

RTTOV v14 Users' Guide

James Hocking
Met Office, Exeter, UK

This documentation was developed within the context of the EUMETSAT Satellite Application Facility on Numerical Weather Prediction (NWP SAF), under the Cooperation Agreement dated 7 September 2021, between EUMETSAT and the Met Office, UK, by one or more partners within the NWP SAF. The partners in the NWP SAF are the Met Office, ECMWF, DWD and Météo France.

Copyright 2025, EUMETSAT, All Rights Reserved.

Change record			
Version	Date	Author / changed by	Remarks
0.1	29/02/24	JH	First draft for v14 beta.
1.0	29/08/24	JH	Updates for PC-RTTOV, late developments, beta comments
1.0.1	25/10/24	JH	Updates after Met Office review
1.0.2	28/11/24	JH	Updates after DRR
1.0.3	29/01/25	JH	Very minor updates just before release

TABLE OF CONTENTS

1	INTRODUCTION AND SCOPE	6
2	OVERVIEW OF RTTOV V14.....	6
3	SUMMARY OF RTTOV CONCEPTS	12
4	CHANGES FROM PREVIOUS VERSIONS	16
4.1	<i>CHANGES FROM RTTOV v13.2</i>	16
5	FORTRAN-2008 UNIX/LINUX INSTALLATION AND TESTING	18
5.1	<i>SYSTEM REQUIREMENTS</i>	18
5.2	<i>PACKAGE CONTENTS AND COEFFICIENT FILES</i>	19
5.3	<i>COMPILING THE CODE</i>	21
5.4	<i>TESTING YOUR INSTALLATION</i>	24
5.4.1	<i>VERIFYING THE RTTOV BUILD.....</i>	25
5.4.2	<i>RUNNING EXAMPLES OF CODE CALLING RTTOV v14</i>	26
6	RTTOV TYPES, SUBROUTINES, LIBRARIES, AND EXECUTABLES	27
6.1	<i>RTTOV DERIVED TYPES</i>	27
6.2	<i>RTTOV LIBRARIES AND SUBROUTINES</i>	27
6.3	<i>LINKING YOUR CODE AGAINST RTTOV</i>	29
6.4	<i>RTTOV EXECUTABLES</i>	29
6.5	<i>DOXYGEN DOCUMENTATION</i>	30
7	RUNNING RTTOV V14 FOR YOUR APPLICATIONS.....	31
7.1	<i>SET RTTOV OPTIONS</i>	32
7.2	<i>READ COEFFICIENT/OPTICAL PROPERTY FILES</i>	33
7.3	<i>ALLOCATE RTTOV DATA STRUCTURES</i>	33
7.3.1	<i>CORE RTTOV STRUCTURES</i>	33
7.3.2	<i>TRAJECTORY STRUCTURES.....</i>	35
7.4	<i>SPECIFYING INPUT PROFILES</i>	35
7.4.1	<i>PRESSURE LEVELS AND INTERPOLATION</i>	38
7.4.2	<i>TRACE GASES.....</i>	40
7.4.3	<i>GAS OPTICAL DEPTH REGRESSION LIMITS</i>	41
7.4.4	<i>CLOUD LIQUID WATER (CLW) FOR MW SIMULATIONS</i>	42
7.5	<i>SPECIFYING THE CHANNELS TO SIMULATE</i>	42
7.6	<i>SPECIFYING SURFACE EMISSIVITY AND REFLECTANCE</i>	43
7.7	<i>CHECKING RTTOV INPUTS</i>	43
7.8	<i>OUTPUTS FROM RTTOV v14</i>	44
7.8.1	<i>TRANSMITTANCE OUTPUTS.....</i>	45
7.8.2	<i>RADIANCE OUTPUTS.....</i>	45
7.8.3	<i>QUALITY FLAGS.....</i>	46
7.8.4	<i>ADDITIONAL OUTPUTS</i>	46
7.9	<i>CALLING THE TANGENT LINEAR (TL), ADJOINT (AD) AND JACOBIAN (K) MODELS</i>	47
7.10	<i>MULTI-THREADED EXECUTION</i>	49
7.11	<i>RTTOV PERFORMANCE</i>	49
7.12	<i>SUMMARY OF STEPS FOR RUNNING RTTOV v14</i>	51
8	DETAILS OF SPECIFIC RTTOV CAPABILITIES.....	52
8.1	<i>SOLAR RADIATION</i>	52
8.1.1	<i>ENABLING SOLAR RADIATION.....</i>	52

8.1.2	<i>TOP-OF-ATMOSPHERE SOLAR IRRADIANCE</i>	52
8.1.3	<i>CLEAR-SKY RAYLEIGH SCATTERING</i>	53
8.1.4	<i>OUTPUT TOP-OF-ATMOSPHERE REFLECTANCES</i>	53
8.1.5	<i>UV SIMULATIONS</i>	54
8.2	<i>SIMPLE CLOUD (NON-SCATTERING)</i>	55
8.3	<i>SURFACE EMISSIVITY AND REFLECTANCE</i>	56
8.3.1	<i>RTTOV CALCULATION OF SURFACE EMISSIVITY</i>	57
8.3.2	<i>RTTOV CALCULATION OF SURFACE BRDF</i>	59
8.3.3	<i>RTTOV CALCULATION OF SURFACE DIFFUSE REFLECTANCE</i>	59
8.3.4	<i>SUMMARY OF EMISSIVITY AND REFLECTANCE CALCULATIONS</i>	60
8.3.5	<i>CALLING RTTOV EMISSIVITY/BRDF MODELS OUTSIDE RTTOV</i>	60
8.3.6	<i>EMISSIVITY ATLASES</i>	61
8.3.7	<i>BRDF ATLAS</i>	63
8.3.8	<i>SPECULAR VS LAMBERTIAN SURFACES</i>	64
8.3.9	<i>PER-CHANNEL EFFECTIVE SKIN TEMPERATURE</i>	65
8.3.10	<i>HETEROGENOUS SURFACES</i>	65
8.3.11	<i>TL/AD/K FOR EMISSIVITY AND REFLECTANCE</i>	66
8.4	<i>SCATTERING SIMULATIONS</i>	68
8.4.1	<i>AEROSOLS AND HYDROMETEORS</i>	68
8.4.2	<i>SCATTERING SOLVERS</i>	68
8.4.3	<i>CLOUD OVERLAP</i>	72
8.4.4	<i>RADAR SIMULATOR</i>	74
8.4.5	<i>RAYLEIGH MULTIPLE SCATTERING</i>	75
8.4.6	<i>TREATMENT OF POLARISED SCATTERING FOR MW SENSORS</i>	75
8.4.7	<i>PREDEFINED HYDROMETEOR OPTICAL PROPERTIES</i>	76
8.4.8	<i>GENERATING CUSTOM HYDROTABLE FILES</i>	80
8.4.9	<i>PREDEFINED AEROSOL OPTICAL PROPERTIES</i>	80
8.4.10	<i>GENERATING CUSTOM AEROSOL OPTICAL PROPERTY FILES</i>	82
8.4.11	<i>EXPLICIT OPTICAL PROPERTIES</i>	82
8.4.12	<i>SUPPORT FOR DYNAMIC EMISSIVITY RETRIEVALS</i>	84
8.5	<i>INCLUSION OF NON-LOCAL THERMODYNAMIC EQUILIBRIUM EFFECTS</i>	85
8.6	<i>PC-RTTOV FOR HYPERSPECTRAL IR SOUNDERS</i>	86
8.7	<i>SPECIAL GAS OPTICAL DEPTH COEFFICIENT FILES</i>	89
8.7.1	<i>SIMULATION OF ZEEMAN EFFECT (SSMI/S, AMSU-A, ATMS)</i>	89
8.7.2	<i>SIMULATION OF SSU RADIANCES</i>	90
8.7.3	<i>SIMULATION OF NIMBUS PMR RADIANCES</i>	90
8.7.4	<i>SIMULATION OF MTG-LI</i>	90
9	<i>LIMITATIONS OF RTTOV V14</i>	91
10	<i>REPORTING AND KNOWN BUGS FOR RTTOV V14</i>	91
11	<i>FREQUENTLY ASKED QUESTIONS</i>	91
12	<i>GLOSSARY</i>	91
13	<i>REFERENCES</i>	93
14	<i>ANNEXES</i>	97
ANNEX A - COEFFICIENT INFORMATION AND CONVERSION TOOLS		97
1.	<i>RTTOV_CONV_COEF.EXE</i>	97
2.	<i>RTTOV_COEF_INFO.EXE</i>	98

ANNEX B – RTTOV_ERRORHANDLING INTERFACE	99
ANNEX C – COEFFICIENT READING AND DEALLOCATION SUBROUTINES.....	100
1. RTTOV_READ_COEFS INTERFACE	100
2. RTTOV_DEALLOC_COEFS INTERFACE	102
ANNEX D – (DE)ALLOCATION AND INITIALISATION SUBROUTINES.....	103
1. RTTOV_ALLOC_DIRECT INTERFACE	103
2. RTTOV_ALLOC_TL INTERFACE	105
3. RTTOV_ALLOC_AD INTERFACE	106
4. RTTOV_ALLOC_K INTERFACE	107
5. RTTOV_ALLOC_PROFILES INTERFACE	108
6. RTTOV_INIT_PROFILES INTERFACE	108
7. RTTOV_ALLOC_EMIS_REFL INTERFACE	109
8. RTTOV_INIT_EMIS_REFL INTERFACE	109
9. RTTOV_ALLOC_TRANSMISSION INTERFACE	110
10. RTTOV_INIT_TRANSMISSION INTERFACE	110
11. RTTOV_ALLOC_RADIANCE INTERFACE	111
12. RTTOV_INIT_RADIANCE INTERFACE	111
13. RTTOV_ALLOC_OPT_PARAM INTERFACE	112
14. RTTOV_INIT_OPT_PARAM INTERFACE	112
15. RTTOV_INIT_OPT_PARAM_SOLAR INTERFACE	112
16. RTTOV_ALLOC_REFLECTIVITY INTERFACE	113
17. RTTOV_INIT_REFLECTIVITY INTERFACE	113
18. RTTOV_ALLOC_EMIS_RET_TERMS INTERFACE	113
19. RTTOV_INIT_EMIS_RET_TERMS INTERFACE	113
20. RTTOV_ALLOC_DIAGNOSTIC_OUTPUT INTERFACE	114
21. RTTOV_INIT_DIAGNOSTIC_OUTPUT INTERFACE	114
22. RTTOV_ALLOC_PCCOMP INTERFACE	114
23. RTTOV_INIT_PCCOMP INTERFACE	114
24. RTTOV_ALLOC_TRAJ_ALL INTERFACE	115
ANNEX E – OPTICAL PROPERTY CALCULATION SUBROUTINES.....	116
1. RTTOV_BPR_INIT INTERFACE	116
2. RTTOV_BPR_CALC INTERFACE	116
3. RTTOV_BPR_DEALLOC INTERFACE	116
4. RTTOV_ASYM_CALC INTERFACE	117
5. RTTOV_LCOEF_CALC INTERFACE	117
ANNEX F – SURFACE EMISSIVITY SUBROUTINES	118
1. RTTOV_SETUP_EMIS_ATLAS INTERFACE	118
2. RTTOV_GET_EMIS INTERFACE	119
3. RTTOV_DEALLOCATE_EMIS_ATLAS INTERFACE	122
4. RTTOV_GET_SEA_EMIS INTERFACE	123
ANNEX G – SURFACE REFLECTANCE SUBROUTINES	124
1. RTTOV_SETUP_BRDF_ATLAS INTERFACE	124
2. RTTOV_GET_BRDF INTERFACE	124
3. RTTOV_DEALLOCATE_BRDF_ATLAS INTERFACE	125
4. RTTOV_GET_SEA_BRDF INTERFACE	126
ANNEX H – CORE RTTOV SUBROUTINES	127
1. RTTOV_DIRECT INTERFACE	127

2. RTTOV_K INTERFACE	128
3. RTTOV_TL INTERFACE	130
4. RTTOV_AD INTERFACE	132

ANNEX I – UTILITY ROUTINES AND TOOLS 134

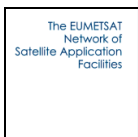
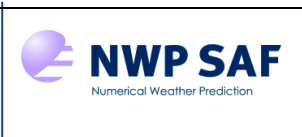
1. RTTOV_USER_CHECK_OPTIONS INTERFACE	134
2. RTTOV_USER_CHECK_PROFILE INTERFACE	134
3. RTTOV_USER_CHECK_EMIS_REFL INTERFACE	135
4. RTTOV_USER_CHECK_OPT_PARAM INTERFACE	135
5. RTTOV_PRINT_OPTS INTERFACE	136
6. RTTOV_PRINT_INFO INTERFACE	136
7. RTTOV_PRINT_PROFILE INTERFACE	136
8. RTTOV_PRINT_RADIANCE_QUALITY INTERFACE	136
9. RTTOV_WMO2RTTOV_SAT_ID INTERFACE	137
10. RTTOV_CALC_GEO_SAT_ANGLES INTERFACE	137
11. RTTOV_CALC_SOLAR_ANGLES INTERFACE	137
12. RTTOV_SCALE_REF_GAS_PROF INTERFACE	138
13. RTTOV_AER_CLIM_PROF INTERFACE	139
14. CREATE_AER_CLIM_PROF.EXE	139
15. RTTOV_EMISSIVITY_RETRIEVAL INTERFACE	140
16. RTTOV_GET_PC_PREDICTINDEX INTERFACE	140
17. RTTOV_OBS_TO_PC.EXE	141
18. RTTOV_ZUTILITY_MOD MODULE	141
19. SCATTERING OPTICAL PROPERTY GENERATION TOOLS	142

ANNEX J – RTTOV DERIVED TYPES 143

1. OPTIONS STRUCTURE	143
2. CHANPROF STRUCTURE	145
3. PROFILE STRUCTURE	146
4. EMISSIVITY/REFLECTANCE STRUCTURE	148
5. EXPLICIT OPTICAL PROPERTY STRUCTURE	148
6. TRANSMISSION STRUCTURE	149
7. RADIANCE STRUCTURE	149
8. SECONDARY RADIANCE STRUCTURE	150
9. RADAR REFLECTIVITY STRUCTURE	151
10. EMISSIVITY RETRIEVAL TERMS STRUCTURE	151
11. DIAGNOSTIC OUTPUT STRUCTURE	151
12. PC AND RECONSTRUCTED RADIANCE STRUCTURE	151

ANNEX K – RTTOV CONSTANTS 152

1. CONSTANTS FOR OPTIONS	152
2. CONSTANTS FOR INPUT PROFILE	153
3. CONSTANTS FOR OPTICAL PROPERTIES	154
4. CONSTANTS FOR QUALITY FLAGS	155
5. ADDITIONAL USEFUL CONSTANTS	155

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

1 Introduction and Scope

This document is structured as follows. Section 2 gives a broad overview of the RTTOV v14 fast radiative transfer model. **Section 3** provides a brief introduction to the key concepts involved in running RTTOV **and it is recommended to all users**. Section 4 provides an overview of changes since RTTOV v13. A separate document **docs/rttov14_user_interface_changes.pdf** gives detailed information to aid users upgrading their RTTOV implementation to v14 and is **strongly recommended** to existing users. Section 5 gives the instructions for compiling RTTOV, verifying the build and, more generally, for running the test suite. Section 6 provides a reference for the RTTOV libraries, executables, subroutines, and derived types. Section 7 provides a detailed step-by-step guide for implementing RTTOV in your own application. Section 8 gives details of various additional simulation capabilities of RTTOV: you should consult the parts of this section relevant to your application. Section 9 lists the limitations of RTTOV v14. The procedure for reporting bugs or learning about known bugs is given in section 10. Finally, a frequently asked questions (FAQ) section and glossary are provided in sections 11 and 12. The Annexes give full details of subroutine interfaces, usage of executables provided with the package, and descriptions of derived types and constants that are required in your application. This document relates to version 14 of the RTTOV code and all its sub-versions (14.x). The document is only updated with new minor releases of RTTOV v14: the document version is given in the header. To obtain a copy of the RTTOV v14 code go to <https://nwp-saf.eumetsat.int/site/register/> and register on the website. Once you are registered you can log-in and click on the “Software Downloads” link to subscribe to RTTOV v14: once you agree to the licence you will be able to download the code immediately.

RTTOV v14 is an evolution of RTTOV v13, adding and upgrading many features as documented here. RTTOV v14 represents a significant update of the RTTOV code and user interface. The RTTOV v13 code is still available but cannot be guaranteed to be upgraded for new instruments and capability. RTTOV v12 and earlier are no longer supported.

The RTTOV v14 Science and Validation Report describes or gives links to the scientific basis of the model and describes in more detail the new scientific changes made. The RTTOV v14 Test Plan and Test Log document the tests carried out on the new code before delivery. The most up to date versions of these reports, including this user guide, can be viewed at the NWP SAF web site: <https://nwp-saf.eumetsat.int/site/software/rttov/> in pdf format on the RTTOV pages. There are also Performance Test Logs which document the run times of each RTTOV v14.x release compared to those for the previous version on a few platforms.

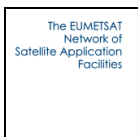
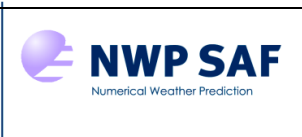
2 Overview of RTTOV v14

This section gives a brief overview of the RTTOV v14 model. More details can be found in the references given in this section and elsewhere in this user guide. RTTOV v14 is a development of the fast radiative transfer model for TOVS, RTTOV, originally developed at ECMWF in the early 90's (Eyre, 1991) for TOVS. Subsequently the original code has gone through several developments (e.g., Saunders *et al.*, 1999; Matricardi *et al.*, 2001), more recently within the EUMETSAT NWP Satellite Application Facility (SAF), e.g., Saunders *et al.* (2018), of which RTTOV v14 is the latest version. The model allows rapid simulations of radiances for satellite ultraviolet, visible, infrared or microwave downward (i.e., not limb) scanning radiometers given an atmospheric profile of temperature, variable gas concentrations, aerosol, hydrometeor, and surface properties, referred to as the state vector. The only mandatory variable gas for RTTOV is water vapour. Optionally ozone, carbon dioxide, nitrous oxide, methane, carbon monoxide and sulphur dioxide can be variable with all other constituents assumed to be constant. The state vector for RTTOV is given in Table 7.2 and Annex J.

The core of RTTOV simulates clear-sky radiances. For channels with a significant thermally emitted contribution, the top of the atmosphere upwelling clear-sky radiance, $L_{Ctr}(\nu, \theta)$, at a frequency ν and viewing angle θ can be written as:

$$L_{Ctr}(\nu, \theta) = \tau_s(\nu, \theta)\varepsilon_s(\nu, \theta)B(\nu, T_s) + \int_{\tau_s}^1 B(\nu, T) d\tau + (1 - \varepsilon_s(\nu, \theta))\tau_s^2(\nu, \theta) \int_{\tau_s}^1 \frac{B(\nu, T)}{\tau^2} d\tau \quad (2.1)$$

where τ_s is the surface to space transmittance, ε_s is the surface emissivity, T_s is the surface skin temperature, and $B(\nu, T)$ is the Planck function for a frequency ν and temperature T . The transmittances, τ , are computed by means of a linear regression in gas absorption optical depth based on variables (“predictors”) calculated from the input profile vector. A number of different predictor sets are supported, named after the RTTOV versions in which they were introduced. More information on these is given in section 3.

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

The spectral range of RTTOV v14 in the ultraviolet/visible/infrared is 0.24-100 μm in wavelength (100 – 42000 cm^{-1} in wavenumber), governed by the range of the LBLRTM (Line-by-Line Radiative Transfer Model) datasets on which the gas absorption optical depth regression coefficients are based. In the microwave the frequency range is from 1 – 800 GHz which is covered using the Liebe-89 MPM line-by-line model (Liebe, 1989). The full list of currently supported platforms and sensors is given in Tables 2.1 and 2.2. In most cases, coefficients can be made for new sensors as soon as spectral response data are available and these can be used with RTTOV immediately without any code updates.

In addition to the clear-air simulations described above (and covered in detail in this user guide) there are options to include solar radiation (section 8.1), a simple UV/visible/IR cloud scheme based on a single cloud top pressure and cloud fraction (section 8.2), and treatment of hydrometeor and aerosol scattering (section 8.4). RTTOV provides many options related to the treatment of surface emissivity and reflectance (section 8.3). RTTOV also provides a Principal Components-based model for simulating hyperspectral IR sounders (section 8.6). Finally, a number of other capabilities are supported including a correction for non-local thermodynamic equilibrium effects (section 8.5), and treatment of the Zeeman effect in high-peaking microwave sensor channels (section 8.7.1).

An important feature of the RTTOV model is that it not only computes the forward (or direct) radiative transfer calculation but also the gradient of the radiances with respect to the state vector variables at the location in state space specified by the input state vector values. Given a state vector, \mathbf{x} , a radiance vector, \mathbf{y} , is computed:

$$\mathbf{y} = H(\mathbf{x}) \quad (2.2)$$

where H is the radiative transfer model (also referred to as the observation operator). The Jacobian matrix \mathbf{H} gives the change in radiance $\delta\mathbf{y}$ for a change in any element of the state vector $\delta\mathbf{x}$ assuming a linear relationship about a given atmospheric state \mathbf{x}_0 :

$$\delta\mathbf{y} = \mathbf{H}(\mathbf{x}_0)\delta\mathbf{x} \quad (2.3)$$

The elements of \mathbf{H} contain the partial derivatives $\partial y_i / \partial x_j$ where the subscript i refers to channel number and j to position in state vector. The Jacobian gives the top-of-atmosphere radiance change for each channel given unit perturbations at each respective level of the profile vectors and in each of the surface/cloud parameters. It shows clearly, for a given profile, which layers in the atmosphere are most sensitive to changes in temperature and variable gas concentrations for each channel. `rttov_k` (and its associated subroutines ending in “_k”) compute the $\mathbf{H}(\mathbf{x}_0)$ matrix for each input profile.

It is not always necessary to store and access the full Jacobian matrix \mathbf{H} and so the RTTOV package has routines to only output the *tangent linear* values $\delta\mathbf{y}$, the change in top of atmosphere radiances \mathbf{y}_n for each channel n , for a given change in atmospheric profile, $\delta\mathbf{x}$, about an initial atmospheric state \mathbf{x}_0 .

$$\delta y(x_0) = \left[\delta x \frac{\delta y_1}{\delta x}, \delta x \frac{\delta y_2}{\delta x}, \delta x \frac{\delta y_3}{\delta x}, \dots, \delta x \frac{\delta y_{nchan}}{\delta x} \right] \quad (2.4)$$

The tangent linear routines all have “_tl” as an ending. Conversely the adjoint routines (ending in “_ad”) compute the change in up to nel elements (nel =number of elements) of the state vector (e.g., T, q, ozone, surface variables etc.) $\delta\mathbf{x}$ for an assumed atmospheric state, \mathbf{x}_0 , given a change in the radiances, $\delta\mathbf{y}$.

$$\delta x(x_0) = \left[\delta y \frac{\delta x_1}{\delta y}, \delta y \frac{\delta x_2}{\delta y}, \delta y \frac{\delta x_3}{\delta y}, \dots, \delta y \frac{\delta x_{nel}}{\delta y} \right] \quad (2.5)$$

These routines are normally used as part of the variational assimilation of radiances. Information about running the RTTOV TL (tangent linear), AD (adjoint), and K (Jacobian) models is given in section 7.9 and some general information on TL/AD and K codes is available at: https://nwp-saf.eumetsat.int/site/software/rttov/documentation/rttov-mathematical-overview/#TLADK_coding. If you only want to compute radiances with the forward model the TL/AD/K routines are not required.

You can experiment with RTTOV simulations and 1D-Var retrievals directly online with a Web Based Satellite Sounding Training Application <https://sounding.trainhub.eumetsat.int/>. A selection of satellite instruments and input profiles from the NWP SAF profile database are used for the simulation and the retrieval tools. You can modify the temperature, humidity, observation and background errors, noise, satellite zenith angle, or surface emissivity, and see how it impacts simulations and retrievals. The tool can display the results of the K-Model (Jacobian) as well as the result of the direct model (brightness temperatures and radiances), and the differences between the different runs.

Platform	RTTOV ID	Sat ID range	Platform	RTTOV ID	Sat ID range
NOAA*	1	1 - 21	DSCOVR	42	1
DMSP	2	1 - 4, 7 - 19	CLARREO	43	1
Meteosat	3	1 - 7	TICFIRE	44	1
GOES	4	4 - 19	Reserved	45	-
GMS	5	1 - 5	ISS	46	1
FY2	6	2 - 5, 7 - 8	HJ1	47	2
TRMM	7	1	GEOKOMPSAT2	48	1
ERS	8	1 - 2	GCOM-C	49	1
EOS	9	1 - 2	SMOS	50	1
METOP	10	1 - 3	ORS	51	6
ENVISAT	11	1	FY4	52	1 - 2
MSG	12	1 - 4	TROPICS	53	0, 3, 5 - 7
FY1	13	3 - 4	GF5	54	1
ADEOS	14	2	HY2	55	1
MTSAT	15	1 - 2	CloudSat	56	1
CORIOLIS	16	1	CloudCore	57	1
JPSS (SNPP)	17	0	FORUM	58	1
GIFTS	18	1	TEMPEST-D	59	0, 1
Sentinel3	19	1 - 2	Jason-CS	60	1
MeghaTropique	20	1	Electro-L	61	2
Kalpana	21	1	OMS	62	1
Meteor	22	25	Sentinel2	63	1 - 2
FY3	23	1 - 7	CloudCtrl	64	1
COMS	24	1	Trishna	65	1
Meteor-M	25	1 - 2	SBG	66	1
Reserved	26	-	SAPHIR-HY	67	1
CALIPSO	27	1	AWS	68	1
Reserved	28	-	CMIMMW	69	1
GCOM-W	29	1	EarthCARE	70	1
Nimbus	30	1 - 7	Sentinel5P	71	1
Himawari	31	8 - 9	Polsir	72	1
MTG	32	1 - 2	Oceansat	73	3
Saral	33	1	Microsat2B	74	0
Metop-SG	34	1 - 2	GOSAT-GW	75	1
Landsat	35	4 - 5, 7 - 9	WSF-M	76	1
Jason	36	2	CO2M	77	1
GPM	37	1	PREFIRE	78	1
INSAT-1	38	1 - 4	CIMR	79	1
INSAT-2	39	1 - 5	FY-4M	80	1
INSAT-3	40	4 - 6	Jiheng	81	1
Reserved	41	-	Microsat2C	82	1

*"NOAA-5" is TIROS-N

Table 2.1: Platforms supported by RTTOV as of October 2024. Shading indicates platforms for which coefficients are not yet available but can be requested.

Sensor	RTTOV ID	Sensor Chans => RTTOV Chans
HIRS	0	1-16 => 1-16 (Nimbus-6) / 1-19 => 1-19
MSU	1	1-4 => 1-4
SSU	2	1-3 => 1-3
AMSU-A	3	1-15 => 1-15
AMSU-B	4	1-5 => 1-5
AVHRR	5	1-5 => 1-5 (NOAA-5-14) / 1-6 => 1-6
SSMI	6	1-7 => 1-7
VTPR1	7	1-32 => 1-32 (1-8 x 4) 4 sets of SRFs for channels 1-8 on the 4 platforms. It is not known which SRF set belongs to which platform.
Spare	8	-
TMI	9	1-9 => 1-9
SSMIS	10	1-24 => 1-24 (Zeeman <i>rtcoef</i> file recommended for channels 19-22)
AIRS	11	1-2378 => 1-2378
HSB	12	1-4 => 1-4
MODIS	13	1-36 => 1-36
ATSR	14	1-4 => 1-4 (ERS-1 has 3 additional 12um channels 5,6,7, see headers) 1-7 => 1-7 (ATSR-2/AATSR)
MHS	15	1-5 => 1-5
IASI	16	1-8461 => 1-8461
AMSR-E	17	1-12 => 1-12
GMS imager	18	1-2 => 1-2 (GMS-1/2/3/4) 1-4 => 1-4 (GMS-5)
ATMS	19	1-22 => 1-22
MVIRI MVIRI-VIS	20	1-2 => 1-2 (IR channels) VIS (0.7um channel): variable by platform, multiple channels representing time-evolving SRFs, see coefficient download page
SEVIRI	21	1-12 => 1-12
GOES imager	22	1-4 => 2-5 (GOES 8-12) 1-5 => 1-5 (GOES 13-15)
GOES sounder	23	1-12 => 1-12 / 1-18 => 1-18
MTSAT imager	24	1-5 => 1-5
FY2-3/4 VISSR	25	1-5 => 1-5
FY1 MVISR	26	1-3 => 1-3
CrIS (NSR)	27	1-1305 => 1-1305
CrIS-FSR	28	1-2211 => 1-2211
VIIRS	29	1-22 => 1-22
WINDSAT	30	1-16 => 1-16
GIFTS	31	-
SSM-T1	32	1-7 => 1-7
SSM-T2	33	1-5 => 1-5
SAPHIR	34	1-6 => 1-6
MADRAS	35	1-9 => 1-9
Reserved	36	-
VHRR	37	1-3 => 1-3
INSAT imager	38	1-6 => 1-6
INSAT sounder	39	1-19 => 1-19
MWTS	40	1-4 => 1-4
MWHS	41	1-5 => 1-5
IRAS	42	1-26 => 1-26
MWRI	43	1-10 => 1-10
ABI	44	1-16 => 1-16
COMS MI	45	1-5 => 1-5
MSUMR	46	1-6 => 1-6
Reserved	47	-
IIR	48	1-3 => 1-3
ESA MWR	49	1-2 => 1-2
Reserved	50-53	-
SCAMS	54	1-5 => 1-5



SMMR	55	1-10 => 1-10
AHI	56	1-16 => 1-16
MTG IRS	57	1-1953 => 1-1953
AltiKa	58	1-2 => 1-2
IASI-NG	59	1-16921 => 1-16921
Landsat TM	60	1-6 => 1-6
MTG FCI	61	1-16 => 1-16
AMSR1	62	1-16 => 1-16
AMSR2	63	1-14 => 1-14
FY2-2 VISSR	64	1-2 => 1-2
SLSTR	65	1-9 => 1-9 (F1+F2 are the same as S7+S8)
Landsat TIRS	66	1-2 => 1-2
AMR	67	1-3 => 1-3
OLI	68	1-9 => 1-9
IRIS	69	1-862 => 1-862
ICI	70	1-13 => 1-13
GMI	71	1-13 => 1-13
MWTS-2	72	1-13 => 1-13
MWHS-2	73	1-15 => 1-15
ASTER	74	(1-2,3N,3B,4-14) => 1-15
Reserved	75	-
MTVZA-GY	76	1-29 => 1-29
MetImage	77	1-20 => 1-20
MWS	78	1-24 => 1-24
MWI	79	1-26 => 1-26
EPIC	80	5-10 => 1-6 (NB not all UV channels supported yet)
MRIR	81	1-5 => 1-5
SI	82	1-579 => 1-579
Reserved	83-86	-
MERSI-1	87	20 => 1 (no SRF data received for VIS channels)
MERSI-2	88	1-25 => 1-25
ECOSTRESS	89	1-5 => 1-5
IRMSS	90	4 => 1
OLCI	91	315 channels in 21 groups (one for each channel) of 15 representing different detector SRFs (see <i>rtcoef</i> file headers)
THIR	92	1-2 => 1-2
AMI	93	1-16 => 1-16
IKFS2	94	1-2701 => 1-2701
LI	95	2 channels corresponding to SRFs at centre of each FOV and edge of each FOV (see section 8.4.7)
SGLI	96	18-19 => 1-2 (no SRF data received for VIS channels)
HIRAS (NSR)	97	1-1369 => 1-1369
GIIRS	98	1-1682 => 1-1682 / 1-1690 => 1-1690 (GIIRS-1 contains a subset of the channels in the GIIRS-2 file and the files are identical for those common channels).
AGRI	99	1-7/8,9-14 => 1-13 (FY-4A, coefs include only one 3.9um channel) 1-15 => 1-15 (FY-4B+)
PMR	100	9 channels which incorporate view angle, set the zenith angle to zero in simulations
MIRAS	101	1-2 => 1-2
COWVR	102	1-12 => 1-12
TROPICS	103	1-12 => 1-12
VIMS	104	9-12 => 1-4 (no SRF data received for VIS channels)
DPR	105	1-2 => 1-2
HY2 MWRI	106	1-9 => 1-9
CPR	107	1 => 1
SAPHIR NG	108	-
FORUM	109	1-5001 => 1-5001
TEMPEST-D	110	1-5 => 1-5
Reserved	111	-
VIRR	112	1-10 => 1-10

SIRS	113	1-8 => 1-8 (Nimbus-3) / 1-14 => 1-14 (Nimbus-4)
HIRAS-FSR	114	1-2275 => 1-2275
HRIR	115	1 => 1 (Nimbus-1/2) / 1-2 => 1-2 (Nimbus-3)
AMR-C	116	1-6 => 1-6
MSUGS	117	1-10 => 1-10
Reserved	118	-
GEMS1	119	1-16 => 1-16
Sentinel2 MSI	120	1-13 => 1-13
HYMS	121	1-1040 => 1-1040
TIR	122	1-4 => 1-4
SBG	123	1-8 => 1-8
GOME-2	124	1-3142 => 1-3142
SAPHIR-HY	125	1-600 => 1-600
AWS	126	1-19 => 1-19
CMIMMW	127	-
EarthCARE MSI	128	7-9 => 1-3 (no SRF data received for VIS channels)
TROPOMI	129	-
Sentinel5	130	-
MERSI-LL	131	2-7 => 1-6 (no SRF data received for VIS channel)
MWTS-3	132	1-17 => 1-17
HIRAS-2	133	1-3041 => 1-3041
Polsir	134	1-6 => 1-6
ESMR	135	1 => 1 (Nimbus-5) / 1-2 => 1-2 (Nimbus-6)
NEMS	136	1-5 => 1-5
EarthCARE CPR	137	1 => 1
SSH	138	1-16 => 1-16
Reserved	139	-
FY-3E/F MWHS-2	140	1-15 => 1-15
SSTM	141	1-2 => 1-2
MicroSat2B MSI	142	1-6 => 1-6
AMSR-3	143	1-21 => 1-21
WSFM MWI	144	1-17 => 1-17
Reserved	145	-
SCR	146	-
CO2M CLIM	147	1-3 => 1-3
Reserved	148	-
PREFIRE TIRS	149	-
CIMR	150	5 frequencies each at 4 pols (V, H, Stokes 3/4)
FY4 MWS	151	-
MWRI-RM	152	1-26 => 1-26
MERSI-RM	153	1-8 => 1-8
FY-3 PMR	154	1-2 => 1-2
Jiheng	155	1-18 => 1-18
MERSI-3	156	1-25 => 1-25
MWRI-2	157	1-22 => 1-22
AIRS L1C	158	1-2645 => 1-2645
EOIR	159	-

The channel order in the coefficient files can be determined using the **rttov_coef_info.exe** executable (see Annex A), or by looking at the following web page which gives the spectral response functions and passbands used by RTTOV and the linked pages show the channel order in the coefficient files:

<https://nwp-saf.eumetsat.int/site/software/rttov/download/coefficients/spectral-response-functions/>

Table 2.2: Instruments supported by RTTOV as of October 2024. Coefficients are not yet available for shaded sensors but can be requested, as can coefficients for sensors not listed above. The channel mappings correspond to recommended coefficient files for RTTOV v14 (see section 3). Older coefficient files that do not support UV/VIS/NIR channels may contain subsets of the channels given above.

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

3 Summary of RTTOV concepts

This section provides a brief overview of the steps required when running RTTOV. Existing users should read this section for information on coefficient files. New users should read this section to familiarise themselves with the key concepts related to RTTOV. You should also look through the code in `src/test/example_fwd.F90` as this provides a fully-commented template for running a basic simulation using the RTTOV forward model. The `src/test/` directory also contains examples for running other kinds of simulation. A detailed description of the steps involved in running RTTOV is given in section 7. The `rttov-quick-start.pdf` guide in the `docs/` directory guides new users through a simple clear-sky forward model simulation.

Thermal vs solar channels

RTTOV accounts for thermal emission for all channels at wavelengths above $3\mu\text{m}$ (referred to as “thermal” channels). Thermal emission is ignored at wavelengths below $3\mu\text{m}$. Solar radiation is only included in channels at wavelengths below $5\mu\text{m}$. These are referred to as “solar” channels. Channels below $3\mu\text{m}$ are solar-only channels, while those in the range $3\text{-}5\mu\text{m}$ are mixed thermal+solar channels.

Optical depth coefficient files

Calculation of gas absorption optical depths is carried out by a predictor-based regression scheme. The coefficients for the optical depth regression are instrument-specific and are stored in RTTOV coefficient files whose names begin “`rtcoef_`”. Coefficient files can be found in sub-directories within the `rtcoef_rttov14/` directory of the RTTOV installation. The RTTOV package includes coefficient files for a selection of sensors. Additional coefficient files for all supported sensors can be downloaded from the RTTOV v14 coefficients download page:

<https://nwp-saf.eumetsat.int/site/software/rttov/download/coefficients/coefficient-download/>


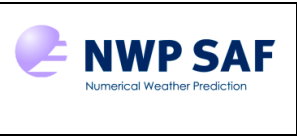
RTTOV implements different sets of gas optical depth predictors named after the RTTOV versions in which they were introduced. Coefficients based on “v13 predictors” (Hocking *et al.*, 2021) are generally recommended while coefficients also exist based on the v7 (Matricardi *et al.*, 2001), v8 (Matricardi, 2003), and v9 (RTTOV v9 Science and Validation Report) predictors. From a user perspective, it is simply a matter of supplying the desired coefficient file based on any of the above sets of predictors to RTTOV.

The optical depth parameterisation is carried out for layers defined by a fixed set of pressure levels. For non-hyperspectral sensors, coefficients are provided on a standard set of 54 levels (54L) shown in Table 3.1, while coefficients for hyperspectral sounders are provided on 101L which can be found on the RTTOV v14 coefficients download page.

Each coefficient file allows certain gases to vary in the simulations. Water vapour is always a variable gas and is a mandatory input to RTTOV. Other trace gas inputs are optional (if allowed by the coefficient file): O_3 , CO_2 , CO , NO_2 , CH_4 , SO_2 . Coefficient file names contain either “`_o3`”, “`_o3co2`”, or “`_7gas`”, indicating which variable gases are supported, where “`_7gas`” indicates all of the previously listed gases may vary. The greater the number of variable gases, the more computationally expensive the simulation.

To avoid producing an overwhelming number of different types of coefficient files and thus simplify matters for users and the development team, a reduced set of optical depth coefficients are recommended for use with RTTOV v14 compared with previous RTTOV versions:

- **MW sensors:** v13 predictors variable O_3 . Coefficients for all sensors with top hat passbands, and in addition coefficients based on measured spectral response functions where available. All coefficients on 54L.
- **UV/VIS/IR multi-spectral sensors:** v13 predictors variable O_3+CO_2 . For selected sensors also v13 predictors 7gas. All coefficients on 54L.
- **UV/VIS/IR hyperspectral sensors:** v13 predictors variable O_3+CO_2 . For selected sensors also v13 and v9 predictors 7gas. All coefficients on 101L.

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

RTTOV v14 supports all **ASCII** optical depth coefficient files that are compatible with RTTOV v13. Binary/Fortran unformatted files - see below - must be regenerated for RTTOV v14, and netCDF4 replaces HDF5 as the portable binary file format in v14. **However, any coefficient file not included in the above list is considered deprecated.** You are **strongly recommended** to use the files above. Requests for deprecated coefficient files for new sensors will be considered on a case-by-case basis, but in the first instance we will ask if you can use a recommended file instead. **When RTTOV v15 is released (currently due in early 2027), deprecated optical depth coefficient files will no longer be supported or generated.** For information about the deprecated coefficient files not listed above, please see the RTTOV v13 user guide.

The optical depth parameterisation is trained over a specific range of zenith angles up to a maximum angle of 85°. The recommended coefficient files above support the full range of zenith angles for all channels on geostationary sensors, and for all solar channels (channels below 5µm) on all sensors. For all other channels (i.e., those above 5µm on low-earth-orbit platforms), the maximum valid zenith angle is 75°. For UV/VIS/IR sensors, the recommended coefficient files above include all channels (UV/visible/near-IR/IR) for each sensor for which we have information about the spectral responses.

The latest UV/visible/IR RTTOV coefficients have been trained using the profile dataset described in the RTTOV v12 Science and Validation Report. The background profiles for trace gases (CO₂, CH₄, and N₂O in particular) are appropriate for contemporary simulations. When simulating older instruments, it is recommended to specify a more appropriate CO₂ profile which requires use of a coefficient file allowing variable CO₂.

Coefficients for microwave sensors have been updated for the RTTOV v14 release to enable variable O₃ for all sensors. Ozone is important in the sub-mm (e.g., for MetopSG ICI), and it was decided to move to variable O₃ for all sensors to ensure consistency.

The channel numbering in RTTOV coefficient files *always* begins at one. You can check the channel numbering for any coefficient file using the **rttov_coef_info.exe** tool (located in the **bin/** directory after compiling RTTOV) which prints out information about a coefficient file (see Annex A for details on its use). Table 2.2 gives the mappings from instrument channel numbers to the RTTOV channel numbers. For ASCII files you can also examine the file headers directly.

The channel order in the coefficient files generally matches the convention used for each instrument. The channel order can be seen by using the **rttov_coef_info.exe** tool noted above, examining the headers of ASCII coefficient files, or by looking at the page on the RTTOV website which gives information about the spectral response functions or pass bands for each instrument: <https://nwp-saf.eumetsat.int/site/software/rttov/download/coefficients/spectral-response-functions/>

Planck-weighting of coefficients



The accuracy of simulations for very broad thermal channels (e.g., SEVIRI channel 4 at 3.9 µm) is poor with significant biases noted (~1-2K) (see e.g., Brunel and Turner, 2003). To mitigate this, the line-by-line optical depths in the coefficient generation are weighted with the Planck function across the instrument channel and the coefficients are then computed for these Planck-weighted optical depths resulting in much reduced biases. Whether coefficients are Planck-weighted or not for a channel can be determined by examining the PLANCK_WEIGHTED section in the coefficient file (if it is not present there are no Planck-weighted channels).

Information on the RTTOV website

Plots are available on the RTTOV website showing the fit of the optical depth regression to the underlying line-by-line simulations used in training the coefficients.
<https://nwp-saf.eumetsat.int/site/software/rttov/download/coefficients/comparison-with-lbl-simulations/>

The following page provides the definitive source for RTTOV v14 coefficient files:
<https://nwp-saf.eumetsat.int/site/software/rttov/download/coefficients/coefficient-download/>

New or updated coefficient files will be made available from the RTTOV pages on the NWP SAF web site for the latest RTTOV version. If you require a coefficient file that is not available for download on the website, you can request it via the NWP SAF Helpdesk (<https://nwp-saf.eumetsat.int/site/help-desk/>). However, please refer to the note above regarding deprecated coefficient files.

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

Basics of running RTTOV

The **rttov_options** structure (described fully in Annex J) provides a list of logical flags and other settings to configure the simulation, such as whether solar radiation should be activated, and whether hydrometeor and/or aerosol scattering should be included. All aspects of RTTOV simulations can be configured in your code via the members of this structure.

The coefficient files are read in using the **rttov_read_coefs** subroutine. If Principal Component (PC), or hydrometeor/aerosol scattering simulations are being performed, the additional associated coefficient and/or optical property files for those simulations are read in the same call.

You must then allocate some derived types or structures to hold various input and output quantities. These structures are all described fully in Annex J and more details are given in section 7. The main structures are:

- **chanprof** structure – array holding a list of channel and profile indices to simulate
- **profile** structure – array holding the input atmospheric and surface variables
- **emis_refl** structure – array holding the input/output surface emissivities, reflectances, and other related data
- **transmittance** structure – to hold the output simulated transmittances
- **radiance** structure – to hold the output simulated radiances

Various other optional structures for input/output of data are also defined, some of which are required for specific types of simulation. These are described elsewhere in this user guide. In a typical application you would then loop over input profiles. For each profile, the input profile data are read into the **profile** and **emis_refl** structures. Then RTTOV is called, and the simulated radiances are written out or stored. It is possible to pass multiple profiles into RTTOV in a single call if desired. There is no fixed limit on the number of profiles you can pass into RTTOV in one go: this is limited only by the memory available. Some kinds of simulations take more memory than others. Once all simulations are complete, you should call the relevant deallocation subroutines to release allocated memory.

Additional RTTOV features

RTTOV provides a range of additional features related to the radiative transfer simulations. Some of these are briefly noted here with full descriptions given in the relevant sections of the user guide.

Surface emissivity and reflectance: RTTOV provides internal models/values for surface emissivities and reflectances, but it also allows you to input your own values instead. A selection of land surface emissivity atlases and a BRDF atlas are provided which can be used to obtain values to pass into RTTOV.

Scattering simulations: in addition to clear-sky simulations, RTTOV is capable of simulating scattering due to hydrometeors and aerosols. Several radiative transfer solvers are implemented for thermal and solar radiation separately which trade off speed against accuracy. For hydrometeor simulations, several cloud overlap treatments are implemented. Collections of pre-defined optical properties are provided with RTTOV for a variety of aerosol and hydrometeor species. Alternatively, it is possible to provide the optical property profiles as explicit inputs to RTTOV.

Principal Components model: PC-RTTOV provides a more efficient way of simulating many channels of hyperspectral IR sounders. Normal RTTOV simulations are run for a specific subset of sensor channels, and the resulting radiances are used to compute PC scores for the full spectrum allowing reconstructed radiances to be computed for any subset of channels or all sensor channels.

Referencing RTTOV

RTTOV v14.0 has the following DOI: https://doi.org/10.15770/EUM_SEC_CLM_1007

When citing RTTOV it may be described as: “RTTOV is an operational product developed and distributed by the EUMETSAT Satellite Application Facility for Numerical Weather Prediction (NWP SAF) and can be downloaded from <https://nwp-saf.eumetsat.int/>.”

And the most recent general paper on the software is:



Saunders, R., Hocking, J., Turner, E., Rayer, P., Rundle, D., Brunel, P., Vidot, J., Roquet, P., Matricardi, M., Geer, A., Bormann, N., and Lupu, C., 2018: An update on the RTTOV fast radiative transfer model (currently at version 12), *Geosci. Model Dev.*, 11, 2717-2737, <https://doi.org/10.5194/gmd-11-2717-2018>

Layer Number	Pressure** hPa	Tmax K	Tmin K	Qmax ppmv*	Qmin ppmv*	O ₃ max ppmv*	O ₃ min ppmv*	O ₃ Ref ppmv*
1	0.01	249.04	148.92	5.64	1.00	1.408	0.042	0.308
2	0.03	257.93	161.31	6.73	1.22	1.453	0.088	0.351
3	0.06	271.92	174.30	7.76	1.47	1.583	0.139	0.454
4	0.13	289.58	187.34	8.27	1.69	1.867	0.199	0.648
5	0.23	308.84	200.35	8.51	1.90	2.215	0.292	0.922
6	0.41	327.44	205.94	8.58	2.24	2.542	0.454	1.272
7	0.67	339.15	201.41	8.46	2.75	3.142	0.642	1.731
8	1.08	341.45	193.33	8.20	3.15	4.448	0.724	2.389
9	1.67	337.76	184.38	7.98	3.25	6.324	0.680	3.272
10	2.50	328.59	177.78	7.82	3.06	8.252	0.574	4.310
11	3.65	317.50	175.66	7.72	2.88	9.818	0.625	5.409
12	5.19	308.20	174.06	7.64	2.77	11.395	1.166	6.358
13	7.22	299.68	170.73	7.55	2.62	12.636	1.733	6.936
14	9.84	294.40	167.35	7.44	2.50	12.840	1.601	7.085
15	13.17	290.18	164.88	7.28	2.44	12.348	1.021	6.817
16	17.33	285.20	162.48	7.08	2.31	11.532	0.573	6.131
17	22.46	282.14	161.48	6.86	1.95	10.450	0.353	5.196
18	28.69	281.80	161.78	6.60	1.62	9.266	0.221	4.287
19	36.17	281.29	162.29	6.30	1.42	8.054	0.135	3.490
20	45.04	276.54	163.58	6.02	1.33	7.087	0.081	2.794
21	55.44	269.53	165.43	6.05	1.33	6.254	0.051	2.192
22	67.51	265.32	166.81	7.69	1.23	5.248	0.046	1.674
23	81.37	263.33	163.69	13.52	0.76	4.588	0.040	1.230
24	97.15	262.20	161.96	19.09	0.19	4.004	0.027	0.877
25	114.94	261.00	166.27	26.93	0.01	3.299	0.016	0.669
26	134.83	259.41	169.16	67.91	0.01	2.821	0.016	0.547
27	156.88	259.70	169.56	193.68	0.01	2.508	0.015	0.438
28	181.14	261.21	170.03	499.80	0.01	2.161	0.012	0.336
29	207.61	263.36	172.37	1089.12	0.01	1.727	0.012	0.241
30	236.28	267.28	175.61	1969.20	0.01	1.278	0.015	0.172
31	267.10	274.01	179.55	3428.40	0.01	0.925	0.016	0.127
32	300.00	281.55	183.38	5505.60	0.01	0.701	0.015	0.098
33	334.86	289.42	186.23	8040.00	0.65	0.589	0.016	0.079
34	371.55	296.90	189.02	10944.00	1.41	0.498	0.016	0.068
35	409.89	301.38	192.38	13956.00	1.82	0.404	0.015	0.060
36	449.67	303.53	196.43	17016.00	2.24	0.322	0.015	0.056
37	490.65	305.81	200.00	20124.00	2.64	0.265	0.015	0.053
38	532.58	309.68	202.13	23196.00	3.29	0.223	0.015	0.051
39	575.15	313.86	202.18	26088.00	3.74	0.195	0.013	0.050
40	618.07	316.90	195.79	28884.00	5.31	0.181	0.011	0.049
41	661.00	319.98	189.95	31548.00	6.44	0.149	0.009	0.048
42	703.59	324.83	189.95	34056.00	6.40	0.126	0.009	0.047
43	745.48	330.86	189.95	36504.00	7.72	0.120	0.009	0.047
44	786.33	335.12	189.95	38832.00	8.49	0.116	0.008	0.045
45	825.75	337.49	189.95	41088.00	8.06	0.114	0.008	0.044
46	863.40	340.54	189.95	43200.00	7.70	0.112	0.008	0.042
47	898.93	344.39	189.95	45240.00	7.38	0.110	0.007	0.041
48	931.99	347.73	189.95	47004.00	7.10	0.105	0.006	0.039
49	962.26	349.58	189.95	49500.00	6.86	0.101	0.006	0.036
50	989.45	350.00	189.95	50496.00	6.66	0.099	0.006	0.032
51	1013.29	350.09	189.95	48468.00	6.49	0.097	0.006	0.029
52	1033.54	350.09	189.95	47508.00	6.35	0.094	0.006	0.028
53	1050.00	350.09	189.95	47724.00	6.24	0.094	0.006	0.027

*The gas units here are ppmv with respect to **dry** air because RTTOV is trained on gas profiles in these units, and this is the context in which these limits are applied within RTTOV.

**Pressure represents the lower pressure half-level bounding each layer (see section 7.4)

Table 3.1: Pressure levels for RTTOV v14 54 level coefficients (on 53 layers) and profile limits within which the gas optical depth calculations are trained (including the stretching factor - see section 7.4.3). The default ozone profile is also given in the right-hand column. Note that some coefficients are based on a standard 101 level pressure profile which extends down to 1100 hPa.

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

4 Changes from previous versions

4.1 Changes from RTTOV v13.2

RTTOV v14.0 represents a major update from v13. This section provides an overview of the changes since v13.2. A separate document **rttov14_user_interface_changes.pdf** is provided in the **docs/** directory of the package describing changes in the user interface between v13.2 and v14.0 in detail. Users are strongly recommended to consult this when updating their code from earlier versions to v14.0. The most significant changes from a technical implementation perspective are highlighted in **bold red** in the list below.

Summary of main new features/changes in RTTOV v14

General:



- **Revise the input profile structure so that all input variables (temperature, gases, scattering inputs) are provided on the same vertical grid. This improves consistency with NWP model fields.**
- New option to output overcast BTs in addition to existing overcast radiances.
- New option to enable UV/VIS/NIR Jacobians in terms of reflectance (now the default) instead of radiance.
- New diagnostic output structure containing per-profile outputs, currently geometric heights of pressure half- and full-levels, and computed effective hydro fraction.
- Improved user-level routines for checking inputs to RTTOV before running full simulations.
- Improved notification to users via the `radiance%quality(:)` output when input values are clipped by various parameterisations within RTTOV.
- Zeeman coefficients based on v13 predictors enabled.
- **Default values of various options have changed since RTTOV v13.**

Surface emissivity/reflectance:

- **Emissivity and reflectance inputs/outputs refactored into a single new `rttov_emis_refl` data structure.**
- Enable full user control over diffuse reflectance at all wavelengths in the same way as for emissivity and BRDF.
- Diffuse reflectance is added to BRDF from sea sun-glint model for VIS channels to allow for consistency in treatment of ocean colour/sub-surface scattering.
- USGS water reflectance datasets extended to the UV (used for sea surface reflectance).
- **Enable fully flexible heterogenous surface capability: users can specify surface and near-surface properties for multiple surfaces per profile, and the properties are combined before the radiance solver is called.**
- Implement interface to CAMEL v3 IR land surface emissivity atlas datasets.
- Enable optional return of nearby land IR emissivity/BRDF within a user-specified distance if atlas has no data at the original location.
- Emissivity retrieval output structure generalised for dynamic emissivity retrievals in clear-sky cases, and for Chou-scaling solver in addition to delta-Eddington, and for all cloud overlap options.

Scattering:

- **Scattering for MW sensors now run through the main RTTOV interface in a very similar way to IR sensors from a technical perspective (the separate RTTOV-SCATT model no longer exists).**
- Delta-Eddington solver implemented within RTTOV for infrared and microwave sensors.
- Radar solver implemented in RTTOV, and passive radiances are computed alongside radar reflectivities.
- Cloud overlap options from RTTOV-SCATT available as additional options in RTTOV.
- Consistent unit conversions applied for hydrometeor concentrations in the UV/VIS/IR and MW.
- Allow separate units selection for hydrometeors and aerosols.
- UV/VIS/IR hydrometeor optical properties made fully flexible allowing any combination of particle types to be used in the same simulation (as implemented in earlier RTTOV versions for aerosols and MW hydrometeors).
- Explicit optical property inputs can be used for MW simulations as well as UV/VIS/IR.
- Explicit optical property phase functions and Legendre coefficients are no longer active variables in the TL/AD/K.
- Tang *et al* modification to Chou-scaling solver implemented to improve accuracy particularly for hydrometeor simulations with ice clouds in the far-IR.

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

- MFASIS-NN updates include improved accuracy by better treatment of water vapour and heterogenous surfaces, and code optimisation.

PC-RTTOV:

- New PC coefficients for IASI, IASI-NG, and Hamming apodised MTG-IRS supporting simulations over all surface types, with all trace gases, and optionally with either the NLTE correction, aerosol scattering, or hydrometeor scattering.
- Input profiles for PC-RTTOV simulations are no longer modified when the *apply_reg_limits* option is true. Values falling outside the limits are still flagged via the *radiance%quality(:)* output as for standard RTTOV simulations.

Wrapper:

- Python and C++ interfaces fully updated with respect to the changes in RTTOV.
- Enable return of explicit optical property Jacobians through wrapper.
- Enable user specification of radar K inputs so that the full Jacobian matrix can be computed for radar simulations.
- Add wrapper interface to the *rttov_aer_clim_prof* subroutine.
- Add a new *StoreEmisRefl* option and accessor functions to obtain surface emissivity/reflectance values used in the simulations so that the input emissivity/reflectance values are not overwritten.
- Rename *Options*, *Atlas*, and *Profiles* C++ source files with *Rttov* prefix to avoid potential name clashes/confusion with unrelated external libraries.
- Technical improvements to the C++ interface.

GUI:

- The RTTOV GUI is now a pure Python application that uses the *pyrttov* interface.
- The GUI now supports MW scattering simulations.
- The GUI no longer supports PC-RTTOV simulations.

Technical:


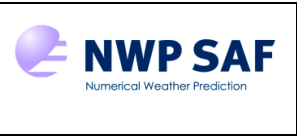
- **Numerous updates to the RTTOV Fortran interface to improve clarity and consistency. This includes significant changes to the *rttov_options* structure, updates to various module and subroutine names and subroutine interfaces, and other variable and derived type name changes.**
- HDF5 has been replaced by netCDF4 for large coefficient files and emissivity/BRDF atlas files. HDF5 is no longer an explicit dependence.
- New subroutine *rttov_wmo2rttov_sat_id* that maps WMO satellite IDs to RTTOV platform/satellite ID pairs.
- Reduce the number of memory allocations done within RTTOV to decrease run-time.
- Allow external allocation of all “trajectory” (internal state) data structures for single-threaded runs. This can improve performance in cases where many calls are made to RTTOV and the parallel interface is not used.

Capabilities removed:

- **Surface implicitly lies on bottom pressure half-level so RTTOV v14 cannot be called for profiles on fixed pressure levels with a separate surface pressure that is independent from the pressure levels.**
- FASTEM-1/2/3/4 and TESSEM2 microwave sea surface emissivity models.
- JONSWAP wave spectrum option for solar sea BRDF model.
- Solar single-scattering solver for clouds/aerosols.
- MFASIS-LUT fast visible solver for clouds based on look-up tables.
- HTFRTC Principal Components based model.
- Deprecated options removed: *grid_box_avg_cloud*, *dtau_test*, *reg_limit_extrap*, *spacetop*.

Coefficient file compatibility:

- RTTOV v14 can read all ASCII RTTOV v13-compatible optical depth coefficient files.
- Binary/Fortran unformatted optical depth coefficient files must be regenerated using RTTOV v14.
- HDF5 optical depth coefficient files cannot be read by v14: it is recommended to download the corresponding netCDF4 file from the RTTOV v14 coefficients download page.
- All other aerosol/hydrometeor optical property files and MFASIS-NN and PC-RTTOV coefficient files are mutually incompatible between v13 and v14: you must download new files from the RTTOV v14 coefficients download page.

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

5 FORTRAN-2008 UNIX/LINUX installation and testing

5.1 System requirements

RTTOV is designed for UNIX/Linux systems. The software is successfully tested on Intel and Cray systems and for a range of Fortran 2008 compilers listed below. The code makes use of some common F2003 features and a small number of common F2008 intrinsics.

The following system components are needed before running RTTOV v14:

- UNIX or Linux operating system
- Fortran2008 compiler (see below)
- Perl v5.6 or later
- make utilities
- gzip, gunzip, bzip2, bunzip2, xz, unxz
- About 1 GB of free disk space is the minimum required, although more is necessary for hyperspectral sounder coefficient, scattering optical property, PC-RTTOV, and emissivity/BRDF atlas data files.
- Memory requirements are strongly dependent on the instrument being simulated and the kind of simulations being performed: the simplest clear-sky direct model simulations require of the order of 10 MB.
- It is recommended to compile RTTOV against the netCDF library (v4.1 or higher) so that all RTTOV features are available (see section 5.3 below).
- The Python interface allows much RTTOV functionality to be called from Python scripts without writing any Fortran code. It requires that f2py is installed which is part of the Numpy package. Numpy v1.x is fully supported and Numpy v2.0 has been successfully tested (but see known compiler issues below). RTTOV is compatible with Python3: the Python wrapper should work with almost any Python3 environment with Numpy installed. Python2 is not supported. Successful compilation with f2py also requires binutils v2.26 or later.
- The RTTOV GUI (graphical user interface) is a tool intended primarily for educational purposes. It enables RTTOV simulations and visualisation of results for a single profile at a time. It is based on the Python interface but has additional Python environment requirements. To use the GUI it is recommended to set up a GUI-compatible Python environment as described in the GUI user guide in the **docs/** directory.

The RTTOV v14 package is available as a compressed tar file named **rttov140.tar.xz** which should be copied to your 'top' RTTOV directory (e.g., ~user/rttov14) and extracted. This will create various sub-directories (see below).

RTTOV v14 will not work with older versions of some compilers. The following list gives the versions of common compilers tested successfully. Earlier versions are untested and are **not supported**:

- gfortran – v4.8.5, v7.3.0, v8.1.0, v9.4.0, v11.2.0
- ifort – v16.0.1, v17.0.7, v18.0.3, v18.0.5, v19.0.0
- pgf90 – v18.7
- NAG – v6.1, v7.0
- Cray Fortran – v8.3.4

The Python wrapper has been successfully tested with:

- Python v3.10.2 / Numpy v1.26.4
- Python v3.11.6 / Numpy v1.24.2
- Python v3.12.4 / Numpy v2.0.1

Known compiler issues:

- Multithreaded compilation (enabled by passing *-j* to Make) does not work consistently with some compilers (notably gfortran and pgf90).
- RTTOV v14 has not been tested on IBM or MacOS platforms due to the lack of suitable systems within the NWP SAF.
- Gfortran v4.9.1 gives incorrect results in some cases through the parallel interface due to a compiler bug and so that version of gfortran is not recommended. In general, a recent version of gfortran should always be preferred.
- With nagfor v6.1 compiled with OpenMP support, the MW hydrotable generation code segfaults. This can be avoided either by compiling without the OpenMP compiler flag for the affected source file (see the *build/arch/nagfor-openmp* compiler flag file) or using a different compiler. Nagfor v7.0 is unaffected by this issue.

- Python wrapper compilation with gfortran and ifort has been successfully tested with Python v3.12/Numpy v2.0 in addition to earlier Python versions with Numpy v1.x. However, pgf90 with OpenMP support has not yet been successfully tested with Python v3.12/Numpy v2.0 and so earlier versions are recommended for the Portland compiler if multi-threaded simulations are required in the Python wrapper, or otherwise use the non-OpenMP-enabled compiler flags. On the NWP SAF test system it was necessary to manually add the Portland compiler libraries in the *build/arch/pgf90* file with Numpy v2.0 as meson (the build system used by f2py in Numpy v2 rather than distutils which is used in Numpy v1.x) failed to pick them up automatically.

5.2 Package contents and coefficient files

Extract the package using the command:

```
$ tar xvf rttov140.tar.xz
```

The following subdirectories are created:

brdf_data/	BRDF atlas data (data must be downloaded from web site)
build/	Scripts used in building RTTOV and files containing flags for various compilers/architectures
data/	Various ancillary data files
docs/	Documentation
emis_data/	Emissivity atlas data (data must be downloaded from web site)
gui/	RTTOV GUI Python source code
src/	RTTOV Fortran source code
rtcoef_rttov14/	RTTOV v14 coefficient files (see below)
rttov_test/	test scripts, input profiles for tests, and reference output for tests
wrapper/	RTTOV Python/C++ wrappers and example code calling RTTOV from Python and C++

Section 3 gives information about the available RTTOV optical depth coefficients. A comprehensive list of RTTOV v14 coefficients is available via the RTTOV v14 web page:


<https://nwp-saf.eumetsat.int/site/software/rttov/download/coefficients/coefficient-download/>

The *rtcoef_rttov14/* directory contains sub-directories for each kind of coefficient file. The *rttovXpredYL/* directories contain optical depth coefficients based on vX predictors on Y levels. The sub-directories are as follows:

rttov13pred54L/	UV/VIS/IR/MW sensors, all gas combinations, UV/VIS/IR solar enabled
rttov13pred101L/	hi-res UV/VIS/IR sounder files, all gas combinations, solar enabled
rttov9pred101L/	hi-res UV/VIS/IR sounder files, "7gas", solar enabled
aertable_visir/	aerosol optical property files for UV/VIS/IR sensors
hydrotable_visir/	hydrometeor optical property files for UV/VIS/IR sensors
hydrotable_mw/	hydrometeor optical property files for MW sensors
mfasis_nn/	MFASIS neural network (NN) coefficient files
pc/	Principal Components (PC-RTTOV) coefficient files

Deprecated optical depth coefficient file directories (see section 3):

rttov7pred54L/	IR/MW sensors, variable O ₃ for IR and some MW
rttov7pred101L/	hi-res IR sounder files, variable O ₃
rttov8pred51L/	SSU only, variable O ₃ +CO ₂
rttov8pred54L/	IR sensors, variable O ₃ +CO ₂
rttov9pred54L/	VIS/IR sensors, all gas combinations (primarily variable O ₃ +CO ₂), solar enabled

<p>The EUMETSAT Network of Satellite Application Facilities</p>	 <p>NWP SAF Numerical Weather Prediction</p>	<h2 style="text-align: center;">RTTOV v14 Users' Guide</h2>	<p>Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025</p>
---	--	---	--



In addition to the recommended optical depth coefficient files described in section 3 there are:

- Zeeman coefficient files for SSMI/S. New coefficients are available based on the v13 predictors. These are expected by default in the **rttov13pred54L/** directory even though they are based on different numbers of levels.
- SSU and PMR optical depth coefficient files based on v13 predictors are not yet available so you should continue to use the existing v8 predictor files. It is planned to generate new coefficients based on the v13 predictors.
- PC coefficients for IASI, IASI-NG, and MTG-IRS. The PC-RTTOV coefficients must be used with the optical depth coefficient files with which they were trained. These are the corresponding v13 predictor “7gas” coefficients files in the **rttov13pred101L/** directory.
- Aerosol optical property files for most UV/visible/IR sensors (default directory **aertable_visir/**), and hydrometeor optical properties for most sensors (default directories **hydrotable_mw/** for MW sensors, and **hydrotable_visir/** for all other sensors).
- The MFASIS-NN fast solar cloud solver (section 8.4.2) requires neural network coefficient files (default directory **mfasis_nn/**) for the given sensor in addition to the *rtcoef* and the *hydrotable* files.
- The “ARO-scaling” polarisation treatment for MW hydrometeor simulations (see section 8.4.6) requires an additional file “ScalingFactorForBulkProperties.rssp” (default directory **hydrotable_mw/**). This file is sensor independent. It is only available in ASCII format and cannot be converted to other formats (see below).

