SM-01 Module 2

Generated by Doxygen 1.7.4

Mon Nov 4 2013 05:41:55

Contents

Chapter 1

Module Index

1.1 Modules

Here is a list of all modules:

Chapter 2

Data Structure Index

2.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Chapter 4

Module Documentation

4.1 General Sky Model Module

Data Structures

- struct _smdat_
- struct _smspec_
- struct _smparam_
- struct _smgrid_

Files

- file sm_general.c
- file sm_general.h
- file sm_general.h

Defines

- #define SM_GENERAL_H
- #define SM_BOOL
- #define F SM_F
- #define T SM_T
- #define SM_NMAXERR 100
- #define SM_LENLINE 160
- #define SM_MAXLEN 4000
- #define SM_MAXPAR 21
- #define SM_TOL 1e-7
- #define SM_SR_IN_ARCSEC2 4.254517e+10
- #define SM_GAS_CONST 8.31447
- #define SM_MOL_AIR_DRY 0.0289644
- #define SM_GRAV_ACC 9.80665

- #define SM_ERROR_FOF SM_ERROR_FOPEN
- #define SM_ERROR_ISM SM_ERROR_NOMEM
- #define SM_ERROR_EIS SM_ERROR_SUBROUTINE
- #define SM_ERROR_ISD SM_ERROR_INSUFF_DATA
- #define SM_ERROR_FOPEN_TXT "File opening failed"
- #define SM_ERROR_UFS_TXT "Unexpected file structure"
- #define SM_ERROR_IFE_TXT "Invalid file name extension"
- #define SM_ERROR_BDR_TXT "Bad directory"
- #define SM_ERROR_NDA_TXT "No data"
- #define SM_ERROR_ISD_TXT "Insufficient data points"
- #define SM_ERROR_IDG_TXT "Inconsistent data grids"
- #define SM_ERROR_IDR_TXT "Invalid data range"
- #define SM_ERROR_IOD_TXT "Invalid order of data points"
- #define SM_ERROR_IIP_TXT "Invalid input parameter(s)"
- #define SM_ERROR_IOV_TXT "Invalid object value(s)"
- #define SM_ERROR_IOS_TXT "Invalid object structure"
- #define SM_ERROR_SUBROUTINE_TXT "Error in subroutine"
- #define SM_ERROR_ACCES_TXT "Permission denied"
- #define SM_ERROR_LOOP_TXT "Too many symbolic links"
- #define SM_ERROR_NAMETOOLONG_TXT "Pathname too long"
- #define SM_ERROR_NOENT_TXT "File/dir does not exist"
- #define SM_ERROR_NOTDIR_TXT
- #define SM_ERROR_ROFS_TXT
- #define SM_ERROR_FAULT_TXT
- #define SM_ERROR_INVAL_TXT "Mode was incorrectly specified"
- #define SM_ERROR_IO_TXT "I/O error occurred"
- #define SM_ERROR_NOMEM_TXT "Insufficient memory"
- #define SM_ERROR_TXTBSY_TXT
- #define SM_ERROR_EXIST_TXT "File/dir already exists"
- #define SM_ERROR_NOSPC_TXT "No space left on device"
- #define SM_ERROR_PERM_TXT
- #define SM_ERROR_LINK_TXT "Could not create symbolic link"
- #define SM_ERROR_UNDEF_TXT "Undefined error"
- #define SM_ERROR_FOF_TXT SM_ERROR_FOPEN_TXT
- #define SM_ERROR_ISM_TXT SM_ERROR_NOMEM_TXT
- #define SM_ERROR_EIS_TXT SM_ERROR_SUBROUTINE_TXT

Typedefs

- typedef enum SM_BOOL smbool
- typedef enum _sm_error_code_ sm_error_code
- typedef struct _smdat_ smdat
- typedef struct _smspec_ smspec
- typedef struct _smparam_ smparam
- typedef struct _smgrid_ smgrid

Enumerations

- enum SM_BOOL { SM_F, SM_T }
- enum _sm_error_code_ {

  SM_ERROR_FOPEN = CPL_ERROR_EOL + 11, SM_ERROR_UFS = CPL_-ERROR_EOL + 12, SM_ERROR_IFE = CPL_ERROR_EOL + 13, SM_ERROR_-NDA = CPL_ERROR_EOL + 20,

  SM_ERROR_INSUFF_DATA = CPL_ERROR_EOL + 21, SM_ERROR_IDG = CPL_-ERROR_EOL + 22, SM_ERROR_IDR = CPL_ERROR_EOL + 23, SM_ERROR_-IOD = CPL_ERROR_EOL + 24,

  SM_ERROR_IIP = CPL_ERROR_EOL + 30, SM_ERROR_IOV = CPL_ERROR_-EOL + 31, SM_ERROR_IOS = CPL_ERROR_EOL + 32, SM_ERROR_SUBROUTINE = CPL_ERROR_EOL + 40,

  SM_ERROR_ACCES = CPL_ERROR_EOL + 50, SM_ERROR_LOOP = CPL_-ERROR_EOL + 51, SM_ERROR_NAMETOOLONG = CPL_ERROR_EOL + 52, SM_ERROR_NOENT = CPL_ERROR_EOL + 53,

  SM_ERROR_NOTDIR = CPL_ERROR_EOL + 54, SM_ERROR_ROFS = CPL_-ERROR_EOL + 55, SM_ERROR_FAULT = CPL_ERROR_EOL + 56, SM_ERROR_-INVAL = CPL_ERROR_EOL + 57,

  SM_ERROR_IO = CPL_ERROR_EOL + 58, SM_ERROR_NOMEM = CPL_ERROR_-EOL + 59, SM_ERROR_TXTBSY = CPL_ERROR_EOL + 60, SM_ERROR_-EXIST = CPL_ERROR_EOL + 61,

  SM_ERROR_NOSPC = CPL_ERROR_EOL + 62, SM_ERROR_PERM = CPL_-ERROR_EOL + 63, SM_ERROR_BADUSERINPUT = CPL_ERROR_EOL + 70, SM_ERROR_LINK = CPL_ERROR_EOL + 71,

  SM_ERROR_RFM = CPL_ERROR_EOL + 81, SM_ERROR_UNDEF = CPL_-ERROR_EOL + 80 }

Functions

- cpl_error_code sm_spec_malloc (smspec *spec, const int size)
- cpl_error_code sm_spec_create (smspec *outspec, const double limlam[2], const double dlam)
- cpl_error_code sm_spec_read (smspec *spec, const char *filename)
- cpl_error_code sm_spec_readrange (smspec *spec, const char *filename, const double limlam[2], const int step)
- cpl_error_code sm_spec_readcpl (smspec *spec, const cpl_table *cpltab)
- cpl_error_code sm_spec_readfits (smspec *spec, const char *filename)
- cpl_error_code sm_spec_readfitsrange (smspec *spec, const char *filename, const double limlam[2], const int step)
- cpl_error_code sm_spec_copy (smspec *outspec, const smspec *inspec)
- cpl_error_code sm_spec_compgrids (const smspec *spec1, const smspec *spec2)
- cpl_error_code sm_spec_join (smspec *spec, const smspec *errfunc, const int errflag)
- cpl_error_code sm_spec_split (smspec *spec, smspec *errfunc, const int errflag)

- cpl_error_code sm_spec_changetype (smspec *spec, const int type)
- cpl_error_code sm_spec_scalerange (smspec *spec, const double limlam[2], const char op, const double c)
- cpl_error_code sm_spec_scale (smspec *spec, const char op, const double c)
- cpl_error_code sm_spec_modval (smspec *spec, const double lam, const char op, const double c)
- cpl_error_code sm_spec_calc (smspec *spec, const char op, const smspec *opspec)
- cpl_error_code sm_spec_funct (smspec *spec, const char *funct, const char baselab)
- cpl_error_code sm_spec_functnoerr (smspec *spec, const char *funct)
- cpl_error_code sm_spec_convunits (smspec *spec, const double factor, const int lamexp)
- cpl_error_code sm_spec_changegrid (smspec *spec, const double factor, const char *scale)
- cpl_error_code sm_spec_average (const smspec *spec, const double limlam[2], double mean[3])
- cpl_error_code sm_spec_write (const smspec *spec, const char *filename)
- cpl_error_code sm_spec_print (const smspec *spec)
- cpl_error_code sm_spec_writecpl (cpl_table *cpltab, const smspec *spec)
- cpl_error_code sm_spec_writefits (const smspec *spec, const char *filename)
- cpl_error_code sm_spec_free (smspec *spec)
- cpl_error_code sm_spec_extract (smspec *outspec, const smspec *inspec, const double limlam[2])
- cpl_error_code sm_spec_rebin (smspec *outspec, const smspec *inspec)
- cpl_error_code sm_spec_interpol (smspec *outspec, const smspec *inspec)
- cpl_error_code sm_spec_convolve (smspec *spec, const int nkpix, const double *kernel)
- cpl_error_code sm_param_read (FILE *stream, smparam par[])
- cpl_error_code sm_param_readcheck (FILE *stream, smparam par[], const char *parname, const int npar)
- cpl_error_code sm_param_check (smparam par[], const char *parname, const int npar)
- cpl_error_code sm_grid_malloc (smgrid *xy, const int nx, const int ny)
- cpl_error_code sm_grid_read (smgrid *xy, const char *filename)
- cpl_error_code sm_grid_extract (const smgrid *xy, const double x0, const double y0, double *outval)
- cpl_error_code sm_grid_write (const smgrid *xy, const char *filename)
- cpl_error_code sm_grid_print (const smgrid *xy)
- cpl_error_code sm_grid_free (smgrid *xy)
- cpl_error_code sm_basic_chdir (const char *dir)
- cpl_error_code sm_basic_access (const char *pathname, const int mode)
- cpl_error_code sm_basic_mkdir (const char *dir, const mode_t mode)
- cpl_error_code sm_basic_createdir (const char *dir, const mode_t mode)
- void sm_basic_initstring (char *str, const long n)
- cpl_boolean sm_basic_isnumber (char *str)
- char * sm_basic_replacestring (char *instring, char *oldsubstr, char *newsubstr)
- char * sm_basic_rmcntrl (char *str)

- void sm_basic_rmcntrl_inplace (char *str)
- char * sm_basic_strtrim (char *str)
- void sm_basic_strtrim_inplace (char *str)
- void sm_basic_terminatestring (char *str)
- cpl_error_code sm_basic_interpollin (const double *x_out, double *y_out, const long n_out, const double *x_ref, const double *y_ref, const long n_ref)

4.1.1 Detailed Description

This module provides basic definitions and functions for both sky model modules (*sm_module1* and *sm_module2*).

4.1.2 Define Documentation

4.1.2.1 #define F SM_F

Boolean false

4.1.2.2 #define SM_BOOL

Boolean type (F = false, T = true)

4.1.2.3 #define SM_ERROR_ACCES_TXT "Permission denied"

4.1.2.4 #define SM_ERROR_BDR_TXT "Bad directory"

4.1.2.5 #define SM_ERROR_EIS SM_ERROR_SUBROUTINE

4.1.2.6 #define SM_ERROR_EIS_TXT SM_ERROR_SUBROUTINE_TXT

4.1.2.7 #define SM_ERROR_EXIST_TXT "File/dir already exists"

4.1.2.8 #define SM_ERROR_FAULT_TXT

Value:

```
"Pathname points outside accessible " \
                                "address space"
```

4.1.2.9 #define SM_ERROR_FOF SM_ERROR_FOPEN

4.1.2.10 #define SM_ERROR_FOF_TXT SM_ERROR_FOPEN_TXT

4.1.2.11 #define SM_ERROR_FOPEN_TXT "File opening failed"

4.1.2.12 #define SM_ERROR_IDG_TXT "Inconsistent data grids"

4.1.2.13 #define SM_ERROR_IDR_TXT "Invalid data range"

4.1.2.14 #define SM_ERROR_IFE_TXT "Invalid file name extension"

4.1.2.15 #define SM_ERROR_IIP_TXT "Invalid input parameter(s)"

4.1.2.16 #define SM_ERROR_INVAL_TXT "Mode was incorrectly specified"

4.1.2.17 #define SM_ERROR_IO_TXT "I/O error occurred"

4.1.2.18 #define SM_ERROR_IOD_TXT "Invalid order of data points"

4.1.2.19 #define SM_ERROR_IOS_TXT "Invalid object structure"

4.1.2.20 #define SM_ERROR_IOV_TXT "Invalid object value(s)"

4.1.2.21 #define SM_ERROR_ISD SM_ERROR_INSUFF_DATA

4.1.2.22 #define SM_ERROR_ISD_TXT "Insufficient data points"

4.1.2.23 #define SM_ERROR_ISM SM_ERROR_NOMEM

4.1.2.24 #define SM_ERROR_ISM_TXT SM_ERROR_NOMEM_TXT

4.1.2.25 #define SM_ERROR_LINK_TXT "Could not create symbolic link"

4.1.2.26 #define SM_ERROR_LOOP_TXT "Too many symbolic links"

4.1.2.27 #define SM_ERROR_NAMETOOLONG_TXT "Pathname too long"

4.1.2.28 #define SM_ERROR_NDA_TXT "No data"

4.1.2.29 #define SM_ERROR_NOENT_TXT "File/dir does not exist"

4.1.2.30 #define SM_ERROR_NOMEM_TXT "Insufficient memory"

4.1.2.31 #define SM_ERROR_NOSPC_TXT "No space left on device"

4.1.2.32 #define SM_ERROR_NOTDIR_TXT

Value:

```
"Component used as directory in pathname " \
                                  "is not a directory"
```

4.1.2.33 #define SM_ERROR_PERM_TXT

Value:

```
"File system does not support creation of " \
                                "directories"
```

4.1.2.34 #define SM_ERROR_ROFS_TXT

Value:

```
"Write permission requested for file/dir " \
                                "on read-only file system"
```

4.1.2.35 #define SM_ERROR_SUBROUTINE_TXT "Error in subroutine"

4.1.2.36 #define SM_ERROR_TXTBSY_TXT

Value:

```
"Write access requested to executable " \
                                "which is being executed"
```

4.1.2.37 #define SM_ERROR_UFS_TXT "Unexpected file structure"

4.1.2.38 #define SM_ERROR_UNDEF_TXT "Undefined error"

4.1.2.39 #define SM_GAS_CONST 8.31447

Universal gas constant [J/(molK)]

4.1.2.40 #define SM_GENERAL_H

4.1.2.41 #define SM_GRAV_ACC 9.80665

Earth-surface gravitational acceleration [m/ s^2]

4.1.2.42 #define SM_LENLINE 160

Maximum number of characters per line

4.1.2.43 #define SM_MAXLEN 4000

Maximum number of string characters

4.1.2.44 #define SM_MAXPAR 21

Maximum number of space-, tab-, or '='-separated strings per line (see sm_param_read)

4.1.2.45 #define SM_MOL_AIR_DRY 0.0289644

Molar mass of dry air [kg/mol]

4.1.2.46 #define SM_NMAXERR 100

Maximum number of error codes

4.1.2.47 #define SM_SR_IN_ARCSEC2 4.254517e+10

steradians in arcsec^2

4.1.2.48 #define SM_TOL 1e-7

Required relative accuracy for data comparisons

4.1.2.49 #define T SM_T

Boolean true

4.1.3 Typedef Documentation

4.1.3.1 typedef enum _sm_error_code_ sm_error_code

Enumeration structure for sky model related error codes

4.1.3.2 typedef enum SM_BOOL smbool

4.1.3.3 typedef struct _smdat_ smdat

Structure for data points

Parameters

lam	wavelength
flux	flux
dflux1	(lower) flux error
dflux2	upper flux error

4.1.3.4 typedef struct _smgrid_ smgrid

Structure for coordinate grids

Parameters

nx	number of *x* coordinate values
ny	number of *y* coordinate values
∗*xpos*	vector of *x* coordinate values
∗*ypos*	vector of *y* coordinate values
∗∗*val*	matrix of data values for the coordinate grid

4.1.3.5 typedef struct _smparam_ smparam

Structure for a parameter value in different data types

Parameters

c	string (maximum length SM_LENLINE)
i	integer number
d	float number of double precision
n	number of value (for arrays; counted backwards)

4.1.3.6 typedef struct _smspec_ smspec

Structure for spectra

Parameters

type	number of columns (1 = no flux, 2 = no errors, 3 = symmetric error, 4 = lower and upper error)
n	number of wavelengths
∗*dat*	vector of data points defined by smdat structure

4.1.4 Enumeration Type Documentation

4.1.4.1 enum _sm_error_code_

Enumeration structure for sky model related error codes

Enumerator:

SM_ERROR_FOPEN

SM_ERROR_UFS

SM_ERROR_IFE

SM_ERROR_NDA

SM_ERROR_INSUFF_DATA
SM_ERROR_IDG
SM_ERROR_IDR
SM_ERROR_IOD
SM_ERROR_IIP
SM_ERROR_IOV
SM_ERROR_IOS
SM_ERROR_SUBROUTINE
SM_ERROR_ACCES
SM_ERROR_LOOP
SM_ERROR_NAMETOOLONG
SM_ERROR_NOENT
SM_ERROR_NOTDIR
SM_ERROR_ROFS
SM_ERROR_FAULT
SM_ERROR_INVAL
SM_ERROR_IO
SM_ERROR_NOMEM
SM_ERROR_TXTBSY
SM_ERROR_EXIST
SM_ERROR_NOSPC
SM_ERROR_PERM
SM_ERROR_BADUSERINPUT
SM_ERROR_LINK
SM_ERROR_RFM
SM_ERROR_UNDEF

4.1.4.2 enum SM_BOOL

Enumerator:

SM_F
SM_T

4.1.5 Function Documentation

4.1.5.1 cpl_error_code sm_basic_access (const char * *pathname,* const int *mode*)

Check for file or directory existence

This function provides a wrapper for the intrinsic function access(). It reports back all access() errors.

INPUT:

Parameters

pathname	name of a file or directory
mode	mode for accessibility checks F_OK = file existence X_OK = file exec permission W_OK = file write permission

OUTPUT:

Returns

CPL_ERROR_NONE on success, or on failure: SM_ERROR_ROFS, SM_ERROR_INVAL, SM_ERROR_TXTBSY, SM_ERROR_ACCES, SM_ERROR_LOOP, SM_ERROR_NAMETOOLONG, SM_ERROR_NOENT, SM_ERROR_NOTDIR, SM_ERROR_FAULT, SM_ERROR_IO, SM_ERROR_NOMEM, SM_ERROR_UNDEF

ERRORS:

- see access manual for details

4.1.5.2 cpl_error_code sm_basic_chdir (const char * *dir*)

Change working directory

This function provides a wrapper for the intrinsic function chdir(). It reports back all chdir() errors.

INPUT:

Parameters

dir	new working directory

OUTPUT:

Returns

CPL_ERROR_NONE on success, or on failure: SM_ERROR_ACCES, SM_ERROR_LOOP, SM_ERROR_NAMETOOLONG, SM_ERROR_NOENT, SM_ERROR_NOTDIR, SM_ERROR_FAULT, SM_ERROR_IO, SM_ERROR_NOMEM, SM_ERROR_UNDEF

ERRORS:

- see chdir manual for details

4.1.5.3 cpl_error_code sm_basic_createdir (const char * *dir*, const mode_t *mode*)

Create directory including error handling

This function creates a new directory with the specified permissions.

INPUT:

Parameters

dir	name of directory
mode	access permissions

OUTPUT:

Returns

CPL_ERROR_NONE on success, SM_ERROR_SUBROUTINE else.

ERRORS:

- see mkdir manual for details

NOTE:

- routine uses sm_basic_mkdir

4.1.5.4 void sm_basic_initstring (char * *str*, const long *n*)

Initialise a string variable.

This function initialises a given string *str* of length *n* with "\0".

INPUT:

Parameters

n	length of string

OUTPUT:

Parameters

str	string

Returns

mothing

4.1.5.5 cpl_error_code sm_basic_interpollin (const double * *x_out*, double * *y_out*, const long *n_out*, const double * *x_ref*, const double * *y_ref*, const long *n_ref*)

Linear interpolation.

This function calculates the interpolated y-values *y_out* at the positions *x_out* with respect to the reference x/y pairs (*x_ref* / *y_ref*).

Note

Points outside the range of the reference vectors will be extrapolated based on the

first / last values in the reference vectors for the low / high end, respectively.

INPUT:

Parameters

x_out	desired output x-spacing
n_out	length of *x_out* / *y_out*
x_ref	reference x-spacing
y_ref	reference y-values at *x_ref*
n_ref	length of *x_ref* / *y_ref*

OUTPUT:

Parameters

y_out	requested y-values at *x_out*

Returns

CPL_ERROR_NONE on success, CPL_ERROR_DIVISION_BY_ZERO else

4.1.5.6 cpl_boolean sm_basic_isnumber (char * *str*)

Check if string contains number.

This function checks whether *str* contains a valid number. A leading "+" or "-" is treated as a sign. A single "." is identified as a decimal point. Finally, an "e" is accepted if an exponent follows this letter.

Note

Surrounding spaces are not treated.

INPUT:

Parameters

str	string

OUTPUT:

Returns

CPL_TRUE if *str* is number, CPL_FALSE else.

4.1.5.7 cpl_error_code sm_basic_mkdir (const char * *dir,* const mode_t *mode*)

Create directory

This function provides a wrapper for the intrinsic function mkdir(). It reports back all mkdir() errors.

