_TL, iVar)
- subroutine, public azimuth_emissivity_module::azimuth_emissivity_ad (AZCoeff, e_Azimuth_AD, Wind_↩ Speed_AD, Azimuth_Angle_AD, iVar)

9.15.1 Detailed Description

Azimuth_Emissivity_Module.f90.

9.16 src/MW/Water/CRTM_FASTEM/CRTM_Fastem1.f90 File Reference

CRTM_Fastem1.f90.

Modules

- module crtm_fastem1

  Module with the old Fastem procedures.

Functions/Subroutines

- subroutine, public **crtm_fastem1::fastem1** (Frequency, Sat_Zenith_Angle, SST, Wind_Speed, Emissivity, dEH_dWindSpeed, dEV_dWindSpeed)

9.16.1 Detailed Description

CRTM_Fastem1.f90.

9.17 src/MW/Water/CRTM_FASTEM/CRTM_FASTEM_MODULE.f90 File Reference

CRTM_FASTEM_MODULE.f90.

Modules

- module crtm_fastem_module

  Container module with all the existing CRTM FASTEM versions.

Functions/Subroutines

- integer function, public crtm_fastem_module::crtm_fastem_emiss (Frequency, Angle, Water_Temperature, Salinity, Wind_Speed, Wind_Direction, Emissivity, Reflectivity, FASTEM_Version, Sensor_Azimuth_Angle, Transmittance)
- integer function, public crtm_fastem_module::compute_fastem_sfcoptics (Frequency, Angles, Water↩ _Temperature, Salinity, Wind_Speed, Wind_Direction, iVar, Emissivity, Reflectivity, FASTEM_Version, Sensor_Azimuth_Angle, Transmittance)
- integer function, public crtm_fastem_module::compute_fastem_sfcoptics_tl (Water_Temperature_TL, Salinity_TL, Wind_Speed_TL, Wind_Direction_TL, Transmittance_TL, iVar, Emissivity_TL, Reflectivity_↩ TL, FASTEM_Version)
- integer function, public crtm_fastem_module::compute_fastem_sfcoptics_ad (Emissivity_AD, Reflectivity↩ _AD, Water_Temperature_AD, Salinity_AD, Wind_Speed_AD, Wind_Direction_AD, Transmittance_AD, iVar, FASTEM_Version)
- integer function, public crtm_fastem_module::crtm_fastem_init (MWwaterCoeff_File, Version)
- integer function, public **crtm_fastem_module::crtm_fastem_destroy** ()

Variables

- logical, save, public **crtm_fastem_module::csem_mwwatercoeff_init** = .FALSE.

9.17.1 Detailed Description

CRTM_FASTEM_MODULE.f90.

9.18 src/MW/Water/CRTM_FASTEM/CRTM_FastemXX.f90 File Reference

CRTM_FastemXX.f90.

Modules

- module crtm_fastemxx

  Container Module for the Fastem4/5/6 models.

Functions/Subroutines

- subroutine, public crtm_fastemxx::compute_fastemxx (MWwaterCoeff, Frequency, n_Angles, Zenith_Angle, Temperature, Salinity, Wind_Speed, iVar, Emissivity, Reflectivity, Azimuth_Angle, Transmittance)
- subroutine, public crtm_fastemxx::compute_fastemxx_tl (MWwaterCoeff, Temperature_TL, Salinity_TL, Wind_Speed_TL, iVar, Emissivity_TL, Reflectivity_TL, Azimuth_Angle_TL, Transmittance_TL)
- subroutine, public crtm_fastemxx::compute_fastemxx_ad (MWwaterCoeff, Emissivity_AD, Reflectivity_AD, iVar, Temperature_AD, Salinity_AD, Wind_Speed_AD, Azimuth_Angle_AD, Transmittance_AD)

9.18.1 Detailed Description

CRTM_FastemXX.f90.

9.19 src/MW/Water/CRTM_FASTEM/CRTM_LowFrequency_MWSSEM.f90 File Reference

CRTM_LowFrequency_MWSSEM.f90.

Modules

- module crtm_lowfrequency_mwssem

  Module containg subroutines to compute microwave ocean emissivity components (FWD, TL, and AD) for low frequencies.

Functions/Subroutines

- subroutine, public crtm_lowfrequency_mwssem::lowfrequency_mwssem (Frequency, Zenith_Angle, Temperature, Salinity, Wind_Speed, Emissivity, iVar)
- subroutine, public **crtm_lowfrequency_mwssem::lowfrequency_mwssem_tl** (Temperature_TL, Salinity_TL, Wind_Speed_TL, Emissivity_TL, iVar)
- subroutine, public **crtm_lowfrequency_mwssem::lowfrequency_mwssem_ad** (Emissivity_AD, Temperature_AD, Salinity_AD, Wind_Speed_AD, iVar)

9.19.1 Detailed Description

CRTM_LowFrequency_MWSSEM.f90.

9.20 src/MW/Water/CRTM_FASTEM/CRTM_MWwaterCoeff_Define.f90 File Reference

CRTM_MWwaterCoeff_Define.f90.

Modules

- module crtm_mwwatercoeff_define

  Module defining the MWwaterCoeff object.