The RTTOV distribution includes v13 predictor UV/VIS/IR and MW coefficient files for a commonly used subset of sensors. Coefficient files for all supported sensors including hi-res sounder coefficient files, scattering optical property, MFASIS-NN, and PC-RTTOV files, and the emissivity and BRDF atlas datasets are available from the RTTOV web site.

For the purposes of running the example programs and scripts you should ensure coefficient files are placed in the appropriate directories. An interactive script **rttov_coef_download.sh** is available in the **rtcoef_rttov14/** directory which can be used to download any or all coefficient files into the standard locations in the coefficients directory. Note that you only need to download the coefficients required for the simulations you wish to carry out and for the sensor(s) you want to simulate. For example, there is no need to download any hi-res UV/VIS/IR sounder coefficients unless you want to run simulations for an instrument of that kind.

Most coefficient files are supplied in ASCII format. In some cases, such as hyperspectral UV/VIS/IR sounders, ASCII files are very large and so netCDF4 format is used instead for efficiency. It is possible to convert coefficient files between ASCII, netCDF, and Fortran unformatted (“binary”) formats, the latter two being more efficient (though note that the binary format is not portable between systems). It is also possible to extract a subset of channels which reduces the file size and can improve performance, particularly for hyperspectral sounders. The **rttov_conv_coef.exe** program performs these tasks and is described in Annex A. If you require an ASCII version of a netCDF coefficient file, this can be requested via the NWP SAF Helpdesk.

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
---	---	---	---

5.3 Compiling the code

An interactive shell script is available to compile RTTOV v14 which asks some questions and then runs the necessary commands to do the compilation:

```
$ cd src
$ ../build/rttov_compile.sh
```

There must be a file in the **build/arch/** directory containing the compilation flags you wish to use. There are example files for various common compilers, or you can create a new one: more details on this are given below in the section “Creating an architecture configuration file”. To make use of multi-threaded execution via the **rttov_parallel_*** routines RTTOV must be compiled with OpenMP. There are compiler flag files in **build/arch/** for compiling with OpenMP support with gfortran, pgf90, ifort and NAG.

RTTOV may be compiled immediately without requiring any external libraries. However, some features of RTTOV have external dependencies:

Reading netCDF coefficient files : requires the netCDF library.
Emissivity*/BRDF atlases : require the netCDF library.
Python interface and RTTOV GUI : require that f2py is installed.

**The TELSEM2 MW emissivity atlas may be used without any external dependencies as the atlas data files are in ASCII format, but the other atlases require the netCDF library.*

Compiling with the netCDF library is recommended. **Before compiling with netCDF you must first edit the build/Makefile.local file with the location of the netCDF library (and possibly also with the location of the HDF5 library on which your netCDF library is built).** This involves specifying the path to the library installation in the “NETCDF_PREFIX” variable and uncommenting one “FFLAGS_NETCDF” definition and one “LDFLAGS_NETCDF” definition appropriate to your build of the library. Since netCDF v4 depends on HDF5, you may also need to specify the HDF5 library location in “HDF5_PREFIX” and uncomment “LDFLAGS_HDF5”.



Similarly, if you want to compile against an external LAPACK library (optional, does not impact functionality), you must also edit the LAPACK section of **build/Makefile.local** with the details of this library before running the **rttov_compile.sh** script. This can be useful if the LAPACK routines included in the RTTOV source result in conflicts with other libraries in your system.

Once the code is compiled you will find **bin/** and **lib/** directories in your top-level RTTOV directory containing the RTTOV binaries and libraries (section 6). One library is created for each subfolder within **src/** and you should link all required libraries in your application (at the very least **librttov14_main** and **librttov14_coef_io** – see section 6.3 for more information). Tables 6.2 and 6.3 list all libraries and executables produced by the build process.

The file **src/test/Makefile_examples** is an example stand-alone Makefile for the **example_*.F90** demonstration programs. It has a section at the top which describes the variables that should be edited with paths appropriate for your system. This is intended as a demonstration of how to link your own code against the RTTOV libraries: the **example_*.F90** executables are compiled by the RTTOV build process so it is **not** necessary to use this Makefile to compile the example code.

Notes on compiling with the netCDF library:

1. The netCDF v4 library must be built with the Fortran interface (see the netCDF documentation).
2. In **build/Makefile.local** the FFLAGS_NETCDF variable defines the **_RTTOV_NETCDF** macro. It is important to supply this macro to the compiler so that the sections of code which do NETCDF I/O are included in the compilation. For most Linux-based Fortran compilers this is achieved by passing **-D_RTTOV_NETCDF** as seen in the FFLAGS_NETCDF variable, but for XLF on AIX it is passed using **-WF,-D_RTTOV_NETCDF**.
3. Note that if you do NOT compile with the netCDF library, you must NOT supply the **_RTTOV_NETCDF** macro to the compiler (so the FFLAGS_NETCDF lines in **Makefile.local** must be commented out).
4. If you specified the HDF5 library on which your netCDF build depends in **Makefile.local**, then before running RTTOV ensure the HDF5 library is in your **\$LD_LIBRARY_PATH** or system equivalent.

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

Compiling RTTOV manually

It is recommended to use the **build/rttov_compile.sh** script described in the previous section, but manual compilation of RTTOV is possible. *As noted above, if compiling with netCDF you must first edit the build/Makefile.local file with the location of the netCDF library (and possibly also the HDF5 library).*

RTTOV makes use of some LAPACK subroutines: the source code for these is included with RTTOV and by default you do not have to worry about this. However, if you wish instead to compile against an external library containing the LAPACK subroutines you can specify this library in **build/Makefile.local** in a similar manner to the netCDF library.

The general compilation procedure is then as follows:

```
$ cd src
$ ../build/Makefile.PL RTTOV_NETCDF=1 RTTOV_F2PY=1 RTTOV_USER_LAPACK=1
$ make ARCH=myarch INSTALLDIR=myinstalldir
```

The arguments in *italics* are optional.

The second step (running **Makefile.PL** to regenerate the RTTOV Makefiles) is not required if you are compiling RTTOV for the first time "out-of-the-box" and you do not require either netCDF or Python-related code to be compiled and you do not want to compile against an external LAPACK library. However, if you are compiling RTTOV with the netCDF library or a LAPACK library, or you want to compile the RTTOV Python interface, you must run **Makefile.PL** with one or more of the arguments shown above:

```
RTTOV_NETCDF=1      - required if compiling RTTOV against the netCDF library
RTTOV_F2PY=1       - required if compiling the RTTOV Python interface (required for the RTTOV GUI)
RTTOV_USER_LAPACK=1 - required if compiling RTTOV against an external LAPACK library
```

The arguments to "make" are:

ARCH - this argument is optional: if omitted RTTOV is compiled with gfortran-openmp, otherwise "myarch" should correspond to the name of one of the files in the build/arch/ directory. These files contain build flags for various common compilers/platforms. You can add new ones: see below for details. Using *-openmp flags will enable multi-threaded execution via the RTTOV parallel interfaces: this is generally recommended but is not necessary if you are implementing your own parallel architecture (e.g., in an NWP system).

INSTALLDIR - by default the build process creates output directories (e.g. **lib/** and **bin/**) in the top-level RTTOV directory. You can optionally specify "myinstalldir" to be another path relative to the top-level RTTOV directory to contain the **lib/**, **bin/** and other subdirectories (useful if compiling RTTOV with more than one set of compiler flags).

Some examples are given below, all run from within the **src/** directory:

```
>> Compile RTTOV with gfortran-openmp compiler flags without any external dependencies:
$ make
```

```
>> Compile all RTTOV code excluding the Python interface with gfortran-openmp with the netCDF library:
```


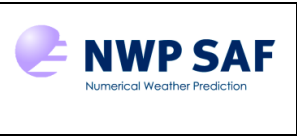
First edit build/Makefile.local with the location of your netCDF installation.

```
$ ../build/Makefile.PL RTTOV_NETCDF=1
$ make
```

```
>> Compile all RTTOV code with ifort-openmp compiler flags with the netCDF library:
```

First edit build/Makefile.local with the location of your netCDF installation.

```
$ ../build/Makefile.PL RTTOV_NETCDF=1 RTTOV_F2PY=1
$ make ARCH=ifort-openmp
```


		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

Compilation: more details

This section is not essential reading but contains some additional information about compilation which some users may find useful.

General notes

Programs should be compiled with the C-style preprocessor options enabled to make use of the #include statements for subroutine declarations. Note for most compilers this implies you need **.F90** as the file extension which is what is provided. For users with HP compilers it may be necessary to convert the **.F90** file extensions to **.f90** for all the routines.

Specifying an installation directory

As described above, by default the build process creates new directories (**bin/**, **lib/** etc) in the top-level RTTOV directory. It is possible to specify a subdirectory where the new directories will be placed by supplying the **INSTALLDIR** argument to “make”. This feature is useful if compiling RTTOV with different compiler flags or with different compilers. After compilation and testing the build directories can be moved to an arbitrary location (i.e., outside the RTTOV directory).

Specifying external dependencies

The file **build/Makefile.local** is used to specify the locations of external libraries, in particular the netCDF library. As noted above, for the netCDF code to be compiled requires the **_RTTOV_NETCDF** macro to be passed to the compiler. It is equally important that if the netCDF code is *not* required this macro is *not* supplied to the compiler.

The file contains templates for linking against the netCDF library, against the HDF5 library (sometimes required with netCDF4), and against a LAPACK library. An example for DrHook is also included: note that in the case of DrHook, the RTTOV source code includes **yomhook.F90**, a dummy routine, which must be removed from the **src/main/** directory if you want to run with DrHook enabled.

Regenerating the Makefiles

If the RTTOV code dependencies change for some reason or, if source files are added or removed from the **src/** directory, or if you want to include (or remove) the netCDF or Python capabilities, the Makefiles must be regenerated. This is easily achieved as follows:

```
$ cd src/
$ ../build/Makefile.PL RTTOV_NETCDF=1 RTTOV_F2PY=1 RTTOV_USER_LAPACK=1
$ make ARCH=myarch INSTALLDIR=mydir clean
```



where **RTTOV_NETCDF=1**, **RTTOV_F2PY=1**, and **RTTOV_USER_LAPACK=1** are described above. It is important to remember that if **RTTOV_NETCDF=1** was supplied to **Makefile.PL**, then **Makefile.local** must supply the **_RTTOV_NETCDF** macro to the compiler, and likewise, if **RTTOV_NETCDF=1** is *not* supplied to **Makefile.PL**, **Makefile.local** must *not* supply the macro.

It is good practice to do a “make clean” after running **Makefile.PL** to avoid problems when recompiling.

Creating an architecture configuration file

If the required architecture is not included in the **build/arch/** directory bundled with RTTOV or if you would like to customise the installation of RTTOV it is possible to create a new configuration file. This configuration file must be installed in the **build/arch/** directory and define the following macros:

FC : the name of the Fortran 2008 compiler.
FC77 : the name of the Fortran 77 compiler; this might be the same as **FC**, possibly with some special options.
CC: the name of the C compiler.
LDFLAGS_ARCH : specific flags to pass to the linker.
FFLAGS_ARCH : specific flags for the Fortran compiler.

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

CFLAGS_ARCH: specific flags for the C compiler.
AR : the command to create a library from object files.

NB The Fortran 77 and C compilers are used to compile a few specific source files. However, the RTTOV v14 software must be compiled with a Fortran 2008 compiler.

This configuration file may also define the following macros:

FFLAG_MOD: this is the flag used by the Fortran compiler to locate module files; it defaults to -I, but it is possible to override this setting.

CPP: the name of the pre-processor; defaults to cpp.

Specific flags for some RTTOV source files; defining **FFLAGS_ARCH_a** will force the build system to compile unit **a.F90** with these specific flags.

To use the Python interface (including the GUI), RTTOV must be compiled with f2py support. The following macros should be defined:

F2PY: this defines the f2py command and specifies the Fortran compiler being used (see the f2py documentation for relevant compiler names).

F2PYFLAGS_ARCH: compiler flags to pass to the F2PY compilation. This should specify the PIC (position independent code) flag in the appropriate form for the relevant Fortran compiler.

F2PYLDFLAGS_ARCH: linker flags for the F2PY compilation.

The existing files in **build/arch/** provide useful templates.

5.4 Testing your installation

NB Due to the way the Intel Fortran compiler manages memory, users compiling with ifort on Linux may need to increase the stack size by executing the following command before running tests described in this section or running RTTOV in your own application:

```
$ ulimit -s unlimited
```

In addition, if running the PC-RTTOV K model with multiple threads under ifort, you may need to increase the OpenMP stack size as well:

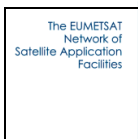
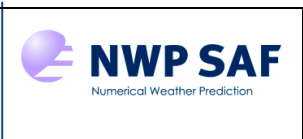
```
$ export OMP_STACKSIZE=1000M
```

You should first navigate to the **rttov_test/** directory which contains the relevant scripts and data for testing RTTOV. Unless otherwise specified all files and directories referred to in this section may be found in this directory.

RTTOV has a comprehensive and flexible test suite based on the **rttov_test.exe** executable which is run via the **rttov_test.pl** script. This allows most aspects of RTTOV to be configured and tested from the command-line and can compare the simulated outputs with reference data. Some shell scripts are included which run the test suite for a range of instruments, profiles, and options. These are described in section 5.4.1 below. The output from these tests can be visualised using a graphical interface written in Python (**rttov_test_plot.py**) which is described in the RTTOV test suite documentation in the **docs/** directory.

IMPORTANT NOTE: This test suite is intended for testing the code: it is not generally recommended to run RTTOV for your applications through this test harness. The input profiles supplied with the test suite have been selected for technical testing of the code, so should be checked carefully before being used for your own applications.

In addition to the test suite, there are several shell scripts that are used to run the **example_*.exe** demonstration programs which can be used as a template for your own code that calls RTTOV. These are also described below.

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

5.4.1 Verifying the RTTOV build

Several shell scripts are provided which run the RTTOV test suite for various instruments to test different aspects of RTTOV. Some of the scripts require coefficient files to be downloaded from the website (for example for hi-res IR sounders, aertable and/or hydrotatable optical property files, MFASIS-NN coefficients, or PC-RTTOV coefficients). Note that the hi-res IR sounder tests defined in the test suite expect coefficient files in netCDF format.

To verify your RTTOV installation you can run the following script without downloading any additional coefficient files:

```
$ ./test_rttov14.sh ARCH=myarch BIN=bindir
```

The **ARCH** parameter should match the one used when you compiled RTTOV (which is *gfortran-openmp* if you didn't explicitly specify **ARCH** when compiling). The **BIN** parameter is optional and is only required if the **INSTALLDIR** parameter was supplied when compiling RTTOV, i.e., if the location of **bin/** is not in the top-level RTTOV directory. If specified, **BIN** must give the location of the directory containing binary executables relative to the top-level RTTOV distribution directory (e.g., if you specified **INSTALLDIR=install/gfortran-openmp** when building RTTOV then you should use **BIN=install/gfortran-openmp/bin**). If you get failures related to the HDF5 or netCDF libraries not being found, remember to add the relevant HDF5 and/or netCDF **lib/** directories to your **\$LD_LIBRARY_PATH** (or equivalent) before running the test, e.g.

```
$ export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/full/path/to/hdf5/lib/
```

It is also possible to execute the test suite in a distributed computing environment (e.g., supercomputer) via a scheduler: you can optionally supply, for example, **SCHED_CMD=aprun**. Each executable run by **rttov_test.pl** will be called via **aprun** in this case. Additional arguments can be supplied using quotes: e.g. **SCHED_CMD="aprun --abc"**.

The **rttov_test14.sh** script runs the RTTOV direct, TL, AD, and K models for a range of instruments and compares the results to the supplied reference data. The test suite reports whether each individual test was successful or not. There may be cases where there are differences in the least significant digits between test output and the reference output due to compiler-dependent rounding errors (especially in the Jacobian output from the K model): these will be reported as differences but are not cause for concern. The **rttov_test/** directory contains several other shell scripts can be used to test different types of RTTOV simulations, but as noted above some require you to download the relevant coefficient files from the website:


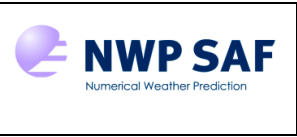
<code>test_fwd.sh</code>	tests the forward model for a wide range of instruments
<code>test_rttov14.sh</code>	tests the full code (direct/TL/AD/K) for a range of instruments
<code>test_rttov14_hires.sh</code>	tests the full code for hi-res IR sounders including scattering simulations
<code>test_solar.sh</code>	tests visible/near-IR solar clear-sky simulations
<code>test_multi_instrument.sh</code>	tests RTTOV running for multiple instruments together including scattering
<code>test_pc.sh</code>	tests the PC-RTTOV calculations

You can run any of these scripts in the same way as described above for **test_rttov14.sh**. Note that the tests expect netCDF format coefficient files for hyperspectral IR sounders. The script **test_core.sh** calls all the above scripts and this is run in the same way as the individual scripts. However, it is not necessary to run all the scripts to verify your RTTOV installation: calling **test_rttov14.sh** is sufficient.

A full description of the RTTOV test suite may be found in **docs/rttov-test.pdf** (under the top-level RTTOV directory). A brief overview is given here, but it is not necessary to read this to use RTTOV.

The **tests.0/** directory contains data required to run the tests: for each instrument this defines profile data, the channel and profile lists, specification of surface emissivity and reflectance, a reference to the RTTOV coefficients, and so on. Note that a small number of tests defined in **tests.0/** are intended for developers only and the relevant coefficient files are not available (this mostly involves coefficient files that have historically been available but are no longer generated). It is certainly not necessary or recommended to run every test defined in **tests.0/** and the tests that are run by the scripts listed above use coefficients which are available in the package or on the website.

Test outputs for the **myarch** architecture are located in **tests.1.myarch/** – these are created when the tests are run. Test reference output created on the NWP SAF test platforms is held in directories with names ending in **.2**.

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

The **rttov_test.exe** binary executable created during the building of RTTOV (and located in the **bin/** directory of the build) is used to run one or more tests. It is controlled by the **rttov_test.pl** Perl script. A typical test run involves a command like:

```
$ ./rttov_test.pl ARCH=myarch BIN=bindir TEST_LIST=hirs/001,avhrr/001 DIRECT=1
```

The **ARCH** and **BIN** parameters are the same as described above. The **TEST_LIST** parameter provides a list of tests defined in **tests.0/** to run. In this case, only the direct code is being tested (**DIRECT=1**). The test suite documentation provides a complete list of arguments which may be supplied to **rttov_test.pl** which allow almost all aspects of RTTOV to be tested. The full list of arguments may be listed by typing:

```
$ ./rttov_test.pl ARCH=myarch HELP=1
```

5.4.2 Running examples of code calling RTTOV v14

Several examples of running the RTTOV forward model and an example of calling the K model are provided which are intended to form a basis for your own applications. They are all in the **src/test/** directory:

- **example_fwd.F90** – simple example for clear-sky direct model simulations
- **example_k.F90** – simple example calling K model for clear-sky simulations.
- **example_atlas_fwd.F90** – same as **example_fwd.F90**, but demonstrates use of emissivity and BRDF atlases
- **example_aer_file_fwd.F90** – aerosol scattering simulations using an aertable optical property file.
- **example_aer_param_fwd.F90** – aerosol scattering simulations passing explicit optical properties.
- **example_hydro_visir_file_fwd.F90** – VIS/IR hydrometeor scattering using a hydrotatable optical property file.
- **example_hydro_mw_file_fwd.F90** – MW hydrometeor scattering using a hydrotatable optical property file.
- **example_hydro_param_fwd.F90** – hydrometeor scattering simulations passing explicit optical properties.
- **example_hydro_mfasis_nn_fwd.F90** – visible MFASIS-NN hydrometeor scattering simulation.
- **example_pc_fwd.F90** – demonstrates calling PC-RTTOV

Each of these programs may be run via a shell script in the **rttov_test/** directory with the name **run_example_*.sh** corresponding to the executable name. Near the top of each script is a small section where inputs may be configured such as the coefficient file and its location and the name of the input file(s) for profile data. The scripts may be run by typing (for example):

```
$ ./run_example_fwd.sh ARCH=myarch BIN=bindir
```

The **ARCH** and **BIN** arguments are described above in section 5.4.1. Test reference outputs are in **test_example.2/**. Input files for the script are in **test_example.1/**, and this is also where the test outputs are written. The outputs consist of files named **output_example_*.dat.myarch** and diff files named **diff_example_*.myarch** showing the differences between the test outputs and the reference outputs. The diff files should typically have zero size. In some cases, they might show differences in the least significant digits which result from compiler-dependent factors and are not cause for concern.

IMPORTANT NOTE: The source code for these examples is intended to be used as a template for your own applications calling RTTOV. The code contains comments showing which sections you will want to modify. In particular, you should modify the code that reads the input profile data and writes the output data. These examples have been created to be human-readable and easy to understand, but this is not efficient or desirable for typical real-world applications of RTTOV.

6 RTTOV types, subroutines, libraries, and executables

This section summarises the derived types and subroutines intended to be used in your own programs, and the output libraries and executables from the build process. It also provides information on how to link your code to RTTOV, and on producing RTTOV interface documentation using Doxygen.

6.1 RTTOV derived types

Table 6.1 provides a list of derived types (structures) which you may use in your application. All types are defined in the `src/main/rttov_types.F90` module. The two exceptions to this are `rttov_emis_atlas_data` which is stored in `src/emis_atlas/rttov_emis_atlas_mod.F90` and `rttov_brdf_atlas_data` which is stored in `src/brdf_atlas/rttov_brdf_atlas_mod.F90`. These are used to hold data for the land surface atlases.

Annex J provides details of all the derived types for which you may need to interact explicitly with the individual member variables in your code.

Type name	Purpose
<code>rttov_options</code>	RTTOV options structure to configure simulations.
<code>rttov_coefs</code>	Coefficients structure for RTTOV optical depth coefficients, hydrometeor/aerosol optical properties, MFASIS-NN coefficients, and PC-RTTOV coefficients.
<code>rttov_chanprof</code>	Define channel/profile indexes to simulate.
<code>rttov_profile</code>	Input atmospheric and surface profile data.
<code>rttov_emis_refl</code>	Input/output surface emissivity and reflectance data.
<code>rttov_transmission</code>	Calculated atmospheric transmittances.
<code>rttov_radiance</code>	Calculated satellite-seen radiances, brightness temperatures, and reflectances.
<code>rttov_radiance2</code>	Secondary direct model radiance outputs (optional).
<code>rttov_opt_param</code>	Explicit aerosol/hydrometeor optical property profiles (optional, required for scattering simulations with explicit optical properties).
<code>rttov_reflectivity</code>	Calculated radar reflectivities for hydrometeor scattering simulations (optional, required for radar simulations).
<code>rttov_emis_retrieval_terms</code>	Output structure containing data enabling dynamic emissivity retrievals (optional).
<code>rttov_diagnostic_output</code>	Output structure containing additional per-profile data computed during simulations (optional).
<code>rttov_pccomp</code>	Calculated PC scores and reconstructed radiances from PC-RTTOV (optional, required for PC-RTTOV simulations).
<code>rttov_traj</code>	Holds various internal variables (optional).
<code>rttov_traj_dyn</code>	Holds various internal variables (optional).
<code>rttov_traj_sta</code>	Holds various internal variables (optional).
<code>rttov_emis_atlas_data</code>	Holds data for an emissivity atlas for one month (from <code>rttov_emis_atlas_mod</code> , optional, required when using emissivity atlas).
<code>rttov_brdf_atlas_data</code>	Holds data for a BRDF atlas for one month (from <code>rttov_brdf_atlas_mod</code> , optional, required when using BRDF atlas).

Table 6.1: User-level derived types.

6.2 RTTOV libraries and subroutines

The build process produces a library for every directory within `src/` that is included in the build target provided to `make`. Table 6.2 lists all libraries created in the `lib/` directory by the compilation and the associated user-level subroutines contained therein. All user-level subroutine interfaces are detailed in the Annexes.

Library name	User-level subroutines	Description
<code>librttov14_main.a</code>	<code>rttov_direct</code> , <code>rttov_tl</code> , <code>rttov_ad</code> , <code>rttov_k</code> <code>rttov_alloc_direct</code> , <code>rttov_alloc_tl</code> <code>rttov_alloc_ad</code> , <code>rttov_alloc_k</code> <code>rttov_alloc_profiles</code> , <code>rttov_init_profiles</code>	Core library, always required. Interfaces to the direct, TL, AD, K models. Allocation and initialisation subroutines for RTTOV structures.

	rttov_alloc_emis_refl, rttov_init_emis_refl rttov_alloc_transmission rttov_init_transmission rttov_alloc_radiance rttov_init_radiance rttov_alloc_opt_param rttov_init_opt_param rttov_init_opt_param_solar rttov_alloc_reflectivity rttov_init_reflectivity rttov_alloc_emis_ret_terms rttov_init_emis_ret_terms rttov_alloc_diagnostic_output rttov_init_diagnostic_output rttov_alloc_pccomp, rttov_init_pccomp rttov_alloc_traj_all, rttov_errorhandling rttov_user_check_options rttov_user_check_profile rttov_user_check_emis_refl rttov_user_check_opt_param rttov_get_sea_emis, rttov_get_sea_brdf	Subroutine to set the logical unit for error messages. User subroutines to verify input data. Subroutines to obtain sea surface emissivities and BRDFs from the internal RTTOV surface models.
librttov14_netcdf.a	N/A	Required if RTTOV was compiled with netCDF.
librttov14_coef_io.a	rttov_read_coefs, rttov_dealloc_coefs rttov_get_pc_predictindex	Coef input/output, always required. Read coefficient files, deallocate memory and get PC-RTTOV predictor channel sets.
librttov14_parallel.a	rttov_parallel_direct, rttov_parallel_tl rttov_parallel_ad, rttov_parallel_k	Parallel interfaces for multi-threaded execution using OpenMP.
librttov14_emis_atlas.a	rttov_setup_emis_atlas, rttov_get_emis rttov_deallocate_emis_atlas	Interface to emissivity atlases.
librttov14_brdf_atlas.a	rttov_setup_brdf_atlas, rttov_get_brdf rttov_deallocate_brdf_atlas	Interface to BRDF atlas.
librttov14_other.a	rttov_print_opts, rttov_print_info rttov_print_profile rttov_print_radiance_quality rttov_wmo2rttov_sat_id rttov_calc_geo_sat_angles rttov_calc_solar_angles rttov_scale_ref_gas_prof rttov_aer_clim_prof rttov_emissivity_retrieval rttov_bpr_calc, rttov_bpr_init rttov_bpr_dealloc rttov_asym_calc, rttov_lcoef_calc load_bfield_lut compute_bfield, compute_kb_angles	Ancillary subroutines: <ul style="list-style-type: none"> • print out contents of options, coef, and profile structures for debugging. • print quality output in human readable form. • map WMO satellite IDs to RTTOV platform and satellite ID couplets • calculate GEO sensor angles and solar angles. • generate scaled copies of the RTTOV background trace gas profiles. • generate climatological aerosol profiles. • compute dynamic emissivity retrieval using emissivity retrieval terms optional output. • calculate “bpr” parameter, asymmetry parameter, and Legendre coefficients for given phase function. • obtain values of magnetic field strength and direction for Zeeman simulations
librttov14_wrapper.a	See separate wrapper user guide for wrapper API.	Link against this library when compiling your own C++ code which uses the RTTOV wrapper.
rttov_wrapper_f2py.so		F2PY library for Python interface to RTTOV.
The remaining libraries are not intended for linking: librttov14_test.a, librttov14_mw_scatt_coef.a		

Table 6.2: Libraries produced by the build process and associated user-level subroutines.

6.3 Linking your code against RTTOV

When linking against RTTOV (for example in a Makefile that you are creating) you only need to link against the RTTOV libraries which contain subroutines that you have used (see Table 6.2). The order in which the libraries are linked is important: if library *A* depends on library *B* then *A* must appear before (i.e., to the left of) *B* in the linking step. You can link the RTTOV libraries (specifying the path to your RTTOV **lib/** directory) as follows:

```
-L/path/to/RTTOV/lib \
-lrttov14_wrapper -lrttov14_brdf_atlas -lrttov14_emis_atlas -lrttov14_other \
-lrttov14_parallel -lrttov14_coef_io -lrttov14_netcdf -lrttov14_main
```

The file **src/test/Makefile_examples** is an example Makefile for the **example_*.F90** executables which may be used as a template for compiling your own code which calls RTTOV. As noted in section 5.3 the example executables are built when RTTOV is compiled: this example Makefile is only for demonstration purposes.

6.4 RTTOV executables

Table 6.3 gives a list of all executables produced by the build process in **bin/** and their purpose. Most test executables are intended to be called via the Perl and shell scripts found in **rttov_test/**. See section 5.4 for more information on calling test programs. Aside from the test programs, the most used executable is the coefficient conversion tool (Annex A).

When calling RTTOV from Python (including when using the RTTOV GUI) the file **rttov_wrapper_f2py.so** must be in your current directory or in your **\$PYTHONPATH**. This file is found in the **lib/** directory after compilation. See the separate user guides for the wrapper and the GUI in the **docs/** directory.

Executable name	Purpose
Useful executables related to coefficient and optical property files.	
rttov_coef_info.exe	Print out information about a given <i>rtcoef_</i> coefficient file, primarily intended for use with binary and netCDF coefficient files (Annex A).
rttov_conv_coef.exe	Convert coefficients between formats (ASCII, binary, netCDF) and extract channel subsets to reduce file sizes (Annex A, section 5.2).
rttov_make_aertable.exe	Generate aerosol optical property (<i>rttov_aertable_</i>) files (section 8.4.10).
rttov_scatt_make_coef.exe	Generate hydrometeor optical property (<i>rttov_hydratable_</i>) files for MW sensors (section 8.4.8).
Executables for example programs that can be used as a basis for your code: run these using the corresponding <i>run_example_*.sh</i> scripts (section 5.4.2)	
example_fwd.exe	Example program demonstrating simple forward model call.
example_k.exe	Example program demonstrating simple K model call.
example_atlas_fwd.exe	Example program demonstrating forward model call with use of emissivity and BRDF atlases.
example_aer_file_fwd.exe	Example program demonstrating forward model call with aerosol scattering using pre-defined optical properties.
example_aer_param_fwd.exe	Example program demonstrating forward model call with aerosol scattering using explicit optical property profiles.
example_hydro_mfasis_nn_fwd.exe	Example program demonstrating forward model call with hydrometeor scattering using the MFASIS-NN fast solar solver.
example_hydro_mw_file_fwd.exe	Example program demonstrating forward model call with hydrometeor scattering using pre-defined optical properties for MW sensors.
example_hydro_visir_file_fwd.exe	Example program demonstrating forward model call with hydrometeor scattering using pre-defined optical properties for VIS/IR sensors.
example_hydro_param_fwd.exe	Example program demonstrating forward model call with hydrometeor scattering using explicit optical property profiles.
example_pc_fwd.exe	Example program demonstrating forward model call for PC-RTTOV.

Executable for main test suite: this should be run via the <i>rttov_test.pl</i> script in <i>rttov_test/</i> (section 5.4.1)	
<i>rttov_test.exe</i>	Main test suite executable: should be called using <i>rttov_test.pl</i> .
Developer executables for testing ancillary RTTOV capabilities: not intended for users to run	
<i>rttovscatt_test.exe</i>	Developer test program for MW scattering via delta-Eddington solver, run using the <i>test_rttovscatt.sh</i> script.
<i>rttov_uwiremis_atlas_test.exe</i>	Developer test program for UWIREmis IR emissivity atlas.
<i>rttov_camel_atlas_test.exe</i>	Developer test program for CAMEL IR emissivity atlas.
<i>rttov_camel_clim_atlas_test.exe</i>	Developer test program for CAMEL climatology IR emissivity atlas.
<i>rttov_telsem2_atlas_test.exe</i>	Developer test program for TELSEM2 MW emissivity atlas.
<i>rttov_cnrm_mw_atlas_test.exe</i>	Developer test program for CNRM MW emissivity atlas.
<i>rttov_brdf_atlas_test.exe</i>	Developer test program for BRDF atlas.
<i>rttov_test_get_sea_emis_brdf.exe</i>	Developer test program for <i>rttov_get_sea_emis/brdf</i> subroutines
<i>rttov_test_calc_geo_sat_angles.exe</i>	Developer test program for <i>rttov_calc_geo_sat_angles</i> subroutine
<i>rttov_test_calc_solar_angles.exe</i>	Developer test program for <i>rttov_calc_solar_angles</i> subroutine
<i>rttov_test_wmo2rttov_sat_id.exe</i>	Developer test program for <i>rttov_wmo2rttov_sat_id</i> subroutine
<i>rttov_test_model_alloc.exe</i>	Developer test program for <i>rttov_alloc_direct/tl/ad/k</i> subroutines.
<i>rttov_zutility_test.exe</i>	Developer test program for the routines in <i>rttov_zutility_mod</i> (see Annex I).
Additional executables.	
<i>create_aer_clim_prof.exe</i>	Generate a file containing climatological aerosol profiles from combinations of the OPAC pre-defined particle types (Annex I).
<i>rttov_obs_to_pc.exe</i>	Demonstrating conversion of observations to PC-space for PC assimilation applications (section 8.6, Annex I).
<i>rttov_test_get_pc_predictindex.exe</i>	Test program for <i>rttov_get_pc_predictindex</i> subroutine, can be used to obtain the PC-RTTOV predictor channel numbers on the command-line (section 8.6, Annex I).
<i>rttov_make_opt_param.exe</i>	Used by main test suite to create explicit optical property input files from hydrometeor/aerosol profiles and associated hydrotable/aertable files. Not intended to be run explicitly by users.


Table 6.3: Executables produced by the RTTOV build process.

6.5 Doxygen documentation

Doxygen markup has been added to the user-level subroutines and data types. If you have Doxygen installed, you can generate the documentation by running the following from the top-level RTTOV directory:

```
$ doxygen docs/doxygen_config_user
```

The resulting documentation can be found in **docs/doxygen_doc_user/**. HTML documentation can be found in **docs/doxygen_doc_user/html/index.html**. This may be helpful as a reference when reading sections 7 and 8: section 7 describes the technical aspects and the process of calling RTTOV while section 8 describes additional simulation capabilities of RTTOV.

<p>The EUMETSAT Network of Satellite Application Facilities</p>		<h2 style="text-align: center;">RTTOV v14 Users' Guide</h2>	<p>Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025</p>
---	---	---	--

7 Running RTTOV v14 for your applications

This section discusses key aspects of setting up and running basic clear-sky RTTOV simulations in Fortran. Additional simulation capabilities are discussed in section 8. RTTOV includes interfaces to allow much RTTOV functionality to be called from Python or C++ code. This allows you to run RTTOV without writing any Fortran code. There is a separate document in the **docs/** directory which describes the wrapper interface. However, it is still important to be aware of the contents of this section of the user guide and the relevant parts of section 8, as you will not be able to run RTTOV correctly without understanding them.

There are several example programs in the **src/test/** directory which can serve as a basis for your application such as **example_fwd.F90** and **example_k.F90** demonstrating clear-sky forward and K model runs. These are listed in section 5.4.2 with instructions on how to run them.

The module **rttov_types** defines the derived types used by RTTOV that you will need in your program. These are listed in Table 6.1 and Annex J gives full details. There are also two additional derived types contained in **rttov_emis_atlas_mod** and **rttov_brdf_atlas_mod** which are required when using the emissivity and BRDF atlases: see section 6.1, sections 8.3.6 and 8.3.7, and **example_atlas_fwd.F90**.

The module **rttov_kinds** defines the standard RTTOV integer, real and logical kinds (**jpim**, **jprv** and **jplm** respectively). The default RTTOV integer kind is 32-bit and the default real kind is double precision (64-bit): it is *not* recommended (and hence not supported) to change the latter to single precision (32-bit). This is particularly true if running the AD or K models as this can significantly affect the adjoint/Jacobian output.

The **rttov_types** and **rttov_kinds** modules should always be used in your code which calls RTTOV. You will also see that the example code uses the **rttov_const** module: this contains many parameters (constants) used by RTTOV, some of which may be useful in your program (see Annex K). The files **rttov_types.F90**, **rttov_kinds.F90** and **rttov_const.F90** can be found in **src/main/**.

Figure 7.1 gives a process diagram of the minimal set of subroutines to call when running RTTOV v14.

The following sections describe the recommended steps to be taken in coding a program which calls RTTOV v14. These involve preparing structures and arrays with the necessary input data for RTTOV and creating appropriate structures and arrays to hold the output of the model. To provide some context, the syntax for calling **rttov_direct** is provided here with the arguments containing input data highlighted in bold. The subroutine interface is described fully in Annex H.

```
call rttov_direct(errorstatus, opts, coefs, chanprof, profiles, emis_refl,
                 transmission, radiance, radiance2,
                 aer_opt_param, hydro_opt_param, refl_cloud_top,
                 reflectivity, emis_retrieval_terms, diag_output,
                 pccomp, channels_rec, traj, traj_dyn, traj_sta)
```

Input arguments to the RTTOV direct model:

- **opts** - options to configure the simulation, section 7.1
- **coefs** - coefficients/optical properties, section 7.2
- **chanprof** - channels/profiles to simulate, section 7.5
- **profiles** - input profile and surface variables, section 7.4
- **emis_refl** - surface emissivity and reflectance data, section 8.3
- **aer_opt_param** - explicit aerosol optical properties, section 8.4.11
- **hydro_opt_param** - explicit hydrometeor optical properties, section 8.4.11
- **refl_cloud_top** - cloud top reflectance for simple cloud scheme, section 8.2
- **channels_rec** - channels for which to compute reconstructed radiances in PC-RTTOV, section 8.6
- **traj**, **traj_dyn**, **traj_sta** - internal “trajectory” structures, section 7.3.2

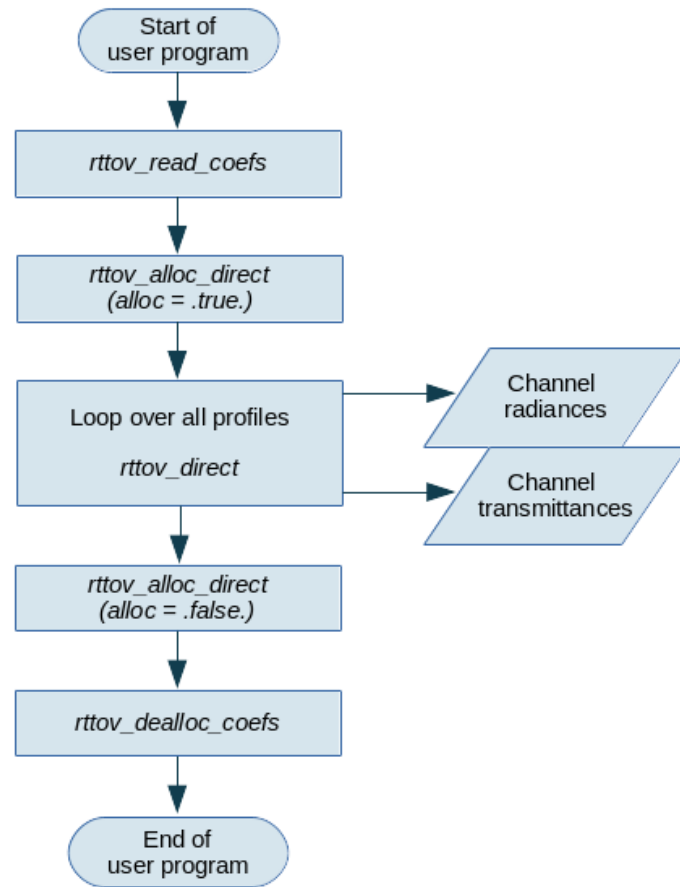


Figure 7.1: Process diagram of user program calling RTTOV v14 forward model. You can call individual subroutines for allocating/deallocating each input/output structure separately instead of calling the `rttov_alloc_direct` subroutine (section 7.3).



7.1 Set RTTOV options

You must declare a variable of the `rttov_options` derived type, for example named “`opts`”. This structure contains various variables for configuring the simulation. The full structure is described in Annex J. The options are grouped by function among sub-types.

It is recommended to check the default options carefully and to ensure that all options relevant to your simulation are appropriately specified. **Note that default values for several options have changed since RTTOV v13.** Options are discussed throughout this user guide in the relevant sections.

It is important to set options covering the widest range of required types of simulation *before* reading the coefficients and allocating RTTOV structures. For example, if hydrometeor scattering simulations are required then set `opts%scatt%hydrometeors` to true at the start before reading the coefficients and allocating the `profiles(:)` structure (see sections 7.2 and 7.3 and Annex C). This ensures that the hydrometeor optical properties are read in, and the hydrometeor profile array variables are allocated. You can subsequently set the `hydrometeors` option to false to run clear-sky simulations without reallocating structures or re-reading coefficients. This applies to other options including (but not limited to) `opts%rt_all%solar`, `opts%surface%lambertian`, and `opts%scatt%aerosols`.

You can also initialise the logical unit for error/warning messages. This is performed by `rttov_errorhandling`, an optional subroutine which can be called at any time (see Annex B). This logical unit number is stored as a module variable, so modifying its value is not a thread-safe operation.

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

7.2 Read coefficient/optical property files

All RTTOV simulations require an optical depth (*rtcoef*) coefficient file for the sensor being simulated. Aerosol/hydrometeor simulations may require optical property (*rttov_aertable/rttov_hydratable*) files for the given sensor (see section 8.4). MFASIS-NN requires the corresponding neural network (*rttov_mfasis_nn*) coefficient file (see section 8.4.2). PC-RTTOV requires a PC coefficient (*pccoef*) file (see section 8.6). Finally, the ARO-scaling polarisation treatment for MW hydrometeor simulations requires an additional sensor-independent file (section 8.4.6).

The data from all files required for the simulations to be performed is read into the **coefs** structure (derived type **rttov_coefs**) using a single call to the **rttov_read_coefs** subroutine. This provides various options for how to specify the files to be read in, it automatically detects the format of input files (ASCII, binary, or netCDF – see section 5.2), and it allows a subset of sensor channels to be read in from the files. Certain options must be set before calling this routine to enable reading of additional optical property and/or coefficient files. Full details are given in Annex C.

After reading the coefficients you may wish to call the subroutine **rttov_user_check_options** (see Annex I) which checks the consistency of the input options with the coefficient file and reports any issues: this can be useful for debugging purposes. Section 7.7 gives more information on checking RTTOV inputs.

The optical depth coefficient file defines the variable trace gases allowed in the input profile (see section 3). In order for a gas to vary in the simulation, the coefficient file must contain coefficients for that gas, and you must set the corresponding flag in the options structure to true (see section 7.4.2). The fewer gases simulated, the faster the code will run. In all cases, water vapour is a mandatory input. All other trace gases (O₃, CO₂, CO, N₂O, CH₄, SO₂) are optional.

Where a coefficient file supports a variable gas and you do *not* supply an input profile (the corresponding flag is set to false – section 7.4.2), then RTTOV uses a pre-defined background profile for that gas. For most gases, the background profile is the mean profile of the training set. For SO₂ the background profile is a low concentration profile emulating a typical anthropogenic scenario. The background profiles can be found in the REFERENCE_PROFILE section of ASCII optical depth coefficient files or alternatively, they are available on the RTTOV coefficients download web page.

If you provide profiles for a gas which is not variable in the optical depth coefficient file, RTTOV ignores the input gas profiles. Issues like this which are harmless to a simulation, but potentially indicate a configuration error, can optionally be flagged by the **rttov_user_check_options** subroutine mentioned above (see Annex I, and section 7.7).

It is recommended to look at the header sections of the optical depth coefficient file for the sensor you wish to simulate as there is useful information there such as the definition of channel numbers and the polarisation (for MW sensors) assumed for each channel for that instrument etc. You can use the **rttov_coef_info.exe** executable (see Annex A) to view the headers of any coefficient file including those in binary (Fortran unformatted) or netCDF format.

7.3 Allocate RTTOV data structures

7.3.1 Core RTTOV structures

RTTOV provides a collection of subroutines for allocating/deallocating and initialising the data structures required for inputs and outputs to the model. The interfaces to all these routines are given in Annex D.

RTTOV provides top-level routines that can be used to (de)allocate any or all arguments to the core RTTOV routines (**rttov_direct**, **rttov_tl**, **rttov_ad**, **rttov_k** – see Annex H). These routines are named **rttov_alloc_direct**, **rttov_alloc_tl**, **rttov_alloc_ad**, and **rttov_alloc_k**.

Sometimes it is necessary or desirable to allocate RTTOV structures individually. Every structure containing array members has an associated (de)allocation subroutine.

Every structure also has an associated initialisation subroutine which can be used to set the data structure members to zero. This is particularly useful when making multiple calls to RTTOV, especially for the AD/K models where inputs must usually be initialised to zero before every call (see section 7.9).

The relevant data structures, **rttov_direct** argument names, and allocation and initialisation routine names are given in Table 7.1.

Data type	<i>rttov_direct</i> argument	Allocation/initialisation subroutines	Description
rttov_profile	profiles(<i>nprofiles</i>)	rttov_alloc_profiles rttov_init_profiles	Input profile and surface variables.
rttov_emis_refl	emis_refl(<i>nsurfaces</i>)	rttov_alloc_emis_refl rttov_init_emis_refl	Input/output surface emissivity, reflectance, and related variables.
rttov_transmission	transmission	rttov_alloc_transmission rttov_init_transmission	Output transmittances.
rttov_radiance rttov_radiance2	radiance radiance2	rttov_alloc_radiance rttov_init_radiance	Output radiances, brightness temperatures, and reflectances.
rttov_opt_param	aer_opt_param hydro_opt_param	rttov_alloc_opt_param rttov_init_opt_param rttov_init_opt_param_solar*	Input profiles of explicit aerosol/hydrometeor optical properties.
rttov_reflectivity	reflectivity	rttov_alloc_reflectivity rttov_init_reflectivity	Output radar reflectivities from radar solver.
rttov_emis_retrieval_terms	emis_retrieval_terms	rttov_alloc_emis_ret_terms rttov_init_emis_ret_terms	Output data that can be used for dynamic surface emissivity retrievals.
rttov_diagnostic_output	diag_output	rttov_alloc_diagnostic_output rttov_init_diagnostic_output	Additional per-profile outputs.
rttov_pccomp	pccomp	rttov_alloc_pccomp rttov_init_pccomp	Output PC scores, and reconstructed radiances and brightness temperatures from PC-RTTOV.
rttov_traj rttov_traj_dyn rttov_traj_sta	traj	rttov_alloc_traj_all (no initialisation routine)	Data structures containing internal RTTOV state.

Table 7.1: allocation and initialisation subroutines for the data types of arguments to *rttov_direct*.

*See section 8.4.11 for more information about explicit optical property initialisation.



The input/output data structures should usually be allocated once before any calls to RTTOV are made, and then they should be deallocated at the end of your code after all calls to RTTOV are completed. The same routines are used for allocation and deallocation, the action being determined by a logical argument **alloc** passed to each routine (true for allocate, false for deallocate). More generally it is recommended to pass the **alloc_flag** and **dealloc_flag** constants defined in **rttov_const** (Annex K).

Some allocation subroutines have an optional logical argument **direct**. This should be set to true (or omitted) when allocating the structures for the direct model inputs/outputs (e.g., **radiance**, **transmission**). This argument may be set to false when allocating the corresponding TL/AD/K arguments (e.g., **radiance_tl/ad/k**, **transmission_tl/ad/k**): in the latter case the members that are not used by the TL/AD/K models are not allocated, thereby saving memory. There is no harm in omitting this argument in all cases: all members will then be allocated.

Most allocation subroutines have an optional logical argument **init**. When this is set to true, the members of the newly allocated structure will be initialised (usually to zero) by a call to the corresponding initialisation routine.

Before calling the allocation subroutines, it is important to first specify the options to be used in the simulations. This is because RTTOV allocates only the data structure members required for the simulations being carried out in order to be as efficient in memory usage as possible. If you plan to carry out different types of simulations (e.g., with/without solar radiation, with/without hydrometeors), then you should specify the broadest selection of options (e.g., **opts%rt_all%solar=true**, **opts%scatt%hydrometeors=true**) before calling the allocation subroutines. Then you can set the options as required before each subsequent call to **rttov_direct**.

Some allocation routines accept a **coefs** argument (type **rttov_coefs**) containing the RTTOV coefficients read by **rttov_read_coefs** (section 7.2). This should be populated before calling the allocation routines.

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

7.3.2 Trajectory structures

This section describes a method that can yield a performance boost for RTTOV simulations on some architectures where the RTTOV OpenMP parallel interface (section 7.10) is not being used. This is of particular relevance if you are implementing your own architecture for running RTTOV in parallel (e.g., in an NWP system) as the technique described here is not compatible with RTTOV's own parallel interface. If you are running RTTOV on a workstation with multiple cores and performance is critical then the RTTOV parallel interface will give a greater performance boost and is recommended.

RTTOV allocates memory internally to pass calculation results around between subroutines, and to store calculation results from the direct model which can then be re-used in the TL/AD/K models without repeating the full calculations. If many calls to RTTOV are being made it may be more efficient to allocate the internal data structures before calling RTTOV and then to deallocate these structures once all calls to RTTOV have been made. This can be achieved using the optional **traj**, **traj_dyn**, and **traj_sta** arguments to **rttov_direct** of types **rttov_traj**, **rttov_traj_dyn**, and **rttov_traj_sta** respectively (and **traj_tl/ad/k** and **traj_tl/ad/k_dyn** for the TL/AD/K models). These can be allocated using the **rttov_alloc_traj_all** subroutine (Annex D) or using the **rttov_alloc_direct/tl/ad/k** subroutines mentioned above. You should make a second call to deallocate the structures once they are no longer required as for other structures. Note that you can pre-allocate any combination of these structures and pass them into RTTOV, i.e., it is not mandatory to allocate/pass either all of them or none of them.

Important note: the trajectory structures are sized according to the number of channels (size of the **chanprof(:)** array, section 7.5), profiles, surfaces, and levels, and according to certain options such as the active trace gases, scattering configuration, and solar radiation, so if any of these values change for any calls to RTTOV then new trajectory structures must be allocated. It is also required that the **coefs** variable (of type **rttov_coefs**) be declared as a TARGET when passing any of the **traj**, **traj_dyn**, and/or **traj_sta** arguments into RTTOV.

Whether this capability offers performance benefits is dependent on the compiler and architecture: memory allocations are more expensive on some systems than others. If you are interested in minimising run-time it is recommended to carry out a test on your system to see if this offers a benefit: it has been observed to be beneficial on some Intel/Linux-based systems (especially with gfortran) and on some NEC supercomputers. Use of the trajectory arguments should never be detrimental to performance in most cases because RTTOV allocates the structures internally if the arguments are not present. **However**, for hydrometeor scattering simulations using the maximum/random overlap parameterisation (section 8.4.3), allocating the **traj_dyn** structure outside of RTTOV will require more memory than allowing RTTOV to allocate it internally because the arrays must be sized maximally to accommodate the largest possible number of cloud columns generated by the simulation. When allocated internally, RTTOV allocates the data structure according to the actual number of cloud columns computed at run-time. External allocation of all three trajectory structures has been observed to slow down direct model hydrometeor simulations with the DOM solver so some experimentation is warranted for this case to test simulation speed. It may be better to pre-allocate only the **traj** and **traj_sta** structures.

7.4 Specifying input profiles

RTTOV takes as input an array **profiles(:)** of type **rttov_profile** and of size **nprofiles**. This array specifies atmospheric and surface variables for an arbitrary number of profiles that will be passed into RTTOV. You specify **nprofiles** as the number of profiles you wish RTTOV to process in each call.

RTTOV v14 represents the atmosphere via a set of pressure “half-levels”. All other atmospheric parameters (temperature, water vapour, trace gas concentrations, aerosols, hydrometeors, etc) are specified for the layers bound by these pressure levels (see Figure 7.2). The surface *always* (implicitly) lies on the bottom pressure half-level. This means that, unlike previous RTTOV versions, RTTOV v14 cannot be run on fixed pressure levels with the surface pressure specified independently of the pressure levels. The pressure half-levels will usually vary from profile to profile. Figure 7.2 shows the pressure “full-levels” that interleave the half-levels. It is not necessary to specify the full-levels: if they are unspecified (i.e., all zero) RTTOV computes them as the mean of adjacent full-level pressures. This is consistent with some NWP models, but if you are using fields from an NWP model that computes pressure full-levels differently, it is recommended to supply the full-levels explicitly.

In the RTTOV documentation and code, any reference to (pressure) “levels” means half-levels. The full-levels are associated with the atmospheric layers bound by the half-levels, so the terms “full-levels” and “layers” refer to the same thing.

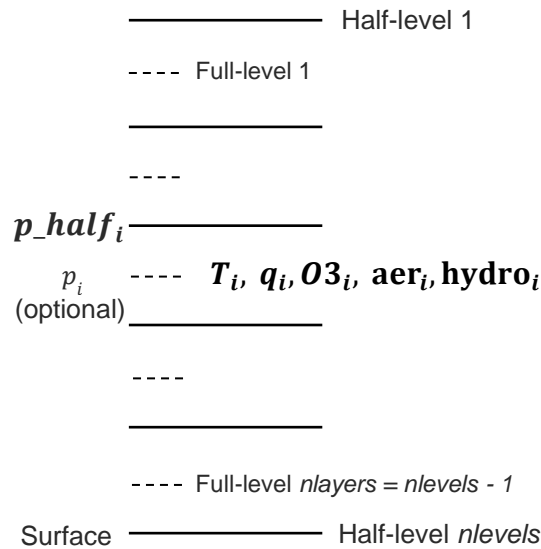


Figure 7.2: RTTOV input profile structure. Layers are bound by pressure half-levels. All input profile quantities (temperature, water vapour, trace gases, aerosol, and hydrometeor information) are provided on the pressure full-levels which interleave the half-levels. The surface implicitly lies on the bottom pressure half-level.

RTTOV also allows each profile to be associated with one or more surfaces, each covering a fraction of the area within the field of view. The treatment of multiple surfaces is described fully in section 8.3.10. The input profile includes near-surface atmospheric and surface skin variables for each of the surfaces.

All profiles passed to RTTOV have the same number of levels (**nlevels**) and the same number of surfaces (**nsurfaces**).

There are many variables in the RTTOV profile structure. Some are mandatory for all simulations, while some are only mandatory under certain configurations of the model. A few are always optional. Table 7.2 lists all profile variables and gives the conditions under which each is required (where relevant). You must populate the relevant variables for all **nprofiles** profiles before each call to RTTOV. Table 7.2 also indicates which profile variables are active, i.e., not assumed constant, in the TL/AD/K models (see section 7.9).

For geostationary sensors, RTTOV provides a subroutine, **rttov_calc_geo_sat_angles**, which can be used to calculate and set the **zenangle** and **azangle** members (satellite zenith and azimuth angles) of a **profiles(:)** array assuming the **latitude** and **longitude** members have been populated for all profiles. This is described in Annex I.

Input profile members	Description	Units	Mandatory ?	When used	Variable for TL?
<i>nlevels</i>	Number of pressure half-levels <i>Populated by allocation routine.</i>		Y		N
<i>nlayers</i>	Number of pressure full-levels/ atmospheric layers (i.e. <i>nlevels</i> -1) <i>Populated by allocation routine.</i>		Y		N
<i>nsurfaces</i>	Number of surfaces. <i>Populated by allocation routine.</i>		Y		N
<i>gas_units</i>	Units of gas concentrations (<i>must be the same for all profiles</i>)	0,1,2	Y		N
<i>p_half(:)</i>	Pressure half-levels (<i>nlevels</i>)	hPa	Y		Y if <i>opts % interpolation % pressure</i>

					<i>gradients true</i>
<i>p(:)</i>	Pressure full-levels (<i>nlayers</i>)	hPa	N	Always, computed internally as mean of adjacent half-level pressures if not specified.	Y if <i>opts % interpolation % pressure_gradients true</i>
<i>t(:)</i>	Temperatures on full-levels	K	Y		Y
<i>q(:)</i>	Water vapour concentration on full-levels	ppmv or kg/kg	Y		Y
<i>o3(:)</i>	O ₃ concentration on full-levels	ppmv or kg/kg	N	If <i>o3_data</i> option .true.	Y
<i>co2(:)</i>	CO ₂ concentration on full-levels	ppmv or kg/kg	N	If <i>co2_data</i> option .true.	Y
<i>n2o(:)</i>	N ₂ O concentration on full-levels	ppmv or kg/kg	N	If <i>n2o_data</i> option .true.	Y
<i>co(:)</i>	CO concentration on full-levels	ppmv or kg/kg	N	If <i>co_data</i> option .true.	Y
<i>ch4(:)</i>	CH ₄ concentration on full-levels	ppmv or kg/kg	N	If <i>ch4_data</i> option .true.	Y
<i>so2(:)</i>	SO ₂ concentration on full-levels	ppmv or kg/kg	N	If <i>so2_data</i> option .true.	Y
<i>clw(:)</i>	Cloud liquid water on full-levels treated as absorbing medium; not used in scattering simulations.	kg/kg	N	MW non-scattering only if <i>clw_data</i> option .true.	Y
<i>mmr_aer</i>	Aerosol units: true => kg/kg, false => cm ⁻³ (<i>must be the same for all profiles</i>)	T/F	N	Aerosol simulations with <i>aertable</i> file	N
<i>mmr_hydro</i>	Hydrometeor units: true => kg/kg, false => g.m ⁻³ (<i>must be the same for all profiles</i>)	T/F	N	Hydrometeor simulations with <i>hydrotable</i> file	N
<i>flux_conversion(:) size (nhydro)</i>	Use to specify units of flux for rain and snow hydrometeor types (<i>nhydro</i> determined by hydrometeor optical property file)	0,1,2	N	Optional with hydrometeor simulations with <i>hydrotable</i> file	N
<i>aerosols(:,:) size (naer, nlayers)</i>	Grid box average aerosol components/species concentration on full-levels (<i>naer</i> determined by aerosol optical property file)	kg/kg or cm ⁻³	N	Aerosol simulations with <i>aertable</i> file	Y
<i>hydro(:,:) size (nhydro, nlayers) MW or (nhydro+1, nlayers) UV/VIS/IR</i>	Grid box average hydrometeor concentration on full-levels (<i>nhydro</i> determined by hydrometeor optical property file)	kg/kg or g.m ⁻³	N	Hydrometeor simulations with optical property file	Y
<i>hydro_frac(:,:) size (1, nlayers) or same as hydro(:,:) depending on per_hydro_frac option</i>	Cloud/hydrometeor fractional cover on full-levels	0-1	N	Hydrometeor simulations	Y
<i>hydro_frac_eff</i>	Effective cloud/hydrometeor fraction for the whole profile	0-1	N	Hydrometeor simulations with cloud overlap parameterisation allowing user-specified <i>hydro_frac_eff</i>	Y
<i>clwde_param</i>	Cloud liquid water effective diameter parameterisation	1	N	UV/VIS/IR hydrometeor simulations with <i>hydrotable</i> file	N
<i>icede_param</i>	Ice effective diameter parameterisation	1,2,3,4	N	UV/VIS/IR hydrometeor simulations with <i>hydrotable</i> file	N
<i>hydro_deff(:,:) size (nhydro, nlayers)</i>	Hydrometeor effective diameter	µm	N	Optional with UV/VIS/IR hydrometeor simulations with <i>hydrotable</i> file	Y
<i>surface_fraction(:) size (nsurfaces-1)</i>	Fractional coverage of surfaces 1, ..., <i>nsurfaces-1</i> . RTTOV automatically calculates fraction for the final surface.	0-1	N	Ignore for <i>nsurfaces=1</i> but must be specified for <i>nsurfaces>1</i> .	Y

<i>near_surface(isurf) % t2m</i> <i>1 <= isurf <= nsurfaces</i>	2m temperature	K	N	If <i>use_t2m</i> option is .true.	Y
<i>near_surface(isurf) % q2m</i>	2m water vapour	ppmv or kg/kg	N	If <i>use_q2m</i> option is .true.	Y
<i>near_surface(isurf) % wind_u10m,</i> <i>near_surface(isurf) % wind_v10m</i>	10m wind u, v components	m/s	N	Sea surface emissivity and BRDF models (except ISEM)	Y
<i>near_surface(isurf) % wind_fetch</i>	Wind fetch (length of water over which the wind has blown). Typical default: 100000 m	m	N	Sea surface BRDF model	Y
<i>skin(isurf) % surftype</i> <i>1 <= isurf <= nsurfaces</i>	Surface type (land = 0, sea = 1, sea-ice = 2)	0,1,2	Y	Emissivity and BRDF models and atlases	N
<i>skin(isurf) % watertype</i>	Water type (fresh = 0, ocean = 1)	0,1	N	Surface BRDF model and BRDF atlas	N
<i>skin(isurf) % t</i>	Surface skin temperature	K	N	Usually required, but not if <i>use_tskin_eff</i> option is true.	Y
<i>skin(isurf) % salinity</i>	Ocean salinity	Practical salinity unit	N	FASTEM-5/6, SURFEM-Ocean	Y
<i>skin(isurf) % foam_fraction</i>	Ocean foam fraction	0-1	N	FASTEM-5/6 if <i>use_foam_fraction</i> option .true.	Y
<i>skin(isurf) % snow_fraction</i>	Surface snow cover fraction	0-1	N	IR emissivity atlases	N
<i>skin(isurf) % fastem(1:5)</i>	FASTEM land/sea-ice parameters		N	FASTEM for land/sea-ice surface types	Y
<i>zenangle</i>	Satellite zenith angle	deg	Y		N
<i>azangle</i>	Satellite azimuth angle (0-360; measured clockwise, east=+90)	deg	N	Using FASTEM-5/6, SURFEM-Ocean, or solar option	N
<i>sunzenangle</i>	Solar zenith angle	deg	N	Solar option, NLTE option	N
<i>sunazangle</i>	Solar azimuth angle (0-360; measured clockwise, east=+90)	deg	N	Solar option	N
<i>latitude</i>	Latitude (-90 to +90)	deg	Y	For radiation path geometry and emissivity and BRDF atlases	N
<i>longitude</i>	Longitude (0-360)	deg	N	Emissivity and BRDF atlases	N
<i>elevation</i>	Surface elevation	km	Y	For radiation path geometry	N
<i>cfraction</i>	Cloud fraction for simple cloud	0-1	N	Simple cloud scheme only	Y
<i>ctp</i>	Cloud top pressure for simple cloud	hPa	N	Simple cloud scheme only (if <i>cfraction</i> > 0)	Y
<i>be</i>	Earth magnetic field strength	Gauss	N	Zeeman simulations	N
<i>cosbk</i>	Cosine of the angle between the Earth magnetic field and wave propagation direction		N	Zeeman simulations	N
<i>date(1:3)</i>	Date of the profile as year (e.g. 2024), month (1-12), and day (1-31)		N	Optional for solar calculations.	N
<i>time(1:3)</i>	Time of profile as hour, minute, second.		N	Not used by core RTTOV. Used by <i>rttov_calc_solar_angles</i> subroutine.	N

Table 7.2: Profile input variables.