INPUT:

Parameters

dir	name of directory
mode	access permissions

OUTPUT:

Returns

CPL_ERROR_NONE on success, or on failure: SM_ERROR_EXIST, SM_ERROR_-NOSPC, SM_ERROR_PERM, SM_ERROR_ROFS, SM_ERROR_INVAL, SM_ERROR_-TXTBSY, SM_ERROR_ACCES, SM_ERROR_LOOP, SM_ERROR_NAMETOOLONG, SM_ERROR_NOENT, SM_ERROR_NOTDIR, SM_ERROR_FAULT, SM_ERROR_-IO, SM_ERROR_NOMEM, SM_ERROR_UNDEF

ERRORS:

- see mkdir manual for details

4.1.5.8 char* sm_basic_replacestring (char * *instring,* char * *oldsubstr,* char * *newsubstr*)

Substitutes substring in string

Parameters

instring	input string
oldsubstr	substring to be replaced
newsubstr	new substring

Returns

input string with substituted substring

4.1.5.9 char* sm_basic_rmcntrl (char * *str*)

Remove control characters from string.

This function removes all `ASCII` control characters from *str* using `iscntrl()`.

INPUT:

Parameters

str	string.

OUTPUT:

Returns

string.

4.1.5.10 void sm_basic_rmcntrl_inplace (char * *str*)

Remove control characters from string.

This function removes all `ASCII` control characters from *str* using `iscntrl()`. In contrast to `sm_basic_rmcntrl()`, the action is performed in place, i.e. the input gets overwritten.

INPUT & OUTPUT:

Parameters

str	string

OUTPUT:

Returns

nothing

4.1.5.11 char* sm_basic_strtrim (char * *str*)

Remove leading and trailing blanks from string.

This function removes all leading and trailing " " characters from *str* using `isspace()`.

INPUT:

Parameters

str	string

OUTPUT:

Returns

string

4.1.5.12 void sm_basic_strtrim_inplace (char * *str*)

Remove leading and trailing blanks from string.

This function removes all leading and trailing " " characters from *str* using `isspace()`. In contrast to `sm_basic_strtrim()`, the action is performed in place, i.e. the input gets overwritten.

INPUT & OUTPUT:

Parameters

str	string

OUTPUT:

Returns

nothing

4.1.5.13 void sm_basic_terminatestring (char * *str*)

Put "\0" at end of string.

This function places a "\0" at end of the input string.

Note

No checks are performed for writing beyond the boundary of allocated memory for the input string.

INPUT & OUTPUT:

Parameters

str	string

OUTPUT:

Returns

nothing

4.1.5.14 cpl_error_code sm_grid_extract (const smgrid * *xy*, const double *x0*, const double *y0*, double * *outval*)

Extracts value of an smgrid structure at position (x_0, y_0) by means of bilinear interpolation

INPUT:

Parameters

xy	input smgrid structure
x0	*x* coordinate position for extraction
y0	*y* coordinate position for extraction

OUTPUT:

Parameters

outval	value of smgrid structure at position (x_0, y_0)

ERRORS:

- NDA: No data
- ISD: Insufficient data points
- IOD: Invalid order of data points
- IIP: Invalid input parameter(s)

4.1.5.15 cpl_error_code sm_grid_free (smgrid * xy)

Frees memory occupied by an smgrid structure

INPUT:

Parameters

xy	smgrid structure of size $n_x \times n_y$

OUTPUT:

Parameters

xy	smgrid structure of size zero

ERRORS:

- none

4.1.5.16 cpl_error_code sm_grid_malloc (smgrid * xy, const int nx, const int ny)

Allocates memory for an smgrid structure of size $n_x \times n_y$. All values are zero.

INPUT:

Parameters

xy	smgrid structure of size zero
nx	number of *x* coordinates
ny	number of *y* coordinates

OUTPUT:

Parameters

xy	smgrid structure of size $n_x \times n_y$ if there is sufficient memory (otherwise size remains zero)

ERRORS:

- IIP: Invalid input parameter(s)
- ISM: Insufficient memory

4.1.5.17 cpl_error_code sm_grid_print (const smgrid * *xy*)

Prints smgrid structure on stdout

INPUT:

Parameters

xy	smgrid structure

ERRORS:

- none

4.1.5.18 cpl_error_code sm_grid_read (smgrid * *xy*, const char * *filename*)

Reads smgrid structure from ASCII file.

Required file structure:

- Comment lines have to be marked by #.
- Data lines:
  - 1: *nx ny*
  - 2: *xpos* (*nx* values)
  - 3: *ypos* (*ny* values)
  - 4-(*nx+3*): *val* (*ny* values each line)

INPUT:

Parameters

filename	name of input file

OUTPUT:

Parameters

xy	output smgrid structure

ERRORS:

- FOF: File opening failed
- UFS: Unexpected file structure

- ISM: Insufficient memory

4.1.5.19 cpl_error_code sm_grid_write (const smgrid * *xy*, const char * *filename*)

Writes smgrid structure to ASCII file or on stdout (indicated by "stdout" as filename)

INPUT:

Parameters

xy	smgrid structure
filename	name of output file or "stdout"

ERRORS:

- FOF: File opening failed

4.1.5.20 cpl_error_code sm_param_check (smparam *par[]*, const char * *parname*, const int *npar*)

Checks a parameter definition held by an array of smparam structures. The first string of the read line is expected to be the parameter name. It is checked whether it matches the input name. In the case of a deviation, an error is returned. The routine also checks the number of parameter values (not counting the parameter name), which is also an input parameter of the routine. An error is returned if there is a mismatch. In the case of errors, all entries of the smparam array are set to the default values "", 0, and 0.0.

INPUT:

Parameters

par	array of SM_MAXPAR smparam structures
parname	name of parameter
npar	number of expected parameter values

OUTPUT:

Parameters

par	smparam array (error: set to the data type default values, otherwise untouched)

ERRORS:

- EIS: Error in subroutine (see sm_param_read)
- IOV: Invalid object value(s)

4.1.5.21 cpl_error_code sm_param_read (FILE * *stream*, smparam *par[]*)

Reads a line from a file and returns the space-, tab-, or '='-separated individual strings in an array of SM_MAXPAR smparam structures. A smparam structure contains each string converted into integer and double precision floating point numbers. Comment lines marked by # and empty lines are not considered. In this case, the next line with data is read.

INPUT:

Parameters

stream	name of stream (to be defined by fopen command)
par	array of SM_MAXPAR smparam structures (to transfer parameter values)

OUTPUT:

Parameters

par	smparam array that contains the read value(s) as character string ("c"), integer ("i"), or double precision floating point number ("d"). The different data types can be accessed by adding the corresponding suffixes "c", "i", or "d" to the name of the smparam structure. An additional suffix "n" element indicates the number of read values minus the array index, i.e. *par*[0].n provides the number of values in the read line.

ERRORS:

- IIP: Invalid input parameter(s)
- UFS: Unexpected file structure
- ISM: Insufficient memory

In the case of problems, the default values "", 0, and 0.0 are used for the corresponding data types.

4.1.5.22 cpl_error_code sm_param_readcheck (FILE * *stream*, smparam *par[]*, const char * *parname*, const int *npar*)

Reads and checks a line from a file and returns the space-, tab-, or '='-separated individual strings in an array of SM_MAXPAR smparam structures. A smparam structure contains each string converted into integer and double precision floating point numbers. Comment lines marked by # and empty lines are not considered. In this case, the next line with data is read.

The first string of the read line is expected to be the parameter name. It is checked whether it matches the input name. In the case of a deviation, an error is returned. The routine also checks the number of parameter values (not counting the parameter name), which is also an input parameter of the routine. An error is returned if there is a mismatch.

INPUT:

Parameters

stream	name of stream (to be defined by fopen command)
par	array of SM_MAXPAR smparam structures (to transfer parameter values)
parname	name of parameter
npar	number of expected parameter values

OUTPUT:

Parameters

par	smparam array that contains the read value(s) as character string ("c"), integer ("i"), or double precision floating point number ("d"). The different data types can be accessed by adding the corresponding suffixes "c", "i", or "d" to the name of the smparam structure. An additional suffix "n" element indicates the number of read values minus the array index, i.e. *par*[0].n provides the number of values in the read line.

ERRORS:

- EIS: Error in subroutine (see sm_param_read)
- IOV: Invalid object value(s)

In the case of problems, the default values "", 0, and 0.0 are used for the corresponding data types.

4.1.5.23 cpl_error_code sm_spec_average (const smspec * *spec*, const double *limlam[2]*, double *mean[3]*)

Calculates mean of a spectrum (and its error function if given) in the given wavelength range

INPUT:

Parameters

spec	input spectrum
limlam	lower and upper wavelength limit for operation (Take HUGE_VAL for unconstrained limits.)

OUTPUT:

Parameters

mean	mean values for spectrum (and lower and upper error function if given)

ERRORS:

- NDA: No data
- IIP: Invalid input parameter(s)

4.1.5.24 cpl_error_code sm_spec_calc (smspec * *spec*, const char *op*, const smspec * *opspec*)

Performs addition, subtraction, multiplication, division, or equalisation of two spectra with error functions. Moreover, a spectrum can be used as a power of another spectrum. Error propagation (adding of squared errors) is considered. For asymmetric errors positive fluxes are required for multiplicative operations.

The two subtraction operators '-' and '_' differ by the order of the spectra, i.e., '-' and '_' imply $x_1 - x_2$ and $x_2 - x_1$, respectively. The two division operators '/' and '|' differ by the order of the spectra, i.e., '/' and '|' imply x_1/x_2 and x_2/x_1, respectively.

INPUT:

Parameters

spec	input smspec structure
op	operation: '+', '-', '_', '*', '/', '|', '^', or '='
opspec	smspec structure to modify spec (identical wavelengths needed!)

OUTPUT:

Parameters

spec	modified spectrum (no modification in the case of errors)

ERRORS:

- IIP: Invalid input parameter(s)
- CPL: Division by zero [warning]
- CPL: Access out of range [warning]

4.1.5.25 cpl_error_code sm_spec_changegrid (smspec * *spec*, const double *factor*, const char * *scale*)

Modifies the wavelengths of an smspec structure by multiplying a factor and/or performing $\log_{10}(x)$ or 10^x.

INPUT:

Parameters

spec	input smspec structure
factor	wavelengths are multiplied by this value (c).
scale	wavelength scale, options: • "log": logarithmic ($\log_{10}(cx)$) • "exp": exponential ($c\,10^x$) $\rightarrow$ reverses "log" option • "lin": linear (cx)

OUTPUT:

Parameters

spec	spectrum with modified wavelength scale (no modification in the case of errors)

ERRORS:

- IDR: Invalid data range
- IIP: Invalid input parameter(s)

4.1.5.26 cpl_error_code sm_spec_changetype (smspec * *spec*, const int *type*)

Changes spectrum type (1 = no flux, 2 = no errors, 3 = symmetric error, 4 = lower and upper error). For changes to 1 or 2 the obsolete columns are set to zero. For a change from 4 to 3 the lower and upper errors are made symmetric by simple averaging. For a change from 1-3 to 4 or 3 it is ensured that the lower and upper errors are equal (using the lower errors as reference). Only for a change from 4 to 4 nothing is done.

INPUT:

Parameters

spec	spectrum of arbitrary type
type	spectrum type (see smspec)

OUTPUT:

Parameters

spec	spectrum of given *type*

ERRORS:

- IIP: Invalid input parameter(s)

4.1.5.27 cpl_error_code sm_spec_compgrids (const smspec * *spec1*, const smspec * *spec2*)

Tests agreement of wavelength grids of two smspec structures. Returns $\neq 0$ for disagreement.

INPUT:

Parameters

spec1	first spectrum
spec2	second spectrum

ERRORS:

- IDG: Inconsistent data grids

4.1.5.28 cpl_error_code sm_spec_convolve (smspec * *spec*, const int *nkpix*, const double * *kernel*)

Convolution of smspec structure with given kernel. Errors are convolved in the same way as fluxes → not suitable for statistical noise.

Note

The centre of the convolution function is shifted by -0.5 pixels for an even number of kernel pixels.

INPUT:

Parameters

spec	input spectrum (smspec structure)
nkpix	number of kernel pixels
kernel	vector of kernel values (required: sum of all values = 1)

OUTPUT:

Parameters

spec	convolved input spectrum

ERRORS:

- IIP: Invalid input parameter(s)

4.1.5.29 cpl_error_code sm_spec_convunits (smspec * *spec*, const double *factor*, const int *lamexp*)

Modifies the flux of an smspec structure by multiplying a *factor* times λ^{lamexp}.

INPUT:

Parameters

spec	input smspec structure
factor	constant factor
lamexp	power for wavelengths

OUTPUT:

Parameters

spec	modified spectrum

ERRORS:

- IDR: Invalid data range

4.1.5.30 cpl_error_code sm_spec_copy (smspec * *outspec*, const smspec * *inspec*)

Copies smspec structure

INPUT:

Parameters

inspec	input spectrum

OUTPUT:

Parameters

outspec	output spectrum

ERRORS:

- none

4.1.5.31 cpl_error_code sm_spec_create (smspec * *outspec*, const double *limlam[2]*, const double *dlam*)

Initialisation of a wavelength grid that is characterised by constant bin size. The spectrum type 3 is set (see smspec).

INPUT:

Parameters

limlam	lower and upper limit of wavelength range
dlam	bin size

OUTPUT:

Parameters

outspec	smspec structure that provides wavelengths, fluxes (= 0.), and flux errors (= 0.) of a spectrum

ERRORS:

- IIP: Invalid input parameter(s)

4.1.5.32 cpl_error_code sm_spec_extract (smspec * *outspec*, const smspec * *inspec*, const double *limlam[2]*)

Extracts subspectrum in a given wavelength range of the input spectrum (:: smspec structure)

INPUT:

Parameters

inspec	input spectrum
limlam	wavelength limits for extraction

OUTPUT:

Parameters

outspec	extracted spectrum

ERRORS:

- IIP: Invalid input parameter(s)

4.1.5.33 cpl_error_code sm_spec_free (smspec * *spec*)

Frees memory occupied by an smspec structure

INPUT:

Parameters

spec	smspec structure with *n* data points

OUTPUT:

Parameters

spec	smspec structure with zero data points

ERRORS:

- none

4.1.5.34 cpl_error_code sm_spec_funct (smspec * *spec*, const char * *funct*, const char *baselab*)

Applies exponential or logarithmic function. Names of functions equal those of <math.h> excepting "exp10", "expmag", and "logmag" which correspond to 10^x, $10^{-0.4x}$, and $-2.5\log_{10}(x)$, respectively. Correction of resulting error asymmetry by means of the recipe $\frac{1}{2}(f(x+\mathrm{d}x)-f(x-\mathrm{d}x))$ for spectrum type 3 (see smspec).

INPUT:

Parameters

spec	input smspec structure
funct	"exp" or "log"
baselab	'e', 'd' (10), or 'm' ($10^{-0.4}$)

OUTPUT:

Parameters

spec	modified spectrum if function is valid

ERRORS:

- IIP: Invalid input parameter(s)
- CPL: Access out of range [warning]

4.1.5.35 cpl_error_code sm_spec_functnoerr (smspec * *spec*, const char * *funct*)

Applies mathematical function to flux. A possible error function is not changed. Names of functions equal those of <math.h> excepting "exp10", "expmag", and "logmag" which correspond to 10^x, $10^{-0.4x}$, and $-2.5\log_{10}(x)$, respectively.

INPUT:

Parameters

spec	input smspec structure
funct	"acos", "asin", "atan", "cos", "sin", "tan", "cosh", "sinh", "tanh", "exp", "exp10", "expmag", "log", "log10", "logmag", "sqrt", "ceil", "fabs", "floor"

OUTPUT:

Parameters

spec	modified spectrum if function is valid

ERRORS:

- IIP: Invalid input parameter(s)
- CPL: Access out of range [warning]

4.1.5.36 cpl_error_code sm_spec_interpol (smspec * *outspec*, const smspec * *inspec*)

Linear interpolation of smspec structure for conversion to finer wavelength grid. Optimal for sparse, unbinned data.

INPUT:

Parameters

outspec	desired wavelength grid
inspec	wavelengths and fluxes (per wavelength unit) of input spectrum

OUTPUT:

Parameters

outspec	interpolated input spectrum

ERRORS:

- none

4.1.5.37 cpl_error_code sm_spec_join (smspec * *spec*, const smspec * *errfunc*, const int *errflag*)

Uses the flux of a second smspec structure to fill the error vector of the first spectrum. If two error columns are used, lower or upper error has to be specified and the function has to be called for either case (starting with the lower error).

Note

Pre-existing error data in spec will be overwritten.

INPUT:

Parameters

spec	spectrum without errors (2 columns)
errfunc	error function (2 columns)
errflag	error flag (0 = symmetric error, 1 = lower error, 2 = upper error)

OUTPUT:

Parameters

spec	spectrum with errors (3 columns)

ERRORS:

- IIP: Invalid input parameter(s)

4.1.5.38 cpl_error_code sm_spec_malloc (smspec * *spec*, const int *size*)

Allocates memory for an smspec structure of given size. Assumes symmetric errors (see smspec).

INPUT:

Parameters

spec	smspec structure (no memory allocated)
size	number of data points

OUTPUT:

Parameters

spec	smspec structure with allocated memory (all values = 0)

ERRORS:

- ISM: Insufficient memory

4.1.5.39 cpl_error_code sm_spec_modval (smspec * *spec*, const double *lam*, const char *op*, const double *c*)

Modifies the flux of a given wavelength belonging to an smspec structure by adding, subtracting, multiplying, or dividing a constant. Moreover, the flux can be raised to a constant power or the flux of the selected wavelength can be substituted.

The two subtraction operators '-' and '_' differ by the order of spectrum and constant, i.e., '-' and '_' imply $x - c$ and $c - x$, respectively. The two division operators '/' and '|' differ by the order of spectrum and constant, i.e., '/' and '|' imply x/c and c/x, respectively.

INPUT:

Parameters

spec	input smspec structure
lam	wavelength (accuracy ruled by SM_TOL)
op	operation: '+', '-', '*', '/', '^', '=', '_', or '|'
c	constant (double precision float)

OUTPUT:

Parameters

spec	modified spectrum (no modification in the case of errors)

ERRORS:

- see sm_spec_scalerange

4.1.5.40 cpl_error_code sm_spec_print (const smspec * *spec*)

Prints smspec structure on stdout

INPUT:

Parameters

spec	input smspec structure

ERRORS:

- none

4.1.5.41 cpl_error_code sm_spec_read (smspec * *spec*, const char * *filename*)

Fills smspec structure by data read from a file. Each line of the file must consist of wavelength and flux. The presence of one or two optional error columns is detected automatically. Header lines are allowed if they are marked by #.

INPUT:

Parameters

filename	name of data file

OUTPUT:

Parameters

spec	output spectrum with read values

ERRORS:

- FOF: File opening failed
- UFS: Unexpected file structure
- CPL: Access out of range [warning]

4.1.5.42 cpl_error_code sm_spec_readcpl (smspec * *spec*, const cpl_table * *cpltab*)

Reads smspec structure from CPL table with one to four columns. The column IDs "lam", "flux", "dflux" (for symmetric errors), "dflux1" (lower error), and "dflux2" (upper error) are mandatory.

INPUT:

Parameters

cpltab	CPL table with one to four columns named "lam", "flux", "dflux"/"dflux1", or "dflux2"

OUTPUT:

Parameters

spec	spectrum of structure smspec

ERRORS:

- NDA: No data
- IOS: Invalid object structure
- CPL: Access out of range [warning]

4.1.5.43 cpl_error_code sm_spec_readfits (smspec * *spec*, const char * *filename*)

Reads smspec structure from FITS file by means of CPL

INPUT:

Parameters

filename	name of FITS file (extensions "fits" or "mt")

OUTPUT:

Parameters

spec	spectrum of structure smspec

ERRORS:

- IFE: Invalid file name extension
- FOF: File opening failed
- UFS: Unexpected file structure

4.1.5.44 cpl_error_code sm_spec_readfitsrange (smspec * *spec*, const char * *filename*, const double *limlam[2]*, const int *step*)

Reading of a spectrum from FITS file in a given wavelength range by means of CPL. The wavelength range of the extracted spectrum tends to be wider than the interval between the provided limits and depends on the given step size.

INPUT:

Parameters

filename	name of FITS file (extensions "fits or "mt")
limlam	wavelength limits for extraction
step	step in FITS table rows for search of given wavelength range (optimum: square root of total number of rows)

OUTPUT:

Parameters

spec	spectrum of structure smspec

ERRORS:

- IIP: Invalid input parameter(s)
- IFE: Invalid file name extension
- FOF: File opening failed
- UFS: Unexpected file structure

4.1.5.45 cpl_error_code sm_spec_readrange (smspec * *spec*, const char * *filename*, const double *limlam[2]*, const int *step*)

Reading of a spectrum from file in a given wavelength range. The wavelength range of the extracted spectrum tends to be wider than the interval between the provided limits and depends on the given step size. The presence of one or two optional error columns is detected automatically. Header lines are allowed if they are marked by #.

