ecRad radiation scheme: User Guide

Robin J. Hogan
European Centre for Medium Range Weather Forecasts, Reading, UK

Document version 1.2.0 (October 2019) applicable to *ecRad* version 1.2.x

Contents

Chapter 1

Introduction

1.1 What is *ecRad*?

ecRad is an atmospheric radiation scheme designed for computing profiles of solar (or *shortwave*) and thermal infrared (or *longwave*) irradiances from the surface up to the middle mesosphere. It is incorporated into the Integrated Forecasting System (IFS), the weather forecast model used operationally by the European Centre for Medium-Range Weather Forecasts (ECMWF), in which it is used to compute radiative heating and cooling rates of the atmosphere and surface. An offline version of the scheme is available for educational and non-commercial research use only.

A scientific overview of *ecRad* was provided by Hogan and Bozzo (2018). It incorporates the Rapid Radiative Transfer Model for GCMs (RRTMG; Iacono et al., 2008) for representing absorption by atmospheric gases and a flexible treatment of the optical properties of aerosol particles. Three different solvers capable of representing the effects of subgrid cloud structure are available: McICA (Pincus et al., 2003), Tripleclouds (Shonk and Hogan, 2008) and SPARTACUS (Hogan et al., 2016). It is coded in Fortran 2003 in a way that is efficient and flexible.

1.2 License

The Meteorological Services of ECMWF Member States are permitted to use *ecRad* (along with any IFS software) for any in-house purpose. Other researchers may apply for an institutional license to use the software, as described on the *ecRad* web site*. The license is the same as for the OpenIFS software and permits educational and non-commercial research use only, and prohibits redistribution. The full terms of the license are available on the *ecRad* web site.

1.3 Overview of this document

Chapter 2 describes how to compile and use the offline version of *ecRad*, which is essentially a Unix program that reads a configuration file and a NetCDF file containing a description of the atmospheric state, and outputs a NetCDF file containing the computed irradiance profiles. Chapter 3 describes how to incorporate *ecRad* into a larger Fortran program, such as an atmospheric model. At some future date this document will be expanded to include chapters describing the internal architecture and the detailed scientific documentation.

*https://confluence.ecmwf.int/display/ECRAD

Chapter 2

Using the offline radiation scheme

2.1 Compiling the package

The offline version of *ecRad* is designed to be used on a Unix-like platform. You will need a Fortran compiler that supports the 2003 standard, such as `gfortran`. As a prerequisite, you will need to install the NetCDF library, including the Fortran interface (packages to install on a Linux system are typically called `libnetcdff-dev` or `libnetcdff-devel`). To run some of the tests, you will also need to install the NCO utilities for manipulating NetCDF Files.

First unpack the package and enter the subdirectory as follows:

```
tar xvfz ecrad-1.2.0.tar.gz
cd ecrad-1.2.0
```

On a non-GNU platform you may need to untar and unzip the package using the `tar` and `gunzip` commands separately. The `README` file contains concise instructions on compilation and testing, while the `NOTICE` file outlines the license conditions. The subdirectories are as follows:

`radiation` The *ecRad* souce code for atmospheric radiation

`radsurf` Incomplete source code for radiation interactions with complex surfaces; this will probably be removed as a separate package is being developed

`ifsaux` Source code providing a (sometimes dummy) IFS environment

`ifsrrtm` The IFS implementation of the RRTMG gas optics scheme

`utilities` Source code for useful utilities, such as reading NetCDF files

`driver` The source code for the offline driver program `ecrad`

`ifs` Source files from the IFS that are used to illustrate how *ecRad* can be incorporated into a large model, but note that these files are not used in the offline version

`mod` Where Fortran module files are written

`lib` Where the static libraries are written

`bin` Where the executable `ecrad` is written

`data` Contains configuration data files read at run-time

`test` Test cases including Matlab code to plot the outputs

`include` Automatically generated interface blocks for non-module routines

Compilation on different platforms using different compilers is facilitated by the various `Makefile_include.<prof>` files in the top-level directory: if you type

```
make
```

or

```
make PROFILE=gfortran
```

the code will be compiled using the `gfortran` compiler via the Makefile variables set in the `Makefile_include.gfortran` file. Using instead `PROFILE=pgi` will use the `Makefile_include.pgi` file to attempt to compile with the PGI compiler. If everything goes to plan this should create the executable `bin/ecrad` and various static libraries in the `lib` directory.

One common reason the code doesn't compile out of the box is that it can't find the NetCDF library files. Since version 1.2.0, the *ecRad* Makefile uses the `nf-config` script that comes with recent versions of the NetCDF library to create the Makefile variables `NETCDF_INCLUDE` and `NETCDF_LIB`. If `nf-config` is not available on your system, or it fails to correctly locate the NetCDF library files, then the cleanest way to fix this is to create a `Makefile_include.local` file that defines `NETCDF_INCLUDE` and `NETCDF_LIB` explicity to contain arguments for the compile and link operations, respectively. Suppose you installed NetCDF in `/path/to/netcdf` and you use the `gfortran` compiler then your file might contain:

```
include Makefile_include.gfortran
NETCDF = /path/to/netcdf
NETCDF_INCLUDE = -I$(NETCDF)/include
NETCDF_LIB = -L$(NETCDF)/lib -lnetcdff -lnetcdf -Wl,-rpath,$(NETCDF)/lib
```

You should then be able to build the code with

```
make PROFILE=local
```

Examples of such configurations for the ECMWF and University of Reading computer systems may be found in `Makefile_include.ecmwf` and `Makefile_include.uor`.

To compile in single precision, type

```
make PROFILE=gfortran SINGLE_PRECISION=1
```

To compile with debugging options (no optimization, bounds checking and initializing real numbers with not-a-number), type

```
make PROFILE=gfortran DEBUG=1
```

Finer tuning may be achieved by specifying the optimization and debugging flags explicitly, for example

```
make PROFILE=gfortran OPTFLAGS="-O1" DEBUGFLAGS="-g1 -pg"
```

Remember that if you change the compile settings you will probably want to recompile everything, in which case you first need to remove all compiled files with

```
make clean
```

2.2 Running the offline radiation scheme

To test the code, type

```
make test
```

which runs `make` in each of the subdirectories of the `test` directory. The `README` files in these directories provide more information on what they are doing, and some Matlab scripts are provided to visualize the outputs.