7.4.1 Pressure levels and interpolation

The pressure half-levels should cover the entire atmosphere with an adequate resolution. Very coarse layering will reduce the accuracy of the simulations. Ideally the top-most half-level should be at or close to the top of the atmosphere and may be set to 0 hPa. At the very least, for accurate simulations the input profile must span the range of pressures over which

the weighting functions of the channels being simulated are significantly greater than zero. Generally, it is recommended to use the native vertical grid of the NWP model from which the profile data are taken.

RTTOV carries out the gas absorption optical depth calculation on a fixed set of pressure levels (usually 54 or 101 levels) defined by the optical depth coefficient file. RTTOV interpolates the input profiles onto the coefficient levels internally, computes the gas absorption optical depths, and interpolates the optical depths back to the user input pressure levels. All other calculations are carried out on the input pressure levels. Figure 7.3 illustrates the interpolation steps in RTTOV.

The internal RTTOV interpolator is enabled by setting **opts%interpolation%enable_interp** to true. This is set to true by default and should not usually be changed.

The interpolation of input profiles from user levels to coefficient levels is based on the pressure full-levels. As noted above, the full-levels for the user profile may be input explicitly, but otherwise (if all zero) they are computed by RTTOV as the mean of adjacent half-level pressures. The coefficient full-levels are always computed like this. The interpolation of profiles from user levels to coefficient levels assumes the profile variables are defined at the full-level pressures.

Where the top-most input pressure full-level is greater than (lies below) the top coefficient full-level, RTTOV extrapolates the input profile to the top coefficient full-level as part of the interpolation. This is done by comparing each variable (temperature, gas concentrations) in the top input layer to the minimum/maximum values used in training the optical depth coefficients in the corresponding coefficient layer. The variables are extrapolated upwards maintaining the same relative position between the minimum/maximum limits of the training profiles. This yields physically plausible extrapolated profiles and is done independently for each interpolated variable. RTTOV ensures that extrapolated values never exceed the minimum/maximum regression limits of the training profiles, and so extrapolated values never result in warnings, or the regression limits quality flag being set (see sections 7.4.3 and 7.8.3).

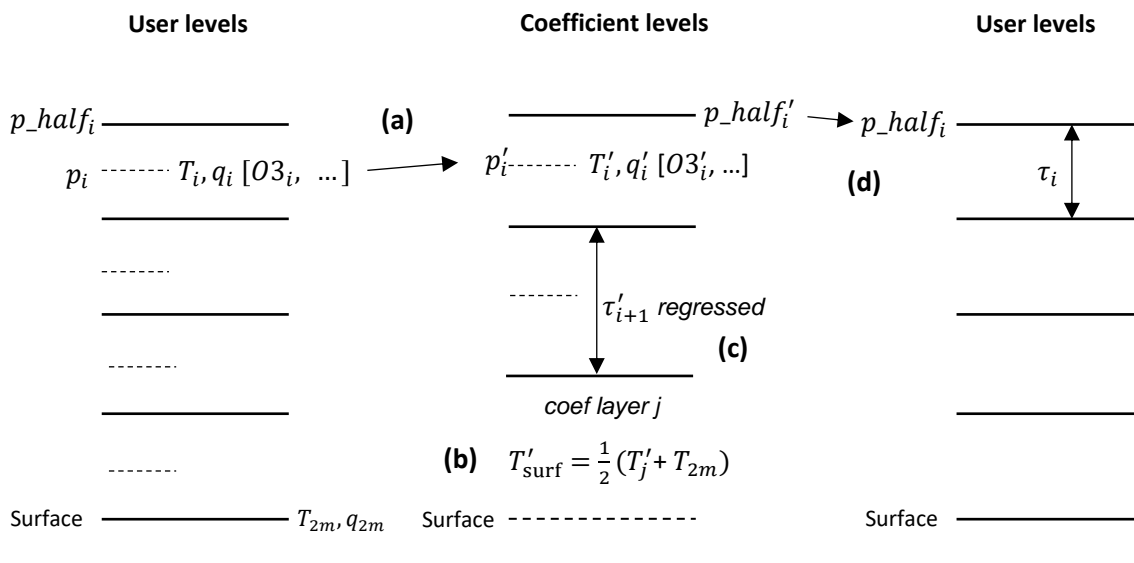


Figure 7.3: (a) Interpolation of input profiles of temperature and gas concentrations from to user levels to coefficient levels is based on the pressure full levels. Quantities on coefficient levels are denoted by primes ('). (b) For the coefficient layer containing the surface, the temperature and water vapour values incorporate the 2m T and q values if the relevant options are enabled (only temperature is shown). (c) Layer gas absorption optical depths are computed using the gas optical depth regression for the layers bound by the coefficient levels. (d) Interpolation of the layer optical depths back onto user levels is based on the pressure half-levels.

RTTOV implements several interpolation methods based on log-linear interpolation in pressure, and the interpolation method described in Rochon *et al.* (2007) and in the RTTOV v9 Science and Validation Report. An important feature of the Rochon interpolation scheme is that all input levels in an interpolation contribute to the interpolated profile with appropriate weight which means that Jacobians do not exhibit excessively large or small sensitivity to particular levels

purely as a result of the interpolation. This is particularly important for the adjoint and Jacobian of the input profile interpolation step (see (a) in Figure 7.3).

The interpolation modes are listed below in Table 7.3 and are described in more detail in Hocking (2014). That report shows results for previous versions of RTTOV, and these do not apply to RTTOV v14. Similar results for the current version of RTTOV can be found in the RTTOV v14 Science and Validation Report. For RTTOV v14, testing indicates that in general mode 5 yields the smallest interpolation errors of all the modes followed by mode 4, and that mode 4 gives smoother Jacobians than mode 5. Mode 4 is set as the default, and modes 4 or 5 are generally recommended. You may wish to conduct a study to find out which interpolation mode works best in your application. The interpolation mode is set in the `opts%interpolation%interp_mode` option.

Some calculations within RTTOV require values of temperature and water vapour concentration on the pressure half-levels. This includes the atmospheric Planck radiances used in solving the radiative transfer equation. It also includes the calculations of local radiation path geometry when Earth curvature is taken into account (when the `opts%rt_all%plane_parallel` option is false, the default) and also when the effect of atmospheric refraction is included (when `opts%rt_all%refraction` is true, the default, and the `plane_parallel` option is false). Temperature and water vapour values on half-levels are interpolated linearly in log pressure from the full-level values. The values in the first half-level (at the top of the atmosphere) are taken from the first full-level. The values in the bottom half-level (at the surface) are taken from the profile `near_surface%t2m` and `near_surface%q2m` variables if the `opts%rt_all%use_t2m` and `opts%rt_all%use_q2m` options are true respectively. If either option is false, the corresponding bottom half-level value is taken from the bottom full-level. Note that in this bottom half-level the values represent the atmospheric temperature and water vapour just above the surface. The radiative skin temperature is taken from the profile `skin%t` variable (or the `emis_refl%tskin_eff(:)` input – see section 8.3.9).

The RTTOV profile structure also includes an elevation variable (`profile%elevation`): this is used in the calculations which determine the local path angles of the radiation in each atmospheric layer (when the `plane_parallel` option is false). Note that the surface always lies on the bottom pressure half-level regardless of the value of the `elevation`.

interp mode	Profile interp (user->coef levels)	Optical depth interp (coef->user levels)	Description
interp_rochon (1)	Rochon	Rochon on optical depths	Original RTTOV interpolation method, Jacobians may show oscillations.
interp_loglinear (2)	Log-linear	Log-linear on optical depths	May be beneficial in execution time for direct-model calculations, but not suitable for TL/AD/K.
interp_rochon_loglinear (3)	Rochon	Log-linear on optical depths	Similar to mode 1, but with somewhat reduced oscillations.
interp_rochon_wfn (4)	Rochon	Rochon on weighting function	Default , no oscillations, smaller interpolation errors in general compared to modes 1-3, but most computationally expensive method.
interp_rochon_loglinear_wfn (5)	Rochon	Log-linear on weighting function	No oscillations, but Jacobians may show small “artefacts” due to interpolation, smallest interpolation errors in general, slightly faster than mode 4. Recommended as an alternative to mode 4.

Table 7.3: Interpolation options available in RTTOV. The constants corresponding to the integer values are in the `rttov_const` module (Annex K).

7.4.2 Trace gases

Section 3 gives information on the variable gases available with each type of coefficient file. If you are supplying profiles for an optional trace gas you must set the corresponding options flag to true: e.g., if you supply ozone data, then you must set `opts%rt_all%o3_data` to true. If the flag is true, you must supply a valid input profile. If a coefficient file with trace gas coefficients is used and suitable trace gas profile concentrations are not available, then you must set the corresponding

data flag (e.g., **o3_data**) to false. In this case RTTOV uses a fixed background trace gas profile for the calculation. This applies to all gases except water vapour which is always mandatory. Section 7.2 provides additional information on this.

The “_7gas” v13 predictor optical depth coefficient files (see section 3) include variable SO₂. For the purposes of running RTTOV, SO₂ is treated in the same way as other gases. You should set the **opts%rt_all%so2_data** flag to true to indicate you are supplying SO₂ profiles and the input profile data are specified in **profiles(:)%so2(:)**. The SO₂ optical depth prediction has been trained using a selection of profiles which cover both “clean” (low-SO₂) and “volcanic” (high-SO₂) atmospheres (see the RTTOV v12 Science and Validation Report). If you do not supply an SO₂ profile with an SO₂-enabled coefficient file, then RTTOV uses a “clean” background SO₂ profile. This is different to other gases where the mean profile of the training dataset is used.

The v13 predictor SO₂-enabled coefficients show larger errors (as measured by comparisons to the line-by-line model used to train RTTOV) in the spectral regions where SO₂ is active over the whole training set. However, when comparing over an independent profile dataset with fixed (background) SO₂ the accuracy compared to the LBL model is very similar to an equivalent coefficient file enabling all variable gases *except* SO₂ (see the RTTOV v13 Science and Validation Report). In other words, it is the cases with high SO₂ concentrations that have larger errors.

RTTOV has a subroutine **rttov_scale_ref_gas_prof** which provides a method of supplying scaled copies of the RTTOV background profiles for the optional gases. This could be useful, for example, if you do not have a CO₂ profile, but wish to specify a lower concentration corresponding to a specific time in the past: this subroutine allows you to specify the maximum CO₂ ppmv value and the RTTOV background profile will be scaled to have this maximum value. In order to use this subroutine, you must set the relevant gas flags (e.g. **o3_data** or **co2_data**) to true for the gases you wish to include in the simulation. You should then populate the **profiles(:)** structure with all of the data (**p(:)**, **t(:)**, **q(:)**, etc) except for the trace gas(es) for which you intend to use the scaled background profiles. Just before the call to RTTOV, you call **rttov_scale_ref_gas_prof** passing the coefficients structure, the **profiles(:)** array, and specifying the relevant arguments for scaling the gas(es) of interest. The subroutine then populates those gas arrays in the **profiles(:)** structures. Annex I gives more information about the subroutine.

The units of the gas concentrations in the input profile are determined by the **profiles(:)%gas_units** variable: all profiles passed into RTTOV in a single call must use the same gas units. The units apply to all trace gas input profiles and the 2m (near-surface) water vapour variable. Table 7.4 lists the available options for gas units.

profiles(:)%gas_units	Description
gas_units_ppmv (2)	ppmv over moist air
gas_units_kg_per_kg (1)	kg/kg over moist air (default)
gas_units_ppmvdry (0)	ppmv over dry air: this is intended primarily for coefficient generation and testing but is a valid input option.

Table 7.4: Options for gas units. The constants corresponding to the integer values are in the *rttov_const* module (Annex K).

7.4.3 Gas optical depth regression limits

The range of temperatures, and water vapour and gas concentrations over which the optical depth computations are valid depends on the training datasets which were used. This is defined in the coefficient file and for RTTOV v14 is mainly based on the 91-level 83 diverse profile dataset from ECMWF analyses for temperature, water vapour and ozone (<https://nwp-saf.eumetsat.int/site/software/atmospheric-profile-data/>). For other gases a range of profile datasets were used based on models and measurements. The limits for temperature, water vapour and ozone are given in Table 3.1. These “regression limits” are derived from the strict profile dataset minimum/maximum envelopes by applying a stretching factor (+/-10% for temperature max/min respectively, and +/-20% for each gas max/min). The RTTOV coefficient files contain the *strict* min/max envelopes: the stretched limits that are used within RTTOV are calculated when the coefficients are read in. More details on the profile datasets used for the different gases can be found in Matricardi (2008). The training profiles used for RTTOV coefficients are intended to be valid over the whole satellite era from 1970 and into the 2020s. They are described in the RTTOV v12 Science and Validation Report. Sulphur dioxide was added as an optional variable gas later, and the training profiles used for this are also described in the RTTOV v12 Science and Validation Report. Note that RTTOV v13-compatible (and earlier) coefficient files contain the profile envelope data on the coefficient pressure levels which correspond to RTTOV half-levels. When these limits are read in from the coefficient file, values on adjacent pairs of half-levels are averaged to obtain the limits that are applied to interpolated input profiles on coefficient full-levels (layers).

RTTOV compares the input profiles to the regression limits and prints warnings when the regression limits are exceeded, but the simulations are performed regardless. RTTOV indicates if the regression limits are exceeded for some quantity for all channels for a given profile in the **radiance%quality** array by setting the **qflag_reg_limits** flag (section 7.8.3). The printed warnings about profiles exceeding the regression limits can be suppressed by setting **opts%config%verbose** to false (default: true). Input profiles can be clipped to the corresponding regression limit when a limit is exceeded by setting **opts%config%apply_reg_limits** to true (default: false). This also prevents printed warnings being output. When **opts%config%apply_reg_limits** is true the profile values are only clipped to the regression limits for the purposes of the gas absorption optical depth calculation. The source function for the radiative transfer equation and other calculations are always based on the unmodified input profiles. When warnings are printed out for gases, the values are in ppmv over dry air regardless of the setting of the **gas_units** variable. You may wish to perform your own sensitivity study to decide what setting of the **apply_reg_limits** option is best for your application.

PC-RTTOV (section 8.6) uses its own set of regression limits for the gas optical depth calculations. In this case the checking/flagging works the same way as for standard RTTOV simulations, including the **verbose** option. Note however that when the **apply_reg_limits** option is true only aerosol or hydrometeor profiles are clipped to the training limits for PC-RTTOV simulations. The temperature and gas concentrations are never modified.

As described in section 7.7, the **rttov_user_check_profile** subroutine can be used to test profiles against both the hard limits and the regression limits before calling RTTOV. This allows out-of-bounds profiles to be rejected without running full simulations if desired. This routine has an optional **quality** argument which is analogous to a single element of the **radiance%quality** array. See Annex I for details of this subroutine.

7.4.4 Cloud liquid water (CLW) for MW simulations

For MW simulations it is possible to supply a cloud liquid water (CLW) profile in the **profiles(:)%clw(:)** array which is treated as a purely absorbing medium. Note that this is for non-scattering simulations only: the **clw(:)** profile member array is ignored for hydrometeor scattering simulations (see section 8.4). If you wish to include CLW in the “clear-sky” simulations, you must set **opts%clw_absorption%clw_data** to true. By default, RTTOV ignores any CLW content in layers above 322hPa. This limit is specified in the **opts%clw_absorption%clw_cloud_top** option allowing you to modify it if you wish. The Jacobian model generates non-zero CLW Jacobians even for layers with zero CLW content, but only those layers lying below the pressure limit specified in **clw_cloud_top**. RTTOV allows a choice of liquid water permittivity parameterisations. These are selected using the **opts%clw_absorption%permittivity_param** option: see Table 7.5. These same options are also available for generating MW hydrotable files. Rosenkranz (2015) is the recommended option and is the one used for the cloud liquid water and rain hydrometeors in the NWP SAF MW hydrotable files.

permittivity_param	Description
clw_perm_liebe (1)	Liebe (1989)
clw_perm_rosenkranz (2)	Rosenkranz (2015), supports frequencies up to 1000GHz, can increase run-time by ~50% (default , recommended)
clw_perm_tkc (3)	Turner, Kneifel, Cadeddu or TKC (2016)

Table 7.5: Available settings for **opts%clw_absorption%permittivity_param**. The constants corresponding to the integer values are in the **rttov_const** module (Annex K).

7.5 Specifying the channels to simulate

As described above, the **profiles(:)** array contains a list of the profiles for which to calculate radiances. RTTOV offers the flexibility to calculate radiances for a different set of channels for each profile in **profiles(:)**, though often in practice radiances for the same set of channels will be calculated for every profile. You should allocate a **chanprof(:)** array (derived type **rttov_chanprof**): this defines which channels are simulated for each profile. The size is the total number of channels to simulate over all profiles in each call to RTTOV. Each element in the array has two members: **chanprof(j)%prof** (the profile index), and **chanprof(j)%chan** (the channel index), where **j** runs from 1 up to the total number of channels to simulate per call to RTTOV. Table 7.6 illustrates how **chanprof(:)** should be set up for three different sensors and for 2 profiles per RTTOV call: all channels for the first profile are specified, followed by all channels for the second profile, and so on. Note that for PC-RTTOV Principal Components calculations the channels you *must*

simulate are determined by `opts%pcrttov%pc_band` and `opts%pcrttov%pc_reg_set` as described in section 8.6 and in this case you *must* simulate the same channels for every profile.

The RTTOV channel numbering *always begins at 1* for any coefficient file regardless of the original instrument channel numbering. This means that with IR-only and UV/visible/IR coefficient files for the same instrument, RTTOV may use different indices to refer to the same channel (see for example SEVIRI in Table 2.2). If a subset of n possibly non-consecutive channels is read from the coefficient file (by supplying the `channels(:)` argument to `rttov_read_coefs`), then this subset of channels is identified in `chanprof(:)%chan` by the numbers 1 to n rather than by their original channel index in the coefficient file. For example, if coefficients for only channels 3 and 5 of a sensor with five channels are read from the coefficient file, then `chanprof(1:2)%chan` should be `(/1, 2/)` corresponding to the channels 3 and 5 respectively. Similarly, if the `rttov_conv_coef.exe` executable (Annex A) is used to create a coefficient file containing a subset of n possibly non-consecutive instrument channels, then these will be identified by the indices 1 to n when reading this new coefficient file using `rttov_read_coefs`.

If PC-RTTOV computations are being carried out (section 8.6) you may request radiances to be reconstructed from the PC scores by setting `opts%pcrttov%rec_rad` to true. In this case the array `channels_rec(:)` must be supplied containing the indices of the channels for which reconstructed radiances are required. The reconstructed radiance channel numbers are again always counted from 1. If you passed the `channels_rec(:)` argument to `rttov_read_coefs` to specify some subset of n possibly non-consecutive channels for which radiances may be reconstructed, then this subset of channels is identified in the `channels_rec(:)` argument to `rttov_direct` (and TL/AD/K) by the numbers 1 to n just as for `chanprof(:)%chan`. If PC-RTTOV is not being used or reconstructed radiances are not required the `channels_rec(:)` argument should be omitted.

Input structure	HIRS (2 profiles/call)	SSM/I (2 profiles/call)	AMSU-B (2 profiles/call)
<code>size(chanprof)</code>	38	14	10
<code>size(profiles)</code>	2	2	2
<code>chanprof(:) % chan</code>	1,2,3 ...,19,1,2,3...,19	1,2,3,4,5,6,7,1,2,3,4,5,6,7	1,2,3,4,5,1,2,3,4,5
<code>chanprof(:) % prof</code>	1,1,1,...,1,2,2,2...,2	1,1,1,1,1,1,2,2,2,2,2,2	1,1,1,1,1,2,2,2,2,2

Table 7.6: Examples of `chanprof(:)` inputs for RTTOV.

7.6 Specifying surface emissivity and reflectance

Surface emissivity and reflectance inputs and outputs are provided via the `emis_refl(:)` array argument (derived type `rttov_emis_refl`) to `rttov_direct`. This structure and the related inputs/outputs are covered in detail in section 8.3.

7.7 Checking RTTOV inputs

By default, RTTOV carries out a range of internal checks on the inputs provided to catch problematic values. Several user-level subroutines are provided which allow you to run checks on inputs before calling RTTOV: you may prefer to use these and turn off the internal checking. This section describes these capabilities.

RTTOV *always* carries out checks to ensure that the simulations have not been configured in such a way that will cause errors. This includes checking for mutually incompatible option settings, invalid option values, inconsistencies between options and the coefficient files that have been read in, presence of optional arguments that are mandatory given the selected options, and invalid specifications in the `chanprof` input argument. If any check fails, RTTOV returns immediately with a non-zero error status. This checking cannot be turned off.

A user-level subroutine, `rttov_user_check_options`, is provided for “debugging” your simulations. This performs most of the same checks that are done internally for strictly illegal settings. It can also check for “harmless-but-dubious” settings that will not cause any problems in simulations but might indicate a misunderstanding about how RTTOV works (for example, setting the `o3_data` option to true with an optical depth coefficient file that does not support variable ozone). These latter checks can optionally be turned off. The interface to this routine is given in Annex I.

By default, RTTOV carries out checks on the input `profiles` (section 7.4) and `emis_refl` (section 8.3) arguments, and, if present, the `aer_opt_param` and `hydro_opt_param` arguments (section 8.4.11). These checks test for unphysical values and for values that lie outside the hard limits that RTTOV imposes. Inputs falling outside these bounds can result in

numerical failures within RTTOV or otherwise in unreliable outputs and so if they occur, the simulation is aborted and a non-zero error status is returned.

The hard limits for **profile** variables are given in table 7.7 below.

A user-level subroutine, **rttov_user_check_profile**, can be used to carry out the same checks on one input profile at a time. This allows you to screen out bad profiles before calling RTTOV. RTTOV generally prints error messages for all fatal errors, but this output can be turned off in **rttov_user_check_profile** so that profile screening can be carried out silently. If you use this subroutine, then you don't need to run the internal profile checking, so you can set the **opts%config%check_profiles** option to false. This option also disables the internal checking of **emis_refl** and **aer/hydro_opt_param** structures, and so user-level subroutines **rttov_user_check_emis_refl** and **rttov_user_check_opt_param** can be used before calling RTTOV to check for invalid inputs in these structures if you wish. The interfaces for these user-level subroutines can be found in Annex I.

The **rttov_user_check_profile** subroutine can also be used to check the input temperature and gas concentration profiles against the RTTOV gas optical depth regression limits (section 7.4.3), and input aerosol profiles against the PC-RTTOV regression limits (section 8.6). Full details of this subroutine are given in Annex I.

Variable	Minimum	Maximum
Satellite zenith angle	0	75° for v7/v8 predictors 85.3° for v9/v13 predictors
Pressure of bottom half-level (i.e., surface pressure)	400 hPa	1100 hPa
Temperature (inc. 2m T and Tskin)	90 K	1250 K for Tskin over land only 400 K all other cases
Water vapour (inc. 2m q)	1E-11 ppmv / 6.2E-18 kg/kg	60000 ppmv / 3.7E-1 kg/kg
O ₃	1E-11 ppmv / 1.7E-17 kg/kg	1000 ppmv / 1.7E-3 kg/kg
CO ₂	1E-11 ppmv / 1.5E-17 kg/kg	1000 ppmv / 1.5E-3 kg/kg
CO	1E-11 ppmv / 9.7E-18 kg/kg	10 ppmv / 9.7E-6 kg/kg
N ₂ O	1E-11 ppmv / 1.5E-17 kg/kg	10 ppmv / 1.5E-5 kg/kg
CH ₄	1E-11 ppmv / 5.5E-18 kg/kg	50 ppmv / 2.8E-5 kg/kg
SO ₂	1E-11 ppmv / 2.2E-17 kg/kg	1000 ppmv / 2.2E-3 kg/kg
Cloud liquid water (clear-sky MW only)	0 kg/kg	1 kg/kg
10m wind speed = (wind_u10m ² + wind_v10m ²) ^{1/2}	0 m/s	100 m/s
Cloud top pressure (simple cloud; only applies if cfraction > 0)	50 hPa	1100 hPa
Magnetic field Be (Zeeman only)	0.2 Gauss	0.7 Gauss

Table 7.7: Hard limits for input profiles variables as defined in **rttov_const** (Annex K). Input values must not lie outside the interval [minimum, maximum]. NB ppmv and kg/kg values for gases in this table are over dry air.



7.8 Outputs from RTTOV v14

The syntax for the call to **rttov_direct** is given again, this time with the output arguments in bold:

```
call rttov_direct(errorstatus, opts, coefs, chanprof, profiles, emis_refl,
                transmission, radiance, radiance2,
                aer_opt_param, hydro_opt_param, refl_cloud_top,
                reflectivity, emis_retrieval_terms, diag_output,
                pccomp, channels_rec, traj, traj_dyn, traj_sta)
```

Output arguments from the RTTOV direct model:

- **errorstatus** - return status indicating whether call was successful or not, see below
- **emis_refl** - output emissivity and reflectance data, section 8.3
- **transmission** - computed atmospheric transmittances, section 7.8.1
- **radiance** - computed radiances, brightness temperatures, and reflectances, section 7.8.2
- **radiance2** - secondary radiance outputs (direct model only), section 7.8.2

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

- reflectivity - radar reflectivity outputs, section 8.4.4
- emis_retrieval_terms - data enabling dynamic emissivity retrievals, section 8.4.12
- diag_output - additional per-profile outputs, section 7.8.4
- pccomp - output PC scores and reconstructed radiances from PC-RTTOV, section 8.6

errorstatus is an integer error return code. If the value of **errorstatus** is non-zero, a fatal error occurred during processing (more generally success is indicated by a return code **errorstatus_success** and failure by **errorstatus_fatal**, both of which are contained in the **rttov_const** module, see Annex K).

Emissivity and reflectance outputs in the **emis_refl** argument are described in section 8.3.

7.8.1 Transmittance outputs

The **transmission** structure of type **rttov_transmission** contains separate transmittances for the thermally emitted and solar calculations. This is because the solar calculations are derived from different optical depth predictor calculations. For mixed thermal+solar channels the thermal and solar transmittances for the surface-satellite path will therefore not be exactly equal, and in the case of channels using Planck-weighted coefficients (see section 3) may be significantly different.

The transmittance structure members are described in Annex J. The primary transmittance outputs are for the clear column and include gas absorption and aerosols (if present), but never hydrometeors. For hydrometeor scattering simulations, additional hydrometeor-only transmittances are output excluding gas and aerosols.

If you only require output transmittances, you can set **opts%rt_all%transmittances_only** to true (for the direct model only): in this case no radiances or surface emissivities or reflectances are calculated which results in faster simulations. Note that you must still provide the **emis_refl** and **radiance** arguments to **rttov_direct**.

7.8.2 Radiance outputs

The radiance structures are described in Annex J where you can find tables describing the member arrays. The primary radiance outputs are in the **radiance** argument of type **rttov_radiance**. This includes radiances, brightness temperatures (for thermal channels), and reflectances (for solar channels, see section 8.1). There are “clear” members which contain clear-sky radiances (**clear**), brightness temperatures (**bt_clear**), and reflectances (**refl_clear**). For aerosol scattering simulations (section 8.4), these also include the effects of aerosols. The **total** radiance member and corresponding **bt** and **refl** members give the satellite-seen radiance incorporating any cloud or hydrometeors, either via the simple cloud scheme (section 8.2) or full hydrometeor scattering (section 8.4) (and including aerosols if present). The **cloudy** radiance member contains the cloud-affected radiance assuming there was no clear column. For hydrometeor scattering two-column overlap schemes (section 8.4.3) and the simple cloud scheme (section 8.2), this is the same as the radiance of the cloudy column. For aerosol simulations without hydrometeors, the **cloudy** radiance is always zero (the simple cloud scheme is not applied for aerosol simulations).

The **overcast** radiance output represents radiances assuming a black, opaque cloud at each pressure half-level bounding the bottom of each layer. This is also computed for solar channels, optionally using the **relf_cloud_top** input argument to **rttov_direct**. Section 8.2 gives more information on this. This is calculated for clear-sky (non-scattering) simulations. For thermal channels it is also calculated when the Chou-scaling thermal solver (section 8.4.2) is selected for scattering simulations (clear column only, including aerosols if present). The **opts%config%bt_overcast_calc** option can be set to true to enable additional output of brightness temperature equivalents of the overcast radiances in **radiance%bt_overcast**. Overcast radiances are not calculated for PC-RTTOV simulations (section 8.6).

If the optional **radiance2** argument of type **rttov_radiance2** is supplied, it is populated with additional thermal radiance outputs (solar radiation is ignored). This is not populated for PC-RTTOV simulations (section 8.6) or for scattering simulations that do not use the Chou-scaling thermal solver (section 8.4.2). The **radiance2** quantities have no TL/AD/K equivalents and are not active variables in these models.

A general point to note is that RTTOV is optimised for simulating top-of-atmosphere radiances. This means that in the gas absorption optical depth training, layers that are optically deep in the atmosphere (i.e., to which the satellite is not sensitive) are omitted. In practice this means that when considering a layer that is optically deep in the atmosphere (i.e.,

where the atmosphere above the layer is opaque), any radiances or optical depths/transmittances computed for that layer should not be assumed to be correct (noting that they do not affect the top-of-atmosphere radiances).

7.8.3 Quality flags

The **radiance%quality** output array is a bit mask which is used to flag cases where an input value has exceeded an upper or lower bound that is applied for a parameterisation within RTTOV. If no bounds were exceeded the quality value corresponding to a given output radiance is zero. However, one or more bits may be set indicating specific warnings. The bit positions are contained in the **rttov_const** module and are listed in Table 7.8.

Bit position	Description
qflag_reg_limits	Gas regression limits exceeded in optical depth computation (section 7.4.3).
qflag_pc_aer_reg_limits	PC-RTTOV aerosol regression limits exceeded (section 8.6).
qflag_pc_hydro_reg_limits	PC-RTTOV hydrometeor regression limits exceeded (section 8.6).
qflag_emis_limits	Sea surface emissivity model limits exceeded (section 8.3.1).
qflag_hydro_deff_limits	Hydrometeor Deff limits exceeded (section 8.4.7).
qflag_hydro_t_limits	Hydrometeor temperature limits exceeded (section 8.4.7).
qflag_hydro_conc_limits	Hydrometeor concentration limits exceeded (section 8.4.7).
qflag_delta_edd_ext_limits	Delta-Eddington thermal solver extinction limits exceeded (section 8.4.2).
qflag_mfasis_nn_limits	MFASIS neural network parameter limit exceeded (section 8.4.2).

Table 7.8: Bit positions (contained in the *rttov_const* module, see Annex K) for RTTOV quality flags. It is strongly recommended to use the constants rather than the explicit bit numbers when checking these flags.

In Fortran you can test for a specific bit using the intrinsic BTEST function. For example, to check whether the gas optical depth regression limits were exceeded for channel *i* you can use:

```
IF (BTEST(radiance%quality(i), qflag_reg_limits)) THEN...
```

There is a subroutine **rttov_print_radiance_quality** (see Annex I) which prints out a human-readable interpretation of any warning flags set in a **radiance%quality** array element which may be useful for debugging. It is recommended to read the relevant section of the user guide for each warning to understand the implications of the corresponding bounds being exceeded. It does not necessarily mean that the computed radiance should not be trusted.



7.8.4 Additional outputs

The **diag_output** optional argument of type **rttov_diagnostic_output** can be used to obtain additional per-profile information from the simulations. Currently this includes the geometric heights (altitudes) of the pressure half-levels and full-levels which may be of use, for example, with radar simulations (see section 8.4.4). For hydrometeor scattering simulations the **hydro_frac_eff** output contains the effective hydrometeor fraction for two-column overlap schemes (section 8.4.3), or more generally is equal to one minus the weight of the clear column.

The **emis_retrieval_terms** optional argument of type **rttov_emis_retrieval_terms** can be supplied to obtain transmittances, radiances, and other data that can be used for dynamic emissivity retrievals. See section 8.4.12 for more information.

For PC-RTTOV simulations (section 8.6), the **pccomp** structure of derived type **rttov_pccomp** is mandatory. This structure contains the computed PC scores, and if requested the reconstructed radiances. The **pccomp** argument is not required if PC simulations are not being performed.

The members of all these structures are given in Annex J.

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

7.9 Calling the tangent linear (TL), adjoint (AD) and Jacobian (K) models

The tangent linear (TL), adjoint (AD) and Jacobian (K) models are typically used in assimilation and retrieval applications. They provide information about the gradient of the forward model given an input profile. The TL, AD and K models share some features:

- All three models take the same input atmospheric profile and surface parameters as the direct model.
- All three models output the direct model radiances, BTs and reflectances.
- As for the direct model all three models may be run using multiple threads via OpenMP (see section 7.10).
- Not all profile variables are “active” in the TL, AD and K models: Table 7.2 indicates which variables can be perturbed in the TL and hence which may be non-zero in the Jacobian and AD output.

RTTOV TL model

The RTTOV TL model (see Annex H for the interface) takes as input a **profiles_tl(:)** profile structure containing a set of perturbations to the input profile. The TL model calculates the linearisation of the direct model evaluated for the given input profile and it outputs the change in satellite seen radiances that result from the given profile perturbation for this linearisation.

By default the pressure levels (**p_half(:)** and, if specified, **p(:)**) are not active variables in the TL/AD/K so that perturbations in these members of **profiles_tl(:)** will be ignored. Setting the **opts%interpolation%pressure_gradients** option to true enables these as active TL/AD/K variables. The input **profiles_tl(:)%p(:)** is only used if the input **profiles(:)%p(:)** were specified.

The **profiles_tl(:)** array should be allocated to be the same size as the input **profiles(:)** array. As an example the subroutine **rttov_make_profile_inc** in the **src/test/rttov_test_make_inc_mod.F90** module is used to generate profile perturbations for the RTTOV test suite.

The aerosol/hydrometeor explicit optical properties (section 8.4.11) are optionally enabled as active variables in the TL model. You can supply perturbations to the optical properties via the **aer/hydro_opt_param_tl** structure. The Legendre coefficients (**lcoef**) and phase function (**pha**) are *not* active TL variables so should not be assigned. The **aer/hydro_opt_param_tl** arguments to the tangent linear model are optional: if omitted, the explicit optical properties are treated as static variables.

Surface emissivity and reflectance TL inputs and outputs are described in detail in section 8.3.11. The TL inputs (where relevant) are provided in the **emis_refl_tl** argument via the **emis_in(:)**, **brdf_in(:)**, and **diffuse_refl_in(:)** members, and the output TL values (which may be the same as the inputs or may be computed by RTTOV depending on **calc_emis(:)**, etc) are in the **emis_out(:)**, **brdf_out(:)**, and **diffuse_refl_out(:)** members. You can also supply input TL perturbations for the **specularity(:)** member when using the Lambertian surface option (section 8.3.8), and input TL perturbations for the **tskin_eff(:)** member when using per-channel effective skin temperature inputs (section 8.3.9).



The output radiance, brightness temperature and reflectance perturbations are contained in the **radiance_tl** structure. For PC-RTTOV (section 8.6) the output perturbations in PC scores and optionally reconstructed radiances are in the **pccomp_tl** structure. For hydrometeor scattering radar simulations (section 8.4.4), the output reflectivity perturbations are contained in the **reflectivity_tl** structure.

RTTOV AD and K models

The file **src/test/example_k.F90** provides an example of calling the RTTOV K model for a clear-sky simulation.

The RTTOV AD model (see Annex H for the interface) takes as input the gradient of a scalar function (e.g. a cost function) with respect to the satellite seen radiance (or BT or reflectance) in **radiance_ad** and it outputs the gradient of the same scalar function with respect to the profile variables in **profiles_ad(:)**. In this case the **profiles_ad(:)** array must be allocated to be the same size as the input **profiles(:)** array.

The RTTOV K model (see Annex H for the interface) takes as input radiance (or BT or reflectance) perturbations in **radiance_k** and it outputs the gradient of each forward model radiance (or BT or reflectance) with respect to each input profile variable evaluated for the given input profile in **profiles_k(:)**. The gradients are scaled by the perturbations in

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

radiance_k: typically, the input perturbations are set to a value of 1 in radiance or BT or reflectance units for each channel. For the K model, **profiles_k(:)** must be allocated to be the same size as the **chanprof(:)** array: each element of the array contains the gradient of the forward model for the corresponding channel.

Aside from the size of the **profiles_ad/k(:)** arrays the AD and K models are very similar. For thermal channels (those with wavelength > 3µm) the input perturbations in **radiance_ad/k** may be supplied either in radiance or BT: if the **opts%config%adk_bt** flag is false the input perturbations must be in **radiance_ad/k%total(:)** (in units of radiance), otherwise the input perturbations must be specified in **radiance_ad/k%bt(:)** (in Kelvin). For solar-only channels (those with wavelength < 3µm) the input perturbation may be supplied in either radiance or reflectance: if the **opts%config%adk_refl** flag is false the input perturbations must be in **radiance_ad/k%total(:)** (in units of radiance), otherwise the input perturbations must be specified in **radiance_ad/k%refl(:)** (no units). Note that for each channel, RTTOV takes AD/K inputs from only one of **total(:)**, **bt(:)**, and **refl(:)** depending on the settings of the **adk_bt** and **adk_refl** options (both true by default). Non-zero values of the other AD/K inputs are ignored.

In general, all AD/K input and output structures (aside from the input radiance/BT/reflectance perturbations) should be initialised to zero before calling the AD/K models. It is important to remember to reinitialise these structures before each call when making multiple calls to the AD/K subroutines. The following structures and arrays should always be set to zero:

- **profiles_ad/k** structure (use **rttov_init_profiles** subroutine)
- **emis_refl_ad/k** structure (use **rttov_init_emis_refl** subroutine)
- **transmission_ad/k** structure (use **rttov_init_transmission** subroutine)
- **radiance_ad/k** structure (use **rttov_init_radiance** subroutine **before** specifying the perturbations in **total(:)** and/or **bt(:)**; for PC-RTTOV **all** members of the structure should be set to zero – see below)

See Annex D for all **rttov_init_*** subroutine interfaces for initialising RTTOV structures to zero.

By default the pressure levels (**p_half(:)** and, if specified, **p(:)**) are not active variables in the TL/AD/K so that these members of **profiles_ad/k(:)** will be zero. Setting the **opts%interpolation%pressure_gradients** option to true enables these as active TL/AD/K variables, but this does increase the run-time, especially for the K model. The **profiles_ad/k(:)%p(:)** outputs are only non-zero if the input **profiles(:)%p(:)** were specified.



The aerosol/hydrometeor explicit optical properties (section 8.4.11) are optionally enabled as active variables in the AD and K models. When using these arguments, you should ensure the **aer/cld_opt_param_ad/k** structures are zeroed using **rttov_init_opt_param** subroutine. The Legendre coefficients (**lcoef**) and phase function (**pha**) are *not* active AD/K variables. The **aer/hydro_opt_param_ad/k** arguments to the AD/K models are optional: if omitted, the explicit optical properties are treated as static variables.

Surface emissivity and reflectance AD/K outputs are described in detail in section 8.3.11. The AD/K outputs are stored in the **emis_refl_ad/k** arguments via the **emis_in(:)**, **brdf_in(:)**, and **diffuse_refl_in(:)** members. For the Lambertian surface option, the **specularity(:)** member contains AD/K outputs (section 8.3.8), and similarly for the **tskin_eff(:)** member when using per-channel effective skin temperature inputs (section 8.3.9).

For hydrometeor scattering radar simulations, you can supply input AD perturbations in either **reflectivity_ad%zef(:,:)** or **reflectivity_ad%azef(:,:)** instead of, or alongside, the radiance/BT AD perturbations, depending on your application. The use of the K model with the radar solver is discussed in detail in section 8.4.4. Before calling AD/K simulations with the radar solver, all non-perturbed members of **reflectivity_ad/k** should be initialised to zero.

For PC-RTTOV (section 8.6), the input perturbations must be specified in the **pccomp_ad/k** structure: the member to which the perturbation applies depends on the setting of **opts%perrttov%rec_rad** and **opts%config%adk_bt**. If reconstructed radiances are not required (**opts%perrttov%rec_rad** is false), then the input PC score perturbation should be specified in **pccomp_ad/k%total_pccores**. If reconstructed radiances are required, the input radiance/BT perturbation should be specified in **pccomp_ad/k%total_pccomp** if **opts%config%adk_bt** is false, or **pccomp_ad/k%bt_pccomp** if **opts%config%adk_bt** is true. Before calling the PC-RTTOV AD/K models, **all** elements of the **radiance_ad/k** structure and all non-perturbed members of the **pccomp_ad/k** structure should be initialised to zero.

The PC-RTTOV AD and K models differ slightly in the output: for the AD model the output adjoint is in the **profiles_ad(:)** array as for standard RTTOV. For the K model, **profiles_k(:)** contains the Jacobians for the PC predictor

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

channel set. These are **always** in terms of radiances (not brightness temperatures), regardless of the setting of **opts%config%adk_bt**. If **opts%pcrttov%rec_rad** is false the PC Jacobians are output in the **profiles_k_pc(:)** array. This must have size equal to the number of PC scores multiplied by the number of profiles. These are gradients of PC scores with respect to the profile variables and in this case the setting of **opts%config%adk_bt** has no impact on the units of the Jacobians. Alternatively if **opts%pcrttov%rec_rad** is true the PC Jacobians are output in the **profiles_k_rec(:)** array which must have size equal to the number of reconstructed radiance channels multiplied by the number of profiles. In this case the setting of **opts%config%adk_bt** determines the units of the output (radiances or brightness temperatures). Note that only one of **profiles_k_pc(:)** or **profiles_k_rec(:)** should be supplied to **rttov_k** depending on the setting of **opts%pcrttov%rec_rad**.

7.10 Multi-threaded execution

If RTTOV is compiled with OpenMP (see section 5.3) then substantial gains in execution time can be made by calling RTTOV through the parallel interface. This is of relevance to users who are not implementing parallelism for RTTOV simulations themselves (common in NWP systems), but are instead, for example, running RTTOV on a workstation with multiple cores. The interfaces to the parallel subroutines are almost identical to **rttov_direct**, **rttov_tl**, **rttov_ad** and **rttov_k**: the subroutines are named **rttov_parallel_direct**, **rttov_parallel_tl** and so on. The only difference is a final optional parameter named **nthreads** which specifies the number of threads to use.

Each thread may be assigned simulations for multiple channels (possibly across multiple profiles), but the smallest unit of computation for a single thread is a simulation for one channel for one profile. Therefore, to make use of N threads, you must be simulating at least N individual channel radiances (for one or more profiles) and to obtain optimal performance you should usually be simulating many channels and/or profiles with each call to the parallel interface. For PC-RTTOV each thread is assigned at least one *profile* so you must call the parallel interface for at least N profiles to make use of N threads for PC-RTTOV simulations. The same applies to polarimetric instruments such as Windsat and COWVR.

7.11 RTTOV performance

This section provides a rough guide to the relative cost of various RTTOV options and simulation types. It is difficult to be precise as execution times vary greatly depending on the compiler, compilation flags, computer architecture, the combination of RTTOV options specified at run-time, and the configuration of the input profile. Some measurements of execution time (in particular, comparisons against the previous version of RTTOV) can be found in the RTTOV v14 Performance Test Logs available on the RTTOV web site.

General recommendations

As noted elsewhere in this user guide, unless you are implementing parallel execution of RTTOV simulations yourself, it is recommended to make use of multiple threads in RTTOV via OpenMP (section 7.10). Otherwise, you may see substantial performance benefits by allocating the “trajectory” structures outside of RTTOV (section 7.3.2).



Number of profiles per call

It is not possible to give a fixed recommendation, but it is common to see performance improvements by passing multiple profiles to RTTOV per call. In many cases one can find an optimum number for best performance for a particular simulation type on a given system. For performance critical applications it is recommended to carry out experimentation to determine this. MFASIS-NN (section 8.4.2) has been particularly optimised for vector architectures and it is recommended to pass in many profiles in each call.

Coefficients, trace gases, solar radiation

Simulations with coefficient files supporting more optional trace gases are slower than those with fewer optional gases. Simulations using v13 predictor coefficients are typically up to ~10% slower than the equivalent simulation using the old (v7/v8/v9) predictor coefficients. Simulations using coefficient files based on a larger number of levels (e.g. 101L) run slower than those with coefficients on fewer levels (e.g. 54L).

For non-scattering MW simulations with CLW absorption, the run-times with the Liebe or TKC are only a little longer than without CLW. However, the Rosenkranz option is significantly more expensive (approximately an additional 30-40% in total simulation run-time) than Liebe due to the relative complexity of the parameterisation.

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

Number of input levels, interpolation

Execution time increases with the number of levels in the input profile. The computational cost depends on the selected interpolation mode: the modes can be ranked from fastest to slowest as 2, 3, 1, 5, 4. For the direct model and for the TL/AD/K with the *pressure_gradients* option set to false, the variation in cost between the modes is not very large. If *pressure_gradients* is true (so that pressure is an active variable in the TL/AD/K) the variation in the cost between modes is larger.

Surface emissivity and reflectance

Use of the internal surface emissivity and BRDF models for sea surfaces is generally more expensive than providing input values for surface emissivity and reflectance. The computational cost of ISEM is insignificant, and the cost of IREMIS is very small. For FASTEM, computation cost is modest (10-20% of the run-time for clear-sky simulations), with FASTEM-6 being slightly more expensive than FASTEM-5. SURFEM-Ocean is considerably more expensive, being itself approximately as costly as the rest of the clear-sky simulation. Use of the Lambertian surface option also increases run-time slightly (~10% additional run-time for clear-sky). The sea surface BRDF model is expensive due to its complexity, adding approximately 50% to the total run-time of clear-sky simulations.

The additional cost of heterogenous surfaces (**nsurfaces**>1) can vary from being negligible to several tens of percent in run-time per additional surface. This is strongly dependent on the surface emissivity/BRDF model(s) used as discussed above.

Scattering simulations

The cost of scattering simulations increases with the number of layers containing scattering particles in the input profile. For aerosol simulations with Chou-scaling, the extra cost over the clear-sky case results mainly from the computation of the layer aerosol optical depths: the integration of the radiative transfer equation is the same as the clear-sky case. For cloud simulations it is both the layer cloud optical depth calculations and the additional cloud columns generated by the cloud overlap scheme which increase run-time. The radiative transfer equation is integrated for every cloud column and the number of columns can theoretically be up to twice the number of cloudy layers for maximum/random overlap (although in practice it is usually much less than this). Two-column overlap schemes (recommended in the MW) are naturally much faster.

The Chou-scaling solver takes roughly two-thirds the run-time of delta-Eddington for the same IR simulation.


The DOM multiple scattering solver is, broadly speaking, at least one or two orders of magnitude more expensive than the fast solvers. However, there are various factors which affect the speed of the DOM algorithm significantly (the input hydrometeor profile, the number of DOM streams, etc). In particular, the cost of DOM simulations grows rapidly as the number of DOM streams increases, and as the number of layers containing scattering particles increases. The second of these factors particularly affects the solver for solar radiances: solar DOM simulations run significantly faster with fewer scattering layers. However, activating the DOM Rayleigh multiple scattering increases run-times significantly because it includes scattering in (almost) all layers.

Aerosol/hydrometeor scattering simulations with explicit optical properties are slightly faster than those which use the pre-defined optical properties though it must be remembered that explicit optical properties mandate the use of the more expensive maximum/random overlap (primarily of relevance for MW simulations). For DOM simulations the effect of this becomes less noticeable as the cost of the radiative transfer calculation is larger. Using the Baran ice scheme is slightly more expensive than using the Baum ice properties for the fast solvers in the UV/VIR/IR. It is significantly more expensive when using DOM and/or for solar simulations because the phase function needs to be explicitly reconstructed for the Baran scheme which is a relatively expensive calculation.

The MFASIS-NN direct model is roughly twice as fast as DOM with 4 streams, at least an order of magnitude faster than DOM with 16 streams and two orders of magnitude if Rayleigh multiple scattering is included. As the number of cloudy layers increases DOM quickly becomes significantly more expensive while the impact on MFASIS-NN run-times is much more modest. For the TL model and especially the AD and K models, MFASIS-NN is very much faster than DOM.

PC-RTTOV



For simulating a full spectrum of a hyperspectral sounder, PC-RTTOV is substantially faster than standard RTTOV. The PC-RTTOV K model has known performance issues, particularly with reconstructed radiances: these may be addressed in a future release.

<p>The EUMETSAT Network of Satellite Application Facilities</p>		<h2 style="text-align: center;">RTTOV v14 Users' Guide</h2>	<p>Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025</p>
---	---	---	--

7.12 Summary of steps for running RTTOV v14

You need to ensure the following in your program which calls RTTOV:

- Create an instance of the **rttov_options** type and set any of the member flags as required. The full list of options is given in Annex J.
- Optional: you may wish to call **rttov_errorhandling** to specify the logical unit to use for error messages.
- An instance of the **rttov_coefs** type should be populated by calling **rttov_read_coefs**.
- Optional: you may wish to call **rttov_user_check_options** to ensure the selected options are consistent with the coefficient file(s).
- Allocate the input/output structures to RTTOV with the number of channels and profiles you want to run with (e.g., by calling **rttov_alloc_direct/tl/ad/k**, see Annex D). If the emissivity and/or BRDF atlases are required these should be initialised now (see Annexes F and G). See **src/test/example_*.F90** for example code detailing the input variables required for RTTOV.
- Initialise an array of **rttov_profile** structures with your atmospheric and surface data. This is shown in Table 7.2 and listed in Annex J.
- Initialise an array of type **rttov_chanprof** with the channel and profile indexes as described in section 7.5.
- Choose whether RTTOV should supply surface emissivities and/or reflectances or you are providing input values via an array of type **rttov_emis_refl** (section 8.3). You can optionally supply values from the land surface emissivity/BRDF atlases (sections 8.3.6 and 8.3.7).
- Optional: you may wish to check the input profiles for unphysical or out-of-specification values before calling RTTOV. This can be achieved using the **rttov_user_check_profile** subroutine (Annex I). If this is used, **opts%config%check_profiles** can be set to false in which case you may wish to use the **rttov_user_check_emis_refl** and/or **rttov_user_check_opt_param** subroutines to check relevant inputs. You may also find the **rttov_print_profile** subroutine (Annex I) useful for debugging purposes: this prints out the contents of a single profile structure.
- Call RTTOV (**rttov_direct**) with the input/output variables and with the coefficient structure corresponding to the instrument you want to simulate.
- When all RTTOV calls are made, then you should free memory by de-allocating the various input and output structures with the **rttov_dealloc_coefs**, **rttov_alloc_direct/tl/ad/k** (see Annexes C and D). If the emissivity and or BRDF atlases were initialised, they should also be de-allocated now (see Annexes F and G).
- Most user-level RTTOV subroutines return an error status. This variable should be tested after each subroutine call: non-zero values indicate that an error occurred.
- If you want to run hydrometeor or aerosol scattering simulations, follow the additional guidance in section 8.4.

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

8 Details of specific RTTOV capabilities

8.1 Solar radiation

8.1.1 Enabling solar radiation

RTTOV can simulate radiances for UV, visible and near-IR channels, and can include the effects of solar radiation in short-wave IR channels. To include solar radiation the logical flag `opts%rt_all%solar` must be set to true. With this flag set, the contribution of solar radiation is included in all channels with wavelengths below 5 μ m (in practice, for some very high-peaking short-wave IR CO₂ channels around 4 μ m solar radiation may be disabled as it has no significant impact: this applies to some hyperspectral IR sounders such as IASI). For channels below 3 μ m terrestrial atmospheric and surface emission is ignored so the *only* contribution is solar radiation and output radiances in such channels are zero if the `solar` option is false.

Solar simulations can be performed with any v13 predictor coefficient files, or with the older v9 predictor coefficients (see section 3). These files contain a SOLAR_SPECTRUM section which includes a flag for each channel which is 0 for “thermal” channels (no significant solar contribution), 2 for “solar” channels (no significant thermally emitted contribution) and 1 for channels with both thermal and solar contributions. These flags are read into the coefficients structure in `coefs%coef%ss_val_chn(:)` which has one element per channel read from the coefficient file. Users have noted that a solar contribution due to sun glint can be detected in window channels in the thermal IR. By default these channels are considered thermal-only, but it is possible to include the solar contribution for them by setting the element of `coefs%coef%ss_val_chn(:)` corresponding to the channel in question to 1 after the coefficient file has been read, or by editing the values in the second column of the SOLAR_SPECTRUM section directly in the coefficient file. Note though that for sensors in low earth orbit, optical depth coefficients for IR channels are only trained for zenith angles up to 65° and thus for larger solar zenith angles the simulation accuracy will be degraded. You should *not* change any elements of this array corresponding to UV/VIS/NIR channels as simulation of thermal radiation for these channels will not work correctly.

In addition to the profile variables required for non-solar simulations (section 7.4, table 7.2), the solar zenith angle, and satellite and solar azimuth angles must also be specified in the input profile structure in `profiles(:)%sunzenangle`, `profiles(:)%azangle` and `profiles(:)%sunazangle` respectively. Note that solar radiation is only included if the solar zenith angle is less than ~85°. Figure 8.1 illustrates the definition of azimuth angle.

RTTOV provides a subroutine, `rttov_calc_solar_angles`, which can be used to calculate and set the `sunzenangle` and `sunazangle` members of a `profiles(:)` array assuming the `latitude`, `longitude`, `date(:)`, and `time(:)` members have been populated for all profiles. This is described in Annex I.

When solar radiation is enabled, the surface BRDF is required for solar channels. See section 8.3 for full details on this.

8.1.2 Top-of-atmosphere solar irradiance

The SOLAR_SPECTRUM section in the RTTOV optical depth coefficient files also provides the top of atmosphere band-integrated solar irradiance at 1 AU (astronomical unit) in units of mW/m²/cm⁻¹. These values are read into the `coefs%coef%ss_solar_spectrum(:)` array and are calculated using the AER solar source function software (based on Kurucz, 1992). These irradiances are used for the solar calculations: it is possible to change the values in the coefficient file or directly in the `coefs%coef%ss_solar_spectrum(:)` array after reading the coefficients if desired. If the `profile(:)%date(:)` array has been populated with a date for the profile (year, month, day) then these are used to adjust the band solar irradiances according to the Earth-sun distance. If the `date(:)` contains a date earlier than or equal to the default value (1/1/1950), no adjustment is performed. The date is used to adjust the solar spectrum value for the Earth-sun distance according to the month and day of the year.

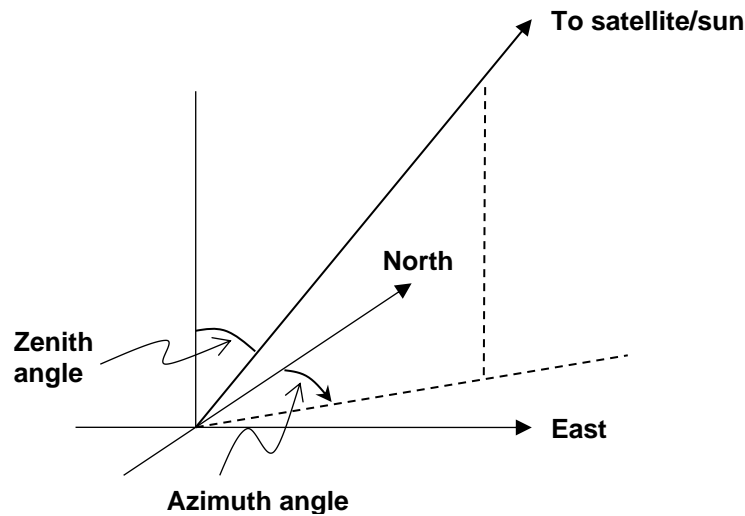


Figure 8.1: Azimuth angles are defined as the angle in the clockwise direction between North and the projection of the satellite view-path or sun-surface path onto the horizontal plane at the surface.

8.1.3 Clear-sky Rayleigh scattering

When using v13 predictor coefficient files (section 3), the Rayleigh extinction is parameterised in the code. This means it is possible to exclude the effects of Rayleigh scattering entirely if desired. For v9 predictor files, the Rayleigh extinction is included in the line-by-line simulations used to train the optical depth coefficients, which means for these it is possible only to exclude the scattering towards the sensor and not the Rayleigh scattering extinction.



A single-scattering parameterisation is implemented which accounts for radiation Rayleigh-scattered towards the sensor. This single-scattering contribution can be turned off by setting the `opts%rt_all%rayleigh_single_scatt` option to false. The single-scattering calculation for the v13 predictors is described in the RTTOV v13 Science and Validation Report. The calculation used with v9 predictors is described in the RTTOV v11 Science and Validation Report. In practice the differences between the two are small.

The `opts%rt_all%rayleigh_max_wavelength` option is used to specify the maximum wavelength in microns of channels for which Rayleigh scattering will be considered (default 2 μ m). The `opts%rt_all%rayleigh_min_pressure` option can be used to specify the minimum pressure in hPa at which to consider Rayleigh scattering: Rayleigh scattering is ignored in layers below this pressure. The default is 0hPa, so that Rayleigh scattering is included in all layers. Both options apply to the Rayleigh single-scattering contribution, and, for v13 predictors, also to the Rayleigh extinction parameterisation. You can turn Rayleigh scattering off entirely by setting `rayleigh_max_wavelength` to 0, in which case the output transmittances (for v13 predictor coefficients) will include only gas absorption and not Rayleigh extinction.

It is also possible to simulate full Rayleigh multiple scattering using the RTTOV DOM solver, including for clear-sky profiles: see section 8.4.5. The `rayleigh_max_wavelength` and `rayleigh_min_pressure` options also apply in this case.

8.1.4 Output top-of-atmosphere reflectances

The output radiance structure (section 7.8.2, Annex J) contains arrays for satellite-seen reflectances. These are bi-directional reflectance factors (BRFs) calculated as the ratio of the satellite seen radiance to that which would result if the surface was a perfect Lambertian reflector. BRFs are only calculated for channels with a solar contribution. The output radiance arrays (e.g., `total(:)`, `clear(:)`) are populated for all channels. The output brightness temperature arrays are not populated for channels with no thermally emitted contribution, and you should be careful in interpreting the output brightness temperatures for channels with both solar and emitted contributions since these may differ significantly from physical temperatures.

		RTTOV v14 Users' Guide	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	-------------------------------	---

8.1.5 *UV simulations*

RTTOV has a basic capability for simulating UV sensors. UV simulations are performed in the same way as those for visible/near-IR channels. However, some caveats apply in the UV, and these are discussed below.

Gas absorption optical depth parameterisation

Validation of RTTOV against the line-by-line model used for training coefficients (see RTTOV web site), and against an independent reference radiative transfer model (Wang and Tuinder, 2022) indicate that the v13 predictors provide reasonable accuracy in the UV. Developments are planned more generally for the v13 predictors to improve accuracy across the spectrum including the UV.

Rayleigh scattering

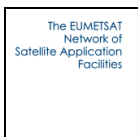
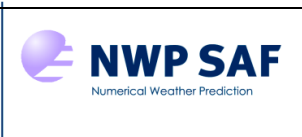
Section 8.1.3 describes the Rayleigh scattering treatment in RTTOV. The Rayleigh single scattering, enabled by default and applied in the UV, is inadequate at UV wavelengths. Omitting multiple-scattering interactions results in top of atmosphere radiances being underestimated. The errors due to omitting multiple scattering can be very large, particularly for channels broadly in the range 300-500 nm, and as such the DOM solver (section 8.4.2) with Rayleigh multiple scattering enabled (section 8.4.5) is recommended although this is computationally expensive. The fast treatment of Rayleigh multiple scattering will be addressed in a future release.

Scattering simulations

Section 8.4 gives details on hydrometeor and aerosol simulations in RTTOV. Hydrometeor and aerosol optical property files are available for UV sensors. For all particle types, the underlying refractive index datasets contain data down to at least 0.25 μm . Where data do not exist for shorter wavelengths, the optical properties are extrapolated at constant value. Currently the MFASIS fast parameterisation has not been extended to the UV, but this is planned for a future release.

Surface BRDF

The RTTOV sea surface reflectance model (section 8.3.2) applies in the UV, and the refractive index datasets used contain data for wavelengths above 0.2 μm . However, the land surface BRDF atlas (section 8.3.7) only contains data for wavelengths above 0.4 μm . BRDF values are extrapolated at constant value to shorter wavelengths. It is planned to address land surface reflectances more thoroughly in a future release.

		RTTOV v14 Users' Guide	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	-------------------------------	---

8.2 Simple cloud (non-scattering)

The treatment of hydrometeor scattering is detailed in section 8.4, but the simplest cloud approach described here is automatically applied for all clear-sky UV/visible/IR simulations. Assuming a black, opaque cloud at a single level that fills the radiometer field of view, the simulation of cloud affected radiances $L_{Cld}(v, \theta)$ is defined as:

$$L_{Cld}(v, \theta) = \tau_{Cld}(v, \theta)B(v, T_{Cld}) + \int_{\tau_{Cld}}^1 B(v, T) d\tau \quad (8.1)$$

where $\tau_{Cld}(v, \theta)$ is the cloud top to space transmittance and T_{Cld} the cloud top temperature determined from the cloud top pressure in the input state vector (**profiles(:)%ctp**). The emissivity of the cloud top is assumed to be one which is a reasonable assumption for optically thick water cloud at infrared radiances but not valid for optically thin cloud or any cloud at microwave frequencies. For partially cloud-filled fields of view, the cloudy radiance given by equation 8.1 is combined with the clear sky radiance (equation 2.1) using the input fractional cloud cover C as follows:

$$L(v, \theta) = (1 - C)L_{Clr}(v, \theta) + CL_{Cld}(v, \theta) \quad (8.2)$$

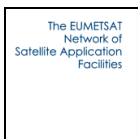
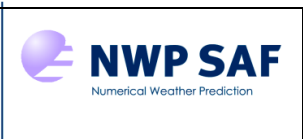
This simple cloud scheme is activated by providing a cloud top pressure in **profiles(:)%ctp** and a non-zero cloud fraction in **profiles(:)%cfraction** (with 1 indicating fully overcast). The relevant radiance outputs (section 7.8.2 and Annex J) in the radiance structure are **clear(:)** (clear radiance), **cloudy(:)** (radiance assuming 100% cloud fraction) and **total(:)** (cloudy radiance as in equation 8.2). The cloudy radiance is calculated by interpolating the **overcast(:, :)** radiances to the cloud top pressure.

The simple cloud scheme is **not** applied if scattering calculations have been activated by setting **opts%scatt%aerosols** and/or **opts%scatt%hydrometeors** to true (section 8.4).

This simplistic cloud treatment has also been implemented for UV/visible/near-infrared channels with no thermally emitted component. In this case the cloudy radiance $L_{Cld}(v, \theta, \theta_{sol})$, where θ_{sol} is the solar zenith angle, is calculated as the reflected component of the solar radiation from the cloud top assuming the cloud is optically thick and acts as a Lambertian reflector with a default BRDF of $0.7/\pi$ for wavelengths below $1\mu\text{m}$ and $0.6/\pi$ for wavelengths above $1\mu\text{m}$. If the **refl_cloud_top(1:nchanprof)** optional argument to the core RTTOV model routines (Annex H) contains values greater than zero, these are used in preference to the default BRDFs for the corresponding channels. The cloudy radiance includes the effects of clear-sky Rayleigh scattering above the cloud top (see section 8.1.3). This is a very crude treatment of cloud in UV, visible and near-infrared channels but may be useful for qualitative applications such as simulated imagery, particularly for visible channels. Note that **refl_cloud_top(:)** is not an active variable in the TL/AD/K models and so there is no corresponding TL/AD/K argument.

Notes

For channels which would normally have significant thermally emitted and solar contributions (for example, channels around $3.9\mu\text{m}$), no solar radiation is included in the simple cloudy radiances in accordance with the assumption that the cloud top is perfectly emissive (there is no reflection of solar radiation from the cloud top). The simple cloud scheme is never applied for MW sensors.