INPUT:

Parameters

filename	name of file with wavelength and flux (and error) columns
limlam	wavelength limits for extraction
step	step in file lines for search of given wavelength range (optimum: square root of total number of lines)

OUTPUT:

Parameters

spec	spectrum of structure smspec

ERRORS:

- IIP: Invalid input parameter(s)
- FOF: File opening failed
- UFS: Unexpected file structure
- CPL: Access out of range [warning]

4.1.5.46 cpl_error_code sm_spec_rebin (smspec * *outspec*, const smspec * *inspec*)

Rebins smspec structure by using bins with constant fluxes. Conserves integral of fluxes. Errors are rebinned in the same way as fluxes → not suitable for statistical noise.

INPUT:

Parameters

outspec	desired wavelength grid
inspec	wavelengths and fluxes (per wavelength unit) of input spectrum

OUTPUT:

Parameters

outspec	rebinned input spectrum

ERRORS:

- none

4.1.5.47 cpl_error_code sm_spec_scale (smspec * *spec*, const char *op*, const double *c*)

Modifies the flux of an smspec structure by adding, subtracting, multiplying, or dividing a constant. Moreover, the flux can be raised to a constant power or the full spectrum can be given a constant flux value.

The two subtraction operators '-' and '_' differ by the order of spectrum and constant, i.e., '-' and '_' imply $x - c$ and $c - x$, respectively. The two division operators '/' and '|' differ by the order of spectrum and constant, i.e., '/' and '|' imply x/c and c/x, respectively.

INPUT:

Parameters

spec	input smspec structure
op	operation: '+', '-', '*', '/', '^', '=', '_', or '|'
c	constant (double precision float)

OUTPUT:

Parameters

spec	modified spectrum (no modification in the case of errors)

ERRORS:

- see sm_spec_scalerange

4.1.5.48 cpl_error_code sm_spec_scalerange (smspec * *spec,* const double *limlam[2],* const char *op,* const double *c*)

Modifies the flux of an smspec structure in a given wavelength range by adding, subtracting, multiplying, or dividing a constant. Moreover, the flux can be raised to a constant power or the selected wavelength range can be given a constant flux value.

The two subtraction operators '-' and '_' differ by the order of spectrum and constant, i.e., '-' and '_' imply $x - c$ and $c - x$, respectively. The two division operators '/' and '|' differ by the order of spectrum and constant, i.e., '/' and '|' imply x/c and c/x, respectively.

INPUT:

Parameters

spec	input smspec structure
limlam	lower and upper wavelength limit for operation (Take HUGE_VAL for unconstrained limits.)
op	operation: '+', '-', '_', '*', '/', '|', '^', or '='
c	constant (double precision float)

OUTPUT:

Parameters

spec	modified spectrum (no modification in the case of errors)

ERRORS:

- IIP: Invalid input parameter(s)
- CPL: Division by zero [warning]
- CPL: Access out of range [warning]

4.1.5.49 cpl_error_code sm_spec_split (smspec * *spec,* smspec * *errfunc,* const int *errflag*)

Writes error function of an smspec structure to another spectrum. The error vector of the former spectrum is filled with zero. If two error columns are used, lower or upper error has to be specified and the function has to be called for either case.

INPUT:

Parameters

spec	spectrum with errors (3 columns)

OUTPUT:

Parameters

spec	spectrum without errors (2 columns)

errfunc	error function (2 columns)
errflag	error flag (0 = symmetric error, 1 = lower error, 2 = upper error)

ERRORS:

- IIP: Invalid input parameter(s)

4.1.5.50 cpl_error_code sm_spec_write (const smspec * *spec*, const char * *filename*)

Writes smspec structure to ASCII file or on stdout (indicated by "stdout" as filename)

INPUT:

Parameters

spec	input smspec structure
filename	name of output file or "stdout"

ERRORS:

- FOF: File opening failed
- NDA: No data
- IOV: Invalid object value(s)

4.1.5.51 cpl_error_code sm_spec_writecpl (cpl_table * *cpltab*, const smspec * *spec*)

Writes smspec structure to CPL table

INPUT:

Parameters

spec	input smspec structure

OUTPUT:

Parameters

cpltab	output CPL table

ERRORS:

- NDA: No data
- IOV: Invalid object value(s)

4.1.5.52 cpl_error_code sm_spec_writefits (const smspec * *spec*, const char * *filename*)

Writes smspec structure to FITS file by means of CPL

INPUT:

Parameters

spec	input smspec structure
filename	name of output file

ERRORS:

- none

4.2 Sky Model Module 2

Data Structures

- struct _smparmodel_

Files

- file sm_scatmoonlight.c
- file sm_scatmoonlight.h
- file sm_skyemcomp.c
- file sm_skyemcomp.h
- file sm_skyemcomp.h

Defines

- #define SM_SKYEMCOMP_H
- #define SM_FILENAMELIST "sm_filenames.dat"
- #define SM_RRSTEP 1000
- #define SM_LAM_UNIT 1e-6
- #define SM_RADMINLAM 1.3 * 1e-6 / SM_LAM_UNIT
- #define SM_ERAD 6371.
- #define SM_SIGMAX 4.
- #define SM_NSIGBIN 20

Typedefs

- typedef struct _smparmodel_ smparmodel

Functions

- cpl_error_code sm_etc_calcmodel (cpl_table *skytable, const cpl_parameterlist *params)
- cpl_error_code sm_etc_getparams (smparmodel *modelpar, const cpl_parameterlist *params)
- const cpl_parameter * sm_etc_parameterlist_find_const (const cpl_parameterlist *self, const char *name, cpl_type type)
- cpl_error_code sm_etc_splitstring (cpl_array *outval, char *instring)
- cpl_error_code sm_etc_readfilenames (smparmodel *modelpar, const cpl_parameterlist *params)
- cpl_error_code sm_etc_writetable (cpl_table *skytable, const smspec *radiance, const smspec *transmission)
- cpl_error_code sm_comp_skyemcomp (smspec *radiance, smspec *transmission, const smparmodel modelpar)
- cpl_error_code sm_comp_extrapoltrans (smspec *spec, const smparmodel modelpar, const double lim[2])
- double sm_comp_alttoairmass (const double alt)
- cpl_error_code sm_comp_getmolspec (smspec *spec, const cpl_parameterlist *libfilepar, const double lim[2])
- cpl_error_code sm_comp_readlibstruct (cpl_parameterlist *libfilepar, const smparmodel modelpar, const char *spectype)
- double sm_comp_vactoair_single (const double lam)
- cpl_error_code sm_comp_vactoair_spec (smspec *spec)
- double sm_comp_airtovac_single (const double lam)
- cpl_error_code sm_comp_airtovac_spec (smspec *spec)
- double sm_comp_calcrayleighscat1 (const double lam)
- cpl_error_code sm_comp_calcrayleighscat (smspec *rscattrans)
- double sm_comp_calcmiescat1 (const double lam)
- cpl_error_code sm_comp_calcmiescat (smspec *mscattrans)
- cpl_error_code sm_comp_scaletranscurv (smspec *trans, const smparmodel modelpar)
- cpl_error_code sm_comp_lunskybright (smspec *spec, const smparmodel modelpar, const smspec *solspec, const smspec *molabstrans, const smspec *rscattrans, const smspec *mscattrans)
- cpl_error_code sm_comp_lunskybright_v (smspec *spec, const smparmodel modelpar, const smspec *solspec, const smspec *molabstrans, const smspec *rscattrans, const smspec *mscattrans)
- cpl_error_code sm_comp_getmoonalbedo (smspec *albedo, const smparmodel modelpar)
- cpl_error_code sm_comp_scatstarlight (smspec *spec, const smparmodel modelpar, const smspec *molabstrans)
- cpl_error_code sm_comp_zodskybright (smspec *spec, const smparmodel modelpar, const smspec *solspec, const smspec *molabstrans, const smspec *rscattrans, const smspec *mscattrans)
- cpl_error_code sm_comp_telem (smspec *spec, const smparmodel modelpar)
- cpl_error_code sm_comp_extrapolrad (smspec *spec, const smspec *trans, const smparmodel modelpar, const double lim[2])

- cpl_error_code sm_comp_getlinespec (smspec *spec, const smparmodel modelpar, const smspec *rscattrans, const smspec *mscattrans)
- cpl_error_code sm_comp_readvarpar (cpl_table *varpar, const smparmodel modelpar)
- cpl_error_code sm_comp_scalelinetab (cpl_table *linedat, const smspec *linetab, const cpl_table *varpar)
- cpl_error_code sm_comp_convertlinetab (smspec *spec, const cpl_table *linedat)
- cpl_error_code sm_comp_convertlinetabo (smspec *spec, const cpl_table *linedat)
- cpl_error_code sm_comp_airglowcont (smspec *spec, const smparmodel modelpar, const smspec *molabstrans, const smspec *rscattrans, const smspec *mscattrans)

4.2.1 Detailed Description

This module provides functions for the computation of the sky model for the ESO ETC.

4.2.2 Define Documentation

4.2.2.1 #define SM_ERAD 6371.

average earth radius in km

4.2.2.2 #define SM_FILENAMELIST "sm_filenames.dat"

List of data paths and file names

4.2.2.3 #define SM_LAM_UNIT 1e-6

μm

4.2.2.4 #define SM_NSIGBIN 20

Minimum number of bins for σ width of airglow lines

4.2.2.5 #define SM_RADMINLAM 1.3 * 1e-6 / SM_LAM_UNIT

Minimum wavelength for extraction of radiance $\rightarrow$ faster code

4.2.2.6 #define SM_RRSTEP 1000

Line jumps for wavelength searches in LBLRTM/RFM library spectra

4.2.2.7 #define SM_SIGMAX 4.

Extension of Gaussian for airglow lines in σ

4.2.2.8 #define SM_SKYEMCOMP_H

4.2.3 Typedef Documentation

4.2.3.1 typedef struct _smparmodel_ smparmodel

Structure for sky emission parameters

Parameters

alt	altitude of object above horizon [0,90]
airmass	line-of-sight airmass
alpha	separation of sun and moon as seen from earth [0,180]
rho	separation of moon and object [0,180]
altmoon	altitude of moon above horizon [-90,90]
moondist	distance to moon (mean distance = 1; [0.945,1.055])
lon_ecl	heliocentric ecliptic longitude of object [-180,180]
lat_ecl	ecliptic latitude of object [-90,90]
ncomp	number of emissivity/temperature pairs ($\leq$ SM_MAXPAR)
eps[]	emissivity (factor for conversion black body $\rightarrow$ grey body)
temp[]	temperature in K
msolflux	monthly-averaged solar radio flux [s.f.u.]
season	period of the year (1: Dec/Jan, ..., 6: Oct/Nov.; 0: entire year)
time	period of the night (x/3 of night, x = 1,2,3; 0: entire night)
vac_air	vac[uum] or air wavelengths
pwv	precipitable water vapour in mm (-1: bimonthly mean)
rtcode	radiative transfer code L(BLRTM) or R(FM) for molecular spectra
resol	resolution of molecular spectra in library (crucial for run time)
libpath	directory path for library of molecular spectra
libstruct	name of file containing the structure of the selected LBLRTM/RFM library
libstruct1	name of structure file for time-dependent library
libstruct2	name of structure file for PWV-dependent library
datapath	directory path for input data files
solspecname	solar spectrum
lunirrname	file for lunar irradiance model parameters
miephasename	file for Mie scattering phase functions
mscatname	file for multiple scattering correction factors
o3transname	file for UV/optical ozone transmission
starspecname	mean spectrum of scattered starlight
zodtabname	V-brightness of zodiacal light [$10^{-8}\,\mathrm{W\,m^{-2}\,\mu m^{-1}\,sr^{-1}}$]

linetabname	sky line table
vardatname	file for airglow lines scaling parameters
acontname	file for airglow continuum (scaling parameters included)
incl	rules inclusion of sky emission components • format: "xxxxxxx" where x = "Y" (yes) or x = "N" (no) • position: – 1: scattered moonlight – 2: scattered starlight – 3: zodiacal light – 4: thermal emission by telescope/instrument – 5: molecular emission of lower atmosphere – 6: sky emission lines of upper atmosphere – 7: airglow continuum (residual continuum)

4.2.4 Function Documentation

4.2.4.1 cpl_error_code sm_comp_airglowcont (smspec * *spec*, const smparmodel *modelpar*, const smspec * *molabstrans*, const smspec * *rscattrans*, const smspec * *mscattrans*)

Reads airglow continuum spectrum (with uncertainties) and scales it depending on the emission layer width (related to airmass), the monthly-averaged solar flux in sfu, the season of the year, and the time of the day. Moreover, the change of continuum emission by molecular absorption, Rayleigh scattering, and aerosol extinction is considered by provided transmission curves. For the scattering curves the effective optical depth is reduced by airmass-dependent recipes which were obtained by 3D atmospheric scattering calculations for extended airglow emission.

INPUT:

Parameters

spec	desired wavelength grid
modelpar	sky emission parameters (see typedef of smparmodel)
molabstrans	transmission curve from radiative transfer code (scaled to airmass = 1)
rscattrans	transmission curve for Rayleigh scattering
mscattrans	transmission curve for Mie scattering

OUTPUT:

Parameters

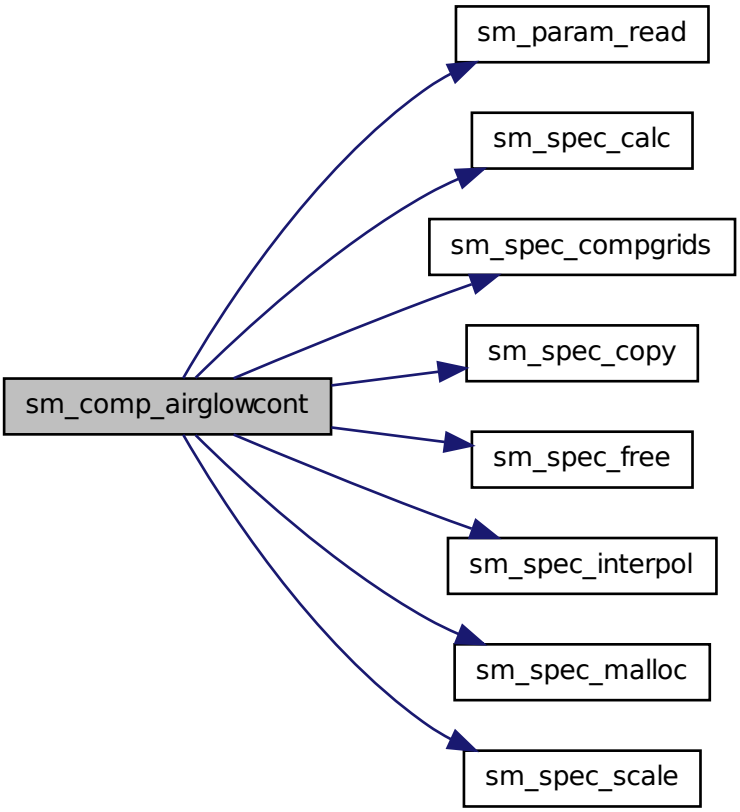
spec	airglow continuum

ERRORS:

- NDA: No data

- IDG: Inconsistent data grids
- FOF: File opening failed
- UFS: Unexpected file structure
- IIP: Invalid input parameter(s)
- ISM: Insufficient memory

Here is the call graph for this function:



4.2.4.2 double sm_comp_airtovac_single (const double *lam*)

Converts air wavelength [μm] to vacuum wavelength by using the formula of Edlen (1966)

INPUT:

Parameters

lam	air wavelength in μm

RETURN:

- vacuum wavelength in μm

ERRORS:

- none

4.2.4.3 cpl_error_code sm_comp_airtovac_spec (smspec * *spec*)

Converts spectrum of air wavelengths to spectrum of vacuum wavelengths by using the formula of Edlen (1966)

INPUT:

Parameters

spec	spectrum with air wavelengths

OUTPUT:

Parameters

spec	spectrum with vacuum wavelengths

ERRORS:

- none

4.2.4.4 double sm_comp_alttoairmass (const double *alt*)

Calculates airmass from altitude angle in deg using the formula of Rozenberg (1966). The maximum airmass is 40.

INPUT:

Parameters

alt	altitude angle in deg

RETURN:

- airmass

ERRORS:

- none

4.2.4.5 cpl_error_code sm_comp_calcmiescat (smspec * *mscattrans*)

Calculates transmission curve for Mie (aerosol) extinction by using the parametrisation given by Patat (2011) and a cut in the optical depth for wavelengths below $0.4\,\mu\mathrm{m}$

INPUT:

Parameters

mscattrans	desired wavelength grid in $\mu\mathrm{m}$

OUTPUT:

Parameters

mscattrans	Mie extinction curve (transmission at zenith)

ERRORS:

- none

4.2.4.6 double sm_comp_calcmiescat1 (const double *lam*)

Calculates optical depth my Mie (aerosol) extinction for given wavelength. The parametrisation of Patat (2011) is used and a cut in the optical depth for wavelengths below $0.4\,\mu\mathrm{m}$ is applied.

INPUT:

Parameters

lam	wavelength in $\mu\mathrm{m}$

RETURN:

- optical depth at zenith for given wavelength

ERRORS:

- none

4.2.4.7 cpl_error_code sm_comp_calcrayleighscat (smspec * *rscattrans*)

Calculates Rayleigh scattering transmission curve by using the parametrisation given by Liou (2002, P. 352) and the mean pressure profile of Paranal

INPUT:

Parameters

rscattrans	desired wavelength grid in μm

OUTPUT:

Parameters

rscattrans	Rayleigh scattering curve (transmission at zenith)

ERRORS:

- none

4.2.4.8 double sm_comp_calcrayleighscat1 (const double *lam*)

Calculates optical depth by Rayleigh scattering for given wavelength. The parametrisation of Liou (2002, P. 352) and and the mean pressure profile of Paranal are used.

INPUT:

Parameters

lam	wavelength in μm

RETURN:

- optical depth at zenith for given wavelength

ERRORS:

- none

4.2.4.9 cpl_error_code sm_comp_convertlinetab (smspec * *spec*, const cpl_table * *linedat*)

Converts table of emission lines with central wavelength, line flux, flux uncertainty, and line width into line spectrum with given wavelength grid. Gaussian line shape is assumed.

INPUT:

Parameters

spec	desired wavelength grid as smspec structure
linedat	CPL table with line data

OUTPUT:

Parameters

spec	sky line spectrum

ERRORS:

- NDA: No data
- ISD: Insufficient data points

4.2.4.10 cpl_error_code sm_comp_convertlinetabo (smspec * *spec*, const cpl_table * *linedat*)

Converts table of emission lines with central wavelength, line flux, flux uncertainty, and line width into line spectrum with given wavelength grid.

INPUT:

Parameters

spec	desired wavelength grid as smspec structure
linedat	CPL table with line data

OUTPUT:

Parameters

spec	sky line spectrum

ERRORS:

- NDA: No data
- ISD: Insufficient data points

4.2.4.11 cpl_error_code sm_comp_extrapolrad (smspec * *spec*, const smspec * *trans*, const smparmodel *modelpar*, const double *lim[2]*)

Extracts and rebins library LBLRTM/RFM molecular radiance spectrum and extrapolates it to the requested airmass, which is computed by means of the formula of Rozenberg (1966). For the optical depth dependent extrapolation, the corresponding transmission curve as provided by sm_comp_extrapoltrans is used.

INPUT:

Parameters

spec	desired wavelength grid
trans	transmission curve belonging to radiance spectrum (scaled to airmass = 1)
modelpar	sky emission parameters (see typedef of smparmodel)
lim	minimum and maximum wavelength extraction limits

OUTPUT:

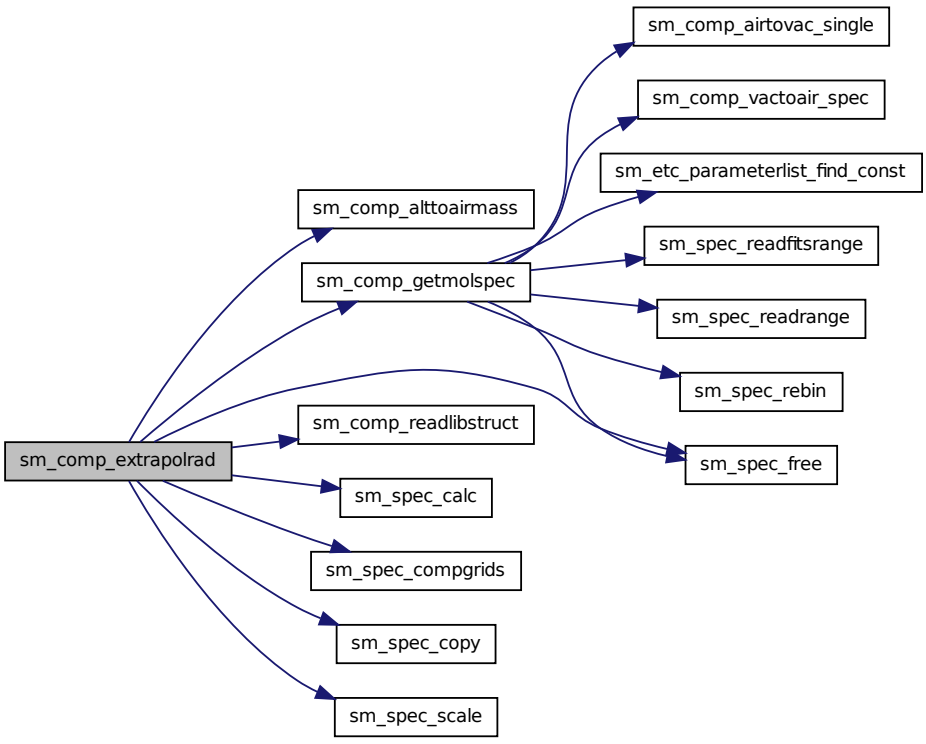
Parameters

spec	extracted, rebinned, and scaled LBLRTM/RFM radiance spectrum

ERRORS:

- NDA: No data
- IOD: Invalid order of data points
- IIP: Invalid input parameter(s)
- IDG: Inconsistent data grids

Here is the call graph for this function:



4.2.4.12 cpl_error_code sm_comp_extrapoltrans (smspec * *spec*, const smparmodel *modelpar*, const double *lim[2]*)

Extracts and rebins library LBLRTM/RFM molecular transmission spectrum and extrapolates it to airmass = 1. For the airmass calculations, the formula of Rozenberg (1966) is used.

