EUMETSAT ROM SAF

RADIO OCCULTATION METEOROLOGY

ROPP-9.1 Release Notes

Version 9.1

30th June 2019

The ROM SAF Consortium

Danish Meteorological Institute (DMI)

European Centre for Medium-Range Weather Forecasts (ECMWF)

Institut d'Estudis Espacials de Catalunya (IEEC)

Met Office (METO)

DOCUMENT AUTHOR TABLE

	Author(s)	*Function*	*Date*	*Comment*
Prepared by:	I Culverwell	ROPP Development Team	30/06/2019	
Reviewed by:	F Warrick	ROPP Development Team	30/06/2019	
Approved by:	K B Lauritsen	ROM SAF Project manager	30/06/2019	

DOCUMENT CHANGE RECORD

Issue/Revision	*Date*	*By*	*Description*
8.0	31/12/2014	DO	Release Notes for ROPP Release v8.0
8.1	31/12/2015	IC	Release Notes for ROPP Release v8.1
9.0	28/02/2017	IC	Release Notes for ROPP Release v9.0
9.1	30/06/2019	IC	Release Notes for ROPP Release v9.1

ROM SAF

The Radio Occultation Meteorology Satellite Application Facility (ROM SAF) is a decentralised processing centre under EUMETSAT which is responsible for operational processing of GRAS radio occultation (RO) data from the Metop satellites and radio occultation data from other missions. The ROM SAF delivers bending angle, refractivity, temperature, pressure, humidity, and other geophysical variables in near real-time for NWP users, as well as reprocessed Climate Data Records (CDRs) and Interim Climate Data Records (ICDRs) for users requiring a higher degree of homogeneity of the RO data sets. The CDRs and ICDRs are further processed into globally gridded monthly-mean data for use in climate monitoring and climate science applications.

The ROM SAF also maintains the Radio Occultation Processing Package (ROPP) which contains software modules that aids users wishing to process, quality-control and assimilate radio occultation data from any radio occultation mission into NWP and other models.

The ROM SAF Leading Entity is the Danish Meteorological Institute (DMI), with Cooperating Entities: (i) European Centre for Medium-Range Weather Forecasts (ECMWF) in Reading, United Kingdom; (ii) Institut D'Estudis Espacials de Catalunya (IEEC) in Barcelona, Spain; and (iii) Met Office in Exeter, United Kingdom. To get access to our products or to read more about the ROM SAF please go to: http://www.romsaf.org

Intellectual Property Rights

List of Contents

Index of Tables

Executive Summary

This document pertains to the first update to the ninth major release of the Radio Occultation Processing Package (ROPP) – hereafter referred to as simply **ROPP-9.1**. It describes the set of files, documentation, dependency packages, a summary 'Quick Start' guide to build/installation, a list of supported platforms and known problems.

ROPP is a collection of software modules (provided as source code), supporting build scripts, data files and documentation. The complete package is split into several modules. Users may wish to integrate a subset of ROPP code into their own software applications, individually linking modules to their own code. These users may not require the complete ROPP-9.1 distribution package. Alternatively, users may wish to use the executable tools provided as part of each module as standalone applications for RO data processing. These users should download the complete ROPP-9.1 release.

1. Introduction

1.1 Purpose of the document

This document pertains to the first update to the ninth major release of the Radio Occultation Processing Package (ROPP) – hereafter referred to simply as **ROPP-9.1**. It describes the set of files, documentation, dependency packages, a summary 'Quick Start' guide to build/installation, a list of supported platforms and known problems.

1.2 Applicable and reference documents

1.2.1 Applicable documents

The following documents explain the context of ROPP within the ROM SAF.

[AD.1] Proposal for the third Continuous Development and Operations Phase (ROM SAF CDOP-3) March 2017 – February 2022.
Ref: SAF/ROM/DMI/MGT/CDOP3/001

[AD.2] Co-operation Agreement between EUMETSAT and DMI, Lead Entity for the CDOP-3 of the ROM SAF, signed at the 86th Council meeting on 7th December 2016.
Ref: EUM/C/85/16/DOC/19

[AD.3] Product Requirements Document (PRD).
Ref: SAF/GRAS/METO/MGT/PRD/001

[AD.4] System Requirements Document (SRD).
Ref: SAF/ROM/DMI/RQ/SRD/001

1.2.2 Reference documents

The following documents provide supplementary or background information, and could be helpful in conjunction with this document:

[RD.1] ROPP Change Log – Summary of changes from ROPP-9 (v9.0)
Ref: SAF/ROM/METO/SRN/ROPP/016

[RD.2   ROPP Overview
Ref: SAF/ROM/METO/UG/ROPP/001

[RD.3] ROPP User Guides
Ref: SAF/ROM/METO/UG/ROPP/002 – ROPP_IO module
Ref: SAF/ROM/METO/UG/ROPP/004 – ROPP_PP module
Ref: SAF/ROM/METO/UG/ROPP/005 – ROPP_APPS module
Ref: SAF/ROM/METO/UG/ROPP/006 – ROPP_FM module
Ref: SAF/ROM/METO/UG/ROPP/007 – ROPP_1D-Var module

[RD.4] WMO FM94 (BUFR) specification for radio occultation data
Ref: SAF/ROM/METO/FMT/BUFR/001

[RD.5] ROPP Test Plan
Ref: SAF/ROM/METO/TP/ROPP/001

1.3 Acronyms and abbreviations

AIX	Advanced Interactive eXecutive (IBM)
API	Application Programming Interface
APPS	Applications module of ROPP
BUFR	Binary Universal Form for the Representation of data (also: FM94) (WMO)
Beidou	Chinese GNSS navigation system. Beidou-2 also known as COMPASS
CDOP	Continuous Development and Operations Phase (EUMETSAT)
CDR	Climate Data Record
CMA	Chinese Meteorological Agency
DMI	Danish Meteorological Institute; ROM SAF Leading Entity
ECMWF	The European Centre for Medium-range Weather Forecasts
EPS	EUMETSAT Polar Satellite System
EUMETSAT	EUropean organisation for the exploitation of METeorological SATellites
FY-3C/D	GNSS navigation receivers (CMA)
GCC	GNU Compiler Collection (not to be confused with `gcc`, the GCC C-compiler)
CHAMP	Challenging Mini-satellite Payload (Germany)
GNOS	GNSS Radio Occultation Sounder (China)
GNU	GNU's Not Unix
GPS	Global Positioning System
GNSS	Global Navigation Satellite System (generic GPS/GLONASS/Galileo/Beidou)
COSMIC	Constellation Observing System for Meteorology Ionosphere and Climate (USA/Taiwan)
GRACE-A/B	Gravity Recovery and Climate Experiment (Germany/USA)
GRACE-FO	GRACE Follow-on experiment (Germany/USA)
GRAS	GNSS Receiver for Atmospheric Sounding (EPS/Metop)
GRIB	GRIdded Binary format (WMO)
HDF5	Hierarchical Data Format version 5
IBM	International Business Machines Corporation
ICDR	Interim Climate Data Record
I/Q	In-phase and Quadrature signal components
IEEC	Institut d'Estudis Espacials de Catalunya
ISRO	Indian Space Research Organisation
KMA	Korean Meteorological Agency
KOMPSAT-5	GNSS radio occultation receiver (KMA)
Megha-Tropiques	Tropical water cycle (and RO) experiment (India/France)
Met Office	Meteorological Office of the United Kingdom
MetDB	Meteorological DataBase (Met Office)
Metop	Meteorological Operational Polar satellite (EUMETSAT)
NCO	Numerically Controlled Oscillator
netCDF	Network Common Data Format
NRT	Near Real Time
OS	Operating System
POSIX	Portable Operating System Interface
RHEL	Red Hat Enterprise Linux
RO	Radio Occultation (also: GPS-RO)
ROM SAF	Radio Occultation Meteorology SAF (formerly GRAS SAF)
ROPP	Radio Occultation Processing Package
RS	Raw Sampling
SAF	Satellite Application Facility (EUMETSAT)
SNR	Signal to Noise Ratio