		RTTOV v14 Users' Guide	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	-------------------------------	---

8.3 Surface emissivity and reflectance

There are three channel-dependent surface variables required in general for RTTOV simulations:

- **Emissivity** – the efficiency with which the surface emits radiation, required for all **thermal channels**.
- **BRDF** – the ratio of reflected radiance towards the satellite to incoming solar irradiance (assuming the sun is treated as a point source). In the absence of an atmosphere the output BRF (simulated reflectance) would be equal to the surface BRDF multiplied by π . The BRDF is used to compute the contribution to the satellite-seen radiance of the direct surface-reflected solar beam. It is usually a function of the solar and satellite zenith and azimuth angles. It is required for all **solar channels** when solar radiation is enabled (see section 8.1).
- **Diffuse reflectance** – a BRDF-like reflectance used for radiation emitted or scattered downward by the atmosphere. For thermal channels this is used for the downwelling radiation emitted/scattered by the atmosphere. It is often equal to $(1-\text{emissivity})$ and it represents a specular reflectance. For pure solar channels it is only used for the downward Rayleigh-scattered radiation along the satellite-surface path when Rayleigh single-scattering is enabled (see section 8.1.3). It represents the satellite-surface-satellite BRDF multiplied by π . It is often equal to $BRDF * \pi$ in the absence of better information (noting that $BRDF$ is the sun-surface-satellite BRDF described above). Required for **all channels**.

RTTOV can provide values for these three quantities, or you can input values for RTTOV to use. In all cases, RTTOV returns the values that were used in the simulations. The input and output variables are contained in the **rttov_emis_refl** structure which is described in Annex J. The members of this structure are arrays of size **nchanprof** (the size of the **chanprof(:)** input array – section 7.5) containing data corresponding to each channel being simulated. An array of this data type **emis_refl(:)** of size **nsurfaces** is passed as an argument to the core RTTOV routines (Annex H). Here, **nsurfaces** is the number of surfaces associated with each profile as described in sections 7.4 and 8.3.10. Surface emissivities and reflectances are required for each surface independently.

There are three logical array members of the **rttov_emis_refl** structure: **calc_emis(:)**, **calc_brdf(:)**, and **calc_diffuse_refl(:)**. Where the elements are set to true, RTTOV will compute or otherwise provide values of emissivity, BRDF, and/or diffuse reflectance respectively for the corresponding channels. If you want to supply your own values, you set the corresponding elements to false and provide the input values in the corresponding elements of the **emis_in(:)**, **brdf_in(:)**, and/or **diffuse_refl_in(:)** member arrays respectively.

The surface type specified in the **profiles(:)%skin(:)%surftype** variable (Table 7.2, Annex J) determines the type of emissivity and/or reflectance calculation performed for channels where **calc_emis/brdf/diffuse_refl(:)** are true.

The values of emissivity, BRDF, and diffuse reflectance used by RTTOV in the simulation are stored in the **emis_out(:)**, **brdf_out(:)**, and **diffuse_refl_out(:)** members of the **rttov_emis_refl** structure. If you provide input values to use, the output values will be identical to those input values.

Notes

In most cases you should set **calc_diffuse_refl(:)** to true for all channels and allow RTTOV to provide values for the diffuse reflectance. The ability to specify this independently is provided to give full control over surface parameters to expert users.

When carrying out simulations with no solar radiation, the BRDF inputs should be ignored (the BRDF-related arrays will not be allocated in the **emis_refl** data structure in this case). When carrying out simulations only for pure solar channels (at wavelengths below 3 μm), the emissivity inputs can be ignored.

For MW sensors it is important to define the polarisation status of each channel. This is specified in the POLARISATION section of the optical depth coefficient files (the section is named “FASTEM” in RTTOV v13-compatible and earlier coefficient files), and the polarisation IDs are defined in Table 8.1. Different polarisations are defined for cross-track scanners, conical scanners, and polarimetric instruments. In general you shouldn't need to worry about this unless you specifically want to change the polarisation of a particular microwave channel in which case you should edit the coefficient file to modify the relevant polarisation ID(s) or you can modify the values directly in **coefs%coef%polarisation(:)** after the coefficients have been read in (section 7.2). The gas absorption coefficients are

independent of the channel polarisations. **NB** you should not set any polarisation ID from another value to 7 as additional information is then required defining the relative proportions of V and H: if you need to do this, contact the NWP SAF help desk for further advice.

Pol ID in <i>rtcoef</i> file	Definition	Examples of applicable sensors
0	Average of vertical and horizontal polarisation i.e. 0.5(H+V)	SSMIS
1	Nominal vertical at nadir rotating with view angle QV	AMSU-A/B, MSU, MHS
2	Nominal horizontal at nadir rotating with view angle QH	AMSU-A, MSU, MHS
3	Vertical V	SSM/I, SSMIS, AMSR, Windsat
4	Horizontal H	SSM/I, SSMIS, AMSR, Windsat
5	+ 45 minus -45 (3rd stokes vector) S3	Windsat, COWVR
6	Left circular - right circular (4th stokes vector) S4	Windsat, COWVR
7	Channel-specific fixed combination of H and V	TROPICS

Table 8.1: Definition of polarisation status.

8.3.1 RTTOV calculation of surface emissivity

RTTOV provides a choice of sea surface emissivity models. Different models are available for use with IR and MW sensors. These are activated for profiles where the surface type is set to sea and where **calc_emis(:)** is set to true. For land and sea-ice surface types, RTTOV can provide emissivities as described below.

Infrared sensors – sea surfaces

The IR sea emissivity model is selected via the **opts%surface%ir_sea_emis_model** option (Table 8.2).

The IREMIS model (RTTOV v12 Science and Validation Report) is recommended and is the default. This is a physically based emissivity parameterisation. The profile variables used by IREMIS are the 10m wind speed u/v components, skin temperature, and zenith angle.

The older ISEM model (Sherlock, 1999) remains available. This is a much simpler parameterisation only in terms of zenith angle, and as such this model does not provide any TL/AD/K emissivity sensitivity through the input profile variables, so you can manage the TL/AD/K as if you had provided the emissivity values directly to RTTOV (see section 8.3.11).

PC-RTTOV is trained using the IREMIS model and this is automatically selected for sea surfaces when **calc_emis(:)** is true. See section 8.6.

opts%surface%ir_sea_emis_model	Description
ir_emis_model_isem (1)	ISEM – simple parameterisation in terms of zenith angle only
ir_emis_model_irem (2)	IREMIS (default) – more physically based parameterisation in terms of zenith angle, 10m wind u/v components, and skin temperature.

Table 8.2: Options for IR sea emissivity model. The constants corresponding to the integer values are in the *rttov_const* module (Annex K).

Infrared sensors – land/sea-ice surfaces

For land and sea-ice surface types RTTOV assigns fixed emissivity values of 0.98 (land) and 0.99 (sea-ice) where **calc_emis(:)** is true. In this case, the TL/AD/K is treated as if you had input emissivity values to RTTOV (see section 8.3.11).

Microwave sensors – sea surfaces

The MW sea surface emissivity model is selected via the **opts%surface%mw_sea_emis_model** option (Table 8.5).

SURFEM-Ocean (Kilic *et al.*, 2023), the default, is the recommended option. It is a neural network parameterisation of an optimised configuration of the PARMIO reference model (English *et al.*, 2020). It is valid over all frequencies supported by RTTOV (0.5-700 GHz) and supports all instruments including polarimetric sensors. It is intended to supersede FASTEM (see below). The profile variables used by SURFEM-Ocean are the 10m wind speed u/v components, skin temperature, salinity, zenith angle, and azimuth angle. RTTOV ensures that the values provided to the SURFEM-Ocean neural network do not exceed the bounds used in the training. These bounds are given in Table 8.3. If the bounds are exceeded the corresponding quality flag `qflag_emis_limits` is set in the `radiance%quality(:)` output (see section 7.8.3).

Variable	Minimum value	Maximum value
Skin temperature	-2°C / 271.15 K	32°C / 305.15 K
Salinity	0 ‰	40 ‰
Wind speed ($\sqrt{u^2 + v^2}$)	0 m/s	50 m/s

Table 8.3: input variable training limits for SURFEM-Ocean. Values outside these limits are clipped to the corresponding limit for the SURFEM-Ocean calculation and the corresponding quality flag is set.

FASTEM is an older MW sea surface emissivity model. Previously six versions of FASTEM were available in RTTOV, but now only FASTEM-5 and FASTEM-6 are implemented (Table 8.4). FASTEM-5/6 were retained as many users are successfully using them operationally (especially FASTEM-6), and they remain of interest as they can optionally take user-specified foam fraction as an input parameter. FASTEM-6 is recommended over FASTEM-5. FASTEM-5 is not recommended for polarimetric sensors, and neither FASTEM version should be used for channels above 200 GHz.

The profile variables used by FASTEM for sea surface emissivities are the 10m wind speed, skin temperature, salinity, zenith angle, and azimuth angle. By default, FASTEM calculates a value for the ocean surface foam fraction: you can optionally supply an explicit value for the foam fraction in the profile structure (`profiles(:)%skin(:)%foam_fraction`) by setting `opts%surface%use_foam_fraction` to true.

FASTEM-1	Fast emissivity model fitted to geometric optics model (English and Hewison, 1998).
FASTEM-2	As FASTEM-1 but with reflectivity dependence on atmospheric transmittance (Deblonde, 2000).
FASTEM-3	As FASTEM-2, but with azimuthal dependence (Liu and Weng, 2003).
FASTEM-4	Fast emissivity model as previous versions but fitted to a new two-scale scattering model (Liu <i>et al.</i> , 2011).
FASTEM-5	As FASTEM-4, but with a different foam coverage parametrisation and some fitting issues introduced with FASTEM-4 now corrected (Bormann <i>et al.</i> , 2012).
FASTEM-6	As FASTEM-5, but with an improved azimuthal dependence (Kazumori and English, 2014, and RTTOV v12 Science and Validation Report). Implements FASTEM-3 azimuthal dependence for Stokes 3/4 channels on polarimetric sensors (e.g., Windsat, COWVR).

Table 8.4. Summary of FASTEM versions. Versions in grey are no longer available in RTTOV.

NB The SURFEM-Ocean and FASTEM models apply a non-specular correction when computing the diffuse reflectance. This means that if you were to run RTTOV with `calc_emis(:)` and `calc_diffuse_refl(:)` set to true with one of these emissivity models, and then you were to repeat the simulation inputting the returned emissivities to RTTOV with `calc_emis(:)` set to false, you would also need to input the diffuse reflectances computed by the emissivity model and set `calc_diffuse_refl(:)` to false in order to obtain identical output radiances.

<code>opts%surface%mw_sea_emis_model</code>	Description
<code>mw_emis_model_fastem5</code> (1)	FASTEM-5 – profile inputs are 10m wind u/v components, skin temperature, salinity, zenith angle, azimuth angle, and, optionally, foam fraction. Not recommended for polarimetric sensors (Stokes 3/4 component channels), and only applies to channels at frequencies below 200 GHz.
<code>mw_emis_model_fastem6</code> (2)	FASTEM-6 – profile inputs are as for FASTEM-5. Applicable to all sensors (including polarimetric sensors) with channels below 200 GHz.
<code>mw_emis_model_surfem_ocean</code> (3)	SURFEM-Ocean (default) – profile inputs are 10m wind u/v components, skin temperature, salinity, zenith angle, and azimuth angle. Applicable to all sensors supported by RTTOV (including polarimetric sensors, and sensors with sub-mm channels).

Table 8.5: Options for MW sea emissivity model. The constants corresponding to the integer values are in the `rttov_const` module (Annex K).

Microwave sensors – land/sea-ice surfaces

For land/sea-ice surfaces for MW sensors, it is recommended to use the emissivity atlases (section 8.3.6) which provide emissivity values that can be passed into RTTOV. These can be used as a first guess for a dynamic emissivity retrieval (section 8.4.12). If it is not possible or practical to use the atlases, or the emissivity atlas does not have data at the given location, there is also a land and sea-ice FASTEM parameterisation. This is entirely separate to the FASTEM-5/6 sea surface emissivity models and is not affected by the setting of `opts%surface% mw_sea_emis_model`. This parameterisation is very old and is deprecated but provides a source of emissivities in the absence of anything better. The parameterisation computes emissivities based on the parameters specified in `profiles(:)%skin(:)%fastem(1:5)`. Values for these parameters for different surface types are given in Table 8.6.

Surface type	FASTEM parameters 1:5
Typical RTTOV default for land	3.0, 5.0, 15.0, 0.1, 0.3
Summer land surface	
Forest	1.7, 1.0, 163.0, 0.0, 0.5
Open grass	2.2, 1.3, 138.0, 0.0, 0.42
Bare soil	2.3, 1.9, 21.8, 0.0, 0.5
Winter surface type	
Forest and snow	2.9, 3.4, 27.0, 0.0, 0.0
Deep dry snow	3.0, 24.0, 60.0, 0.1, 0.15
Frozen soil	117.8, 2.0, 0.19, 0.2, 0.35
Sea ice	
Grease ice	23.7, 7.7, 17.3, 0.0, 0.15
Baltic nilas	1.6, 3.3, 2.2, 0.0, 0.0
New ice (no snow)	2.9, 3.4, 27.0, 0.0, 0.0
New ice (snow)	2.2, 3.7, 122.0, 0.0, 0.15
Brash ice	3.0, 5.5, 183.0, 0.0, 0.0
Compact pack ice	2.0, 1700000.0, 49000000.0, 0.0, 0.0
Fast ice	1.5, 77.8, 703.0, 0.1, 0.35
Lake ice + snow	1.8, 67.1, 534.0, 0.1, 0.15
Multi-year ice	1.5, 85000.0, 4700000.0, 0.0, 0.0

Table 8.6: Values for FASTEM parameters for various land and sea-ice surface types. Taken from English and Hewison (1998).

8.3.2 RTTOV calculation of surface BRDF

For solar-affected channels, RTTOV implements a physically-based sun-glint model to compute sea surface BRDFs where `calc_brdf(:)` is true. This model is based on the Yoshimori *et al.* (1995) wave facet model and uses the Elfouhaily *et al.* (1997) wave spectrum parameterisation. The `opts%surface%solar_sea_refl_model` allows selection between solar sea BRDF models, but currently only one is implemented so this option should not be modified from the default value. The profile variables used are the 10m wind speed, wind fetch, and watertype (fresh or ocean) along with the satellite and solar zenith and azimuth angles. The sun-glint model accounts for glint from the water surface but does not account for sub-surface scattering (ocean colour). Therefore, for pure solar channels (at wavelengths below 3 μm) the diffuse reflectance (strictly, the diffuse reflectance divided by π) is added to the BRDF from the sun-glint model. This is either the diffuse reflectance provided by the user or otherwise that calculated by RTTOV (section 8.3.3).

For land and sea-ice surfaces where `calc_brdf(:)` is true, RTTOV calculates the BRDF as $(1-\text{emissivity})/\pi$ for solar-affected thermal channels (wavelengths 3-5 μm), and is to $0.3/\pi$ (land) and $0.8/\pi$ (sea-ice) for pure solar channels (wavelengths below 3 μm). For the pure solar channels these are very crude values, so it is recommended to use the BRDF atlas instead (section 8.3.7).

8.3.3 RTTOV calculation of surface diffuse reflectance

The calculation of diffuse reflectances when `calc_diffuse_refl(:)` is true depends on the surface type, the emissivity model, and the spectral region.

For thermal channels (at wavelengths above 3 μm), in most cases the diffuse reflectance is calculated as $(1-\text{emissivity})$. The exception is for the MW sea surface emissivity models which apply a non-specular correction to this value.

For pure solar channels (at wavelengths below 3 μm), the diffuse reflectance is computed from USGS ocean or coastal water reflectance spectra (Kokaly *et al.*, 2017) for sea surfaces (the choice depending on the profile **watertype**), or otherwise for land/sea-ice surfaces is set to the $BRDF*\pi$. If you have information on sub-surface scattering (ocean colour) you may wish to use this to provide input values for the diffuse reflectance.

8.3.4 Summary of emissivity and reflectance calculations

The specification of emissivity, BRDF, and diffuse reflectance is complicated because it depends on the setting of the corresponding **calc_***(**:**) variable (true/false), the surface type, the spectral region, and in some cases the selected surface model. This is summarised in Table 8.7.

Variable	calc_* Boolean	Surface type	Variable values at channel wavelengths		
			<3 μm	3-5 μm	>5 μm
Emissivity	calc_emis T	Sea	N/A	IR: ISEM, IREMIS MW: SURFEM-Ocean, FASTEM-5/6	
		Land		IR: 0.98 MW: FASTEM land/sea-ice parameterisation	
		Sea-ice		IR: 0.99 MW: FASTEM land/sea-ice parameterisation	
	calc_emis F	All		User input emissivity (e.g., from atlas) <i>emis_refl%emis_in(:)</i>	
BRDF	calc_brdf T	Sea	Sunglint BRDF model + (diffuse reflectance)/ π	Sunglint BRDF model	N/A
		Land	$0.3/\pi$	$(1-\text{emissivity})/\pi$	
		Sea-ice	$0.8/\pi$		
	calc_brdf F	All	User input BRDF (e.g., from atlas) <i>emis_refl%brdf_in(:)</i>		
Diffuse reflectance	calc_diffuse_refl T	Sea	Fixed USGS spectra	IR: $(1-\text{emissivity})$ MW: computed by SURFEM-Ocean/FASTEM-5/6 if <i>calc_emis T</i> otherwise $(1-\text{emissivity})$	
		Land	$brdf*\pi$	$(1-\text{emissivity})$	
		Sea-ice			
	calc_diffuse_refl F	All	User input diffuse reflectance <i>emis_refl%diffuse_refl_in(:)</i>		

Table 8.7: summarising emissivity, BRDF, and diffuse reflectance in different spectral regions for each surface type.



8.3.5 Calling RTTOV emissivity/BRDF models outside RTTOV

A subroutine **rttov_get_sea_emis** is available which returns emissivities from the internal emissivity models without having to run a full simulation. This is described fully in Annex F. This is primarily intended to provide access to the sea surface emissivity models (as the name indicates), but if the profile surface type is set to land or sea-ice, then the relevant emissivity calculations/values are returned. This includes the FASTEM land/sea-ice parameterisation, which in turn requires the profile **fastem(1:5)** parameters to be set appropriately (section 8.3.1).

The diffuse reflectance cannot be returned from **rttov_get_sea_emis**. In most cases this would be computed by RTTOV as $(1-\text{emissivity})$, but where this is not the case (the MW sea surface emissivity models), the calculation requires the total atmospheric transmittance which is not available without running the full RTTOV simulation.

Similarly, a subroutine **rttov_get_sea_brdf** is available which returns BRDFs from the internal surface reflectance models without having to run a full simulation. This is described fully in Annex G.

These routines are intended to be analogous to the **rttov_get_emis** and **rttov_get_brdf** routines that are used to access the emissivity and BRDF atlases respectively (sections 8.3.6 and 8.3.7).

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

8.3.6 Emissivity atlases

RTTOV provides IR and MW land surface emissivity atlases. These are called externally to RTTOV and the emissivities are passed to RTTOV in the **emis_in(:)** member of **emis_refl** with the corresponding elements of **calc_emis(:)** set to false. Each atlas provides monthly climatological emissivity values. There are three IR and two MW emissivity atlases each of which has slightly different inputs, outputs, and configuration options. However, all atlases are accessed via a common interface (see Annex F).

UWIRemis, CAMEL single-year, and CAMEL climatology IR emissivity atlases

In the IR there are three atlases:

- University of Wisconsin UWIRemis atlas. This is described in Borbas and Ruston (2010). The resolution is 0.1° in lat/lon.
- CAMEL single-year atlas. These datasets are based on a single month of data from a particular year. Version 2 of this atlas is described in the RTTOV v12 Science and Validation Report and is based on data from 2007 only. RTTOV v14 also supports the CAMEL v3 atlas (recommended over v2): these data are available for every month from 2000 to 2021 from the NASA LP DAAC website (<https://lpdaac.usgs.gov/products/cam5k30cfv003/>). This has a resolution of 0.05° in lat/lon.
- CAMEL climatology atlas. These datasets are multi-year climatologies. Version 2 of this atlas is described in Borbas and Feltz (2019), and RTTOV v14 supports the CAMEL climatology v3 atlas (recommended over v2). This has a resolution of 0.05° in lat/lon. The CAMEL climatology atlas provides a more sophisticated treatment of snow than the other IR atlases (see below).



These atlases work in very similar ways. The atlases can optionally include a correction for zenith angle effects (see Borbas, 2014): this is specified when the atlases are initialised. By default, the atlases, once loaded, can be used with any IR instrument. However, if you are only using the atlas with a single instrument, you can supply the corresponding coefficients structure to the atlas initialisation call. This allows some pre-calculations to be done that result in emissivities being returned much more rapidly from the atlas: it is recommended to use this option where possible, but the resulting atlas data can only be used with the instrument for which they were initialised. PC-RTTOV is trained using the CAMEL v3 climatology IR emissivity atlas and it is recommended to use this atlas to provide land and sea-ice emissivities for PC-RTTOV simulations. See section 8.6.

The CAMEL climatology atlas provides emissivity standard deviations computed from the climatology (which are also available with the CAMEL single-year atlas), and it includes a more sophisticated treatment of snow than the other two atlases. For the UWIRemis and CAMEL single-year atlases, if the profile **snow_fraction** is greater than zero, the atlas emissivity value is linearly combined with a fixed snow emissivity spectrum. In contrast, the CAMEL climatology atlas treats snow as follows:

- if the input **snow_fraction** = 1: if the atlas emissivity contains snow, then the snow PC spectrum is used, otherwise the fixed snow emissivity spectrum is used.
- if 0 < **snow_fraction** < 1: if the atlas emissivity contains snow, then the atlas emissivity is returned without modification, otherwise the atlas emissivity and fixed snow spectrum are linearly combined according to the **snow_fraction**.
- if **snow_fraction** = 0: if the **snow_correction** flag is true (recommended, this is an argument to **rttov_get_emis**) then if the atlas emissivity contains snow contamination the atlas will attempt to find a snow-free PC spectrum to use. If it cannot find one (or if **snow_correction** is false) it simply returns the atlas emissivity.

The UWIRemis and CAMEL single-year atlases optionally output the atlas eigenvectors and eigenvalues corresponding to the returned emissivities. This feature is intended for advanced users. It has not been implemented for the CAMEL climatology atlas due to the complexity of this atlas: each returned emissivity can be a combination of multiple spectra, each with their own PC decomposition.

The following summarises the emissivity atlas inputs/outputs and treatment for different surface types:

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

Surface type:

- sea: returns negative values
- sea-ice: a fixed ice emissivity spectrum is used
- land: returns emissivities (where it has data) linearly combined with a snow emissivity spectrum according to the **skin%snow_fraction** profile variable (see above for more information about the treatment of snow).

Input profile variables:

- mandatory: **latitude, longitude, skin%surftype**
- **skin%snow_fraction** (this modifies emissivities if the value is >0, but see above for more information regarding the CAMEL climatology atlas)
- **zenangle, sunzenangle** (only required if angular correction was requested; **sunzenangle** only needs to be <85° (day) or >85° (night), the specific value does not matter for the atlas)

Outputs:

- emissivities (negative values where atlas has no data)
- optional: emissivity standard deviation (requires a flag at setup, requires more memory)
- optional: quality flag
- optional (UWIRemis and CAMEL single-year only): atlas PC eigenvalues and eigenvectors

TELSEM2 MW atlas and interpolator

TELSEM2 (Wang *et al.*, 2016) is valid for all MW satellite instruments that RTTOV can simulate including instruments with sub-mm channels. When loaded the atlas data may be used with any MW instrument. TELSEM2 includes emissivities for climatological sea-ice and therefore it ignores the specified **surftype** and simply returns a value if it has data. The native resolution of the emissivities is 0.25° lat/lon, but optionally a different resolution may be specified to return emissivity values integrated over a larger area. For frequencies below 19GHz the atlas returns the emissivity value for 19GHz. For frequencies above 85GHz the atlas returns the emissivity at 85GHz except for certain classes of sea-ice (see the reference for more information about this atlas).

Surface type:

- sea/sea-ice/land: returns emissivity values if it has data, otherwise negative values.

Input profile variables:

- mandatory: **latitude, longitude, zenangle**

Outputs:

- emissivities (negative values where atlas has no data)
- optional: emissivity standard deviation for each channel
- optional: emissivity covariance matrix for all channels specified (one matrix per profile)

CNRM MW atlas

The CNRM MW atlas is described in Karbou *et al.* (2006) and Karbou *et al.* (2010). Unlike the other atlases the CNRM atlas is only compatible with a subset of satellite instruments. These are: AMSU-A, AMSU-B/MHS, SSMI/S, and ATMS. In addition, the CNRM atlas datasets are generated from data gathered over a specific year. The year is specified when setting up the dataset. Currently data are available for 2015 (the default if unspecified when setting up the atlas) and 2014.

Surface type:

- sea/sea-ice: returns negative values
- land: returns emissivity values if it has data, otherwise negative values.

Input profile variables:

- mandatory: **latitude, longitude, zenangle, skin%surftype**

Outputs:

- emissivities (negative values where atlas has no data)

All emissivity atlases

The input variables used by each atlas are summarised in Table 8.8.

A common interface is provided to both IR and MW atlases. Three subroutines are used to access the atlases:

- **rttov_setup_emis_atlas** : this allocates the necessary arrays and reads the data for a given atlas for a given month. In some cases, the atlas data are specific to a particular sensor.
- **rttov_get_emis** : this returns surface emissivity values at a given latitude/longitude for the specified channels of the given instrument. This is called when populating the **emis_in(:)** array for input to RTTOV.
- **rttov_deallocate_emis_atlas** : deallocates memory used by the atlas and is called once the atlas is no longer required.

To initialise data for an emissivity atlas you must first declare a variable of type **rttov_emis_atlas_data** which is defined in the module **rttov_emis_atlas_mod**. This variable is passed into each of the above subroutines, and it contains the atlas data for a specific month and (where relevant) for a specific instrument. This enables you to allocate data for multiple months and/or instruments simultaneously.

The interfaces of these subroutines are described in Annex F. Note that if an atlas does not contain emissivity data for the given surface type or lat/lon location it will return negative values for the corresponding emissivities. You should check the returned emissivities and handle negative values accordingly, for example by setting the corresponding elements of **calc_emis(:)** to true. An example of using the atlas subroutines with a forward model call can be found in the program **src/test/example_atlas_fwd.F90**.

TELSEM2 MW atlas	latitude, longitude, zenangle
CNRM MW atlas	latitude, longitude, zenangle, skin%surftype
UWIRemis and CAMEL IR emissivity atlases	latitude, longitude, skin%surftype, skin%snow_fraction zenangle, sunzenangle – <i>sat/sun zenith angles only required with optional angular correction; sunzenangle only needs to be <85° (day) or >85° (night)</i>
BRDF atlas	latitude, longitude, skin%surftype, skin%watertype zenangle, sunzenangle, azangle, sunazangle

Table 8.8: Profile variables used by the emissivity and BRDF atlases.

8.3.7 BRDF atlas

RTTOV provides a BRDF atlas (Vidot and Borbas, 2013) which is similar in many ways to the IR emissivity atlases. It provides monthly climatological land surface BRDF values based on the MODIS BRDF MCD43 product including climatological snow. The atlas is called outside of RTTOV and the BRDFs are input to RTTOV in the **brdf_in(:)** member of **emis_refl** with the corresponding elements of **calc_brdf(:)** set to false. The BRDF atlas is valid for wavelengths in the range 0.4 - 2.5 μm . It returns zero (or negative) BRDFs for channels at wavelengths larger than 2.5 μm , but for channels at wavelengths below 0.4 μm , BRDFs are extrapolated at constant value to provide crude values for use in the UV.

Surface type:



- sea: returns Lambertian BRDFs from USGS ocean or freshwater spectra (Kokaly *et al.*, 2017) (no sun-glint)
- sea-ice: returns negative values
- land: returns BRDF values if it has data (inc. climatological snow), otherwise negative values

Input profile variables:

- mandatory: **latitude, longitude, skin%surftype, zenangle, sunzenangle, azangle, sunazangle**
- **skin%watertype** (used for sea surface types)

Outputs:

- BRDFs (negative values where atlas has no data)
- optional: directional-hemispherical (black-sky) albedo
- optional: quality flag

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

The interface to the BRDF atlas is similar to the emissivity atlases:

- **rttov_setup_brdf_atlas** : this allocates the necessary arrays and reads the data for a given atlas for a given month. The atlas data may be loaded for a specific sensor, or they can be initialised for use with any sensor with visible/near-IR channels.
- **rttov_get_brdf** : this returns surface BRDF values at a given latitude/longitude in the required channels. This should be called when populating the **brdf_in(:)** array for input to RTTOV.
- **rttov_deallocate_brdf_atlas** : deallocates atlas arrays and is called once the atlas is no longer required.

To initialise data for the BRDF atlas you must first declare a variable of type **rttov_brdf_atlas_data** which is defined in the module **rttov_brdf_atlas_mod**. This variable is passed into each of the above subroutines and to hold the atlas data for a specific month and, optionally, a specific instrument.

The interfaces of these subroutines are described in Annex G. Note that if an atlas does not contain BRDF data for the given surface type or lat/lon location it will return negative values for the corresponding BRDFs. You should check the returned BRDFs and handle negative values accordingly, for example by setting the corresponding elements of **calc_brdf(:)** to true. An example of using the atlas subroutines with a forward model call can be found in the program **src/test/example_atlas_fwd.F90**.

8.3.8 *Specular vs Lambertian surfaces*

By default, RTTOV treats the surface as a specular reflector for downwelling emitted atmospheric radiation (equation 2.1). The reflection of downward radiation over snow or multi-year sea-ice is better characterised by assuming a Lambertian approximation rather than specular reflection. This is particularly relevant for microwave sensors as sea-ice and snow have high reflectances at these frequencies. True Lambertian reflection requires an integral over a range of angles to cover the hemisphere which would be difficult and costly in the RTTOV framework. Matzler (2005) has developed an approximation as a function of optical depth providing a fixed angle of $\sim 55^\circ$ to be used for the downward radiation. The option of Lambertian reflection is possible in the MW and the IR. To invoke Lambertian reflection set the **opts%surface%lambertian** option to true. By default, it is false. Significant changes (up to 10K) will be seen in microwave window channels with the largest differences for nadir views. More details are provided in the RTTOV v11 Science and Validation Report.

RTTOV includes an option **opts%surface%lambertian_fixed_angle** to enable the parameterisation given in Guedj *et al.* (2010) for the downwelling angle in terms of the total atmospheric optical depth: if true (the default), the fixed angle of 55° is used, otherwise if false the parameterisation for the angle is used.



Most surfaces are not truly Lambertian or truly specular but lie somewhere between the two. To this end, the **rttov_emis_refl** structure has a **specularity(:)** member (Annex J) which is allocated and used when the **lambertian** option is true, and specifies the weighting applied when linearly combining the downwelling specular and Lambertian radiances. The valid range of values is 0 (fully Lambertian, default) to 1 (fully specular - the same as if **lambertian** is false). This specularity parameter is specified individually for each channel for each profile being simulated, and the specularity is an active variable in the TL/AD/K models.

Notes

For profiles/channels where an internal sea surface emissivity model is being used (i.e., SURFEM-Ocean or FASTEM-5/6 in the MW, and ISEM or IREMIS in the IR) the Lambertian option is not valid and so is not applied for these channels. When activated the Lambertian option is applied for sea surfaces where the corresponding elements of **calc_emis(:)** are false and for land and sea-ice surfaces regardless of **calc_emis(:)**.

For scattering simulations (section 8.4), the Lambertian option and specularity parameter can be used with the Chou-scaling solver (but not with the Tang modification). However, the Lambertian option is currently ignored by the DOM solver for which the surface is assumed to be fully Lambertian, and by the delta-Eddington solver which assumes specular reflection. The Lambertian option is also not compatible with PC-RTTOV simulations (section 8.6).

By default, RTTOV uses the linear-in-tau approximation when computing the Planck emission from each atmospheric layer (see the RTTOV v9 Science and Validation Report). This assumes that the Planck source radiance within a layer varies linearly with the optical depth through the layer which is more accurate for optically thick layers than the simpler

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

layer-average formulation that assumes the source is the mean of the Planck radiance evaluated at the upper and lower boundaries of each layer. The linear-in-tau approach is always used for upwelling radiances, and by default also for specular downwelling radiances. For downwelling radiances, the impact on top-of-atmosphere radiances of linear-in-tau vs layer-average is negligible because the two schemes only differ significantly for optically thick layers under which conditions the downwelling emission contributes little to the satellite-seen radiance. For non-specular simulations (**lambertian** option set to true), the downwelling radiance is *always* computed using the layer-average approach. If consistency is required for specular simulations (**lambertian** option set to false) the **opts%rt_all%rad_down_lin_tau** option can be set to false to use layer-average for downwelling radiances. This may also yield a small performance boost with only a small impact on simulated top-of-atmosphere radiances.

8.3.9 Per-channel effective skin temperature

RTTOV enables you to supply per-channel effective skin temperatures instead of a per-profile skin temperature that is used for all channels. Channels at different frequencies are sensitive to different depths below the nominal surface, and this option provides a way to capture the different effective skin temperature in each channel. This is primarily intended for experiments into coupling of atmospheric and sub-surface RT models for microwave sensors, but it applies across the spectrum if enabled. It is activated by setting the **opts%surface%use_tksin_eff** option to true.

In this case the **profiles(:)%skin(:)%t** input variable is ignored. Per-channel skin temperatures are input via **emis_refl(:)%tksin_eff(:)**. After direct model calls, **emis_refl(:)%tksin_eff(:)** always contains the skin temperatures used in the simulations (including when the **use_tksin_eff** option is false).

When the **use_tksin_eff** option is true, for the TL model, you can supply per-channel skin temperature perturbations in **emis_refl_tl(:)%tksin_eff(:)**, and for the AD/K models, the skin temperature adjoints/Jacobians are in **emis_refl_ad/k(:)%tksin_eff(:)**.

8.3.10 Heterogenous surfaces

RTTOV allows each profile to be associated with multiple surfaces, each of which has its own associated near-surface, skin, and emissivity/reflectance properties, and an associated area coverage fraction. This allows RTTOV to represent the case where the field of view falls over multiple surface types such as land and sea (in coastal areas), or sea and sea-ice. Surface properties are combined for all associated surfaces (see below), and the resulting values are used in a single evaluation of the radiative transfer equation. This is an approximation: a more accurate treatment would involve solving top-of-atmosphere radiances for all surfaces and combining the resulting radiances. This approximation is made for efficiency, and the impact of this is documented in the RTTOV v14 Science and Validation Report.

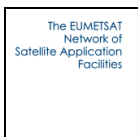
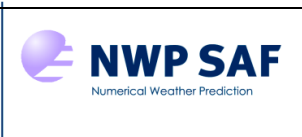
The number of surfaces, **nsurfaces**, may be any value but is typically either one or two in most use cases. It is defined when allocating data structures for RTTOV and is the same for all profiles being passed into RTTOV. The **near_surface(:)** and **skin(:)** array members of the **profiles(:)** structure (see section 7.4, Annex J) are of size **nsurfaces** allowing you to specify near-surface and skin parameters for each surface associated with each profile. The **emis_refl(:)** argument (Annex H) to RTTOV is an array of size **nsurfaces** which allows you to specify all surface emissivity and reflectance properties independently for each surface.

The **surface_fraction(:)** member of the **profiles(:)** structure is of size (**nsurfaces-1**). For cases where **nsurfaces>1**, you must specify the surface coverage fraction for surfaces 1, 2, ..., **nsurfaces-1**. RTTOV computes the fraction of the final surface as **(1 - SUM(surface_fraction(:)))** and in this way ensures that the sum of all surface fractions is one.

If you do not require the heterogenous surface capability, then set **nsurfaces=1**. In this case, the **near_surface(:)** and **skin(:)** members of the **profiles(:)** structure are of size one, as is the **emis_refl(:)** argument. You should not specify any **profiles(:)%surface_fraction(:)** because RTTOV automatically sets this to one in this case.

The combined surface leaving radiance for thermal channels is computed as shown in equation 8.3:

$$L_{surf} = \sum_{i=1}^{nsurfaces} f_i \epsilon_i B(T_{skin_i}) \quad (8.3)$$

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

where B is the Planck function, and f_i , ε_i , T_{skin_i} are the surface fraction, emissivity, and skin temperature associated with surface index i respectively.

The combined surface BRDF and diffuse reflectance are computed according to equation 8.4:

$$x_{surf} = \sum_{i=1}^{nsurfaces} f_i x_i \quad (8.4)$$

where x_{surf} is the total BRDF (or diffuse reflectance), and f_i and x_i are the surface fraction and BRDF (or diffuse reflectance) for surface index i respectively.

When using the **lambertian** option (section 8.3.8), it is applied to all surfaces. You can control whether each individual surface is treated as specular, Lambertian, or a mixture of the two, using the **specularity(:)** members of the **emis_refl(:)** structure array. For heterogeneous surfaces with the **lambertian** option, an additional diffuse reflectance is computed:

$$r_{surf,spec} = \sum_{i=1}^{nsurfaces} f_i s_i r_i \quad (8.5)$$

where r represents diffuse reflectance, f is the surface fraction, and s is the specularity. From equation 8.4 we have the usual specular diffuse reflectance r_{surf} . Equation 8.5 is the diffuse reflectance weighted by the specularity parameter s_i for each surface, $r_{surf,spec}$. When computing the downwelling radiance contribution to the TOA radiance for the **lambertian** option, r_{surf} is the reflectance used for the downwelling specular radiance, and $(r_{surf} - r_{surf,spec})$ is the reflectance used for the downwelling Lambertian radiance. It should be noted that for heterogeneous surfaces (**nsurfaces>1**) with the **lambertian** option, there is no clear relationship between the **dnclear** and **refldnclear** members of the **rttov_radiance2** secondary radiance output structure (section 7.8.2, Annex J) unlike in the case of fully specular or homogeneous surface (**nsurfaces=1**) simulations.

8.3.11 TL/AD/K for emissivity and reflectance

The **calc_emis**, **calc_brdf**, and **calc_diffuse_refl** members of the **rttov_emis_refl** structure (Annex J) are only allocated for the direct model **emis_refl** structure as they are not relevant in the TL/AD/K structures. Therefore, you should not try to set these arrays in the **emis_refl_tl/ad/k** structures that are passed to the TL/AD/K models.

The cases where emissivity/reflectance TL perturbations are calculated or where user input values are used depends on the setting of the relevant **calc_*(:)** variable (true/false), the surface type, the spectral region, and in some cases the selected surface model. This is summarised in Table 8.9 which provides an overview of the TL emissivity, BRDF, and diffuse reflectance inputs and calculations described in the subsections above.

Where an active model or calculation is carried out in RTTOV for the emissivity, BRDF, or diffuse reflectance, then the corresponding TL perturbation is computed from the relevant variables within RTTOV. In this case any input perturbations provided in the corresponding elements of the **emis_in(:)**, **brdf_in(:)**, and **diffuse_refl_in(:)** members of the **emis_refl_tl** structure are ignored in favour of the computed values. In these cases, for the AD/K model, the corresponding elements of the **emis_in(:)**, **brdf_in(:)**, and **diffuse_refl_in(:)** members of the **emis_refl_ad/k** structure are zero on output.

- For emissivity, this applies to sea surfaces when using SURFEM-Ocean, FASTEM-5/6, or IREMIS, and for land/sea-ice surfaces when using the FASTEM land/sea-ice parameterisation. Note that it does not apply to ISEM because this is only parameterised by zenith angle which is not an active TL/AD/K variable.
- For BRDF, this applies to the solar sea BRDF model for all solar-affected channels, and to mixed thermal+solar channels over land/sea-ice.
- For diffuse reflectance, this applies to land/sea-ice surfaces for all channels when **calc_diffuse_refl(:)** is true, and to sea surfaces for thermal channels when **calc_diffuse_refl(:)** is true.



In all other cases, a TL perturbation may be supplied in the corresponding elements of the **emis_in(:)**, **brdf_in(:)**, and **diffuse_refl_in(:)** members of the **emis_refl_tl** structure. In these cases, for the AD/K model, the corresponding elements of the **emis_in(:)**, **brdf_in(:)**, and **diffuse_refl_in(:)** members of the **emis_refl_ad/k** structure contain the computed emissivity, BRDF or diffuse reflectance AD/K sensitivity on output.

- For emissivity, this applies to sea surfaces when using ISEM, or otherwise where the user inputs emissivity values (**calc_emis(:)** set to false).
- For BRDF, this applies to land/sea-ice surfaces in pure solar channels where **calc_brdf(:)** is set to true, or otherwise where the user inputs BRDF values (**calc_brdf(:)** set to false).
- For diffuse reflectance, this applies to sea surfaces in pure solar channels where **calc_diffuse_refl(:)** is set to true, or otherwise where the user inputs diffuse reflectance values (**calc_diffuse_refl(:)** set to false).

In all cases, as for the direct model values, the TL perturbations used in the calculations (whether computed by RTTOV or input by the user) are stored in the **emis_out(:)**, **brdf_out(:)**, and **diffuse_refl_out(:)** members of the **emis_refl_tl** structure after the call to the TL model. For AD/K simulations, the **emis_out(:)**, **brdf_out(:)**, and **diffuse_refl_out(:)** members of the **emis_refl_ad/k** structure should usually be initialised to zero before calling RTTOV, and the **emis_in(:)**, **brdf_in(:)**, and **diffuse_refl_in(:)** members will contain the respective AD/K on output (see section 7.9 for more information about TL/AD/K simulations).

Variable	calc_* Boolean	Surface type	$\delta\epsilon, \delta b, \delta r$ values at channel wavelengths		
			<3 μm	3-5 μm	>5 μm
Emissivity TL $\delta\epsilon$	calc_emis T	Sea	N/A	IR: ISEM: user input $\delta\epsilon$ (<i>emis_refl_tl%emis_in</i>) IR: IREMIS: $\delta\epsilon$ computed MW: SURFEM-Ocean, FASTEM-5/6 $\delta\epsilon$ computed	
		Land		IR: user input $\delta\epsilon$ (<i>emis_refl_tl%emis_in</i>) MW: FASTEM land/sea-ice param $\delta\epsilon$ computed	
		Sea-ice		IR: user input $\delta\epsilon$ (<i>emis_refl_tl%emis_in</i>) MW: FASTEM land/sea-ice param $\delta\epsilon$ computed	
	calc_emis F	All		User input $\delta\epsilon$ <i>emis_refl_tl%emis_in</i>	
BRDF TL δb	calc_brdf T	Sea	Sunglint BRDF model δb computed + δr	Sunglint BRDF model δb computed	N/A
		Land	User input δb	$-\delta\epsilon/\pi$	
		Sea-ice	<i>emis_refl_tl%brdf_in</i>		
calc_brdf F	All	User input δb <i>emis_refl_tl%brdf_in</i>			
Diffuse reflectance TL δr	calc_diffuse_refl T	Sea	User input δr <i>emis_refl_tl%diffuse_refl_in</i>	IR: $-\delta\epsilon$ MW: $\delta\epsilon$ computed by SURFEM-Ocean/FASTEM-5/6 if <i>calc_emis T</i> otherwise $-\delta\epsilon$	
		Land	$\delta b * \pi$	$-\delta\epsilon$	
		Sea-ice			
calc_diffuse_refl F	All	User input δr <i>emis_refl_tl%diffuse_refl_in</i>			

Table 8.9: summarising the emissivity, BRDF, and diffuse reflectance TL inputs/calculations in each spectral region, and for each surface type. For most ocean surface models, where the **calc_*** variable is true, the corresponding TL variables are computed from the profile TL inputs for the variables used by the model such as 10m wind *u/v* components, skin temperature, salinity, etc. The exceptions are ISEM for emissivity and diffuse reflectance in VIS/NIR channels (<3 μm) over sea.

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

8.4 Scattering simulations

RTTOV provides the capability to simulate scattering by aerosols (affecting UV/VIS/IR sensors) and hydrometeors (all sensors). Note that the latter is distinct from the “simple cloud” treatment described in section 8.2. Unlike previous versions of RTTOV, scattering simulations for MW sensors are now run in a very similar way to IR sensors through the core RTTOV model. RTTOV provides a number of different passive solvers for computing thermal and solar radiances affected by scattering, and also a radar solver for MW hydrometeor simulations. For UV/VIS channels RTTOV provides an option to simulate Rayleigh multiple scattering. For hydrometeor simulations, a number of options are available to treat cloud overlap. Pre-defined sets of optical properties of aerosols and hydrometeors are provided via *aertable* and *hydrotable* files. Alternatively, you can input vertical profiles of explicit optical properties for each channel being simulated. This section describes these features, and how to run scattering simulations.

Example programs demonstrating different kinds of scattering simulation are available in the **src/test/** directory. These are listed in section 5.4.2 and are noted in the relevant sections below.

8.4.1 Aerosols and hydrometeors

Scattering simulations are run in the same way as the clear-sky (non-scattering) simulations described in section 7 but require some additional configuration options and inputs. Aerosols and hydrometeors are treated in a similar way, but there are some key differences.

Hydrometeor simulations are enabled by setting **opts%scatt%hydrometeors** to true. Currently different sets of pre-defined hydrometeor optical properties are available in the UV/VIS/IR and in the MW (section 8.4.7). Hydrometeor simulations require a cloud/hydrometeor fraction input representing the fractional area coverage of the cloud within the field of view. Radiances are computed for the clear column (no scattering), and for one or more cloud columns (determined by the cloud overlap parameterisation – section 8.4.3), and the resulting radiances are summed with suitable weights to give the final top of atmosphere radiance.

Aerosol simulations are enabled by setting **opts%scatt%aerosols** to true. Various sets of pre-defined aerosol optical properties are available (section 8.4.9) for UV/VIS/IR sensors. Aerosols are assumed to fill the field of view, and as such aerosol scattering is included in the “clear” radiances output from RTTOV (e.g., **radiance%clear**, **radiance%bt_clear**, **radiance%refl_clear** – see section 7.8.2), and in all cloud columns for simulations with both aerosols and hydrometeors.

Notes

Input hydrometeor and aerosol concentrations are grid box averages. These can typically be taken directly from NWP model fields.

All aerosol and hydrometeor scattering inputs in the top-most layer (layer 1) are silently ignored by RTTOV. This is because the top-most pressure half-level may be at or arbitrarily close to 0 hPa, in which case the thickness of the top layer is not well defined. The thickness of the layer is required when calculating the optical depth due to scattering particles. This applies when running simulations with pre-defined optical properties (sections 8.4.7, 8.4.9) and with explicit optical property inputs (section 8.4.11).

The scattering solvers that support MW sensors cannot simulate Stokes 3/4 channels. Therefore, when scattering is enabled, the radiances and brightness temperatures are zero for these polarimetric channels, but scattering is computed for all other channels in the simulation.

See section 7.8 for information on RTTOV outputs and how these relate to scattering simulations.

8.4.2 Scattering solvers

Separate scattering solvers are implemented for thermal and solar radiation, and they are selected independently. The thermal solver is specified by the **opts%scatt%thermal_solver** option, and the solar solver (activated only when **opts%rt_all%solar** is true) is specified by the **opts%scatt%solar_solver** option. See section 8.1 for more information

on solar radiation. These solvers are only activated when either **opts%scatt%hydrometeors** and/or **opts%scatt%aerosols** is true.

opts%scatt%thermal_solver	Description
thermal_solver_dom (1)	DOM thermal solver.
thermal_solver_chou (2)	Chou-scaling thermal solver.
thermal_solver_delta_edd (3)	Delta-Eddington thermal solver (default).
opts%scatt%solar_solver	Description
solar_solver_dom (1)	DOM solar solver (default).
solar_solver_mfasis_nn (2)	MFASIS-NN solar solver.

Table 8.10: Options for thermal and solar solvers. The constants corresponding to the integer values are in the *rttov_const* module (Annex K).



Discrete Ordinates Method (DOM) – thermal emission and solar source terms

The DOM solver is a pseudo-exact algorithm for solving the monochromatic radiative transfer equation. The implementation of the DOM algorithm is described in the RTTOV v12 Science and Validation Report. DOM can be applied for thermal emission at IR wavelengths, and, separately, for solar radiation. The number of DOM streams determines the accuracy of the simulation but increasing the number of streams can dramatically increase the run-time. The number of streams is set in **opts%scatt%dom_nstreams**: this should be an even integer greater than or equal to 2 (odd values are rounded up). The default value is 8. This value should not exceed the number of Legendre coefficients for the phase functions provided as input (see below). The *hydrotable* and *aertable* files contain 32 coefficients for every phase function, so this is the maximum number allowed when using these pre-defined particle types. If you supply your own optical properties (described below) there is no fixed upper limit. You may wish to experiment to find the most suitable value for your application.

Important notes:

- By default, RTTOV calculates the path of radiation through the atmosphere accounting for the curvature of the Earth (and optionally also refraction). The DOM algorithm as implemented requires a strictly plane-parallel atmospheric geometry. **Therefore, if the DOM solver is being used either for thermal emission or solar radiation calculations then the strict plane-parallel geometry is applied for all simulated radiances (including clear-sky radiances).** It is possible to turn on the strict plane-parallel geometry for any simulation by setting **opts%rt_all%plane_parallel** to true, but it is not necessary to do so when using DOM as RTTOV does this automatically.
- When using DOM, the surface is treated as a Lambertian reflector (this is independent of the **lambertian** option and does *not* make use of the **specularity** parameter – see section 8.3.8). When solving for thermal emission the diffuse reflectance is used for surface reflected radiation. When solving for solar radiation the reflectance is equal to the BRDF multiplied by π : if this value exceeds one the albedo is capped at one. See section 8.3 for more information on surface emissivity and reflectance.
- The DOM algorithm is relatively slow and has relatively large memory requirements. The number of DOM streams and the number of layers in the input profile have a direct impact on the speed. In addition, the more layers containing scattering particles, the slower it runs. There are two parameters which can be used to increase efficiency:
 - **opts%scatt%dom_accuracy**: for solar radiation the azimuthal loop within the DOM algorithm will exit early if the computed radiance increment is smaller than this fraction of the total radiance (it exits once this condition has been met twice). This is identical to the accuracy parameter in the DISORT model (Stamnes *et al.*, 1998). The default is 0. (i.e., no truncation). This option should be used with some caution: it is recommended that you carry out sensitivity tests.
 - **opts%scatt%dom_opdep_threshold**: if this value is greater than zero, then the DOM solver excludes any atmospheric layers below the level at which the absorption optical depth to space exceeds this value. This typically only affects IR channels where there is significant atmospheric absorption. A value of around 10 can decrease run-time noticeably for sounding channels with negligible impact on radiances. The default is 0 (i.e., no layers are excluded).

For solar radiation, the DOM solar solver automatically combines adjacent clear layers (i.e., layers containing no scattering particles). This can reduce the run-time significantly by reducing the total number of layers processed by the DOM solver. This optimisation does not apply when Rayleigh multiple scattering is enabled because all layers contain scattering particles for affected channels (see section 8.4.5). This optimisation is not

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

applied in the thermal solver because each layer typically has a different temperature and therefore also emission source term, and so they must be treated individually.

The optical properties of scattering particles required for this solver are:

- extinction coefficient
- single-scattering albedo
- at least `opts%scatt%dom_nstreams` coefficients of the Legendre decomposition of the phase function
- for solar radiation the full phase function is also required

Chou-scaling – thermal emission source term only

This is the fast multiple-scattering parameterisation for IR instruments available in previous versions of RTTOV and described in the RTTOV v9 Science and Validation Report (available on the NWP SAF web site). This solver cannot be used for MW sensors.

The modification to Chou-scaling described by Tang *et al.* (2018) has been implemented in RTTOV. This is particularly intended to improve the accuracy of the Chou-scaling approximation for ice clouds in the far-IR. The Tang modification is enabled by setting the `opts%scatt%chou_tang_mod` option to true. The Tang correction involves a multiplicative factor which is set via the `opts%scatt%chou_tang_factor` option. This defaults to 0.05 which is an empirically derived value, and it is not necessarily recommended to change this, but you may wish to experiment. The implementation of the Tang correction in RTTOV is described in Labonnote *et al.* (2022). The Tang correction should be considered an experimental feature that will be investigated and developed further in future releases. It is currently recommended primarily for ice clouds in the far-IR, but if activated it is applied to all scattering simulations. Its applicability to the short-wave and mid-IR, and to liquid water clouds and aerosols is being investigated.

The optical properties of scattering particles required for this solver are:

- extinction coefficient
- single-scattering albedo
- the “b” parameter or “bpr” (the fraction of back-scattered radiation) computed from the phase function



Delta-Eddington – thermal emission source term only

This is the fast multiple-scattering parameterisation originally implemented in the RTTOV-SCATT MW scattering model in earlier RTTOV releases (Bauer *et al.*, 2006). Further information on the historical development of RTTOV-SCATT including optical properties and cloud overlap treatment (all now implemented in RTTOV) can be found in the RTTOV v8, v9, v10, v11 and v13 Science and Validation Reports. In RTTOV v14 the delta-Eddington solver can be used for IR and MW sensors, and for hydrometeor and/or aerosol scattering simulations.

The delta-Eddington solver is a plane-parallel solver but does not internally enforce strictly plane-parallel calculations in the same way as DOM and as such does not impose this on the solar solver being used. By default (unless the `opts%rt_all%plane_parallel` option is true), the clear-sky optical depths are computed as usual, accounting for Earth curvature, and optionally atmospheric refraction.

The delta-Eddington solver internally applies a maximum to the computed total layer extinction of gas and hydrometeors. When this occurs, it is notified via the `qflag_delta_edd_ext_limits` quality flag (see section 7.8.3). This does not necessarily imply that simulated radiances have larger errors but rather is for information purposes.

By default, hydrometeor (and aerosol) simulations return zero hydrometeor (aerosol) AD/K sensitivity in layers where the input hydrometeor (aerosol) concentration is zero. For MW hydrometeor simulations using the pre-defined optical properties (section 8.4.7) and using the delta-Eddington solver with any two-column cloud overlap scheme (section 8.4.3), you can set the `opts%scatt%zero_hydro_tl` option to true to enable TL/AD/K sensitivity to hydrometeor concentration in all layers, even where the input concentration is zero. This also applies to the radar solver if enabled (section 8.4.4). NB This option has no effect if units of flux are used for rain/snow inputs (section 8.4.7) as the flux conversion is not well defined for zero hydrometeor concentration.

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

The optical properties of scattering particles required for the delta-Eddington solver are:

- extinction coefficient
- single-scattering albedo
- the asymmetry parameter computed from the phase function

MFASIS-NN – solar source term only

The MFASIS-NN fast VIS/NIR scattering parameterisation (Scheck, 2021) is a neural network-based solver that is trained on RTTOV simulations with the DOM solar solver with Rayleigh multiple scattering enabled. Currently it supports only hydrometeor simulations in selected VIS/NIR channels. Support for aerosol simulations is planned for a future release.

MFASIS-NN must be used with the pre-defined hydrometeor optical property files provided by the NWP SAF (section 8.4.7). It cannot be used with the explicit optical property inputs (section 8.4.11). In addition to the hydrotable file, an MFASIS-NN coefficient file (with prefix *rttov_mfasis_nn*) must be read in. All files are read in the same call to **rttov_read_coefs** (see section 7.2 and Annex C). Hydrometeors must be enabled (**opts%scatt%hydrometeors=true**) and MFASIS-NN must be selected as the **opts%scatt%solar_solver** before reading the coefficients. You can change, for example, the selected solar solver later in your code if desired.

The example program **src/test/example_hydro_mfasis_nn_fwd.F90** demonstrates a call to the MFASIS-NN direct model and can be used as a template for your own programs.

In RTTOV, MFASIS-NN simulations are performed in the same way as other hydrometeor simulations through calls to the RTTOV direct/TL/AD/K models, but there are a few key differences:

- When simulating a channel using MFASIS-NN, the RTTOV gas optical depth parameterisation is not called for that channel. If simulating *only* VIS/NIR channels with MFASIS-NN, *all* calculations related to the gas optical depth parameterisation are omitted (including profile interpolation, see section 7.4.1) which can result in faster simulations.
- When running MFASIS-NN simulations, you can also simulate IR cloudy radiances with any of the available thermal solvers. However, solar radiation is treated only for channels supported by the MFASIS-NN coefficient file, and any solar-affected channels not supported will exclude solar radiation entirely. This means that output radiances and reflectances for unsupported VIS/NIR channels will be zero, and short-wave IR channels (not supported by MFASIS-NN) will have no solar contribution.
- Hydrometeor concentrations can be supplied for any of the pre-defined UV/VIS/IR hydrometeor types (see section 8.4.7), but MFASIS-NN is not trained using the Baran ice scheme so any hydrometeor concentrations corresponding to the Baran scheme will be ignored: only Baum ice optical properties are used. There are no restrictions on the cloud liquid water types.
- The MFASIS-NN code has been particularly optimised for vector architectures. To take advantage of this it is recommended to pass in as many profiles as possible per call.

MFASIS-NN is trained using the DOM solver and as such assumes a Lambertian surface with albedo given by π times the surface BRDF and this value is capped at one, as for DOM (see above). The surface BRDF options are the same as for other RTTOV calls.

Most MFASIS-NN neural network files are trained over the full range of parameters accepted by RTTOV and applicable to the optical property files. The one current exception is the range of zenith angles used in the training for sensors with specific viewing geometries (notably the DSCOVER EPIC sensor located at the Lagrange L1 point). For these latter cases, a quality flag (see section 7.8.3), **qflag_mfasis_nn_limits**, is defined which is set if the zenith angle limits used in training the neural network are exceeded.

The **rttov_user_check_profile** subroutine (see section 7.7 and Annex I) has an optional **do_mfasis_nn** logical argument, which if true will exclude checks on profile variables not relevant to MFASIS-NN. In practice this is currently only the skin temperature.

8.4.3 Cloud overlap

For hydrometeor simulations, the input grid box average concentration of hydrometeors in each layer is associated with an area coverage fraction (**hydro_frac**), and an assumption must be made about how to treat the cloud overlap between layers. In the general case, the cloud overlap parameterisation generates one or more “cloud columns”, each with a unique distribution of hydrometeors among the layers. Each column has an associated weight. The weights for all columns, including that for the “clear column” (with no hydrometeors), sum to one. Top of atmosphere radiances are computed for every cloud column individually using the selected solver(s), and the final output radiance is the sum of these multiplied by their corresponding weights. If aerosols are enabled, they are included in the clear column and all cloud columns.

By default, only one input **hydro_frac** is specified per layer. The first dimension of the **profiles(:)%hydro_frac(:,:)** array is of size 1. All hydrometeors present in each layer are assumed to be well-mixed within the cloudy fraction of each layer. If the **opts%cloud_overlap%per_hydro_frac** option is set to true, then the dimensions of the **hydro_frac(:,:)** array are the same as those of the **hydro(:,:)** array (see section 8.4.7). In this case an individual **hydro_frac** may be specified for each individual hydrometeor type. This **per_hydro_frac** option is only compatible with the two-column hydrometeor weighted cloud overlap option (see below), and the radar solver (section 8.4.4).

RTTOV provides several cloud overlap parameterisation options selected via the **opts%cloud_overlap%overlap_param** option. These options are summarised in table 8.11 and described below.

opts%cloud_overlap%overlap_param	Description
cloud_overlap_auto_select (0)	Automatic selection of overlap_param at run-time: for VIS/IR sensors use max/random and for MW sensors use two-column hydro-weighted scheme (default).
cloud_overlap_max_random (1)	Maximum-random overlap, recommended for VIS/IR sensors.
cloud_overlap_2col_max_frac (2)	Two column, effective frac is maximum value among layers at pressures below opts%cloud_overlap%two_col_max_frac_max_p .
cloud_overlap_2col_weighted (3)	Two column, effective frac is computed as hydrometeor-weighted average, recommended for MW sensors.
cloud_overlap_2col_user_frac (4)	Two column, user specifies effective frac in hydro_frac_eff profile variable.

Table 8.11: Options for cloud overlap parameterisations. The constants corresponding to the integer values are in the *rttov_const* module (Annex K).

Maximum-random overlap

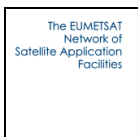
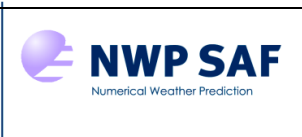
This is described RTTOV v9 Science and Validation Report. This is the most expensive overlap option because it can generate many cloud columns. For UV/VIS/IR simulations, it is generally the recommended option, because the two-column schemes described below are only accurate in a restricted range of scenarios for these sensors.

When running the tangent linear (TL), adjoint (AD), or K models, you are advised to avoid specifying layers with a **hydro_frac** equal to 1.0. Instead, a value very close to 1.0 should be used (e.g., 0.999999). In addition, you are advised not to specify identical values of **hydro_frac** on adjacent layers. The reason for this is that the **hydro_frac** Jacobians are very sensitive to perturbations in these cases: the direct model is not differentiable for fully overcast layers or where identical values of **hydro_frac** are in adjacent layers. If this advice is not followed, RTTOV makes very small adjustments to the input **hydro_frac** profile in accordance with the above advice to ensure consistency between the direct, TL, AD and K models. These adjustments are sufficiently small to have negligible impact on the direct model radiances, but they do introduce additional cloud columns. These recommendations on the values in **hydro_frac** do not apply when running the direct model alone.

Two-column overlap schemes

The remaining overlap schemes are two-column schemes meaning that they compute or use a single effective hydro fraction (*C*) for the whole profile, and there is just one cloud column in addition to the clear column. The total top of atmosphere radiance is computed as the weighted sum of the clear and cloudy radiances as shown in equation 8.6.

$$L_{Total} = (1 - C)L_{clr} + CL_{cld} \quad (8.6)$$

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

The effective hydro fraction used in the simulation is available via the diagnostic output structure in the **hydro_frac_eff(:)** member (Annex J).

The two column schemes differ only in how the effective hydro fraction is computed:

Maximum fraction

This computes the effective hydro fraction as the maximum value in the input **profiles(:)%hydro_frac(1,:)** profile among all layers at pressures less than **opts%cloud_overlap%two_col_max_frac_max_p**. By default, this threshold is set to 750 hPa: this is intended for fast simulation of cloudy radiances in upper tropospheric humidity channels for IR sensors, but it should be used with caution. It may also be useful for VIS channels in cases with thick cloud. In that case the **two_col_max_frac_max_p** threshold may be increased to cover all layers in the profile (e.g., by setting it to a large value like 1100 hPa).

Hydrometeor-weighted fraction

The effective hydro fraction is computed from the input **profiles(:)%hydro_frac(:,:)** weighted by the corresponding hydrometeor concentrations. This is the same as the default scheme implemented in RTTOV-SCATT in previous versions of RTTOV and is the recommended option for MW sensors. It is documented in Geer *et al.* (2009a,b). This is a recommended approach for MW sensors when speed is important. By default, the effective hydro fraction for each profile is computed as:

$$\frac{\sum_{i=1}^{nhydro} \sum_{j=1}^{nlayers} q_{i,j} \cdot f_j \cdot \Delta z_j}{\sum_{i=1}^{nhydro} \sum_{j=1}^{nlayers} q_{i,j} \cdot \Delta z_j}$$

where $q_{i,j}$ is the hydrometeor concentration for hydro index i in layer j , f_j is the hydro fraction in layer j , and Δz_j is the thickness of layer j (as computed internally by RTTOV). If the **per_hydro_frac** option is set to true whereby one hydro fraction is input per hydrometeor type in each layer (see above) then the effective hydro fraction for each profile is computed as:

$$\frac{\sum_{i=1}^{nhydro} \sum_{j=1}^{nlayers} q_{i,j} \cdot f_{i,j} \cdot \Delta z_j}{\sum_{i=1}^{nhydro} \sum_{j=1}^{nlayers} q_{i,j} \cdot \Delta z_j}$$



where $f_{i,j}$ is the hydro fraction for hydro index i in layer j .

User input effective hydro fraction

By selecting this **overlap_param** option, you specify the effective hydro fraction explicitly for each profile in the **profiles(:)%hydro_frac_eff** profile member (see Geer *et al.*, 2009b).