Functions/Subroutines

- pure logical function, public crtm_mwwatercoeff_define::crtm_mwwatercoeff_associated (self)
- pure subroutine, public **crtm_mwwatercoeff_define::crtm_mwwatercoeff_destroy** (self)
- pure subroutine, public **crtm_mwwatercoeff_define::crtm_mwwatercoeff_create** (self, ndim_subgrp, dims_subgrp)
- subroutine, public **crtm_mwwatercoeff_define::crtm_mwwatercoeff_inspect** (self, pause)
- logical function, public **crtm_mwwatercoeff_define::crtm_mwwatercoeff_validrelease** (self)
- subroutine, public **crtm_mwwatercoeff_define::crtm_mwwatercoeff_info** (self, Info)
- subroutine, public **crtm_mwwatercoeff_define::crtm_mwwatercoeff_defineversion** (Id)

9.20.1 Detailed Description

CRTM_MWwaterCoeff_Define.f90.

9.21 src/MW/Water/CRTM_FASTEM/CRTM_MWwaterLUT_Define.f90 File Reference

CRTM_MWwaterLUT_Define.f90.

Modules

- module crtm_mwwaterlut_define

  Module defining the MWwaterLUT object containing the Look-Up Table (LUT) for the microWave (MW) sea surface emissivity model.

Functions/Subroutines

- pure logical function, public crtm_mwwaterlut_define::mwwaterlut_associated (self)
- pure subroutine, public **crtm_mwwaterlut_define::mwwaterlut_destroy** (self)
- pure subroutine, public **crtm_mwwaterlut_define::mwwaterlut_create** (self, n_Angles, n_Frequencies, n↩_Temperatures, n_Wind_Speeds)
- subroutine, public **crtm_mwwaterlut_define::mwwaterlut_inspect** (self, pause)
- logical function, public **crtm_mwwaterlut_define::mwwaterlut_validrelease** (self)
- subroutine, public **crtm_mwwaterlut_define::mwwaterlut_info** (self, Info)
- subroutine, public **crtm_mwwaterlut_define::mwwaterlut_defineversion** (Id)

9.21.1 Detailed Description

CRTM_MWwaterLUT_Define.f90.

9.22 src/MW/Water/CRTM_FASTEM/Foam_Utility_Module.f90 File Reference

Foam_Utility_Module.f90.

Modules

- module foam_utility_module

  Helper module containing the foam-related utility routines for the CRTM implementation of FASTEM4 and FASTEM5.

Functions/Subroutines

- subroutine, public foam_utility_module::foam_coverage (FCCoeff, wind_speed, coverage)
- subroutine, public **foam_utility_module::foam_coverage_tl** (FCCoeff, wind_speed, wind_speed_TL, coverage_TL)
- subroutine, public **foam_utility_module::foam_coverage_ad** (FCCoeff, wind_speed, coverage_AD, wind↩_speed_AD)
- subroutine, public foam_utility_module::foam_reflectivity (FRCoeff, Zenith_Angle, Frequency, Rv, Rh)

9.22.1 Detailed Description

Foam_Utility_Module.f90.

9.23 src/MW/Water/CRTM_FASTEM/Large_Scale_Correction_Module.f90 File Reference

Large_Scale_Correction_Module.f90.

Modules

- module large_scale_correction_module

  Module containing the large-scale correction procedures for the CRTM implementations of FASTEM4 and FASTEM5.

Functions/Subroutines

- subroutine, public large_scale_correction_module::large_scale_correction (LSCCoeff, Frequency, cos_↩ Z, Wind_Speed, Rv_Large, Rh_Large, iVar)
- subroutine, public large_scale_correction_module::large_scale_correction_tl (Wind_Speed_TL, Rv_Large↩ _TL, Rh_Large_TL, iVar)
- subroutine, public large_scale_correction_module::large_scale_correction_ad (Rv_Large_AD, Rh_Large_↩ AD, Wind_Speed_AD, iVar)

9.23.1 Detailed Description

Large_Scale_Correction_Module.f90.

9.24 src/MW/Water/CRTM_FASTEM/Liu.f90 File Reference

Liu.f90.

Modules

- module liu

  Liu Ocean Permittivity module.

Functions/Subroutines

- subroutine, public liu::liu_ocean_permittivity (Temperature, Salinity, Frequency, Permittivity, iVar)

  PURPOSE: Subroutine to compute ocean permittivity according to the reference, Liu, Q. et al. (2010) An improved fast microwave water emissivity model. IEEE Trans. Geosci. Remote Sensing, accepted June 25, 2010.
- subroutine, public liu::liu_ocean_permittivity_tl (Temperature_TL, Salinity_TL, Frequency, Permittivity_TL, i↩ Var)

  PURPOSE: Subroutine to compute ocean permittivity according to the reference, Liu, Q. et al. (2010) An improved fast microwave water emissivity model. IEEE Trans. Geosci. Remote Sensing, accepted June 25, 2010.
- subroutine, public liu::liu_ocean_permittivity_ad (Permittivity_AD, Frequency, Temperature_AD, Salinity_AD, iVar)

  PURPOSE: Subroutine to compute ocean permittivity according to the reference, Liu, Q. et al. (2010) An improved fast microwave water emissivity model. IEEE Trans. Geosci. Remote Sensing, accepted June 25, 2010.