TanDEM-X	German Earth observation satellite carrying an RO sounder
TerraSAR-X	German Earth observation satellite carrying an RO sounder
UCAR	University Center for Atmospheric Research (Boulder, CO, USA)

1.4   Definitions

RO data products from the GRAS instrument onboard Metop and RO data from other missions are grouped in data levels (level 0, 1, 2, or 3) and product types (NRT, Offline, CDR, or ICDR). The data levels and product types are defined below.[1] The lists of variables should not be considered as the complete contents of a given data level, and not all data may be contained in a given data level.

Data levels:

Level 0: Raw sounding, tracking and ancillary data, and other GNSS data before clock correction and reconstruction;

Level 1A: Reconstructed full resolution excess phases, total phases, pseudo ranges, SNR's, orbit information, I, Q values, NCO (carrier) phases, navigation bits, and quality information;

Level 1B: Bending angles and impact parameters, tangent point location, and quality information;

Level 2: Refractivity, geopotential height, "dry" temperature profiles (level 2A), pressure, temperature, specific humidity profiles (level 2B), surface pressure, tropopause height, planetary boundary layer height (level 2C), ECMWF model level coefficients (level 2D); quality information;

Level 3: Gridded or resampled data, that are processed from level 1 or 2 data, and that are provided as, e.g., daily, monthly, or seasonal means on a spatiotemporal grid, including metadata, uncertainties and quality information.

Product types:

NRT product: Data product delivered less than: (i) 3 hours after measurement (SAF Level 2 for EPS); (ii) 80 min after measurement (SAF Level 2 for EPS-SG Global Mission); (iii) 40 min after measurement (SAF Level 2 for EPS-SG Regional Mission).

Offline product: Data product delivered from less than 5 days to up to 6 months after measurement, depending on the requirements. The evolution of this type of product is driven by new scientific developments and subsequent product upgrades.

CDR: Climate Data Record generated from a dedicated reprocessing activity using a fixed set of processing software[2].   The data record covers an extended time period of several years (with a fixed end point) and constitutes a homogeneous data record appropriate for climate usage.

ICDR: An Interim Climate Data Record regularly extends in time a (Fundamental or Thematic) CDR using a system having optimum consistency with and lower latency than the system used to generate the CDR[3].

---

[1] Note that the level definitions differ partly from the WMO definitions:
http://www.wmo.int/pages/prog/sat/dataandproducts_en.php
[2] (i) GCOS 2016 Implementation Plan; (ii) *http://climatemonitoring.info/home/terminology/*
[3] *http://climatemonitoring.info/home/terminology/* (the ICDR definition was endorsed at the 9th session of the joint CEOS/CGMS Working Group Climate Meeting on 29 March 2018)

2. Overview

This document pertains the the first update to the ninth major release of the Radio Occultation Processing Package (ROPP) – hereafter referred to simply as **ROPP-9.1**. It describes the set of files, documentation, dependency packages, a summary 'Quick Start' guide to build/installation, a list of supported platforms and known problems.

ROPP is a collection of software modules (provided as source code), supporting build scripts, data files and documentation. The complete package is split into several modules. Users may wish to integrate a subset of ROPP code into their own software applications, individually linking modules to their own code. These users may not require the complete ROPP-9.1 distribution package. Alternatively, users may wish to use the executable tools provided as part of each module as stand-alone applications for RO data processing. These users should download the complete ROPP-9.1 release. Section 5 has guidance for unpacking and building ROPP.

The download website is organized as:

```
ROPP Download (root) page
  |-- ROPP-9.1
    |-- Essentials
    |-- Documents
    |-- Code
  |-- Dependency Packages
  |-- Previous Versions
```

The `Essentials` sub-directory contains the Release Notes (this file), a Change Log [RD.1] and a compressed tar archive file (`ROPP-9.1.tar.gz`) which provides the complete release for ROPP-9.1. The `Code` sub-directory contains compressed tar files for the individual components (source and build modules) for this release. These are provided should users wish to access individual modules rather than download the complete package. Related documentation is also available separately in the `Documents` sub-directory. Third-party dependency packages (not part of the ROPP distribution and for those whose licences permit it) can be downloaded from the `Dependency Packages` directory. See the `README` files in these directories for more information. The `Previous Releases` directory contains a small number of older ROPP releases, but please note that support for these is limited and decreases with age.

ROPP-9.1 IO, PP, FM, 1DVAR and APPS modules have been fully validated using operational Level 1b and Level 2 GRAS data. As such, it is guaranteed for operational use. All parts of this release have been tested against CHAMP, GRACE-A, COSMIC and GRAS data. In addition, limited samples of data from TerraSAR-X, C/NOFS and SAC-C have been tested with the BUFR encoding and decoding tools and appropriate GFZ-to-ROPP and UCAR-to-ROPP converters.

Almost all files described here can be downloaded by registered and logged-in users from the ROM SAF website:

<p align="center"><u>http://www.romsaf.org</u> > User Login > ROPP > ROPP Download</p>

Any exceptions are noted below.

Before being able to download any ROPP-9.1 files, you will have accepted the Terms and Conditions of the full ROPP Software Licence, a copy of which is available in the `ROPP-9.1 Essentials sub-`directory for reference. Note also the contents of the `COPYRIGHT` file.

All comments on, or questions about, the ROPP software should in the first instance be reported via the ROM SAF Helpdesk at <u>http://www.romsaf.org</u> > Helpdesk

3. Release file set

The 'Status' column in the following tables shows the current status of files since the previous release (v9.0), by a traffic light system:

Unchanged	Updated	New

A summary of the main differences between this package (v9.1) and the previous release (v9.0) are documented in the Change Log ([RD.1]).