Additional cloud overlap options

The **opts%cloud_overlap%col_threshold** option can be used to set a threshold such that columns with weights below this value are ignored. The weights corresponding to these columns are added to the clear column weight. This feature can be used to speed up hydrometeor scattering simulations by avoiding calculations for columns with negligible weights. For the maximum/random overlap option (see above), it is not recommended to set this to a value larger than approximately 1E-4. For two-column overlap schemes (see above), in particular with MW sensors, a value of 1E-3 may be reasonable. For two-column schemes, there is only one effective cloud fraction: if this is less than the **col_threshold**, then the clear-sky radiance is returned. It is recommended to conduct a sensitivity study to understand the implications of the **col_threshold** setting for accuracy in your given application. By default, the **col_threshold** option is zero meaning all columns are included in the simulation. It is also worth noting that cloud overlap parameterisations are typically non-differentiable in certain conditions and setting **col_threshold** to a positive value can increase the range of situations in which this occurs, so you may wish to leave **col_threshold** at zero when calling the TL, AD, or K models (see the notes above related to the maximum/random overlap option).

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

For the two-column hydrometeor weighted cloud overlap option (see above), the hydrometeor concentration is used in computing the effective hydro fraction, and, by default, there is therefore a sensitivity to the effective hydro frac in the hydrometeor concentration TL/AD/K. This can result in very jagged hydrometeor Jacobians. The **opts%cloud_overlap%hydro_frac_tlad** option may be set to false to disable this sensitivity: it may result in smoother Jacobians but note that it breaks the consistency between the direct model and the TL/AD/K models. This option is recommended for advanced users only. The setting of this option is ignored for other cloud overlap options.

8.4.4 Radar simulator

An optional feature of RTTOV is to output vertical profiles of radar reflectivities for MW hydrometeor scattering simulations alongside the passive radiance outputs. Radar simulations are enabled by setting **opts%scatt%radar** to true when carrying out a MW hydrometeor scattering simulation. Radar simulations require a hydrotable file containing reflectivity data for the sensor (section 8.4.7). An additional argument, **reflectivity** (of derived type **rttov_reflectivity**) must be supplied to **rttov_direct** (Annex H). This contains the output reflectivity and attenuated reflectivity profiles at each full-level pressure (section 7.4.1) for each channel in units of dBZ (Annex J). In layers containing no hydrometeors, the reflectivities are set to the constant **min_reflectance** = -999 dBZ (in the **rttov_const** module, Annex K). The geometric heights corresponding to the pressure full-levels can be obtained via the diagnostic output structure (**diag_output%geometric_height(:,:)**, section 7.8.4, Annex J). It is up to the user to interpolate from these levels onto the levels (range gates) of the relevant sensor.

For consistency with the delta-Eddington solver (section 8.4.2), the reflectivity calculation itself (based on hydrometeor optical properties) is plane-parallel, but the clear-sky optical depths account for the Earth curvature (and optionally refraction) unless the **opts%rt_all%plane_parallel** option is true.

The cloud overlap parameterisation (section 8.4.3) is ignored in the radar solver. By default, the layer hydrometeor concentrations are always scaled by the corresponding layer **profiles(:)%hydro_frac(1,:)**. In the calculation of attenuation, the extinction on any layer from any hydrometeor type is also scaled by the corresponding **hydro_frac(1,:)**. This is equivalent to assuming random cloud overlap between levels (in other words, full mixing of clear and cloudy columns). If the **opts%scatt%per_hydro_frac** option is true (section 8.4.3), then both hydrometeor concentrations and extinctions are scaled by the **hydro_frac(:,:)** for the corresponding hydrometeor type in each layer.

Care needs to be taken when supplying a single profile of hydro fractions (i.e., when **per_hydro_frac** is false). Quite often forecast models generate zero cloud fraction below cloud, but precipitation is still present. In this case, the cloud fraction may need to be modified by the user to become a precipitation fraction, otherwise zero reflectivity will be generated. For best results, supply a full set of hydrometeor fractions appropriate to the forecast model.

The **opts%scatt%zero_hydro_tlad** option (see delta-Eddington solver in section 8.4.2) applies to the radar solver.

The radar solver behaves in the same way as other types of simulations for TL/AD simulations (see section 7.9). For the K model there are some restrictions. The RTTOV Jacobian outputs are of size **nchanprof**, i.e., one Jacobian per channel simulated. However, the full radar reflectivity Jacobian matrix is of size **nchanprof * nlayers** because the radar direct model computes reflectivities for each pressure full-level (layer). Therefore, to construct the full reflectivity Jacobian you must call **rttov_k** multiple times, providing input reflectivity K perturbations in one layer at a time, with all other layer perturbations set to zero. Input perturbations may be specified in unattenuated or attenuated reflectivity (i.e., in **reflectivity_k%zef(i,:)** or **reflectivity_k%azef(i,:)** for layer **i**), and the other reflectivity K member should be initialised to zero. When computing radar Jacobians, all **radiance_k** members must be set to zero. To compute passive radiance Jacobians, follow the advice in section 7.9. In this case either deactivate the radar simulator or otherwise ensure that all **reflectivity_k** members are initialised to zero.

One final caveat for the radar solver is that there is no parameterisation of multiple scattering. This is expected to cause only small errors up to CloudSat frequency (e.g., around 90 GHz). This may be improved in a future release of RTTOV.

8.4.5 Rayleigh multiple scattering

As described in section 8.1.3 RTTOV includes a single-scattering approximation for atmospheric (molecular) Rayleigh scattering. By default, this is also applied in the case of hydrometeor and aerosol scattering simulations. However, when using this parameterisation with the DOM solver, multiple interactions between the Rayleigh single-scattered radiation and the hydrometeor or aerosol particles are not accounted for. This is done for reasons of efficiency, but in the case of thick cloud, for example, this can result in significant under-estimation of the TOA radiance in visible channels, especially for larger satellite or solar zenith angles.

RTTOV provides the option for full Rayleigh multiple scattering with the DOM solar solver. This option can only be used with v13 predictor optical depth coefficient files because Rayleigh extinction is not included in the training simulations as described in section 8.1.3. To enable Rayleigh multiple scattering, set the **opts%scatt%rayleigh_multi_scatt** option to true and run your DOM simulation (with aerosol and/or hydrometeors, using the RTTOV pre-defined optical properties or explicit optical properties) as normal. You can perform clear-sky Rayleigh multiple scattering calculations in the same way but set the input hydrometeor or aerosol concentrations to zero (at least one of **opts%scatt%hydrometeors** or **opts%scatt%aerosols** must be true to activate the DOM solver). Setting the **rayleigh_multi_scatt** option to true overrides the **rayleigh_single_scatt** option for scattering simulations with the DOM solar solver.

Enabling Rayleigh multiple scattering precludes the merging of adjacent “clear” (non-scattering) layers as noted in section 8.4.2, and so this option can increase run-times significantly. As such this option is primarily intended for off-line/non-operational applications. To help mitigate this, the **rayleigh_max_wavelength** option can be used to ensure Rayleigh scattering is included only in relevant channels, and the **rayleigh_min_pressure** option can be used to exclude Rayleigh scattering from the top-most layers of the atmosphere which contribute the least to the top of atmosphere radiance (thus allowing these “clear” layers to be combined as per the optimisation described in section 8.4.2). These two options are described in section 8.1.3.

8.4.6 Treatment of polarised scattering for MW sensors

RTTOV currently offers two approximate schemes to improve the representation of polarized scattering for MW hydrometeor simulations. Hydrometeor optical properties (in the hydrotable files) are not normally polarised, and hence for MW sensors are stored per discrete frequency, not per channel. To approximate the effect of preferentially oriented ice hydrometeors in creating polarisation effects, the optical properties can be varied as a function of channel polarisation and (in the case of the newer scheme) zenith angle too.

The polarisation option is selected via the **opts%scatt%mw_pol_mode** option. The options are given in table 8.12.

opts%scatt%mw_pol_mode	Description
mw_pol_mode_no_pol (0)	Disable polarised scattering
mw_pol_mode_empirical (1)	Empirical scaling of extinction for V and H polarised channels, scale factor controlled by opts%scatt%ice_polarisation option (default)
mw_pol_mode_aro_scaling (2)	ARO scaled polarisation for V, H, QV, QH polarised channels

Table 8.12: Options for MW polarisation treatment. The constants corresponding to the integer values are in the *rttov_const* module (Annex K).

The default and older polarisation scheme (Barlakas *et al.*, 2021) is fully empirical and is active when **mw_pol_mode = mw_pol_mode_empirical**. It models the differences in extinction between V and H channels in conical scanning MW radiometers. In this scheme, the extinction of frozen particles (snow, graupel, and cloud ice, in the default microphysical setup, see section 8.4.7) is increased in horizontally polarised channels and decreased in vertically polarised channels by the factor α . This is set by **opts%scatt%ice_polarisation**, which specifies the polarisation ratio $(1 + \alpha)/(1 - \alpha)$. The value is applied globally and the default value of 1.40 ($\alpha = 0.166667$) has been tuned to give the best fit to observations from GMI (Barlakas *et al.*, 2021).

A newer, more physically based scheme (Barlakas *et al.*, 2022) is optionally available with **mw_pol_mode = mw_pol_mode_aro_scaled**. Additional benefits of this scheme are that it is applicable to cross-track sounders as well as to conical microwave imagers, and that it adjusts the single scattering albedo and asymmetry as well as the extinction. There is no scaling defined for Legendre coefficients so this option cannot be used with the DOM solver. It includes code

to rotate the scaling factors according to the zenith angle and base polarisation of cross track sounders. It is based on a lookup table of scaling factors between optical properties generated using Azimuth Random Orientation (ARO) particles compared to the standard Totally Random Orientation (TRO) particles used in the hydrotables. However, it has not yet seen much testing, so it is not recommended as the default, but only for scientific exploration at this stage. The lookup table is sensor-independent and must be loaded at the same time as the hydrotable by setting `opts%scatt%mw_pol_mode = mw_pol_mode_aro_scaled` before the call to `rttov_read_coefs` (see section 7.2 and Annex C).

The empirical and ARO scaled polarisation options are *not* applied if using explicit optical properties (section 8.4.11). RTTOV will run in this case, but the `mw_pol_mode` setting will be ignored.

Hydrometeor polarisation can be turned off completely using `opts%scatt%mw_pol_mode = mw_pol_mode_no_pol`. It is also possible to gain some further control of polarisation, by using hydrotables with per-channel (i.e., polarised) optical properties. In this case the internal polarisation must be turned off (`opts%scatt%mw_pol_mode = mw_pol_mode_no_pol`) since the polarisation of optical properties will then come from the hydrotable file. For this option, controls in the hydrotable generation software can be activated to allow the polarisation of extinction to be set on a per-hydrometeor basis, and in future, scattering databases will be made available to represent (azimuthally random) preferentially oriented particles. One drawback is that this option is only suitable for V and H polarisations and polarisation effects cannot be rotated to follow the polarisations of cross-track sounders. This option is further documented in the `readme.txt` of the hydrotable generation software (see section 8.4.8).

8.4.7 Predefined hydrometeor optical properties

RTTOV hydrotable files (prefix `rttov_hydrotable_`) contain optical properties for an arbitrary collection of particle types. The hydrotable is read in at the same time as the gas optical depth coefficient file (section 7.2). You then supply vertical profiles of hydrometeor concentrations in the `profiles(:)%hydro(1:nhydro, 1:nlayers)` array. The hydrotable contains optical properties for `nhydro` particle types, and the first index of the `hydro(:, :)` array corresponds to the particle type in the order in which they are defined in the hydrotable.

The `profiles(:)%mmr_hydro` logical flag selects the units: if true hydrometeor concentrations are supplied as mass mixing ratio (kg/kg), and if false, as density (g/m³). All profiles passed into RTTOV in a single call must use the same units. Also note the option for inputting rain/snow fluxes described below. Input hydrometeor concentrations are grid box average values which is typically what NWP model cloud/hydrometeor fields represent.

Currently, the hydrotable files supplied for MW sensors and UV/VIS/IR sensors contain optical properties for different sets of particles. In the future it is planned to provide consistent optical properties across the whole spectrum.

MW sensors

The supplied hydrotables for MW sensors contain optical properties for five particle types in the following order: rain, snow, graupel, cloud liquid water, cloud ice water (see table 8.13). The generation of the bulk optical properties in the hydrotables is documented by Geer *et al.* (2021). A tool is available enabling you to generate your own hydrotables for MW sensors (section 8.4.8).

Constants are available in `rttov_const` (Annex K) giving the index for each particle type in the supplied hydrotables. It is recommended to use these in your code.

Index	Particle type	Description
<code>hydro_index_rain = 1</code>	Rain	Mie sphere, Marshall-Palmer size distribution.
<code>hydro_index_snow = 2</code>	Snow	ARTS large plate aggregate, Field07 tropical size distribution.
<code>hydro_index_graupel = 3</code>	Graupel	ARTS column, Field07 tropical size distribution.
<code>hydro_index_clw = 4</code>	Cloud liquid water	Mie sphere, Gamma size distribution implemented within the new modified gamma framework.
<code>hydro_index_ciw = 5</code>	Cloud ice water	ARTS large column aggregate, Gamma PSD with generalised modified gamma parameters $\mu = 0$, $\lambda = 1e4$, $\gamma = 1$ and N_0 free.

Table 8.13: hydrometeor types in the NWP SAF hydrotables for MW. The `hydro_index_*` constants are available in `rttov_const` (Annex K).

For rain and snow particle types, while it is recommended to input concentrations in kg/kg or g/m³ (as described above), it is possible to input concentrations as fluxes (kg/m²/s). This is activated by setting the elements of **profiles(:)%flux_conversion(1:nhydro)** corresponding to the relevant particle types in the hydrotable file to **flux_conv_rain** (numerical value 1, for rain fluxes) or **flux_conv_snow** (numerical value 2, for snow fluxes). Other elements should be left as **flux_conv_none** (numerical value 0, units according to **mmr_hydro**). The constants **flux_conv_*** are specified in **rttov_const** (Annex K). The **profiles(:)%flux_conversion(:)** array must be set for every profile being simulated. The internal conversion from flux to g/m³ is made using assumptions of particle size distribution, density, and fall speed specific to rain or snow. This is strongly deprecated and may be removed in a future release as fall speed assumptions belong outside of RTTOV.

The main MW hydrotables contain extinction coefficients, single scattering albedo, and asymmetry parameter. These are compatible only with the delta-Eddington thermal solver. Hydrotables are also available for some sensors with active radar instruments such as CloudSat CPR and GPM DPR. These also include reflectivity and may be used with the radar solver (section 8.4.4).

Optical properties in the MW hydrotables are parameterised by temperature and by hydrometeor concentration (water content) and are interpolated based on the temperature and water content in each layer. Quality flags are set (**qflag_hydro_t_limits**, **qflag_hydro_conc_limits**) when the minimum/maximum temperature limits are exceeded, or when the maximum water content limit is exceeded in some layer for some hydrometeor type (see section 7.8.3 for more information on quality flags). Note that the temperature flag in particular is for information only and can generally be ignored. The limits are defined separately for each particle type and note that the water content limit is based on the hydrometeor concentration in the cloudy fraction of the layer rather than the input grid box average value.

For MW sensors with multiple channels at the same frequency with different polarisations, usually a single table of (unpolarised) optical properties is stored for all channels at that frequency. This reduces the size of the hydrotables considerably. A similar approach is taken for hyperspectral MW sensors whereby optical property tables are stored at a subset of frequencies across the spectral range of the sensor and individual channels use the table for the nearest frequency. This exploits the fact that optical properties change relatively slowly in frequency in the MW spectral region. For sensors on different platforms with the same channels (e.g., AMSU-A on various NOAA and Metop platforms), only one hydrotable is generated (e.g., *rttov_hydrotable_noaa_amsua.dat*) without an individual satellite ID. This can be used for the given sensor (AMSU-A in this example) on all platforms, for example by using symbolic links.

Particle type	Parameterisation limits			
	Min temperature:	Max temperature:	Min water content:	Max water content:
Rain	233 K	303 K	0.001 g/m ³	10 g/m ³
Snow	203 K	273 K	0.001 g/m ³	10 g/m ³
Graupel	203 K	273 K	0.001 g/m ³	10 g/m ³
Cloud liquid water	233 K	303 K	0.001 g/m ³	10 g/m ³
Cloud ice water	203 K	273 K	0.001 g/m ³	10 g/m ³

Table 8.14. Summarising parameterisation limits associated with each MW hydrometeor type. Water content minima are not applied in the code. Water contents refer to the concentration in the cloudy fraction of each layer, not to the input grid box average values.

UV/VIS/IR sensors

The supplied hydrotables for UV/VIS/IR sensors contain optical properties for seven particle types. In addition, an eighth particle type is available based on the ice properties of the Baran scheme (see below). The Baran scheme is parameterised in the code, so no data is stored in the hydrotable for this particle type. The optical properties are the same as those available with previous versions of RTTOV. The difference is that all particle types may now be used together in any combination without limitation, as has been true for aerosol and MW hydrometeor simulations for some time. The optical properties available are listed in Table 8.15.

In this case, the number of hydrometeors in the hydrotable file is 7, but the first index of the **profiles(:)%hydro(:,:)** ranges from 1 to 8, with 8 being used for the Baran ice scheme. This parameterisation is described in Vidot *et al.* (2015). There are two parameterisations to choose from, **baran2014** (numerical value 1) and **baran2018** (numerical value 2, the default), selected by **opts%scatt%baran_ice_version**. The **baran20*** constants are defined in **rttov_const** (Annex K).

The 2018 parameterisation is more spectrally consistent and as such is recommended over the 2014 parameterisation. When using the DOM solver with the Baran scheme, the Legendre coefficients for the phase functions are calculated at run-time, and as such the Baran scheme is significantly slower than using the Baum ice properties with the DOM solver. The Baran scheme is defined for a given range of temperatures and ice water contents (see Table 8.17). If the maximum temperature is exceeded the **qflag_hydro_t_limits** quality flag is set and the maximum temperature is used. Temperatures below the minimum are not flagged as the optical properties asymptote at lower temperatures so extrapolation to lower temperatures is reasonable. If the maximum ice water content is exceeded, the **qflag_hydro_conc_limits** quality flag is set and the maximum ice water content limit is used. This is based on the concentration in the cloudy fraction of the layer, not the input grid box average value. See section 7.8.3 for more information on quality flags.

The Baum ice database was created for a range of ice water contents (see Table 8.17). For this particle type ice water contents are not modified or notified to the user when the limits are exceeded.

Index	Particle type	Description
opac_stco_index = 1	Stratus Continental	OPAC cloud liquid water types (Hess <i>et al.</i> , 1998). Each is based on a different size distribution, and the refractive index data are taken from Segelstein (1981).
opac_stma_index = 2	Stratus Maritime	
opac_cucc_index = 3	Cumulus Continental Clean	
opac_cucp_index = 4	Cumulus Continental Polluted	
opac_cuma_index = 5	Cumulus Maritime	
clw_deff_index = 6	Cloud liquid water parameterised by particle size	Size distribution is consistent with that assumed for the Mie cloud liquid water optical properties available with the libRadtran software (Emde <i>et al.</i> , 2016; http://www.libradtran.org). Refractive index data are from Segelstein (1981). Optical properties are interpolated based on cloud effective diameter (see below).
ice_baum_index = 7	Ice cloud parameterised by particle size	Ice optical properties from the Baum <i>et al.</i> (2011) database. These are interpolated based on cloud effective diameter (see below).
nhydro +1 = 8	Baran ice scheme	The Baran ice scheme parameterises optical properties in terms of temperature and ice water content. See below for more information on this scheme.

Table 8.15: hydrometeor types in the NWP SAF hydrotables for UV/VIS/IR. The *_index constants are available in **rttov_const** (Annex K). The final index is for the Baran ice scheme which is not stored in the hydrotable. In the general case, if the hydrotable contains **nhydro** particle types, the Baran scheme index is **nhydro+1**.

Two of the particle types, **clw_deff_index**=6 and **ice_baum_index**=7, are parameterised in terms of particle effective diameter (Deff). You can input values for particle size directly, and this is recommended if this information is available in your application. Particle sizes are specified in the **profiles(:)%hydro_deff(1:nhydro, 1:nlayers)** array. This allows effective diameters to be specified for any particle type by assigning values to the corresponding hydrometeor index. This allows the code to handle different hydrotables in the future which may have different/more types parameterised by Deff. Currently you should assign Deff values to hydro indices **clw_deff_index** and **ice_baum_index** if you are using these particle types: for any particle types not parameterised by Deff (such as the OPAC cloud liquid water types), any input values are ignored. Non-zero Deff values are also ignored where the corresponding **hydro(:,:)** concentration is zero. The particle types parameterised in terms of Deff have minimum and maximum limits (see Table 8.17), and where user input Deff values exceed these limits, they are clipped to the corresponding limit and this is notified to the user via the **qflag_hydro_deff_limits** quality flag (section 7.8.3).

RTTOV provides parameterisations for cloud liquid and ice Deff. These are activated by setting the corresponding **hydro_deff(:,:)** elements to zero for a particle type which depends on Deff. Note that any Deff value computed by a parameterisation that falls outside the minimum/maximum Deff limits for the given particle type (see Table 8.17) is clipped to the corresponding limit, but this is *not* notified via the output quality flags (section 7.8.3).

For cloud liquid water types (i.e., **clw_deff_index**), RTTOV implements the parameterisation based on Martin *et al* (1994). A minimum Deff of 10 µm is applied. This parameterisation was explicitly determined for warm stratocumulus clouds and as such may be limited in its applicability in convective regimes, for example. Users are advised to provide cloud particle size information consistent with their NWP model assumptions where possible, but this parameterisation provides a reasonable estimate in the absence of better information. There is a **profiles(:)%clwde_param** variable

(default `clwde_martin=1`, defined in `rttov_const`, see Annex K) for selecting among different parameterisations, but currently only one is implemented so `clwde_param` should not be modified.

For ice cloud types parameterised by particle size (i.e., `ice_baum_index`), RTTOV implements four Deff parameterisations in terms of temperature and ice water content, as described in Table 8.16. The selection is made in `profiles(:)%icede_param`, and it is recommended to use the constants given in Table 8.16 (defined in `rttov_const`, Annex K). Although some applicability limits are applied or defined for the ice Deff parameterisations, values outside these limits are not currently notified via the output quality flags (section 7.8.3).

UV/VIS/IR hydrotables are available for all sensors containing optical properties supporting all solvers implemented in RTTOV (section 8.4.2), excluding the radar solver (section 8.4.4). For IR hyperspectral sounders, “fast-only” hydrotables are available which exclude Legendre coefficients and phase functions. These files cannot be used with the DOM solver, but they are much smaller and are recommended if only the fast thermal solvers (Chou-scaling or delta-Eddington) are being used.

Parameterisation	Applicability
<code>icede_ou_liou = 1</code> Ou and Liou (1995)	Developed for cirrus clouds, depends on temperature, RTTOV restricts temperatures to the range -60 to -20°C when applying this parameterisation.
<code>icede_wyser = 2</code> Wyser (1998)	Developed to represent ice cloud in large-scale models, depends on temperature and ice water content, no details given on T or IWC limits. This is the recommended option based on comparisons to observations carried out with RTTOV.
<code>icede_boudala = 3</code> Boudala <i>et al.</i> (2002)	Based on observations of high latitude cirrus, depends on temperature and ice water content, the paper gives the following explicit limits though they are not applied within RTTOV: IWC in the range 0.001 to 0.45g/m ³ , temperature in the range -40 to 0°C.
<code>icede_mcfarquhar = 4</code> McFarquhar <i>et al.</i> (2003)	Based on observations of tropical cirrus anvils produced by deep convection, depends on ice water content, the plots in the paper suggest the observations went up to ~1g/m ³ but no limits are applied by RTTOV.

Table 8.16. Ice effective diameter parameterisations. The `icede_*` constants are defined in `rttov_const` (Annex K).



Particle type	Parameterisation limits					
	Min Deff:	Max Deff:	Min temperature:	Max temperature:	Min ice water content:	Max ice water content:
OPAC STCO	N/A					
OPAC STMA						
OPAC CUCC						
OPAC CUCP						
OPAC CUMA						
CLW Deff	2 µm	52 µm	N/A			
Baum ice	10 µm	120 µm	N/A		4.984E-5 g/m ³	0.1831 g/m ³
Baran ice	N/A		193.157 K	273.127 K	6.0E-06 g/m ³	1.969466 g/m ³

Table 8.17. Summarising parameterisation limits associated with each UV/VIS/IR hydrometeor type. Limits shaded grey are not flagged when exceeded. Ice water contents refer to the concentration in the cloudy fraction of each layer, not to the input grid box average values.

All sensors

In summary, to run hydrometeor scattering simulations with pre-defined optical properties, in addition to the usual steps for a clear-sky (non-scattering simulation), do the following:

- Set `opts%scatt%hydrometeors` to true
- Set `opts%scatt%user_hydro_opt_param` to false (this is the case by default)
- Ensure the hydrotable file is read in the call to `rttov_read_coefs` (see section 7.2 and Annex C).
- Specify the units of hydrometeor concentration in `profiles(:)%mmr_hydro` (true by default for units of kg/kg), and optionally in `profiles(:)%flux_conversion(:)` for MW sensors.
- Populate the input `profiles(:)%hydro(:,:)` array with grid box average hydrometeor concentrations in appropriate units for each particle type defined in the hydrotable file (the first index is the particle type, the second is layer number).

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

- Populate the input `profiles(:)%hydro_frac(:,:)` array with the layer cloud fractions from 0-1 (section 8.4.3).
- Optionally specify Deff and/or select Deff param (UV/VIS/IR only).

The example programs `src/test/example_hydro_visir_file_fwd.F90` and `src/test/example_hydro_mw_file_fwd.F90` demonstrate these steps for hydrometeor scattering simulations using pre-defined optical property files and can be used as templates for your own program (section 5.4.2).

8.4.8 Generating custom hydrotable files

As described in section 8.4.7, the default NWP SAF MW hydrotables contain properties for five particle types: rain, snow, graupel, cloud water and cloud ice. You may also create your own MW hydrotable files with optical properties that are more consistent with the assumptions behind your input hydrometeor profiles, or to add new hydrometeor types: hydrotables can contain optical properties for any number of hydrometeor types. New hydrotables can be created using the UNIX shell script `src/mw_scatt_coef/hydro_table_generation.ksh`. This script may need editing, for example to point to the location of your RTTOV binaries. The hydrotable generation is based around an input “`channels.dat`” file, examples of which are in the same directory. These define the parameters for the calculations and the instruments and channels for which to generate coefficients. The existing `channels.dat*` files provide useful templates. See the associated `readme.txt` in the `src/mw_scatt_coef/` directory for full details, or the scientific report Geer *et al.* (2021).

There is no tool available to generate custom hydrotable files for UV/VIS/IR sensors for use with RTTOV. This will be investigated for a future release. In the meantime, the explicit optical property inputs may be used instead (section 8.4.11).

8.4.9 Predefined aerosol optical properties

RTTOV aertable files (prefix `rttov_aertable_`) contain optical properties for an arbitrary collection of particle types. The aertable is read in at the same time as the gas optical depth coefficient file (section 7.2). You then supply vertical profiles of aerosol concentrations in the `profiles(:)%aerosols(1:naer, 1:nlayers)` array. The aertable contains optical properties for `naer` particle types, and the first index of the `aerosols(:,:)` array corresponds to the particle type in the order in which they are defined in the aertable.

The `profiles(:)%mmr_aer` logical flag selects the units: if true aerosol concentrations are supplied as mass mixing ratio (kg/kg), and if false, as particle number density (cm⁻³). All profiles passed into RTTOV in a single call must use the same units. Input aerosol concentrations are grid box average values which is typically what NWP model/GCM aerosol fields represent.

Currently, three types of aertable files are supplied for UV/VIS/IR sensors containing optical properties for different sets of particles. A tool is available enabling you to generate your own aertable files for spherical particles (section 8.4.10).

The three sets of aerosol properties currently produced by the NWP SAF are:

- **OPAC**, denoted by `_opac` in the filename. These contain thirteen types of aerosol, the first ten based on the OPAC (Hess *et al.*, 1998) components, two volcanic ash species, and an “Asian dust” species. Particle types 1-11 are described in detail in Matricardi (2005). The second volcanic ash particle type (number 12) uses a log-normal size distribution function calculated for radii between 0.005 and 20µm with parameters derived from aircraft measurements of the 2010 Icelandic eruption (Johnson *et al.*, 2012). The refractive indices are from Pollack *et al.* (1973). The optical properties for the Asian dust particle type (number 13) are calculated using a linear combination of size distributions for the MINM, MIAM and MICM aerosol particles for radii between 0.01 and 60µm: the weights were obtained by fitting to a particle size distribution derived from sky radiometer measurements made at Dunhuang, China (Han *et al.*, 2012). The refractive indices are from Volz (1972, 1973).
- **CAMS**, denoted by `_cams` in the filename. These contain nine types of aerosol based on a subset of the species represented in the ECMWF CAMS model. The CAMS species are described in Bozzo *et al.* (2017).
- **ICON-ART**, denoted by `_icon` in the filename. These contain seven types of aerosol based on a subset of the species represented in the ICON-ART model. The ICON-ART species are described in Muser *et al.* (2022).

Table 8.18 lists the species contained in each of these files with the index variables that can be found in **rttov_const** (Annex K).



For most aerosol types (OPAC, CAMS, and ICON-ART) the input is dry aerosol mass ratio. The only exceptions are the CAMS sea salt types for which the input is mass ratio at 80% relative humidity (consistent with outputs from the CAMS model), and the ICON-ART sea salt types for which the input is mass ratio at 70% relative humidity (consistent with outputs from the ICON-ART model). Therefore aerosol concentration fields from the CAMS or ICON-ART models can be input directly to RTTOV when using the corresponding aertables.

You can define any mix of the various aerosol components/species defined in the supplied aertable file. An example input aerosol profile using the OPAC types is given in the file: **rttov_test/test_example.1/aer_prof.dat**. RTTOV also provides a subroutine to calculate various climatological aerosol profiles with pre-defined mixtures of OPAC components (see the RTTOV v9 Science and Validation Report). The binary **create_aer_clim_prof.exe** (see Annex I) can be used to generate climatological aerosol profiles off-line. The list of climatological compositions output by this program is also shown in Annex I. The file **data/prof_aerosol_clim.dat** contains climatological aerosol profiles generated using the standard RTTOV 101 pressure levels (using the file **data/plevs.dat**) with T and q profiles taken from the file **data/prof.dat** for a latitude of zero, surface elevation of zero, and scale factor 1.0. You may wish to re-run the executable to generate your own sets of aerosol profiles for different input parameters, profiles or numbers of levels. Alternatively, the example program **src/test/example_aer_file_fwd.F90** gives an example of calling the **rttov_aer_clim_prof.F90** subroutine (Annex I) to obtain climatological profiles at run-time.

Index	OPAC file aerosol type	Abbreviation
aer_opac_index_inso = 1	Insoluble	INSO
aer_opac_index_waso = 2	Water soluble	WASO
aer_opac_index_soot = 3	Soot	SOOT
aer_opac_index_ssam = 4	Sea salt (acc mode)	SSAM
aer_opac_index_sscm = 5	Sea salt (coa mode)	SSCM
aer_opac_index_minm = 6	Mineral (nuc mode)	MINM
aer_opac_index_miam = 7	Mineral (acc mode)	MIAM
aer_opac_index_micm = 8	Mineral (coa mode)	MICM
aer_opac_index_mitr = 9	Mineral transported	MITR
aer_opac_index_suso = 10	Sulphated droplets	SUSO
aer_opac_index_vola = 11	Volcanic ash	VOLA
aer_opac_index_vapo = 12	New volcanic ash	VAPO
aer_opac_index_asdu = 13	Asian dust	ASDU
Index	CAMS file aerosol type	Abbreviation
aer_cams_index_bcar = 1	Black Carbon	BCAR
aer_cams_index_dus1 = 2	Dust, bin 1, 0.03-0.55 micron, ref. index: Woodward 2001	DUS1
aer_cams_index_dus2 = 3	Dust, bin 2, 0.55-0.90 micron, ref. index: Woodward 2001	DUS2
aer_cams_index_dus3 = 4	Dust, bin 3, 0.90-20.0 micron, ref. index: Woodward 2001	DUS3
aer_cams_index_sulp = 5	Ammonium sulphate	SULP
aer_cams_index_ssa1 = 6	Sea salt, bin 1, 0.03-0.5 micron	SSA1
aer_cams_index_ssa2 = 7	Sea salt, bin 2, 0.5-5.0 micron	SSA2
aer_cams_index_ssa3 = 8	Sea salt, bin 3, 5.0-20.0 micron	SSA3
aer_cams_index_omat = 9	Hydrophilic organic matter	OMAT
Index	ICON-ART file aerosol type	Abbreviation
aer_icon_index_soot = 1	Soot/black carbon	SOOT
aer_icon_index_dusa = 2	Dust mode A	DUSA
aer_icon_index_dusb = 3	Dust mode B	DUSB
aer_icon_index_dusc = 4	Dust mode C	DUSC
aer_icon_index_ssaa = 1	Sea salt mode A	SSAA
aer_icon_index_ssab = 2	Sea salt mode B	SSAB
aer_icon_index_ssac = 3	Sea salt mode C	SSAC

Table 8.18: aerosol types in the NWP SAF aertables. The aer*index* constants are available in **rttov_const** (Annex K).

In summary, to run aerosol scattering simulations with pre-defined optical properties, in addition to the usual steps for a clear-sky (non-scattering simulation), do the following:

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

- Set **opts%scatt%aerosols** to true
- Set **opts%scatt%user_aer_opt_param** to false (this is the case by default)
- Ensure the aertable file is read in the call to **rttov_read_coefs** (see section 7.2 and Annex C).
- Specify the units of aerosol concentration in **profiles(:)%mmr_aer** (true by default for units of kg/kg).
- Populate the input **profiles(:)%aerosol(:,:)** array with mean layer aerosol concentration in appropriate units for one or more aerosol types defined in the aertable file (the first index is the particle type, the second is layer number).

The example program **src/test/example_aer_file_fwd.F90** demonstrates these steps for aerosol scattering simulations using pre-defined optical property files and can be used as a template for your own programs (section 5.4.2).

8.4.10 Generating custom aerosol optical property files

RTTOV provides a tool **rttov_make_aertable.exe** which allows you to generate custom *rttov_aertable* files for use with RTTOV. This tool is currently limited to spherical particles. Custom aertable files can contain an arbitrary number of aerosol species. Just as for the pre-defined aertable files described above, when the **profiles** structure is allocated, the first dimension of the **profiles(:)%aerosols(:,:)** array is the number of aerosol species in your *rttov_aertable* file and you specify the species concentrations in the order in which they are defined in the file. There is a file in the **docs/** directory, **readme_rttov_make_aertable.txt**, which describes in detail how to use this tool.



8.4.11 Explicit optical properties

This is an alternative and slightly more complicated way to run scattering simulations for aerosols and/or hydrometeors instead of using the pre-defined optical property files. This provides greater flexibility than using the pre-defined particle types. In this case you provide vertical profiles of the optical properties required by the solver(s) being used for each sensor channel being simulated. The process is extremely similar for hydrometeors and aerosols.

Explicit optical property inputs are activated by setting the **opts%scatt%user_hydro_opt_param** and/or **opts%scatt%user_aer_opt_param** options to true (for hydrometeors and/or aerosols respectively). In this case no hydrotable or aertable files are required, and the **profiles(:)%hydro(:,:)** (for hydrometeors) or **profiles(:)%aerosols(:,:)** (for aerosols) arrays are not used.

Input optical properties are provided to RTTOV via the **hydro_opt_param** and/or **aer_opt_param** arguments (of derived type **rttov_opt_param**). This structure has member arrays for each optical property required by the RTTOV solvers. All members are allocated when calling the allocation subroutine (see section 7.3 and Annex D), but you can specify some dimensions to be zero (**nmom**, **nphangle** – see below) if the corresponding members are not required by the solver(s) you are using. The members are as follows:

- **ext(:,:)** : extinction coefficient (units: km⁻¹), dimensions (**nlayers**, **nchanprof**). Always required.
- **ssa(:,:)** : single scattering albedo [0-1] (no units), dimensions (**nlayers**, **nchanprof**) . Always required.
- **bpr(:,:)** : “b parameter” [0-1] (no units), dimensions (**nlayers**, **nchanprof**). Only required by the Chou-scaling thermal solver for channels with thermal emission (wavelength > 3μm). This is the fraction of back-scattered radiation from each layer and can be calculated from the phase function using a supplied subroutine (see below).
- **asym(:,:)** : asymmetry parameter [-1,1] (no units), dimensions (**nlayers**, **nchanprof**). Only required by the delta-Eddington thermal solver for channels with thermal emission (wavelength > 3μm). This represents the relative amount of forward or backward scattering and can be calculated from the phase function using a supplied subroutine (see below).
- **nmom** : number of Legendre coefficients specified for all phase functions. Only required by the DOM solver (for thermal emission or solar radiation).
- **lcoef(:,:,:)** : coefficients of Legendre decomposition of phase function (no units), dimensions (**1:nmom+1**, **nlayers**, **nchanprof**). Only required by the DOM solver (for thermal emission or solar radiation). These can be computed using a supplied subroutine (see below). Note that the “zeroth” moment in **lcoef(1,,:)** must always be one and **nmom** coefficients are supplied in addition to this where **nmom** is greater than or equal to the number of DOM streams specified in the simulation (**opts%scatt%dom_nstreams**).

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

- **phangle(:)** : the angles over which phase functions are defined (units: degrees), dimension (**nphangle**). This should cover the full range of scattering angles monotonically from 0° to 180° inclusive. The angle grid does not have to be evenly spaced. Only required by the DOM solar solver.
- **pha(:, :, :)** : azimuthally-averaged phase function, dimensions (**nphangle**, **nlayers**, **nchanprof**). Only required by the DOM solar solver but may optionally be used for the calculation of the *b* parameter, asymmetry parameter, and Legendre coefficients (as for **phangle(:)**). Phase functions should be normalised such that the integral over all scattering angles is 4π . Given phase function *P* defined over scattering angles Θ from 0 to π , and zenith angle θ and azimuthal angle ϕ we require

$$4\pi = \int_0^\pi P(\theta) d\theta = \int_0^\pi P(\theta) \sin(\theta) d\theta \int_0^{2\pi} d\phi$$

so that:

$$1 = \frac{1}{2} \int_0^\pi P(\theta) \sin(\theta) d\theta$$

The phase functions for the UV/VIS/IR pre-defined particle types are defined over one of two angular arrays both of which can be found in the **rttov_const** module (Annex K) as **phangle_lores(1:208)** and **phangle_hires(1:498)**. Alternatively, you can define your own angle grid.

The dimension **nlayers** corresponds to the number of full-levels in the input profile structure (section 7.4.1), and the dimension **nchanprof** represents the total number of channels being simulated (section 7.5).

For hydrometeor simulations, the hydro fraction is required in **profiles(:)%hydro_frac(1,:)** and input optical properties apply to the cloudy fraction of each layer rather than representing grid box average values. There is a non-linear relationship between hydrometeor amount and the optical properties which means they cannot be provided as grid box average values (unless the cloud fraction is one). This means the optical property inputs cannot be used with any overlap scheme that computes cloud fractions within RTTOV. Currently the hydrometeor property inputs are only supported with the max/random overlap option.

A subroutine is provided to calculate the *b* parameters from the phase functions. This is relatively slow and so is not performed internally within RTTOV. It can make use of multiple threads if you compile RTTOV with OpenMP support (section 5.3). You may find it beneficial to calculate the *b* parameters off-line and store them for future use if this is practical for your application. To generate the values:

- call **rttov_bpr_init** to initialise some tables to speed up the calculation
- call **rttov_bpr_calc** to calculate the *b* parameters from the phase functions: this is called once for every phase function (i.e., for each layer containing scattering particles and for each channel)
- call **rttov_bpr_dealloc** to release allocated memory

The **rttov_asym_calc** subroutine is provided to calculate the asymmetry parameter for a phase function. This is quite fast.



In addition, the **rttov_lcoef_calc** subroutine is provided to calculate the Legendre coefficients for a phase function. This is quite fast, and you only need to calculate as many coefficients as DOM streams you will specify for your simulations.

The interfaces for all these subroutines are described in Annex E.

Note that if you already have values for **bpr**, **asym** and/or the Legendre coefficients and you are not running solar simulations (i.e., **opts%rt_all%solar** is false) then the **phangle** and **pha** arrays (i.e., phase angles and phase functions) do *not* need to be populated for the call to RTTOV.

If solar scattering calculations are being performed, you must populate the **phangle(:)** member array and then call the **rttov_init_opt_param_solar** subroutine (see Annex D) to pre-calculate some values related to the phase angles. This is not required if **opts%rt_all%solar** is false.

When running the TL/AD/K models with explicit optical properties, the Legendre coefficients and the phase function are not active TL/AD/K variables. All other properties are active variables. See section 7.9.

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

The following limitations apply to simulations with explicit optical properties:

- The MFASIS-NN solar solver cannot be used (section 8.4.2).
- The radar solver cannot be used (section 8.4.4).
- The maximum/random overlap cloud overlap option *must* be used for hydrometeor simulations (section 8.4.3).
- The MW empirical and ARO scaling polarisation treatments are not applied (section 8.4.6).
- PC-RTTOV cannot be used (section 8.6).

In summary, to run hydrometeor / aerosol simulations with explicit optical properties, in addition to the usual steps for a clear-sky (non-scattering simulation), do the following:

- Set **opts%scatt%hydrometeors** / **opts%scatt%aerosols** to true
- Set **opts%scatt%user_hydro_opt_param** / **opts%scatt%user_aer_opt_param** to true
- Declare a variable of type **rttov_opt_param**, for example **hydro_opt_param** / **aer_opt_param** and allocate the member arrays by calling **rttov_alloc_opt_param** or **rttov_alloc_direct/tl/ad/k** (section 7.3).
- Populate **hydro_opt_param** / **aer_opt_param** with extinction coefficients and single scattering albedos, and, for solar radiation, the phase function and phase angles.
- Populate **hydro_opt_param** / **aer_opt_param** with the *b* parameters (if using the Chou-scaling thermal solver) either from pre-calculated data or by calling **rttov_bpr_calc**
- Populate **hydro_opt_param** / **aer_opt_param** with the asymmetry parameters (if using the delta-Eddington thermal solver) either from pre-calculated data or by calling **rttov_asym_calc**
- Populate **hydro_opt_param** / **aer_opt_param** with the Legendre coefficients (if using the DOM solver) either from pre-calculated data or by calling **rttov_lcoef_calc**
- If performing solar calculations call **rttov_init_opt_param_solar**
- Hydrometeor simulations only: populate the input **profiles(:)%hydro_frac(1,:)** array with cloud fractions (section 8.4.3).
- Pass the **hydro_opt_param** / **aer_opt_param** argument into **rttov_direct**
- When finished with RTTOV call **rttov_alloc_opt_param** or **rttov_alloc_direct/tl/ad/k** again to deallocate **hydro_opt_param** / **aer_opt_param**.

Example programs demonstrating these steps for hydrometeor (**src/test/example_hydro_param_fwd.F90**) and aerosol (**src/test/example_aer_param_fwd.F90**) simulations can be used as a template for your own programs (section 5.4.2).

8.4.12 Support for dynamic emissivity retrievals

RTTOV provides support for dynamic emissivity retrievals following the method of Baordo and Geer (2016). The approach is generalised to clear-sky cases and to all cloud overlap options (section 8.4.3). For scattering simulations, these outputs are available with the Chou-scaling and delta-Eddington thermal solvers (section 8.4.2).

An optional output argument **emis_retrieval_terms** (derived type **rttov_emis_retrieval_terms**) of the RTTOV direct model contains the relevant parameters for the emissivity retrieval (see Annex J). As with other interface structures there are helper routines to (de)allocate and initialise it (see section D). If the argument is present in the call to **rttov_direct** and the conditions described below are met, it will be populated on output. The subroutine **rttov_emissivity_retrieval** can then be used to retrieve the surface emissivities, and optionally also skin temperatures (see Annex I).

The emissivity retrieval structure is populated under the following conditions:

- it is compatible only with homogenous surfaces (i.e., **nsurfaces=1**, see section 8.3.10)
- if **opts%scatt%hydrometeors** is true then either delta-Eddington or Chou-scaling must be selected as the **opts%scatt%thermal_solver** (section 8.4.2), otherwise the structure will not be populated.
- only elements corresponding to thermal channels are populated.
- elements corresponding to Stokes 3/4 channels (on polarimetric sensors) are not populated.

8.5 Inclusion of non-local thermodynamic equilibrium effects

RTTOV can estimate NLTE effects above altitudes of ~40 km in the CO₂ v3 band (around 4.3 μm). Here, local thermodynamic equilibrium breaks down due to the absorption of the strong solar radiation field. At the time of release NLTE coefficients are available for IASI, IASI-NG, and MTG-IRS (based on LBLRTM v12.8), and CrIS and AIRS (based on LBLRTM v12.2). NLTE coefficients for other hyperspectral sounders can be requested via the NWP SAF helpdesk.

To invoke the NLTE correction it is necessary to:

- Use an optical depth (*rtcoef*) coefficient file that contains the NLTE coefficients for the regression.
- Set **opts%rt_all%nlte_correction = .true.**

The NLTE correction is valid (and is applied to channels) between 2200 cm⁻¹ – 2400 cm⁻¹ which corresponds to:

IASI channels 6221 – 7020 (800 channels)	CrIS NSR channels 1165 – 1245 (81 channels)
IASI-NG channels 12441 – 14039 (1599 channels)	CrIS FSR channels 1651 – 1971 (321 channels)
MTG-IRS channels 1870 – 1953 (84 channels)	AIRS channels 1886 – 2121 (236 channels)

The radiance correction scheme is documented in Matricardi *et al.* (2018) and consists of eight predictors. These predictors consist of various combinations of the solar zenith angle, θ^{sol} , the satellite zenith angle, θ^{sat} , and the average kinetic temperature in two broad atmospheric layers above ~51 hPa (i.e., the average temperature between 0.005 hPa and ~0.2 hPa, *T1* and the average temperature between ~0.3 hPa and ~51 hPa, *T2*). The predictors p_j are shown in Table 8.19. The radiance correction, $\Delta R_{ch}^{\text{NLTE}}$, is added to the LTE TOA radiance to give the NLTE TOA radiance, thus,

$$R_{ch}^{\text{NLTE}} = R_{ch}^{\text{LTE}} + \Delta R_{ch}^{\text{NLTE}}$$

where $\Delta R_{ch}^{\text{NLTE}}$ is written as:

$$\Delta R_{ch}^{\text{NLTE}} = \sum_{j=1}^9 c_{ch,j} p_j$$

Here $c_{ch,j}$ are the regression coefficients.



Predictor number	Predictor
1	constant
2	$\cos(\theta^{\text{sol}})$
3	$(\cos(\theta^{\text{sol}}))^{0.5}$
4	$\cos(\theta^{\text{sol}}) \sec(\theta^{\text{sat}})$
5	$\cos(\theta^{\text{sol}}) T1$
6	$(\cos(\theta^{\text{sol}}) \sec(\theta^{\text{sat}}))^2$
7	$\cos(\theta^{\text{sol}}) T2$
8	$\sec(\theta^{\text{sat}}) T1$
9	$\sec(\theta^{\text{sat}}) T2$

Table 8.19: the predictors used in the NLTE algorithm.

You are responsible for ensuring that input values for the solar zenith angle are in range (i.e., >0° and < 90°). For any particular profile, RTTOV will not calculate the correction if the supplied solar zenith angle lies out of range and will extrapolate the correction if the supplied satellite zenith angle is larger than 60° (as long as the solar zenith angle is valid).

The NLTE correction is distinct from the solar simulation capability and as such it is *not* necessary to set **opts%rt_all%solar** to true for the NLTE correction to be applied. The NLTE correction may be used in conjunction with scattering calculations (excluding PC-RTTOV, section 8.6), but be aware that the NLTE corrections are added to the final LTE hydrometeor/aerosol-affected radiances so the NLTE radiation does not interact with the scattering particles.

A more detailed discussion of the science and the impact of including the NLTE correction is available in the RTTOV v12 Science and Validation Report.

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

8.6 PC-RTTOV for hyperspectral IR sounders

A principal component (PC) based version of RTTOV is available for the simulation of the PC scores of the spectra of hyperspectral IR sounders (Matricardi, 2024) based on a truncated set of synthetic eigenvectors. If needed, the PC scores can be used for the efficient and accurate reconstruction of the full radiance spectrum. The PC-based model uses polychromatic RTTOV radiances to predict the principal component scores using a linear regression scheme. To invoke the PC calculations in RTTOV a logical flag `opts%pcrttov%enable_pcrttov` is set to true. An example of calling PC-RTTOV is given in `src/test/example_pc_fwd.F90`: this can be used as a model for your own code (section 5.4.2).

Coefficients

The PC-RTTOV coefficient filenames begin “*pccoef_*” and can be downloaded from the RTTOV web site. These must be used alongside the RTTOV optical depth (*rtcoef*) coefficient files with which they were trained. PC-RTTOV is trained using v13 predictor 101 level “7gas” coefficients. If an incompatible optical depth coefficient file is used an error will result. The PC coefficient file is read in at the same time as the optical depth coefficient file in the call to `rttov_read_coefs` (section 7.2 and Annex C).

PC-RTTOV simulations allow for the variation of H₂O, O₃, CO₂, N₂O, CO and CH₄. PC-RTTOV has different regression limits for gases than the RTTOV optical depth coefficients, but these are treated in the way as for standard RTTOV (see section 7.4.3) with the exception that when the `opts%config%apply_reg_limits` option is true, the input profile is not modified even when the limits are exceeded (but see below for hydrometeor and aerosol simulations). The PC regression limits can be found on the RTTOV coefficients download page:

https://nwp-saf.eumetsat.int/site/software/rttov/download/coefficients/coefficient-download/#Reference_profiles_and_regression_limits

The PC-RTTOV coefficients available for RTTOV v14 have been trained for clear-sky simulations, optionally including an NLTE correction (section 8.5), and for aerosol-affected simulations based on the OPAC optical properties, and for hydrometeor-affected simulations (see section 8 for information on scattering simulations).

The NLTE correction is described in section 8.5. It is not currently possible to apply the NLTE correction to scattering simulations (aerosols or hydrometeors) with PC-RTTOV.

For aerosol simulations, the corresponding OPAC aertable file should be read in at the same time as the RTTOV optical depth and PC-RTTOV coefficient files by the call to `rttov_read_coefs`. It is recommended to use the smaller “fast-only” aertable files. The aerosol simulations are carried out as described in section 8.4 but you must use the Chou-scaling parameterisation (section 8.4.2). The PC coefficients are trained using the radiative effects of each of the OPAC aerosol components separately and the radiative effects of the climatological combinations of the OPAC aerosol components: this includes only indices 1-10 from the RTTOV OPAC aertable files (see Table 8.18). RTTOV ignores any input aerosol concentrations for the volcanic ash and Asian dust species. The minimum/maximum aerosol concentrations (regression limits) used in training the PC profiles can be found on the RTTOV coefficients download page. Note that in some layers for some species there is no variability in the aerosol concentrations used in the PC training (this includes the sulphates component in all layers). RTTOV automatically uses the regression limit values for those layers, which may be zero where there were no training profiles with non-zero concentrations.

For layers where the aerosols varied in the training RTTOV clips the input aerosol profiles to the limits if the `opts%config%apply_reg_limits` option is true (see section 7.4.3). If the aerosol regression limits were exceeded in one of these layers the `radiance%quality(:)` flags have the `qflag_pc_aer_reg_limits` bit set for the PC predictor channels corresponding to the relevant profile (see section 7.8.3). The clipping and flagging is not applied where input aerosol concentrations are zero.

For hydrometeor simulations, the corresponding hydrotable file should be read in at the same time as the RTTOV optical depth and PC-RTTOV coefficient files by the call to `rttov_read_coefs`. It is recommended to use the smaller “fast-only” hydrotable files. The hydrometeor simulations are carried out as described in section 8.4 but you must use the Chou-scaling parameterisation (section 8.4.2), and the maximum/random cloud overlap option (section 8.4.3). The PC coefficients are compatible with all particle types contained in the NWP SAF hydrotables (section 8.4.7) and with the Baran 2018 ice scheme. As such you can assign hydrometeor profiles to any cloud liquid and/or cloud ice particle types. However, the training only includes one liquid cloud type and/or one ice cloud type per layer, so it is not recommended to assign non-zero concentrations to multiple cloud liquid types and/or multiple ice cloud types in any single layer (but

one non-zero liquid cloud and one non-zero ice cloud per layer is fine). RTTOV does not prevent this, but it is flagged in the **qflag_pc_cld_reg_limits** bit of the **radiance%quality(:)** flags. There are some layers in which the training profiles contained no cloud liquid or no cloud ice water. RTTOV automatically sets the corresponding hydrometeor concentrations to zero in those layers. For the hydrometeor types with dependence on effective particle diameter (Deff), the Martin *et al* (cloud liquid) and Wyser (cloud ice) Deff parameterisations were used in the training, but this is not mandated in the code and any available Deff parameterisation may be used or explicit input of Deff values.

For layers where the hydrometeors varied in the training RTTOV clips the input hydrometeor profiles to the limits if the **opts%config%apply_reg_limits** option is true (see section 7.4.3). If the hydrometeor regression limits were exceeded in one of these layers the **radiance%quality(:)** flags have the **qflag_pc_hydro_reg_limits** bit set for the PC predictor channels corresponding to the relevant profile (see section 7.8.3). The clipping and flagging are not applied where input hydrometeor concentrations or the corresponding hydro fractions are zero.

Note that it is not currently possible to enable both hydrometeors and aerosols in PC-RTTOV simulations.

PC coefficient files are trained over all surface types. It is strongly recommended to set **calc_emis(:)** to true for sea profiles (see section 8.3): PC-RTTOV is trained using the IREMIS sea surface emissivity model (section 8.3.1) and this is used automatically for sea profiles when **calc_emis(:)** is true. It is also recommended to use the CAMEL v3 climatology land surface emissivity atlas with the angular correction enabled (see section 8.3.6) for land surfaces as this was used to train the PC-RTTOV coefficients. However, as the PC-RTTOV training encompasses a wide range of surface emissivities, the use of alternative physically realistic sources for surface emissivity should be acceptable. For this reason, RTTOV carries out no checks on how surface emissivity is specified for PC-RTTOV.

The PC coefficient training is carried out with specific option settings and PC-RTTOV simulations should be configured to be consistent with the training. These options are given in Table 8.20. Most of these options are automatically applied within RTTOV, while others generate errors in RTTOV if incorrectly specified (indicated as “Mandatory” in the table). Note for example that the 2m temperature and water vapour inputs are *not* used in the training and are therefore ignored in PC-RTTOV simulations.

RTTOV option	PC-RTTOV setting	Comments
opts % config % transmittances_only	False	Mandatory
opts % config % bt_overcast_calc	False	Applied automatically
opts % config % gas_opdep_calc	True	Applied automatically
opts % config % opdep13_gas_clip	True	Applied automatically
opts % interpolation % interp_mode	interp_rochon_wfn	Applied automatically
opts % rt_all % solar	False	Applied automatically
opts % rt_all % refraction	True	Applied automatically
opts % rt_all % plane_parallel	False	Applied automatically
opts % rt_all % rad_down_lin_tau	True	Applied automatically
opts % rt_all % use_t2m	False	Applied automatically
opts % rt_all % use_q2m	False	Applied automatically
opts % surface % ir_sea_emis_model	ir_emis_model_irem	Applied automatically
opts % surface % lambertian	False	Applied automatically
opts % surface % use_tskin_eff	False	Mandatory
opts % scatt % thermal_solver	thermal_solver_chou	Mandatory (aerosols/hydrometeors)
opts % scatt % chou_tang_mod	False	Applied automatically (aerosols/hydrometeors)
opts % scatt % user_aer_opt_param	False	Mandatory (aerosols)
opts % scatt % user_hydro_opt_param	False	Mandatory (hydrometeors)
opts % scatt % baran_ice_version	baran2018	Applied automatically (hydrometeors)
opts % cloud_overlap % overlap_param	cloud_overlap_max_random	Mandatory (hydrometeors)

Table 8.20: PC-RTTOV option settings. Mandatory implies an error will be thrown if the option is set incorrectly. See Annex K for integer option constants.

Predictor channel sets and number of PC scores

In order to call PC-RTTOV a specific set of channels must be specified in the **chanprof(:)%chan** array for each profile being simulated. The simulated RTTOV radiances for this set of channels comprise the predictors in the PC-RTTOV regression. The size of the predictor channel set determines the number of channels being simulated per profile. The predictor set is selected in **opts%pcrttov%pc_reg_set**. Table 8.21 gives the valid values for the PC-RTTOV coefficients available at the time of the RTTOV v14 release. By choosing a larger predictor set one trades reduced computational efficiency for increased accuracy.

The recommended way of obtaining the channel list is via the subroutine **rttov_get_pc_predictindex** (Annex I): an example of this can be seen in **src/test/example_pc_fwd.F90**. RTTOV will report an error if the input channel list (in **chanprof(:)%chan**) does not match the predictor channel list. It is also possible to obtain the specific channel list for a given predictor set directly from the PC coefficient structure in **coefs%coef_pcomp%pcreg(i,j)%predictindex(:)** for band **i** (see below - usually 1) and predictor set **j**.

NB The predictor channel sets are unique to each individual PC-RTTOV coefficient file for a given instrument.

The variable **opts%pcrttov%pc_band** provides a choice of carrying out calculations for limited spectral bands, but currently the PC coefficient files available on the website contain information for the whole spectrum only (band 1) and hence this variable must always be set to 1.

PC-RTTOV can simulate a maximum number of 400 PCs. This is specified in the **opts%pcrttov%npcscores** option. The choice of the number of PCs to use depends on specific aspects of the application. For radiance simulations, a typical choice is 500 predictors for the PC score simulation and 200 PCs for the radiance reconstruction for IASI/IASI-NG and 300 predictors and 100 PCs for MTG-IRS. The number of simulated PC scores used for the radiance reconstruction can be increased in order to improve accuracy at cost in computational efficiency. In practice, to achieve good accuracy it is strongly recommended to use no fewer than 200 PC scores for IASI and IASI-NG, and no fewer than 100 PC scores for MTG-IRS.

Coefficient file	opts%pcrttov%pc_band	opts%pcrttov%pc_reg_set	Max npcscorers
IASI	1 => full spectrum (all channels)	1, 2, 3 or 4 => 300, 400, 500 or 600 predictors respectively	400
IASI-NG	1 => full spectrum (all channels)	1, 2, 3 or 4 => 300, 400, 500 or 600 predictors respectively	400
MTG-IRS	1 => full spectrum (all channels)	1, 2, 3 or 4 => 250, 300, 350 or 400 predictors respectively	400


Table 8.21. Available options for band and predictor set for PC coefficient files available at the time of the RTTOV v14 release.

PC-RTTOV outputs

The computed PC scores are stored in the **rttov_pcomp** structure (see Annex J) in the **total_pcscores(:)** array. It is possible to reconstruct radiances from the PC scores by setting **opts%pcrttov%rec_rad** to true. In this case the total number of reconstructed radiances for all profiles (or the maximum number of profiles being passed per call to RTTOV) must be passed in the call to **rttov_alloc_pcomp** (Annex D) and the reconstructed channel list **channels_rec(:)** must be supplied to **rttov_direct** (or to the TL, AD or K model; Annex H). The reconstructed radiances are also stored in the **pcomp** structure in the **total_pcomp(:)** array, and the corresponding brightness temperatures are in **bt_pcomp(:)**. The **channels_rec(:)** input argument to **rttov_direct** is mandatory if **opts%pcrttov%rec_rad** is true.

For profiles with variables that fall outside of the range of the PC coefficient training set, it is possible for PC-RTTOV to reconstruct negative radiances. This is very rare and has only been observed over extremely cold surfaces where the skin temperature is very low (e.g., below approximately 200 K, bearing in mind the training set lower bound is ~207 K). Where negative values occur, PC-RTTOV sets the corresponding reconstructed radiance and BT values to zero. If you are simulating very cold profiles, you should consider checking for zero radiance/BT values in the output.

It is technically possible to run PC-RTTOV using coefficient files containing a subset of channels (created using **rttov_conv_coef.exe**, see Annex A) or to read in a subset of channels in the call to **rttov_read_coefs** (see Annex C). However, it is important to note that the subset of extracted channels must include the set of predictor channels being

<p>The EUMETSAT Network of Satellite Application Facilities</p>	 <p>NWP SAF Numerical Weather Prediction</p>	<p>RTTOV v14 Users' Guide</p>	<p>Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025</p>
---	--	-------------------------------	--

used and any channels for which reconstructed radiances are required. In addition, the channels specified in **chanprof(:)%chan** and **channels_rec(:)** are always indexed starting from 1 into the set of channels read into RTTOV. This means you must keep track of the remapped channel numbers. For this reason, it is not recommended to run PC-RTTOV with channel subsets unless you are very confident in what you are doing.

8.7 Special gas optical depth coefficient files

8.7.1 Simulation of Zeeman effect (SSMIS, AMSU-A, ATMS)



For microwave sensors that have high peaking weighting functions in the mesosphere such as SSMIS, channels close to lines of molecular oxygen may be significantly affected by the redistribution of line intensity through Zeeman splitting as described in the RTTOV v10 Science and Validation Report. The absorption for the affected channels will depend on the strength and orientation of the magnetic field. You must specify two input variables for the geomagnetic field in the **rttov_profile** structure, these being the magnitude, **Be**, of the field and the cosine, **cosbk**, of the angle between the field vector and the viewing path considered. For SSMIS, values are available with the satellite data stream, and will therefore already match the geographical location and orientation of the viewing path. For AMSU-A and ATMS, this is not the case, but the values may be obtained from a pre-computed look-up table. For instance, the **rttov_zutility** module provided in the **src/other/** directory may be used to provide values (see Annex I). For a normal run where the Zeeman effect is not computed the variables can be set to any value, including zero, but if the Zeeman effect is to be calculated, **Be** must lie in the range 0.2-0.7 Gauss as this covers the range of values over which the Zeeman coefficients were trained. In particular, **Be** must not be set to zero when calling RTTOV with a Zeeman coefficient file: if the Zeeman effect is not important for an application a non-Zeeman coefficient file should be used instead.

A 'Zeeman' coefficient file will have the Zeeman flag set to one in the 'Fast Model Variables' section and "zeeman" in the filename. To include the Zeeman effect for a given sensor, you must run RTTOV with a Zeeman coefficient file.

As with all RTTOV simulations your input profile pressure levels should reach sufficiently high that they span the weighting functions of the channels being simulated (i.e., sufficiently high that there is insignificant absorption above the top-most input pressure half-level). If this condition is not met there can be significant errors resulting from the treatment of emission from the region of atmosphere above the top input half-level. After interpolating the calculated optical depth profile onto the input pressure levels, RTTOV sets the optical depth of the top-most half-level to zero. This behaviour is intended to mitigate the case where there is significant absorption above the top half-level: in this case the top layer is effectively stretched to reach the space boundary and the effects of emission and absorption from the region of atmosphere above the top half-level are included in the integration of the radiative transfer equation. However, without an accurate representation of the temperature of the atmosphere above the top level the emission term will be in error to some degree. Note that this has negligible impact if the top input profile half-level is sufficiently high because in that case the interpolated optical depth at the top half-level will be close to zero anyway. This applies to all simulations, not only Zeeman-affected simulations, but is of most relevance for very high-peaking channels.

For SSMIS channels 19-22 which are affected by Zeeman splitting, the brightness temperature in channel 20 may be altered by as much as 10 K – the change in column absorption will shift the channel weighting function, but the effect of this will depend on the temperature profile. When you run RTTOV with a non-Zeeman coefficient file, the mixed gas prediction scheme will be based on the usual non-Zeeman predictors. However, when a Zeeman coefficient file is used, the mixed gas scheme will incorporate additional predictors used for the high peaking channels. In the optical depth calculation for channels 1-18 and 23-24 (non-Zeeman), contributions from the additional predictors will be nullified by zero coefficients. In contrast, for channels 19-22 (Zeeman), it is only the new predictors that contribute.

For AMSU-A and ATMS, only channels 14 (AMSU-A) or 15 (ATMS) are affected. These channels, while dominated by oxygen absorption, sounds lower down in the atmosphere than the Zeeman channels of SSMIS, and they are also located further from the oxygen line centres. The impact is therefore much smaller (~0.5K). If you run with a non-Zeeman coefficient file, all channels will use the usual set of mixed gas predictors and the Zeeman effect will not be represented in the relevant channel. If a Zeeman coefficient file is used, then a small set of additional predictors will be included. These will contribute for the Zeeman channel but will be nullified for the other channels by zero coefficients.