9.24.1 Detailed Description

Liu.f90.

9.25 src/MW/Water/CRTM_FASTEM/Ocean_Permittivity.f90 File Reference

Ocean_Permittivity.f90.

Modules

- module ocean_permittivity

  Container module for the sea water complex permittivity model collections.

9.25.1 Detailed Description

Ocean_Permittivity.f90.

9.26 src/MW/Water/CRTM_FASTEM/Small_Scale_Correction_Module.f90 File Reference

Small_Scale_Correction_Module.f90.

Modules

- module small_scale_correction_module

  Module containing the small-scale correction procedures for the CRTM implementations of FASTEM4 and FASTEM5.

Functions/Subroutines

- subroutine, public small_scale_correction_module::small_scale_correction (SSCCoeff, Frequency, cos_↩ Z, Wind_Speed, Correction, iVar)
- subroutine, public small_scale_correction_module::small_scale_correction_tl (SSCCoeff, Wind_Speed_TL, Correction_TL, iVar)
- subroutine, public small_scale_correction_module::small_scale_correction_ad (SSCCoeff, Correction_AD, Wind_Speed_AD, iVar)

9.26.1 Detailed Description

Small_Scale_Correction_Module.f90.

9.27 src/MW/Water/CSEM_WaterMW_SfcOptics.f90 File Reference

CSEM_WaterMW_SfcOptics.f90.

Modules

- module csem_watermw_sfcoptics

  Container module with all the MWWater models available in the CSEM model repository.

Functions/Subroutines

- integer function, public csem_watermw_sfcoptics::csem_compute_watermw_sfcoptics (Surface, SfcOptics, Options, iVar)

  PURPOSE: Function to compute the ocean surface emissivity and reflectivity at microwave frequencies.
- integer function, public csem_watermw_sfcoptics::csem_compute_watermw_sfcoptics_tl (Surface_TL, Atmos_TL, SfcOptics_TL, iVar)

  PURPOSE: Function to compute the ocean surface emissivity and reflectivity tangent-linear at microwave frequencies.
- integer function, public csem_watermw_sfcoptics::csem_compute_watermw_sfcoptics_ad (SfcOptics_AD, Surface_AD, Atmos_AD, iVar)

  PURPOSE: Function to compute the ocean surface emissivity and reflectivity adjoint at microwave frequencies.

9.27.1 Detailed Description

CSEM_WaterMW_SfcOptics.f90.

9.28 src/MW/Water/RTTOV_FASTEM/rttov_fastem5r1.F90 File Reference

Compute FASTEM-4,5,6 emissivity and reflectance for a single channel.

Modules

- module rttov_fastem5r1_module

  Compute RTTOV FASTEM-4,5,6 emissivity and reflectance for a single channel.

Functions/Subroutines

- subroutine rttov_fastem5r1_module::rttov_fastem5r1 (fastem_version, Frequency, Zenith_Angle, Temperature, Salinity, Wind_Speed, Emissivity, Reflectivity, Transmittance, Rel_Azimuth, Supply_Foam_Fraction, Foam_Fraction)

  Compute FASTEM-4,5,6 emissivity and reflectance for a single channel.

9.28.1 Detailed Description

Compute FASTEM-4,5,6 emissivity and reflectance for a single channel.

9.29 src/MW/Water/RTTOV_FASTEM/rttov_fastem5r1_ad.F90 File Reference

AD of FASTEM-4,5,6 emissivity and reflectance calculation.

Modules

- module rttov_fastem5r1_ad_module

  AD of RTTOV FASTEM-4,5,6 emissivity and reflectance calculation.

Functions/Subroutines

- subroutine **rttov_fastem5r1_ad_module::rttov_fastem5r1_ad** (fastem_version, Frequency, Zenith_↩ Angle, Temperature, Salinity, Wind_Speed, Emissivity_ad, Reflectivity_ad, Temperature_ad, Salinity_ad, Wind_Speed_ad, Emissivity, Reflectivity, Transmittance, Rel_Azimuth, Transmittance_ad, Rel_Azimuth_ad, Supply_Foam_Fraction, Foam_Fraction, Foam_Fraction_ad)

9.29.1 Detailed Description

AD of FASTEM-4,5,6 emissivity and reflectance calculation.

9.30 src/MW/Water/RTTOV_FASTEM/rttov_fastem5r1_coef.F90 File Reference

Contains data for the FASTEM-4,5,6 MW sea surface emissivity models.

Modules

- module mod_rttov_fastem5r1_coef

  Contains data for the FASTEM-4,5,6 MW sea surface emissivity models.