3.1 `ROPP-9.1/Essentials` sub-directory

Table 1 Shows the contents of this directory.

File Name	Description	Version	Status
`README.ropp`	Additional notes on the software	v9.1	Updated
`romsaf_ropp_licence.pdf`	ROPP Software Licence	v1.3	Unchanged
`COPYRIGHT`	Copyright notices	n/a	Updated
`ROPP_Release_Notes_v91.pdf`	This file	v9.1	New
`ROPP_Changes_v90-v91.pdf`	Change Log - summary of changes from Release Version 9.0 to 9.1 [RD.1]	v1.0	New
`ropp-9.1.tar.gz`	Compressed tar archive of the whole ROPP-9.1 distribution set	v9.1	New

Table 1. Files in the `Essentials` sub-directory

3.2 `ROPP-9.1/Code` sub-directory

For users not requiring the complete distribution package, this directory contains the component module files as `gzip` compressed `tar` files (known as *tarballs*), as listed in Table 2.

Ref: SAF/ROM/METO/SRN/ROPP/001
Issue: 9.1
Date: 30 June 2019

ROPP-9.1 Release Notes

EUMETSAT
ROM SAF

File Name	*Description*	*Version*	*Status*
`ropp_utils-9.1.tar.gz`	ROPP Utilities module.	v9.1	New
`ropp_io-9.1.tar.gz`	ROPP Input/Output module	v9.1	New
`ropp_pp-9.1.tar.gz`	ROPP Pre-Processor module	v9.1	New
`ropp_fm-9.1.tar.gz`	ROPP Forward Model module	v9.1	New
`ropp_1dvar-9.1.tar.gz`	ROPP 1D-Var module	v9.1	New
`ropp_apps-9.1.tar.gz`	ROPP Applications module	v9.1	New
`ropp_build-9.1.tar.gz`	A collection of configure and build scripts for a number of compilers & platforms suitable for ROPP and the dependency packages. See Section 5	v9.1	New

Table 2. Files in the `Code` sub-directory

Note that the collective content of these separate module tarballs is contained in the main tarball in the `Essentials` sub- directory.

3.3   `ROPP-9.1/Documents` **sub-directory**

The ROPP Overview, User Guides and (automatically generated) Reference Manuals for the ROPP modules, and the BUFR specification for RO data can be found here, as shown in Table 3.

File Name	Description	Version	Status
`README.docs`	Additional notes on the documents	v9.1	Updated
`romsaf_ropp_ov.pdf`	ROPP Overview document [RD.2]	v9.1	Updated
`romsaf_ropp_ug_io.pdf`	ROPP_IO User Guide [RD.3]	v9.1	Updated
`romsaf_ropp_ug_fm.pdf`	ROPP_FM User Guide [RD.3]	v9.1	Updated
`romsaf_ropp_ug_1dvar.pdf`	ROPP_1DVAR User Guide [RD.3]	v9.1	Updated
`romsaf_ropp_ug_pp.pdf`	ROPP_PP User Guide [RD.3]	v9.1	Updated
`romsaf_ropp_ug_apps.pdf`	ROPP_APPS User Guide [RD.3]	v9.1	Updated
`romsaf_bufr.pdf`	BUFR template specification for RO data [RD.4]	v2.5	Updated

`ropp_docs.tar.gz`	A compressed tar archive containing all of the above files plus the Reference Manuals for the individual modules:
	`romsaf_ropp_utils_rm.pdf`
	`romsaf_ropp_io_rm.pdf`
	`romsaf_ropp_pp_rm.pdf`
	`romsaf_ropp_fm_rm.pdf`
	`romsaf_ropp_1dvar_rm.pdf`
	`romsaf_ropp_apps_rm.pdf`
	The Reference Manuals are also included in the respective module distribution files in the `Codes` sub-directory.
		n/a	Updated

Table 3. Files in the `Documents` *sub-directory*

Notes:

1. None.

Please refer to the Overview document [RD.2] for the general content of the individual modules, and to the User Guides [RD.3] for details of the installation and usage of the package.

For users wishing to write their own interfaces to the ROPP routines, or to modify the ROPP code, detailed user-documentation in the form of Reference Manuals, one for each module, are provided. These Reference Manuals are also contained in the main ROPP tarball. Stand-alone application tools also have Unix-style 'man' pages which are installed during the ROPP module building process.

3.4   `Dependency Packages` **directory**

ROPP uses some standard third-party packages. Where licences permit, the latest versions known to be compatible with ROPP can be downloaded from the ROPP Download page on the ROM SAF website; these are listed in Table 4. The ROPP User Guide [RD.3] and file `README.deps` document the original source of these packages.

Object library and header, include and module files generated from the third-party source code should be placed in the target path being used to build ROPP. Alternatively, soft links to their actual locations may be placed in this path. See Section 5 for installation tips. *Unless existing installations are known to have been compiled with the same configuration options, compiler and compiler options as being used for ROPP, we strongly recommend building these dependency packages specifically for ROPP.*

File Name	Description	Version	Status (see Note)
`README.deps`	Additional notes on the dependency packages	v9.0	Unchanged
`zlib-1.2.8.tar.gz`	ZLIB software (for HDF5) http://www.zlib.net/	v1.2.8	Unchanged (2)
`hdf5-1.8.16.tar.gz`	HDF5 software (for netCDF-4) http://www.hdfgroup.org/HDF5/	v1.8.16	Unchanged (2)
`netcdf-c-4.4.0.tar.gz`	NetCDF-Core library http://www.unidata.ucar.edu/software/netcdf/	v4.4.0	Unchanged (2)
`netcdf-fortran-4.4.3.tar.gz`	NetCDF-Fortran library http://www.unidata.ucar.edu/software/netcdf/	v4.4.3	Unchanged (2)
`bufr-24.0.2a.tar.gz`	Met Office 'MetDB' BUFR kernel library and tools http://www.romsaf.org	v24.0.2	Updated (4,7)
`bufr_000387a.tar.gz`	ECMWF BUFR kernel library https://software.ecmwf.int/wiki/display/BUFR	v387	Updated (4,7)
`grib_api-1.14.5-Source.tar.gz`	ECMWF GRIB API https://software.ecmwf.int/wiki/display/GRIB	v1.14.5	Unchanged (5)
`sofa_f-20160503_c.tar.gz`	IAU SOFA library http://www.iausofa.org/	2016-05-03	New (6)

Table 4. Files in the `Dependency Packages` *directory*

Notes:

1. Later versions may become available from the referenced source websites; we do not guarantee that these will work with ROPP. If later versions are proved to be compatible, the new distribution files will be posted here.