		RTTOV v14 Users' Guide	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	-------------------------------	---

8.7.2 *Simulation of SSU radiances*

For SSU, which uses pressure modulated gas cells to define the channels, RTTOV allows you to take some account of inadvertent cell pressure changes that may have occurred over the lifetime of the instrument.

SSU coefficients were updated for RTTOV v10 using more recent molecular spectroscopy (LBLRTMv12.0), more variable gases (now O₃ as well as H₂O, CO₂) and more stratospheric levels (now 51). RTTOV v11 provided an additional set of coefficients with the label '*pmcshift*' in the file name which are based on the same nominal set of cell pressures, and differ only in having a new 'PRESSURE_MODULATED_CELL' section.

When simulating SSU with these pressure modulated cell (PMC) shift coefficients, you must now provide a preferred set of gas cell pressures (in hPa), one for each channel. These may be different from those in the nominal set and should be assigned to the channel array `coefs%coef%pmc_coef%ppmc(1:3)`. This should be done after the coefficient file has been read using the `rttov_read_coefs` subroutine. It is also mandatory to supply an input CO₂ profile when using the PMC shift coefficients as this is required by the cell pressure scheme.

The SSU coefficients are currently based on v8 predictors, and these will continue to be supported until v13 predictor SSU coefficients are available.

8.7.3 *Simulation of Nimbus PMR radiances*

The PMR instrument carried on Nimbus-6 is another special case. The corresponding gas optical depth coefficient file contains coefficients for 9 pseudo channels. These represent PMR observations at one wavenumber at different zenith angles between +15° and -15°. PMR simulations must be run with the profile zenith angle set to 0.

The PMR coefficients are currently based on v8 predictors, and these will continue to be supported until v13 predictor PMR coefficients are available.

8.7.4 *Simulation of MTG-LI*

RTTOV cannot directly simulate MTG-LI observations. The MTG-LI coefficients are primarily intended for simulation of transmittances for atmospheric correction. The instrument has four detectors which together observe most of the Earth disc. The optical depth coefficient file contains two channels representing SRFs corresponding to the centre of each field of view (incidence angle of 0° on the detector) and the edge of the field of view (incidence angle of 5.1° on the detector). It would generally be recommended to use channel 1 for observations nearer the centre of each field of view, and channel 2 for observations nearer the edge of each field of view.

9 Limitations of RTTOV v14

There are several scientific limitations of RTTOV v14 you should be aware of. The main ones are listed here:

- RTTOV only simulates top of atmosphere radiances from a nadir or off-nadir view which intersects with the Earth's surface (i.e., no limb paths or upward viewing paths).
- RTTOV only allows for water vapour, ozone, carbon dioxide, nitrous oxide, methane, carbon monoxide and sulphur dioxide to be variable gases with all others included in the mixed gases transmittance calculation.
- RTTOV can only simulate radiances for instruments for which a gas absorption optical depth coefficient file has been generated. The instruments currently supported are listed in Table 2.2.
- The accuracy of simulations for very broad channels (e.g., SEVIRI channel 4 at 3.9 μm) is poor with significant biases noted ($\sim 1\text{-}2\text{K}$) (see e.g., Brunel and Turner, 2003). This is the case for all versions of RTTOV. A work around is to use Planck weighted coefficient files (which are now standard for all sensors where this is a problem) resulting in much lower biases. Whether coefficients are Planck-weighted can be determined by examining the PLANCK_WEIGHTED section in the coefficient file (if it is not present, there are no Planck-weighted channels).
- PC-RTTOV computations are limited by the configuration of the coefficient training. More information is given in section 8.6.
- UV simulations: limitations with the UV capability are discussed in section 8.1.5.
- Scattering is not supported for Stokes 3/4 channels on polarimetric sensors such as Windsat and COWVR.

10 Reporting and known bugs for RTTOV v14

Bug reports or other comments/feedback can be submitted via the NWP SAF helpdesk: <https://nwp-saf.eumetsat.int/site/help-desk/>. Select RTTOV as the "department" and include the following information:

- RTTOV version number (i.e., v14.0)
- Platform and operating system you are running the code on (e.g. Linux PC, IBM, Cray)
- Compiler used (e.g. *gfortran*, *ifort*, *pgf90*, etc) and compilation flags
- Classification of report as: serious, cosmetic or improvement
- Report of problem including any error messages and/or any input/output files the SAF can use to reproduce the problem.

Once the problem has been analysed it will be posted on the RTTOV web site (see below) with a description of the fix if appropriate. There is also an RTTOV v14 email list where major bugs are announced. When you register to download RTTOV you will be automatically included on this list unless you indicate otherwise.

Known issues and bugs and, where available, corrections, will be provided via the RTTOV web pages as they become known:

<https://nwp-saf.eumetsat.int/site/software/rttov/rttov-v14/code-updates/>

11 Frequently asked questions



Answers to frequently asked questions are available on the RTTOV web site:

<https://nwp-saf.eumetsat.int/site/software/rttov/documentation/rttov-faqs/>

12 Glossary

AD	RTTOV Adjoint model
Aertable	Aerosol optical property file for use with RTTOV
AMSU	Advanced Microwave Sounding Unit
ATMS	Advanced Technology Microwave Sounder
ATOVS	Advanced TIROS Operational Vertical Sounder
B parameter/BPR	Back-scattering parameter for IR scattering simulations
BRDF	Bi-directional Reflectance Distribution Function
BRF	Bi-directional Reflectance Factor
BT	Brightness Temperature

CAMS	Copernicus Atmosphere Monitoring Service
CLW	Cloud Liquid Water
CIW	Cloud Ice Water
CNRM	Centre National de Recherches Météorologiques
CPR	Cloud Profiling Radar
DOM	Discrete Ordinates Method
DPR	Dual-frequency Precipitation Radar
ECMWF	European Centre for Medium-Range Weather Forecasts
EUMETSAT	European Organisation for the Exploitation of Meteorological Satellites
FASTEM	MW surface emissivity model
GCM	Global Circulation Model
GEO	Geostationary
GPM	Global Precipitation Measurement mission
HDF5	Hierarchical Data Format version 5
Hydrotable	Hydrometeor optical property file for use with RTTOV
ICON-ART	ICOsahedral Nonhydrostatic model with Aerosols and Reactive Trace gases
IR	Infrared
IREMIS	Newer physically-based IR sea surface emissivity model
ISEM	Older IR sea surface emissivity model, depends only on zenith angle
IWC	Ice Water Content
K	RTTOV Jacobian model
LBLRTM	Line-By-Line Radiative Transfer Model used to generate RTTOV coefficients for UV/VIS/IR sensors
LEO	Low Earth Orbit
LUT	Look-Up Table
Liebe-89 MPM	Line-by-line model used to generate RTTOV coefficients for MW sensors
MFASIS	Method for FAst Satellite Image Simulation, a fast visible/near-IR cloudy radiance solver
MODIS	MODERate resolution Imaging Spectroradiometer
MHS	Microwave Humidity Sounder
MW	Microwave
netCDF	network Common Data Form
NIR	Near-Infrared (see VIS below)
NLTE	Non Local Thermodynamic Equilibrium
NN	Neural Network
NWP	Numerical Weather Prediction
OPAC	Optical Properties of Aerosols and Clouds
OpenMP	Application Programming Interface supporting multi-platform shared-memory parallel programming
PC	Principal Components
PMC	Pressure Modulated Cell
PMR	Pressure Modulator Radiometer
PW	Planck-Weighted
RTTOV	Radiative Transfer for TOVS
RTTOV-SCATT	RTTOV interface for MW hydrometeor scattering simulations (RTTOV v13 and earlier)
TELSEM2	A Tool to Estimate Land Surface Emissivities from Microwaves to Millimetres
TL	RTTOV Tangent Linear model
SAF	Satellite Applications Facility
SEVIRI	Spinning Enhanced Visible and Infrared Imager
SSMIS	Special Sensor Microwave Imager/Sounder
SSU	Stratospheric Sounding Unit
SURFEM-Ocean	SURface Fast Emissivity Model for Ocean
TIROS	Television Infrared Observation Satellite
TOA	Top Of Atmosphere
TOVS	TIROS Operational Vertical Sounder
UV	Ultraviolet (see VIS below)
VIS	Visible (here used synonymously with UV/VIS/NIR)

		RTTOV v14 Users' Guide	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	-------------------------------	---

13 References

Science and Validation Reports for all versions of RTTOV going back to v6 are available on the RTTOV web site: <https://nwp-saf.eumetsat.int/site/software/rttov/documentation/>

Baordo, F. and Geer, A.J., 2016. Assimilation of SSMIS humidity-sounding channels in all-sky conditions over land using a dynamic emissivity retrieval. *Q.J.R. Meteorol. Soc.*, **142**, 2854-2866.

Barlakas, V, Geer A.J., and Eriksson, P, 2021: Introducing hydrometeor orientation into all-sky millimeter/sub-millimeter assimilation, *Atmos. Meas. Tech.* 14, 3427–3447, 2021, <https://doi.org/10.5194/amt-14-3427-2021>

Barlakas, V, Geer A.J., and Eriksson, P, 2022: Cloud particle orientation and polarisation for cross-track microwave sensors in RTTOV, *NWP SAF report NWP_AVS22_01*, Version 1.0, 18/08/2022
https://nwp-saf.eumetsat.int/publications/vs_reports/nwpsaf-ec-vs-061.pdf

Bauer P., E. Moreau, F. Chevallier, and U. O'Keeffe, 2006: Multiple-scattering microwave radiative transfer for data assimilation applications. *Q.J.R. Meteorol. Soc.* **132**, 1259-1281.

Baum, B. A., P. Yang, A. J. Heymsfield, C. Schmitt, Y. Xie, A. Bansemmer, Y. X. Hu, and Z. Zhang, 2011: Improvements to shortwave bulk scattering and absorption models for the remote sensing of ice clouds. *J. Appl. Meteor. Clim.*, **50**, 1037-1056.

Bormann, N, A. Geer, S. English, 2012: Evaluation of the microwave ocean surface emissivity model FASTEM-5 in the IFS. *ECMWF Technical Memorandum 667*.

Borbas, E. and B. C. Ruston, 2010: The RTTOV UWiremis IR land surface emissivity module. *NWP SAF report*.
https://nwp-saf.eumetsat.int/publications/vs_reports/nwpsaf-mo-vs-042.pdf

Borbas, E, 2014: The RTTOV UWiremis module Investigation into the angular dependence of IR surface emissivity. *NWP SAF report*. https://nwp-saf.eumetsat.int/publications/vs_reports/nwpsaf-mo-vs-050.pdf

Borbas, E. and Feltz M., 2019: Updating the CAMEL surface emissivity atlas for RTTOV. *NWP SAF report NWPSAF-MO-VS-058*. https://nwp-saf.eumetsat.int/publications/vs_reports/nwpsaf-mo-vs-058.pdf

Boudala, F.S., Isaac, G.A., Fu, Q., and Cober, S.G., 2002: Parameterization of effective ice particle size for high latitude clouds. *Int. J. Climatol.*, **22**, 1267-1284.

Bozzo, A., Remy, S., Benedetti, A., Flemming, J., Bechtold, P., Rodwell, M.J., Morcrette, J.J., 2017: Implementation of a CAMS-based aerosol climatology in the IFS. *ECMWF Technical Memorandum 801*



Brunel, P. and S. Turner, 2003: On the use of Planck-weighted transmittances in RTTOV presented at the 13th International TOVS Study Conference, Ste Adele, Canada 29 Oct – 4 Nov 2003.
https://itwg.ssec.wisc.edu/wordpress/wp-content/uploads/2023/05/a20_brunel_poster.pdf

Deblonde, G., 2000. Evaluation of FASTEM and FASTEM2, *NWP SAF report NWPSAF-MO-VS 1*. Available here:
<https://nwp-saf.eumetsat.int/site/download/documentation/rtm/papers/evalfastems.pdf>
<https://nwp-saf.eumetsat.int/site/download/documentation/rtm/papers/evalfastemsfigs.pdf>

Elfouhaily, T., Chapron, B., Katsaros, K., and Vandemark, D., 1997: A unified directional spectrum for long and short wind-driven waves, *J. Geophys. Res.*, **102**(C7), 15781-15796, doi:10.1029/97JC00467.

Emde C., R. Buras-Schnell, A. Kylling, B. Mayer, J. Gasteiger, U. Hamann, J. Kylling, B. Richter, C. Pause, T. Dowling, and L. Bugliaro, 2016. The libradtran software package for radiative transfer calculations (version 2.0.1). *Geoscientific Model Development*, 9(5):1647-1672.

English, S.J. and T.J. Hewison, 1998: A fast generic microwave emissivity model, *Proceedings of SPIE*, 3503, *Microwave Remote Sounding of the Environment*, Eds. T Hayasaka, D.L. Wu, Y. Jin and J. Jiang, 288-300

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

English, S., Prigent, C., Johnson, B., Yueh, S., Dinnat, E., Boutin, J., Newman, S., Anguelova, M., Meissner, T., Kazumori, M., Weng, F., Supply, A., Kilic, L., Bettenhausen, M., Stoffelen, A., and Accadia, C., 2020: Reference-quality emission and backscatter modeling for the ocean. *Bulletin of the American Meteorological Society*, **101** (10), E1593 - E1601. Retrieved from <https://journals.ametsoc.org/view/journals/bams/101/10/bamsD200085.xml> doi: 10.1175/BAMS-D-20-0085.1

Eyre J. R., 1991: A fast radiative transfer model for satellite sounding systems. *ECMWF Technical Memorandum 176*

Geer A.J., P. Bauer and C. W. O'Dell, 2009a: A revised cloud overlap scheme for fast microwave radiative transfer in rain and cloud, *J. App. Met. Clim.*, **48**, 2257–2270

Geer, A.J., R.M. Forbes and P. Bauer, 2009b: Cloud and precipitation overlap in simplified scattering radiative transfer, *EUMETSAT/ECMWF Fellowship Programme Research Report no. 18*.
<https://www.ecmwf.int/sites/default/files/elibrary/2009/9516-cloud-and-precipitation-overlap-simplified-scattering-radiative-transfer.pdf>

Geer A. J., Bauer B., Lonitz K., Barlakas V., Eriksson P., Mendrok J., Doherty A., Hocking J., and Chambon P., 2021: Hydrometeor optical properties for microwave and sub-mm radiative transfer in RTTOV v13.0. *Geosci. Model Dev.*, **14**, 7497–7526, 2021, <https://doi.org/10.5194/gmd-14-7497-2021>

Guedj S., Karbou F., Rabier F., and Bouchard A., 2010: Toward a Better Modeling of Surface Emissivity to Improve AMSU Data Assimilation Over Antarctica. *IEEE Trans. Geosci. Remote Sensing* **48** 4
<https://doi.org/10.1109/TGRS.2009.2036254>

Han, H.-J., B.-J. Sohn, H.-L. Huang, E. Weisz, R. Saunders, and T. Takamura, 2012: An improved radiance simulation for hyperspectral infrared remote sensing of Asian dust, *J. Geophys. Res.*, **117**, D09211, doi:10.1029/2012JD017466.

Hocking, J., 2014: Interpolation methods in the RTTOV fast radiative transfer model. Met Office Forecasting Research Technical Report 590.
https://digital.nmla.metoffice.gov.uk/download/file/digitalFile_911bd873-f30f-4617-9810-ad73b5457ea1

Hocking, J., J. Vidot, P. Brunel, P. Roquet, B. Silveira, E. Turner, and C. Lupu, 2021: A new gas absorption optical depth parameterisation for RTTOV v13. *Geosci. Model Dev.* **14**, 2899-2915, <https://doi.org/10.5194/gmd-14-2899-2021>

Johnson, B., K. Turnbull, P. Brown, R. Burgess, J. Dorsey, A. J. Baran, H. Webster, J. Haywood, R. Cotton, Z. Ulanowski, E. Hesse, A. Woolley, and P. Rosenberg, 2012: In situ observations of volcanic ash clouds from the FAAM aircraft during the eruption of Eyjafjallajökull in 2010, *J. Geophys. Res.*, **117**, D00U24, doi:10.1029/2011JD016760.

Karbou, F., E.Gérard, and F. Rabier, 2006: Microwave land emissivity and skin temperature for AMSU-A and -B assimilation over land, *Q.J.R. Meteorol. Soc.* **132**, No. 620, Part A, pp. 2333-2355(23), doi :10.1256/qj.05.216



Karbou, F., E. Gérard, and F. Rabier, 2010: Global 4DVAR assimilation and forecast experiments using AMSU observations over land. Part I: Impacts of various land surface emissivity parameterizations. *Wea. Forecasting*, **25**, 5–19.

Kazumori, M. and S.J. English, 2015: Use of the ocean surface wind direction signal in microwave radiance assimilation, *Q.J.R. Meteorol. Soc.* **141**, No. 689, Part B, pp 1354–1375, doi: 10.1002/qj.2445

Kilic, L., C. Prigent, C. Jimenez, E. Turner, J. Hocking, S. English, T. Meissner, E. Dinnat, 2023: Development of the SURface Fast Emissivity Model for Ocean (SURFEM-Ocean) based on the PARMIO radiative transfer model. *Earth and Space Science*, **10**, 11, <https://doi.org/10.1029/2022EA002785>

Kokaly, R.F., Clark, R.N., Swayze, G.A., Livo, K.E., Hoefen, T.M., Pearson, N.C., Wise, R.A., Benzel, W.M., Lowers, H.A., Driscoll, R.L., and Klein, A.J., 2017: USGS Spectral Library Version 7: U.S. Geological Survey Data Series 1035, 61 p., <https://doi.org/10.3133/ds1035>.

Kurucz, R.L., 1992: Synthetic infrared spectra, in *Infrared Solar Physics*, *IAU Symp. 154*, edited by D.M. Rabin and J.T. Jefferies, Kluwer, Acad., Norwell, MA.

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

Labonnote, L., Hocking, J. and Vidot, J., 2022: Improvement of the scattering parameterisation in RTTOV. *NWP SAF report NWPSAF-MO_VS-059* https://nwp-saf.eumetsat.int/publications/vs_reports/nwpsaf-mo-vs-059.pdf

Liebe, H.J., 1989: An atmospheric millimeter-wave propagation model. *Int. J. of Infrared and Millimeter Waves*, **10**, 6.

Liu Q. and F. Weng, 2003. Retrieval of sea surface wind vector from simulated satellite microwave polarimetric measurements. *Radio Sci.* **38**, 8078, doi: 10.1029/2002RS002729.

Liu, Q., F. Weng, and S. English, 2011: An Improved Fast Microwave Water Emissivity Model, *IEEE Geosci. Remote Sensing*, **49** 1238 – 1250. doi: 10.1109/TGRS.2010.2064779.

Martin, G.M., D.W. Johnson, and A. Spice, 1994: The Measurement and Parameterization of Effective Radius of Droplets in Warm Stratocumulus Clouds. *J. Atmos. Sci.*, **51**, 1823–1842, [https://doi.org/10.1175/1520-0469\(1994\)051<1823:TMAPOE>2.0.CO;2](https://doi.org/10.1175/1520-0469(1994)051<1823:TMAPOE>2.0.CO;2)

Matricardi, M., F. Chevallier and S. Tjemkes, 2001: An improved general fast radiative transfer model for the assimilation of radiance observations. *ECMWF Technical Memorandum 345*.

Matricardi, M., 2003: RTIASI-4, a new version of the ECMWF fast radiative transfer model for the infrared atmospheric sounding interferometer. *ECMWF Technical Memorandum 425*.

Matricardi, M., 2005: The inclusion of aerosols and clouds in RTIASI, the ECMWF fast radiative transfer model for the Infrared Atmospheric Sounding Interferometer. *ECMWF Technical Memorandum 474*.

Matricardi, M., 2008: The generation of RTTOV regression coefficients for IASI and AIRS using a new profile training set and a new line-by-line database. *ECMWF Technical Memorandum 564*.

Matricardi M., López Puertas M., and Funke B., 2018: Modelling of nonlocal thermodynamic equilibrium effects in the principal component based version of the RTTOV fast radiative transfer model. *JGR Atmos.* **123**, 11, 5741-5761 <https://doi.org/10.1029/2018JD028657>

Matricardi, M., 2024: Enhancing the capabilities of the PC-RTTOV model. *NWP SAF report*. https://nwp-saf.eumetsat.int/publications/vs_reports/nwpsaf-ec-vs-063.pdf

Matzler, C. 2005 On the determination of surface emissivity from Satellite observations. *Geoscience and Remote Sensing Letters*, **2**, 160-163.

McFarquhar, G.M., Iacobellis, S. & Somerville, R.C.J., 2003: SCM simulations of tropical ice clouds using observationally based parameterizations of microphysics. *J. Clim.*, **16**, 1643-1664.

Muser, L., C. Stumpf, L. Scheck, C. Köpken-Watts, 2022: Preparing an extension of RTTOV-DOM for application to ICON-ART aerosol. *NWP SAF report NWPSAF-DWD-VS-059*, Version 1.0, 14/04/2022. https://nwp-saf.eumetsat.int/publications/vs_reports/nwpsaf-dwd-vs-059.pdf



Ou, S. & Liou, K.-N., 1995: Ice microphysics and climatic temperature feedback. *Atmos. Res.*, **35**, 127-138.

Pollack, J. B., O. B. Toon, and B. N. Khare, 1973: Optical properties of some terrestrial rocks and minerals, *Icarus*, **19**, 372–389, doi:10.1016/0019-1035(73)90115-2.

Rochon, Y., L. Garand, D.S. Turner and S. Polavarapu. 2007: Jacobian mapping between vertical co-ordinate systems in data assimilation. *Q.J.R. Meteorol. Soc.* **133** 1547-1558.

Rosenkranz, P.W., 2015: A Model for the Complex Dielectric Constant of Supercooled Liquid Water at Microwave Frequencies. *IEEE Trans. on Geosci. and Remote Sensing*, **53**, 3, 1387-1393.

Saunders R.W., M. Matricardi and P. Brunel, 1999: An Improved Fast Radiative Transfer Model for Assimilation of Satellite Radiance Observations. *Q.J.R. Meteorol. Soc.* **125**, 1407-1425.

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

Saunders, R., Hocking, J., Turner, E., Rayer, P., Rundle, D., Brunel, P., Vidot, J., Roquet, P., Matricardi, M., Geer, A., Bormann, N., and Lupu, C., 2018: An update on the RTTOV fast radiative transfer model (currently at version 12), *Geosci. Model Dev.*, **11**, 2717-2737, <https://doi.org/10.5194/gmd-11-2717-2018>

Scheck L., 2021: A neural network based forward operator for visible satellite images and its adjoint, *JQSRT*, **274**, 107841, <https://doi.org/10.1016/j.jqsrt.2021.107841>.

Segelstein, D., 1981: The Complex Refractive Index of Water. Masters Thesis, Dept Physics, U. Missouri-Kansas City.

Sherlock, V., 1999: ISEM-6: Infrared Surface Emissivity Model for RTTOV-6. *NWP SAF report*.
<https://nwp-saf.eumetsat.int/site/download/documentation/rtm/papers/isem6.pdf>

Stamnes, K., S.-C. Tsay, W. Wiscombe and K. Jayaweera, 1988: Numerically stable algorithm for discrete-ordinate-method radiative transfer in multiple-scattering and emitting layered media. *Applied Optics*, **27**, 2502-2509.

Tang, G., P. Yang, G. W. Kattawar, X. Huang, E. J. Mlawer, B. A. Baum and M. King, 2018: Improvement of the simulation of Cloud Longwave Scattering in Broadband Radiative Transfer Models, *J. of Atmos. Sci.*, 2217–2233, doi.org/10.1175/JAS-D-18-0014.1

Turner, D.D., Kneifel, S., and Cadetdu, M.P., 2016: An Improved Liquid Water Absorption Model at Microwave Frequencies for Supercooled Liquid Water Clouds. *J. Atmos. Oceanic Tech.*, **33**, 33-44, doi: 10.1175/JTECH-D-15-0074.1

Vidot, J. and E. Borbas, 2013: Land surface VIS/NIR BRDF atlas for RTTOV-11: Model and Validation against SEVIRI Land SAF Albedo product. *Q.J.R. Meteorol. Soc.* **140**, 2186–2196, doi: 10.1002/qj.2288

Vidot, J., A. J. Baran, and P. Brunel, 2015: A new ice cloud parameterization for infrared radiative transfer simulation of cloudy radiances: Evaluation and optimization with IIR observations and ice cloud profile retrieval products. *J. Geophys. Res. Atmos.*, **120**, 6937–6951. doi: 10.1002/2015JD023462.

Volz, F. E. (1972), Infrared refractive index of atmospheric aerosol substances, *Appl. Opt.*, **11**, 755–759, doi:10.1364/AO.11.000755.

Volz, F. E. (1973), Infrared optical constants of ammonium sulfate, Sahara dust, volcanic pumice, and flyash, *Appl. Opt.*, **12**, 564–568, doi:10.1364/AO.12.000564.

Wang, D., C. Prigent, L. Kilic, S. Fox, R. C. Harlow, C. Jimenez, F. Aires, C. Grassotti, and F. Karbou, 2017: Surface emissivity at microwaves to millimetre waves over Polar Regions: parameterization and evaluation with aircraft experiments. *J. Atmos. and Oceanic Tech.* <https://doi.org/10.1175/JTECH-D-16-0188.1>

Wang P. and O. Tuinder, 2022: Comparison of RTTOV and DISAMAR for clear sky and aerosol cases, *NWP SAF report NWPSAF-EC_VS-062*, Version 1.0, 15/09/2022
https://nwp-saf.eumetsat.int/publications/vs_reports/nwpsaf-ec-vs-062.pdf

Wyser, K., 1998: The effective radius in ice clouds. *J. Clim.*, **11**, 1793-1802.

Yoshimori, K., K. Itoh, and Y. Ichioka, 1995: Optical characteristics of a wind-roughened water surface: a two-dimensional theory, *Appl. Opt.* **34**, 6236-6247.

14 Annexes

Annex A - Coefficient information and conversion tools

1. RTTOV_CONV_COEF.EXE

The program `rttov_conv_coef.exe` (located in the `bin/` directory of the RTTOV build) is used to convert coefficient and optical property files between ASCII, Fortran unformatted (binary), and netCDF formats and to create coefficient/optical property files for subsets of channels. A help message can be displayed as follows:

```
$ rttov_conv_coef.exe --help
```

The usage is as follows:

```
$ rttov_conv_coef.exe \
  --all-in-one \
  --format-in FORMATTED|UNFORMATTED|NETCDF \
  --format-out FORMATTED|UNFORMATTED|NETCDF \
  --channels 1 2 3 4 5 ... \
  --coef-in ... --aertable-in ... --hydrotable-in ... \
  --f_mfasis_nn-in ... --pccoef-in ... \
  --coef-out ... --aertable-out ... --hydrotable-out ... \
  --f_mfasis_nn-out ... --pccoef-out ... \
  --no-write-coef
```

Argument	Description
<code>--format-in FORMATTED UNFORMATTED NETCDF</code>	Format of input coefficient file(s). FORMATTED=ASCII; UNFORMATTED=binary, NETCDF only applicable if RTTOV compiled with netCDF capability (optional).
<code>--format-out FORMATTED UNFORMATTED NETCDF</code>	Format of output coefficient files(s).
<code>--channels 1 2 3 4 5 ...</code>	List of channels to extract (optional)
<code>--coef-in / --coef-out</code>	Input/output RTTOV coefficient file (output file optional).
<code>--aertable-in / --aertable-out</code>	Input/output aerosol optical property file (optional).
<code>--hydrotable-in / --hydrotable-out</code>	Input/output hydrometeor optical property file (optional).
<code>--mfasis_nn-in / --mfasis_nn-out</code>	Input/output MFASIS-NN coefficient files (optional). If present, the <code>--hydrotable-in/out</code> arguments must also be present.
<code>--pccoef-in / --pccoef-out</code>	Input/output Principal Components coefficient file (optional).
<code>--no-write-coef</code>	By default an <code>rtcoef</code> file is always written out. Specifying this flag disables this. Useful if you only want to convert one of the other file types (optional).
<code>--all-in-one</code>	If present write all coefs (optical depth, scattering, PC) to a single output file. Only applicable to netCDF output (optional).

Most arguments are optional, though both `--coef-in` and `--format-out` must always be specified. It is not usually necessary to specify the input file format as RTTOV determines this automatically, and input file formats need not be the same for all files being read in.

By default, an optical depth coefficient file (`rtcoef_*`) is always created. If `--coef-out` is not specified this file is written to the directory containing the input coefficient file. The `--no-write-coef` argument can be specified to disable output of the `rtcoef` file.

Optical depth coefficient files must be read when reading aertable, hydrotable, MFASIS-NN, and/or `pccoef` files. MFASIS-NN files in addition require the corresponding hydrotable file to be read in.

It is recommended to specify the output filename for all files that are being created.

As described in section 7.5, when you extract some subset of n channels to a new coefficient file the channels will then be identified by the indices 1 to n in RTTOV and not by the original channel numbers. If you are carrying out PC calculations the channels that RTTOV must simulate are prescribed by the PC predictor selection chosen as described in section 8.6. In this case, if you wish to create a smaller coefficient file for use with these simulations, it is strongly recommended that you extract specifically the set of channels used as predictors for the PC calculations you require: if you extract some other subset of channels (which must in any case be a superset of the necessary PC predictor channels) it will become complicated to manage the channel numbering correctly.

Example 1 – convert a netCDF file to binary format:

```
$ rttov_conv_coef.exe --format-out unformatted \  
  --coef-in rtcoef_eos_2_airs_o3co2.nc --coef-out rtcoef_eos_2_airs.bin
```

Example 2 – extract a subset of channels to a netCDF file:

```
$ rttov_conv_coef.exe --format-out netcdf --coef-in rtcoef_eos_2_airs_o3co2.nc \  
  --coef-out rtcoef_eos_2_airs_subset.nc --channels 10 20 30 ...
```

Example 3 – extract IR-only channels from optical depth and aertable/hydrotable files:

```
$ rttov_conv_coef.exe --format-out formatted \  
  --coef-in rtcoef_msg_4_seviri_o3co2.dat \  
  --coef-out rtcoef_msg_4_seviri_o3co2_ironly.dat \  
  --hydrotable-in rttov_hydrotable_msg_4_seviri.dat \  
  --hydrotable-out rttov_hydrotable_msg_4_seviri_ironly.dat \  
  --aertable-in rttov_aertable_msg_4_seviri_opac.dat \  
  --aertable-out rttov_aertable_msg_4_seviri_opac_ironly.dat \  
  --channels 4 5 6 7 8 9 10 11
```

2. RTTOV_COEF_INFO.EXE

The program **rttov_coef_info.exe** (located in the **bin/** directory) can be used to display information about any given *rtcoef_* coefficient file. This is particularly useful for determining the contents of binary or netCDF coefficient files.

The usage is as follows:

```
$ rttov_coef_info.exe --coef ... --format FORMATTED|UNFORMATTED|NETCDF --verbose
```

RTTOV will usually determine format of the coefficient file automatically so the `--format` argument is not generally required. The `--verbose` option prints out additional per-channel information, so the amount of output is greatly increased for hyperspectral sounders.

Argument	Description
<code>--coef</code>	Input coefficient file.
<code>--format</code>	Format of coefficient file (optional)
<code>--verbose</code>	Include additional per-channel information (optional).


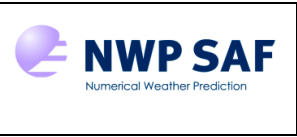
Annex B – RTTOV_ERRORHANDLING interface

```
call rttov_errorhandling (err_unit)
```

rttov_errorhandling may optionally be called at any time to set the Fortran file unit number to which output error messages are written. The default value (**default_err_unit**) is the one given in the **rttov_const** module (currently 0). On most platforms the standard error is 0, but for HP it is 7. You should set the value according to your system. If no call is made, it is the same as calling the routine with the default values.

Note that the error unit is stored in a module variable and calls to this subroutine update the module variable value, so they are not thread-safe.

Type	In/Out	Variable	Description
Integer	Intent(in)	err_unit	Logical error unit

		RTTOV v14 Users' Guide	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	------------------------	---

Annex C – Coefficient reading and deallocation subroutines

Additional information on reading coefficients is given in section 7.2.

1. RTTOV_READ_COEFS interface

```
call rttov_read_coefs (
    err, coefs, opts,
    channels, channels_rec,
    form_coef,
    form_aertable,
    form_hydratable,
    form_mfasis_nn,
    form_pccoef,
    file_coef,
    file_aertable,
    file_hydratable,
    file_mw_pol,
    file_mfasis_nn,
    file_pccoef,
    file_id_coef,
    file_id_aertable,
    file_id_hydratable,
    file_id_mfasis_nn,
    file_id_pccoef,
    instrument, path)
```

This subroutine is used to read the coefficient file(s) for one sensor required for the simulation(s) being performed. Only the arguments relevant to the required coefficient files are necessary. It is not usually necessary to specify format argument(s) (**form_***) as RTTOV determines this automatically. Note that the various input files being read do not need to be in the same format.

In all cases an optical depth coefficient file must be read.

To read an *aertable* file, the **opts%scatt%aerosols** option must be set to true and **opts%scatt%user_aer_opt_param** must be set to false.


To read a *hydratable* file, the **opts%scatt%hydrometeors** option must be set to true and **opts%scatt%user_hydro_opt_param** must be set to false.

To read an MFASIS-NN coefficient file, a hydratable file must be read in and MFASIS-NN must be selected in **opts%scatt%solar_solver** (section 8.4.2).

To read the ARO scaling polarisation coefficient file, a hydratable file must be read in and the ARO scaling polarisation option must be specified in **opts%scatt%mw_pol_mode** (section 8.4.6). This file is only available in ASCII format. The full path to the file may be specified in **file_mw_pol**. Otherwise, RTTOV will attempt to read it assuming the default file name (*ScalingFactorForBulkProperties.rssp*) either in the directory **path** if specified or otherwise in the current directory.

To read a PC-RTTOV file, the **opts%pcrttov%enable_pcrttov** option must be set to true, and the **opts%pcrttov%rec_rad** option must be true if the **channels_rec** argument is supplied to extract data for reconstructing radiances for a subset of sensor channels.

You can choose to open file(s) and pass the corresponding logical unit(s) to this routine via the **file_id_*** argument(s). Alternatively, you can pass in the full path(s) to the file(s) (recommended) via the **file_*** argument(s). If neither of these methods are used, the **instrument** argument is mandatory, specifying the “instrument ID triplet”, i.e., an integer array of (platform ID, satellite ID, instrument ID). The platform and instrument IDs are given in Tables 2.1 and 2.2. You can also use the **rttov_wmo2rttov_sat_id** subroutine (Annex I) to convert WMO satellite IDs to the RTTOV platform and satellite

<p>The EUMETSAT Network of Satellite Application Facilities</p>		<h2 style="text-align: center;">RTTOV v14 Users' Guide</h2>	<p>Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025</p>
---	---	---	--

ID couplets. When using the **instrument** triplet, RTTOV assumes the files are in the current directory, but you can specify an alternative directory via the optional **path** argument. When using the **instrument** triplet, the filename must not contain any additional text beyond the platform and sensor names and the satellite ID number. Many RTTOV coefficient and optical property files have additional text in the filename to distinguish different files. In such cases you should rename the files to remove this additional text when reading them via the **instrument** argument.

Once the optical depth file has been read (via any method described above), the instrument triplet is constructed by the code. Any additional files to be read, as determined by the options settings (described above), will be read in based on the instrument triplet if no other relevant argument (logical unit or file name) is supplied. This means, for example, that you could specify only the optical depth coefficient file path (using the **file_coef** argument) and set the relevant options to read scattering and/or PC-RTTOV files. However, you would need to ensure any additional text in these file names has been removed as noted above.

Important notes:

If the **channels(:)** argument is supplied to extract data for n channels from the coefficient file(s), then within RTTOV these channels are referred to using the indices $1...n$ rather than the original channel numbers. In particular, these new indices should be used when populating the **chanprof(:)%chan** array (section 7.5).

When reading PC-RTTOV coefficient files, if you specify the **channels(:)** argument (not recommended in this case) then this channel list must be a super-set of the channel list which forms the predictor set for the selected PC regression configuration, and you must manage the remapping of the channel numbers.

If you specify **channels_rec(:)** to extract data for reconstructing radiances for a subset of channels in PC-RTTOV, then remember that the reconstructed radiance channel numbers are also renumbered consecutively starting from 1 (as for **channels(:)** described above).

Type	In/Out	Variable	Description
Integer	Intent(out)	err	Return code
Type (rtov_coefs)	Intent(out)	coefs	RTTOV coefficients structure.
Type (rtov_options)	Intent(in)	opts	RTTOV options structure.
Integer	Intent(in), optional	channels(:)	List of channels to extract.
Integer	Intent(in), optional	channels_rec(:)	List of channels for which to calculate radiances from PC scores (only applicable if both opts%pcrttov%enable_pcrttov and opts%pcrttov%rec_rad are true).
Character	Intent(in), optional	form_coef	Format of optical depth coefficient file: should be either “ unformatted ” (binary), “ formatted ” (ASCII) or “ netcdf ” (only applicable if RTTOV has been compiled with netCDF functionality).
Character	Intent(in), optional	form_aertable	Format of aerosol optical property file.
Character	Intent(in), optional	form_hydratable	Format of hydrometeor optical property file.
Character	Intent(in), optional	form_mfasis_nn	Format of MFASIS-NN coefficient file.
Character	Intent(in), optional	form_pccoef	Format of Principal Components coefficient file.
Character	Intent(in), optional	file_coef	Full path of optical depth coefficient file.
Character	Intent(in), optional	file_aertable	Full path of aerosol optical property file.
Character	Intent(in), optional	file_hydratable	Full path of hydrometeor optical property file.
Character	Intent(in), optional	file_mw_pol	Full path of ARO scaling polarisation coefficient file.
Character	Intent(in), optional	file_mfasis_nn	Full path of MFASIS-NN coefficient file.
Character	Intent(in), optional	file_pccoef	Full path of Principal Components coefficient file.
Integer	Intent(in), optional	file_id_coef	Logical unit of pre-opened optical depth coefficient file.
Integer	Intent(in), optional	file_id_aertable	Logical unit of pre-opened aerosol optical property file.
Integer	Intent(in), optional	file_id_hydratable	Logical unit of pre-opened hydrometeor optical property file.
Integer	Intent(in), optional	file_id_mfasis_nn	Logical unit of pre-opened MFASIS-NN coefficient file.
Integer	Intent(in), optional	file_id_pccoef	Logical unit of pre-opened Principal Components coefficient file.
Integer	Intent(in), optional	instrument(3)	platform ID, satellite ID, instrument ID (see Tables 2.1 and 2.2). If no filename is supplied, the instrument argument is used to construct the coefficient file name(s).
Character	Intent(in), optional	path	Used with instrument argument, directory containing coefficient files (if omitted coefficient files are assumed to be in the current directory).

2. RTTOV_DEALLOC_COEFS interface

```
call rttov_dealloc_coefs (err, coefs)
```

rttov_dealloc_coefs is called to de-allocate the memory for the **coefs** structure.

Type	In/Out	Variable	Description
Integer	Intent(out)	err	Return code
Type(rtov_coefs)	Intent(inout)	coefs	RTTOV coefficients structure.

Annex D – (De)allocation and initialisation subroutines

More information on the (de)allocation and initialisation routines is given in section 7.3.

1. RTTOV_ALLOC_DIRECT interface

```
call rttov_alloc_direct (err, alloc, nchanprof, nprofiles, nlevels, nsurfaces,
                        opts, coefs, chanprof, profiles, emis_refl,
                        transmission, radiance, radiance2, refl_cloud_top,
                        aer_maxnmom, aer_nphangle, aer_opt_param,
                        hydro_maxnmom, hydro_nphangle, hydro_opt_param,
                        reflectivity, emis_retrieval_terms, diag_output,
                        npcscores, nchannels_rec, pccomp, channels_rec,
                        traj, traj_dyn, traj_sta, init)
```

rttov_alloc_direct may be called to (de)allocate any or all input arrays and structures for the RTTOV direct model. It is often convenient to call a single allocation subroutine rather than multiple subroutines. The order of the arguments closely mirrors the arguments for **rttov_direct**.

All array arguments passed to this subroutine (e.g., **chanprof**, **profiles**, **emis_refl**) must be unallocated pointers. The array arguments themselves are (de)allocated by the call to this subroutine as well as any members within the derived types.

Due to the large number of optional arguments, it is common that named arguments must be supplied.

Type	In/Out	Variable	Description
Integer	Intent(out)	err	Return code
Logical	Intent(in)	alloc	(De)allocation switch (true=allocate; false=deallocate)
Integer	Intent(in)	nchanprof	Number of channels simulated per call to RTTOV (size of chanprof array)
Integer	Intent(in)	nprofiles	Number of profiles per call to RTTOV
Integer	Intent(in)	nlevels	Number of profile pressure half-levels
Integer	Intent(in)	nsurfaces	Number of surfaces associated with each profile
Type(rttov_options)	Intent(in)	opts	RTTOV options structure
Type(rttov_coefs)	Intent(in)	coefs	RTTOV coefficients structure
Type(rttov_chanprof)	Pointer, optional	chanprof(:)	chanprof structure array pointer: (de)allocated after call
Type(rttov_profile)	Pointer, optional	profiles(:)	Profiles structure array pointer to be (de)allocated: the profiles array itself will be (de)allocated as well as the member arrays.
Type(rttov_emis_refl)	Pointer, optional	emis_refl(:)	Surface emissivity/reflectance structure array pointer to be (de)allocated: the emis_refl array itself will be (de)allocated as well as the member arrays.
Type(rttov_transmission)	Intent(inout), optional	transmission	Transmission structure to be (de)allocated
Type(rttov_radiance)	Intent(inout), optional	radiance	Radiance structure to be (de)allocated
Type(rttov_radiance2)	Intent(inout), optional	radiance2	Secondary radiance structure to be (de)allocated

Continued on next page...

Type	In/Out	Variable	Description
Real	Pointer, optional	refl_cloud_top(:)	Array pointer for cloud top reflectances, optionally used with the simple cloud scheme
Integer	Intent(in), optional	aer_maxnmom	Maximum number of coefficients in Legendre expansion of aerosol phase functions. Required if aer_opt_param argument is present, may be zero unless DOM solver is used.
Integer	Intent(in), optional	aer_nphangle	Number of phase angles over which aerosol phase functions are to be defined. Required if aer_opt_param argument is present, may be zero unless DOM solar solver is used.
Type(rtov_opt_param)	Intent(inout), optional	aer_opt_param	Explicit aerosol optical properties structure to be (de)allocated.
Integer	Intent(in), optional	hydro_maxnmom	Maximum number of coefficients in Legendre expansion of hydrometeor phase functions. Required if hydro_opt_param argument is present, may be zero unless DOM solver is used.
Integer	Intent(in), optional	hydro_nphangle	Number of phase angles over which hydrometeor phase functions are to be defined. Required if hydro_opt_param argument is present, may be zero unless DOM solar solver is used.
Type(rtov_opt_param)	Intent(inout), optional	hydro_opt_param	Explicit hydrometeor optical properties structure to be (de)allocated.
Type(rtov_reflectivity)	Intent(inout), optional	reflectivity	Output reflectivities from radar simulator
Type(rtov_emis_retrieval_terms)	Intent(inout), optional	emis_retrieval_terms	Output data for dynamic emissivity retrievals
Type(rtov_diagnostic_output)	Intent(inout), optional	diag_output	Output diagnostic data
Integer	Intent(in), optional	npcscores	Number of principal components to calculate for each profile multiplied by the number of profiles. Required if pccomp argument present.
Integer	Intent(in), optional	nchannels_rec	The number of channels for which reconstructed radiances are required multiplied by the number of profiles. Required for allocation calls if the channels_ref(:) argument is present or if opts%pcrttov%rec_rad is true and pccomp argument present.
Type(rtov_pccomp)	Intent(inout), optional	pccomp	PC-RTTOV output structure to be (de)allocated
Integer	Pointer, optional	channels_rec(:)	Array pointer for list of channel numbers for which to reconstruct radiances for PC-RTTOV.
Type(rtov_traj)	Intent(inout), optional	traj	Trajectory structure
Type(rtov_traj_dyn)	Intent(inout), optional	traj_dyn	Dynamic trajectory structure
Type(rtov_traj_sta)	Intent(inout), optional	traj_sta	Static trajectory structure
Logical	Intent(in), optional	init	Initialise the newly allocated structures (default: false).

2. RTTOV_ALLOC_TL interface

```
call rttov_alloc_tl (err, alloc, nchanprof, nprofiles, nlevels, nsurfaces,
                    opts, coefs, chanprof, profiles, profiles_tl,
                    emis_refl, emis_refl_tl, transmission, transmission_tl,
                    radiance, radiance_tl, radiance2, refl_cloud_top,
                    aer_maxnmom, aer_nphangle, aer_opt_param, aer_opt_param_tl,
                    hydro_maxnmom, hydro_nphangle, hydro_opt_param,
                    hydro_opt_param_tl, reflectivity, reflectivity_tl,
                    diag_output, diag_output, npcscores, nchannels_rec,
                    pccomp, pccomp_tl, channels_rec, traj, traj_tl,
                    traj_dyn, traj_tl_dyn, traj_sta, init)
```

rttov_alloc_tl may be called to (de)allocate any or all input arrays and structures for the RTTOV TL model. It is often convenient to call a single allocation subroutine rather than multiple subroutines. The order of the arguments closely mirrors the arguments for **rttov_tl**.

All array arguments passed to this subroutine (e.g., **chanprof**, **profiles**, **emis_refl**) must be unallocated pointers. The array arguments themselves are (de)allocated by the call to this subroutine as well as any members within the derived types.

Due to the large number of optional arguments, it is common that named arguments must be supplied. Most of the arguments are common to **rttov_alloc_direct**: only the additional arguments are listed in the following table.

Type	In/Out	Variable	Description
Type(rttov_profile)	Pointer, optional	profiles_tl(:)	Profiles TL structure array pointer to be (de)allocated: the profiles_tl array itself will be (de)allocated as well as the member arrays.
Type(rttov_emis_refl)	Pointer, optional	emi_refl_tl(:)	Surface emissivity/reflectance TL structure array pointer to be (de)allocated: the emis_refl_tl array itself will be (de)allocated as well as the member arrays.
Type(rttov_transmission)	Intent(inout), optional	transmission_tl	Transmission TL structure to be (de)allocated
Type(rttov_radiance)	Intent(inout), optional	radiance_tl	Radiance TL structure to be (de)allocated
Type(rttov_opt_param)	Intent(inout), optional	aer_opt_param_tl	Explicit aerosol optical property TL structure to be (de)allocated.
Type(rttov_opt_param)	Intent(inout), optional	hydro_opt_param_tl	Explicit hydrometeor optical property TL structure to be (de)allocated.
Type(rttov_reflectivity)	Intent(inout), optional	reflectivity_tl	Reflectivities TL for radar simulator
Type(rttov_pccomp)	Intent(inout), optional	pccomp_tl	PC-RTTOV PC scores and reconstructed radiance TL structure to be (de)allocated.
Type(rttov_traj)	Intent(inout), optional	traj_tl	Trajectory TL structure
Type(rttov_traj_dyn)	Intent(inout), optional	traj_tl_dyn	Dynamic trajectory TL structure

3. RTTOV_ALLOC_AD interface

```
call rttov_alloc_ad (err, alloc, nchanprof, nprofiles, nlevels, nsurfaces,
                    opts, coefs, chanprof, profiles, profiles_ad,
                    emis_refl, emis_refl_ad, transmission, transmission_ad,
                    radiance, radiance_ad, radiance2, refl_cloud_top,
                    aer_maxnmom, aer_nphangle, aer_opt_param, aer_opt_param_ad,
                    hydro_maxnmom, hydro_nphangle, hydro_opt_param,
                    hydro_opt_param_ad, reflectivity, reflectivity_ad,
                    diag_output, diag_output, npcscores, nchannels_rec,
                    pccomp, pccomp_ad, channels_rec, traj, traj_ad,
                    traj_dyn, traj_ad_dyn, traj_sta, init)
```

rttov_alloc_ad may be called to (de)allocate any or all input arrays and structures for the RTTOV AD model. It is often convenient to call a single allocation subroutine rather than multiple subroutines. The order of the arguments closely mirrors the arguments for **rttov_ad**.

All array arguments passed to this subroutine (e.g., **chanprof**, **profiles**, **emis_refl**) must be unallocated pointers. The array arguments themselves are (de)allocated by the call to this subroutine as well as any members within the derived types.

Due to the large number of optional arguments, it is common that named arguments must be supplied. Most of the arguments are common to **rttov_alloc_direct**: only the additional arguments are listed in the following table.

Type	In/Out	Variable	Description
Type(rttov_profile)	Pointer, optional	profiles_ad(:)	Profiles AD structure array pointer to be (de)allocated: the profiles_ad array itself will be (de)allocated as well as the member arrays.
Type(rttov_emis_refl)	Pointer, optional	emi_refl_ad(:)	Surface emissivity/reflectance AD structure array pointer to be (de)allocated: the emis_refl_ad array itself will be (de)allocated as well as the member arrays.
Type(rttov_transmission)	Intent(inout), optional	transmission_ad	Transmission AD structure to be (de)allocated
Type(rttov_radiance)	Intent(inout), optional	radiance_ad	Radiance AD structure to be (de)allocated
Type(rttov_opt_param)	Intent(inout), optional	aer_opt_param_ad	Explicit aerosol optical property AD structure to be (de)allocated.
Type(rttov_opt_param)	Intent(inout), optional	hydro_opt_param_ad	Explicit hydrometeor optical property AD structure to be (de)allocated.
Type(rttov_reflectivity)	Intent(inout), optional	reflectivity_ad	Reflectivities AD for radar simulator
Type(rttov_pccomp)	Intent(inout), optional	pccomp_ad	PC-RTTOV PC scores and reconstructed radiance AD structure to be (de)allocated.
Type(rttov_traj)	Intent(inout), optional	traj_ad	Trajectory AD structure
Type(rttov_traj_dyn)	Intent(inout), optional	traj_ad_dyn	Dynamic trajectory AD structure

4. RTTOV_ALLOC_K interface

```
call rttov_alloc_k (err, alloc, nchanprof, nprofiles, nlevels, nsurfaces,
    opts, coefs, chanprof, profiles, profiles_k, emis_refl,
    emis_refl_k, transmission, transmission_k, radiance,
    radiance_k, radiance2, refl_cloud_top, aer_maxnmom,
    aer_nphangle, aer_opt_param, aer_opt_param_k, hydro_maxnmom,
    hydro_nphangle, hydro_opt_param, hydro_opt_param_k,
    reflectivity, reflectivity_k, diag_output, diag_output,
    npcscscores, nchannels_rec, pccomp, pccomp_k, profiles_k_pc,
    profiles_k_rec, channels_rec, traj, traj_k,
    traj_dyn, traj_k_dyn, traj_sta, init)
```

rttov_alloc_k may be called to (de)allocate any or all input arrays and structures for the RTTOV K model. It is often convenient to call a single allocation subroutine rather than multiple subroutines. The order of the arguments closely mirrors the arguments for **rttov_k**.

All array arguments passed to this subroutine (e.g., **chanprof**, **profiles**, **emis_refl**) must be unallocated pointers. The array arguments themselves are (de)allocated by the call to this subroutine as well as any members within the derived types.

Due to the large number of optional arguments, it is common that named arguments must be supplied. Most of the arguments are common to **rttov_alloc_direct**: only the additional arguments are listed in the following table.

Type	In/Out	Variable	Description
Type(rttov_profile)	Pointer, optional	profiles_k(:)	Profiles K structure array pointer to be (de)allocated: the profiles_k array itself will be (de)allocated as well as the member arrays.
Type(rttov_emis_refl)	Pointer, optional	emi_refl_k(:)	Surface emissivity/reflectance K structure array pointer to be (de)allocated: the emis_refl_k array itself will be (de)allocated as well as the member arrays.
Type(rttov_transmission)	Intent(inout), optional	transmission_k	Transmission K structure to be (de)allocated
Type(rttov_radiance)	Intent(inout), optional	radiance_k	Radiance K structure to be (de)allocated
Type(rttov_opt_param)	Intent(inout), optional	aer_opt_param_k	Explicit aerosol optical property K structure to be (de)allocated.
Type(rttov_opt_param)	Intent(inout), optional	hydro_opt_param_k	Explicit hydrometeor optical property K structure to be (de)allocated.
Type(rttov_reflectivity)	Intent(inout), optional	reflectivity_k	Reflectivities K for radar simulator
Type(rttov_pccomp)	Intent(inout), optional	pccomp_k	PC-RTTOV PC scores and reconstructed radiance K structure to be (de)allocated.
Type(rttov_profile)	Pointer, optional	profiles_k_pc(:)	PC score Jacobian structure array pointer to be (de)allocated: the profiles_k_pc array itself will be (de)allocated as well as the member arrays.
Type(rttov_profile)	Pointer, optional	profiles_k_rec(:)	PC reconstructed radiance Jacobian structure array pointer to be (de)allocated: the profiles_rec_k array itself will be (de)allocated as well as the member arrays.
Type(rttov_traj)	Intent(inout), optional	traj_k	Trajectory K structure
Type(rttov_traj_dyn)	Intent(inout), optional	traj_k_dyn	Dynamic trajectory K structure

5. RTTOV_ALLOC_PROFILES interface

```
call rttov_alloc_profiles (err, alloc, profiles, nlevels, nsurfaces,
                           opts, coefs, init)
```

rttov_alloc_profiles is called to allocate or de-allocate the members of an array of **rttov_profile** structures. For the direct, TL, and AD models, the **profiles** argument should be an array of size **nprofiles** (the number of profiles being passed to RTTOV in each call). When allocating **profiles_k(:)** for the K model the size of the array is **nchanprof**. For the PC-RTTOV K model, **profiles_k_pc(:)** should be of size **npcscores** (the total number of PC scores to calculate for all simulated profiles), or **profiles_k_rec(:)** should be of size **nchannels_rec** (the total number of reconstructed radiances for all simulated profiles) – see section 8.6 for more information.

Type	In/Out	Variable	Description
Integer	Intent(out)	err	Return code
Logical	Intent(in)	alloc	(De)allocation switch (true=allocate; false=deallocate)
Type(rttov_profile)	Intent(inout)	profiles (:)	Array of profile structures to be (de)allocated. Array size should be nprofiles (for profiles/_tl/_ad) or nchanprof (for profiles_k). See above for PC-RTTOV profiles_k_pc(:) and profiles_k_rec(:) arrays.
Integer	Intent(in)	nlevels	Number of profile pressure half-levels
Integer	Intent(in)	nsurfaces	Number of surfaces associated with each profile
Type(rttov_options)	Intent(in)	opts	RTTOV options structure
Type(rttov_coefs)	Intent(in)	coefs	RTTOV coefficients structure.
Logical	Intent(in), optional	init	Initialise profile structures after allocation (default: false)

6. RTTOV_INIT_PROFILES interface

```
call rttov_init_profiles (profiles)
```

rttov_init_profiles is used to initialise a previously allocated array of **rttov_profile** structures. This does not modify the **nlevels**, **nlayers**, **nsurfaces**, **gas_units**, **mmr_hydro** or **mmr_aer** members.

Type	In/Out	Variable	Description
Type(rttov_profile)	Intent(inout)	profiles (:)	Array of profile structures to be initialised

7. RTTOV_ALLOC_EMIS_REFL interface

```
call rttov_alloc_emis_refl (err, alloc, emis_refl, nchanprof,
                           opts, direct, init)
```

rttov_alloc_emis_refl is called to allocate or de-allocate the members of an array of **rttov_emis_refl** structures. The **emis_refl** argument should be an array of size **nsurfaces** (the number of surfaces associated with each profile, see sections 7.4 and 8.3.10).

Type	In/Out	Variable	Description
Integer	Intent(out)	err	Return code
Logical	Intent(in)	alloc	(De)allocation switch (true=allocate; false=deallocate)
Type(rttov_emis_refl)	Intent(inout)	emis_refl (:)	Array of emis_refl structures to be (de)allocated. Array size should be nsurfaces .
Integer	Intent(in)	nchanprof	Number of channels simulated per call to RTTOV (size of chanprof array)
Type(rttov_options)	Intent(in)	opts	RTTOV options structure
Logical	Intent(in), optional	direct	Set to true (or omit) when allocating direct model structure, set to false when allocating TL/AD/K model structures (default: true).
Logical	Intent(in), optional	init	Initialise emis_refl structures after allocation (default: false)

8. RTTOV_INIT_EMIS_REFL interface

```
call rttov_init_emis_refl (emis_refl)
```

rttov_init_emis_refl is used to initialise a previously allocated array of **rttov_emis_refl** structures. This is particularly useful when calling the adjoint or K models as the **emis_refl_ad/ emis_refl_k** structures must be initialised before each call to RTTOV. When allocated in the given structure, the **calc_emis**(:), **calc_brdf**(:), and **calc_diffuse_refl**(:) members are initialised to true.

Type	In/Out	Variable	Description
Type(rttov_emis_refl)	Intent(inout), optional	emis_refl (:)	Array of emis_refl structures to initialise.

9. RTTOV_ALLOC_TRANSMISSION interface

```
call rttov_alloc_transmission (err, alloc, transmission, nchanprof, nlevels,
                               opts, direct, init)
```

rttov_alloc_transmission is called to (de)allocate the members of the **rttov_transmission** structure.

Type	In/Out	Variable	Description
Integer	Intent(out)	err	Return code
Logical	Intent(in)	alloc	(De)allocation switch (true=allocate; false=deallocate)
Type(rttov_transmission)	Intent(inout)	transmission	Transmission structure to be (de)allocated
Integer	Intent(in)	nchanprof	Number of channels simulated per call to RTTOV (size of chanprof array)
Integer	Intent(in)	nlevels	Number of profile pressure half-levels
Type(rttov_options)	Intent(in)	opts	RTTOV options structure
Logical	Intent(in), optional	direct	Set to true (or omit) when allocating direct model structure, set to false when allocating TL/AD/K model structures (default: true).
Logical	Intent(in), optional	init	Initialise transmission structure after allocation (default: false)

10. RTTOV_INIT_TRANSMISSION interface

```
call rttov_init_transmission (transmission)
```

rttov_init_transmission is used to initialise a previously allocated **transmission** structure.

Type	In/Out	Variable	Description
Type(rttov_transmission)	Intent(inout)	transmission	Transmission structure to be initialised

11. RTTOV_ALLOC_RADIANCE interface

```
call rttov_alloc_radiance (err, alloc, radiance, nchanprof, nlevels,
                           opts, radiance2, direct, init)
```

rttov_alloc_radiance is called to (de)allocate the members of the **rttov_radiance** structure and optionally also the **rttov_radiance2** structure.

Type	In/Out	Variable	Description
Integer	Intent(out)	err	Return code
Logical	Intent(in)	alloc	(De)allocation switch (true=allocate; false=deallocate)
Type(rttov_radiance)	Intent(inout)	radiance	Radiance structure to be (de)allocated
Integer	Intent(in)	nchanprof	Number of channels simulated per call to RTTOV (size of chanprof array)
Integer	Intent(in)	nlevels	Number of profile pressure half-levels
Type(rttov_options)	Intent(in)	opts	RTTOV options structure
Type(rttov_radiance2)	Intent(inout), optional	radiance2	Secondary radiance structure to be created
Logical	Intent(in), optional	direct	Set to true (or omit) when allocating direct model structure, set to false when allocating TL/AD/K model structures (default: true).
Logical	Intent(in), optional	init	Initialise radiance structure(s) after allocation (default: false)

12. RTTOV_INIT_RADIANCE interface

```
call rttov_init_radiance (radiance, radiance2)
```

rttov_init_radiance is used to initialise a previously allocated **radiance** structure, and optionally also a **radiance2** structure.

Type	In/Out	Variable	Description
Type(rttov_radiance)	Intent(inout)	radiance	Radiance structure to be initialised
Type(rttov_radiance2)	Intent(inout), optional	radiance2	Secondary radiance structure to be initialised

13. RTTOV_ALLOC_OPT_PARAM interface

call `rttov_alloc_opt_param` (err, alloc, opt_param, nchanprof, nlevels, nmom, nphangle, direct, init)

`rttov_alloc_opt_param` is called to (de)allocate the members of the `rttov_opt_param` structure (for explicit aerosol or hydrometeor optical properties, section 8.4.11).

Type	In/Out	Variable	Description
Integer	Intent(out)	err	Return code
Logical	Intent(in)	alloc	(De)allocation switch (true=allocate; false=deallocate)
Type(<code>rttov_opt_param</code>)	Intent(inout)	opt_param	Optical properties structure to be (de)allocated
Integer	Intent(in)	nchanprof	Number of channels simulated per call to RTTOV (size of chanprof array)
Integer	Intent(in)	nlevels	Number of profile pressure half-levels
Integer	Intent(in)	nmom	Maximum number of coefficients in Legendre expansion of phase functions. May be zero unless DOM solver is used.
Integer	Intent(in)	nphangle	Number of phase angles over which phase functions are to be defined. May be zero unless DOM solar solver is used.
Logical	Intent(in), optional	direct	Set to true (or omit) when allocating direct model structure, set to false when allocating TL/AD/K model structures (default: true).
Logical	Intent(in), optional	init	If true, zero the newly allocated structure (default: false). This does <i>not</i> call the <code>rttov_init_opt_param_solar</code> routine (see below).

14. RTTOV_INIT_OPT_PARAM interface

call `rttov_init_opt_param` (opt_param)

`rttov_init_opt_param` used to initialise a previously allocated `opt_param` structure.

Type	In/Out	Variable	Description
Type(<code>rttov_opt_param</code>)	Intent(inout)	opt_param	Optical properties structure to be initialised.

15. RTTOV_INIT_OPT_PARAM_SOLAR interface

call `rttov_init_opt_param_solar` (err, opts, opt_param)

`rttov_init_opt_param_solar` is called to initialise phase angle variables in a previously allocated `rttov_opt_param` structure. The `phangle(:)` member must have been initialised with the phase angle grid on which phase functions are represented. NB This is only required if `opts%rt_all%solar` is true. See section 8.4.11 for more information.

Type	In/Out	Variable	Description
Integer	Intent(out)	err	Return code
Type(<code>rttov_options</code>)	Intent(in)	opts	RTTOV options structure
Type(<code>rttov_opt_param</code>)	Intent(inout)	opt_param	Optical properties structure to be initialised.

16. RTTOV_ALLOC_REFLECTIVITY interface

```
call rttov_alloc_reflectivity (err, alloc, reflectivity,
                               nchanprof, nlevels, init)
```

rttov_alloc_reflectivity is called to (de)allocate the members of the **rttov_reflectivity** structure which contains the output reflectivities for the radar simulator (section 8.4.4).

Type	In/Out	Variable	Description
Integer	Intent(out)	err	Return code.
Logical	Intent(in)	alloc	(De)allocation switch (true=allocate; false=deallocate)
Type(rttov_reflectivity)	Intent(inout)	reflectivity	Radar reflectivity structure to be (de)allocated.
Integer	Intent(in)	nchanprof	Number of channels simulated per call to RTTOV (size of chanprof array)
Integer	Intent(in)	nlevels	Number of profile pressure half-levels
Logical	Intent(in), optional	init	Initialise reflectivity structure after allocation (default: false)

17. RTTOV_INIT_REFLECTIVITY interface

```
call rttov_init_reflectivity (reflectivity)
```

rttov_init_reflectivity is used to initialise a previously allocated **reflectivity** structure.

Type	In/Out	Variable	Description
Type(rttov_reflectivity)	Intent(inout)	reflectivity	Radar reflectivity structure to be initialised.

18. RTTOV_ALLOC_EMIS_RET_TERMS interface

```
call rttov_alloc_emis_ret_terms (err, alloc, emis_retrieval_terms, nchanprof)
```

rttov_alloc_emis_ret_terms is called to (de)allocate the members of the **rttov_emis_retrieval_terms** structure which contains data that can be used for dynamic emissivity retrievals (section 8.4.12).