INPUT:

Parameters

spec	desired wavelength grid
modelpar	sky emission parameters (see typedef of smparmodel)
lim	minimum and maximum wavelength extraction limits

OUTPUT:

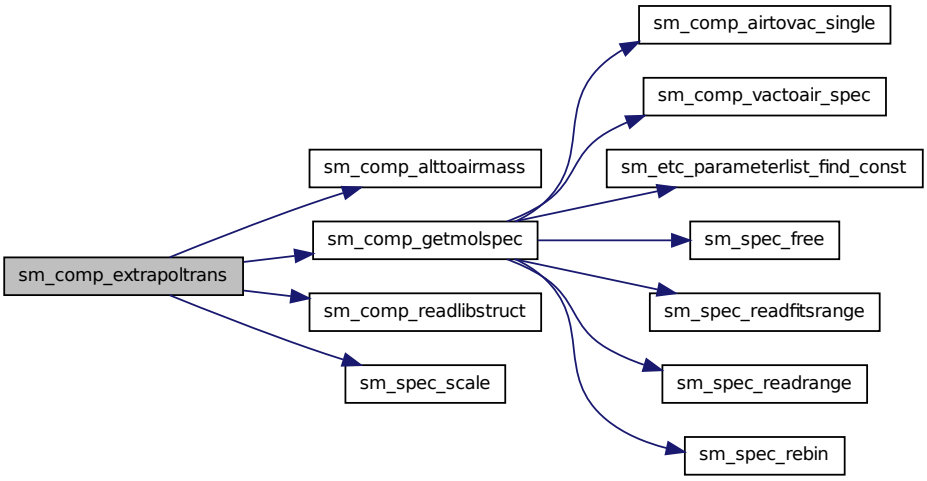
Parameters

spec	extracted, rebinned, and scaled LBLRTM/RFM transmission curve

ERRORS:

- NDA: No data
- IOD: Invalid order of data points
- IIP: Invalid input parameter(s)

Here is the call graph for this function:



4.2.4.13 cpl_error_code sm_comp_getlinespec (smspec * *spec*, const smparmodel *modelpar*, const smspec * *rscattrans*, const smspec * *mscattrans*)

Converts table of airglow emission lines (see Hanuschik 2003 and Rousselot et al. 2000) to spectrum (smspec structure) by applying wavelength-dependent correction factors which depend on object altitude, emission layer height, solar radio flux, period of the year, and period of the night (see sm_comp_readvarpar). The wavelengths are converted from vacuum to air if required. Moreover, the change of line intensity by molecular absorption, Rayleigh scattering, and aerosol extinction is considered. The former correction is realised by a precalculated library of lists of line-specific transmission correction factors for each radiative transfer code transmission spectrum. The latter corrections are performed by provided extinction curves for which the effective optical depth is reduced by airmass-dependent recipes which were obtained by 3D atmospheric scattering calculations for extended airglow emission.

INPUT:

Parameters

spec	desired wavelength grid
modelpar	airglow parameters (see typedef of smparmodel)
rscattrans	transmission curve for Rayleigh scattering
mscattrans	transmission curve for Mie scattering

OUTPUT:

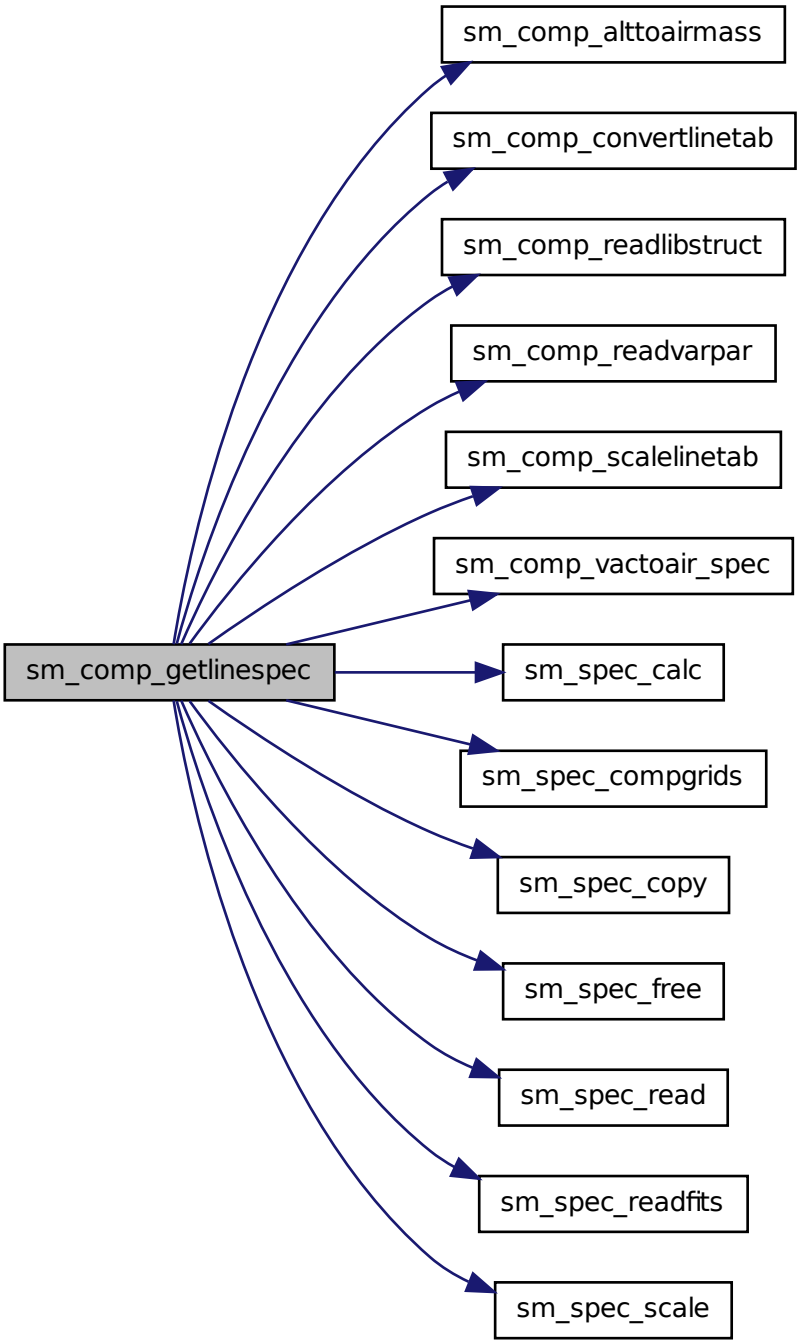
Parameters

spec	airglow emission line spectrum

ERRORS:

- IIP: Invalid input parameter(s)
- IDG: Inconsistent data grids
- see subroutines as well

Here is the call graph for this function:



4.2.4.14 cpl_error_code sm_comp_getmolspec (smspec * *spec*, const cpl_parameterlist * *libfilepar*, const double *lim[2]*)

Extracts and rebins library LBLRTM/RFM molecular radiance or transmission spectra. The file name and the decision between vacuum and air wavelengths have to be provided by a CPL parameter list. Each library file has to consist of the four columns wavelength, radiance/transmission for average atmospheric profiles, and radiance/transmission for minus and plus sigma atmospheric profiles. This structure is converted to the smspec structure, i.e. "lam", "flux", "dflux1" (negative error), and "dflux2" (positive error). The extraction-related wavelength range can be delimitated regardless of the desired grid in order to reduce computing time by avoiding the extraction of wavelengths that do not contribute significantly to the composite sky spectrum.

INPUT:

Parameters

spec	desired wavelength grid
libfilepar	CPL parameter list containing file name (+ path) and vacuum/air parameter
lim	minimum and maximum wavelength extraction limits

OUTPUT:

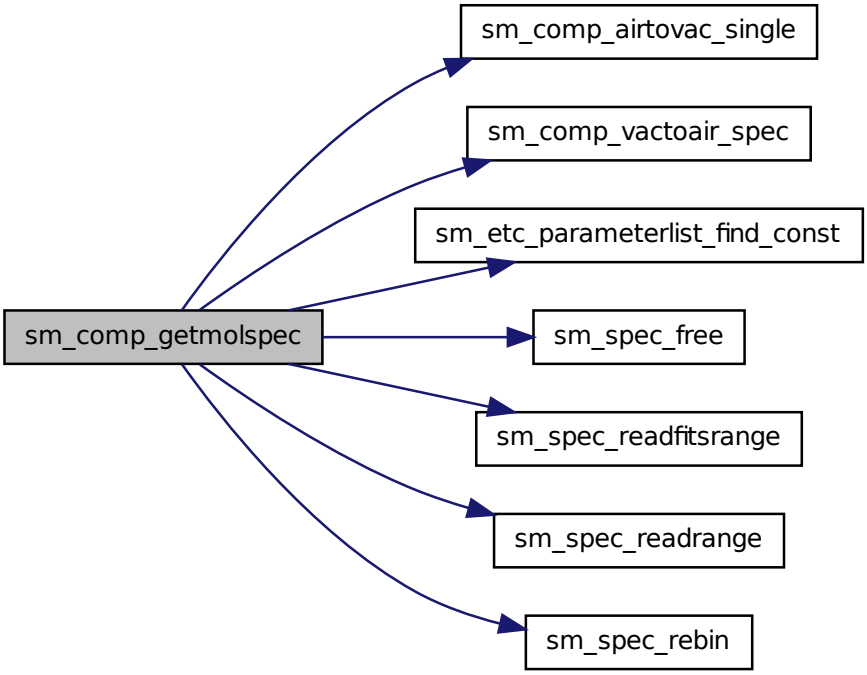
Parameters

spec	extracted and rebinned LBLRTM/RFM spectrum (radiance or transmission)

ERRORS:

- NDA: No data
- IOD: Invalid order of data points
- IIP: Invalid input parameter(s)
- IOS: Invalid object structure
- FOF: File opening failed
- IFE: Invalid file name extension
- CPL: Access out of range [warning]

Here is the call graph for this function:



4.2.4.15 cpl_error_code sm_comp_getmoonalbedo (smspec * *albedo*, const smparmodel *modelpar*)

Calculates the wavelength-dependent disc-equivalent lunar albedo from the Moon illuminance model of Kieffer & Stone (2005) based on ROLO data. Execpt for the libration-dependent terms, the full model is implemented. The required model coefficients are read from a file. As user parameter, the angle between Sun and Moon as seen from Earth is provided by a smparmodel structure. For the calculation of the Moon albedo, it is converted into absolute Moon phase angle and selenographic longitude of the Sun. The latter is required for considering differences in the brightness of the waxing and the waning Moon.

INPUT:

Parameters

albedo	desired wavelength grid
modelpar	sky emission parameters (see typedef of smparmodel)

OUTPUT:

Parameters

albedo	spectrum of lunar albedo

ERRORS:

- IIP: Invalid input parameters
- FOF: File opening failed

4.2.4.16 cpl_error_code sm_comp_lunskybright (smspec * *spec*, const smparmodel *modelpar*, const smspec * *solspec*, const smspec * *molabstrans*, const smspec * *rscattrans*, const smspec * *mscattrans*)

Evaluates predicted lunar part of sky brightness following Krisciunas & Schaefer (1991, PASP 103, 1033).

Original implementation by J. Thorstensen. Modified by F. Patat to generalise to UBVRI filters. Spectroscopic version by S. Noll.

This routine uses the solar spectrum as a proxy for the unattenuated lunar spectrum and two different transmission curves: one with and one wihout molecular absorption. The second one is important for the estimate of the amount of scattered moonlight at the object position and includes Rayleigh scattering and aerosol extinction. The model depends on object and moon altitude, their angular separation, moon phase (expressed by angular distance of moon and sun), and moon distance.

Note

- The model becomes unreliable for moon-object distances around and below 30°. There, the main contribution comes from aerosol scattering, which is highly variable and, therefore, difficult to predict.
- The routine uses the library transmission curve selected depending on the object altitude and not the moon altitude (see sm_comp_getmolspec). This reduces the computing time significantly, but especially the flux in centres of telluric absorption features can be overestimated if the moon is much lower than the object. However, for the composite sky emission spectrum the flux differences should be lower than 10% even in the A-band.

INPUT:

Parameters

spec	desired wavelength grid
modelpar	sky emission parameters (see typedef of smparmodel)
solspec	solar spectrum
molabstrans	transmission curve from radiative transfer code
rscattrans	transmission curve for Rayleigh scattering
mscattrans	transmission curve for Mie scattering

OUTPUT:

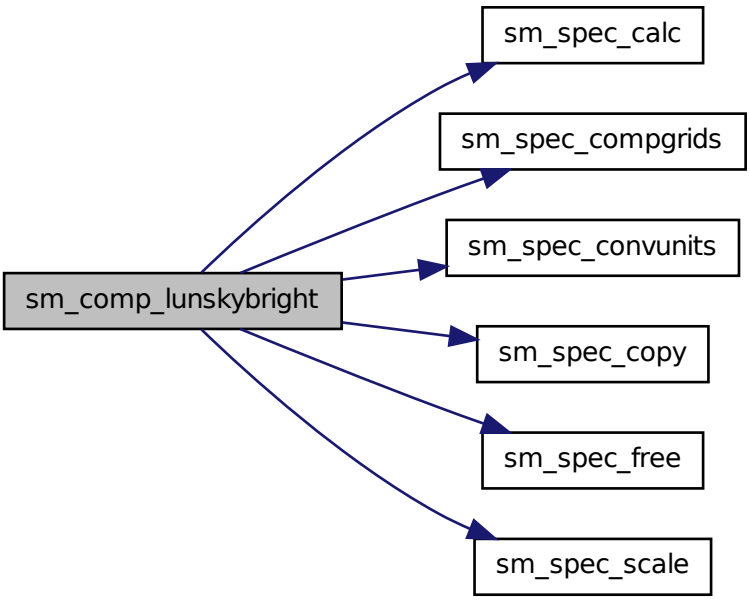
Parameters

spec	spectrum of lunar sky brightness

ERRORS:

- NDA: No data
- IIP: Invalid input parameter(s)
- IDG: Inconsistent data grids
- see subroutines as well

Here is the call graph for this function:



4.2.4.17 cpl_error_code sm_comp_lunskybright_v (smspec * *spec*, const smparmodel *modelpar*, const smspec * *solspec*, const smspec * *molabstrans*, const smspec * *rscattrans*, const smspec * *mscattrans*)

Calculates scattered moonlight. The Moon illuminance is approximated from Kieffer & Stone (2005) based on ROLO data. The scattering in the atmosphere is obtained from

3D single scattering calculations with multiple scattering correction (see sm_scat_moon and subroutines) and the Cerro Paranal extinction curve of Patat et al. (2011). A simple estimate which depends on the optical depth dependent effective airmass is taken for the molecular absorption of the moonlight. The absorption by the stratospheric ozone layer is considered separately. The model depends on object and Moon altitude, their angular separation, Moon phase (expressed by angular distance of Moon and Sun), and Moon distance.

INPUT:

Parameters

spec	desired wavelength grid
modelpar	sky emission parameters (see typedef of smparmodel)
solspec	solar spectrum
molabstrans	transmission curve from radiative transfer code
rscattrans	transmission curve for Rayleigh scattering
mscattrans	transmission curve for Mie scattering

OUTPUT:

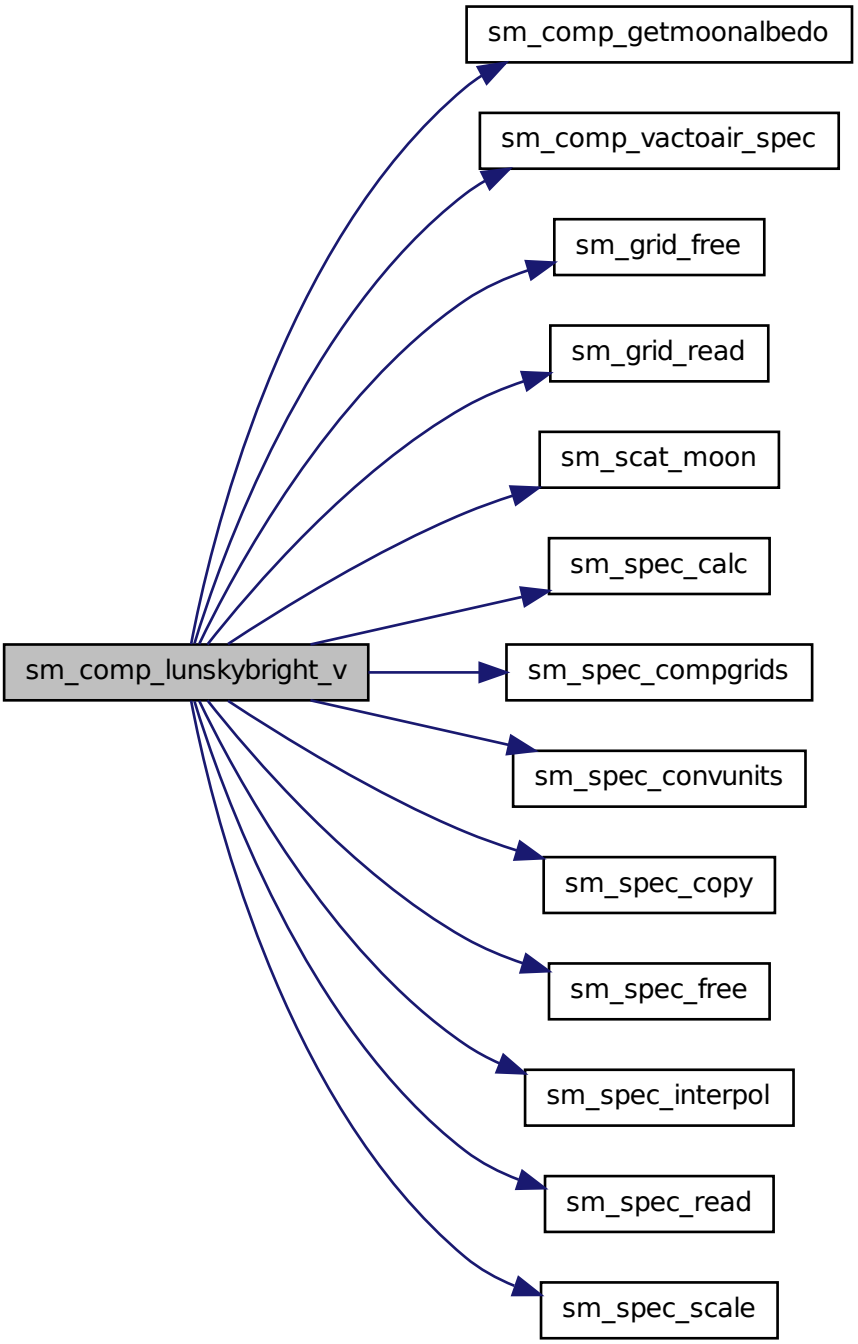
Parameters

spec	spectrum of lunar sky brightness

ERRORS:

- NDA: No data
- IIP: Invalid input parameter(s)
- IDG: Inconsistent data grids
- see subroutines as well

Here is the call graph for this function:



4.2.4.18 cpl_error_code sm_comp_readlibstruct (cpl_parameterlist * *libfilepar,* const smparmodel *modelpar,* const char * *spectype*)

Builds file name of library spectrum or line list depending on input parameters such as library path, radiative transfer code, target altitude or airmass, period of the year, period of the night, precipitable water vapour, resolution, or file type. The required parameters are provided by a smparmodel structure except for the file type which has to be given directly and can be either "R" for radiance spectrum, "T" for transmission spectrum, or "L" for list of line transmissions. The file name format is given by a library structure file. The routine returns a CPL parameter list with the file name and the values of all parameters building the file name plus a parameter for vacuum/air wavelengths. The latter is added to indicate whether the library spectrum (always in vacuum wavelengths) has to be converted to air wavelengths.

INPUT:

Parameters

modelpar	sky emission parameters (see typedef of smparmodel)
spectype	radiance ("R"), transmission ("T"), or line transmission ("L")?