2. `netCDF-4` is the current netCDF release family. With effect from ROPP9.0, build support for 'classic' `netCDF-4` (ie, using '`netCDF-3`' mode) has been dropped, and is only provided for 'full fat' `netCDF-4`. This implies the need for an `HDF5` library, which, in turn, implies the need for a `zlib` compression library, although most Linux/Unix systems will include this as standard. The `HDF5` configure should find the system `zlib` automatically, but it will build without compression support if it cannot be found. The netCDF core and Fortran libraries must be built separately and in that order.

3. The latest supported Met Office BUFR library is `bufr-24.0.2` A v2x.y.z release is required to build the BUFR tools in ROPP-9.1. Release v24.0.2 includes support for GNOS on FY-3C.

4. The ROPP-9.1 BUFR encoder and decoder can be used with either the Met Office or ECMWF BUFR libraries but only one need be installed. The ROPP build system will detect which is present and make the executable with the appropriate source code. If both libraries are installed, the MetDB one will be used in preference.

5. Users attempting to build `GRIB_API` under Cygwin are advised that `buildpack` is likely to fail, since it only runs `make install` if `make check` is successful. The latter currently fails because of some apparent `INT4/INT8` and `REAL4/REAL8` inconsistencies. These appear to be harmless as

far as ROPP is concerned, because if the user runs `make install` directly from the `grib_api` directory, no problems result. This is therefore the recommended course of action for such users.

6. The ROPP mini-configure scripts for the SOFA library simply edit the default makefile, included in the SOFA distribution, which was constructed for gfortran on a linux platform. They are therefore not configuration files in the usual sense of files that generate makefiles.

7. THE ROPP9.1 versions of the BUFR libraries only differ from the standard versions (which do not have a final 'a' in their names) by the inclusion of codes for new satellites and processing centres. This is simply for the sake of completeness: the new codes are included in a namelist included in the ROPP distribution, and this takes precedence over the codes in the library. ROPP9.1 can therefore be run with either of the old BUFR libraries, `bufr-24.0.2` or `bufr_000387`. The only difference is that certain tools outside ROPP, such as the `decbufr` utility in the MetDB BUFR library, will not pick up the new codes if run with the old libraries.

Sample build and configuration scripts are available to assist in building these packages consistently for use with ROPP; see Section 5.4.

4. Previous versions

If you have installed any previous version of ROPP we recommend that you install v9.1 to a separate target path – do not just install v9.1 over an existing installation. This is especially important if the earlier tools or applications built on those libraries are being used for operational production. Only swing over to using v9.1 tools and libraries when you have validated them locally and you are satisfied that they meet your requirements.

Support for v8.0 and v9.0 will be limited to problems which can be reproduced with v9.1. If reported problems with v8.0 or v9.0 have been already fixed at v9.1, we recommend upgrading to v9.1. ROPP versions up to and including v7.1 are no longer supported.

All previous Beta release versions of ROPP, up to and including v9.0-beta, are obsolete and are completely unsupported. If you have previously installed any version of ROPP for Beta testing, under the terms of the Beta licence (to which you agreed when first requesting access to the software) the right to use this software terminated after one year. Any and all ROPP Beta files downloaded for testing purposes are unsupported and should be deleted.

5. Unpacking and building ROPP

This section provides guidance on building the complete ROPP package as a standalone installation for evaluation and testing or to build the application tools for local use. ROPP is designed to be highly modular and key components of the software may be extracted for embedding in user applications via user-callable application program interface (API). In the latter case, the 3rd party dependency packages may not be required. See the User Guide for guidance on use of the main ROPP APIs.

5.1 Unpacking

For simplicity, we recommend downloading the main `ropp-9.1.tar.gz` compressed tar archive (approx. 40 Mb) which contains the complete ROPP-9.1 package set. Download (or copy) this file to a suitable directory, where the content can be extracted with:

```
> tar  -zxvf  ropp-9.1.tar.gz
```

or

```
> gunzip  -c  ropp-9.1.tar.gz  |  tar  -xv
```

if your version of tar does not support decompression (`-z` option).

This will create a sub-directory with the following structure:

```
ropp-9.1
       |- configure
       |- ropp_1dvar
       |- ropp_apps
       |- ropp_fm
       |- ropp_io
       |- ropp_pp
       |- ropp_utils
```

The ropp-9.1 top directory contains additional `README` files and example build scripts; the `configure` sub-directory contains a number of mini-scripts for local build configuration. The other `ropp_*` sub-directories contain the source code, makefiles, test data, etc., for each of the ROPP modules.

Alternatively, if the complete package is not required, the user can manually create the `ropp-9.1` directory (or one with any other suitable name), download the individual module tarballs from the website's ROPP-9.1 directory to it and unpack them here. For example:

```
> mkdir -p $HOME/ROPP/ropp-9.1
> cd $HOME/ROPP/ropp-9.1
...download or copy required tarballs here...
> tar -zxvf ropp_utils-9.1.tar.gz
> tar -zxvf ropp_io-9.1.tar.gz
...etc
```

Note that the `buildpack` script will unpack a module's tarball if the source tree is not found, so it is not necessary to manually unpack the ROPP modules if using this tool.

We suggest that any required dependency packages also be downloaded here; they can then be unpacked and built using the `buildpack` tool (or wrapper script) before building the ROPP modules.

5.2 Preparing to build the ROPP code

The full package tarball contains a plain-text file called `MANIFEST` which lists all of the files in the distribution. The individual module tarballs each contain a similar file for their own module.

Third-party dependency libraries must be installed before attempting to build any ROPP code. The `configure` sub-directory contains some configuration mini-scripts and there are higher-level shell scripts to assist in building the `zlib`, `HDF5`, `netCDF`, `BUFR` (both Met Office and ECMWF), `GRIB-API` (ECMWF) ans `SOFA` (IAU) packages. Review and edit these to suit your local requirements before running them. Refer to the `README.build`, `README.compilers`, `README.unix,` `README.deps` and `README.cygwin` files included in the package for further information.

Any third-party packages already installed (e.g. for previous ROPP builds or for other applications) do not normally need to be re-built but should be in the expected ROPP build path for libraries, as real files or as symbolic links. Note that this release may require a minimum version of a dependency package. Third-party packages may have been updated since the previous release of ROPP; those available on this website in the `Dependency Packages` directory are the latest versions supported by ROPP and the package build scripts. We recommend installing these latest supported versions, but in general it is not necessary to do so. Rebuilding is also recommended if your C or Fortran compiler version has changed since the original build.

It is highly recommended that third-party dependency packages are compiled using the same Fortran compiler being used for the ROPP code. Mixing code at object level, even using different versions of compilers from the same vendor, can cause linking or run-time problems. C code in the dependency packages (there is none in the ROPP package) must be compiled using the `cfortran.h` system or equivalent to ensure calling interface compatibility with the Fortran compiler's object code, which means that the package must be configured for use with a particular Fortran interface, even if the package itself contains no Fortran.