Type	In/Out	Variable	Description
Integer	Intent(out)	err	Return code.
Logical	Intent(in)	alloc	(De)allocation switch (true=allocate; false=deallocate)
Type(rttov_emis_retrieval_terms)	Intent(inout)	emis_retrieval_terms	Emissivity retrieval terms structure to be (de)allocated.
Integer	Intent(in)	nchanprof	Number of channels simulated per call to RTTOV (size of chanprof array)

19. RTTOV_INIT_EMIS_RET_TERMS interface

```
call rttov_init_emis_ret_terms (emis_retrieval_terms)
```

rttov_init_emis_ret_terms is used to initialise a previously allocated **emis_retrieval_terms** structure.

Type	In/Out	Variable	Description
Type(rttov_emis_retrieval_terms)	Intent(inout)	emis_retrieval_terms	Emissivity retrieval terms structure to be initialised.

20. RTTOV_ALLOC_DIAGNOSTIC_OUTPUT interface

```
call rttov_alloc_diagnostic_output (err, alloc, diag_output,
                                     nprofiles, nlevels, opts, init)
```

rttov_alloc_diagnostic_output is called to (de)allocate the members of the **rttov_diagnostic_output** structure which contains additional outputs (section 7.8.4).

Type	In/Out	Variable	Description
Integer	Intent(out)	err	Return code.
Logical	Intent(in)	alloc	(De)allocation switch (true=allocate; false=deallocate)
Type(rttov_diagnostic_output)	Intent(inout)	diag_output	Diagnostic output structure to be (de)allocated.
Integer	Intent(in)	nprofiles	Number of profiles per call to RTTOV
Integer	Intent(in)	nlevels	Number of profile pressure half-levels
Type(rttov_options)	Intent(in)	opts	RTTOV options structure
Logical	Intent(in), optional	init	Initialise reflectivity structure after allocation (default: false)

21. RTTOV_INIT_DIAGNOSTIC_OUTPUT interface

```
call rttov_init_diagnostic_output (diag_output)
```

rttov_init_diagnostic_output is used to initialise a previously allocated **diag_output** structure.

Type	In/Out	Variable	Description
Type(rttov_diagnostic_output)	Intent(inout)	diag_output	Diagnostic output structure to be initialised.

22. RTTOV_ALLOC_PCCOMP interface

```
call rttov_alloc_pccomp (err, alloc, pccomp, npcscorcs, nchannels_rec, init)
```

rttov_alloc_pccomp is called to (de)allocate the members of the **rttov_pccomp** structure containing the outputs for the PC-RTTOV model.

Type	In/Out	Variable	Description
Integer	Intent(out)	err	Return code
Logical	Intent(in)	alloc	(De)allocation switch (true=allocate; false=deallocate)
Type(rttov_pccomp)	Intent(inout)	pccomp	PC-RTTOV outputs structure to be (de)allocated
Integer	Intent(in)	npcscorcs	Number of principal components to calculate for each profile multiplied by the number of profiles.
Integer	Intent(in), optional	nchannels_rec	The number of channels for which reconstructed radiances are required multiplied by the number of profiles. Only needed if opts%pcrttov%rec_rad is true.
Logical	Intent(in), optional	init	Initialise pccomp structure after allocation (default: false)

23. RTTOV_INIT_PCCOMP interface

```
call rttov_init_pccomp (pccomp)
```

rttov_init_pccomp is used to initialise a previously allocated **pccomp** structure.

Type	In/Out	Variable	Description
Type(rttov_pccomp)	Intent(inout)	pccomp	PC-RTTOV outputs structure to be initialised

24. RTTOV_ALLOC_TRAJ_ALL interface

```
call rttov_alloc_traj_all (err, alloc, nchanprof, nprofiles, nlevels, nsurfaces,
    opts, coefs, channels_rec, traj, traj_tl, traj_ad,
    traj_k, traj_dyn, traj_tl_dyn, traj_ad_dyn,
    traj_k_dyn, traj_sta)
```

rttov_alloc_traj_all is called to (de)allocate the members of the **rttov_traj**, **rttov_traj_dyn**, and **rttov_traj_sta** structures. These contain computation results used internally within RTTOV. Allocating them outside RTTOV and passing them in can improve performance in some cases (see section 7.3.2) but note that the trajectory structures *cannot* be used with the RTTOV parallel interfaces. The trajectory structures for the direct, TL, AD, and K models can be allocated in a single call either all together or in any combination.

NB When passing the trajectory structure to RTTOV, the **coefs** variable **MUST** be declared with the TARGET attribute.

Type	In/Out	Variable	Description
Integer	Intent(out)	err	Return code.
Logical	Intent(in)	alloc	(De)allocation switch (true=allocate; false=deallocate)
Integer	Intent(in)	nchanprof	Number of channels simulated per call to RTTOV (size of chanprof array)
Integer	Intent(in)	nprofiles	Number of profiles per call to RTTOV
Integer	Intent(in)	nlevels	Number of profile pressure half-levels
Integer	Intent(in)	nsurfaces	Number of surfaces associated with each profile
Type(rttov_options)	Intent(in)	opts	RTTOV options structure
Type(rttov_coefs)	Intent(in)	coefs	RTTOV coefficients structure.
Type(rttov_traj)	Intent(inout), optional	traj, traj_tl, traj_ad, traj_k	Trajectory structures: a structure for each core RTTOV routine can be (de)allocated with a single call to rttov_alloc_traj_all .
Type(rttov_traj_dyn)	Intent(inout), optional	traj_dyn, traj_tl_dyn, traj_ad_dyn, traj_k_dyn	Dynamic trajectory structures: a structure for each core RTTOV routine can be (de)allocated with a single call to rttov_alloc_traj_all .
Type(rttov_traj_sta)	Intent(inout), optional	traj_sta	Static trajectory structure.

Annex E – Optical property calculation subroutines

These routines are intended for use with the explicit optical property inputs for scattering simulations (section 8.4). They can help in computing optical properties required for the solver(s) being used.

1. RTTOV_BPR_INIT interface

```
call rttov_bpr_init (err, phangle)
```

rttov_bpr_init is called before calculating b parameters from phase functions using **rttov_bpr_calc**: this subroutine prepares some tables to speed up the calculations. This must be called again (after calling **rttov_dealloc_bpr** – see below) if the phase angle array changes. Due to the use of module variables for efficiency, this is not thread-safe.

Type	In/Out	Variable	Description
Integer	Intent(out)	err	Return code
Real	Intent(in)	phangle(:)	Array of angles (degrees) over which phase functions are defined, covering the range 0-180, units degrees.

2. RTTOV_BPR_CALC interface

```
call rttov_bpr_calc (err, pha, bpr, nthreads)
```

rttov_bpr_calc is used to calculate a single b parameter given a phase function. Note that this is a relatively slow calculation and as such is intended to be called “off-line” rather than within performance-critical code. If you compile RTTOV with OpenMP this subroutine can exploit multiple cores/CPU: choose the number of threads with the **nthreads** argument.

Type	In/Out	Variable	Description
Integer	Intent(out)	err	Return code
Real	Intent(in)	pha(:)	Phase function defined on phase angle grid supplied to rttov_bpr_init .
Real	Intent(out)	bpr	Calculated b parameter.
Integer	Intent(in), optional	nthreads	Use multiple threads, requires RTTOV to have been compiled with OpenMP (default = 1).

3. RTTOV_BPR_DEALLOC interface

```
call rttov_bpr_dealloc (err)
```

rttov_bpr_dealloc is called after b parameters have been calculated to deallocate the memory allocated by the call to **rttov_init_bpr**.

Type	In/Out	Variable	Description
Integer	Intent(out)	err	Return code

4. RTTOV_ASYM_CALC interface

```
call rttov_asym_calc (err, pha, phangle, asym)
```

rttov_asym_calc is used to calculate the asymmetry parameter for a given phase function.

Type	In/Out	Variable	Description
Integer	Intent(out)	err	Return code
Real	Intent(in)	pha(:)	Phase function.
Real	Intent(in)	phangle(:)	Array of angles over which phase function pha is defined, covering the range 0-180, units degrees.
Real	Intent(inout)	asym	Calculated asymmetry parameter.

5. RTTOV_LCOEF_CALC interface

```
call rttov_lcoef_calc (err, pha, phangle, nmom, lcoef, ngauss, q, w)
```

rttov_lcoef_calc is used to calculate the Legendre expansion of a given phase function. The **nmom** parameter need only equal the largest value of **opts%scatt%dom_nstreams** with which RTTOV will be called. The coefficients are calculated using a 1000-point Gaussian quadrature by default: you can specify an alternative quadrature size or, for greater efficiency (if making many calls to this subroutine), you can pre-compute a quadrature and the associated weights and pass these in.

Type	In/Out	Variable	Description
Integer	Intent(out)	err	Return code
Real	Intent(in)	pha(:)	Phase function.
Real	Intent(in)	phangle(:)	Array of angles over which phase function pha is defined, covering the range 0-180, units degrees.
Integer	Intent(in)	nmom	Number of Legendre coefficients to calculate (excluding first coefficient which is always unity).
Real	Intent(inout)	lcoef(:)	Calculated Legendre coefficients, array must be at least nmom+1 in size.
Integer	Intent(in), optional	ngauss	Choose size of Gaussian quadrature to use in calculation (the default is 1000 points).
Real	Intent(in), optional	q(:)	You can supply pre-computed quadrature points which may be faster if calling this subroutine many times.
Real	Intent(in), optional	w(SIZE(q))	Weights corresponding to quadrature q .

Annex F – Surface emissivity subroutines

This section describes the interface to the atlas subroutines and a subroutine to obtain emissivities from the internal RTTOV surface models. More information about using the atlases is provided in section 8.3.6. It is possible to use the TELSEM2 atlas without compiling RTTOV with any external library dependencies because the TELSEM2 atlas data are contained in ASCII files. The other atlas data are provided in netCDF files and as such RTTOV must be compiled with netCDF support (see section 5.3).

1. RTTOV_SETUP_EMIS_ATLAS interface

```
call rttov_setup_emis_atlas (err, opts, imonth, atlas_type, atlas,
                             atlas_id, path, coefs, ir_atlas_read_std,
                             ir_atlas_ang_corr, year, camel_version)
```

The atlases provide monthly climatologies so you must provide the month (**imonth**, 1-12) for which to load atlas data. By default, the atlas files are assumed to be in the current directory, but you can optionally specify the **path** to the directory containing the atlas data files. The type of atlas (MW or IR) is selected by the **atlas_type** argument. It is recommended to use the **atlas_type_mw** and **atlas_type_ir** constants defined in **rttov_emis_atlas_mod** to select MW or IR atlases respectively (see Annex K).

The **atlas** argument is used to hold the loaded atlas data. You can initialise data for multiple atlases, months, and instruments simultaneously: the only limitation is the available memory. The atlas argument is of derived type **rttov_emis_atlas_data** which is defined in the module **mod_emis_atlas_data**. You can declare as many instances of this type as you require and load the data from any atlas for any month and instrument into each one by calling this subroutine for each **atlas** variable.

The **atlas_id** argument is used to select between the available atlases: if omitted the default IR or MW atlas is selected. The valid IDs are defined in **rttov_emis_atlas_mod**:

IR instruments

- UWIRemis – uwiremis_atlas_id (1), default
- CAMEL single-year – camel_atlas_id (2)
- CAMEL climatology – camel_clim_atlas_id (3)

MW instruments

- TELSEM2 – telsem2_atlas_id (1), default
- CNRM MW atlas – cnrm_mw_atlas_id (2)

Once initialised the TELSEM2 atlas data can be used with any MW instrument: in this case the optional **coefs** argument is ignored. The CNRM MW atlas is always initialised for a specific instrument and the initialised **atlas** data can only be used with that instrument (AMSU-A, AMSU-B, MHS, SSMI/S or ATMS). In this case the **coefs** argument is mandatory and must contain the coefficients for a compatible sensor.

For the IR atlases the **coefs** argument is optional. If omitted the loaded atlas data can be used with any IR sensor. If the **coefs** argument is supplied the atlas data are loaded for use with that specific instrument. This makes it much faster to obtain emissivities from the atlas, but the loaded atlas data can only be used with that specific instrument. If you are only using the atlas for one IR instrument it is recommended to supply the **coefs** argument.

In addition, the IR atlases can optionally include a zenith angle correction which is activated when reading the data by setting **ir_atlas_ang_corr** to true. The IR atlases can also optionally return emissivity standard deviation data: this must be requested at setup by setting **ir_atlas_read_std** to true but note that this increases the atlas memory requirements significantly. Note that if you do not require the standard deviations, you do not need to download the standard deviation atlas data files.



Both v2 and v3 of the CAMEL single-year and climatology atlases are supported, and this is specified via the **camel_version** argument. Version 3 the default and is recommended. For the CAMEL v3 single-year atlas, the **year** argument must be specified giving the year of the required dataset.

Type	In/Out	Variable	Description
Integer	Intent(out)	err	Return code
Type(rttoV_options)	Intent(in)	opts	RTTOV options structure
Integer	Intent(in)	imonth	Month (1-12) of atlas data to be loaded.
Integer	Intent(in)	atlas_type	Specifies whether to load a MW or IR atlas: atlas_type_mw or atlas_type_ir .
Type(rttoV_emis_atlas_data) (from rttoV_emis_atlas_mod)	Intent(inout)	atlas	Structure to hold the loaded atlas data.
Integer	Intent(in), optional	atlas_id	ID of atlas to use (see above); defaults to 1 for both IR and MW.
Character	Intent(in), optional	path	Path of directory containing emissivity atlas data files. Defaults to the current directory.
Type(rttoV_coefs)	Intent(in), optional	coefs	RTTOV instrument coefficients structure. This is mandatory for the CNRM MW atlas and is ignored by TELSEM2. For the UW and CAMEL IR atlases, if this argument is passed, they will be initialised specifically for use with this instrument: the atlases are then faster to access.
Logical	Intent(in), optional	ir_atlas_read_std	If true, initialise IR atlas error dataset. If the errors (standard deviations) are not required, this should be set to false: the atlas lookups will be faster, and the atlas will require much less memory. Default is false. Not applicable to MW atlases.
Logical	Intent(in), optional	ir_atlas_ang_corr	If true, initialise IR atlas so that zenith angle correction will be applied. This requires the additional angular correction data files. Default is false. Not applicable to MW atlases.
Integer	Intent(in), optional	year	For the CNRM MW atlas: data are available for multiple years as indicated in the atlas filenames. This argument selects the year (default=2015). For the CAMEL v3 single-year atlas: data are available for the years 2000-2021. In this case there is no default, and the year must be specified. Not applicable to the other atlases.
Integer	Intent(in), optional	camel_version	This only applies to the CAMEL atlases and specifies the version. It must be either 2 or 3 (the default and recommended).

2. RTTOV_GET_EMIS interface

The **rttoV_get_emis** subroutine takes the **profiles(:)** array as input: Table 8.8 lists the profile variables used by each atlas. Although there is only one routine to access the IR and MW atlases, each atlas has a number of options which are unique to it. Therefore, the **rttoV_get_emis** subroutine will be described separately for the IR and MW atlases, discussing just those arguments relevant in each case. If an argument is supplied which is not applicable to the given atlas, a warning message is printed, and the argument is ignored. The output **emissivity(:)** array can be used as input to **rttoV_direct**, **rttoV_tl**, **rttoV_ad** and **rttoV_k** via **emis_refl%emis_in(:)** with the corresponding elements of **emis_refl%calc_emis(:)** set to false. However, it is very important to check the emissivities that are returned to ensure they are positive values: the atlases may return negative values if they do not have data for the given surface type or lat/lon location. You may wish to set **calc_emis(:)** to true where the returned emissivities are negative, for example.

The **isurf** argument is required in all cases. It specifies which surface index to use in the **profiles** structures when accessing surface properties. The same index is used in every member of the **profiles(:)** array. To obtain emissivities for multiple

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

surfaces you must call **rttov_get_emis** for each **isurf** index. For homogenous surfaces (i.e., **nsurfaces=1**) set **isurf=1**. See section 8.3.10 for more information about heterogenous surfaces.

In the discussions below, **nchanprof** is the size of the **chanprof(:)** array, **nprofiles** is the size of the **profiles(:)** array, and **nchan** is the largest number of channels computed for any given profile.

Returning IR emissivities (UWIRemis, CAMEL single-year, and CAMEL climatology atlases):

```
call rttov_get_emis (err, opts, chanprof, profiles, isurf, coefs, atlas,
                  emissivity, emis_std, emis_flag,
                  pc_eval, pc_evec, pc_const, max_distance, snow_correction)
```

If the zenith angle correction was selected when calling **rttov_setup_emis_atlas** then the **zenangle** and **sunzenangle** profile members are required: **sunzenangle** is used to determine whether the day or night bias correction should be applied so it only needs to be less than 85° for day or greater than 85° for night (the exact value is not important).

Emissivity values are returned for both land and sea-ice surface types. See section 8.3.6 for a discussion of the treatment of snow and the **snow_correction** argument.

The IR atlases can optionally return the emissivity standard deviations if the **emis_std** argument is passed. The CAMEL climatology atlas has a standard deviation dataset derived from the multi-year climatology which is also available for the CAMEL single-year atlas. Note that this must be selected when calling **rttov_setup_emis_atlas**.

Each surface point in the atlas has an associated flag which may be returned. You may wish to use this flag as a form of quality control (see the IR atlas documentation Borbas and Ruston, 2010, and the RTTOV v12 Science and Validation Report). Note that there is only one flag per profile (i.e., per location). However the **emis_flag(:)** output array is of size **nchanprof** to be consistent with the **emissivity(:)** array.

If the **coefs** argument was passed to **rttov_setup_emis_atlas** you must ensure the **coefs** argument here is compatible with the **atlas** argument.

For the UWIRemis and CAMEL single-year atlases you can optionally obtain the atlas PC scores and eigenvectors. To do this pass the **pc_eval**, **pc_evec** and **pc_const** arguments (all three must be present). The output of PCs from the IR atlases is intended for advanced users. Only the UWIRemis and CAMEL single-year atlases are supported because the extraction of PCs from the CAMEL climatology atlas is much more complicated. The emissivity for **chanprof(:)** index **i** and **profiles(:)** index **j** is reconstructed as follows:



$$emis(i) = SUM(pc_eval(1:numpcs,j) * pc_evec(1:numpcs,i)) + pc_const(i)$$

The first dimension of the **pc_eval** and **pc_evec** arrays is the maximum number of PCs used by the atlas: this is given by the **numpcs** parameter in **mod_uwiremis_atlas** (6 for UWIRemis) and **mod_camel_atlas** (9 for the CAMEL single-year atlas). For the CAMEL atlas, not all 9 PCs will be used in every case: the unused **pc_eval** and **pc_evec** values are always set to zero.

Note that **pc_eval** has dimensions (**numpcs,nprofiles**) and the PC scores are the same for all channels associated with a given profile.

NB The returned PC arrays will be zero if the profile **snow_fraction** is greater than zero or if the surface type is not land. Also note that occasionally the atlas obtains emissivities slightly greater than 1 and these are capped at 1. In this case the PC outputs are unmodified and so the emissivity reconstructed from the PC outputs will not match the capped emissivity returned by the atlas. If the emissivity is exactly 1 and the PC reconstructed emissivity is greater than 1 then this is the reason.

The **max_distance** argument can be specified to optionally search for a nearby valid emissivity (within this given maximum distance) if the atlas has no emissivity data at the original location. In order to be fast, this does not guarantee to return strictly the nearest valid value. This feature may be useful in coastal regions where you wish to specify a land surface emissivity (perhaps in the context of simulating a heterogenous surface), but the atlas does not have a value at the

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

given lat/lon location. The code enforces an upper limit of 100 km for **max_distance** to prevent excessive searches. This is not intended to be used to force the atlas to return a land emissivity for a location that is known not to be land.

Type	In/Out	Variable	Description
Integer	Intent(out)	err	Return code
Type(rttov_options)	Intent(in)	opts	RTTOV options structure
Type(rttov_chanprof)	Intent(in)	chanprof(:)	Chanprof structure.
Type(rttov_profile)	Intent(in)	profiles(:)	Profiles structure.
Integer	Intent(in)	isurf	Index of surface to use in profiles(:)
Type(rttov_coefs)	Intent(in)	coefs	RTTOV instrument coefficient structure.
Type(rttov_emis_atlas_data) (from rttov_emis_atlas_mod)	Intent(in)	atlas	Atlas data previously loaded via rttov_emis_atlas_setup .
Real	Intent(out)	emissivity(nchanprof)	Emissivity values.
Real	Intent(out), optional	emis_std(nchanprof)	Emissivity errors (standard deviations).
Integer	Intent(out), optional	emis_flag(nchanprof)	Emissivity atlas flags.
Real	Intent(out), optional	pc_eval(numpcs, nprofiles)	Atlas PC scores
Real	Intent(out), optional	pc_evec(numpcs, nchanprof)	Atlas PC eigenvectors
Real	Intent(out), optional	pc_const(nchanprof)	Constant values used in reconstructing emissivities from pc_eval and pc_evec .
Real	Intent(in), optional	max_distance	Maximum distance in km to search for nearest available emissivity if there is no value at the given location, default 0 (i.e., do not search).
Logical	Intent(in), optional	snow_correction	CAMEL climatology atlas only: if true then atlas attempts to find snow-free spectra when profile snow_fraction is zero, default true (recommended).

Returning MW emissivities (TELSEM2):

```
call rttov_get_emis (err, opts, chanprof, profiles, isurf, coefs, atlas,
                   emissivity, emis_std, emis_cov, resolution)
```

TELSEM2 consists of both an atlas and an interpolator which carries out interpolation of emissivity values in frequency and in space. The atlas has a nominal spatial resolution of 0.25°. If the **resolution** argument is supplied and is larger than 0.25, the emissivity values returned are integrated over the atlas grid according to the specified resolution. As with the IR atlases, the TELSEM MW atlas can optionally return estimated emissivity errors. It can also optionally return emissivity covariance matrices: this provides emissivity covariances among all channels for each profile.

Type	In/Out	Variable	Description
Integer	Intent(out)	err	Return code
Type(rttov_options)	Intent(in)	opts	RTTOV options structure
Type(rttov_chanprof)	Intent(in)	chanprof(:)	Chanprof structure.
Type(rttov_profile)	Intent(in)	profiles(:)	Profiles structure.
Integer	Intent(in)	isurf	Index of surface to use in profiles(:)
Type(rttov_coefs)	Intent(in)	coefs	RTTOV instrument coefficient structure.
Type(rttov_emis_atlas_data) (from rttov_emis_atlas_mod)	Intent(in)	atlas	Atlas data previously loaded via rttov_emis_atlas_setup .
Real	Intent(out)	emissivity(nchanprof)	Emissivity values.
Real	Intent(out), optional	emis_std(nchanprof)	Emissivity errors (standard deviations).

Real	Intent(out), optional	emis_cov(nprofiles, nchan, nchan)	Emissivity covariances.
Real	Intent(in), optional	resolution	Return emissivities at user-defined resolution. Units are degrees latitude/longitude. The default (i.e. nominal atlas) resolution is 0.25°.

Returning MW emissivities (CNRM MW atlas):

```
call rttov_get_emis (err, opts, chanprof, profiles, isurf, coefs, atlas,
                    emissivity)
```

The CNRM atlas has emissivity datasets **only** for specific instruments and so does not need to carry out interpolation in frequency to instrument channels. It does not provide an estimate of emissivity error. The **coefs** argument must be compatible with the **atlas** data.

Type	In/Out	Variable	Description
Integer	Intent(out)	err	Return code
Type(rttov_options)	Intent(in)	opts	RTTOV options structure
Type(rttov_chanprof)	Intent(in)	chanprof(:)	Chanprof structure.
Type(rttov_profile)	Intent(in)	profiles(:)	Profiles structure.
Integer	Intent(in)	isurf	Index of surface to use in profiles(:)
Type(rttov_coefs)	Intent(in)	coefs	RTTOV instrument coefficient structure.
Type(rttov_emis_atlas_data) (from rttov_emis_atlas_mod)	Intent(in)	atlas	Atlas data previously loaded via rttov_emis_atlas_setup .
Real	Intent(out)	emissivity(nchanprof)	Emissivity values.

3. RTTOV_DEALLOCATE_EMIS_ATLAS interface

```
call rttov_deallocate_emis_atlas (atlas)
```

rttov_deallocate_emis_atlas is called to de-allocate the memory for an emissivity atlas data structure. This should be called for every loaded **atlas** structure.

Type	In/Out	Variable	Description
Type(rttov_emis_atlas_data) (from rttov_emis_atlas_mod)	Intent(inout)	atlas	Atlas data structure to deallocate.

4. RTTOV_GET_SEA_EMIS interface

This subroutine is separate from the land surface emissivity atlas interface routines. It is used to obtain surface emissivities from the internal RTTOV sea surface models without running a full simulation.

```
call rttov_get_sea_emis (err, opts, chanprof, profiles, isurf, coefs, calc_emis,
                        emissivity)
```

The emissivity model is determined by the instrument loaded into the coefficients structure **coefs** (IR or MW), by the surface type of the profile(s), and by the selected IR or MW sea surface emissivity model in the options structure (**opts**). The **profiles(:)** array must be allocated as if for a full RTTOV simulation, but only the relevant members need to be populated. Section 8.3.1 describes the profile variables used by each emissivity model. To summarise:

ISEM - zenith angle
IREMIS - zenith angle, 10m wind u/v, T skin
SURFEM-Ocean - zenith and azimuth angles, 10m wind u/v, T skin, salinity
FASTEM-5/6 - zenith and azimuth angles, 10m wind u/v, T skin, salinity, foam_fraction (if **use_foam_fraction** option is true)

In addition, for MW sensors and land/sea-ice surfaces with **calc_emis(:)** true, the FASTEM land/sea-ice parameterisation is used (section 8.3.1), and the profile **fastem(1:5)** parameters must be specified.

In all cases the profile **surfactype** is required. For sea surface profiles, the selected sea surface emissivity model is called. For land or sea-ice profiles, the emissivities that RTTOV would use are returned for the corresponding channels as described in section 8.3.1. Emissivities are returned in the output **emissivity(:)** argument where the corresponding elements of **calc_emis(:)** are true. Values in the **emissivity(:)** array are untouched where the corresponding elements of **calc_emis(:)** are false.

The **isurf** argument specifies which surface index to use in the **profiles** structures when accessing the surface properties listed above. The same index is used in every member of the **profiles(:)** array. To obtain emissivities for multiple surfaces you must call **rttov_get_sea_emis** for each **isurf** index. For homogenous surfaces (i.e., **nsurfaces=1**) set **isurf=1**. See section 8.3.10 for more information about heterogenous surfaces.

Note that diffuse reflectances cannot be returned from the MW sea surface emissivity models because they require the total atmospheric transmittance which requires the full RTTOV optical depth calculation. In all other cases, the diffuse reflectance for thermal channels is (1-emissivity), so the diffuse reflectance is not returned by this routine.

Type	In/Out	Variable	Description
Integer	Intent(out)	err	Return code
Type(rttov_options)	Intent(in)	opts	RTTOV options structure
Type(rttov_chanprof)	Intent(in)	chanprof(:)	Chanprof structure.
Type(rttov_profile)	Intent(in)	profiles(:)	Profiles structure: only relevant members need to be populated (see above).
Integer	Intent(in)	isurf	Index of surface to use in profiles(:)
Type(rttov_coefs)	Intent(in)	coefs	RTTOV instrument coefficient structure.
Logical	Intent(in)	calc_emis(nchanprof)	Where true, corresponding elements of emissivity(:) are updated; where false, corresponding elements of emissivity(:) are untouched.
Real	Intent(inout)	emissivity(nchanprof)	Emissivity values.

Annex G – Surface reflectance subroutines

This section describes the interface to the atlas subroutines and a subroutine to obtain BRDFs from the internal RTTOV surface model. More information about using the atlas is provided in section 8.3.7. The atlas data are provided in netCDF files and as such RTTOV must be compiled with netCDF support (see section 5.3).

1. RTTOV_SETUP_BRDF_ATLAS interface

```
call rttov_setup_brdf_atlas (err, opts, imonth, atlas, atlas_id, path, coefs)
```

The atlas provides monthly climatology so you must provide the month (**imonth**, 1-12) for which to load atlas data. By default, the atlas files are assumed to be in the current directory, but you can optionally specify the **path** to the directory containing the atlas data files.

The **atlas** argument is used to hold the loaded atlas data. You can initialise data for multiple months and instruments simultaneously: the only limitation is the available memory. The atlas argument is of derived type **rttov_brdf_atlas_data** which is defined in the module **rttov_brdf_atlas_mod**.

The **atlas_id** argument is used to select between the available atlases: currently there is only one BRDF atlas (**atlas_id**=1) so this argument should be omitted.

The **coefs** argument is optional. If omitted the loaded atlas data can be used with any visible/near-IR sensor. If the **coefs** argument is supplied the atlas data are loaded for use with that specific instrument. This makes it much faster to obtain BRDFs from the atlas, but the loaded atlas data can only be used with that specific instrument. If you are only using the atlas for one visible/near-IR instrument it is recommended to supply the **coefs** argument.

Type	In/Out	Variable	Description
Integer	Intent(out)	err	Return code
Type(rttov_options)	Intent(in)	opts	RTTOV options structure
Integer	Intent(in)	imonth	Month (1-12) of atlas data to be loaded.
Type(rttov_brdf_atlas_data) (from rttov_brdf_atlas_mod)	Intent(inout)	atlas	Structure to hold the loaded atlas data.
Integer	Intent(in), optional	atlas_id	ID of atlas to use. Currently there is only one atlas, so this argument can be omitted.
Character	Intent(in), optional	path	Path of directory containing BRDF atlas data files. Defaults to the current directory.
Type(rttov_coefs)	Intent(in), optional	coefs	RTTOV instrument coefficients structure. If this argument is passed the atlas will be initialised specifically for use with this instrument: the atlas is then faster to access.

2. RTTOV_GET_BRDF interface

The **rttov_get_brdf** subroutine takes the **profiles(:)** array as input: the routine uses the **latitude**, **longitude**, **zenangle**, **azangle**, **sunzenangle**, **sunazangle**, **skin%surftype** and **skin%watertype** members of each profile. The output **brdf(:)** array can be used as input to **rttov_direct**, **rttov_tl**, **rttov_ad** and **rttov_k** via **emis_refl%brdf_in(:)** with the corresponding elements of **emis_refl%calc_brdf(:)** set to false. However, it is very important to check the BRDFs that are returned to ensure they are positive values: the atlas may return negative values if it does not have data for the given surface type or lat/lon location. You may wish to set **calc_brdf (:)** to true where the returned BRDFs are negative, for example.

The **isurf** argument specifies which surface index to use in the **profiles** structures when accessing surface properties. The same index is used in every member of the **profiles(:)** array. To obtain BRDFs for multiple surfaces you must call

rttov_get_brdf for each **isurf** index. For homogenous surfaces (i.e., **nsurfaces**=1) set **isurf**=1. See section 8.3.10 for more information about heterogenous surfaces.

In the discussions below, **nchanprof** is the size of the **chanprof(:)** array.

```
call rttov_get_brdf (err, opts, chanprof, profiles, isurf, coefs, atlas,
                  brdf, brdf_flag, bs_albedo, max_distance)
```

The BRDF atlas returns BRDF values for over land and also over sea, but it does not take sun glint into account (set **calc_brdf(:)** to true to use RTTOV's sea surface reflectance model instead). The BRDF atlas can optionally return the directional-hemispherical (black-sky) albedo. In addition, each surface point in the atlas has an associated flag which may be returned. You may wish to use this flag as a form of quality control (see the BRDF atlas documentation in the RTTOV v11 Science and Validation Report). Note that there is only one flag per profile (i.e., per location). However, the **brdf_flag(:)** output array is of size **nchanprof** to be consistent with the **brdf(:)** array. If the **coefs** argument was passed to **rttov_setup_brdf_atlas** you must ensure the **coefs** argument here is compatible with the **atlas** argument.

The **max_distance** argument works in the same way as for the IR emissivity atlases: see the description in Annex F.

Type	In/Out	Variable	Description
Integer	Intent(out)	err	Return code
Type(rttov_options)	Intent(in)	opts	RTTOV options structure
Type(rttov_chanprof)	Intent(in)	chanprof(:)	Chanprof structure.
Type(rttov_profile)	Intent(in)	profiles(:)	Profiles structure.
Integer	Intent(in)	isurf	Index of surface to use in profiles(:)
Type(rttov_coefs)	Intent(in)	coefs	RTTOV instrument coefficient structure.
Type(rttov_brdf_atlas_data) (from rttov_brdf_atlas_mod)	Intent(in)	atlas	Atlas data previously loaded via rttov_brdf_atlas_setup .
Real	Intent(out)	brdf (nchanprof)	BRDF values.
Integer	Intent(out), optional	brdf_flag(nchanprof)	BRDF atlas flags.
Real	Intent(out), optional	bs_albedo(nchanprof)	Directional-hemispherical (black-sky) albedo.
Real	Intent(in), optional	max_distance	Maximum distance in km to search for nearest available BRDF if there is no value at the given location, default 0 (i.e., do not search).

3. RTTOV_DEALLOCATE_BRDF_ATLAS interface

```
call rttov_deallocate_brdf_atlas (atlas)
```

rttov_deallocate_brdf_atlas is called to de-allocate the memory for a BRDF atlas data structure. This should be called for every loaded **atlas** structure.

Type	In/Out	Variable	Description
Type(rttov_brdf_atlas_data) (from rttov_brdf_atlas_mod)	Intent(inout)	atlas	Atlas data structure to deallocate.

4. RTTOV_GET_SEA_BRDF interface

This subroutine is separate from the land surface BRDF atlas interface routines. It is used to obtain surface reflectances from the internal RTTOV sea surface model without running a full simulation.

```
call rttov_get_sea_brdf (err, opts, chanprof, profiles, isurf, coefs, calc_brdf,
                        brdf, diffuse_refl, emissivity)
```

The coefficient file for a suitable UV/VIS/NIR sensor must have been read into the **coefs** structure. The **profiles(:)** array must be allocated as if for a full RTTOV simulation, but only the relevant members need to be populated: satellite and solar zenith and azimuth angles, 10m wind u/v, and wind fetch. The surface type is also required. For sea surface profiles, the sea surface BRDF model is called. For land or sea-ice profiles, the BRDFs that RTTOV would use are returned for the corresponding channels as described in section 8.3.2. Direct solar BRDFs are returned in the output **brdf(:)** argument where the corresponding elements of **calc_brdf(:)** are true. Values in the **brdf(:)** array are untouched where the corresponding elements of **calc_brdf(:)** are false. The same applies to the optional **diffuse_refl(:)** argument if supplied.

Section 8.3 describes the treatment of surface reflectance in detail. In particular, it describes what BRDFs and diffuse reflectances are returned for each surface type and for channels at different wavelengths when **calc_brdf(:)** is true (Table 8.7). As can be seen from Table 8.7, for certain cases the surface emissivity is required to compute the BRDF and/or the diffuse reflectance. In order to obtain the same reflectances as RTTOV for these cases, you must supply the **emissivity(:)** argument populated with emissivities, for example from the **rttov_get_sea_emis** subroutine (see Annex F).

Important note: there are small discrepancies between returned BRDFs from RTTOV and from this external **rttov_get_sea_brdf** subroutine when the **opts%rt_all%refraction** option is true and/or when the profile surface elevation is non-zero. This is because the internal RTTOV sea surface BRDF calculation uses the computed local zenith angles at the surface, and these differ slightly to the input zenith angles in cases where the surface elevation is non-zero and/or if atmospheric refraction is enabled. To obtain identical reflectances to the internal RTTOV values in these cases via this external subroutine would require a fully populated **profiles(:)** data structure to be passed in (as if for a full simulation), and it would require various additional subroutine calls resulting in a more expensive calculation. It was therefore decided not to replicate the full calculations in this external subroutine.

Type	In/Out	Variable	Description
Integer	Intent(out)	err	Return code
Type(rttov_options)	Intent(in)	opts	RTTOV options structure
Type(rttov_chanprof)	Intent(in)	chanprof(:)	Chanprof structure.
Type(rttov_profile)	Intent(in)	profiles(:)	Profiles structure: only relevant members need to be populated (see above).
Integer	Intent(in)	isurf	Index of surface to use in profiles(:)
Type(rttov_coefs)	Intent(in)	coefs	RTTOV instrument coefficient structure.
Logical	Intent(in)	calc_brdf(nchanprof)	Where true, corresponding elements of brdf(:) are updated; where false, corresponding elements of brdf(:) are untouched.
Real	Intent(inout)	brdf(nchanprof)	BRDF values.
Real	Intent(inout), optional	diffuse_brdf(nchanprof)	Diffuse reflectance values.
Real	Intent(in), optional	emissivity(nchanprof)	Emissivity values.

Annex H – core RTTOV subroutines

1. RTTOV_DIRECT interface

```
call rttov_direct (errorstatus, opts, coefs, chanprof, profiles, emis_refl,
                  transmission, radiance, radiance2,
                  aer_opt_param, hydro_opt_param, refl_cloud_top
                  reflectivity, emis_retrieval_terms, diag_output,
                  pccomp, channels_rec, traj, traj_dyn, traj_sta)
```

The **rttov_direct** subroutine runs the RTTOV direct model. The inputs and outputs are described in section 7 with additional information on specific types of simulations given in section 8.

The various files **src/test/example*_fwd.F90** provide examples of running **rttov_direct** for a variety of different types of simulation. These can be used as a template for your own code.

In the following table **nchanprof** is the size of the **chanprof(:)** array (i.e., the total number of radiances to compute), **nprofiles** is the size of the **profiles(:)** array (i.e., the number of profiles to process in each call to RTTOV), and **nsurfaces** is the number of surfaces associated with each profile. **nchannelsrec** is the number of reconstructed radiances required per profile for PC-RTTOV simulations.

The **rttov_parallel_direct** subroutine has the same arguments as **rttov_direct** *except* for the trajectory arguments (**traj**, **traj_dyn**, **traj_sta**) which cannot be used, plus an optional final argument **nthreads** to specify the number of threads.

There is a final optional argument to **rttov_direct** (**lbl_check**) which should be ignored.

Type	In/Out	Variable	Description
Integer	Intent(out)	errorstatus	Return code.
Type(rttov_options)	Intent(in)	opts	RTTOV options structure.
Type(rttov_coefs)	Intent(in)	coefs	RTTOV coefficients structure.
Type(rttov_chanprof)	Intent(in)	chanprof(nchanprof)	Chanprof structure array.
Type(rttov_profile)	Intent(in)	profiles(nprofiles)	Profiles structure array.
Type(rttov_emis_refl)	Intent(inout)	emis_refl(nsurfaces)	Input/output surface emissivity/reflectance structure array.
Type(rttov_transmission)	Intent(inout)	transmission	Output transmittances (0-1).
Type(rttov_radiance)	Intent(inout)	radiance	Output radiances (mW/cm ⁻¹ /sr/m ² , degK, and BRF/unitless).
Type(rttov_radiance2)	Intent(inout), optional	radiance2	Secondary output radiances (mW/cm ⁻¹ /sr/m ²).
Real	Intent(in), optional	refl_cloud_top(nchanprof)	Optional input specifying cloud top reflectance for simple cloud scheme.
Type(rttov_opt_param)	Intent(in), optional	aer_opt_param	Explicit aerosol optical property input profiles.
Type(rttov_opt_param)	Intent(in), optional	hydro_opt_param	Explicit hydrometeor optical property input profiles.
Type(rttov_reflectivity)	Intent(inout), optional	reflectivity	Outputs from the radar simulator.
Type(rttov_emis_retrieval_terms)	Intent(inout), optional	emis_retrieval_terms	Output structure containing data which can be used for dynamic emissivity retrievals.
Type(rttov_diagnostic_output)	Intent(inout), optional	diag_output	Output structure containing additional per-profile information.
Type(rttov_pccomp)	Intent(inout), optional	pccomp	Output structure for PC-RTTOV.
Integer	Intent(in), optional	channels_rec(nchannelsrec)	Channels for which to compute reconstructed radiances from PC-RTTOV.
Type(rttov_traj)	Intent(inout), optional	traj	Trajectory structure to hold internal calculation data.
Type(rttov_traj_dyn)	Intent(inout), optional	traj_dyn	Dynamic trajectory structure to hold internal calculation data.
Type(rttov_traj_sta)	Intent(inout), optional	traj_sta	Static trajectory structure to hold internal calculation data.

2. RTTOV_K interface

```
call rttov_k (errorstatus, opts, coefs, chanprof, profiles, profiles_k,
             emis_refl, emis_refl_k, transmission, transmission_k,
             radiance, radiance_k, radiance2, refl_cloud_top,
             aer_opt_param, aer_opt_param_k, hydro_opt_param,
             hydro_opt_param_k, reflectivity, reflectivity_k, diag_output,
             pccomp, pccomp_k, profiles_k_pc, profiles_k_rec, channels_rec,
             traj, traj_k, traj_dyn, traj_k_dyn, traj_sta)
```

The **rttov_k** subroutine runs the RTTOV Jacobian (or “K”) model. The arguments are similar to **rttov_direct** with additional K model arguments for various inputs/outputs. Section 7.9 provides more information on running the TL/AD/K models.

The file **src/test/example_k.F90** provides an example of running **rttov_k** for clear-sky simulations. This can be used as a template for your own code.

Remember that all K arguments should be initialised to zero before calling **rttov_k**. The exception is the array (or arrays) in which the input perturbations are specified in **radiance_k** (or **pccomp_k** for PC-RTTOV or **reflectivity_k** for the radar solver).

In the following table **nchanprof** is the size of the **chanprof(:)** array (i.e., the total number of radiances to compute), **nprofiles** is the size of the **profiles(:)** array (i.e., the number of profiles to process in each call to RTTOV), and **nsurfaces** is the number of surfaces associated with each profile. **npcscores** is the number of PC scores requested per profile, and **nchannelsrec** is the number of reconstructed radiances required per profile for PC-RTTOV simulations.

The **rttov_parallel_k** subroutine has the same arguments as **rttov_k** *except* for the trajectory arguments (**traj**, **traj_k**, **traj_dyn**, **traj_k_dyn**, **traj_sta**) which cannot be used, plus an optional final argument **nthreads** to specify the number of threads.

Type	In/Out	Variable	Description
Integer	Intent(out)	errorstatus	Return code.
Type(rttov_options)	Intent(in)	opts	RTTOV options structure.
Type(rttov_coefs)	Intent(in)	coefs	RTTOV coefficients structure.
Type(rttov_chanprof)	Intent(in)	chanprof(nchanprof)	Chanprof structure array.
Type(rttov_profile)	Intent(in)	profiles(nprofiles)	Profiles structure array.
Type(rttov_profile)	Intent(inout)	profiles_k(nchanprof)	Output Jacobians on profile variables.
Type(rttov_emis_refl)	Intent(inout)	emis_refl(nsurfaces)	Input/output surface emissivity/reflectance structure array.
Type(rttov_emis_refl)	Intent(inout)	emis_refl_k(nsurfaces)	Input/output surface emissivity/reflectance Jacobians.
Type(rttov_transmission)	Intent(inout)	transmission	Output transmittances (0-1).
Type(rttov_transmission)	Intent(inout)	transmission_k	Jacobian of transmittances.
Type(rttov_radiance)	Intent(inout)	radiance	Direct model output radiances (mW/cm ⁻¹ /sr/m ² , degK, BRF/unitless).
Type(rttov_radiance)	Intent(inout)	radiance_k	Input radiance increments.
Type(rttov_radiance2)	Intent(inout), optional	radiance2	Secondary output radiances (mW/cm ⁻¹ /sr/m ²).
Real	Intent(in), optional	refl_cloud_top(nchanprof)	Optional input specifying cloud top reflectance for simple cloud scheme.
Type(rttov_opt_param)	Intent(in), optional	aer_opt_param	Explicit aerosol optical property input profiles.
Type(rttov_opt_param)	Intent(inout), optional	aer_opt_param_k	Output Jacobian of aerosol optical properties. Optional even if aer_opt_param argument is present.
Type(rttov_opt_param)	Intent(in), optional	hydro_opt_param	Explicit hydrometeor optical property input profiles.
Type(rttov_opt_param)	Intent(inout), optional	hydro_opt_param_k	Output Jacobian of hydrometeor optical properties. Optional even if hydro_opt_param argument is present.
Type(rttov_reflectivity)	Intent(inout), optional	reflectivity	Outputs from the radar simulator.

Type(rtto_v_reflectivity)	Intent(inout), optional	reflectivity_k	Input reflectivity increments for the radar simulator.
Type(rtto_v_diagnostic_output)	Intent(inout), optional	diag_output	Output structure containing additional per-profile information.
Type(rtto_v_pccomp)	Intent(inout), optional	pccomp	Output structure for PC-RTTOV.
Type(rtto_v_pccomp)	Intent(inout), optional	pccomp_k	Input PC or reconstructed radiance increments for PC-RTTOV.
Type(rtto_v_profile)	Intent(inout), optional	profiles_k_pc(npcores* nprofiles)	Jacobians on Principal Component scores. Required if opts%pcrtto_v%rec_rad is .false.
Type(rtto_v_profile)	Intent(inout), optional	profiles_k_rec(nchannelsrec* nprofiles)	Jacobians on profile variables for reconstructed radiance channels. Required if opts%pcrtto_v%rec_rad is .true.
Integer	Intent(in), optional	channels_rec(nchannelsrec)	Channels for which to compute reconstructed radiances from PC-RTTOV.
Type(rtto_v_traj)	Intent(inout), optional	traj	Trajectory structure to hold internal calculation data.
Type(rtto_v_traj)	Intent(inout), optional	traj_k	Trajectory structure to hold internal calculation data.
Type(rtto_v_traj_dyn)	Intent(inout), optional	traj_dyn	Dynamic trajectory structure to hold internal calculation data.
Type(rtto_v_traj_dyn)	Intent(inout), optional	traj_k_dyn	Dynamic trajectory structure to hold internal calculation data.
Type(rtto_v_traj_sta)	Intent(inout), optional	traj_sta	Static trajectory structure to hold internal calculation data.

3. RTTOV_TL interface

```
call rttov_tl (errorstatus, opts, coefs, chanprof, profiles, profiles_tl,
emis_refl, emis_refl_tl, transmission, transmission_tl,
radiance, radiance_tl, radiance2, refl_cloud_top,
aer_opt_param, aer_opt_param_tl, hydro_opt_param,
hydro_opt_param_tl, reflectivity, reflectivity_tl, diag_output,
pccomp, pccomp_tl, channels_rec, traj, traj_tl, traj_dyn,
traj_tl_dyn, traj_sta)
```

The **rttov_tl** subroutine runs the RTTOV tangent linear model. The arguments are similar to **rttov_direct** with additional TL model arguments for various inputs/outputs. Section 7.9 provides more information on running the TL/AD/K models.

In the following table **nchanprof** is the size of the **chanprof(:)** array (i.e., the total number of radiances to compute), **nprofiles** is the size of the **profiles(:)** array (i.e., the number of profiles to process in each call to RTTOV), and **nsurfaces** is the number of surfaces associated with each profile. **nchannelsrec** is the number of reconstructed radiances required per profile for PC-RTTOV simulations.

The **rttov_parallel_tl** subroutine has the same arguments as **rttov_tl** *except* for the trajectory arguments (**traj**, **traj_k**, **traj_dyn**, **traj_k_dyn**, **traj_sta**) which cannot be used, plus an optional final argument **nthreads** to specify the number of threads.

Type	In/Out	Variable	Description
Integer	Intent(out)	errorstatus	Return code.
Type(rttov_options)	Intent(in)	opts	RTTOV options structure.
Type(rttov_coefs)	Intent(in)	coefs	RTTOV coefficients structure.
Type(rttov_chanprof)	Intent(in)	chanprof(nchanprof)	Chanprof structure array.
Type(rttov_profile)	Intent(in)	profiles(nprofiles)	Profiles structure array.
Type(rttov_profile)	Intent(inout)	profiles_tl(nprofiles)	Input profile variable perturbations.
Type(rttov_emis_refl)	Intent(inout)	emis_refl(nsurfaces)	Input/output surface emissivity/reflectance structure array.
Type(rttov_emis_refl)	Intent(inout)	emis_refl_tl(nsurfaces)	Input/output surface emissivity/reflectance perturbations.
Type(rttov_transmission)	Intent(inout)	transmission	Output transmittances (0-1).
Type(rttov_transmission)	Intent(inout)	transmission_tl	Output transmittance perturbations.
Type(rttov_radiance)	Intent(inout)	radiance	Direct model output radiances (mW/cm ¹ /sr/m ² , degK, BRF/unitless).
Type(rttov_radiance)	Intent(inout)	radiance_tl	Output radiance perturbations.
Type(rttov_radiance2)	Intent(inout), optional	radiance2	Secondary output radiances (mW/cm ¹ /sr/m ²).
Real	Intent(in), optional	refl_cloud_top(nchanprof)	Optional input specifying cloud top reflectance for simple cloud scheme.
Type(rttov_opt_param)	Intent(in), optional	aer_opt_param	Explicit aerosol optical property input profiles.
Type(rttov_opt_param)	Intent(in), optional	aer_opt_param_tl	Input aerosol optical property perturbations. Optional even if aer_opt_param argument is present.
Type(rttov_opt_param)	Intent(in), optional	hydro_opt_param	Explicit hydrometeor optical property input profiles.
Type(rttov_opt_param)	Intent(in), optional	hydro_opt_param_tl	Input hydrometeor optical property perturbations. Optional even if hydro_opt_param argument is present.
Type(rttov_reflectivity)	Intent(inout), optional	reflectivity	Outputs from the radar simulator.
Type(rttov_reflectivity)	Intent(inout), optional	reflectivity_tl	Output reflectivity perturbations for the radar simulator.
Type(rttov_diagnostic_output)	Intent(inout), optional	diag_output	Output structure containing additional per-profile information.
Type(rttov_pccomp)	Intent(inout), optional	pccomp	Structure to hold output PC scores and reconstructed radiances.

Type(rtov_pcomp)	Intent(inout), optional	pcomp_tl	Output PC or reconstructed radiance perturbations for PC-RTTOV.
Integer	Intent(in), optional	channels_rec(nchannelsrec)	Channels for which to compute reconstructed radiances from PC-RTTOV.
Type(rtov_traj)	Intent(inout), optional	traj	Trajectory structure to hold temporary data.
Type(rtov_traj)	Intent(inout), optional	traj_tl	Trajectory structure to hold temporary data.
Type(rtov_traj_dyn)	Intent(inout), optional	traj_dyn	Dynamic trajectory structure to hold internal calculation data.
Type(rtov_traj_dyn)	Intent(inout), optional	traj_tl_dyn	Dynamic trajectory structure to hold internal calculation data.
Type(rtov_traj_sta)	Intent(inout), optional	traj_sta	Static trajectory structure to hold internal calculation data.

4. RTTOV_AD interface

```
call rttov_ad (errorstatus, opts, coefs, chanprof, profiles, profiles_ad,
emis_refl, emis_refl_ad, transmission, transmission_ad,
radiance, radiance_ad, radiance2, refl_cloud_top,
aer_opt_param, aer_opt_param_ad, hydro_opt_param,
hydro_opt_param_ad, reflectivity, reflectivity_ad, diag_output,
pccomp, pccomp_ad, channels_rec, traj, traj_ad, traj_dyn,
traj_ad_dyn, traj_sta)
```

The **rttov_ad** subroutine runs the RTTOV adjoint model. The arguments are similar to **rttov_direct** with additional AD model arguments for various inputs/outputs. Section 7.9 provides more information on running the TL/AD/K models.

Remember that all AD arguments should be initialised to zero before calling **rttov_ad**. The exception is the array (or arrays) in which the input perturbations are specified in **radiance_ad** (or **pccomp_ad** for PC-RTTOV or **reflectivity_ad** for the radar solver).

In the following table **nchanprof** is the size of the **chanprof(:)** array (i.e., the total number of radiances to compute), **nprofiles** is the size of the **profiles(:)** array (i.e., the number of profiles to process in each call to RTTOV), and **nsurfaces** is the number of surfaces associated with each profile. **nchannelsrec** is the number of reconstructed radiances required per profile for PC-RTTOV simulations.

The **rttov_parallel_ad** subroutine has the same arguments as **rttov_ad** *except* for the trajectory arguments (**traj**, **traj_k**, **traj_dyn**, **traj_k_dyn**, **traj_sta**) which cannot be used, plus an optional final argument **nthreads** to specify the number of threads.

Type	In/Out	Variable	Description
Integer	Intent(out)	errorstatus	Return code.
Type(rttov_options)	Intent(in)	opts	RTTOV options structure.
Type(rttov_coefs)	Intent(in)	coefs	RTTOV coefficients structure.
Type(rttov_chanprof)	Intent(in)	chanprof(nchanprof)	Chanprof structure array.
Type(rttov_profile)	Intent(in)	profiles(nprofiles)	Profiles structure array.
Type(rttov_profile)	Intent(inout)	profiles_ad(nprofiles)	Output AD on profile variables.
Type(rttov_emis_refl)	Intent(inout)	emis_refl(nsurfaces)	Input/output surface emissivity/reflectance structure array.
Type(rttov_emis_refl)	Intent(inout)	emis_refl_ad(nsurfaces)	Input/output surface emissivity/reflectance AD.
Type(rttov_transmission)	Intent(inout)	transmission	Output transmittances (0-1).
Type(rttov_transmission)	Intent(inout)	transmission_ad	AD of transmittances.
Type(rttov_radiance)	Intent(inout)	radiance	Direct model output radiances (mW/cm ⁻¹ /sr/m ² , degK, BRF/unitless).
Type(rttov_radiance)	Intent(inout)	radiance_ad	Input radiance/BT increments.
Type(rttov_radiance2)	Intent(inout), optional	radiance2	Secondary output radiances (mW/cm ⁻¹ /sr/m ²).
Real	Intent(in), optional	refl_cloud_top(nchanprof)	Optional input specifying cloud top reflectance for simple cloud scheme.
Type(rttov_opt_param)	Intent(in), optional	aer_opt_param	Explicit aerosol optical property input profiles.
Type(rttov_opt_param)	Intent(inout), optional	aer_opt_param_ad	Output AD on aerosol optical properties. Optional even if aer_opt_param argument is present.
Type(rttov_opt_param)	Intent(in), optional	hydro_opt_param	Explicit hydrometeor optical property input profiles.
Type(rttov_opt_param)	Intent(inout), optional	hydro_opt_param_ad	Output AD on hydrometeor optical properties. Optional even if hydro_opt_param argument is present.
Type(rttov_reflectivity)	Intent(inout), optional	reflectivity	Outputs from the radar simulator.
Type(rttov_reflectivity)	Intent(inout), optional	reflectivity_ad	Input reflectivity increments for the radar simulator.
Type(rttov_diagnostic_output)	Intent(inout), optional	diag_output	Output structure containing additional per-profile information.

Type(rtov_pccomp)	Intent(inout), optional	pccomp	Structure to hold output PC scores and reconstructed radiances.
Type(rtov_pccomp)	Intent(inout), optional	pccomp_ad	Input PC or reconstructed radiance increments for PC-RTTOV.
Integer	Intent(in), optional	channels_rec(nchannelsrec)	Channels for which to compute reconstructed radiances from PC-RTTOV.
Type(rtov_traj)	Intent(inout), optional	traj	Trajectory structure to hold internal calculation data.
Type(rtov_traj)	Intent(inout), optional	traj_ad	Trajectory structure to hold internal calculation data.
Type(rtov_traj_dyn)	Intent(inout), optional	traj_dyn	Dynamic trajectory structure to hold internal calculation data.
Type(rtov_traj_dyn)	Intent(inout), optional	traj_ad_dyn	Dynamic trajectory structure to hold internal calculation data.
Type(rtov_traj_sta)	Intent(inout), optional	traj_sta	Static trajectory structure to hold internal calculation data.

Annex I – Utility routines and tools

1. RTTOV_USER_CHECK_OPTIONS interface

```
call rttov_user_check_options (err, opts, coefs, chanprof, strictly_illegal)
```

This subroutine checks that the input options are consistent with one another and with the supplied coefficient structure. It can also optionally check the **chanprof(:)** structure. The routine sets **err** to **errorstatus_fatal** and prints an error message if any inconsistencies are found. By default, it reports strictly illegal settings that would cause run-time failures and also harmless-but-dubious settings that will not cause failures but might indicate a potential misunderstanding on the part of the user. The **strictly_illegal** argument can be set to true to ignore the latter and report only strictly illegal settings. This subroutine is intended for debugging problems with simulations. Note that **options/coefs/chanprof** are always checked for illegal values internally by RTTOV in every simulation anyway. Section 7.7 gives more information on checking RTTOV inputs.

Type	In/Out	Variable	Description
Integer	Intent(out)	err	Return code.
Type(rttov_options)	Intent(in)	opts	Options structure.
Type(rttov_coefs)	Intent(in)	coefs	RTTOV coefficients structure.
Type(rttov_chanprof)	Intent(in), optional	chanprof(:)	RTTOV chanprof structure. Supply this argument to check that the values are valid.
Logical	Intent(in), optional	strictly_illegal	If true, only report issues that will cause a run-time failure. If false (default), also report dubious-but-harmless settings that may indicate a mistake in the configuration.

2. RTTOV_USER_CHECK_PROFILE interface

```
call rttov_user_check_profile (err, opts, coefs, prof, quality,  
do_mfasis_nn, silent_fail)
```

This subroutine checks a single profile for unphysical values. The routine sets **err** to **errorstatus_fatal** and prints an error message if any illegal values are found. Error messages can be suppressed by setting the **silent_fail** argument to true. This can be useful to avoid generating irrelevant output when screening out unphysical profiles before calling RTTOV if the reason for the error is not relevant. If using only the MFASIS-NN solar solver (section 8.4.2), then by setting the **do_mfasis_nn** argument to true, the routine will only check profile variables relevant to this solver.

This routine can also be used to check input profiles against the gas absorption optical depth regression training limits (section 7.4.3) and, for PC-RTTOV simulations with aerosol or cloud scattering, it can check the aerosol/cloud profiles against the relevant PC training limits (section 8.6). This is activated when the **opts%config%verbose** option is true (in which case messages are printed out when regression limits are exceeded – this is independent of the **silent_fail** argument) or when the **quality** argument is present. The latter is analogous to the **radiance%quality(:)** output (section 7.8.3), and the bits relevant to gas optical depth limits or PC aerosol/cloud limits will be set if they are exceeded. Where input aerosol/cloud concentrations are zero, no comparisons are made to the corresponding limits meaning those values are not flagged or modified in the simulations.

Note that the other quality flags that are set internally by RTTOV cannot be checked by this routine (section 7.8.3). For the gas optical depth checks, the comparisons against the optical depth coefficient regression limits are made using the values on the nearest coefficient pressure level below the input profile pressure level (i.e., the profile is not interpolated onto the coefficient levels). This means the flags might not be set in strictly the same cases as when carried out internally by RTTOV, but differences will occur for marginal cases which are not expected to result in significant increases in simulation error. RTTOV *always* carries out checks against regression limits internally in order to set the **radiance%quality(:)** output.

Section 7.7 gives more information on checking RTTOV inputs.

Type	In/Out	Variable	Description
Integer	Intent(out)	err	Return code.
Type(rtov_options)	Intent(in)	opts	Options structure.
Type(rtov_coefs)	Intent(in)	coefs	RTTOV coefficients structure.
Type(rtov_profile)	Intent(in)	prof	Profile structure to check.
Integer	Intent(out), optional	quality	Quality bit mask equivalent to an element of the radiance%quality output.
Logical	Intent(in), optional	do_mfasis_nn	If true, only check variables required by MFASIS-NN simulations (default: false).
Logical	Intent(in), optional	silent_fail	If true, no error messages are printed out when any variable exceeds its corresponding hard limit (warnings about regression limits are controlled by opts%config%verbose).

3. RTTOV_USER_CHECK_EMIS_REFL interface

This subroutine checks an array of **rtov_emis_refl** structures for unphysical values. The routine sets **err** to **errorstatus_fatal** and prints an error message if any illegal values are found. Checks are performed according to the options specified in the **opts** argument. Section 7.7 gives more information on checking RTTOV inputs.

Type	In/Out	Variable	Description
Integer	Intent(out)	err	Return code.
Type(rtov_options)	Intent(in)	opts	Options structure.
Type(rtov_coefs)	Intent(in)	coefs	RTTOV coefficients structure.
Type(rtov_chanprof)	Intent(in)	chanprof(:)	RTTOV chanprof structure.
Type(rtov_profile)	Intent(in)	profiles(:)	Array of profile structures.
Type(rtov_emis_refl)	Intent(in)	emis_refl(:)	Array of emissivity/reflectance structures to check.

4. RTTOV_USER_CHECK_OPT_PARAM interface

This subroutine checks structures containing explicit aerosol and/or hydrometeor optical properties (section 8.4.11) for unphysical values. The routine sets **err** to **errorstatus_fatal** and prints an error message if any illegal values are found. Checks are performed according to the options specified in the **opts** argument. Section 7.7 gives more information on checking RTTOV inputs.

Type	In/Out	Variable	Description
Integer	Intent(out)	err	Return code.
Type(rtov_options)	Intent(in)	opts	Options structure.
Type(rtov_coefs)	Intent(in)	coefs	RTTOV coefficients structure.
Type(rtov_opt_param)	Intent(in), optional	aer_opt_param	Aerosol optical property structure to check.
Type(rtov_opt_param)	Intent(in), optional	hydro_opt_param	Hydrometeor optical property structure to check.

5. RTTOV_PRINT_OPTS interface

call `rttov_print_opts (opts, lu, text)`

This subroutine prints out the contents of the options structure to the selected logical unit (or to **error_unit** if the **lu** argument is omitted, see Annex B).

Type	In/Out	Variable	Description
Type(<code>rttov_opts</code>)	Intent(in)	<code>opts</code>	Options structure.
Integer	Intent(in), optional	<code>lu</code>	Logical unit for output (defaults to <code>error_unit</code>).
Character	Intent(in), optional	<code>text</code>	Additional text to print out.

6. RTTOV_PRINT_INFO interface

call `rttov_print_info (coefs, lu, text, verbose)`

This subroutine prints out some information about RTTOV (the library version number, and largest integer and real values allowed). If a coefficient structure is supplied, information about this is also displayed. Output is printed to the supplied logical unit (or to **error_unit** if the **lu** argument is omitted, see Annex B).

Type	In/Out	Variable	Description
Type(<code>rttov_coefs</code>)	Intent(in), optional	<code>coefs</code>	Coefficients structure.
Integer	Intent(in), optional	<code>lu</code>	Logical unit for output (defaults to <code>error_unit</code>).
Character	Intent(in), optional	<code>text</code>	Additional text to print out.
Logical	Intent(in), optional	<code>verbose</code>	If true, print per-channel information about coefficients (default false). For hyperspectral sensors this can generate a lot of output.

7. RTTOV_PRINT_PROFILE interface

call `rttov_print_profile (profile, lu, text)`

This subroutine prints out the contents of a profile structure to the selected logical unit (or to **error_unit** if the **lu** argument is omitted, see Annex B).

Type	In/Out	Variable	Description
Type(<code>rttov_profile</code>)	Intent(in)	<code>profile</code>	Profile structure.
Integer	Intent(in), optional	<code>lu</code>	Logical unit for output (defaults to <code>error_unit</code>).
Character	Intent(in), optional	<code>text</code>	Additional text to print out.

8. RTTOV_PRINT_RADIANCE_QUALITY interface

call `rttov_print_radiance_quality (quality, lu, text)`

This subroutine prints out a human-readable interpretation of a single element of the **radiance%quality(:)** output (section 7.8.3). The output is written to the selected logical unit (or to **error_unit** if the **lu** argument is omitted, see Annex B).

Type	In/Out	Variable	Description
Integer	Intent(in)	<code>quality</code>	Radiance quality output.
Integer	Intent(in), optional	<code>lu</code>	Logical unit for output (defaults to <code>error_unit</code>).
Character	Intent(in), optional	<code>text</code>	Additional text to print out.

9. RTTOV_WMO2RTTOV_SAT_ID interface

```
call rttov_wmo2rttov_sat_id (err, file_name, wmo_id, platform_id, sat_id)
```

This subroutine maps WMO satellite IDs onto RTTOV platform and satellite ID couplets. It uses a data file, **data/wmo2rttov_sat_id.dat**, which contains the mappings. New IDs can be added to this file without requiring the code to be recompiled. If there is an error (such as the WMO ID not being found in the data file), the routine sets **err** to **errorstatus_fatal** and prints an error message.