You will see in the output of the tests the command line in each invocation of *ecRad*, which is of the form

```
ecrad config.nam input.nc output.nc
```

where `ecrad` needs to be the full path to the *ecRad* executable, `config.nam` is a Fortran namelist file configuring the code, `input.nc` contains the input atmospheric profiles and `output.nc` contains the output irradiance (flux) profiles. The namelist file contains a `radiation` namelist that configures the *ecRad* scheme itself; the parameters available are described in section 2.3. The file also contains a `radiation_config` namelist that configures aspects of the offline package, described in section 2.4. Only the `radiation` namelist is used when *ecRad* is incorporated into an atmospheric model.

The input NetCDF file contains numerous floating-point variables listed in Table 2.1. The dimensions are shown in the order that they are listed by the `ncdump` utility, with the first dimension varying slowest in the file (opposite to the Fortran convention). Most variables are stored as a function of column and level (dimensions named `col` and `level` in Table 2.1, although the actual dimension names are ignored by *ecRad*). The `half_level` dimension corresponds to the mid-points of the levels, plus the top-of-atmosphere and surface, and so must be one more than `level`. The `level_interface` dimension excludes the top-of-atmosphere and surface so must be one less than `level`. The optional `sw_albedo_band` and `lw_emiss_band` dimensions allow for shortwave albedo and longwave emissivity to be specified in user-defined spectral intervals. Some variables can be omitted in which case default values will be used or these fields will be constructed according to `radiation_config` namelist parameters (section 2.4).

Table 2.1: Main variables contained in the input NetCDF file to *ecRad*. Note that some variables are not required if they are not used by the particular solver selected, for example `iseed` is only used by the McICA solver and `inv_cloud_effective_size` is only used by the SPARTACUS solver. Also, only one of `o3_mmr` and `o3_vmr` should be provided. In addition to ozone, further gases can be specified in either mass mixing ratio (suffix `_mmr`) or volume mixing ratio (suffix `_vmr`) units, where the prefixes are `co2` (carbon dioxide), `n2o` (nitrous oxide), `co` (carbon monoxide), `ch4` (methane), `o2` (molecular oxygen), `cfc11` (CFC-11), `cfc12` (CFC-12), `hcfc22` (HCFC-22), `ccl4` (carbon tetrachloride) and `no2` (nitrogen dioxide). These further trace gases may either be specified as variable in space (dimensioned `col,level`) or constant (a scalar value in the file). To override the suffix indicating volume mixing ratio (e.g. to change it to `_mole_fraction`), set the namelist variable `vmr_suffix_str` as described in Table 2.4.

Variable	Dimensions	Description
`solar_irradiance`	–	Solar irradiance at Earth's orbit (W m^{-2})
`skin_temperature`	`col`	Skin temperature (K)
`cos_solar_zenith_angle`	`col`	Cosine of solar zenith angle
`sw_albedo`	`col, sw_albedo_band`	Shortwave albedo (if 1D then assumed spectrally constant)
`lw_emissivity`	`col, lw_emiss_band`	Longwave emissivity (if 1D then assumed spectrally constant)
`iseed`	`col`	Seed for McICA random-number generator (double precision, default: 1, 2, 3...)
`pressure_hl`	`col, half_level`	Pressure at half levels (Pa)
`temperature_hl`	`col, half_level`	Temperature at half levels (K)
`q` or `h2o_mmr`	`col, level`	Specific humidity (kg kg^{-1})
`h2o_vmr`	`col, level`	Water vapour volume mixing ratio (mol mol^{-1})
`o3_mmr`	`col, level`	Ozone mass mixing ratio (kg kg^{-1})
`o3_vmr`	`col, level`	Ozone volume mixing ratio (mol mol^{-1}), used only if `o3_mmr` not provided
`aerosol_mmr`	`col, aer_type, level`	Aerosol mass mixing ratio (kg kg^{-1})
`q_liquid`	`col, level`	Liquid cloud mass mixing ratio (kg kg^{-1})
`q_ice`	`col, level`	Ice cloud mass mixing ratio (kg kg^{-1})
`re_liquid`	`col, level`	Liquid cloud effective radius (m)
`re_ice`	`col, level`	Ice cloud effective radius (m)
`cloud_fraction`	`col, level`	Cloud fraction
`overlap_param`	`col, level_interface`	Cloud overlap parameter (default: compute from decorrelation length of 2 km)
`fractional_std`	`col, level`	Fractional standard deviation of cloud optical depth (default 0)
`inv_cloud_effective_size`	`col, level`	Inverse of cloud effective horizontal size for SPARTACUS solver (m^{-1})

inv_inhom_effective_size	col, level	Inverse of effective horizontal size of cloud inhomogeneities, for SPARTACUS solver (m^{-1}) (default: same as inv_cloud_effective_size)
inv_cloud_effective_separation	col, level	Alternative input to SPARTACUS if inv_cloud_effective_size not present (m^{-1})
inv_inhom_effective_separation	col, level	Alternative input to SPARTACUS if inv_inhom_effective_size not present (m^{-1})

All the test data store input fields in order of increasing pressure, i.e. starting at the top-of-atmosphere and working down to the surface. The output data are then provided using the same convention. If input data are provided in the opposite order then this should be automatically detected and under the bonnet the order is reversed before being passed to the radiation scheme. But if you use this convention then please test the results carefully as this option is not regularly tested. The variables describing cloud properties, particularly sub-grid cloud struture, are defined in detail in section 2.5.

The output NetCDF file contains the typical set of variables listed in Table 2.2. Clear-sky fluxes (i.e. computed on the same input profiles but in the absence of clouds) are provided if the do_clear namelist parameter is set to true (see section 2.3). If you need diagnostic downward fluxes at the surface for just a subset of the spectrum (e.g. ultraviolet or photosynthetically active radiation) then they can be computed from the spectral_flux_dn_* variables, activated if namelist variable do_surface_sw_spectral_flux is set to true. In some contexts it is also useful to have fluxes in each of the shortwave albedo or longwave emissivity spectral intervals. These are named canopy_flux_dn_* and are activated if do_canopy_fluxes_sw or do_canopy_fluxes_lw are set to true. Note that if you want atmospheric heating rates then you will need to compute them yourself from the flux profiles.