5.3 Environment variables

The build system uses an environment variable `ROPP_ROOT` as the root path for all file installations. By default, this will be set to `$HOME` by the `buildpack` script. Pre-define this variable before installing any 3rd party packages or ROPP modules; for instance:

> `export ROPP_ROOT=/usr/local` (POSIX-standard shell such as `sh`, `ksh`, `bash`)

or

> `setenv ROPP_ROOT /usr/local` (`csh`, `tcsh`)

noting that the user must have appropriate file permissions to write to such a location.

The ROPP BUFR encoder and decoder may be interfaced with either the Met Office 'MetDB' or ECMWF BUFR kernel libraries. A BUFR library is optional, and if neither is present, the encoder and decoder application programs in the ROPP_IO module will not be built. If both libraries are detected, the MetDB one will be used in preference. If a BUFR library is to be used, the appropriate environment variable must be pre-set to the directory path for the run-time tables. For instance, if using the Met Office 'MetDB' library (v2x.y.z):

> `export BUFR_LIBRARY=$ROPP_ROOT/data/bufr`

or for the ECMWF BUFR library:

> `export BUFR_TABLES=$ROPP_ROOT/data/bufr/`

(note the required trailing forward slash in the ECMWF path). Please refer to the documentation that comes with with the BUFR package on how to install this library and related files or simply use the `buildpack` script with a `<package>` name of `mobufr` or `ecbufr`. The appropriate environment variable can be pre-defined so that `buildpack` will install the BUFR run-time files to the desired location (the above

examples show the default paths). These two paths may be the same or different; the two sets of run-time files can co-exist. Building the ROPP_IO module will also install files to this location.

Users may find it useful to define various ROPP environment variables by sourcing a copy, suitably amended for their local installation, of the `setroppenv` script, which can be found in the ROPP home directory.

5.4 Example configuration scripts

The main tarball (and the individual `ropp_build` tarball) contains example build and configure shell mini-scripts for installing not only the ROPP code but also the dependent third-party packages in a compatible way. These files are extracted into the `configure` sub-directory and follow the naming style `<package>_configure_<compiler>_<os>` where:

- `<package>` is the package name (`ropp, netcdf, netcdff, ecbufr, ...`);

- `<compiler>` is the compiler ID (`ifort, nagfor, pgf95, gfortran, ...`); and

- `<os>` is the operating system ID, as output by the `uname` command but entirely in lower case (`linux, cygwin, hp-ux, sunos, aix, ...`).

For example, the mini-script file `netcdf_configure_gfortran_linux` can be used to build the netCDF-4 'core' library with GFortran/gcc on a Linux system. These configure mini-scripts are used by the utility `buildpack` script.

The example configure scripts for specific platforms and compilers may need to be edited for optimal local use, or users may create their own following one of the examples. We would be pleased to receive equivalent scripts if you have built the packages using different compilers and/or platforms and which could be included in future updates. We cannot, however, formally support such scripts since their use is highly dependent on the user's target system and compiler setting preferences and we have no means of testing them ourselves.

5.5 The `buildpack` script

Included in the main package (and in the individual `ropp_build` tarball) is a Bash shell script file `buildpack`. The user need not be using Bash as an interactive shell, but Bash needs to be installed (usually at `/bin/bash`) for the script to work; almost all POSIX-based systems should have Bash installed by default, and for many Linux distros, Bash is the default login shell. On the rare systems that do not have Bash installed, follow the main commands for the desired package installation section of this script.

This script may be used to automate the build of any ROPP module or dependency package in a consistent way, using the above mentioned configure scripts. Summary usage can be obtained by:

```
> buildpack -h
```

This script assumes that all archive files (tarballs) are placed in the same working directory and the configure scripts are in a `configure` sub-directory. Packages will be decompressed and un-tar'ed here and resulting files installed to the `ROPP_ROOT/<compiler>` target directory. In general, to build and install a package, it is only necessary to give the command:

```
> buildpack <package> <compiler> [[NO]CLEAN]
```

where:

- `<package>` is one of the supported ROPP module or package names (`ropp_io, netcdf, mobufr, ecbufr, grib...`); and

- `<compiler>` is the required compiler (`ifort, nagfor, gfortran, xlf95, ...`) which matches an appropriate configure script.

- **[[NO]CLEAN]** is explained below.

- (The **<os>** part of the mini configure script name is detected automatically.)

The main steps of **buildpack** are:

1. perform various checks on the environment, install target and available packages;

2. if no source tree is present (first install or deleted at step 2), unpack the latest archive file (tarball) or clean a pre-existing source directory (optionally skip this step if **NOCLEAN** is present);

3. optionally delete the package source tree to force a clean build (if **CLEAN** is present);

4. run the appropriate mini-configure script, build and install and package test commands.

Since some modules are dependent on other modules, the recommended package building order is:

1. **zlib** (needed for HDF5, although it is likely to be installed already)

2. **hdf5** (needed for netCDF)

3. **netcdf** (the netCDF core libraries, needed for ROPP_IO)

4. **netcdff** (the Fortran interface, needed for ROPP_IO)

5. **??bufr** (optional - either **mobufr** for the Met Office or **ecbufr** for the ECMWF package)

6. **grib** (optional - ECMWF package)

7. **sofa** (optional – IAU package)

8. **ropp_utils** (needed)

9. **ropp_io** (almost certainly needed)

10. **ropp_pp** (optional)

11. **ropp_apps** (optional)

12. **ropp_fm** (optional, but needed for ROPP_1DVAR)

13. **ropp_1dvar** (optional)

With effect from ROPP-9.0, build support has only been provided for the 'full' version of netCDF-4. This implies the need for an HDF5 library. Most ROPP files are in the 'classic' netCDF-3 format, and these can still be read and (by default) written. Full netCDF-4 functionality is only needed to build and run the **eum2ropp** and **eum2bufr** tools, which process EUMETSAT 'grouped' netCDF-4 files.

Note that this Bash script has only been tested under Linux (RHEL6) and Windows 10 (Cygwin) and may need minor modifications to run on other systems. *Users are strongly advised to build the external dependency packages with* **buildpack***, otherwise pre-existing installations may not mesh correctly with the ROPP modules which use them.* See **README.build** for more information.