OUTPUT:

Parameters

libfilepar	CPL parameter list containing file name (+ path) of library spectrum and its constituents

ERRORS:

- FOF: File opening failed
- UFS: Unexpected file structure
- IIP: Invalid input parameter(s)

4.2.4.19 cpl_error_code sm_comp_readvarpar (cpl_table * *varpar,* const smparmodel *modelpar*)

Reads data related to airglow scaling from file and fills a CPL table with feature-related correction factors and uncertainties. The resulting factors are related to the standard strengths of airglow emission features of the upper atmosphere given by Hanuschik (2003) and Rousselot et al. (2000). The scaling factors depend on the emission layer width (related to airmass), the monthly-averaged solar flux in sfu, the season of the year, and the time of the day. In addition, the relative Doppler widths σ_D of the different features are provided by the output table. They are derived from the read mol masses and temperatures of the emission layers.

INPUT:

Parameters

modelpar	sky emission parameters (see typedef of smparmodel)

OUTPUT:

Parameters

varpar	airglow variability data as CPL table with columns • *N:* feature number • *fac:* scaling factor for emission lines depending on feature • *dfac:* uncertainties of factors • *relwidth:* relative Doppler width

ERRORS:

- FOF: File opening failed
- UFS: Unexpected file structure
- IIP: Invalid input parameter(s)

4.2.4.20 cpl_error_code sm_comp_scalelinetab (cpl_table * *linedat*, const smspec * *linetab*, const cpl_table * *varpar*)

Writes line data in CPL table and corrects line fluxes by factors provided by *varpar* for the different features.

INPUT:

Parameters

linetab	table of emission lines (provided as smspec structure; dflux1 contains feature number)
varpar	line parameters (scaling factors and Doppler widths) provided as CPL table

OUTPUT:

Parameters

linedat	CPL table with columns "lam", "flux", "dflux", "feat", and "width"

ERRORS:

- NDA: No data
- ISD: Insufficient data points
- IOV: Invalid object value(s)

4.2.4.21 cpl_error_code sm_comp_scaletranscurv (smspec * *trans*, const smparmodel *modelpar*)

Scales transmission curve depending on airmass as calculated by the formula of Rozenberg (1966)

INPUT:

Parameters

trans	transmission curve at airmass = 1
modelpar	sky emission parameters (see typedef of smparmodel)

OUTPUT:

Parameters

trans	transmission curve for given altitude

ERRORS:

- NDA: No data
- IIP: Invalid input parameter(s)

4.2.4.22 cpl_error_code sm_comp_scatstarlight (smspec * *spec*, const smparmodel *modelpar*, const smspec * *molabstrans*)

Reads representative spectrum of scattered starlight, rebins it to given wavelength grid by means of linear interpolation, and considers telluric absorption for an effective airmass of 1.25. The spectrum was calculated by means of integrated starlight photometric data based on Pioneer 10/11 observations (see Toller et al. 1987; Leinert et al. 1998; Melchior et al. 2007), spectroscopic data by Mattila (1980), and 3D atmospheric scattering calculations (see Wolstencroft & van Breda 1967; Staude 1975; Bernstein et al. 2002).

INPUT:

Parameters

spec	desired wavelength grid
modelpar	sky emission parameters (see typedef of smparmodel)
molabstrans	transmission curve from radiative transfer code (scaled to airmass = 1)

OUTPUT:

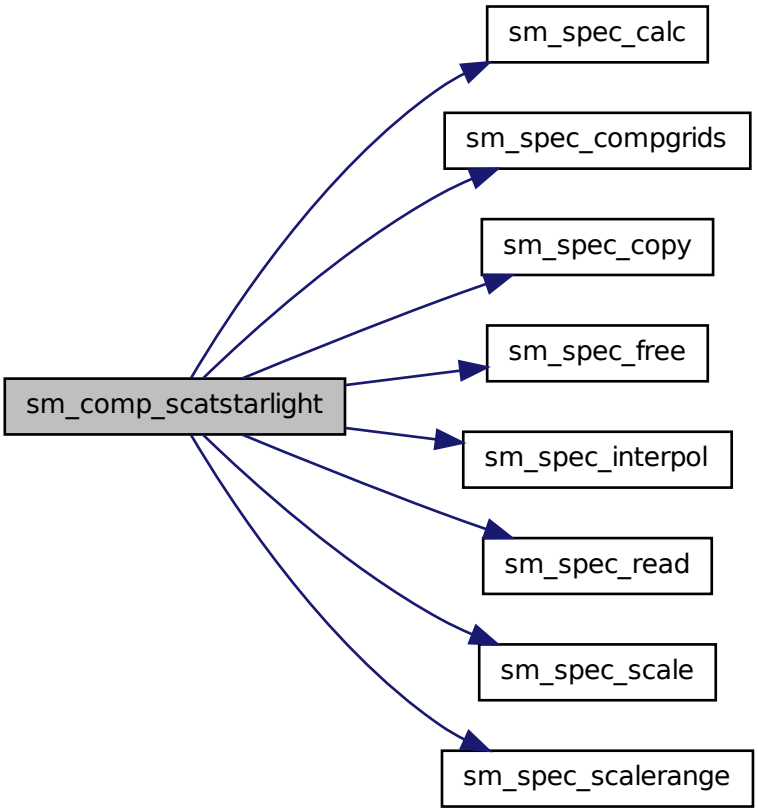
Parameters

spec	spectrum of scattered starlight

ERRORS:

- IDG: Inconsistent data grids
- see sm_spec_rebin

Here is the call graph for this function:



4.2.4.23 cpl_error_code sm_comp_skyemcomp (smspec * *radiance*, smspec * *transmission*, const smparmodel *modelpar*)

Creates sky emission spectrum and transmission curve at object position for given wavelength grid and input parameters. Imports solar spectrum, LBLRTM or RFM radiance and transmission spectra (ASCII or FITS files), and airglow emission line and continuum table, estimates lunar and zodical sky brightness, considers scattered starlight, and calculates thermal telescope/instrument emission. Using the "incl" parameter of the

smparmodel structure, the selection of the seven different components for the computation of the output radiance spectrum can be modified.

INPUT:

Parameters

radiance	wavelength grid of sky emission spectrum
transmission	wavelength grid of transmission curve
modelpar	input parameters related to sky emission (see typedef of smparmodel)

OUTPUT:

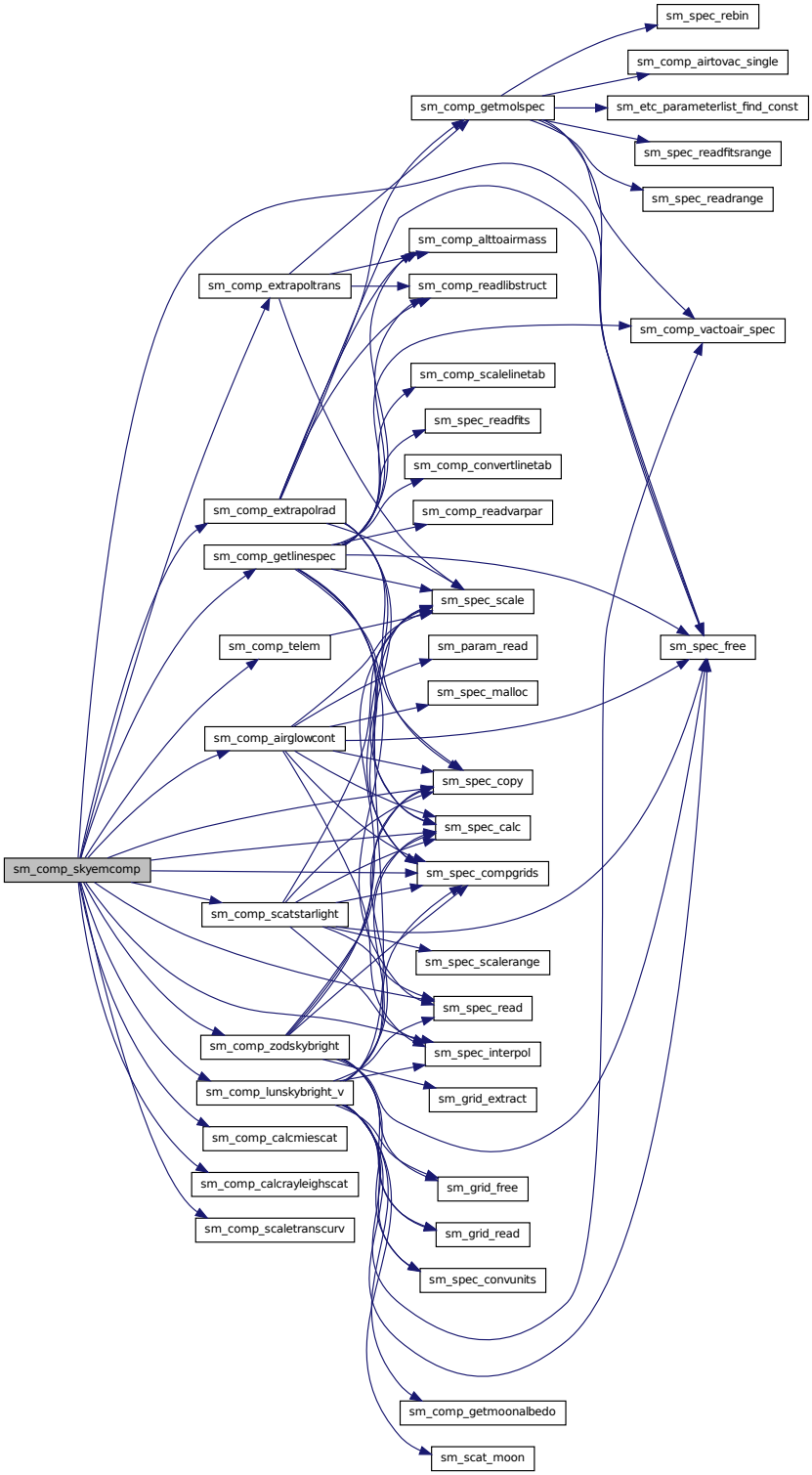
Parameters

radiance	sky emission spectrum (smspec structure, errors considered), units: μm, $\mathrm{phot\,s^{-1}m^{-2}\mu m^{-1}arcsec^{-2}}$
transmission	transmission curve (smspec structure, errors considered), units: μm, [0,1]

ERRORS:

- NDA: No data
- IDG: Inconsistent data grids
- see subroutines as well

Here is the call graph for this function:



4.2.4.24 cpl_error_code sm_comp_telem (smspec * *spec*, const smparmodel *modelpar*)

Computes thermal emission by telesope/instrument. Several components are possible. The routine assumes grey bodies depending on emissivity and temperature for each component. The order of the components matters. The first emissivity and temperature have to be for the main mirror. Any emission or absorption which is not related to the listed optical components is not considered.

The routine assumes that the absorption inside the instrument is handled by a response curve, which is applied somewhere else. For this reason, the emission of the different components is raised depending on the position along the light path. For example, the emission of the main mirror is increased by the reciprocal of the mirror transmission. This factor makes sure that the telescope emission can be added to the other sky model components, for which the flux is reduced by absorption of the main mirror and all other optical components. The routine simulates the apparent telescope/instrument emission that is not removed in the course of typical astronomical flux calibration.

INPUT:

Parameters

spec	desired wavelength grid
modelpar	sky emission parameters (see typedef of smparmodel)

OUTPUT:

Parameters

spec	apparent telescope/instrument emission

ERRORS:

- NDA: No data
- IIP: Invalid input parameter(s)
- IDR: Invalid data range

Here is the call graph for this function:

4.2.4.25 double sm_comp_vactoair_single (const double *lam*)

Converts vacuum wavelength [μm] to air wavelength by using the formula of Edlen (1966)

INPUT:

Parameters

lam	vacuum wavelength in μm

RETURN:

- air wavelength in μm

ERRORS:

- none

4.2.4.26 cpl_error_code sm_comp_vactoair_spec (smspec * *spec*)

Converts spectrum of vacuum wavelengths to spectrum of air wavelengths by using the formula of Edlen (1966)

INPUT:

Parameters

spec	spectrum with vacuum wavelengths

OUTPUT:

Parameters

spec	spectrum with air wavelengths

ERRORS:

- none

4.2.4.27 cpl_error_code sm_comp_zodskybright (smspec * *spec,* const smparmodel *modelpar,* const smspec * *solspec,* const smspec * *molabstrans,* const smspec * *rscattrans,* const smspec * *mscattrans*)

Derives sky brightness caused by the zodiacal light. The procedure is based on recipes provided in Leinert et al. (1998) and uses a table containing the brightness of the zodiacal light at $0.5\,\mu$m [$\mathrm{phot\,s^{-1}m^{-2}\mu m^{-1}arcsec^{-2}}$] dependent on the ecliptic coordinates of the object. For the zodiacal light spectrum a slightly reddened solar spectrum is taken.

The extinction of zodiacal light by molecular absorption, Rayleigh scattering, and aerosol extinction is considered by input transmission curves provided for airmass = 1. For the scattering curves the effective optical depth is reduced by recipes depending on the top-of-atmosphere line-of-sight zodiacal light intensity. The parametrisations used were obtained by 3D atmospheric scattering calculations for extended emission (see, e.g., Bernstein et al. 2002).

INPUT:

Parameters

spec	desired wavelength grid
modelpar	sky emission parameters (see typedef of smparmodel)
solspec	solar spectrum (note: normalisation for Colina et al. 1996)
molabstrans	transmission curve from radiative transfer code (scaled to airmass = 1)
rscattrans	transmission curve for Rayleigh scattering
mscattrans	transmission curve for Mie scattering

OUTPUT:

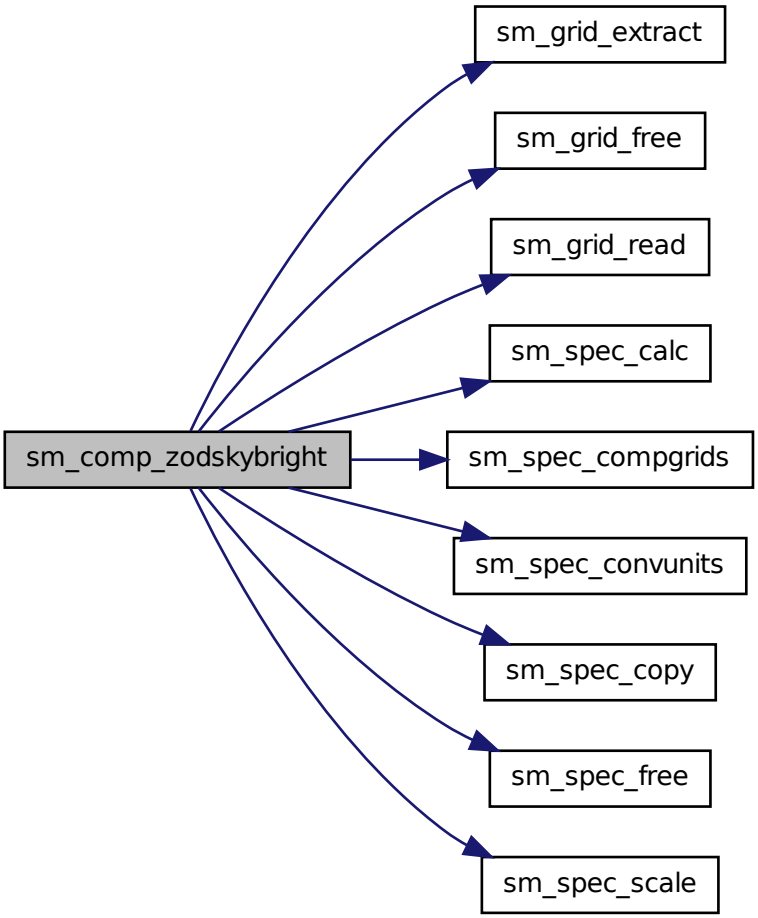
Parameters

spec	spectrum of zodiacal light

ERRORS:

- NDA: No data
- IIP: Invalid input parameter(s)
- IDG: Inconsistent data grids
- see subroutines as well

Here is the call graph for this function:



4.2.4.28 cpl_error_code sm_etc_calcmodel (cpl_table * *skytable*, const cpl_parameterlist * *params*)

Reads wavelength grid as CPL table and sky model parameters as CPL parameter list and uses this information to compute a corresponding radiance spectrum and transmission curve which are committed to the ETC as a CPL table including the input wavelength grid and the derived flux and flux error data. The output columns are "lam", "flux", "dflux1", "dflux2", "trans", "dtrans1", and "dtrans2".

INPUT:

Parameters

skytable	input wavelength grid as CPL table
params	CPL sky model parameter list

OUTPUT:

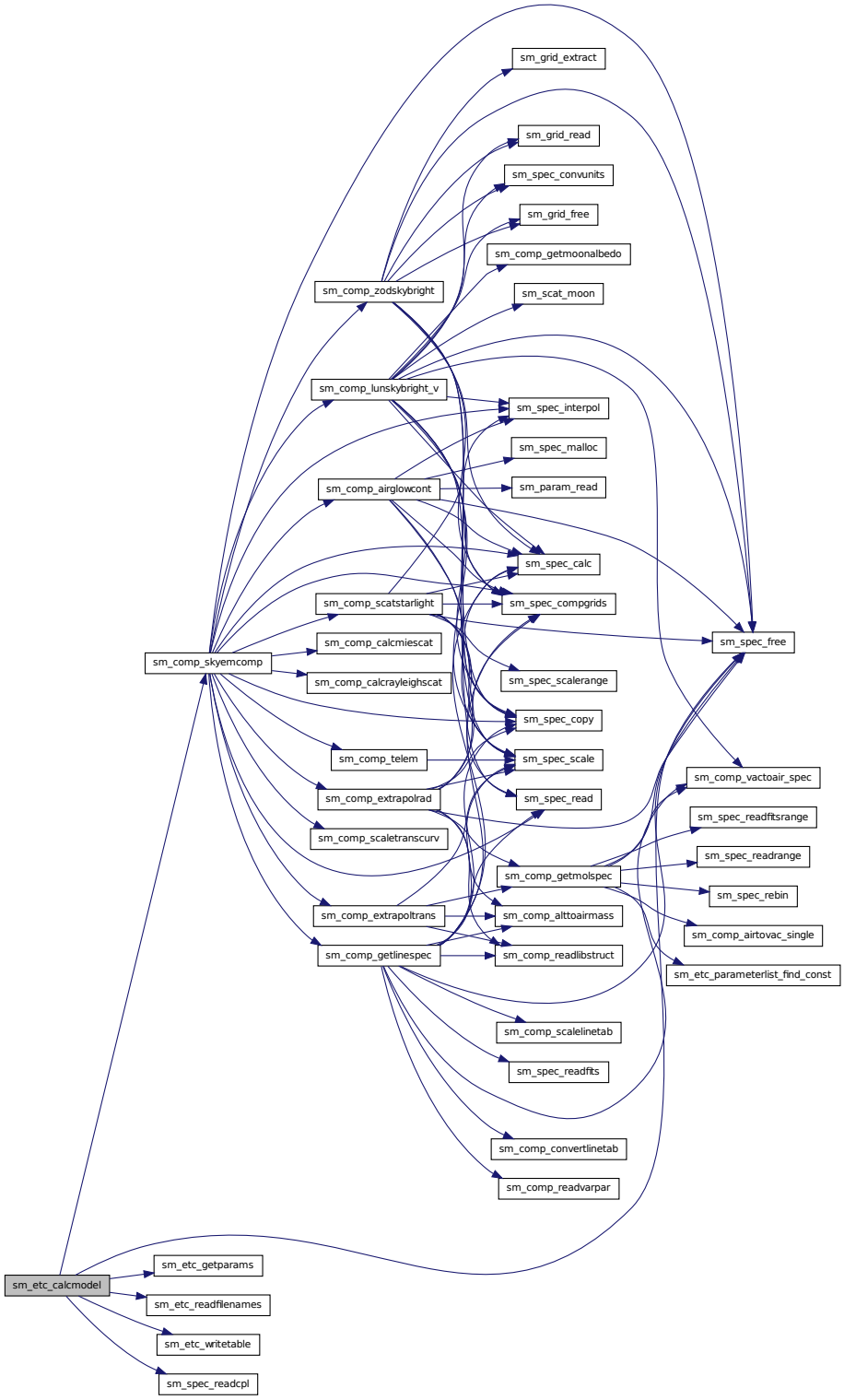
Parameters

skytable	CPL table with full radiance and transmission data

ERRORS:

- see subroutines

Here is the call graph for this function:



4.2.4.29 cpl_error_code sm_etc_getparams (smparmodel * *modelpar,* const cpl_parameterlist * *params*)

Reads sky model parameters from a CPL parameter list and put them in a smparmodel structure. The strings of comma-separated emissivity and temperature values are converted into double arrays. As only non-input parameter the airmass is added to the smparmodel structure. It is computed from altitude angle by using the formula of Rozenberg (1966).

INPUT:

Parameters

params	CPL sky model parameter list

OUTPUT:

Parameters

modelpar	smparmodel structure containing the read parameters

ERRORS:

- see subroutines

4.2.4.30 const cpl_parameter* sm_etc_parameterlist_find_const (const cpl_parameterlist * *self,* const char * *name,* cpl_type *type*)

Error-handling wrapper around cpl_parameterlist_find_const(), catching errors of mismatching parameter name and/or type.

INPUT:

Parameters

self	CPL parameter list
name	parameter name
type	CPL type of parameter

RETURN:

- CPL parameter

ERRORS:

- CPL_ERROR_DATA_NOT_FOUND
- CPL_ERROR_TYPE_MISMATCH

4.2.4.31 cpl_error_code sm_etc_readfilenames (smparmodel * *modelpar*, const cpl_parameterlist * *params*)

Reads data paths and file names from a file (path from CPL parameter list and name from header file) and put them in a smparmodel structure.