Table 2.2: Variables contained in the output NetCDF file from *ecRad*, where all fluxes (or irradiances) have units of W m^{-2}. The band_sw dimension has the same size as the number of shortwave bands in the gas-optics scheme.

Variable	Dimensions	Description
pressure_hl	col, half_level	Pressure at half levels (Pa)
flux_up_sw, flux_dn_sw	col, half_level	Up- and downwelling shortwave fluxes
flux_up_sw_clear, flux_dn_sw_clear	col, half_level	Up- and downwelling clear-sky shortwave fluxes
flux_dn_direct_sw	col, half_level	Direct component of downwelling shortwave flux
flux_dn_direct_sw_clear	col, half_level	Direct component of downwelling clear-sky shortwave flux
flux_up_lw, flux_dn_lw	col, half_level	Up- and down-welling longwave fluxes
flux_up_lw_clear, flux_dn_lw_clear	col, half_level	Up- and down-welling clear-sky longwave fluxes
lw_derivative	col, half_level	Derivative of upwelling longwave flux with respect to surface value (Hogan and Bozzo, 2015)
spectral_flux_dn_sw_surf	col, band_sw	Downwelling surface shortwave flux in each band
spectral_flux_dn_direct_sw_surf	col, band_sw	Direct downwelling surface shortwave flux in each band
spectral_flux_dn_sw_surf_clear	col, band_sw	Clear-sky downwelling surface shortwave flux in each band
spectral_flux_dn_direct_sw_surf_clear	col, band_sw	Clear-sky direct downwelling surface shortwave flux in each band
canopy_flux_dn_diffuse_sw_surf	col, sw_albedo_band	Downwelling diffuse surface shortwave flux in each albedo interval
canopy_flux_dn_direct_sw_surf	col, sw_albedo_band	Downwelling direct surface shortwave flux in each albedo interval
canopy_flux_dn_lw_surf	col, lw_emiss_band	Downwelling surface longwave flux in each emissivity interval
cloud_cover_sw	col	Total cloud cover diagnosed by shortwave solver
cloud_cover_lw	col	Total cloud cover diagnosed by longwave solver

2.3 Configuring the radiation scheme

The detailed settings of *ecRad* are configured using the `radiation` namelist in the namelist file provided as the first command-line argument to the `ecrad` executable. The available namelist parameters are listed in Table 2.3. One of the most important is `directory_name`, which provides the absolute or relative path to the directory containing all the configuration files. This is the `data` directory at the top level of the *ecRad* package. Note that the default values listed in Table 2.3 may differ in some cases from the values used operationally in the IFS (see Table 2 of Hogan and Bozzo, 2018).

Table 2.3: Options for the `radiation` namelist that configures the radiation scheme. The type of each parameter can be inferred from its name: logicals begin with `do_` or `use_`, integers start with `i_` or `n_`, strings end with `_name`, and all other parameters are real numbers.

Parameter	**Default value**, other values	Description
General		
`directory_name`	**.**	Directory containing NetCDF configuration files
`do_sw`	**`true`**	Compute shortwave fluxes?
`do_lw`	**`true`**	Compute longwave fluxes?
`do_sw_direct`	**`true`**	Do direct shortwave fluxes?
`do_clear`	**`true`**	Compute clear-sky fluxes?
Gas and aerosol optics		
`gas_model_name`	**RRTMG-IFS**, Monochromatic	Gas optics model
`use_aerosols`	**`false`**	Do we represent aerosols?
`do_lw_aerosol_scattering`	**`true`**	Do longwave aerosol scattering?
`n_aerosol_types`		Number of aerosol types
`i_aerosol_type_map`		Vector of integers that map from aerosol types to types in the NetCDF aerosol optics file, where positive integers indexe hydrophobic types, negative integers index hydrophilic types and zero indicates a type should be ignored
`aerosol_optics_override_file_name`		Path to an alternative aerosol optics file
Monochromatic scheme		
`mono_lw_wavelength`	**−1.0**	Wavelength of longwave radiation, or if negative, a broadband calculation will be performed
`mono_lw_total_od`	**0.0**	Zenith longwave optical depth of clear-sky atmosphere
`mono_sw_total_od`	**0.0**	Zenith shortwave optical depth of clear-sky atmosphere
`mono_lw_single_scattering_albedo`	**0.538**	Longwave cloud single scattering albedo
`mono_sw_single_scattering_albedo`	**0.999999**	Shortwave cloud single scattering albedo
`mono_lw_asymmetry_factor`	**0.925**	Longwave cloud asymmetry factor
`mono_sw_asymmetry_factor`	**0.86**	Shortwave cloud asymmetry factor
Cloud optics		
`liquid_model_name`	**SOCRATES**, Slingo, Monochromatic	Liquid optics model, including the scheme in the SOCRATES radiation scheme and the older scheme of Slingo (1989)
`ice_model_name`	**Fu-IFS**, Baran2016, Yi, Monochromatic	Ice optics model, including the schemes of Fu (1996), Fu et al. (1998), Baran et al. (2016) and Yi et al. (2013)
`do_lw_cloud_scattering`	**`true`**	Do longwave cloud scattering?
`do_fu_lw_ice_optics_bug`	**`false`**	Reproduce bug in McRad implementation of Fu ice optics (Hogan et al., 2016)?
`liq_optics_override_file_name`		Path to alternative liquid optics file name
`ice_optics_override_file_name`		Path to alternative ice optics file name
Solver		
`sw_solver_name`	Cloudless, Homogeneous, **McICA**, Tripleclouds, SPARTACUS	Shortwave solver; note that the homogeneous solver assumes cloud fills the gridbox horizontally (so ignores cloud fraction) while the cloudless solver ignores clouds completely