5.6   Other build scripts

Other shell wrapper scripts **build*_ropp, build_deps** and **build_ropp** are provided which can be used to further automate the build process by calling **buildpack** with a predetermined sequence of

packages or compilers and which save a copy of all screen output to a disk log file. Review and edit to suit your requirements. Using these tools, a complete check out of ROPP from scratch can be effected by running (in order):

```
> buildzlib_ropp <compiler>

> buildhdf5_ropp <compiler>

> buildnetcdf_ropp <compiler>   (This builds netcdf and netcdff.)

> buildmobufr_ropp <compiler> or buildecbufr_ropp <compiler>

> buildgrib_ropp <compiler>

> buildsofa_ropp <compiler>

> build_ropp <compiler>
```

Or, even more quickly:

```
> build_deps <compiler> zlib hdf5 netcdf netcdff [mobufr|ecbufr] grib
                         sofa

> build_ropp <compiler>
```

6. Building ROPP manually

The low-level build sequence is described in the User Guide (see also `README.unix`), but in summary, consists of the common `configure / make / make install` sequence of commands. If not using the supplied `buildpack` script, this section gives some guidance on the manual procedure.

We suggest that a ROPP 'root' directory be created in a suitable place, for example:

```
> cd <directory-of-your-choice>
> mkdir ropp-9.1
> cd ropp-9.1
> export ROPP_ROOT=$PWD
```

though of course `ROPP_ROOT` can be set to any existing directory to which the user has write access.

As provided, the configure scripts will install the package files to:

`ROPP_ROOT/<compiler>/bin`	executable programs
`ROPP_ROOT/<compiler>/lib` `ROPP_ROOT/<compiler>/lib64`	compiled object libraries. Most libraries will install to the `lib` target directory but packages may install to `lib64` on some 64-bit Linux distros
`ROPP_ROOT/<compiler>/include`	Fortran modules (`.mod`), C header (`.h`) files etc.
`ROPP_ROOT/<compiler>/man/man1`	'man' pages
`ROPP_ROOT/<compiler>/etc` `ROPP_ROOT/<compiler>/share`	various other files
`ROPP_ROOT/data`	Compiler-independent data files (e.g. BUFR run-time tables)

where `<compiler>` is an ID (normally the command line name) for a particular Fortran compiler. If only one compiler is to be used, the appropriate configure script can be edited to remove this extra directory layer.

Then copy the required tar files here and unpack, e.g.:

```
> tar -zxvf ropp_build-9.1.tar.gz
> tar -zxvf ropp_utils-9.1.tar.gz
```

Select a suitable configure script from the `configure` sub-directory. E.g. if using the Intel `ifort` command (on our RHEL6 systems this defaults to v12) under Linux this would be `ropp_configure_ifort_linux`. Review/edit this file to suit your local installation directories and compiler flags. In this example, by default, the ROPP libraries, include, F90 module files, etc will be installed into the `ROPP_ROOT/ifort` tree and the configure will expect similar pre-built third-party dependency files to be found here also.

For example, to build the ROPP_UTILS library:

```
> cd ropp_utils-9.1
> ../configure/ropp_configure_ifort_linux
> make
> make install
> make test
```

Repeat for any other ROPP modules you wish to use. If your application only requires (say) the I/O support (e.g. only the BUFR encoder/decoder tools are wanted), then it is not necessary to build the PP, FM or 1DVAR modules. Alternatively, if only the APPS, PP, FM or 1DVAR callable-API routines are to be implemented in a user's own applications, then the IO module (and netCDF support) need not be built. The configure script will then only build those parts of the ROPP code with no dependence on ROPP_IO (i.e. stand-alone tools with netCDF read/write are not built).

Note that to use the ROPP BUFR encoder and decoder, the appropriate environment variable must be pre-set to the desired directory path to install and use the run-time tables – see Sec 5.3.

7.   Testing

All ROPP modules have a basic user-test option; this is not intended to be a comprehensive test of the software, but just to demonstrate that the package has been correctly installed and programs generate data which can be validated against a reference dataset (provided). The tests can be run in each module by:

```
> cd tests
> make test
```

The screen output (and in some cases log files) should be reviewed to check to whether the test(s) were successful or not. Example log files are provided to show the output for a successful test. (Tip:

```
> make test 2>&1 | tee test.log
```

to also save a log file which can be examined at leisure.)

If using the `buildpack` script, testing will be performed automatically if the build process completed successfully.

8. Portability

A prime requirement of ROPP is that it must be portable so as to compile and run under a range of compilers and platforms. Nevertheless, ROPP can only be supported on platforms also supported by the required third-party packages. In practice this means POSIX-compliant systems (Unix, AIX, Linux, MS Windows/Cygwin, SunOS, OS/X...) and ISO-compliant F95 & C compilers. ROPP depends on some F95 features, so older F90-only compilers cannot be supported. In addition, the ROPP code can only be fully tested and validated on platforms and with compilers available to the ROPP Development Team.

Target compiler/platform combinations we aim to support are listed in the ROPP Overview document. However, difficulties have been encountered with certain compilers, mostly due to explicit or implicit bugs in the compilers themselves, or some limitations in their functionality or interpretation of Fortran standards. The investigation of workarounds continues, although it is not our policy to provide different code for different compilers just because certain compilers do not adhere to accepted standards or have internal bugs.

For this release, the OS/compiler combinations shown in Table 5 have passed all of our comprehensive internal testing ('Test Folder' system) and so should work on similar systems elsewhere. Unless noted, this release of ROPP works with the dependency packages listed in Sec 3.4. The RHEL6 platform is a 64-bit OS with 64-bit compilers (unless noted). The GNU **gcc** compiler is used for compiling C code except where noted.

Platform	Compiler Vendor	Compiler ID	Command	Version	Notes
Linux (RHEL6.8) Intel Xeon	Intel	ifort12	`ifort`	12.0.4	GCC v4.4.7
Linux (RHEL6.8) Intel Xeon	Intel	ifort16	`ifort`	16.0.1	GCC v4.4.7
Linux (RHEL6.8) Intel Xeon	Portland Group	pgf15	`pgf15 (aka pgf95)`	15.7-0	GCC v4.4.7 See Note 1
Linux (RHEL6.8) Intel Xeon	NAGWare	nagfor52	`nagfor`	5.2	GCC v4.4.7
Linux (RHEL6.8) Intel Xeon	GNU GFortran	gfortran	`gfortran`	4.4.7	GCC v4.4.7
Linux (RHEL6.8) Intel Xeon	SUN	sunf95	`sunf95`	8.6	SUN `suncc` v5.12.
Linux (Cray/HPC 64-bit) Cray XC40	Intel	ifort15	`ifort`	15.0.1	GCC 4.9.1
Linux (Cray/HPC 64-bit) Cray XC40	Cray	ftn	`ftn`	8.3.4	GCC 4.9.1

Table 5. OS/compiler combinations passing the Test Folder system

For this release, the OS/compiler combinations shown in Table 6 have failed some element of our comprehensive internal testing ('Test Folder' system).