Variables

- real(fp), parameter, public **mod_rttov_fastem5r1_coef::zero** = 0.0_fp
- real(fp), parameter, public **mod_rttov_fastem5r1_coef::point_5** = 0.5_fp
- real(fp), parameter, public **mod_rttov_fastem5r1_coef::one** = 1.0_fp
- real(fp), parameter, public **mod_rttov_fastem5r1_coef::two** = 2.0_fp
- real(fp), parameter, public **mod_rttov_fastem5r1_coef::three** = 3.0_fp
- real(fp), parameter, public **mod_rttov_fastem5r1_coef::pi** = 3.14159265358979323846264338327 9_fp
- real(fp), parameter, public **mod_rttov_fastem5r1_coef::degrees_to_radians** = PI/180.0_fp
- real(fp), parameter, public **mod_rttov_fastem5r1_coef::transmittance_limit_lower** = 0.00001_fp
- real(fp), parameter, public **mod_rttov_fastem5r1_coef::transmittance_limit_upper** = 0.9999_fp
- real(fp), parameter, public **mod_rttov_fastem5r1_coef::e0_4** = 0.0088419_fp
- real(fp), parameter, public **mod_rttov_fastem5r1_coef::e0_5** = 0.00885418781762_fp
- real(fp), parameter, public **mod_rttov_fastem5r1_coef::min_f** = 1.4_fp

- real(fp), parameter, public **mod_rttov_fastem5r1_coef::max_f** = 200.0_fp
- real(fp), parameter, public **mod_rttov_fastem5r1_coef::min_wind** = 0.3_fp
- real(fp), parameter, public **mod_rttov_fastem5r1_coef::max_wind** = 35.0_fp
- real(fp), dimension(0:38), parameter, public **mod_rttov_fastem5r1_coef::a_coef** = (/ 3.8_fp, 0.0248033↩ _fp, 87.9181727_fp, -0.4031592248_fp, 0.0009493088010_fp, -0.1930858348E-05_fp, -0.002697_fp, -7.↩ 3E-06_fp, -8.9E-06_fp, 5.723_fp, 0.022379_fp, -0.00071237_fp, -6.28908E-03_fp, 1.76032E-04_fp, -9.↩ 22144E-05_fp, 0.1124465_fp, -0.0039815727_fp, 0.00008113381_fp, -0.00000071824242_fp, -2.39357E-03_fp, 3.1353E-05_fp, -2.52477E-07_fp, 0.003049979018_fp, -3.010041629E-05_fp, 0.4811910733E-05_↩ fp, -0.4259775841E-07_fp, 0.149_fp, -8.8E-04_fp, -1.05E-04_fp, 2.033E-02_fp, 1.266E-04_fp, 2.464E-06_fp, -1.849E-05_fp, 2.551E-07_fp, -2.551E-08_fp, 0.182521_fp, -1.46192E-03_fp, 2.09324E-05_fp, -1.28205E-07_fp/)
- real(fp), dimension(36), parameter, public **mod_rttov_fastem5r1_coef::lcoef5** = (/ -5.994667E-02_fp, 9.341346E-04_fp,-9.566110E-07_fp, 8.360313E-02_fp,-1.085991E-03_fp, 6.735338E-07_fp,-2.617296E-02_fp, 2.864495E-04_fp,-1.429979E-07_fp,-5.265879E-04_fp, 6.880275E-05_fp,-2.916657E-07_fp,-1.↩ 671574E-05_fp, 1.086405E-06_fp,-3.632227E-09_fp, 1.161940E-04_fp,-6.349418E-05_fp, 2.466556E-07_fp,-2.431811E-02_fp,-1.031810E-03_fp, 4.519513E-06_fp, 2.868236E-02_fp, 1.186478E-03_fp,-5.↩ 257096E-06_fp,-7.933390E-03_fp, -2.422303E-04_fp, 1.089605E-06_fp,-1.083452E-03_fp,-1.788509E-05_fp, 5.464239E-09_fp, -3.855673E-05_fp, 9.360072E-07_fp,-2.639362E-09_fp, 1.101309E-03_fp, 3.↩ 599147E-05_fp, -1.043146E-07_fp /)
- real(fp), dimension(36), parameter, public **mod_rttov_fastem5r1_coef::lcoef4** = (/ -9.197134E-02_fp, 8.310678E-04_fp,-6.065411E-07_fp, 1.350073E-01_fp,-1.032096E-03_fp, 4.259935E-07_fp,-4.373322E-02_fp, 2.545863E-04_fp, 9.835554E-08_fp,-1.199751E-03_fp, 1.360423E-05_fp,-2.088404E-08_fp,-2.↩ 201640E-05_fp, 1.951581E-07_fp,-2.599185E-10_fp, 4.477322E-04_fp,-2.986217E-05_fp, 9.406466E-08_fp,-7.103127E-02_fp,-4.713113E-05_fp, 1.754742E-06_fp, 9.720859E-02_fp, 1.374668E-04_fp,-2.↩ 591771E-06_fp,-2.687455E-02_fp, -3.677779E-05_fp, 7.548377E-07_fp,-3.049506E-03_fp,-5.412826E-05_fp, 2.285387E-07_fp, -2.201640E-05_fp, 1.951581E-07_fp,-2.599185E-10_fp, 2.297488E-03_fp, 3.↩ 787032E-05_fp, -1.553581E-07_fp /)
- real(fp), dimension(8), parameter, public **mod_rttov_fastem5r1_coef::scoef** = (/ -5.0208480E-06_↩ fp, 2.3297951E-08_fp, 4.6625726E-08_fp, -1.9765665E-09_fp, -7.0469823E-04_fp, 7.5061193E-04_fp, 9.↩ 8103876E-04_fp, 1.5489504E-04_fp /)
- real(fp), dimension(45), parameter, public **mod_rttov_fastem5r1_coef::t_c5** = (/ 0.199277E+00↩ _fp, 0.166155E+00_fp, 0.153272E-01_fp, 0.399234E+01_fp,-0.130968E+01_fp, -0.874716E+00_↩ fp,-0.169403E+01_fp,-0.260998E-01_fp, 0.540443E+00_fp,-0.282483E+00_fp, -0.219994E+00_fp,-0.↩ 203438E-01_fp, 0.351731E+00_fp, 0.208641E+01_fp,-0.693299E+00_fp, 0.867861E-01_fp, 0.619020E-01_fp, 0.595251E-02_fp,-0.475191E+01_fp,-0.430134E-01_fp, 0.248524E+01_fp, 0.388242E-01_↩ fp, 0.194901E+00_fp,-0.425093E-01_fp, 0.607698E+01_fp, -0.313861E+01_fp,-0.103383E+01_fp,-0.↩ 377867E+01_fp, 0.180284E+01_fp, 0.699556E+00_fp, -0.506455E-01_fp,-0.262822E+00_fp, 0.703056E-01_fp, 0.362055E+01_fp,-0.120318E+01_fp, -0.124971E+01_fp, 0.154014E-01_fp, 0.759848E-01_fp,-0.↩ 268604E-01_fp,-0.802073E+01_fp, 0.324658E+01_fp, 0.304165E+01_fp, 0.100000E+01_fp, 0.200000E-01_fp, 0.300000E+00_fp /)
- real(fp), dimension(45), parameter, public **mod_rttov_fastem5r1_coef::t_c4** = (/ -0.675700E-01↩ _fp, 0.214600E+00_fp,-0.363000E-02_fp, 0.636730E+01_fp, 0.900610E+00_fp, -0.524880E+00_↩ fp,-0.370920E+01_fp,-0.143310E+01_fp, 0.397450E+00_fp, 0.823100E-01_fp, -0.255980E+00_fp, 0.↩ 552000E-02_fp, 0.208000E+01_fp, 0.244920E+01_fp,-0.456420E+00_fp, -0.224900E-01_fp, 0.616900E-01_fp,-0.344000E-02_fp,-0.507570E+01_fp,-0.360670E+01_fp, 0.118750E+01_fp, 0.124950E+00_↩ fp, 0.121270E+00_fp, 0.714000E-02_fp, 0.736620E+01_fp, -0.114060E+00_fp,-0.272910E+00_fp,-0.↩ 504350E+01_fp,-0.336450E+00_fp, 0.161260E+00_fp, -0.154290E+00_fp,-0.141070E+00_fp,-0.809000E-02_fp, 0.395290E+01_fp, 0.958580E+00_fp, -0.159080E+00_fp, 0.368500E-01_fp, 0.307100E-01_fp, 0.↩ 810000E-03_fp,-0.619960E+01_fp, -0.172580E+01_fp, 0.641360E+00_fp, 0.100000E+01_fp, 0.200000E-01_fp, 0.300000E+00_fp /)
- real(fp), dimension(120), parameter, public **mod_rttov_fastem5r1_coef::b_coef** = (/ 3.307255E-04_fp,-2.901276E-06_fp,-1.475497E-04_fp, 1.288152E-06_fp, 1.004010E-04_fp, -2.671158E-07_fp, 4.363154E-06_fp,-9.817795E-09_fp,-4.777876E-05_fp, 3.051852E-08_fp, 1.369383E-03_fp,-2.215847E-05_fp,-8.↩ 099833E-04_fp, 1.767702E-05_fp,-5.977649E-06_fp, -1.784656E-07_fp,-9.355531E-07_fp, 5.495131E-08_fp,-3.479300E-05_fp,-3.751652E-07_fp, 2.673536E-04_fp,-1.378890E-06_fp,-8.660113E-05_fp, 2.↩ 871488E-07_fp, 1.361118E-05_fp, -1.622586E-08_fp,-1.232439E-07_fp,-3.067416E-09_fp,-1.835366E-06_fp, 8.098728E-09_fp, 1.255415E-04_fp,-5.145201E-07_fp,-8.832514E-06_fp,-5.105879E-09_fp, 2.↩ 734041E-05_fp, -3.398604E-07_fp, 3.417435E-06_fp,-7.043251E-09_fp, 1.497222E-05_fp,-6.832110E-