`lw_solver_name`	Cloudless, Homogeneous, **McICA**, Tripleclouds, SPARTACUS	Longwave solver
`overlap_scheme_name`	Max-Ran, **Exp-Ran**, Exp-Exp	Cloud overlap scheme; note that SPARTACUS and Tripleclouds only work with the Exp-Ran overlap scheme
`use_beta_overlap`	**`false`**	Use Shonk et al. (2010) 'β' overlap parameter definition, rather than default 'α'?
`cloud_inhom_decorr_scaling`	**0.5**	Ratio of overlap decorrelation lengths for cloud inhomogeneities and boundaries
`cloud_fraction_threshold`	$\mathbf{10^{-6}}$	Ignore clouds with fraction below this
`cloud_mixing_ratio_threshold`	$\mathbf{10^{-9}}$	Ignore clouds with total mixing ratio below this
`cloud_pdf_shape_name`	**Gamma**, Lognormal	Shape of cloud water PDF
`cloud_pdf_override_file_name`		Name of NetCDF file of alternative cloud PDF look-up table
`do_sw_delta_scaling_with_gases`	**`false`**	Apply delta-Eddington scaling to particle-gas mixture, rather than particles only (see Hogan and Bozzo, 2018)
SPARTACUS solver (these parameters have no effect for other solvers)		
`do_3d_effects`	**`true`**	Represent cloud edge effects when SPARTACUS solver selected; note that this option does not affect entrapment, which is also a 3D effect
`n_regions`	2, **3**	Number of regions, where one is clear sky and one or two are cloud (the Tripleclouds solver always assumes three regions regardless of this parameter)
`do_lw_side_emissivity`	**`true`**	Represent effective emissivity of the side of clouds (Schäfer et al., 2016)
`sw_entrapment_name`	Zero, Edge-only, **Explicit**, Non-fractal, Maximum	Entrapment model (Hogan et al., 2019); note that the behaviour in ecRad version 1.0.1 was 'Maximum' entrapment
`do_3d_lw_multilayer_effects`	**`false`**	Maximum entrapment for longwave radiation?
`max_3d_transfer_rate`	**10.0**	Maximum rate of lateral exchange between regions in one layer, for stability of matrix exponential (where the default means that as little as e^{-10} of the radiation could remain in a region)
`max_gas_od_3d`	**8.0**	3D effects ignored for spectral intervals where gas optical depth of a layer exceeds this, for stability
`max_cloud_od`	**16.0**	Maximum in-cloud optical depth, for stability
`use_expm_everywhere`	**`false`**	Use matrix-exponential method even when 3D effects not important, such as clear-sky layers and parts of the spectrum where the gas optical depth is large?
`clear_to_thick_fraction`	**0.0**	Fraction of cloud edge interfacing directly to the most optically thick cloudy region
`overhead_sun_factor`	**0.0**	Minimum tan-squared of solar zenith angle to allow some 'direct' radiation from overhead sun to pass through cloud sides (0.06 used by Hogan et al., 2016)
`overhang_factor`	**0.0**	A detail of the entrapment representation described by Hogan et al. (2019)
Surface		
`do_nearest_spectral_sw_albedo`	**true**	Surface shortwave albedos may be supplied in their own spectral intervals: do we select the nearest to each band of the gas optics scheme, rather than using a weighted average?
`do_nearest_spectral_lw_emiss`	**true**	...likewise but for surface longwave emissivity
`sw_albedo_wavelength_bound`		Vector of the wavelength bounds (m) delimiting the shortwave albedo intervals
`lw_emiss_wavelength_bound`		Vector of the wavelength bounds (m) delimiting the longwave emissivity intervals
`i_sw_albedo_index`		Vector of indices mapping albedos to wavelength intervals
`i_lw_emiss_index`		Vector of indices mapping emissivities to wavelength intervals
Diagnostics		

iverbosesetup	0, 1, 2, **3**, 4, 5	Verbosity in setup, where 1=warning, 2=info, 3=progress, 4=detailed, 5=debug
iverbose	0, **1**, 2, 3, 4, 5	Verbosity in execution
do_save_spectral_flux	**false**	Save flux profiles in each band?
do_save_gpoint_flux	**false**	Save flux profiles in each g-point?
do_surface_sw_spectral_flux	**true**	Save surface shortwave fluxes in each band for subsequent diagnostics?
do_lw_derivatives	**false**	Compute derivatives for Hogan and Bozzo (2015) approximate updates?
do_save_radiative_properties	**false**	Write intermediate NetCDF file(s) of properties sent to solver (radiative_properties*.nc)?
do_canopy_fluxes_sw	**false**	Save surface shortwave fluxes in each albedo interval
do_canopy_fluxes_lw	**false**	Save surface longwave fluxes in each emissivity interval

Several of the entries in Table 2.3 are configured with vectors of numbers, which deserves further explanation. As shown in Table 2.1, aerosols are provided to *ecRad* in the form of the mass mixing ratios of a number of different aerosol types. The optical properties of an arbitrary number of hydrophilic and hydrophobic aerosol types is provided in a NetCDF file, for example `data/aerosol_ifs_rrtm_45R2.nc` in the *ecRad* package. The mapping between the input aerosol concentrations and the aerosol types in the optical-property file may be specified in the `radiation` namelist. The `n_aerosol_types` parameter specifies the number of aerosol concentrations to be provided, with a value of zero having the effect of deactivating aerosols. `i_aerosol_type_map` is a vector of integers of length `n_aerosol_types` indicating which aerosol type to select from the optical-property file. Negative numbers select hydrophilic types, whose optical properties vary with relative humidity, while postitive numbers select hydrophobic types. Zero indicates that an input aerosol type is to be ignored. As an example, the IFS settings (in the `test/ifs` directory) are specified with:

```
aerosol_optics_override_file_name = 'aerosol_ifs_rrtm_46R1_with_NI_AM.nc'
n_aerosol_types = 12
i_aerosol_type_map = -1, -2, -3, 1, 2, 3, -4, 10, 11, 11, -5, 14
```

When *ecRad* is run, the output printed to the terminal includes a description of the aerosol mapping.