Platform	Compiler Vendor	Compiler ID	Command	Version	Notes
					No known issues

Table 6. OS/compiler combinations which have failed in some part of the Test Folder system

The OS/compiler combinations shown in Table 7 have had problems – not necessarily fatal – on one or more external dependency packages. Naturally, if an external dependency cannot be built, neither can any ROPP tool based on it. As can be seen from the *Notes* column, successful installation of netCDF-Fortran v4.4.3 may take a few attempts.

Platform	Compiler Vendor	Compiler ID	Command	Version	Dependency Package	Notes
Linux (RHEL6.8) Intel Xeon	Intel	ifort	`ifort`	v16.0.1	netCDF-Fortran v4.4.3	Initially fails to recognise `nf_inq_path`; builds OK if this failure is followed by a manual `make install`.
Linux (RHEL6.8) Intel Xeon	NAGWare	nagfor	`nagfor`	v5.2	netCDF-Fortran v4.4.3	Initially fails with an undefined reference to `` `exit_' ``; builds OK if this failure is followed by a manual `make install`.
Linux (RHEL6.8) Intel Xeon	NAGWare	nagfor	`nagfor`	v5.2	ECMWF BUFR 000387	Apparently builds OK, but ropp_io cannot use it because the build system cannot find `bufren` in the `bufr` library.
Linux (RHEL6.8) Intel Xeon	NAGWare	nagfor	`nagfor`	v5.2	ECMWF GRIP_API 1.14.5	Error: Line 4 of same_int_long.f90 declares `integer(8)`, `dimension(2) :: x8 = (/1, 2/)`, and '8' is not a valid integer KIND value for nagfor (unusually). (Compiling with `-kind=byte` doesn't help.)
Linux (RHEL6.8) Intel Xeon	GNU GFortran	gfortran	`gfortran`	v4.4.7	netCDF-Fortran v4.4.3	Initially fails to recognise `nf_endian_big` etc; builds OK if this failure is followed by a manual `make install`.

Table 7. OS/compiler combinations which fail to build an external dependency package

In addition, all dependency code and ROPP modules have been successfully built and passed basic user-tests (as provided in the distribution tarballs) under the OS/compiler combinations shown in Table 8, unless explicitly noted otherwise.

Notes:

1. Since the `g95` compiler development appears to have ceased - the latest Cygwin build is from 2010 and Linux from early 2013 – the `g95/Cygwin` combination is no longer supported. This means that although the mini-script configure is still included in the ROPP-9.1 tarballs, this release has not been tested on `g95/Cygwin` and any problems reported which are confined to this combo will not be investigated. It is recommended that the Cygwin-built `gfortran` be used instead. The `g95/Linux` combination will probably not be supported beyond ROPP-9.1 (depending on results of a user survey).

2. CYGWIN_NT-10.0 Latitude 2.6.1(0.305/5/3) 2016-12-16 11:55 x86_64 Cygwin

3. Due to non-POSIX-style command line syntax and file naming conventions, native Windows compilers such as Intel, Salford and others – though perfectly good compilers – generally do not

work with POSIX-style `configure` scripts (as used by HDF5, netCDF and ROPP) under Cygwin, and are therefore not supported.

Platform	Compiler Vendor	Compiler ID	Command	Version	Notes
Windows 10 (Cygwin 64-bit) Latitude 2.6.1	GNU GFortran	gfortran	`gfortran`	v5.3.0	GCC v5.3.0

Table 8. Additional OS/compiler combinations that have passed the basic user-installation tests

9.   Scientific options

Some of the ROPP tools have 'scientific' options, which allow the code to undertake different processing. These options generally involve the use of different sets of parameters, or different physically-based algorithms. We do not consider 'extra', or 'advanced' options here, such as the forward modelling of the L1 and L2 bending angles, which are still best considered as research tools. Nor do we include elements of run 'configuration' files, such as the maximum observation height used by the 1DVAR retrieval tools, whose 'optimum' values depend strongly on the data under consideration, and are probably best chosen by experimentation or expert guidance. Instead, we focus on those options which general users may be interested in using – or avoiding – but which they may not know to be available, or switched on by default.

The primary 'scientific' options, and the default choices in ROPP9.1, are described in Table 9. If the defaults change in any later release, they will be noted here and in the ROPP Change Log [RD.1].

Ref: SAF/ROM/METO/SRN/ROPP/001
Issue: 9.1
Date: 30 June 2019

ROPP-9.1 Release Notes

EUMETSAT
ROM SAF

Tool	Option	Effect	Impact	Default in ROPP9.1	Comment
ropp_fm_bg2ro_1d	`-comp`	Use non-ideal gas compressibility factors and revised refractivity coefficients [RD.3 IIa]	Beneficially reduces forward modelled refractivities and bending angles by ~ 0.1%.	OFF	Included in `-best` set of options in ROPP9.1
ropp_fm_bg2ro_1d	`-new_op`	Uses improved vertical interpolation scheme in forward model [RD.3 IIa]	Reduction in O-B away from the model levels.	OFF	Included in `-best` set of options in ROPP9.
ropp_fm_bg2ro_1d	`-refrac_only`	Do not generate bending angles [RD.3 IIa]	Only calculate refractivities	OFF	New at ROPP9.0
ropp_fm_bg2ro_1d	`-bangle_only`	Do not generate refractivities [RD.3 IIa]	Only calculate bending angles	OFF	New at ROPP9.0
ropp_fm_bg2ro_1d	`-247L`	Output bangles on 'standard' 247 impact height levels [RD.3 IIa]	Stops output impact heights being derived from the refractivity altitudes	OFF	New at ROPP9.0
ropp_fm_bg2ro_1d	`-best`	Use scientifically 'best' options,={`comp`, `new_op`, `247L`} at ROPP9.1	Combination of component impacts, not necessarily linear.	OFF	New at ROPP9.0. Elements subject to change at each release.
ropp_fm_bg2ro_2d	`-comp`	Use non-ideal gas compressibility factors and revised refractivity coefficients [RD.3 IIa]	Beneficially reduces forward modelled refractivities and bending angles by ~ 0.1%.	OFF	Unlikely to be made default, for backward compatibility
ropp_1dvar_refrac	`-comp`	Use non-ideal gas compressibility factors and revised refractivity coefficients [RD.3 IIb]	Some benefits to retrievals	OFF	Unlikely to be made default, for backward compatibility
ropp_1dvar_refrac	`-new_op`	Uses improved vertical interpolation scheme in forward model [RD.3 IIb]	Some benefits to retrievals	OFF	Unlikely to be made default, for backward compatibility
ropp_1dvar_bangle	`-comp`	Use non-ideal gas compressibility factors and revised refractivity coefficients [RD.3 IIb]	Some benefits to retrievals	OFF	Unlikely to be made default, for backward compatibility
ropp_1dvar_bangle	`-new_op`	Uses improved vertical interpolation scheme in forward model [RD.3 IIb]	Some benefits to retrievals	OFF	Unlikely to be made default, for backward compatibility

Table 9. Scientific options available to the ROPP tools.