INPUT:

Parameters

params	CPL sky model parameter list

OUTPUT:

Parameters

modelpar	smparmodel structure containing the read parameters

ERRORS:

- FOF: File opening failed
- see sm_param_readcheck as well

4.2.4.32 cpl_error_code sm_etc_splitstring (cpl_array * *outval*, char * *instring*)

Splits a string of comma-separated values and converts the substrings into double values.

INPUT:

Parameters

instring	comma-separated list of values

OUTPUT:

Parameters

outval	CPL array of double values

ERRORS:

- ISM: Insufficient memory
- NDA: No data

4.2.4.33 cpl_error_code sm_etc_writetable (cpl_table * *skytable*, const smspec * *radiance*, const smspec * *transmission*)

Writes radiance spectrum and transmission curve which are provided as smspec structures to the CPL table of the ETC interface.

INPUT:

Parameters

skytable	input wavelength grid as CPL table
radiance	radiance spectrum as smspec structure
transmission	transmission curve as smspec structure

OUTPUT:

Parameters

skytable	CPL table with full radiance and transmission data

ERRORS:

- NDA: No wavelength data
- IDG: Inconsistent data grids

Chapter 5

Data Structure Documentation

5.1 _smdat_ Struct Reference

`#include <sm_general.h>`

Data Fields

- double lam
- double flux
- double dflux1
- double dflux2

5.1.1 Detailed Description

Structure for data points

Parameters

lam	wavelength
flux	flux
dflux1	(lower) flux error
dflux2	upper flux error

5.1.2 Field Documentation

5.1.2.1 double _smdat_::dflux1

5.1.2.2 double _smdat_::dflux2

5.1.2.3 double _smdat_::flux

5.1.2.4 double _smdat_::lam

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

- src/sm_general.h

5.2 _smgrid_ Struct Reference

`#include <sm_general.h>`

Data Fields

- int nx
- int ny
- double * xpos
- double * ypos
- double ** val

5.2.1 Detailed Description

Structure for coordinate grids

Parameters

nx	number of *x* coordinate values
ny	number of *y* coordinate values
**xpos*	vector of *x* coordinate values
**ypos*	vector of *y* coordinate values
***val*	matrix of data values for the coordinate grid

5.2.2 Field Documentation

5.2.2.1 int _smgrid_::nx

5.2.2.2 int _smgrid_::ny

5.2.2.3 double** _smgrid_::val

5.2.2.4 double* _smgrid_::xpos

5.2.2.5 double* _smgrid_::ypos

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

- src/sm_general.h

5.3 _smparam_ Struct Reference

`#include <sm_general.h>`

Data Fields

- char c [SM_LENLINE+2]
- int i
- double d
- int n

5.3.1 Detailed Description

Structure for a parameter value in different data types

Parameters

c	string (maximum length SM_LENLINE)
i	integer number
d	float number of double precision
n	number of value (for arrays; counted backwards)

5.3.2 Field Documentation

5.3.2.1 char _smparam_::c[SM_LENLINE+2]

5.3.2.2 double _smparam_::d

5.3.2.3 int _smparam_::i

5.3.2.4 int _smparam_::n

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

- src/sm_general.h

5.4 _smparmodel_ Struct Reference

`#include <sm_skyemcomp.h>`

Data Fields

- double alt
- double airmass

- double alpha
- double rho
- double altmoon
- double moondist
- double lon_ecl
- double lat_ecl
- int ncomp
- double eps [SM_MAXPAR]
- double temp [SM_MAXPAR]
- double msolflux
- int season
- int time
- char vac_air [SM_LENLINE+1]
- double pwv
- char rtcode [SM_LENLINE+1]
- double resol
- char libpath [SM_MAXLEN+1]
- char libstruct [SM_LENLINE+1]
- char libstruct1 [SM_LENLINE+1]
- char libstruct2 [SM_LENLINE+1]
- char datapath [SM_MAXLEN+1]
- char solspecname [SM_LENLINE+1]
- char lunirrname [SM_LENLINE+1]
- char miephasename [SM_LENLINE+1]
- char mscatname [SM_LENLINE+1]
- char o3transname [SM_LENLINE+1]
- char starspecname [SM_LENLINE+1]
- char zodtabname [SM_LENLINE+1]
- char linetabname [SM_LENLINE+1]
- char vardatname [SM_LENLINE+1]
- char acontname [SM_LENLINE+1]
- char incl [SM_LENLINE+1]

5.4.1 Detailed Description

Structure for sky emission parameters

Parameters

alt	altitude of object above horizon [0,90]
airmass	line-of-sight airmass
alpha	separation of sun and moon as seen from earth [0,180]
rho	separation of moon and object [0,180]
altmoon	altitude of moon above horizon [-90,90]
moondist	distance to moon (mean distance = 1; [0.945,1.055])
lon_ecl	heliocentric ecliptic longitude of object [-180,180]
lat_ecl	ecliptic latitude of object [-90,90]

ncomp	number of emissivity/temperature pairs ($\leq$ SM_MAXPAR)
eps[]	emissivity (factor for conversion black body $\rightarrow$ grey body)
temp[]	temperature in K
msolflux	monthly-averaged solar radio flux [s.f.u.]
season	period of the year (1: Dec/Jan, ..., 6: Oct/Nov.; 0: entire year)
time	period of the night (x/3 of night, x = 1,2,3; 0: entire night)
vac_air	vac[uum] or air wavelengths
pwv	precipitable water vapour in mm (-1: bimonthly mean)
rtcode	radiative transfer code L(BLRTM) or R(FM) for molecular spectra
resol	resolution of molecular spectra in library (crucial for run time)
libpath	directory path for library of molecular spectra
libstruct	name of file containing the structure of the selected LBLRTM/RFM library
libstruct1	name of structure file for time-dependent library
libstruct2	name of structure file for PWV-dependent library
datapath	directory path for input data files
solspecname	solar spectrum
lunirrname	file for lunar irradiance model parameters
miephasename	file for Mie scattering phase functions
mscatname	file for multiple scattering correction factors
o3transname	file for UV/optical ozone transmission
starspecname	mean spectrum of scattered starlight
zodtabname	V-brightness of zodiacal light [$10^{-8}\,\mathrm{W\,m^{-2}\,\mu m^{-1}\,sr^{-1}}$]
linetabname	sky line table
vardatname	file for airglow lines scaling parameters
acontname	file for airglow continuum (scaling parameters included)
incl	rules inclusion of sky emission components • format: "xxxxxxx" where x = "Y" (yes) or x = "N" (no) • position: – 1: scattered moonlight – 2: scattered starlight – 3: zodiacal light – 4: thermal emission by telescope/instrument – 5: molecular emission of lower atmosphere – 6: sky emission lines of upper atmosphere – 7: airglow continuum (residual continuum)

5.4.2 Field Documentation

5.4.2.1 char _smparmodel_::acontname[SM_LENLINE+1]

5.4.2.2 double _smparmodel_::airmass

5.4.2.3 double _smparmodel_::alpha

5.4.2.4 double _smparmodel_::alt

5.4.2.5 double _smparmodel_::altmoon

5.4.2.6 char _smparmodel_::datapath[SM_MAXLEN+1]

5.4.2.7 double _smparmodel_::eps[SM_MAXPAR]

5.4.2.8 char _smparmodel_::incl[SM_LENLINE+1]

5.4.2.9 double _smparmodel_::lat_ecl

5.4.2.10 char _smparmodel_::libpath[SM_MAXLEN+1]

5.4.2.11 char _smparmodel_::libstruct[SM_LENLINE+1]

5.4.2.12 char _smparmodel_::libstruct1[SM_LENLINE+1]

5.4.2.13 char _smparmodel_::libstruct2[SM_LENLINE+1]

5.4.2.14 char _smparmodel_::linetabname[SM_LENLINE+1]

5.4.2.15 double _smparmodel_::lon_ecl

5.4.2.16 char _smparmodel_::lunirrname[SM_LENLINE+1]

5.4.2.17 char _smparmodel_::miephasename[SM_LENLINE+1]

5.4.2.18 double _smparmodel_::moondist

5.4.2.19 char _smparmodel_::mscatname[SM_LENLINE+1]

5.4.2.20 double _smparmodel_::msolflux

5.4.2.21 int _smparmodel_::ncomp

5.4.2.22 char _smparmodel_::o3transname[SM_LENLINE+1]

5.4.2.23 double _smparmodel_::pwv

5.4.2.24 double _smparmodel_::resol

5.4.2.25 double _smparmodel_::rho

5.4.2.26 char _smparmodel_::rtcode[SM_LENLINE+1]

5.4.2.27 int _smparmodel_::season

5.4.2.28 char _smparmodel_::solspecname[SM_LENLINE+1]

5.4.2.29 char _smparmodel_::starspecname[SM_LENLINE+1]

5.4.2.30 double _smparmodel_::temp[SM_MAXPAR]

5.4.2.31 int _smparmodel_::time

5.4.2.32 char _smparmodel_::vac_air[SM_LENLINE+1]

5.4.2.33 char _smparmodel_::vardatname[SM_LENLINE+1]

5.4.2.34 char _smparmodel_::zodtabname[SM_LENLINE+1]

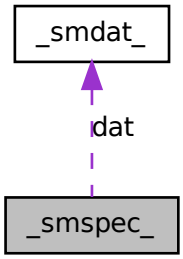
The documentation for this struct was generated from the following file:

- src/sm_skyemcomp.h

5.5 _smspec_ Struct Reference

`#include <sm_general.h>`

Collaboration diagram for _smspec_:



Data Fields

- int type

- int n
- smdat * dat

5.5.1 Detailed Description

Structure for spectra

Parameters

type	number of columns (1 = no flux, 2 = no errors, 3 = symmetric error, 4 = lower and upper error)
n	number of wavelengths
**dat*	vector of data points defined by smdat structure

5.5.2 Field Documentation

5.5.2.1 smdat* _smspec_::dat

5.5.2.2 int _smspec_::n

5.5.2.3 int _smspec_::type

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

- src/sm_general.h

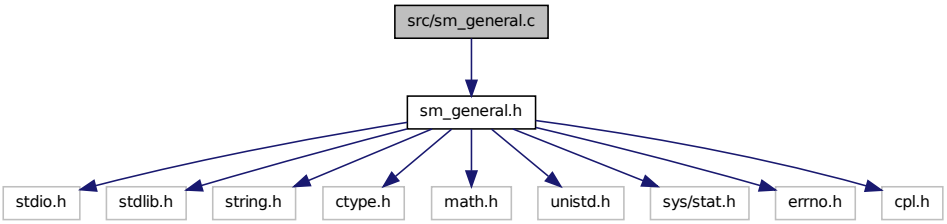
Chapter 6

File Documentation

6.1 src/sm_general.c File Reference

```
#include <sm_general.h>
```

Include dependency graph for sm_general.c:



Functions

- cpl_error_code sm_spec_malloc (smspec *spec, const int size)
- cpl_error_code sm_spec_create (smspec *outspec, const double limlam[2], const double dlam)
- cpl_error_code sm_spec_read (smspec *spec, const char *filename)
- cpl_error_code sm_spec_readrange (smspec *spec, const char *filename, const double limlam[2], const int step)
- cpl_error_code sm_spec_readcpl (smspec *spec, const cpl_table *cpltab)
- cpl_error_code sm_spec_readfits (smspec *spec, const char *filename)
- cpl_error_code sm_spec_readfitsrange (smspec *spec, const char *filename, const double limlam[2], const int step)

- cpl_error_code sm_spec_copy (smspec *outspec, const smspec *inspec)
- cpl_error_code sm_spec_compgrids (const smspec *spec1, const smspec *spec2)
- cpl_error_code sm_spec_join (smspec *spec, const smspec *errfunc, const int errflag)
- cpl_error_code sm_spec_split (smspec *spec, smspec *errfunc, const int errflag)
- cpl_error_code sm_spec_changetype (smspec *spec, const int type)
- cpl_error_code sm_spec_scalerange (smspec *spec, const double limlam[2], const char op, const double c)
- cpl_error_code sm_spec_scale (smspec *spec, const char op, const double c)
- cpl_error_code sm_spec_modval (smspec *spec, const double lam, const char op, const double c)
- cpl_error_code sm_spec_calc (smspec *spec, const char op, const smspec *opspec)
- cpl_error_code sm_spec_funct (smspec *spec, const char *funct, const char baselab)
- cpl_error_code sm_spec_functnoerr (smspec *spec, const char *funct)
- cpl_error_code sm_spec_convunits (smspec *spec, const double factor, const int lamexp)
- cpl_error_code sm_spec_changegrid (smspec *spec, const double factor, const char *scale)
- cpl_error_code sm_spec_average (const smspec *spec, const double limlam[2], double mean[3])
- cpl_error_code sm_spec_write (const smspec *spec, const char *filename)
- cpl_error_code sm_spec_print (const smspec *spec)
- cpl_error_code sm_spec_writecpl (cpl_table *cpltab, const smspec *spec)
- cpl_error_code sm_spec_writefits (const smspec *spec, const char *filename)
- cpl_error_code sm_spec_free (smspec *spec)
- cpl_error_code sm_spec_extract (smspec *outspec, const smspec *inspec, const double limlam[2])
- cpl_error_code sm_spec_rebin (smspec *outspec, const smspec *inspec)
- cpl_error_code sm_spec_interpol (smspec *outspec, const smspec *inspec)
- cpl_error_code sm_spec_convolve (smspec *spec, const int nkpix, const double *kernel)
- cpl_error_code sm_param_read (FILE *stream, smparam par[])
- cpl_error_code sm_param_readcheck (FILE *stream, smparam par[], const char *parname, const int npar)
- cpl_error_code sm_param_check (smparam par[], const char *parname, const int npar)
- cpl_error_code sm_grid_malloc (smgrid *xy, const int nx, const int ny)
- cpl_error_code sm_grid_read (smgrid *xy, const char *filename)
- cpl_error_code sm_grid_extract (const smgrid *xy, const double x0, const double y0, double *outval)
- cpl_error_code sm_grid_write (const smgrid *xy, const char *filename)
- cpl_error_code sm_grid_print (const smgrid *xy)
- cpl_error_code sm_grid_free (smgrid *xy)
- cpl_error_code sm_basic_chdir (const char *dir)
- cpl_error_code sm_basic_access (const char *pathname, const int mode)
- cpl_error_code sm_basic_mkdir (const char *dir, const mode_t mode)

- cpl_error_code sm_basic_createdir (const char *dir, const mode_t mode)
- void sm_basic_initstring (char *str, const long n)
- cpl_boolean sm_basic_isnumber (char *str)
- char * sm_basic_replacestring (char *instring, char *oldsubstr, char *newsubstr)
- char * sm_basic_rmcntrl (char *str)
- void sm_basic_rmcntrl_inplace (char *str)
- char * sm_basic_strtrim (char *str)
- void sm_basic_strtrim_inplace (char *str)
- void sm_basic_terminatestring (char *str)
- cpl_error_code sm_basic_interpollin (const double *x_out, double *y_out, const long n_out, const double *x_ref, const double *y_ref, const long n_ref)