A similar mechanism is used to describe how spectral intervals of the input `sw_albedo` and `lw_emissivity` should be interpretted. This is best explained by considering the configuration of the IFS in Cycle 47R1, which is described by the following namelist variables:

```
sw_albedo_wavelength_bound(1:5) = 0.25e-6, 0.44e-6, 0.69e-6, 1.19e-6, 2.38e-6
i_sw_albedo_index(1:6) = 1,2,3,4,5,6
do_nearest_spectral_sw_albedo = false
lw_emiss_wavelength_bound(1:2) = 8.0e-6, 13.0e-6
i_lw_emiss_index(1:3) = 1,2,1
do_nearest_spectral_lw_emiss = true
```

The IFS describes surface albedo in six spectral intervals. The vector `sw_albedo_wavelength_bounds` here provides the wavelengths, in metres, of the five boundaries between these intervals, where the first interval is taken to include all wavelengths shorter than the first value (in this case 0.25 μm) and the last includes all wavelengths longer than the last value (in this case 2.38 μm). The vector `i_sw_albedo_index` specifies which of the elements of the input `sw_albedo` field should be used in each of the six spectral intervals. Surface emissivity is described similarly: there are three spectral intervals specified by the two boundaries in `lw_emiss_wavelength_bound`. The corresponding vector `i_lw_emiss_index` contains two occurrences of the index `1`, indicating that the first element of `lw_emissivity` is used both for wavelengths smaller than 8 μm and wavelengths larger than 13 μm (i.e. outside the infrared atmospheric window). The second element is then used for wavelengths between these two boundaries. Thus even though there are three spectral intervals, only two elements are needed in `lw_emissivity`. Finally, the logicals `do_nearest_spectral_sw_albedo` and `do_nearest_spectral_lw_emiss` specify whether the bands of the gas optics scheme used in *ecRad* will use a single value of albedo or emissivity from the input fields (chosen to be the spectral interval with the largest overlap in wavenumber space with each band of the gas-optics scheme), or whether they will weight the spectral intervals

by their overlap with each band of the gas-optics scheme. The mapping from spectral interval to band is printed on standard output when *ecRad* is run, as shown in the example in section 2.6.

2.4 Configuring the offline package

In addition to the namelist parameters described in section 2.3 an additional set of parameters are available in the `radiation_config` namelist that are specific to the offline version of *ecRad* and are listed in Table 2.4. In general if these parameters are present in the namelist then they will override the corresponding variable provided in the input file.

Table 2.4: Options for the `radiation_config` namelist that configures additional aspects of the offline radiation scheme. All entries must be scalars. If an override parameter is present then it need not be included in the input file. The cloud effective sizes (used by the SPARTACUS solver) may be specified for low, middle and high clouds according to the cloud layer pressure p and the surface pressure p_0.

Parameter	Description
Execution control	
`nrepeat`	Number of times to repeat, for benchmarking
`istartcol`	Start at specified input column (1 based)
`iendcol`	End at specified input column (1 based)
`iverbose`	Verbosity in offline setup (default 2)
`do_parallel`	Use OpenMP parallelism? (default `true`)
`nblocksize`	Number of columns per block when using OpenMP
`do_save_inputs`	Sanity check: save input variables in `inputs.nc`
`do_correct_unphysical_inputs`	If input variables out of physical bounds, correct them and issue a warning
`vmr_suffix_str`	Suffix for variables containing volume mixing ratios (default '`_vmr`')
Override input variables	
`solar_irradiance_override`	Override solar irradiance (W m^{-2})
`skin_temperature`	Override skin temperature (K)
`cos_solar_zenith_angle`	Override cosine of solar zenith angle
`sw_albedo`	Override shortwave albedo
`lw_emissivity`	Override longwave emissivity
`fractional_std`	Override cloud optical depth fractional standard deviation
`overlap_decorr_length`	Override cloud overlap decorrelation length (m)
`inv_effective_size`	Override inverse of cloud effective size (m^{-1})
`low_inv_effective_size`	...for low clouds ($p > 0.8p_0$, where p is pressure and p_0 surface pressure)
`middle_inv_effective_size`	...for mid-level clouds ($0.45p_0 < p \leq 0.8p_0$)
`high_inv_effective_size`	...for high clouds ($p \leq 0.45p_0$)
Scale input variables	
`q_liquid_scaling`	Scaling for liquid water mixing ratio
`q_ice_scaling`	Scaling for ice water mixing ratio
`cloud_fraction_scaling`	Scaling for cloud fraction (capped at 1)
`overlap_decorr_length_scaling`	Scaling for cloud overlap decorrelation length
`effective_size_scaling`	Scaling for cloud effective size
`h2o_scaling, co2_scaling...`	Scaling for specific humidity and carbon dioxide; equivalents available for `o3`, `co`, `ch4`, `n2o`, `o2`, `cfc11`, `cfc12`, `hcfc22` and `ccl4`
Parameterize input variables	
`cloud_inhom_separation_factor`	Set inhomogeneity separation scale to be this multiplied by cloud separation scale
`cloud_separation_scale_surface`	Surface cloud separation scale in pressure-dependent parameterization
`cloud_separation_scale_toa`	Top-of-atmosphere cloud separation scale in pressure-dependent parameterization
`cloud_separation_scale_power`	Power in cloud separation scale parameterization

2.5 Describing cloud structure

Probably more than any other 1D radiation scheme, *ecRad* allows the user to define in detail the statistical properties of the sub-grid cloud distribution, and in this section the relevant variables and namelist parameters are

explained in more detail. In an operational context most of these variables need to be parameterized, but in developing new solvers we need to perform explicit radiation calculations on realistic high resolution 3D cloud fields, and compare them to *ecRad* simulations in which the profiles of these variables have been extracted from the 3D cloud fields. This has been done by Schäfer et al. (2016), Hogan et al. (2016) and Hogan et al. (2019). Explicit radiation calculations on a 3D cloud field can either be performed using the Independent Column Approximation (ICA) and compared to *ecRad*'s McICA or Tripleclouds solvers, or using a fully 3D solver (e.g. Monte Carlo) and comparing it to *ecRad*'s SPARTACUS solver. Note that *ecRad* can itself perform ICA calculations on 3D cloud fields, by flattening the two horizontal dimensions of a 3D dataset into a single 'column' dimension, and using the *ecRad*'s 'Homogeneous' solver in which any cloud is assumed to homogeneously fill each of the narrow columns (so cloud fraction is not used as it is implicitly taken to be 0 or 1).