10. Known problems

The following issues have arisen during testing and review, but are not considered blockers to this release and will be addressed for a future release.

- Cygwin `g95` is known to fail when building test programs for several dependency libraries. As development of this compiler seems to have ceased, we are no longer supporting this combination. For preference, use `gfortran` on Cygwin (part of the GCC package under `setup.exe`), which builds and runs all tests normally.

- Similar remarks apply to the other failures in building the external dependency packages as detailed above.

- Since ROPP8.0 it has been possible to forward model or carry out 1dvar retrievals with the L1 and L2 bending angles directly. This increases the memory requirements, particularly for ropp_1dvar, whose **R** matrix has four times as many elements as it has for neutral bending angles. Users should therefore be prepared to increase available system memory when using this option.

- Validation testing has revealed occasional difficulties when using the $\log p$ or $\log q$ options with the default 'minropp' minimization scheme in ropp_1dvar. Until these are resolved, we recommend using minropp without the $\log p,q$ options, or the Levenberg-Marquardt minimiser (with or without the $\log p,q$ options). The two minimisers are compared and contrasted in ROM SAF Report 11, which the user should consult for further information.

- Versions of the Intel Fortran compiler from `ifort14` (v14.0.3) onwards show occasional small differences to those of the recommended compiler `ifort12` in the results of their ropp_1dvar retrievals. The root of the problem appears to be optimisation. When ROPP is compiled without optimisation (-O2 → -O0 in the configuration scripts) the differences between the various `ifort(≥14)` compilers vanish. And the unoptimised `ifort14/15/16` is closer to the optimised `ifort12`, although, understandably, it is around twice as slow to run. It therefore appears that the default optimisations in these later versions of `ifort` may be a little over-aggressive. If such matters are important to users – for example, if they wish to maintain consistency in a long timeseries of processing which has already started – then they are recommended to use `ifort12`. Other users are likely to be indifferent to such issues, since the differences between the results obtained with optimised and unoptimised `ifort` compliers are scientifically negligible.

- ROPP9.0 and beyond include algorithms to calculate Planetary Boundary Layer Heights (PBLHs). Beta testing revealed that these estimates can be systematically too high in polar regions, particularly Antarctica. Work is being carried out to understand why. *Until the matter is resolved, users are advised to treat ROPP-derived PBLHs over polar regions with caution.*

- ROPP_PP contains routines to process GRAS 'Raw Sampling' (RS) data (strictly, this is full-sampling closed loop data). While this code works, the pre-processing of GRAS RS data itself is still in development by EUMETSAT, file formats are liable to change, and the ROPP_PP code has not been extensively tested. **The GRAS RS functionality in the ROPP_PP code therefore has only 'unvalidated prototype' status.** (In practice, this data format has been superseded by a newer level 1 product format devised by EUMETSAT, and realised as grouped data in netCDF4 data files. The RS sampling tools are therefore likely to be retired soon.)

- The low frequency SNRs provided by EUMETSAT are known to be too small in noisy parts of the profile (because they are derived by averaging higher frequency SNRs). Until this is fixed, or steps are taken to correct it in ROPP, results based on these data should be used with caution.

- ropp_1dvar can sometimes fail because the time difference between the observation time and the background time exceeds some threshold. This is due to a long-standing problem regarding the definition of the background time. A proper fix will be implemented in ROPP10.0. Until then, users hitting the problem are advised to bypass the check on spatial and temporal proximity entirely (by

setting `genqc_colocation_apply` to `.false.` in the configuration file), or to allow a large difference between the two times (by setting `genqc_max_time_sep` to something huge in the configuration file).

- It is currently not possible to process EUMETSAT level 1a data to acceptable quality with the default settings and algorithms within ROPP (DMI, pers. comm.) Work is in hand to correct this, but until the required modifications are in the official release of ROPP (probably ROPP10.0), users wishing to process such data should seek further advice by contacting the ROM SAF Helpdesk, by the method described at the end of Section 12.

11. Limitations

1. ROPP is being implemented in a phased approach, and not all of the total planned functionality for the package is available in this release. See the ROPP Overview document for more details.

2. Support for previous versions will be limited to problems which can also be reproduced with the latest version. We recommend that users should upgrade to the latest version when released. In general, only the current release, ROPP-9.1, is fully supported. The two immediately preceding releases, ROPP-8.1 and ROPP-9.0, have limited support.

3. ROPP supports writing of more than one RO profile to a netCDF file. It is assumed that the number of data points in the first profile is sufficient to hold all subsequent profiles to be written to the file (this is a fundamental limitation of saving 2D arrays in netCDF). As a consequence, should the first file written to a multifile have no valid samples in the profile (say, all refractivities are invalid and set missing), then that 'Level 2a' section will be written with zero data points. Subsequent profiles having at least one valid sample will then not be correctly saved in the multifile. See the ROPP_IO User Guide for more details.

4. All ROPP stand-alone tools use the ROPP_IO routines to read data from input ROPP format netCDF files. By default, the data are checked against their valid range attributes, and each data level tested for the availability of valid height (or time) values. All input files are therefore required to have valid time, impact parameter or geopotential height information to pass the range checks on reading for Level1a, Level1b and Level2 data respectively. See the ROPP_IO User Guide for more details.

12. And finally...

Data files, documentation, etc., in this release are not definitive and not all final intended functionality may be present, since ROPP is being developed and released in planned stages. The software is not guaranteed to be bug-free, but has been tested with operational GRAS data (Level 1a and 1b from EUMETSAT and ROM SAF Level 2 GRM-01 refractivity products) and also with COSMIC, CHAMP, GRACE-A, FY-3C (and to a limited extent, TerraSAR-X, C/NOFS and SAC-C) occultations. Nevertheless, the code – with the portability and other limitations noted above – has passed a defined set of tests and the complete package has been formally reviewed and approved for general release as software having *operational* status.

Despite all this, we encourage you to carefully check the building and installation, to test the running of the ROPP software and to review the package in its entirety. The ROPP Development team welcomes any feedback if there were any problems, or just to confirm there were no issues.

Updates to the ROPP code to:

- improve the science
- enhance functionality
- fix any reported bugs
- extend portability
- make ROPP easier to build

will be released from time to time as minor or major releases. Registered users will be notified when an update is available. This may be in the form of a complete replacement file, a patch file or – in the case of trivial changes – instructions for local file editing. Implementation of such updates is at the discretion of the user, but support for older releases will become increasingly limited.

All comments on, or questions about, the ROPP software should, in the first instance, be reported via the ROM SAF Helpdesk at **`http://www.romsaf.org`** `> Helpdesk > New enquiry`