Type	In/Out	Variable	Description
Integer	Intent(out)	err	Return code.
Character	Intent(in)	file_name	Full path to the data/wmo2rttov_sat_id.dat file.
Integer	Intent(in)	wmo_id	Input WMO satellite ID.
Integer	Intent(inout)	platform_id	Output RTTOV platform ID.
Integer	Intent(inout)	sat_id	Output RTTOV satellite ID.

10. RTTOV_CALC_GEO_SAT_ANGLES interface

```
call rttov_calc_geo_sat_angles (err, coefs, profiles, &
                                geo_sat_lon, geo_sat_lat, geo_sat_height)
```

This subroutine calculates and populates the geostationary satellite zenith and azimuth angles (**zenangle**, **azangle**) in an array of **rttov_profile** structures using the pre-populated profile **latitude** and **longitude** members. Simply populate the **profiles(:)** array (at least those required members) and call this subroutine before calling RTTOV. The longitude of the sub-satellite point must be specified, and optionally the satellite height (default 35800 km), and sub-satellite latitude (default 0 degrees, but note that it should be close to zero anyway for a GEO sensor) may also be specified. The azimuth angle calculation ignores any input satellite latitude and assumes it is zero.

The subroutine sets **err** to **errorstatus_fatal** and prints an error message if the latitudes do not lie in the range [-90,+90] degrees, or if a computed zenith angle is larger than RTTOV's allowed maximum because this would cause an error if passed into RTTOV. The allowed maximum depends on the supplied coefficients (section 3).

Type	In/Out	Variable	Description
Integer	Intent(out)	err	Return code.
Type(rttov_coefs)	Intent(in)	coefs	RTTOV coefficients structure.
Type(rttov_profile)	Intent(inout)	profiles(:)	RTTOV profiles structures populated with (at least) latitude and longitude .
Integer	Intent(in), optional	geo_sat_lon	Longitude of sub-satellite point (degrees).
Integer	Intent(in), optional	geo_sat_lat	Latitude of sub-satellite point (degrees), default 0. Only used for zenith angle calculation.
Integer	Intent(in), optional	geo_sat_height	Satellite altitude, default 35800 km.

11. RTTOV_CALC_SOLAR_ANGLES interface

```
call rttov_calc_solar_angles (err, profiles)
```

This subroutine calculates and populates the solar zenith and azimuth angles (**sunzenangle**, **sunazangle**) in an array of **rttov_profile** structures given the pre-populated **latitude**, **longitude**, **date(:)**, and **time(:)** members. Simply populate the **profiles(:)** array (at least those required members) and call this subroutine before calling RTTOV.

The subroutine sets **err** to **errorstatus_fatal** and prints an error message if the latitudes do not lie in the range [-90,+90] degrees. The date/time inputs are not currently validated. Note that solar zenith angles are returned for all valid inputs,

even those that exceed RTTOV's maximum solar zenith angle (section 8.1). Such solar zenith angles may be passed into RTTOV, but solar radiation will not be included in the simulation for those profiles.

Type	In/Out	Variable	Description
Integer	Intent(out)	err	Return code.
Type(rtov_profile)	Intent(inout)	profiles(nprofiles)	RTTOV profiles structures populated with (at least) latitude , longitude , date(:) , and time(:) .

12. RTTOV_SCALE_REF_GAS_PROF interface

```
call rtov_scale_ref_gas_prof (err, coefs, profiles,
                             o3_col_int, o3_col_int_du, o3_max_ppmv,
                             co2_col_int, co2_max_ppmv,
                             n2o_col_int, n2o_max_ppmv,
                             co_col_int, co_max_ppmv,
                             ch4_col_int, ch4_max_ppmv,
                             so2_col_int, so2_max_ppmv,
                             log_p_interp)
```

This subroutine is intended to make it easy to pass scaled copies of the RTTOV background trace gas profiles into RTTOV simulations. A typical use case would be in simulating an older instrument for which you do not have an explicit CO₂ profile, but for which the RTTOV background profile is inappropriate because the concentrations are contemporary (~400 ppmv). See section 7.4.2 for more information on trace gases in RTTOV.

First set up your simulation to specify all trace gases you wish to include, for example, by setting **opts%rt_all%co2_data** to true. Then allocate and populate the array of **rtov_profile** structures with all the profile data as usual except for the CO₂ profiles. Finally, before calling RTTOV, call **rtov_scale_ref_gas_prof** passing in the **profiles(:)** array and, for example, the maximum CO₂ concentration in ppmv. The subroutine populates the **co2(:)** member of each element of **profiles(:)** with the RTTOV background CO₂ profile scaled to have the maximum CO₂ value you specified. This subroutine may be used to populate one or more of the optional trace gas species in a single call, and the scale factors are determined either by providing maximum values in ppmv *over dry air*, or by providing column-integrated amounts in kg/m² (or, for ozone only, a column-integrated amount in Dobson units). The column-integration is calculated as the sum over all layers of the layer delta-pressure multiplied by the layer average gas concentration in kg/kg over moist air divided by the acceleration due to gravity. The subroutine allows any value of **gas_units** in the input profiles. The optical depth coefficient file must support the variable gas(es) being requested via the arguments.

Type	In/Out	Variable	Description
Integer	Intent(out)	err	Return code.
Type(rtov_coefs)	Intent(in)	coefs	RTTOV coefficients structure.
Type(rtov_profile)	Intent(inout)	profiles(:)	RTTOV profiles structures populated with all data except the gas profiles in question.
Real	Intent(in), optional	o3_col_int o3_col_int_du o3_max_ppmv	Specify one of these (column integrated amount kg/m ² , column integrated amount in Dobson units, or maximum amount in ppmv over dry air) to populate profiles with suitably scaled copies of the RTTOV background ozone profile.
Real	Intent(in), optional	co2_col_int co2_max_ppmv	Specify one of these (column integrated amount kg/m ² , or maximum amount in ppmv over dry air) to populate profiles with suitably scaled copies of the RTTOV background CO ₂ profile.
Real	Intent(in), optional	*_col_int *_max_ppmv	... similarly for the other trace gases ...
Logical	Intent(in), optional	log_p_interp	By default, profile interpolation is linear in pressure. If this argument is present and set to true, interpolation is linear in log(pressure).

13. RTTOV_AER_CLIM_PROF interface

```
call rttov_aer_clim_prof (p_half, t, q, gas_units, mm_aer,
                        latitude, elevation, scale_factor, aer_prof)
```

This subroutine is used to generate aerosol profiles representing ten different climatological compositions using the first 10 aerosol types in the RTTOV OPAC aerosol optical property files (i.e., excluding the volcanic ash and Asian dust types, see section 8.4.9). An example of calling this subroutine can be seen in **src/test/example_aer_file_fwd.F90**.

The inputs are the pressure half-levels, the temperature and water vapour profiles on the full-levels (section 7.4.1), and the latitude (degrees), and surface elevation (km). You can select the input gas units (section 7.4.2), the output aerosol units (section 8.4.9), and you can also specify a scale factor that is applied to the resulting aerosol profiles.

The output is an array containing ten sets of aerosol profiles for the RTTOV OPAC components on the full-levels. The output array provides data for all aerosol types in the OPAC aertable files (section 8.4.9), but only the first ten are non-zero. You can copy the relevant climatological composition directly into **profiles(:)%aerosols(:,j)** from **aer_prof(:,j)** where **j** is the climatological composition index (1-10). The ten climatological compositions are (in order):

- | | | | |
|---|----------------------|----|-------------------|
| 1 | Continental clean | 6 | Maritime clean |
| 2 | Continental average | 7 | Maritime polluted |
| 3 | Continental polluted | 8 | Maritime tropical |
| 4 | Urban | 9 | Arctic |
| 5 | Desert | 10 | Antarctic |

Type	In/Out	Variable	Description
Real	Intent(in)	p_half(nlevels)	Pressure half-levels (hPa).
Real	Intent(in)	t(nlayers)	Temperature on full-levels (K).
Real	Intent(in)	q(nlayers)	Water vapour concentration on full-levels (units: see gas_units argument).
Integer	Intent(in)	gas_units	Units for water vapour: 2=>ppmv over moist air; 1=>kg/kg over moist air; otherwise=>ppmv over dry air (see section 7.4.2, and Annex K for constants that can be used)
Logical	Intent(in)	mmr_aer	Output aerosol units: true => kg/kg, false => cm ⁻³
Real	Intent(in)	latitude	Latitude of profile (degrees).
Real	Intent(in)	elevation	Elevation of profile (km)
Real	Intent(in)	scale_factor	Output profiles are multiplied by this scale factor.
Real	Intent(out)	aer_prof(naer_opac, nlayers,10)	Output aerosol number densities on full-levels for each of the naer_opac=13 RTTOV OPAC aerosol types (section 8.4.9, only the first ten are non-zero), for the 10 compositions listed above (units determined by mmr_aer argument)

14. CREATE_AER_CLIM_PROF.EXE

This executable provides a way of calling the **rttov_aer_clim_prof** subroutine (see above) from the command line. It requires files in the current directory specifying the pressure half-level profile (**plevs.dat**), and the temperature and water vapour profiles (**prof.dat**). The units of water vapour are ppmv over moist air. Example input files can be found in the **data/** directory of the RTTOV distribution. The program prompts you for a latitude (degrees), an elevation (km), and a scale factor. The calculated aerosol profiles are multiplied by the scale factor.

The output is written to the file **prof_aerosol_clim.dat** in the current directory and consists of ten columns, one for each of the first ten OPAC aerosol types. Each column contains ten consecutive profiles for the different climatological compositions. See the description of the **rttov_aer_clim_prof** subroutine for more information. The output aerosol units are number densities (cm⁻³) so you must set the RTTOV **profiles(:)%mmr_aer** variable to false when passing these profiles into RTTOV.

15. RTTOV_EMISSIVITY_RETRIEVAL interface

After calling `rttov_direct` with the optional `emis_retrieval_terms` argument (section 8.4.12), this subroutine can be used to carry out the emissivity retrieval calculation given the corresponding observed brightness temperatures, and optionally also skin temperature retrievals given specified values for the emissivity. Negative values are returned for any channels where the retrieval fails.

```
call rttov_emissivity_retrieval(chanprof, coefs, emis_terms, obs_tb,
                             retrieved_emis, specified_emis, retrieved_tskin)
```

Type	In/Out	Variable	Description
Type(<code>rttov_chanprof</code>)	Intent(in)	<code>chanprof(nchanprof)</code>	RTTOV <code>chanprof</code> structure as used in call to <code>rttov_direct</code> .
Type(<code>rttov_coefs</code>)	Intent(in)	<code>coefs</code>	RTTOV coefficients structure.
Type(<code>rttov_emis_retrieval_terms</code>)	Intent(in)	<code>emis_terms</code>	Emissivity retrieval terms structure populated by a call to <code>rttov_direct</code> .
Real	Intent(in)	<code>obs_tb(nchanprof)</code>	Observed brightness temperatures corresponding to simulated brightness temperatures (K).
Real	Intent(out)	<code>retrieved_emis(nchanprof)</code>	Retrieved emissivities.
Real	Intent(in), optional	<code>specified_emis(nchanprof)</code>	Specified emissivity for skin temperature retrieval.
Real	Intent(out), optional	<code>retrieved_tskin(nchanprof)</code>	Retrieved skin temperatures (K).

16. RTTOV_GET_PC_PREDICTINDEX interface

```
call rttov_get_pc_predictindex (err, opts, coefs, predictindex)
```

This subroutine may be used to obtain the indices for the specified set of PC-RTTOV regression channels (section 8.6), which depends on the setting of `opts%pcrttov%pc_band` and `opts%pcrttov%pc_reg_set`. These options must be set with values appropriate to the sensor being simulated, and the PC coefficients must have been read in (using the `rttov_read_coefs` subroutine, Annex C) before calling this subroutine. The example program `example_pc_fwd.F90` demonstrates the use of the `rttov_get_pc_predict_index` subroutine.

Type	In/Out	Variable	Description
Integer	Intent(out)	<code>err</code>	Return code.
Type(<code>rttov_options</code>)	Intent(in)	<code>opts</code>	RTTOV options structure
Type(<code>rttov_coefs</code>)	Intent(in)	<code>coefs</code>	RTTOV coefficients structure, pre-populated with optical depth and PC-RTTOV coefficients.
Integer, Pointer	Intent(out)	<code>predictindex(:)</code>	The output channel list. This should be an unallocated pointer on input. It is allocated with the appropriate size by the routine.

The executable `rttov_test_get_pc_predictindex` can be used to run the above subroutine on the command line:

```
$ rttov_test_get_pc_predictindex.exe \
  --coef ... --pccoef ... --pc_band ... --pc_reg_set ...
```

where the arguments are the optical depth coefficient file name, the corresponding PC coefficient file name, and the required PC band and regression set respectively.

17. RTTOV_OBS_TO_PC.EXE

The program `rttov_obs_to_pc.exe` (located in the `bin/` directory, source code is in `src/other/rttov_obs_to_pc.F90`) demonstrates how to convert radiance observations into PC-space which is necessary, for example, in applications involving the assimilation of PCs.

The usage of the example executable is as follows:

```
$ rttov_obs_to_pc.exe \
  --rtcoef_file ... \
  --pccoef_file ... \
  --obs_file ... \
  --pc_band ... \
  --pc_reg_set ... \
  --npcscores ... \
  --obs_bt
```

This program reads in a set of observations (radiances or BTs) from the specified “`obs_file`” and calculates the required number of PC scores based on the specified input optical depth and PC-RTTOV coefficient files and the specified regression band and predictor sets (see section 8.6 for details on PC-RTTOV). PC scores are written to standard out.

You would typically need to modify the source code to suit your own application.

Argument	Description
<code>--rtcoef_file</code>	Optical depth coefficient file.
<code>--pccoef_file</code>	PC coefficient file.
<code>--obs_file</code>	ASCII file containing white-space-separated observation values. Units are Kelvin if <code>--obs_bt</code> is present, otherwise <code>mW/m-2/sr-1/cm-1</code> .
<code>--pc_band</code>	PC band to use (usually 1).
<code>--pc_reg_set</code>	PC regression predictor set to use.
<code>--npcscores</code>	The number of PC scores to calculate.
<code>--obs_bt</code>	Optional: if present, input observations are BTs, otherwise they are radiances (see <code>--obs_file</code>).

18. RTTOV_ZUTILITY_MOD module

The module `src/other/rttov_zutility_mod.F90` may be used to obtain values for the magnetic field strength and orientation for use with the Zeeman coefficient files. The subroutines described here are not thread-safe due to the use of module variables to store the B-field data.

The look-up table (LUT) must first be loaded with the following function:

```
errorstatus = load_bfield_lut(filename_LUT)
```

One LUT is supplied in `data/Be_LUT.2007.txt`. The LUT is stored in the `rttov_zutility` module in the array `BField`. To return the field strength and orientation there are three options:

Return B-field components and total strength given latitude and longitude:

```
call compute_bfield(latitude, longitude,
                   Bx, By, Bz, Be)
```

Return B-field total strength and B-field orientation given latitude and longitude, and sensor zenith and azimuth angles:

```
call compute_bfield(latitude, longitude, sensor_zenang, sensor_aziang,
                   Be, cos_bkang, cos_baziang)
```

Return B-field total strength and B-field orientation given latitude and longitude, sensor zenith angle, sun-sensor relative azimuth angle, and date/time of observation:

```
call compute_bfield(latitude, longitude, sensor_zenang, sensor_relative_aziang,
                    Julian_day, utc_time, Be, cos_bkang, cos_baziang)
```

The arguments to these routines are detailed in the table below.

Type	In/Out	Variable	Description
Real	Intent(in)	latitude	Latitude of location (-90 to +90)
Real	Intent(in)	longitude	Longitude of location (accepts 0 to 360 or -180 to 180)
Real	Intent(in)	sensor_zenang	Sensor zenith angle
Real	Intent(in)	sensor_aziang	Sensor azimuth angle (0 to 360, North=0, positive clockwise)
Real	Intent(in)	sensor_relative_aziang	Solar azimuth angle minus sensor azimuth angle
Integer	Intent(in)	Julian_day	Julian day 1=Jan 1, 365=Dec 31 (366 leap year)
Real	Intent(in)	utc_time	Universal time 0.00-23.999 (GMT, Z time)
Real	Intent(out)	Bx, By, Bz	Magnetic field components: east, north and zenith (positive upwards) respectively. Units: Gauss.
Real	Intent(out)	Be	Magnetic field strength. Units: Gauss.
Real	Intent(out)	cos_bkang	Cosine of the angle between the magnetic field Be vector and the wave propagation direction k.
Real	Intent(out)	cos_baziang	Cosine of the azimuth angle of the Be vector in the (v, h, k) coordinates system, where v, h and k comprise a right-hand orthogonal system, similar to the (x, y, z) Cartesian coordinates. The h vector is normal to the plane containing the k and z vectors, where k points to the wave propagation direction and z points to the zenith. $h = (z \text{ cross } k) / z \text{ cross } k $. The azimuth angle is the angle on the (v, h) plane from the positive v axis to the projected line of the Be vector on this plane, positive counter-clockwise.

Finally, a subroutine is available to return the field orientation:

```
call compute_kb_angles(Bx, By, Bz, sensor_zenang, sensor_aziang,
                        cos_bkang, cos_baziang)
```

For this subroutine, the B-field components and sensor angles are inputs, and the B-field angles are outputs.

19. Scattering optical property generation tools

RTTOV provides some tools for generating custom aertables and hydrotables for use in scattering simulations.

The **rttov_make_aertable.exe** executable can be used to generate custom aertables based on spherical particles using Mie theory. This is described in **docs/readme_rttov_make_aertable.txt**.

There is a tool to generate custom hydrotables for MW sensors with many options available to configure the optical properties. This is described in **src/mw_scatt_coef/readme.txt**.

Currently no tool exists to generate custom hydrotables for UV/VIS/IR sensors.

Annex J – RTTOV derived types

RTTOV's derived types are defined in the module `rttov_types.F90`. The derived types required in your code that calls RTTOV are described here.

1. Options structure

The `rttov_options` structure holds switches and values that configure various aspects of RTTOV. The first step in running RTTOV is to declare an instance of this structure and to set the members to appropriate values (section 7.1). The options are held in several sub-types which group them by function. Options marked in grey should not usually be modified: they are implemented for testing purposes or to provide scope for extending RTTOV's capabilities in the future.

For integer options selecting between different parameterisations, see the relevant section of the user guide for an explanation the options and Annex K for the associated integer constants defined in `rttov_const`.

Type	Variable	Description
General configuration options: <code>opts % config</code>		
Logical	<code>verbose</code>	If false only messages for fatal errors are output (default = false). Section 7.4.3.
Logical	<code>apply_reg_limits</code>	If true, profile values outside the limits of the training data for the gas optical depth regression are clipped to the limits before input to the regression (default = false). Section 7.4.3.
Logical	<code>check_profiles</code>	If true, input <code>profiles(:)</code> are checked for unphysical values and whether they lie within the regression limits. The <code>emis_refl(:)</code> and (if relevant) the <code>aer/hydro_opt_param</code> inputs are also checked for unphysical values (default = true). Section 7.7.
Logical	<code>transmittances_only</code>	Direct model only: if true only transmittances are calculated. Output radiances, and surface emissivities and reflectances are zero. This is more efficient if only transmittances are required (default = false). Section 7.8.1.
Logical	<code>bt_overcast_calc</code>	If true, compute overcast brightness temperatures in <code>radiance%bt_overcast</code> output array (default = false). Section 7.8.2.
Logical	<code>gas_opdep_calc</code>	If false, disables the gas optical depth calculation and simulations are run assuming zero optical depths (or very near-zero depending on the solver). Intended for investigating certain scattering scenarios (default = true).
Logical	<code>adk_bt</code>	Determines input increment for AD/K routines for thermal channels (wavelengths $\geq 3\mu\text{m}$): if true increments are in BT, otherwise they are in radiance (default = true). Section 7.9.
Logical	<code>adk_refl</code>	Determines input increment for AD/K routines for non-thermal channels (wavelengths $< 3\mu\text{m}$): if true increments are in reflectance, otherwise they are in radiance (default = true). Section 7.9.
Logical	<code>opdep13_gas_clip</code>	If true, negative layer optical depths for individual gases are reset to zero in the v13 predictor calculations. This can cause convergence problems in some DA systems in which case this can be set to false (default = true).

Interpolation options: <code>opts % interpolation - section 7.4.1</code>		
Logical	<code>enable_interp</code>	If true, input profiles may be input on arbitrary pressure levels, and they are interpolated by RTTOV for the gas optical depth regression. Should always be true (default = true).
Integer	<code>interp_mode</code>	Select interpolation mode, only applies if <code>enable_interp</code> is true (default = <code>interp_rochon_wfn</code>).
Logical	<code>pressure_gradients</code>	If true, the profile <code>p_half</code> and <code>p</code> members are active variables in the TL/AD/K, only applies if <code>enable_interp</code> is true (default = false). Section 7.9.

General radiative transfer options: <code>opts % rt_all</code>		
Logical	<code>o3_data</code>	Set to true if supplying O ₃ profiles to RTTOV (default = false). Coefficient file must support variable O ₃ . Section 7.4.2.
Logical	<code>co2_data</code>	Set to true if supplying CO ₂ profiles to RTTOV (default = false). Coefficient file must support variable CO ₂ . Section 7.4.2.
Logical	<code>n2o_data</code>	Set to true if supplying N ₂ O profiles to RTTOV (default = false). Coefficient file must support variable N ₂ O. Section 7.4.2.
Logical	<code>co_data</code>	Set to true if supplying CO profiles to RTTOV (default = false). Coefficient file must support variable CO. Section 7.4.2.

Logical	ch4_data	Set to true if supplying CH ₄ profiles to RTTOV (default = false). Coefficient file must support variable CH ₄ . Section 7.4.2.
Logical	so2_data	Set to true if supplying SO ₂ profiles to RTTOV (default = false). Coefficient file must support variable SO ₂ . Section 7.4.2.
Logical	solar	Enable solar radiation in simulations (default = false). Section 8.1.
Real	rayleigh_max_wavelength	Specify maximum channel wavelength for which to enable Rayleigh scattering (units = μm, default = 2 μm). Set to zero to disable Rayleigh extinction and scattering entirely with v13 predictor coefficients. Section 8.1.3.
Real	rayleigh_min_pressure	Rayleigh scattering is ignored in layers at pressures below this (units = hPa, default = 0 hPa). Section 8.1.3.
Logical	rayleigh_single_scatt	Enable Rayleigh single scattering parameterisation. If false, Rayleigh extinction is still included in the simulations. The <i>rayleigh_multi_scatt</i> option overrides this for DOM simulations (default = true). Section 8.1.3.
Logical	nlte_correction	Enable the NLTE bias correction (default = false). Coefficient file must support the NLTE correction. Section 8.5.
Logical	refraction	Enable atmospheric refraction in simulations (default = true). This is ignored for solvers that assume strict plane parallel geometry or if the plane_parallel option is true. Section 7.4.1.
Logical	plane_parallel	Enable strict plane parallel geometry (no Earth curvature). It is not necessary to set this for solvers that assume plane parallel geometry (default = false). Section 7.4.1.
Logical	rad_down_lin_tau	Use linear-in-tau (true) or layer-average (false) for downwelling layer emission in radiative transfer equation integration (default = true). The latter is slightly faster and has negligible impact on TOA radiances. Applies to clear-sky calculations, and to the Chou-scaling thermal scattering solver. Lambertian downwelling radiances always use the layer-average.
Logical	use_t2m	Enable use of the 2m temperature profile variable (default = true). Section 7.4.1.
Logical	use_q2m	Enable use of the 2m water vapour profile variable (default = true). Section 7.4.1.

Surface-related options: *opts % surface* - section 8.3

Integer	solar_sea_refl_model	Select solar reflectance model for ocean surfaces (default = solar_refl_model_elfouhaily). There is currently only one option.
Integer	ir_sea_emis_model	Select IR emissivity model for ocean surfaces (default = ir_emis_model_iremis).
Integer	mw_sea_emis_model	Select MW emissivity model for ocean surfaces (default = mw_emis_model_surfem_ocean).
Logical	use_foam_fraction	Enable use of user-supplied profiles(:)%skin(:)%foam_fraction in emissivity models, where supported (default = false).
Logical	lambertian	Enable Lambertian surface option which allows surfaces to be treated as a configurable mixture of Lambertian and specular reflectors (default = false).
Logical	lambertian_fixed_angle	Use fixed effective zenith angle (true) or parameterised angle (false) when computing Lambertian downwelling radiances (default = true).
Logical	use_tskin_eff	If true, use per-channel effective skin temperatures input via emis_refl(:)%tskin_eff(:) instead of profiles(:)%skin(:)%t .

Cloud liquid water absorption (non-scattering) options: *opts % clw_absorption* - section 7.4.4

Logical	clw_data	Activate cloud liquid water absorption (non-scattering). Not compatible with the opts%scatt%hydrometeors option (default = false).
Integer	permittivity_param	Select liquid water permittivity parameterisation (default = clw_perm_rosenkranz).
Real	clw_cloud_top	Lower pressure limit for CLW absorption calculations, i.e., CLW is ignored in layers at pressures lower than this (units = hPa, default = 322 hPa).

Scattering options: *opts % scatt* - section 8.4

Logical	hydrometeors	Enable scattering by hydrometeors (default = false).
Logical	aerosols	Enable scattering by aerosols (default = false).
Integer	thermal_solver	Select solver for thermal radiation (default = thermal_solver_delta_edd).
Integer	solar_solver	Select solver for solar radiation (default = solar_solver_dom).
Logical	radar	Enable radar reflectivity simulations alongside passive radiance simulations (default = false).
Logical	user_hydro_opt_param	Enable explicit input of hydrometeor optical property profiles per channel (default = false).
Logical	user_aer_opt_param	Enable explicit input of aerosol optical property profiles per channel (default = false).
Integer	baran_ice_version	Select version of Baran ice scheme (default = baran2018).

Logical	rayleigh_multi_scatt	Enable Rayleigh multiple scattering (default = false). Currently only applicable with solar_solver=solar_solver_dom .
Integer	dom_nstreams	Select number of streams for simulations using the DOM solver (default = 8). Minimum value 2.
Real	dom_accuracy	Convergence criterion for termination of DOM azimuthal loop for solar DOM simulations (default = 0 i.e., no early termination).
Real	dom_opdep_threshold	DOM simulations ignore layers at level-to-space absorption optical depths larger than this, not applied if ≤ 0 (default = 0).
Logical	chou_tang_mod	Enable Tang modification to Chou-scaling (default = false).
Real	chou_tang_factor	Empirical factor used in Tang modification to Chou-scaling (default = 0.05)
Integer	mw_pol_mode	Select treatment of polarised scattering (default = mw_pol_mode_empirical). Only applies to MW sensors.
Real	ice_polarisation	Polarised scattering factor for ice hydrometeors for empirical polarisation mw_pol_mode (default = 1.4). Only applies to MW sensors.
Logical	zero_hydro_tlad	Enable hydrometeor TL/AD/K sensitivity in layers with zero hydrometeor concentration (default = false). Currently only applies to MW sensors and two-column overlap schemes with the delta-Eddington thermal solver.



Cloud overlap options: <i>opts % cloud_overlap</i> - section 8.4.3		
Integer	overlap_param	Select cloud overlap parameterisation (default = cloud_overlap_auto_select).
Logical	per_hydro_frac	Enable input of separate hydro_frac for each hydrometeor. When false, a single hydro_frac applies per layer with all hydrometeors well mixed in that fraction (default = false). Can only be true for radar simulations, and for the cloud_overlap_2col_weighted overlap scheme.
Real	col_threshold	Cloud columns with weights (multi-column overlap) or effective cloud fractions (two-column overlap) smaller than this value are treated as clear (default = 0 i.e., retain all columns). Value must be in range 0-1.
Real	two_col_max_frac_max_p	For cloud_overlap_2col_max_frac , this is the maximum pressure applied when determining the effective cloud fraction (units = hPa, default = 750 hPa).
Logical	hydro_frac_tlad	Switch to enable/disable hydrometeor TL/AD/K sensitivity to hydro_frac . NB setting false breaks the consistency between the direct and TL/AD/K models (default = true). If true, ignored unless the cloud_overlap_2col_weighted overlap scheme is used.

PC-RTTOV options: <i>opts % pcrttov</i> - section 8.6		
Logical	enable_pcrttov	Enable PC-RTTOV simulations (default = false).
Integer	npcscores	Number of PC scores to simulate (default = -1). The maximum number depends on the instrument being simulated. Minimum value is 1.
Logical	rec_rad	If false output only PC scores, if true compute reconstructed radiances and BTs (default = false).
Integer	pc_band	Select PC spectral band (default = 1). Valid values depend on the PC coefficient file. For current coefficients this should always be 1.
Integer	pc_reg_set	Select PC regression channel set (default = -1). Valid values depend on the PC coefficient file and vary between instruments. Minimum value is 1.

2. Chanprof structure

The **rttov_chanprof** structure is used to specify the channel and profile indices for each call to RTTOV. An array should be declared of size equal to the total number of radiances to be computed (across all channels and profiles). Each element of the **chanprof(:)** array provides a channel index and a profile index. The array should be ordered so that all channels for the first profile are listed, followed by all channels for the second profile, and so on (section 7.5).

Type	Variable	Description
Integer	chan	Channel index.
Integer	prof	Profile index.

		RTTOV v14 Users' Guide	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	-------------------------------	---

3. Profile structure



The **rttov_profile** structure is composed of atmospheric variables and general profile data, and member arrays of two sub-structures for near-surface and surface skin variables. You should declare an array of type **rttov_profile** of size **nprofiles**, the number of profiles being passed into RTTOV per call.

For integer members selecting among options, see Annex K for the associated integer constants defined in **rttov_const**.

Type	Variable	Description
Surface skin – type <i>rttov_skin</i>		
Integer	surftype	Surface type, valid values: surftype_land , surftype_sea , surftype_seaice
Integer	watertype	Water type for surftype_sea , valid values: watertype_fresh_water , watertype_ocean_water . Used by sea surface solar BRDF model and BRDF atlas.
Real	t	Radiative skin temperature (K). Not used if opts%surface%use_tskin_eff is true.
Real	salinity	Practical salinity unit ‰. Used by MW sea surface emissivity models.
Real	fastem(1:5)	Land/sea-ice surface parameters for the FASTEM land/sea-ice parameterisation.
Real	foam_fraction	Foam fraction (0-1) to use in FASTEM-5/6 if opts%surface%use_foam_fraction is true. By default, FASTEM-5/6 calculate the foam fraction internally.
Real	snow_fraction	Surface snow coverage fraction (0-1). Used by IR emissivity atlas.

Near-surface– type <i>rttov_near_surface</i>		
Real	t2m	Temperature (K). Used if opts%rt_all%use_t2m is true.
Real	q2m	Water vapour (units as per <i>gas_units</i>). Used if opts%rt_all%use_q2m is true.
Real	wind_u10m	10m wind U component (m/s). Used by sea surface emissivity models (except ISEM) and sea surface solar BRDF model.
Real	wind_v10m	10m wind V component (m/s). Used by sea surface emissivity models (except ISEM) and sea surface solar BRDF model.
Real	wind_fetch	Wind fetch (m) (length of water over which the wind has blown, typical value 100000m for open ocean). Used by sea surface solar BRDF model.

Atmospheric profile – type <i>rttov_profile</i>		
Integer	nlevels	Number of pressure half-levels, populated by allocation routine.
Integer	nlayers	Number of pressure full-levels/atmospheric layers (i.e. nlevels-1), populated by allocation routine.
Integer	nsurfaces	Number of surfaces associated with profile, populated by allocation routine.
Integer	gas_units	Units for gas concentrations, must be the same for all profiles, valid values: gas_unit_ppmv (ppmv over moist air), gas_unit_kg_per_kg (kg/kg over moist air, default), gas_unit_ppmvdry (ppmv over dry air).
Real	p_half(nlevels)	Pressure half-levels (hPa)
Real	p(nlayers)	Pressure full-levels (hPa). This is optional. If all zeros RTTOV computes internally as the mean of adjacent half-level pressures.
Real	t(nlayers)	Temperature on full-levels (K)
Real	q(nlayers)	Water vapour concentration on full-levels (units as per <i>gas_units</i>)
Real	o3(nlayers)	O ₃ concentration on full-levels (units as per <i>gas_units</i>). Used if o3_data true and coefficients support variable O ₃ .
Real	co2(nlayers)	CO ₂ concentration on full-levels (units as per <i>gas_units</i>). Used if co2_data true and coefficients support variable CO ₂ .
Real	n2o(nlayers)	N ₂ O concentration on full-levels (units as per <i>gas_units</i>). Used if n2o_data true and coefficients support variable N ₂ O.
Real	co(nlayers)	CO concentration on full-levels (units as per <i>gas_units</i>). Used if co_data true and coefficients support variable CO.
Real	ch4(nlayers)	CH ₄ concentration on full-levels (units as per <i>gas_units</i>). Used if ch4_data true and coefficients support variable CH ₄ .
Real	so2(nlayers)	SO ₂ concentration on full-levels (units as per <i>gas_units</i>). Used if so2_data true and coefficients support variable SO ₂ .
Real	clw(nlayers)	Cloud liquid water concentration on full-levels (kg/kg). Used if clw_data true for MW only. Treats cloud as absorbing medium only; not compatible with scattering simulations.
Logical	mmr_aer	Units for aerosols: (must be the same for all profiles) True => kg/kg (default), False => cm ⁻³

		<h1 style="text-align: center;">RTTOV v14 Users' Guide</h1>	Doc ID : NWPSAF-MO-UD-055 Version : 1.0.3 Date : 29/01/2025
--	--	---	---

Logical	mmr_hydro	Units for hydrometeors: (must be the same for all profiles) True => kg/kg (default), False => g/m ³
Integer	flux_conversion(nhydro)	Optionally use units of flux for rain (flux_conv_rain) or snow (flux_conv_snow) hydrometeor types. Default is flux_conv_none meaning units are determined by mmr_hydro . The flux conversion is deprecated.
Real	aerosols(naer, nlayers)	Grid box average aerosol concentrations on full-levels (units as per mmr_aer), naer is the number of aerosol types defined in the aertable file. Not used if opts%scatt%user_aer_opt_param is true.
Real	hydro(nhydro, nlayers)	Grid box average hydrometeor concentrations on full-levels (units as per mmr_hydro), nhydro is the number of hydrometeor types defined in the hydrotable file (for UV/VIS/IR sensors, this first dimension is of size nhydro+1 , where the additional element corresponds to the Baran ice scheme). Not used if opts%scatt%user_hydro_opt_param is true.
Real	hydro_frac(:, nlayers)	Cloud/hydrometeor fractional cover on full-levels (0-1). First dimension size 1 if opts%cloud_overlap%per_hydro_frac is false, otherwise same size as hydro(:,:) .
Real	hydro_frac_eff	Effective hydrometeor fraction for the whole profile. Used if the cloud_overlap_2col_user_frac overlap scheme is used.
Integer	clwde_param	Cloud liquid water effective diameter parameterisation. Currently only one parameterisation is implemented so this should not be modified: clwde_martin (Martin <i>et al.</i> , 1994)
Integer	icede_param	Ice cloud effective diameter parameterisation, valid values: icede_ou_liou , icede_wyser (default), icede_boudala , icede_mcfarquhar
Real	hydro_deff(nhydro, nlayers)	Input particle effective diameter for each hydrometeor type (units: μm). Only applies to those particle types parameterised by size in the hydrotable file. Where zero, effective diameters are parameterised according to clwde_param and/or icede_param for liquid and ice cloud respectively.
Real	surface_fraction(nsurfaces-1)	Fractional coverage of surfaces 1, ..., nsurfaces-1 (0-1). RTTOV automatically calculates the fraction for the final surface. The sum of the values must not exceed 1. Do not specify if nsurfaces=1 as the surface fraction is implicitly set to 1.
Type(rtov_near_surface)	near_surface(nsurfaces)	Near-surface parameters for each surface, see above.
Type(rtov_skin)	skin(nsurfaces)	Surface skin parameters for each surface, see above.
Real	cfraction	Cloud fraction (0-1) for simple cloud scheme.
Real	ctp	Cloud top pressure (hPa) for simple cloud scheme, ignored if cfraction=0 .
Real	zenangle	Local satellite zenith angle (degrees), maximum valid value depends on coefficient file and channel (see section 3).
Real	azangle	Local satellite azimuth angle (0-360°; measured clockwise, east=90°, see Figure 8.1). Used in solar simulations and by MW sea surface emissivity models.
Real	sunzenangle	Local solar zenith angle (degrees), solar radiation only included up to 85°.
Real	sunazangle	Local solar azimuth angle (0-360°; measured clockwise, east=90°, see Figure 8.1)
Real	elevation	Surface elevation (km)
Real	latitude	Latitude (deg) -90° to +90°
Real	longitude	Longitude (deg) 0-360°. Used by emissivity and BRDF atlases.
Real	Be	Earth magnetic field strength (Gauss). Used with Zeeman coefficients only.
Real	cosbk	Cosine of the angle between the Earth magnetic field and wave propagation direction. Used with Zeeman coefficients only.
Integer	date(3)	Year, month, day. Used by solar calculations to adjust the solar irradiance based on the time of year.
Integer	time(3)	Hour, minute, second. Not currently used by core RTTOV models, but is used by <i>rtov_calc_solar_angles</i> subroutine (Annex I).
Character (len=128)	id	User may give text ID to each profile. Not used by RTTOV.

4. Emissivity/reflectance structure

The **rttov_emis_refl** structure is used to pass emissivity, reflectance, and related surface values into and out from RTTOV. You should declare an array of type **rttov_emis_refl** of size **nsurfaces**, the number of surfaces associated with each profile. Each member array is of size **nchanprof**. See section 8.3 for details on surface emissivity and reflectance in RTTOV.

Type	Variable	Description
Logical	calc_emis(:)	Where true, channel emissivities are provided by RTTOV. Where false, you provide input emissivity values in emis_in(:) . Only relevant for thermal channels.
Real	emis_in(:)	Input emissivities, used where calc_emis(:) is false.
Real	emis_out(:)	Emissivities used by RTTOV.
Logical	calc_diffuse_refl(:)	Where true, channel diffuse reflectances are provided by RTTOV. Where false, you provide input diffuse reflectance values in diffuse_refl_in(:) . Relevant for all channels.
Real	diffuse_refl_in(:)	Input diffuse reflectances, used where calc_diffuse_refl(:) is false.
Real	diffuse_refl_out(:)	Diffuse reflectances used by RTTOV.
Real	specularity(:)	Input surface specularity (0-1). Only used if opts%surface%lambertian option is true.
Real	tskin_eff(:)	Input per-channel effective skin temperature (K). Only used if opts%surface%use_tskin_eff option is true. On output, always contains skin temperatures used by RTTOV.
Logical	calc_brdf(:)	Where true, channel BRDFs are provided by RTTOV. Where false, you provide input BRDF values in brdf_in(:) . Only relevant for solar channels when opts%rt_all%solar is true.
Real	brdf_in(:)	Input BRDFs, used where calc_brdf(:) is false.
Real	brdf_out(:)	BRDFs used by RTTOV.

5. Explicit optical property structure

The **rttov_opt_param** structure is used to specify profiles of explicit optical properties for each channel for aerosol and hydrometeor scattering simulations. See section 8.4.11 for full details including information on which properties are required by each solver.

Type	Variable	Description
Real	ext(nlayers,nchanprof)	Extinction coefficients (km^{-1}).
Real	ssa(nlayers,nchanprof)	Single scattering albedo (0-1).
Real	bpr(nlayers,nchanprof)	"b" parameters (0-1): represents the fraction of backscattered radiation at each layer.
Real	asym(nlayers,nchanprof)	Asymmetry parameters (-1 – 1).
Real	phangle(nphangle)	Angles over which the phase function is defined (degrees).
Real	pha(nphangle,nlayers,nchanprof)	Phase functions (no units, must integrate to 4π over all scattering angles).
Integer	nmom	Number of Legendre coefficients for phase function expansions (excluding the zeroth coefficient which is always unity).
Real	lcoef(1:nmom+1,nlayers,nchanprof)	Phase function Legendre coefficients. First coefficient is always 1.

6. Transmission structure

The **rttov_transmission** structure contains output transmittances. The transmittances are unitless and lie in the interval [0, 1]. There are separate outputs for thermal channels and solar-affected channels for the surface-satellite transmittances. This is because the solar optical depth calculation uses the “effective” path length (the combined path length along the sun-surface-satellite path) while the thermal optical depths are calculated using the path length along the surface-satellite path. In addition, some channels use Planck-weighted coefficients (section 3) for the thermal component calculations and for these channels it is necessary also to calculate non-Planck-weighted transmittances for the solar calculations. For aerosol simulations the output transmittances include both aerosol and gas extinction. The primary transmittance outputs are for the clear column and as such never include hydrometeors. Hydrometeor simulations additionally output cloud-only transmittances which exclude gas absorption and aerosols.

Type	Variable	Description
Real	tau_total(nchanprof)	Transmittance from surface to top of atmosphere (TOA) along the satellite view path for the clear column. Only populated for channels with a significant thermally emitted contribution.
Real	tau_levels(nlevels,nchanprof)	Transmittance from each pressure half-level to TOA along the satellite view path for the clear column. Only populated for channels with a significant thermally emitted contribution.
Real	tausun_total_path2(nchanprof)	Transmittance for combined sun-surface-satellite path for the clear column. Only populated for solar-affected channels.
Real	tausun_levels_path2(nlevels, nchanprof)	Transmittance from TOA to each pressure half-level to TOA along combined sun-surface-satellite path for the clear column. Only populated for solar-affected channels.
Real	tausun_total_path1(nchanprof)	Transmittance from surface to TOA along the satellite view path for the clear column. Only populated for solar-affected channels.
Real	tausun_levels_path1(nlevels, nchanprof)	Transmittance from each pressure half-level to TOA along the satellite view path for the clear column. Only populated for solar-affected channels.
Real	tau_total_cld(nchanprof)	Transmittance of hydrometeors only (excluding gas) from surface to TOA along the satellite view path. Only available for hydrometeor scattering simulations.
Real	tau_levels_cld(nlevels, nchanprof)	Transmittance of hydrometeors only (excluding gas and aerosols) from each pressure half-level to TOA along the satellite view path. Only available for hydrometeor scattering simulations.

7. Radiance structure

The **rttov_radiance** structure contains the output radiances in units of $\text{mW/cm}^{-1}/\text{sr/m}^2$ for all channels, output brightness temperatures in K for thermal channels, and output reflectances for solar-affected channels (solar simulations only, see section 8.1).

Type	Variable	Description
Radiances - units of $\text{mW/cm}^{-1}/\text{sr/m}^2$		
Real	clear(nchanprof)	Clear sky top of atmosphere radiance output for each channel. This includes aerosol scattering for aerosol simulations.
Real	total(nchanprof)	Total simulated radiance including hydrometeor and/or aerosol scattering if enabled, or otherwise including the simple cloud scheme.
Real	cloudy(nchanprof)	For hydrometeor scattering simulations this is the total radiance from all cloudy columns as if there was no clear column (for two-column overlap schemes, this is the radiance of the cloudy column). For aerosol scattering simulations <i>without</i> hydrometeors, this is 0. For clear-sky (non-scattering) simulations, this is the simple cloud radiance assuming cfraction=1 .
Real	overcast(nlayers,nchanprof)	Level to space overcast radiance at the level bounding the bottom of each layer. For thermal channels (wavelengths $> 3\mu\text{m}$) this assumes

		an opaque black cloud. For solar channels with no thermally emitted component (wavelengths < 3µm), this consists of reflected solar radiation according to assumptions described in section 7.8.2. This is calculated only for clear-sky (non-scattering) simulations, and, for thermal channels only, for scattering simulations using the Chou-scaling thermal solver (including the effect of aerosol scattering if enabled). It is <i>not</i> calculated for PC-RTTOV simulations.
Brightness temperatures – units of deg K.		
Real	bt(nchanprof)	BT equivalent to total(:) radiance.
Real	bt_clear(nchanprof)	BT equivalent to clear(:) radiance.
Real	bt_overcast(nlayers,nchanprof)	BT equivalent to overcast(:,:) radiance. Only allocated/computed when the opts%config%bt_overcast_calc option is true.
Bi-directional reflectance factors (BRFs) – unitless		
Real	refl(nchanprof)	Reflectance (BRF) equivalent to total(:) radiance.
Real	refl_clear(nchanprof)	Reflectance (BRF) equivalent to clear(:) radiance.
Quality flag output		
Integer	quality(nchanprof)	This is a bit mask which flags cases where input profile values have exceeded the limits of internal parameterisations (see section 7.8.3 for the bit positions and their descriptions). If zero, no limits were exceeded.

8. Secondary radiance structure

The **rttov_radiance2** structure holds the “secondary” output radiances in units of $\text{mW/cm}^{-1}/\text{sr/m}^2$. These are only calculated within the direct model, but the direct model outputs can be obtained from the TL/AD/K models. These are calculated only for “clear-sky” (non-scattering) simulations, and scattering simulations using the Chou-scaling solver (clear column only, including the effect of aerosol scattering if enabled), but they are *not* calculated for PC-RTTOV simulations. They are only calculated for thermal channels and include no solar contributions.

The downwelling radiances (**dnclear, refldnclear, down**) are all computed for the combined properties of all surfaces in the case of heterogeneous surface simulations (section 8.3.10). For Lambertian simulations, the downwelling radiances are computed using the combined specularity over all surfaces. In this particular case there is not a simple relationship between the radiances in this structure, and between these radiances and the TOA radiances in the **rttov_radiance** structure.

Similarly, in the general case, the **surf** radiance in the bottom layer is the surface-leaving radiance for all surfaces combined. For this reason, it incorporates the surface emissivity. For the homogenous surface case (**nsurfaces=1**), you can obtain the black-body surface-leaving radiance by dividing **surf(nlayers,:)** by the corresponding surface emissivity **emis_refl(1)%emis_out(:)**.

Type	Variable	Description
Radiances - units of $\text{mW/cm}^{-1}/\text{sr/m}^2$		
Real	upclear(nchanprof)	Clear sky upwelling radiance at top of atmosphere including surface emission term but omitting downwelling reflected radiance term.
Real	dnclear(nchanprof)	Clear sky downwelling radiance at surface.
Real	refldnclear(nchanprof)	Reflected clear sky downwelling radiance contribution to top of atmosphere radiance.
Real	up(nlayers,nchanprof)	Summed upwelling atmospheric emission term at top of atmosphere for layers down to the level bounding the bottom of each layer.
Real	down(nlayers,nchanprof)	Summed downwelling atmospheric emission term at bottom of layer for layers down to the level bounding the bottom of each layer.
Real	surf(nlayers,nchanprof)	Radiance emitted by a black cloud at the level bounding the bottom of each layer. For the bottom layer (index nlayers), this is evaluated for the surface skin temperature <i>and includes the surface emissivity</i> (i.e., this value is <i>not</i> for a black cloud/surface). See description above.

9. Radar reflectivity structure

The `rttov_reflectivity` structure holds the output reflectivities from the radar simulator. See section 8.4.4 for more details.

Type	Variable	Description
Real	<code>zef(nlayers, nchanprof)</code>	Radar reflectivities without including the effect of attenuation [dBZ]
Real	<code>azef(nlayers, nchanprof)</code>	Radar reflectivities including attenuation [dBZ]

10. Emissivity retrieval terms structure

The `rttov_emis_retrieval_terms` structure holds outputs from RTTOV that can be used for dynamic emissivity retrievals (see section 8.4.12). The radiances are in units of $\text{mW/cm}^{-1}/\text{sr/m}^2$. These outputs are only available from the RTTOV direct model. The value of `ncolumns` is 0 for clear-sky simulations (the clear column has index 0), 1 for two-column cloud overlap schemes, and $2 * \text{nlayers}$ for the max/random overlap scheme (the theoretical maximum number of columns for max/random, elements for unused columns are set to zero in all arrays).

Type	Variable	Description
Real	<code>bsfc(nchanprof)</code>	Surface blackbody Planck radiance.
Real	<code>column_weight(0:ncolumns, nchanprof)</code>	Weight of each cloud column, $\text{SUM}(\text{column_weight}(:,i)) = 1$ for any channel index i . Used for linearly combining per-column transmittances and radiances.
Real	<code>tau_sfc(0:ncolumns, nchanprof)</code>	Along-path surface to space transmittance.
Real	<code>rad_up(0:ncolumns, nchanprof)</code>	TOA upwelling radiance from atmosphere excluding surface emission and reflection.
Real	<code>rad_down(0:ncolumns, nchanprof)</code>	Surface downwelling radiance (including cosmic term).

11. Diagnostic output structure

The `rttov_diagnostic_output` structure holds the additional per-profile outputs from RTTOV. These are obtained by declaring a variable of type `rttov_diagnostic_output`, allocating it, and passing it into RTTOV.

Type	Variable	Description
Real	<code>geometric_height(nlayers, nprofiles)</code>	Altitude of pressure full-levels (m).
Real	<code>geometric_height_half(nlayers, nprofiles)</code>	Altitude of pressure half-levels (m), excluding the top level since this may be at or arbitrarily close to 0 hPa.
Real	<code>hydro_frac_eff(nprofiles)</code>	For hydrometeor simulations only, this is one minus the weight of the clear column in the general case. For two-column overlap schemes, this is the weight of the cloudy column (section 8.4.3).

12. PC and reconstructed radiance structure

The `rttov_pccomp` structure contains the outputs of PC-RTTOV simulations (section 8.6). The `npcscores` dimension should be equal (or greater than) the number of Principal Components being simulated (`opts%pcrttov%npcscores`) multiplied by the number of profiles being passed to RTTOV per call (`nprofiles`). The `*pccomp` arrays are only allocated if `opts%pcrttov%rec_rad` is true, and their size is the number of reconstructed radiance channels multiplied by `nprofiles`.

Type	Variable	Description
Real	<code>total_pcscores(npcscores)</code>	Computed PC scores for PC-RTTOV radiances.
Real	<code>total_pccomp(nchannels_rec)</code>	Radiances reconstructed using <code>total_pcscores</code> .
Real	<code>bt_pccomp(nchannels_rec)</code>	BTs equivalent to <code>total_pccomp</code> reconstructed radiances.

Annex K – RTTOV constants

RTTOV's constants are defined in the module `src/main/rttov_const.F90`, with some additional emissivity atlas related constants defined in `src/emis_atlas/rttov_emis_atlas_mod.F90`. This section gathers all constants that may be useful in your code that calls RTTOV.

1. Constants for options

Several options (section 7.1, Annex J) are integers that allow selection between different calculations or parameterisations. It is recommended to use the constants in the table below when specifying options in your code: this makes your code self-documenting/easier to read and protects it against changes in the numerical values associated with individual options in future releases. The values in brackets are the current numerical values of each constant.

Option	Integer constants defined in <i>rttov_const</i>	Description
opts % interpolation % interp_mode	interp_rochon (1)	Rochon interpolation of profiles and optical depths.
	interp_loglinear (2)	Log-linear interpolation of profiles and optical depths.
	interp_rochon_loglinear (3)	Rochon interpolation of profiles and log-linear interpolation of optical depths.
	interp_rochon_wfn (4)	Rochon interpolation of profiles and weighting functions.
	interp_rochon_loglinear_wfn (5)	Rochon interpolation of profiles and log-linear interpolation of weighting functions.
opts % surface % solar_sea_refl_model	solar_refl_model_elfouhaily (1)	Elfouhaily <i>et al.</i> (2017) wave spectrum parameterisation for solar sea surface BRDF model
opts % surface % ir_sea_emis_model	ir_emis_model_irem (1)	IREM IR sea surface emissivity model
	ir_emis_model_irem (2)	IREMIS IR sea surface emissivity model
opts % surface % mw_sea_emis_model	mw_emis_model_fastem5 (1)	FASTEM-5 MW sea surface emissivity model
	mw_emis_model_fastem6 (2)	FASTEM-6 MW sea surface emissivity model
	mw_emis_model_surfem_ocean (3)	SURFEM-Ocean MW sea surface emissivity model
opts % clw_absorption % permittivity_param	clw_perm_liebe (1)	Liebe (1989) permittivity parameterisation
	clw_perm_rosenkranz (2)	Rosenkranz (2015) permittivity parameterisation
	clw_perm_tkc (3)	Turner, Kneifel, Cadeddu (2016) permittivity parameterisation
opts % scatt % thermal_solver	thermal_solver_dom (1)	DOM thermal solver
	thermal_solver_chou (2)	Chou-scaling solver
	thermal_solver_delta_edd (3)	Delta-Eddington solver
opts % scatt % solar_solver	solar_solver_dom (1)	DOM solar solver
opts % scatt % baran_ice_version	solar_solver_mfasis_nn (2)	MFASIS neural-network solver
	baran2014 (1)	Baran 2014 ice cloud scheme
opts % scatt % mw_pol_mode	baran2018 (2)	Baran 2018 ice cloud scheme
	mw_pol_mode_no_pol (0)	Disable polarised scattering
	mw_pol_mode_empirical (1)	Empirical scaling of extinction for V and H polarised channels, scale factor controlled by opts%scatt%ice_polarisation
opts % cloud_overlap % overlap_param	mw_pol_mode_aro_scaling (2)	ARO scaled polarisation for V, H, QV, QH polarised channels
	cloud_overlap_auto_select (0)	Automatic selection of overlap_param at run-time: for VIS/IR sensors use max/random and for MW sensors use two-column hydro-weighted scheme.
	cloud_overlap_max_random (1)	Maximum-random overlap, recommended for VIS/IR sensors
	cloud_overlap_2col_max_frac (2)	Two column, effective frac is maximum value among layers at pressures below opts%cloud_overlap%two_col_max_frac_max_p option
	cloud_overlap_2col_weighted (3)	Two column, effective frac is computed as hydrometeor-weighted average, recommended for MW sensors
cloud_overlap_2col_user_frac (4)	Two column, user specifies effective frac in hydro_frac_eff profile variable	

2. Constants for input profile

Several members of the profile structure (Annex J) are integers that allow selection between different calculations or parameterisations. It is recommended to use the constants in the table below when specifying these profile variables in your code: this makes your code self-documenting/easier to read and protects it against changes in the numerical values associated with individual options in future releases. The values in brackets are the current numerical values of each constant.

Profile variable	Integer constants defined in <i>rttov_const</i>	Description
profile % gas_units	gas_unit_ppmvdry (0)	Input gas profiles and 2m water vapour in units of ppmv over dry air
	gas_unit_kg_per_kg (1)	Input gas profiles and 2m water vapour in units of kg/kg over moist air
	gas_unit_ppmv (2)	Input gas profiles and 2m water vapour in units of ppmv over moist air
profile % skin % surftype	surftype_land (0)	Land surface
	surftype_sea (1)	Ocean or water surface
	surftype_seaice (2)	Sea-ice surface
profile % skin % watertype	watertype_fresh_water (0)	For surftype_sea , fresh water
	watertype_ocean_water (1)	For surftype_sea , saline water
profile % clwde_param	clwde_martin (1)	Martin <i>et al</i> CLW Deff parameterisation
profile % icede_param	icede_ou_liou (1)	Ou and Liou ice Deff parameterisation
	icede_wyser (2)	Wyser ice Deff parameterisation
	icede_boudala (3)	Boudala <i>et al</i> ice Deff parameterisation
	icede_mcfarquhar (4)	McFarquhar <i>et al</i> ice Deff parameterisation
profile % flux_conversion(:)	flux_conv_none (0)	Units for corresponding hydrometeor are determined by profile%mmr_hydro
	flux_conv_rain (1)	Units for corresponding hydrometeor are rain flux (kg/m ² /s)
	flux_conv_snow (2)	Units for corresponding hydrometeor are snow flux (kg/m ² /s)

Hard limits for some profile variables (section 7.7 and table 7.7) are stored as constants:

Variable	Minimum	Maximum
Satellite zenith angle	0° (no constant)	zenmax_lo (75°) for v7/v8 predictors zenmax_hi (85.3°) for v9/v13 predictors
Pressure of bottom half-level (i.e., surface pressure)	pmin (400 hPa)	pmax (1100 hPa)
Temperature (inc. 2m T and Tskin)	tmin (90 K)	tskin_land_max (1250 K) for Tskin over land only tmax (400 K) all other cases
Water vapour (inc. 2m q)	qmin (1E-11 ppmv over dry air) qmin_kgkg (6.23E-18 kg/kg over moist air)	qmax (60000 ppmv over dry air) qmax_kgkg (0.271 kg/kg over moist air)
O ₃	o3min (1E-11 ppmv over dry air)	o3max (1000 ppmv over dry air)
CO ₂	co2min (1E-11 ppmv over dry air)	co2max (1000 ppmv over dry air)
CO	comin (1E-11 ppmv over dry air)	comax (10 ppmv over dry air)
N ₂ O	n2omin (1E-11 ppmv over dry air)	n2omax (10 ppmv over dry air)
CH ₄	ch4min (1E-11 ppmv over dry air)	ch4max (50 ppmv over dry air)
SO ₂	so2min (1E-11 ppmv over dry air)	so2max (1000 ppmv over dry air)
Cloud liquid water (clear-sky MW only)	clwmin (0 kg/kg)	clwmax (1 kg/kg)
10m wind speed = (wind_u10m ² + wind_v10m ²) ^{1/2}	0 m/s (no constant)	wmax (100 m/s)
Cloud top pressure (simple cloud; only applies if cfraction > 0)	ctpmin (50 hPa)	ctpmax (1100 hPa)
Magnetic field Be (Zeeman only)	bemin (0.2 Gauss)	bemax (0.7 Gauss)

3. Constants for optical properties

Constants are provided for the indices for each hydrometeor and aerosol particle type in the hydrotable and aertable optical property files provided with RTTOV (sections 8.4.7 and 8.4.9).

Indices of hydrometeor types in hydrotables for microwave sensors:

Index	Particle type
hydro_index_rain = 1	Rain
hydro_index_snow = 2	Snow
hydro_index_graupel = 3	Graupel
hydro_index_clw = 4	Cloud liquid water
hydro_index_ciw = 5	Cloud ice water

Indices of hydrometeor types in hydrotables for visible/infrared sensors (note that the index for the Baran ice scheme is always the last one, for example, **ice_baum_index+1**):

Index	Particle type
opac_stco_index = 1	OPAC Stratus Continental
opac_stma_index = 2	OPAC Stratus Maritime
opac_cucc_index = 3	OPAC Cumulus Continental Clean
opac_cucp_index = 4	OPAC Cumulus Continental Polluted
opac_cuma_index = 5	OPAC Cumulus Maritime
clw_deff_index = 6	Cloud liquid water parameterised by particle size
ice_baum_index = 7	Baum ice cloud parameterised by particle size

Indices of aerosol types in OPAC, CAMS, and ICON-ART aertables:

Index	OPAC file aerosol type	Abbreviation
aer_opac_index_inso = 1	Insoluble	INSO
aer_opac_index_waso = 2	Water soluble	WASO
aer_opac_index_soot = 3	Soot	SOOT
aer_opac_index_ssam = 4	Sea salt (acc mode)	SSAM
aer_opac_index_sscm = 5	Sea salt (coa mode)	SSCM
aer_opac_index_minm = 6	Mineral (nuc mode)	MINM
aer_opac_index_miam = 7	Mineral (acc mode)	MIAM
aer_opac_index_micm = 8	Mineral (coa mode)	MICM
aer_opac_index_mitr = 9	Mineral transported	MITR
aer_opac_index_suso = 10	Sulphated droplets	SUSO
aer_opac_index_vola = 11	Volcanic ash	VOLA
aer_opac_index_vapo = 12	New volcanic ash	VAPO
aer_opac_index_asdu = 13	Asian dust	ASDU
Index	CAMS file aerosol type	Abbreviation
aer_cams_index_bcar = 1	Black Carbon	BCAR
aer_cams_index_dus1 = 2	Dust, bin 1, 0.03-0.55 micron, ref. index: Woodward 2001	DUS1
aer_cams_index_dus2 = 3	Dust, bin 2, 0.55-0.90 micron, ref. index: Woodward 2001	DUS2
aer_cams_index_dus3 = 4	Dust, bin 3, 0.90-20.0 micron, ref. index: Woodward 2001	DUS3
aer_cams_index_sulp = 5	Ammonium sulphate	SULP
aer_cams_index_ssa1 = 6	Sea salt, bin 1, 0.03-0.5 micron	SSA1
aer_cams_index_ssa2 = 7	Sea salt, bin 2, 0.5-5.0 micron	SSA2
aer_cams_index_ssa3 = 8	Sea salt, bin 3, 5.0-20.0 micron	SSA3
aer_cams_index_omat = 9	Hydrophilic organic matter	OMAT
Index	ICON-ART file aerosol type	Abbreviation
aer_icon_index_soot = 1	Soot/black carbon	SOOT
aer_icon_index_dusa = 2	Dust mode A	DUSA
aer_icon_index_dusb = 3	Dust mode B	DUSB

aer_icon_index_dusc = 4	Dust mode C	DUSC
aer_icon_index_ssaa = 1	Sea salt mode A	SSAA
aer_icon_index_ssab = 2	Sea salt mode B	SSAB
aer_icon_index_ssac = 3	Sea salt mode C	SSAC

4. Constants for quality flags

As discussed in section 7.8.3, the **radiance%quality(:)** output is a bit mask that provides information about when internal parameterisation limits have been exceeded by the input profile data.

Bit position integer constants defined in <i>rttov_const</i>	Description
qflag_reg_limits (0)	Gas regression limits exceeded in optical depth computation (section 7.4.3).
qflag_pc_aer_reg_limits (1)	PC-RTTOV aerosol regression limits exceeded (section 8.6).
qflag_pc_hydro_reg_limits (2)	PC-RTTOV hydrometeor regression limits exceeded (section 8.6).
qflag_emis_limits (5)	Sea surface emissivity model limits exceeded (section 8.3.1).
qflag_hydro_deff_limits (10)	Hydrometeor Deff limits exceeded (section 8.4.7).
qflag_hydro_t_limits (11)	Hydrometeor temperature limits exceeded (section 8.4.7).
qflag_hydro_conc_limits (12)	Hydrometeor concentration limits exceeded (section 8.4.7).
qflag_delta_edd_ext_limits (15)	Delta-Eddington thermal solver extinction limits exceeded (section 8.4.2).
qflag_mfasis_nn_limits (16)	MFASIS neural network parameter limit exceeded (section 8.4.2).

5. Additional useful constants

This table lists some other constants defined in **rttov_const** that may be useful in your code:

Constants defined in <i>rttov_const</i>	Description
errorstatus_success (0)	Return status value from RTTOV subroutines indicating no errors.
errorstatus_fatal (1)	Return status value from RTTOV subroutines indicating an error occurred.
alloc_flag (.true.)	Pass as “alloc” argument to rttov_alloc_* subroutines to indicate allocation.
dealloc_flag (.false.)	Pass as “alloc” argument to rttov_alloc_* subroutines to indicate deallocation.
nphangle_lores (208)	Size of phangle_lores(:) array.
phangle_lores(1:nphangle_lores)	Contains lower resolution phase angle grid used for optical property calculations for thermal channels.
nphangle_hires (498)	Size of phangle_hires(:) array.
phangle_hires(1:nphangle_hires)	Contains higher resolution phase angle grid used for optical property calculations for solar channels.
min_reflectivity (-999.0)	Radar reflectivity value set in layers containing no hydrometeors.

Some additional constants related to the emissivity atlases are defined in **src/emis_atlas/rttov_emis_atlas_mod.F90**:

Constants defined in <i>rttov_emis_atlas_mod</i>	Description
atlas_type_mw (1)	Pass as “atlas_type” argument to rttov_setup_emis_atlas subroutine to initialise a MW emissivity atlas.
atlas_type_ir (2)	“atlas_type” to initialise an IR emissivity atlas.
uwiremis_atlas_id (1)	Pass as “atlas_id” argument to rttov_setup_emis_atlas subroutine to initialise UWIR emissivity atlas.
camel_atlas_id (2)	“atlas_id” for CAMEL single-year IR emissivity atlas.
camel_clim_atlas_id (3)	“atlas_id” for CAMEL multi-year (climatology) IR emissivity atlas.
telsem2_atlas_id (1)	“atlas_id” for TELSEM2 MW emissivity atlas.
cnrm_mw_atlas_id (2)	“atlas_id” for CNRM MW emissivity atlas.

End of User Guide