The input variables describing the profile of cloud properties are given in the lower half of Table 2.1. The most basic are the liquid and ice mass mixing ratios (`q_liquid` and `q_ice`), which are gridbox-mean quantities, and the corresponding effective radii (`re_liquid` and `re_ice`). Effective radius is assumed to be horizontally constant within a gridbox, even if the water content varies. For all cloud optics models, effective radius is defined as

$$r_{e,\mathrm{liq}} = \frac{3\mathrm{LWC}}{4\rho_{\mathrm{liq}} A_{\mathrm{liq}}}; \tag{2.1}$$

$$r_{e,\mathrm{ice}} = \frac{3\mathrm{IWC}}{4\rho_{\mathrm{ice}} A_{\mathrm{ice}}}, \tag{2.2}$$

where LWC and IWC are the liquid and ice water contents (i.e. the mass mixing ratios multiplied by the air density), ρ_{liq} and ρ_{ice} are the densities of liquid water and solid ice, and A_{liq} and A_{ice} are the total projected cross-sectional areas of liquid droplets and ice particles per unit volume of air (so units of m^{-1}).

Cloud fraction is simply the fractional horizontal area of a given model layer that contains cloud. The layers are assumed to be thin enough that cloud fraction is constant with height within a layer, i.e. cloud fraction by volume is equal to cloud fraction by area. The horizontal variability of cloud water content in a layer is specified by the fractional standard deviation (`fractional_std`), defined as the standard deviation of the in-cloud water content, divided by the in-cloud mean water content. The in-cloud mean water content is the gridbox-mean water content divided by cloud fraction. Note that since effective radius is assumed constant across a gridbox, cloud optical depth is proportional to water path and so `fractional_std` can also be thought of as the horizontal fractional standard deviation of cloud optical depth. Moreover, *ecRad* assumes that horizontal variations of liquid and ice water content are perfectly correlated. As shown in Table 2.4, fractional standard deviation can be overriden through a namelist parameter; for example, in the IFS this value is set to 1.

Cloud overlap is needed by the Exp-Ran and Exp-Exp overlap schemes, and is specified at the interface (or half-level) between each layer by `overlap_param`, the overlap parameter as defined by Hogan and Illingworth (2000). To compute this at half-level $i + 1/2$ of a high-resolution 3D cloud field, you need the cloud fractions in the upper and lower lower layers, c_i and c_{i+1}, and the combined cloud cover of the cloud in these two layers, C. Then from Eqs. 1, 2 and 4 of Hogan and Illingworth (2000) you can compute the overlap parameter:

$$\alpha_{i+1/2} = \frac{C_{\mathrm{rand}} - C}{C_{\mathrm{max}} - C}, \tag{2.3}$$

where the combined cloud covers that would be obtained from the random and maximum overlap assumptions are

$$C_{\mathrm{rand}} = c_i + c_{i+1} - c_i c_{i+1}; \tag{2.4}$$

$$C_{\mathrm{max}} = \max(c_i, c_{i+1}). \tag{2.5}$$

Alternatively, cloud overlap can be parameterized as in most atmospheric models in terms of an overlap decorrelation length as shown in Table 2.4, which implements Eq. 5 of Hogan and Illingworth (2000). In addition to describing how cloud boundaries overlap, *ecRad* needs to know how sub-grid cloud inhomogeneities are vertically correlated. This cannot be specified at each layer, but is rather specified via the namelist variable `cloud_inhom_decorr_scaling` in Table 2.3, which gives the ratio of the decorrelation lengths for cloud inhomogeneities and cloud boundaries. The default value of 0.5 was obtained from observations of ice clouds by Hogan and Illingworth (2003).

The variables and parameters above are all used by the McICA and Tripleclouds solvers to represent cloud properties relevant for 1D radiative transfer. In order to use the SPARTACUS solver to represent 3D radiative effects, we also need a means to specify the *normalized cloud perimeter length, L*, in each model layer. If we imagine a horizontal slice through the sub-grid cloud field, then L is the total cloud perimeter length divided by the area of the domain, with units of inverse metres. This variable is not provided to SPARTACUS directly, since it tends to be strongly dependent on the cloud fraction. Rather we specify either the *cloud effective size,* C_S, or the *cloud effective separation,* C_X, which tend to be less dependent on cloud fraction. Normalized perimeter length is related to the former via Eq. 29 of Hogan et al. (2019):

$$L = 4c(1-c)/C_S, \tag{2.6}$$

and to the latter via (Fielding et al., 2020)

$$L = 4\left[c(1-c)\right]^{1/2}/C_X, \tag{2.7}$$

where c is the cloud fraction. The variables $1/C_S$ and $1/C_X$ may be specified directly in the input file as `inv_cloud_effective_size` and `inv_cloud_effective_separation`, respectively. If both are present then the former will take precedence. The reason that reciprocals are provided is that then a value of zero (corresponding to C_S or C_X of infinity) indicates no 3D effects are to be simulated in a particular layer. If you have a high resolution cloud scene and you wish to wish to run SPARTACUS on it then you need to compute the perimeter length from it (e.g. use a contouring function on a field containing 0 for clear sky and 1 for cloud, and then compute the length of the 0.5 contour), and knowing also cloud fraction you can invert (2.6) or (2.7).

In the context of an atmospheric model, we recommend that C_X is parameterized using the namelist parameters at the bottom of Table 2.4 scheme with the values of Fielding et al. (2020):

```
cloud_separation_scale_toa     = 14000.0, ! Value of C_X at top-of-atmosphere (m)
cloud_separation_scale_surface = 2500.0,  ! Value of C_X at surface (m)
cloud_separation_scale_power   = 3.5,     ! Describes pressure dependence of C_X
cloud_inhom_separation_factor  = 0.75     ! Defines size of cloud inhomogeneities
```

These numbers are used in the namelist in the `test/ifs` case. Note that the first number shown here, C_X^{TOA}, is valid for a model with a horizontal grid spacing of around 100 km, but this parameter was found by Fielding et al. (2020) to be dependent on horizontal grid spacing Δx in a way that can be fitted with

$$C_X^{\mathrm{TOA}} = 1.62\,\Delta x^{0.47}, \tag{2.8}$$

where both C_X^{TOA} and Δx are in km. The surface value of C_X can be assumed to be 2.5 km for all model resolutions.