09_fp, -2.315959E-03_fp,-1.023585E-06_fp, 5.154471E-05_fp, 9.534546E-06_fp,-6.306568E-05_fp, -4.↩
378498E-07_fp,-2.132017E-06_fp, 1.612415E-08_fp,-1.929693E-06_fp,-6.217311E-09_fp, -1.656672E-
04_fp, 6.385099E-07_fp, 2.290074E-06_fp, 1.103787E-07_fp,-5.548757E-06_fp, 5.275966E-08_fp,-4.↩
653774E-07_fp, 1.427566E-09_fp,-3.197232E-06_fp,-4.048557E-09_fp, -1.909801E-04_fp,-3.387963E-
07_fp, 4.641319E-05_fp, 4.502372E-07_fp,-5.055813E-05_fp, 2.104201E-07_fp,-4.121861E-06_fp,-1.↩
633057E-08_fp,-2.469888E-05_fp, 4.492103E-08_fp, -4.582853E-03_fp,-5.373940E-06_fp, 9.713047E-
04_fp, 1.783009E-05_fp,-4.539091E-04_fp, 7.652954E-07_fp,-6.708905E-06_fp, 2.148401E-08_fp, 8.↩
054350E-05_fp, 3.069258E-07_fp, -6.405746E-05_fp,-9.694284E-08_fp, 1.914498E-05_fp, 1.336975E-
07_fp,-4.561696E-06_fp, 3.769169E-08_fp,-6.105244E-07_fp, 2.433761E-10_fp,-3.961735E-06_fp, 1.↩
995636E-08_fp, 1.350148E-06_fp, 3.678149E-07_fp, 1.261701E-05_fp,-2.011440E-07_fp,-2.361347E-
05_fp, 2.943147E-08_fp,-1.304551E-07_fp,-1.119368E-09_fp, 8.469458E-06_fp,-2.292171E-09_fp, 1.↩
419156E-03_fp,-3.838338E-06_fp, 8.222562E-05_fp,-1.106098E-06_fp,-5.482327E-05_fp, 3.083137E-
07_fp, 4.418828E-06_fp,-1.302562E-08_fp, 3.768883E-05_fp,-5.012753E-08_fp, -9.396649E-06_fp, 2.↩
764698E-07_fp, 1.745336E-05_fp,-1.427031E-07_fp,-3.879930E-06_fp, -1.117458E-08_fp, 5.688281E-08↩
_fp, 1.513582E-09_fp, 6.778764E-06_fp,-7.691286E-09_fp /)