6.1.1 Detailed Description

Basic routines used for the sky model

Author

Stefan Noll & ESO In-Kind Team Innsbruck

Since

22 Sep 2009

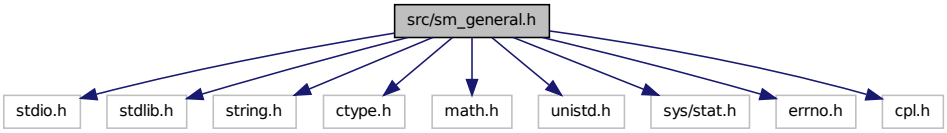
Date

18 Apr 2013

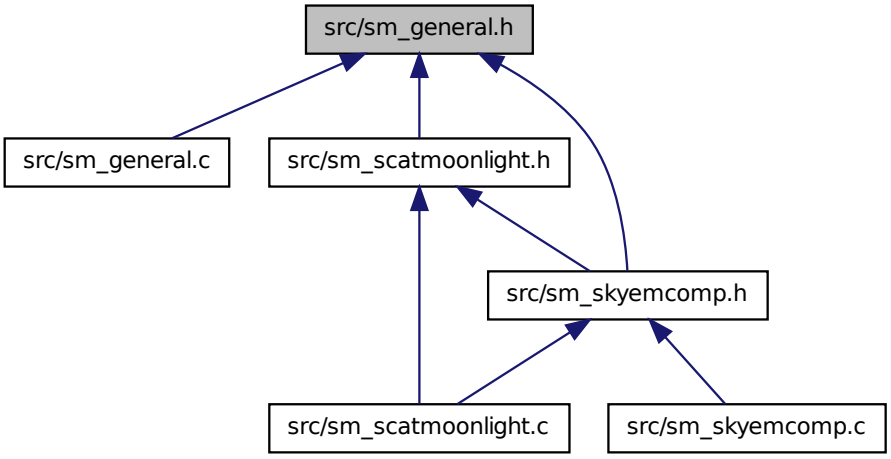
6.2 src/sm_general.h File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
#include <math.h>
#include <unistd.h>
#include <sys/stat.h>
#include <errno.h>
#include <cpl.h>
```

Include dependency graph for sm_general.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct _smdat_
- struct _smspec_
- struct _smparam_
- struct _smgrid_

Defines

- #define SM_GENERAL_H
- #define SM_BOOL

- #define F SM_F
- #define T SM_T
- #define SM_NMAXERR 100
- #define SM_LENLINE 160
- #define SM_MAXLEN 4000
- #define SM_MAXPAR 21
- #define SM_TOL 1e-7
- #define SM_SR_IN_ARCSEC2 4.254517e+10
- #define SM_GAS_CONST 8.31447
- #define SM_MOL_AIR_DRY 0.0289644
- #define SM_GRAV_ACC 9.80665
- #define SM_ERROR_FOF SM_ERROR_FOPEN
- #define SM_ERROR_ISM SM_ERROR_NOMEM
- #define SM_ERROR_EIS SM_ERROR_SUBROUTINE
- #define SM_ERROR_ISD SM_ERROR_INSUFF_DATA
- #define SM_ERROR_FOPEN_TXT "File opening failed"
- #define SM_ERROR_UFS_TXT "Unexpected file structure"
- #define SM_ERROR_IFE_TXT "Invalid file name extension"
- #define SM_ERROR_BDR_TXT "Bad directory"
- #define SM_ERROR_NDA_TXT "No data"
- #define SM_ERROR_ISD_TXT "Insufficient data points"
- #define SM_ERROR_IDG_TXT "Inconsistent data grids"
- #define SM_ERROR_IDR_TXT "Invalid data range"
- #define SM_ERROR_IOD_TXT "Invalid order of data points"
- #define SM_ERROR_IIP_TXT "Invalid input parameter(s)"
- #define SM_ERROR_IOV_TXT "Invalid object value(s)"
- #define SM_ERROR_IOS_TXT "Invalid object structure"
- #define SM_ERROR_SUBROUTINE_TXT "Error in subroutine"
- #define SM_ERROR_ACCES_TXT "Permission denied"
- #define SM_ERROR_LOOP_TXT "Too many symbolic links"
- #define SM_ERROR_NAMETOOLONG_TXT "Pathname too long"
- #define SM_ERROR_NOENT_TXT "File/dir does not exist"
- #define SM_ERROR_NOTDIR_TXT
- #define SM_ERROR_ROFS_TXT
- #define SM_ERROR_FAULT_TXT
- #define SM_ERROR_INVAL_TXT "Mode was incorrectly specified"
- #define SM_ERROR_IO_TXT "I/O error occurred"
- #define SM_ERROR_NOMEM_TXT "Insufficient memory"
- #define SM_ERROR_TXTBSY_TXT
- #define SM_ERROR_EXIST_TXT "File/dir already exists"
- #define SM_ERROR_NOSPC_TXT "No space left on device"
- #define SM_ERROR_PERM_TXT
- #define SM_ERROR_LINK_TXT "Could not create symbolic link"
- #define SM_ERROR_UNDEF_TXT "Undefined error"
- #define SM_ERROR_FOF_TXT SM_ERROR_FOPEN_TXT
- #define SM_ERROR_ISM_TXT SM_ERROR_NOMEM_TXT
- #define SM_ERROR_EIS_TXT SM_ERROR_SUBROUTINE_TXT

Typedefs

- typedef enum SM_BOOL smbool
- typedef enum _sm_error_code_ sm_error_code
- typedef struct _smdat_ smdat
- typedef struct _smspec_ smspec
- typedef struct _smparam_ smparam
- typedef struct _smgrid_ smgrid

Enumerations

- enum SM_BOOL { SM_F, SM_T }
- enum _sm_error_code_ {

  SM_ERROR_FOPEN = CPL_ERROR_EOL + 11, SM_ERROR_UFS = CPL_-ERROR_EOL + 12, SM_ERROR_IFE = CPL_ERROR_EOL + 13, SM_ERROR_-NDA = CPL_ERROR_EOL + 20,

  SM_ERROR_INSUFF_DATA = CPL_ERROR_EOL + 21, SM_ERROR_IDG = CPL_-ERROR_EOL + 22, SM_ERROR_IDR = CPL_ERROR_EOL + 23, SM_ERROR_-IOD = CPL_ERROR_EOL + 24,

  SM_ERROR_IIP = CPL_ERROR_EOL + 30, SM_ERROR_IOV = CPL_ERROR_-EOL + 31, SM_ERROR_IOS = CPL_ERROR_EOL + 32, SM_ERROR_SUBROUTINE = CPL_ERROR_EOL + 40,

  SM_ERROR_ACCES = CPL_ERROR_EOL + 50, SM_ERROR_LOOP = CPL_-ERROR_EOL + 51, SM_ERROR_NAMETOOLONG = CPL_ERROR_EOL + 52, SM_ERROR_NOENT = CPL_ERROR_EOL + 53,

  SM_ERROR_NOTDIR = CPL_ERROR_EOL + 54, SM_ERROR_ROFS = CPL_-ERROR_EOL + 55, SM_ERROR_FAULT = CPL_ERROR_EOL + 56, SM_ERROR_-INVAL = CPL_ERROR_EOL + 57,

  SM_ERROR_IO = CPL_ERROR_EOL + 58, SM_ERROR_NOMEM = CPL_ERROR_-EOL + 59, SM_ERROR_TXTBSY = CPL_ERROR_EOL + 60, SM_ERROR_-EXIST = CPL_ERROR_EOL + 61,

  SM_ERROR_NOSPC = CPL_ERROR_EOL + 62, SM_ERROR_PERM = CPL_-ERROR_EOL + 63, SM_ERROR_BADUSERINPUT = CPL_ERROR_EOL + 70, SM_ERROR_LINK = CPL_ERROR_EOL + 71,

  SM_ERROR_RFM = CPL_ERROR_EOL + 81, SM_ERROR_UNDEF = CPL_-ERROR_EOL + 80 }

Functions

- cpl_error_code sm_spec_malloc (smspec *spec, const int size)
- cpl_error_code sm_spec_create (smspec *outspec, const double limlam[2], const double dlam)
- cpl_error_code sm_spec_read (smspec *spec, const char *filename)
- cpl_error_code sm_spec_readrange (smspec *spec, const char *filename, const double limlam[2], const int step)

- cpl_error_code sm_spec_readcpl (smspec *spec, const cpl_table *cpltab)
- cpl_error_code sm_spec_readfits (smspec *spec, const char *filename)
- cpl_error_code sm_spec_readfitsrange (smspec *spec, const char *filename, const double limlam[2], const int step)
- cpl_error_code sm_spec_copy (smspec *outspec, const smspec *inspec)
- cpl_error_code sm_spec_compgrids (const smspec *spec1, const smspec *spec2)
- cpl_error_code sm_spec_join (smspec *spec, const smspec *errfunc, const int errflag)
- cpl_error_code sm_spec_split (smspec *spec, smspec *errfunc, const int errflag)
- cpl_error_code sm_spec_changetype (smspec *spec, const int type)
- cpl_error_code sm_spec_scalerange (smspec *spec, const double limlam[2], const char op, const double c)
- cpl_error_code sm_spec_scale (smspec *spec, const char op, const double c)
- cpl_error_code sm_spec_modval (smspec *spec, const double lam, const char op, const double c)
- cpl_error_code sm_spec_calc (smspec *spec, const char op, const smspec *opspec)
- cpl_error_code sm_spec_funct (smspec *spec, const char *funct, const char baselab)
- cpl_error_code sm_spec_functnoerr (smspec *spec, const char *funct)
- cpl_error_code sm_spec_convunits (smspec *spec, const double factor, const int lamexp)
- cpl_error_code sm_spec_changegrid (smspec *spec, const double factor, const char *scale)
- cpl_error_code sm_spec_average (const smspec *spec, const double limlam[2], double mean[3])
- cpl_error_code sm_spec_write (const smspec *spec, const char *filename)
- cpl_error_code sm_spec_print (const smspec *spec)
- cpl_error_code sm_spec_writecpl (cpl_table *cpltab, const smspec *spec)
- cpl_error_code sm_spec_writefits (const smspec *spec, const char *filename)
- cpl_error_code sm_spec_free (smspec *spec)
- cpl_error_code sm_spec_extract (smspec *outspec, const smspec *inspec, const double limlam[2])
- cpl_error_code sm_spec_rebin (smspec *outspec, const smspec *inspec)
- cpl_error_code sm_spec_interpol (smspec *outspec, const smspec *inspec)
- cpl_error_code sm_spec_convolve (smspec *spec, const int nkpix, const double *kernel)
- cpl_error_code sm_param_read (FILE *stream, smparam par[])
- cpl_error_code sm_param_readcheck (FILE *stream, smparam par[], const char *parname, const int npar)
- cpl_error_code sm_param_check (smparam par[], const char *parname, const int npar)
- cpl_error_code sm_grid_malloc (smgrid *xy, const int nx, const int ny)
- cpl_error_code sm_grid_read (smgrid *xy, const char *filename)
- cpl_error_code sm_grid_extract (const smgrid *xy, const double x0, const double y0, double *outval)
- cpl_error_code sm_grid_write (const smgrid *xy, const char *filename)
- cpl_error_code sm_grid_print (const smgrid *xy)

- cpl_error_code sm_grid_free (smgrid *xy)
- cpl_error_code sm_basic_chdir (const char *dir)
- cpl_error_code sm_basic_access (const char *pathname, const int mode)
- cpl_error_code sm_basic_mkdir (const char *dir, const mode_t mode)
- cpl_error_code sm_basic_createdir (const char *dir, const mode_t mode)
- void sm_basic_initstring (char *str, const long n)
- cpl_boolean sm_basic_isnumber (char *str)
- char * sm_basic_replacestring (char *instring, char *oldsubstr, char *newsubstr)
- char * sm_basic_rmcntrl (char *str)
- void sm_basic_rmcntrl_inplace (char *str)
- char * sm_basic_strtrim (char *str)
- void sm_basic_strtrim_inplace (char *str)
- void sm_basic_terminatestring (char *str)
- cpl_error_code sm_basic_interpollin (const double *x_out, double *y_out, const long n_out, const double *x_ref, const double *y_ref, const long n_ref)

6.2.1 Detailed Description

Header for basic routines used for the sky model

Author

Stefan Noll & ESO In-Kind Team Innsbruck

Since

22 Sep 2009

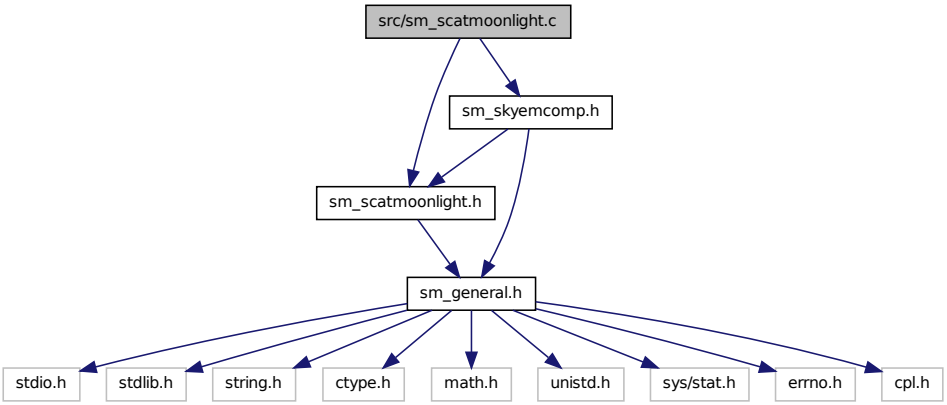
Date

11 Dec 2012

6.3 src/sm_scatmoonlight.c File Reference

```
#include <sm_scatmoonlight.h>
#include <sm_skyemcomp.h>
```

Include dependency graph for sm_scatmoonlight.c:



Functions

- cpl_error_code sm_scat_moon (smspec *spec, const double z, const double zmoon, const double rho, const smspec *molabstrans, const smspec *rscattrans, const smspec *mscattrans, const smgrid *miephase, const smgrid *multiscat)
- cpl_error_code sm_scat_createdatatab (cpl_table *tab)
- cpl_error_code sm_scat_calcextcurv (cpl_table *tautab)
- cpl_error_code sm_scat_calctau (double *lam0, double *tau0r, double *tau0m, const double tau0, const cpl_table *tautab)
- double sm_scat_tau2lam (const double tau0, const smspec *scattrans)
- cpl_error_code sm_scat_calcscatmoon (double *iscat, double *xeff, const int ns, const double z0, const double zeta, const double theta, const double flux0, const double tau0, const double cextr, const double cextm, const double cexta, const smgrid *miephase, const double lam0)
- double sm_scat_geteffcoldens (const double z, const double sig, const int ns, const int phasefunc)
- double sm_scat_getmolecdens (const double h)
- double sm_scat_getaerosoldens (const double h)
- double sm_scat_getmiephasefunc (const double theta)
- cpl_error_code sm_scat_calcscatspec (smspec *spec, const cpl_table *tab, const smspec *molabstrans, const smspec *rscattrans, const smspec *mscattrans)

6.3.1 Detailed Description

Routines for calculation of scattered moonlight

Author

Stefan Noll & ESO In-Kind Team Innsbruck

Since

05 Jun 2012

Date

21 Jan 2013

6.3.2 Function Documentation

6.3.2.1 cpl_error_code sm_scat_calcextcurv (cpl_table * *tautab*)

Calculates Rayleigh and Mie extinction curves and the total optical depths depending on wavelength. The parametrisations are optimised for Cerro Paranal. The covered wavelengths range from 0.3 to $30.0\,\mu\mathrm{m}$.

INPUT:

Parameters

tautab	empty CPL table

OUTPUT:

Parameters

tautab	table with optical depths by Rayleigh scattering and aerosol extinction

ERRORS:

- none

6.3.2.2 cpl_error_code sm_scat_calcscatmoon (double * *iscat,* double * *xeff,* const int *ns,* const double *z0,* const double *zeta,* const double *theta,* const double *flux0,* const double *tau0,* const double *cextr,* const double *cextm,* const double *cexta,* const smgrid * *miephase,* const double *lam0*)

Calculation of intensity of scattered moonlight, which is scattered into the given line of sight through the Earth's atmosphere, and effective airmass relative to a direct path from zenith to observer. Rayleigh scattering, aerosol extinction, and molecular absorption are considered. For the latter, the same particle distribution as for Rayleigh scattering is assumed (optimal for molecular oxygen). The wavelength-dependent phase functions for Mie scattering were calculated based on optimised log-normal aerosol distributions (Warneck & Williams 2012) and the Bohren-Huffman Mie scattering algorithm (IDL code BHMIE). The single scattering formalism in an spherical atmosphere used is described by Wolstencroft & van Breda (1967), Staude (1975), and Bernstein et al. (2002). As in the latter reference polarisation is neglected. For estimating the contribution of multiple scattering, 3D double scattering calculations were performed, which consider mixed

scattering at air molecules, aerosols, and the surface. The results of the multiple scattering calculations are provided as data table to this routine.

INPUT:

Parameters

ns	maximum number of data points (- 1) along a path in the atmosphere
z0	line-of-sight zenith distance in rad
zeta	zenith distance of radiation source in rad
theta	angle between line of sight and radiation source
flux0	flux of radiation source
tau0	optical depth at zenith
cextr	Rayleigh extinction cross section in cm ^{2}
cextm	Mie extinction cross section in cm ^{2}
cexta	molecular absorption cross section in cm ^{2}
miephase	grid structure for Mie scattering phase functions
lam0	wavelength for Mie scattering

OUTPUT:

Parameters

iscat	intensity scattered into line-of-sight
xeff	effective airmass

ERRORS:

- IIP: Invalid input parameter(s)

6.3.2.3 cpl_error_code sm_scat_calcscatspec (smspec * *spec*, const cpl_table * *tab*, const smspec * *molabstrans*, const smspec * *rscattrans*, const smspec * *mscattrans*)

Calculates spectrum of scattering intensities. The conversion of optical depths into wavelengths is performed by means of the transmission curves for Rayleigh and Mie scattering. Scattering intensities are interpolated. The scattering intensities are roughly corrected for molecular absorption by means of effective airmasses corresponding to the extinction optical depths based on scattering and absorption.

INPUT:

Parameters

spec	desired wavelength grid
tab	list of scattering intensities and effective airmasses for different optical depths
molabstrans	transmission curve from radiative transfer code
rscattrans	transmission curve for Rayleigh scattering
mscattrans	transmission curve for aerosol extinction

OUTPUT:

Parameters

spec	spectrum of scattering intensities

ERRORS:

- IDG: Inconsistent data grids

6.3.2.4 cpl_error_code sm_scat_calctau (double * *lam0,* double * *tau0r,* double * *tau0m,* const double *tau0,* const cpl_table * *tautab*)

Retrieves wavelength, Rayleigh optical depth, and Mie optical depth for an input total optical depth. The extinction curves used are optimised for Cerro Paranal. For optical depths beyond the covered wavelength range, the limiting wavelength and optical depths consistent with the Rayleigh and Mie fractions for this wavelength are returned.

INPUT:

Parameters

tau0	optical depth at zenith
tautab	CPL table with extinction curves

OUTPUT:

Parameters

lam0	wavelength related to input optical depth
tau0r	optical depth by Rayleigh scattering
tau0m	optical depth by Mie extinction

ERRORS:

- none

6.3.2.5 cpl_error_code sm_scat_createdatatab (cpl_table * *tab*)

Writes CPL table with grid of zenithal optical depths for atmospheric scattering calculations.

INPUT:

Parameters

tab	empty CPL table

OUTPUT:

Parameters

tab	grid of zenithal optical depths

ERRORS:

- none

6.3.2.6 double sm_scat_getaerosoldens (const double *h*)

Provides aerosol density in cm $^{-3}$ for given height in km. Uses exponential formula. The constants are from Staude (1975).

INPUT:

Parameters

h	height above ground in km

RETURN:

- aerosol density in cm $^{-3}$

ERRORS:

- none

6.3.2.7 double sm_scat_geteffcoldens (const double *z*, const double *sig*, const int *ns*, const int *phasefunc*)

Calculates effective column density in cm $^{-2}$ for a path from top of atmosphere to point of reference by considering the given zenith distance. The size of the volume elements for the integration increases with height in order to consider the decrease of density with height.

INPUT:

Parameters

z	zenith distance in rad at point of reference
sig	height in km of point of reference relative to centre of earth
ns	maximum number of data points (- 1) along a path in the atmosphere
phasefunc	0 = Rayleigh scattering, 1 = Mie (aerosol) scattering

RETURN:

- effective column density in cm $^{-2}$ for given path

ERRORS:

- IIP: Invalid input parameter(s)

6.3.2.8 double sm_scat_getmiephasefunc (const double *theta*)

Provides phase function for Mie scattering. For scattering angles up to 20 deg the empirical Green et al. (1971) phase function for aerosols is used. The phase function for larger scattering angles is extrapolated by a linear fit of the phase function in a log P - log θ diagram. This simple approach neglects the typical increase of phase functions for large scattering angles. However, in view of the strong peak for forward scattering (P covers about two orders of magnitude), the details of the phase function beyond the empirical part are not crucial for scattering calculations.

INPUT:

Parameters

theta	scattering angle

RETURN:

- value of phase function for given scattering angle

ERRORS:

- none

6.3.2.9 double sm_scat_getmolecdens (const double *h*)

Provides molecular density in cm $^{-3}$ for given height in km. Uses barometric formula. The constants are from Staude (1975).

INPUT:

Parameters

h	height above ground in km

RETURN:

- molecular density in cm $^{-3}$

ERRORS:

- none

6.3.2.10 cpl_error_code sm_scat_moon (**smspec** * *spec,* const double *z,* const double *zmoon,* const double *rho,* const **smspec** * *molabstrans,* const **smspec** * *rscattrans,* const **smspec** * *mscattrans,* const **smgrid** * *miephase,* const **smgrid** * *multiscat*)

Atmospheric scattering calculations for moonlight. Required input parameters are line-of-sight zenith distance, zenith distance of Moon, and angle between line of sight and

Moon. All parameters have to be provided in deg. Be aware that there are impossible combinations of these three parameters, which cause a programme termination without scattering results. In the case of a successful completion, the procedure returns a spectrum of scattering intensities per arcsec ^{2} for an input total Moon flux of 1. The required conversion from τ to wavelength is carried out by means of the input zenithal transmission curves for Rayleigh and Mie scattering. Before multiplication, the transmission curve for molecular absorption is adapted to the effective airmasses related to the scattering paths of both scattering processes. This procedure works best for Rayleigh scattering and absorption by O $_2$.

INPUT:

Parameters

spec	desired wavelength grid
z	line-of-sight zenith distance in deg
zmoon	zenith distance of Moon in deg
rho	angle between line of sight and Moon in deg
molabstrans	transmission curve from radiative transfer code
rscattrans	transmission curve for Rayleigh scattering
mscattrans	transmission curve for Mie scattering
miephase	grid structure for Mie scattering phase functions
multiscat	grid structure for multiple scattering correction factors

OUTPUT:

Parameters

spec	spectrum of scattered moonlight (input flux = 1)

ERRORS:

- IIP: Invalid input parameter(s)
- IDG: Inconsistent data grids

6.3.2.11 double sm_scat_tau2lam (const double *tau0,* const smspec * *scattrans*)

Conversion of an optical depth into a wavelength by means of a given transmission curve.

INPUT:

Parameters

tau0	optical depth at zenith
scattrans	transmission curve

RETURN:

- wavelength related to input optical depth

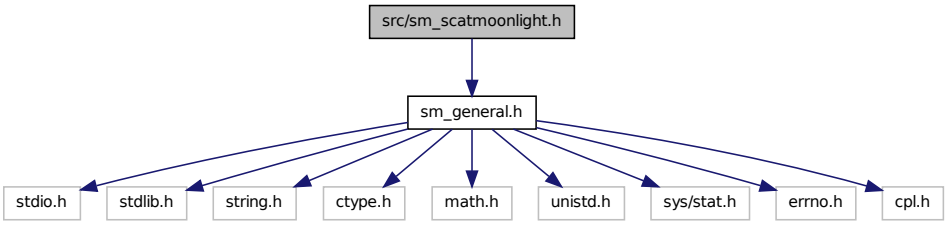
ERRORS:

- none

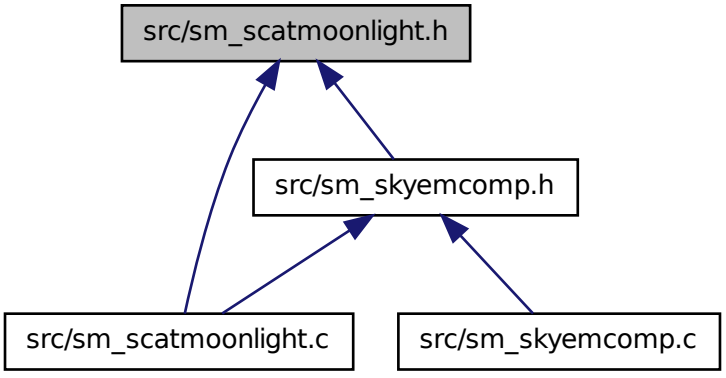
6.4 src/sm_scatmoonlight.h File Reference