2.6 Checking the configuration

When `ecrad` is run, it outputs to the screen a summary of the configuration options, the files read and written and details of the aerosol mapping. This can be used to check that *ecRad* has been configured as intended. The following is an example from the default test in the `test/ifs` directory, in the case of `iverbosesetup=2` and `iverbose=1` in the `radiation` namelist:

```
------------------------- OFFLINE ECRAD RADIATION SCHEME -------------------------
Copyright (C) 2014-2019 European Centre for Medium-Range Weather Forecasts
Contact: Robin Hogan (r.j.hogan@ecmwf.int)
Floating-point precision: double
General settings:
  Data files expected in "../../data"
  Clear-sky calculations are ON                          (do_clear=T)
  Saving intermediate radiative properties OFF           (do_save_radiative_properties=F)
  Saving spectral flux profiles ON                       (do_save_spectral_flux=T)
  Gas model is "RRTMG-IFS"                               (i_gas_model=1)
  Aerosols are ON                                        (use_aerosols=T)
  Clouds are ON                                          (do_clouds=T)
Surface settings:
```

```
  Saving surface shortwave spectral fluxes OFF               (do_surface_sw_spectral_flux=F)
  Saving surface shortwave fluxes in ableDo bands ON         (do_canopy_fluxes_sw=T)
  Saving surface longwave fluxes in emissivity bands ON      (do_canopy_fluxes_lw=T)
  Longwave derivative calculation is ON                      (do_lw_derivatives=T)
  Nearest-neighbour spectral albedo mapping OFF              (do_nearest_spectral_sw_albedo=F)
  Nearest-neighbour spectral emissivity mapping ON           (do_nearest_spectral_lw_emiss=T)
Cloud settings:
  Cloud fraction threshold = .100E-05                        (cloud_fraction_threshold)
  Cloud mixing-ratio threshold = .100E-08                    (cloud_mixing_ratio_threshold)
  Liquid optics scheme is "SOCRATES"                         (i_liq_model=2)
  Ice optics scheme is "Fu-IFS"                              (i_ice_model=2)
  Longwave ice optics bug in Fu scheme is OFF                (do_fu_lw_ice_optics_bug=F)
  Cloud overlap scheme is "Exp-Exp"                          (i_overlap_scheme=2)
  Use "beta" overlap parameter is OFF                        (use_beta_overlap=F)
  Cloud PDF shape is "Gamma"                                 (i_cloud_pdf_shape=1)
  Cloud inhom decorrelation scaling = .500                   (cloud_inhom_decorr_scaling)
Solver settings:
  Shortwave solver is "McICA"                                (i_solver_sw=2)
  Shortwave delta scaling after merge with gases OFF         (do_sw_delta_scaling_with_gases=F)
  Longwave solver is "McICA"                                 (i_solver_lw=2)
  Longwave cloud scattering is ON                            (do_lw_cloud_scattering=T)
  Longwave aerosol scattering is OFF                         (do_lw_aerosol_scattering=F)
Warning: turning on do_surface_sw_spectral_flux as required by do_canopy_fluxes_sw
Reading ../../data/RADRRTM
Reading ../../data/RADSRTM
Weighting of 6 albedo values in 14 shortwave bands (wavenumber ranges in cm-1):
  2600 to  3250: 0.00 0.00 0.00 0.00 0.00 1.00
  3250 to  4000: 0.00 0.00 0.00 0.00 0.00 1.00
  4000 to  4650: 0.00 0.00 0.00 0.00 0.69 0.31
  4650 to  5150: 0.00 0.00 0.00 0.00 1.00 0.00
  5150 to  6150: 0.00 0.00 0.00 0.00 1.00 0.00
  6150 to  7700: 0.00 0.00 0.00 0.00 1.00 0.00
  7700 to  8050: 0.00 0.00 0.00 0.00 1.00 0.00
  8050 to 12850: 0.00 0.00 0.00 0.93 0.07 0.00
 12850 to 16000: 0.00 0.00 0.48 0.52 0.00 0.00
 16000 to 22650: 0.00 0.00 1.00 0.00 0.00 0.00
 22650 to 29000: 0.00 0.99 0.01 0.00 0.00 0.00
 29000 to 38000: 0.00 1.00 0.00 0.00 0.00 0.00
 38000 to 50000: 0.83 0.17 0.00 0.00 0.00 0.00
   820 to  2600: 0.00 0.00 0.00 0.00 0.00 1.00
Mapping from 16 longwave bands to emissivity intervals: 1 1 1 1 1 2 2 2 1 1 1 1 1 1 1 1
Reading NetCDF file ../../data/socrates_droplet_scattering_rrtm.nc
Reading NetCDF file ../../data/fu_ice_scattering_rrtm.nc
Reading NetCDF file ../../data/aerosol_ifs_rrtm_46R1_with_NI_AM.nc
Aerosol mapping:
   1 -> hydrophilic type 1: Sea salt, bin 1, 0.03-0.5 micron, OPAC
   2 -> hydrophilic type 2: Sea salt, bin 2, 0.50-5.0 micron, OPAC
   3 -> hydrophilic type 3: Sea salt, bin 3, 5.0-20.0 micron, OPAC
   4 -> hydrophobic type 1: Desert dust, bin 1, 0.03-0.55 micron, (SW) Dubovik et al. 2002...
   5 -> hydrophobic type 2: Desert dust, bin 2, 0.55-0.90 micron, (SW) Dubovik et al. 2002...
   6 -> hydrophobic type 3: Desert dust, bin 3, 0.90-20.0 micron, (SW) Dubovik et al. 2002...
   7 -> hydrophilic type 4: Hydrophilic organic matter, OPAC
   8 -> hydrophobic type 10: Hydrophobic organic matter, OPAC (hydrophilic at RH=20%)
   9 -> hydrophobic type 11: Black carbon, OPAC
  10 -> hydrophobic type 11: Black carbon, OPAC
  11 -> hydrophilic type 5: Ammonium sulfate (for sulfate), GACP Lacis et al https://gacp...
  12 -> hydrophobic type 14: Stratospheric sulfate (hydrophilic ammonium sulfate at RH 20%-30%)
Reading NetCDF file ../../data/mcica_gamma.nc
Reading NetCDF file ecrad_meridian.nc
  Warning: variable co_vmr not found
  Warning: variable no2_vmr not found
Writing NetCDF file inputs.nc
Performing radiative transfer calculations
Writing NetCDF file ecrad_meridian_default_out.nc
--------------------------------------------------------------------------------
```

Chapter 3

Incorporating *ecRad* into another program

ecRad can be called within a larger program, and indeed it has been incorporated into several atmospheric models (the IFS, Meso-NH and ICON). Pending a full description here of how to do this, see the `ifs/radiation_setup.F90` in the *ecRad* package to see how it is configured in the IFS, and `ifs/radiation_scheme.F90` for how it is run.