- real(fp), dimension(9), parameter, public **mod_rttov_fastem5r1_coef::x** = (/ 0.0_fp, 1.4_fp, 6.8_fp, 10.7_fp, 19.35_fp, 37._fp, 89._fp, 150._fp, 200._fp/)
- real(fp), dimension(9), parameter, public **mod_rttov_fastem5r1_coef::y** = (/ 0.0_fp, 0.1_fp, 0.6_fp, 0.9_fp, 1._fp, 1.0_fp, 0.4_fp, 0.2_fp, 0.0_fp/)
- real(fp), dimension(6, 6, 2), parameter, public **mod_rttov_fastem5r1_coef::coef_mk_azi** = RESHAPE((/ 4.401E-02, -1.636E+01, 1.478E+00, -4.800E-02, 3.202E-06, -6.002E-05, 4.379E-02, -1.633E+01, 1.↩
453E+00, -4.176E-02, 5.561E-06, -4.644E-05, 5.009E-02, -1.638E+01, 1.520E+00, -3.994E-02, 1.330E-05, 1.113E-05, 5.165E-02, -1.638E+01, 1.543E+00, -4.066E-02, 1.494E-05, 1.010E-05, 5.553E-02, -1.638E+01, 1.602E+00, -4.246E-02, 1.903E-05, 7.524E-06, -9.131E-05, 1.251E+00, 6.769E-01, -2.913E-02, 1.092E+00, -1.806E-04, -1.234E-07, -8.179E-03, -1.040E+01, 4.477E-01, 0.000E+00, 3.390E-05, -1.938E-05, -8.007E-03, -1.039E+01, 4.610E-01, 0.000E+00, 4.419E-05, 1.362E-04, -1.013E-03, -9.235E+00, 3.844E-01, 0.↩
000E+00, 2.891E-04, 1.519E-04, -7.865E-04, -9.234E+00, 3.884E-01, 0.000E+00, 6.856E-04, 1.910E-04, -2.224E-04, -9.232E+00, 3.982E-01, 0.000E+00, 1.673E-03, 3.554E-04, 5.226E-04, 9.816E-01, -7.783E-03, 0.000E+00, 2.437E+01 /), (/6,6,2/))
- real(fp), dimension(5), parameter, public **mod_rttov_fastem5r1_coef::fr_coeff** = (/ 0.07_fp, -1.748e-3_fp, -7.336e-5_fp, 1.044e-7_fp, -0.93_fp /)

9.30.1 Detailed Description

Contains data for the FASTEM-4,5,6 MW sea surface emissivity models.

9.31 src/MW/Water/RTTOV_FASTEM/rttov_fastem5r1_tl.F90 File Reference

TL of FASTEM-4,5,6 emissivity and reflectance calculation.

Modules

- module rttov_fastem5r1_tl_module

  TL of RTTOV FASTEM-4,5,6 emissivity and reflectance calculation.

Functions/Subroutines

- subroutine rttov_fastem5r1_tl_module::rttov_fastem5r1_tl (fastem_version, Frequency, Zenith_Angle, Temperature, Salinity, Wind_Speed, Temperature_tl, Salinity_tl, Wind_Speed_tl, Emissivity, Reflectivity, Emissivity_tl, Reflectivity_tl, Transmittance, Rel_Azimuth, Transmittance_tl, Rel_Azimuth_tl, Supply_Foam↩
_Fraction, Foam_Fraction, Foam_Fraction_tl)

  TL of FASTEM-4,5,6 emissivity and reflectance calculation.

9.31.1 Detailed Description

TL of FASTEM-4,5,6 emissivity and reflectance calculation.