```
#include <sm_general.h>
```

Include dependency graph for sm_scatmoonlight.h:



This graph shows which files directly or indirectly include this file:



- #define SM_SCATMOONLIGHT_H
- #define SM_NS 20
- #define SM_LNSCALE 0.5

- #define SM_R 6371.
- #define SM_HMAX 200.
- #define SM_H 2.64
- #define SM_N0_R 2.67e19
- #define SM_H0_R 7.99
- #define SM_N0_M 1.11e4
- #define SM_H0_M 1.2
- #define SM_SSA_M 0.97
- #define SM_NTAU1 7
- #define SM_DTAU1 0.01
- #define SM_NTAU2 31
- #define SM_DTAU2 0.03
- #define SM_NTAU3 20
- #define SM_DTAU3 0.2
- #define SM_REFTAU0 0.02
- cpl_error_code sm_scat_moon (smspec *spec, const double z, const double zmoon, const double rho, const smspec *molabstrans, const smspec *rscattrans, const smspec *mscattrans, const smgrid *miephase, const smgrid *multiscat)
- cpl_error_code sm_scat_createdatatab (cpl_table *tab)
- cpl_error_code sm_scat_calcextcurv (cpl_table *tautab)
- cpl_error_code sm_scat_calctau (double *lam0, double *tau0r, double *tau0m, const double tau0, const cpl_table *tautab)
- double sm_scat_tau2lam (const double tau0, const smspec *scattrans)
- cpl_error_code sm_scat_calcscatmoon (double *iscat, double *xeff, const int ns, const double z0, const double zeta, const double theta, const double flux0, const double tau0, const double cextr, const double cextm, const double cexta, const smgrid *miephase, const double lam0)
- double sm_scat_geteffcoldens (const double z, const double sig, const int ns, const int phasefunc)
- double sm_scat_getmolecdens (const double h)
- double sm_scat_getaerosoldens (const double h)
- double sm_scat_getmiephasefunc (const double theta)
- cpl_error_code sm_scat_calcscatspec (smspec *spec, const cpl_table *tab, const smspec *molabstrans, const smspec *rscattrans, const smspec *mscattrans)

6.4.1 Detailed Description

Header for routines for calculation of scattered moonlight

Author

Stefan Noll & ESO In-Kind Team Innsbruck

Since

05 Jun 2012

Date

25 Jan 2013

6.4.2 Define Documentation

6.4.2.1 #define SM_DTAU1 0.01

First τ step size

6.4.2.2 #define SM_DTAU2 0.03

Second τ step size

6.4.2.3 #define SM_DTAU3 0.2

Second τ step size

6.4.2.4 #define SM_H 2.64

Altitude of observer in km

6.4.2.5 #define SM_H0_M 1.2

Scale height of aerosol number density in km

6.4.2.6 #define SM_H0_R 7.99

Scale height of atmosphere in km

6.4.2.7 #define SM_HMAX 200.

Top of atmosphere in km

6.4.2.8 #define SM_LNSCALE 0.5

Scaling factor in km for unitless logarithmic bins

6.4.2.9 #define SM_N0_M 1.11e4

Density of aerosols at surface in cm $^{-3}$

6.4.2.10 #define SM_N0_R 2.67e19

Density of atmosphere at surface in cm $^{-3}$

6.4.2.11 #define SM_NS 20

Number of points for integration of particle number density

6.4.2.12 #define SM_NTAU1 7

Number of τ values for first step size

6.4.2.13 #define SM_NTAU2 31

Number of τ values for second step size

6.4.2.14 #define SM_NTAU3 20

Number of τ values for third step size

6.4.2.15 #define SM_R 6371.

Radius of Earth in km

6.4.2.16 #define SM_REFTAU0 0.02

Reference scattering optical depth for effective absorption airmass calculation

6.4.2.17 #define SM_SCATMOONLIGHT_H

Number of points for integration of particle number density

6.4.2.18 #define SM_SSA_M 0.97

Single scattering albedo for aerosols

6.4.3 Function Documentation

6.4.3.1 cpl_error_code sm_scat_calcextcurv (cpl_table * *tautab*)

Calculates Rayleigh and Mie extinction curves and the total optical depths depending on wavelength. The parametrisations are optimised for Cerro Paranal. The covered wavelengths range from 0.3 to $30.0\,\mu\mathrm{m}$.

INPUT:

Parameters

tautab	empty CPL table

OUTPUT:

Parameters

tautab	table with optical depths by Rayleigh scattering and aerosol extinction

ERRORS:

- none

6.4.3.2 cpl_error_code sm_scat_calcscatmoon (double * *iscat,* double * *xeff,* const int *ns,* const double *z0,* const double *zeta,* const double *theta,* const double *flux0,* const double *tau0,* const double *cextr,* const double *cextm,* const double *cexta,* const smgrid * *miephase,* const double *lam0*)

Calculation of intensity of scattered moonlight, which is scattered into the given line of sight through the Earth's atmosphere, and effective airmass relative to a direct path from zenith to observer. Rayleigh scattering, aerosol extinction, and molecular absorption are considered. For the latter, the same particle distribution as for Rayleigh scattering is assumed (optimal for molecular oxygen). The wavelength-dependent phase functions for Mie scattering were calculated based on optimised log-normal aerosol distributions (Warneck & Williams 2012) and the Bohren-Huffman Mie scattering algorithm (IDL code BHMIE). The single scattering formalism in an spherical atmosphere used is described by Wolstencroft & van Breda (1967), Staude (1975), and Bernstein et al. (2002). As in the latter reference polarisation is neglected. For estimating the contribution of multiple scattering, 3D double scattering calculations were performed, which consider mixed scattering at air molecules, aerosols, and the surface. The results of the multiple scattering calculations are provided as data table to this routine.

INPUT:

Parameters

ns	maximum number of data points (- 1) along a path in the atmosphere
z0	line-of-sight zenith distance in rad
zeta	zenith distance of radiation source in rad
theta	angle between line of sight and radiation source
flux0	flux of radiation source
tau0	optical depth at zenith
cextr	Rayleigh extinction cross section in cm ^{2}
cextm	Mie extinction cross section in cm ^{2}
cexta	molecular absorption cross section in cm ^{2}
miephase	grid structure for Mie scattering phase functions
lam0	wavelength for Mie scattering

OUTPUT:

Parameters

iscat	intensity scattered into line-of-sight
xeff	effective airmass

ERRORS:

- IIP: Invalid input parameter(s)

6.4.3.3 cpl_error_code sm_scat_calcscatspec (smspec * *spec,* const cpl_table * *tab,* const smspec * *molabstrans,* const smspec * *rscattrans,* const smspec * *mscattrans*)

Calculates spectrum of scattering intensities. The conversion of optical depths into wavelengths is performed by means of the transmission curves for Rayleigh and Mie scattering. Scattering intensities are interpolated. The scattering intensities are roughly corrected for molecular absorption by means of effective airmasses corresponding to the extinction optical depths based on scattering and absorption.

INPUT:

Parameters

spec	desired wavelength grid
tab	list of scattering intensities and effective airmasses for different optical depths
molabstrans	transmission curve from radiative transfer code
rscattrans	transmission curve for Rayleigh scattering
mscattrans	transmission curve for aerosol extinction

OUTPUT:

Parameters

spec	spectrum of scattering intensities

ERRORS:

- IDG: Inconsistent data grids

6.4.3.4 cpl_error_code sm_scat_calctau (double * *lam0,* double * *tau0r,* double * *tau0m,* const double *tau0,* const cpl_table * *tautab*)

Retrieves wavelength, Rayleigh optical depth, and Mie optical depth for an input total optical depth. The extinction curves used are optimised for Cerro Paranal. For optical depths beyond the covered wavelength range, the limiting wavelength and optical depths consistent with the Rayleigh and Mie fractions for this wavelength are returned.

INPUT:

Parameters

tau0	optical depth at zenith
tautab	CPL table with extinction curves

OUTPUT:

Parameters

lam0	wavelength related to input optical depth
tau0r	optical depth by Rayleigh scattering
tau0m	optical depth by Mie extinction

ERRORS:

- none

6.4.3.5 cpl_error_code sm_scat_createdatatab (cpl_table * *tab*)

Writes CPL table with grid of zenithal optical depths for atmospheric scattering calculations.

INPUT:

Parameters

tab	empty CPL table

OUTPUT:

Parameters

tab	grid of zenithal optical depths

ERRORS:

- none

6.4.3.6 double sm_scat_getaerosoldens (const double *h*)

Provides aerosol density in cm $^{-3}$ for given height in km. Uses exponential formula. The constants are from Staude (1975).

INPUT:

Parameters

h	height above ground in km

RETURN:

- aerosol density in cm $^{-3}$

ERRORS:

- none

6.4.3.7 double sm_scat_geteffcoldens (const double *z,* const double *sig,* const int *ns,* const int *phasefunc*)

Calculates effective column density in cm $^{-2}$ for a path from top of atmosphere to point of reference by considering the given zenith distance. The size of the volume elements for the integration increases with height in order to consider the decrease of density with height.

INPUT:

Parameters

z	zenith distance in rad at point of reference
sig	height in km of point of reference relative to centre of earth
ns	maximum number of data points (- 1) along a path in the atmosphere
phasefunc	0 = Rayleigh scattering, 1 = Mie (aerosol) scattering

RETURN:

- effective column density in cm $^{-2}$ for given path

ERRORS:

- IIP: Invalid input parameter(s)

6.4.3.8 double sm_scat_getmiephasefunc (const double *theta*)

Provides phase function for Mie scattering. For scattering angles up to 20 deg the empirical Green et al. (1971) phase function for aerosols is used. The phase function for larger scattering angles is extrapolated by a linear fit of the phase function in a log P - log θ diagram. This simple approach neglects the typical increase of phase functions for large scattering angles. However, in view of the strong peak for forward scattering (P covers about two orders of magnitude), the details of the phase function beyond the empirical part are not crucial for scattering calculations.

INPUT:

Parameters

theta	scattering angle

RETURN:

- value of phase function for given scattering angle

ERRORS:

- none

6.4.3.9 double sm_scat_getmolecdens (const double *h*)

Provides molecular density in cm $^{-3}$ for given height in km. Uses barometric formula. The constants are from Staude (1975).

INPUT:

Parameters

h	height above ground in km

RETURN:

- molecular density in cm $^{-3}$

ERRORS:

- none

6.4.3.10 cpl_error_code sm_scat_moon (smspec * *spec*, const double *z*, const double *zmoon*, const double *rho*, const smspec * *molabstrans*, const smspec * *rscattrans*, const smspec * *mscattrans*, const smgrid * *miephase*, const smgrid * *multiscat*)

Atmospheric scattering calculations for moonlight. Required input parameters are line-of-sight zenith distance, zenith distance of Moon, and angle between line of sight and Moon. All parameters have to be provided in deg. Be aware that there are impossible combinations of these three parameters, which cause a programme termination without scattering results. In the case of a successful completion, the procedure returns a spectrum of scattering intensities per arcsec ^{2} for an input total Moon flux of 1. The required conversion from τ to wavelength is carried out by means of the input zenithal transmission curves for Rayleigh and Mie scattering. Before multiplication, the transmission curve for molecular absorption is adapted to the effective airmasses related to the scattering paths of both scattering processes. This procedure works best for Rayleigh scattering and absorption by O $_2$.

INPUT:

Parameters

spec	desired wavelength grid
z	line-of-sight zenith distance in deg
zmoon	zenith distance of Moon in deg
rho	angle between line of sight and Moon in deg
molabstrans	transmission curve from radiative transfer code
rscattrans	transmission curve for Rayleigh scattering
mscattrans	transmission curve for Mie scattering
miephase	grid structure for Mie scattering phase functions
multiscat	grid structure for multiple scattering correction factors

OUTPUT:

Parameters

spec	spectrum of scattered moonlight (input flux = 1)

ERRORS:

- IIP: Invalid input parameter(s)
- IDG: Inconsistent data grids

6.4.3.11 double sm_scat_tau2lam (const double *tau0,* const smspec * *scattrans*)

Conversion of an optical depth into a wavelength by means of a given transmission curve.

INPUT:

Parameters

tau0	optical depth at zenith
scattrans	transmission curve

RETURN:

- wavelength related to input optical depth

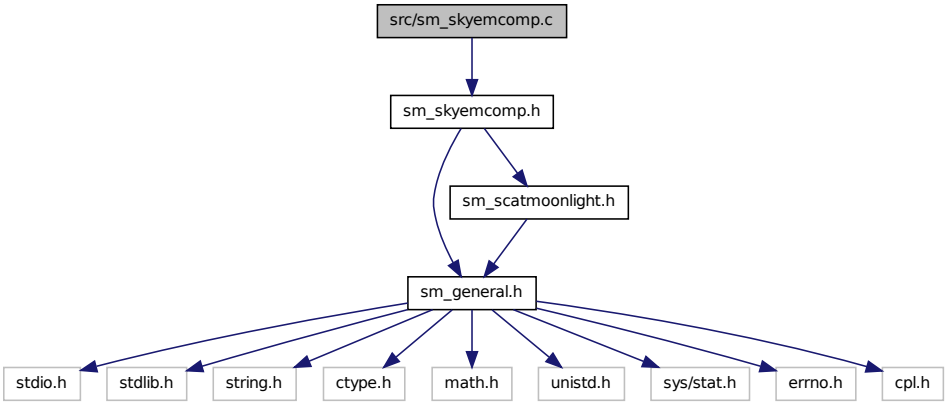
ERRORS:

- none

6.5 src/sm_skyemcomp.c File Reference

```
#include <sm_skyemcomp.h>
```

Include dependency graph for sm_skyemcomp.c:



Functions

- cpl_error_code sm_etc_calcmodel (cpl_table *skytable, const cpl_parameterlist *params)
- cpl_error_code sm_etc_getparams (smparmodel *modelpar, const cpl_parameterlist *params)
- const cpl_parameter * sm_etc_parameterlist_find_const (const cpl_parameterlist *self, const char *name, cpl_type type)
- cpl_error_code sm_etc_splitstring (cpl_array *outval, char *instring)
- cpl_error_code sm_etc_readfilenames (smparmodel *modelpar, const cpl_parameterlist *params)
- cpl_error_code sm_etc_writetable (cpl_table *skytable, const smspec *radiance, const smspec *transmission)
- cpl_error_code sm_comp_skyemcomp (smspec *radiance, smspec *transmission, const smparmodel modelpar)
- cpl_error_code sm_comp_extrapoltrans (smspec *spec, const smparmodel modelpar, const double lim[2])
- double sm_comp_alttoairmass (const double alt)
- cpl_error_code sm_comp_getmolspec (smspec *spec, const cpl_parameterlist *libfilepar, const double lim[2])
- cpl_error_code sm_comp_readlibstruct (cpl_parameterlist *libfilepar, const smparmodel modelpar, const char *spectype)
- double sm_comp_vactoair_single (const double lam)
- cpl_error_code sm_comp_vactoair_spec (smspec *spec)
- double sm_comp_airtovac_single (const double lam)

- cpl_error_code sm_comp_airtovac_spec (smspec *spec)
- double sm_comp_calcrayleighscat1 (const double lam)
- cpl_error_code sm_comp_calcrayleighscat (smspec *rscattrans)
- double sm_comp_calcmiescat1 (const double lam)
- cpl_error_code sm_comp_calcmiescat (smspec *mscattrans)
- cpl_error_code sm_comp_scaletranscurv (smspec *trans, const smparmodel modelpar)
- cpl_error_code sm_comp_lunskybright (smspec *spec, const smparmodel modelpar, const smspec *solspec, const smspec *molabstrans, const smspec *rscattrans, const smspec *mscattrans)
- cpl_error_code sm_comp_lunskybright_v (smspec *spec, const smparmodel modelpar, const smspec *solspec, const smspec *molabstrans, const smspec *rscattrans, const smspec *mscattrans)
- cpl_error_code sm_comp_getmoonalbedo (smspec *albedo, const smparmodel modelpar)
- cpl_error_code sm_comp_scatstarlight (smspec *spec, const smparmodel modelpar, const smspec *molabstrans)
- cpl_error_code sm_comp_zodskybright (smspec *spec, const smparmodel modelpar, const smspec *solspec, const smspec *molabstrans, const smspec *rscattrans, const smspec *mscattrans)
- cpl_error_code sm_comp_telem (smspec *spec, const smparmodel modelpar)
- cpl_error_code sm_comp_extrapolrad (smspec *spec, const smspec *trans, const smparmodel modelpar, const double lim[2])
- cpl_error_code sm_comp_getlinespec (smspec *spec, const smparmodel modelpar, const smspec *rscattrans, const smspec *mscattrans)
- cpl_error_code sm_comp_readvarpar (cpl_table *varpar, const smparmodel modelpar)
- cpl_error_code sm_comp_scalelinetab (cpl_table *linedat, const smspec *linetab, const cpl_table *varpar)
- cpl_error_code sm_comp_convertlinetab (smspec *spec, const cpl_table *linedat)
- cpl_error_code sm_comp_convertlinetabo (smspec *spec, const cpl_table *linedat)
- cpl_error_code sm_comp_airglowcont (smspec *spec, const smparmodel modelpar, const smspec *molabstrans, const smspec *rscattrans, const smspec *mscattrans)

6.5.1 Detailed Description

Routines for the ETC sky model

Author

Stefan Noll & ESO In-Kind Team Innsbruck

Since

02 Oct 2009

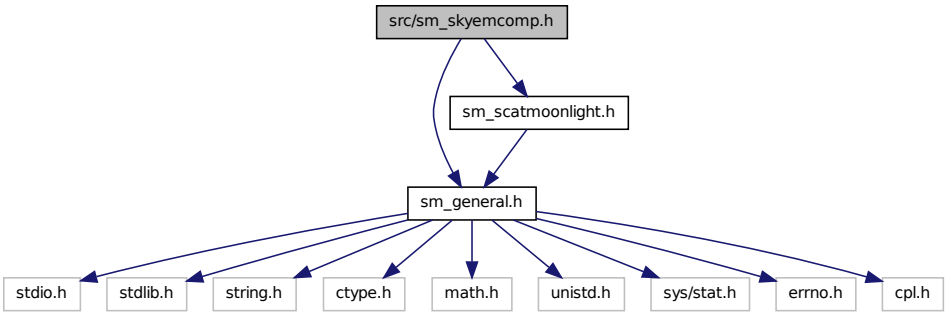
Date

31 Oct 2013

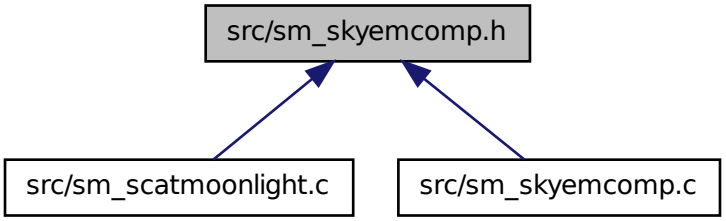
6.6 src/sm_skyemcomp.h File Reference

```
#include <sm_general.h>
#include <sm_scatmoonlight.h>
```

Include dependency graph for sm_skyemcomp.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct _smparmodel_

Defines

- #define SM_SKYEMCOMP_H

- #define SM_FILENAMELIST "sm_filenames.dat"
- #define SM_RRSTEP 1000
- #define SM_LAM_UNIT 1e-6
- #define SM_RADMINLAM 1.3 * 1e-6 / SM_LAM_UNIT
- #define SM_ERAD 6371.
- #define SM_SIGMAX 4.
- #define SM_NSIGBIN 20

Typedefs

- typedef struct _smparmodel_ smparmodel

Functions

- cpl_error_code sm_etc_calcmodel (cpl_table *skytable, const cpl_parameterlist *params)
- cpl_error_code sm_etc_getparams (smparmodel *modelpar, const cpl_parameterlist *params)
- const cpl_parameter * sm_etc_parameterlist_find_const (const cpl_parameterlist *self, const char *name, cpl_type type)
- cpl_error_code sm_etc_splitstring (cpl_array *outval, char *instring)
- cpl_error_code sm_etc_readfilenames (smparmodel *modelpar, const cpl_parameterlist *params)
- cpl_error_code sm_etc_writetable (cpl_table *skytable, const smspec *radiance, const smspec *transmission)
- cpl_error_code sm_comp_skyemcomp (smspec *radiance, smspec *transmission, const smparmodel modelpar)
- cpl_error_code sm_comp_extrapoltrans (smspec *spec, const smparmodel modelpar, const double lim[2])
- double sm_comp_alttoairmass (const double alt)
- cpl_error_code sm_comp_getmolspec (smspec *spec, const cpl_parameterlist *libfilepar, const double lim[2])
- cpl_error_code sm_comp_readlibstruct (cpl_parameterlist *libfilepar, const smparmodel modelpar, const char *spectype)
- double sm_comp_vactoair_single (const double lam)
- cpl_error_code sm_comp_vactoair_spec (smspec *spec)
- double sm_comp_airtovac_single (const double lam)
- cpl_error_code sm_comp_airtovac_spec (smspec *spec)
- double sm_comp_calcrayleighscat1 (const double lam)
- cpl_error_code sm_comp_calcrayleighscat (smspec *rscattrans)
- double sm_comp_calcmiescat1 (const double lam)
- cpl_error_code sm_comp_calcmiescat (smspec *mscattrans)
- cpl_error_code sm_comp_scaletranscurv (smspec *trans, const smparmodel modelpar)
- cpl_error_code sm_comp_lunskybright (smspec *spec, const smparmodel modelpar, const smspec *solspec, const smspec *molabstrans, const smspec *rscattrans, const smspec *mscattrans)

- cpl_error_code sm_comp_lunskybright_v (smspec *spec, const smparmodel modelpar, const smspec *solspec, const smspec *molabstrans, const smspec *rscattrans, const smspec *mscattrans)
- cpl_error_code sm_comp_getmoonalbedo (smspec *albedo, const smparmodel modelpar)
- cpl_error_code sm_comp_scatstarlight (smspec *spec, const smparmodel modelpar, const smspec *molabstrans)
- cpl_error_code sm_comp_zodskybright (smspec *spec, const smparmodel modelpar, const smspec *solspec, const smspec *molabstrans, const smspec *rscattrans, const smspec *mscattrans)
- cpl_error_code sm_comp_telem (smspec *spec, const smparmodel modelpar)
- cpl_error_code sm_comp_extrapolrad (smspec *spec, const smspec *trans, const smparmodel modelpar, const double lim[2])
- cpl_error_code sm_comp_getlinespec (smspec *spec, const smparmodel modelpar, const smspec *rscattrans, const smspec *mscattrans)
- cpl_error_code sm_comp_readvarpar (cpl_table *varpar, const smparmodel modelpar)
- cpl_error_code sm_comp_scalelinetab (cpl_table *linedat, const smspec *linetab, const cpl_table *varpar)
- cpl_error_code sm_comp_convertlinetab (smspec *spec, const cpl_table *linedat)
- cpl_error_code sm_comp_convertlinetabo (smspec *spec, const cpl_table *linedat)
- cpl_error_code sm_comp_airglowcont (smspec *spec, const smparmodel modelpar, const smspec *molabstrans, const smspec *rscattrans, const smspec *mscattrans)

6.6.1 Detailed Description

Header for routines concerning the ETC sky model

Author

Stefan Noll & ESO In-Kind Team Innsbruck

Since

02 Oct 2009

Date

31 Oct 2013