When calling *ecRad* from within a model, the parameters listed in Table 2.3 are members of the `config_type` structure, and may be modified within the code at the appropriate place in the configuration stage. The exception is in the case of strings, which are prefixed by `_name` in the namelist. In the `config_type` structure there are equivalent integers to express these parameters, which can be changed using the named constants listed in Table 3.1.

Table 3.1: Integers in the `config_type` structure that represents the strings in Table 2.3, where a namelist parameter named `*_name` would be named `i_*` here.

Variable in `config_type`	Available named constants, **default**
`i_overlap_scheme`	`IOverlapMaximumRandom`, **`IOverlapExponentialRandom`**, `IOverlapExponential`
`i_solver_sw`, `i_solver_lw`	`ISolverCloudless`, `ISolverHomogeneous`, **`ISolverMcICA`**, `ISolverSpartacus`, `ISolverTripleclouds`
`i_3d_sw_entrapment`	`IEntrapmentZero`, `IEntrapmentEdgeOnly`, **`IEntrapmentExplicit`**, `IEntrapmentExplicitNonFractal`, `IEntrapmentMaximum`
`i_gas_model`	`IGasModelMonochromatic`, **`IGasModelIFSRRTMG`**
`i_liq_model`	`ILiquidModelMonochromatic`, **`ILiquidModelSOCRATES`**, `ILiquidModelSlingo`
`i_ice_model`	`IIceModelMonochromatic`, **`IIceModelFu`**, `IIceModelBaran2016`, `IIceModelYi`
`i_cloud_pdf_shape`	**`IPdfShapeGamma`**, `IPdfShapeLognormal`

Bibliography

Baran, A. J., P. Hill, D. Walters, S. C. Hardiman, K. Furtado, P. R. Field and J. Manners, 2016: The impact of two coupled cirrus microphysicsradiation parameterizations on the temperature and specific humidity biases in the tropical tropopause layer in a climate model. *J. Climate,* **29,** 5299–5316.

Fielding, M. D., S. A. K. Schäfer, R. J. Hogan and R. M. Forbes, 2020: Encapsulating cloud geometry for 3D radiative transfer and cloud turbulent mixing parameterizations. *To be submitted to Q. J. R. Meteorol. Soc.*

Fu, Q., 1996: An accurate parameterization of the solar radiative properties of cirrus clouds. *J. Climate,* **9,** 2058–2082.

Fu, Q., P. Yang and W. B. Sun, 1998: An accurate parametrization of the infrared radiative properties of cirrus clouds of climate models. *J. Climate,* **11,** 2223-2237.

Hogan, R. J., and A. J. Illingworth, 2000: Deriving cloud overlap statistics from radar. *Q. J. R. Meteorol. Soc.,* **126**, 2903–2909.

Hogan, R. J., and A. J. Illingworth, 2003: Parameterizing ice cloud inhomogeneity and the overlap of inhomogeneities using cloud radar data. *J. Atmos. Sci.,* **60,** 756–767.

Hogan, R. J., and A. Bozzo, 2015: Mitigating errors in surface temperature forecasts using approximate radiation updates. *J. Adv. Model. Earth Syst.,* **7,** 836–853.

Hogan, R. J., and A. Bozzo, 2016: ECRAD: a new radiation scheme for the IFS. *ECMWF Technical Memorandum 787,* available at http://www.ecmwf.int/en/elibrary/16901-ecrad-new-radiation-scheme-ifs

Hogan, R. J., and A. Bozzo, 2018: A flexible radiation scheme for the ECMWF model. *J. Adv. Model. Earth Syst.,* **10,** doi:10.1029/2018MS001364.

Hogan, R. J., S. A. K. Schäfer, C. Klinger, J.-C. Chiu and B. Mayer, 2016: Representing 3D cloud-radiation effects in two-stream schemes: 2. Matrix formulation and broadband evaluation. *J. Geophys. Res.,* **121,** 8583–8599.

Hogan, R. J., M. D. Fielding, H. W. Barker, N. Villefranque and S. A. K. Schäfer, 2019: Entrapment: An important mechanism to explain the shortwave 3D radiative effect of clouds. *J. Atmos. Sci.,* **76,** 2123–2141.

Iacono, M. J., J. S. Delamere, E. J. Mlawer, M. W. Shephard, S. A. Clough and W. D. Collins, 2008: Radiative forcing by longlived greenhouse gases: Calculations with the AER radiative transfer models. *J. Geophys. Res.,* **113,** D13103, doi: 10.1029/2008JD009944.

Pincus, R., H. W. Barker, and J.-J. Morcrette, 2003: A fast, flexible, approximate technique for computing radiative transfer in inhomogeneous clouds. *J. Geophys. Res. Atmos.,* **108,** 4376, doi:10.1029/2002JD003322.

Schäfer, S. A. K., R. J. Hogan, C. Klinger, J.-C. Chiu and B. Mayer, 2016: Representing 3D cloud-radiation effects in two-stream schemes: 1. Longwave considerations and effective cloud edge length. *J. Geophys. Res.,* **121,** 8567–8582.

Shonk, J. K. P., and R. J. Hogan, 2008: Tripleclouds: an efficient method for representing horizontal cloud inhomogeneity in 1D radiation schemes by using three regions at each height. *J. Climate,* **21,** 2352–2370.

Shonk, J. K. P., R. J. Hogan, J. M. Edwards and G. G. Mace, 2010: Effect of improving representation of horizontal and vertical cloud structure on the Earth's radiation budget: 1. Review and parameterisation. *Q. J. R. Meteorol. Soc.,* **136,** 1191–1204.

Slingo, A., 1989: A GCM parametrization for the shortwave radiative properties of water clouds. *J. Atmos. Sci.,* **46,** 1419–1427.

Yi, B., P. Yang, B. A. Baum, T. L'Ecuyer, L. Oreopoulos, E. J. Mlawer, A. J. Heymsfield and K.-K. Liou, 2013: Influence of ice particle surface roughening on the global cloud radiative effect. *J. Atmos. Sci.,* **70,** 2794–2807.