9.32 src/MW/Water/RTTOV_FASTEM/rttov_tessem_mod.F90 File Reference

Subroutines for TESSEM2 MW sea surface emissivity model.

Data Types

- type rttov_tessem_mod::tessem_net

Modules

- module rttov_tessem_mod

  Subroutines for TESSEM2 MW sea surface emissivity model.

Functions/Subroutines

- subroutine **rttov_tessem_mod::prop_neuralnet** (net, x, y)
- subroutine **rttov_tessem_mod::rttov_tessem** (freq, theta, windspeed, tskin, salinity, emis_h, emis_v)
- subroutine **rttov_tessem_mod::prop_neuralnet_tl** (net, x, x_tl, y_tl)
- subroutine **rttov_tessem_mod::rttov_tessem_tl** (freq, theta, windspeed, tskin, salinity, windspeed_↩ tl, tskin_tl, salinity_tl, emis_h_tl, emis_v_tl)
- subroutine **rttov_tessem_mod::prop_neuralnet_ad** (net, x, x_ad, y_ad)
- subroutine **rttov_tessem_mod::rttov_tessem_ad** (freq, theta, windspeed, tskin, salinity, windspeed_ad, tskin_ad, salinity_ad, emis_h_ad, emis_v_ad)

Variables

- integer(jpim), parameter **rttov_tessem_mod::tessem_nin** = 5
- integer(jpim), parameter **rttov_tessem_mod::tessem_nout** = 1
- integer(jpim), parameter **rttov_tessem_mod::tessem_ncache** = 15
- type(tessem_net) **rttov_tessem_mod::net_h**
- type(tessem_net) **rttov_tessem_mod::net_v**

9.32.1 Detailed Description

Subroutines for TESSEM2 MW sea surface emissivity model.

9.33 src/VisIR/Ice/CSEM_IceIR_SfcOptics.f90 File Reference

CSEM_IceIR_SfcOptics.f90.

Modules

- module csem_iceir_sfcoptics

  Container module with all the IR_ICE models available in the CSEM model repository.

Functions/Subroutines

- integer function, public csem_iceir_sfcoptics::csem_compute_iceir_sfcoptics (Surface, SfcOptics, Options)

  PURPOSE: Function to compute the ice surface emissivity and reflectivity at infrared wavelength.
- integer function, public csem_iceir_sfcoptics::csem_compute_iceir_sfcoptics_tl (SfcOptics_TL)

  PURPOSE: Function to compute the ice surface emissivity and reflectivity tangent-linear at infrared wavelength.
- integer function, public csem_iceir_sfcoptics::csem_compute_iceir_sfcoptics_ad (Surface_AD)

  PURPOSE: Function to compute the ice surface emissivity and reflectivity adjoint at infrared wavelength.

9.33.1 Detailed Description

CSEM_IceIR_SfcOptics.f90.

9.34 src/VisIR/Ice/CSEM_IceVIS_SfcOptics.f90 File Reference

CSEM_IceVIS_SfcOptics.f90.

Modules

- module csem_icevis_sfcoptics

  Container module with all the VIS_ICE models available in the CSEM model repository.

Functions/Subroutines

- integer function, public csem_icevis_sfcoptics::csem_compute_icevis_sfcoptics (Surface, SfcOptics, Options)

  PURPOSE: Function to compute the ice surface emissivity and reflectivity at visible wavelength.
- integer function, public csem_icevis_sfcoptics::csem_compute_icevis_sfcoptics_tl (SfcOptics_TL)

  PURPOSE: Function to compute the ice surface emissivity and reflectivity tangent-linear at visible wavelength.
- integer function, public csem_icevis_sfcoptics::csem_compute_icevis_sfcoptics_ad (Surface_AD)

  PURPOSE: Function to compute the ice surface emissivity and reflectivity adjoint at visible wavelength.

9.34.1 Detailed Description

CSEM_IceVIS_SfcOptics.f90.

9.35 src/VisIR/Land/CSEM_LandIR_SfcOptics.f90 File Reference

CSEM_LandIR_SfcOptics.f90.

Modules

- module csem_landir_sfcoptics

  Container module with all the IR_LAND models available in the CSEM model repository.

Functions/Subroutines

- integer function, public csem_landir_sfcoptics::csem_compute_landir_sfcoptics (Surface, SfcOptics, Options)

  PURPOSE: Function to compute the land surface emissivity and reflectivity at infrared wavelength.
- integer function, public csem_landir_sfcoptics::csem_compute_landir_sfcoptics_tl (SfcOptics_TL)

  PURPOSE: Function to compute the land surface emissivity and reflectivity tangent-linear at infrared wavelength.
- integer function, public csem_landir_sfcoptics::csem_compute_landir_sfcoptics_ad (Surface_AD)

  PURPOSE: Function to compute the Snowsurface emissivity and reflectivity adjoint at infrared wavelength.

9.35.1 Detailed Description

CSEM_LandIR_SfcOptics.f90.

9.36 src/VisIR/Land/CSEM_LandVIS_SfcOptics.f90 File Reference

CSEM_LandVIS_SfcOptics.f90.

Modules

- module csem_landvis_sfcoptics

  Container module with all the VIS_LAND models available in the CSEM model repository.

Functions/Subroutines

- integer function, public csem_landvis_sfcoptics::csem_compute_landvis_sfcoptics (Surface, SfcOptics, Options)

  PURPOSE: Function to compute the land surface emissivity and reflectivity at visible wavelength.
- integer function, public csem_landvis_sfcoptics::csem_compute_landvis_sfcoptics_tl (SfcOptics_TL)

  PURPOSE: Function to compute the land surface emissivity and reflectivity tangent-linear at visible wavelength.
- integer function, public csem_landvis_sfcoptics::csem_compute_landvis_sfcoptics_ad (Surface_AD)

  PURPOSE: Function to compute the Snowsurface emissivity and reflectivity adjoint at visible wavelength.

9.36.1 Detailed Description

CSEM_LandVIS_SfcOptics.f90.

9.37 src/VisIR/Snow/CSEM_SnowIR_SfcOptics.f90 File Reference

CSEM_SnowIR_SfcOptics.f90.

Modules

- module csem_snowir_sfcoptics

  Container module with all the IR_SNOW models available in the CSEM model repository.

Functions/Subroutines

- integer function, public csem_snowir_sfcoptics::csem_compute_snowir_sfcoptics (Surface, SfcOptics, Options)

  PURPOSE: Function to compute the snow surface emissivity and reflectivity at infrared wavelength.
- integer function, public csem_snowir_sfcoptics::csem_compute_snowir_sfcoptics_tl (SfcOptics_TL)

  PURPOSE: Function to compute the snow surface emissivity and reflectivity tangent-linear at infrared wavelength.
- integer function, public csem_snowir_sfcoptics::csem_compute_snowir_sfcoptics_ad (Surface_AD)

  PURPOSE: Function to compute the Snow surface emissivity and reflectivity adjoint at infrared wavelength.

9.37.1 Detailed Description

CSEM_SnowIR_SfcOptics.f90.

9.38 src/VisIR/Snow/CSEM_SnowVIS_SfcOptics.f90 File Reference

CSEM_SnowVIS_SfcOptics.f90.

Modules

- module csem_snowvis_sfcoptics

  Container module of all the VIS_SNOW models available in the CSEM model repository.

Functions/Subroutines

- integer function, public csem_snowvis_sfcoptics::csem_compute_snowvis_sfcoptics (Surface, SfcOptics, Options)

  PURPOSE: Function to compute the snow surface emissivity and reflectivity at visible wavelength.
- integer function, public csem_snowvis_sfcoptics::csem_compute_snowvis_sfcoptics_tl (SfcOptics_TL)

  PURPOSE: Function to compute the snow surface emissivity and reflectivity tangent-linear at visible wavelength.
- integer function, public csem_snowvis_sfcoptics::csem_compute_snowvis_sfcoptics_ad (Surface_AD)

  PURPOSE: Function to compute the Snowsurface emissivity and reflectivity adjoint at visible wavelength.

9.38.1 Detailed Description

CSEM_SnowVIS_SfcOptics.f90.

9.39 src/VisIR/Water/CSEM_WaterIR_SfcOptics.f90 File Reference

CSEM_WaterIR_SfcOptics.f90.

Modules

- module csem_waterir_sfcoptics

  Container module with all the IR_WATER models available in the CSEM model repository.

Functions/Subroutines

- integer function, public csem_waterir_sfcoptics::csem_compute_waterir_sfcoptics (Surface, SfcOptics, Options, iVar)

  PURPOSE: Function to compute the ocean surface emissivity and reflectivity at infrared wavelength.
- integer function, public csem_waterir_sfcoptics::csem_compute_waterir_sfcoptics_tl (Surface_TL, SfcOptics_TL, iVar)

  PURPOSE: Function to compute the ocean surface emissivity and reflectivity tangent-linear at infrared wavelength.
- integer function, public csem_waterir_sfcoptics::csem_compute_waterir_sfcoptics_ad (SfcOptics_AD, Surface_AD, iVar)

  PURPOSE: Function to compute the ocean surface emissivity and reflectivity adjoint at infrared wavelength.

9.39.1 Detailed Description

CSEM_WaterIR_SfcOptics.f90.

9.40 src/VisIR/Water/CSEM_WaterVIS_SfcOptics.f90 File Reference

CSEM_WaterVIS_SfcOptics.f90.

Modules

- module csem_watervis_sfcoptics

  Container module with all the VIS_WATER models available in the CSEM model repository.

Functions/Subroutines

- integer function, public csem_watervis_sfcoptics::csem_compute_watervis_sfcoptics (Surface, SfcOptics, Options, iVar)

  PURPOSE: Function to compute the ocean surface emissivity and reflectivity at visible wavelength.
- integer function, public csem_watervis_sfcoptics::csem_compute_watervis_sfcoptics_tl (SfcOptics_TL)

  PURPOSE: Function to compute the ocean surface emissivity and reflectivity tangent-linear at visible wavelength.
- integer function, public csem_watervis_sfcoptics::csem_compute_watervis_sfcoptics_ad (Surface_AD)

  PURPOSE: Function to compute the ocean surface emissivity and reflectivity adjoint at visible wavelength.

9.40.1 Detailed Description

CSEM_WaterVIS_SfcOptics.f90.

Index