EUMETSAT
ROM SAF
RADIO OCCULTATION METEOROLOGY

The Radio Occultation Processing Package (ROPP) Forward Model Module User Guide

Version 11.0

31 December 2021

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)

Ref: SAF/ROM/METO/UG/ROPP/006

Document Author Table

	Name	Function	Date	Comment
Prepared by:	I. Culverwell	ROM SAF Project Team	31 December 2021	
Reviewed by:	O. Lewis	ROM SAF Project Team	31 December 2021	
Approved by:	K. B. Lauritsen	ROM SAF Project Manager	31 December 2021	

Document Change Record

Issue/Revision	Date	By	Description
Version 0.1	01 Nov 2004	CM	Preliminary release for I–RR
Version 0.7	04 Jun 2005	CM	Intermediate release for CPM.
Version 0.8	17 Oct 2005	DO	First beta release (v0.8)
Version 0.9	10 Nov 2006	DO	Second beta release (v0.9)
Version 1.0	01 Mar 2007	DO	First full release (v1.0)
Version 1.1	01 Mar 2008	DO	Updates to first full release (v1.1)
Version 1.2	01 Jul 2008	HL	Updates to first full release (v1.2)
Version 2.0	01 Dec 2008	HL	Second full release (v2.0)
Version 3.0	01 Jun 2009	HL	Third full release (v3.0)
Version 4.0	01 Nov 2009	HL	Fourth full release (v4.0)
Version 4.1	01 Jun 2010	HL	Updates to fourth full release (v4.1)
Version 5.0	14 Jun 2011	IC	Fifth full release (v5.0)
Version 6.0	31 Oct 2011	IC	Sixth full release (v6.0)
Version 6.1	31 Jan 2013	IC	Updates to sixth full release (v6.1)
Version 7.0	31 Jul 2013	IC	Seventh full release (v7.0)
Version 7.1	31 Dec 2013	IC	Updates to seventh full release (v7.1)
Version 8.0	31 Dec 2014	IC	Eighth full release (v8.0)
Version 8.1	31 Dec 2015	IC	Update to eighth full release (v8.1); split from 1D–Var guide
Version 9.0	28 Feb 2017	IC	Ninth full release (v9.0)
Version 9.1	30 Jun 2019	IC	Update to ninth full release (v9.1)
Version 10.0	30 Sep 2020	IC	Tenth full release (v10.0).
Version 11.0	31 Dec 2021	IC	Eleventh full release (v11.0). Changes from ROPP-10.0 in red — see Change Log (SAF/ROM/METO/SRN/ROPP/018) for details.

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 radio occultation (RO) data from the Metop and Metop-SG 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 aid 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

All intellectual property rights of the ROM SAF products belong to EUMETSAT. The use of these products is granted to every interested user, free of charge. If you wish to use these products, EUMETSAT's copyright credit must be shown by displaying the words "copyright (year) EUMETSAT" on each of the products used.

Contents

1 Introduction

1.1 Purpose of this document

This document, the ROPP_FM User Guide ([RD.2e]), describes the forward model module of the Radio Occultation Processing Package (ROPP). The forward models are designed to compute refractivity and bending angle profiles from background atmospheric profiles of pressure, temperature and humidity data. In addition to refractivity, the so-called dry temperature is also computed. Forward model output can be compared with radio occultation observations and used as part of a 1D–Var retrieval or in a 3D–Var or 4D–Var NWP data assimilation system.

1.2 Applicable and reference documents

1.2.1 Applicable documents

The following documents have a direct bearing on the contents of this document.

[AD.1] Proposal for the Third Continuous Development and Operations Phase (ROM SAF CDOP-3) March 2017 – February 2022, as endorsed by Council 7th December 2016

[AD.2] Product Requirements Document (PRD). SAF/GRAS/METO/MGT/PRD/001

[AD.3] ROPP User Licence. SAF/ROM/METO/LIC/ROPP/002

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 Architectural Design Document (ADD). SAF/ROM/METO/ADD/ROPP/001

[RD.2] The ROPP User Guides:

[RD.2a] Overview. SAF/ROM/METO/UG/ROPP/001
[RD.2b] ROPP_IO. SAF/ROM/METO/UG/ROPP/002
[RD.2c] ROPP_PP. SAF/ROM/METO/UG/ROPP/004
[RD.2d] ROPP_APPS. SAF/ROM/METO/UG/ROPP/005
[RD.2e] ROPP_FM. SAF/ROM/METO/UG/ROPP/006
[RD.2f] ROPP_1DVAR. SAF/ROM/METO/UG/ROPP/007
[RD.2g] ROPP_UTILS. SAF/ROM/METO/UG/ROPP/008

1.3 Acronyms and abbreviations

AC	Analysis Correction (NWP assimilation technique)
API	Application Programming Interface
Beidou	Chinese GNSS navigation system. Beidou-2 also known as COMPASS
BG	Background
BUFR	Binary Universal Format for data Representation
CASE	Computer Aided Software Engineering
CDR	Climate Data Record
CF	Climate and Forecasts (CF) Metadata Convention
CGS	Core Ground Segment
CHAMP	Challenging Mini–Satellite Payload
CLIMAP	Climate and Environment Monitoring with GPS–based Atmospheric Profiling (EU)
CMA	Chinese Meteorological Agency
C/NOFS	Communications/Navigation Outage Forecasting System (US)
CODE	Centre for Orbit Determination in Europe
COSMIC	Constellation Observing System for Meteorology, Ionosphere & Climate
DMI	Danish Meteorological Institute
DoD	US Department of Defense
EC	European Community
ECF	Earth–centred, Fixed coordinate system
ECI	Earth–centred, Inertial coordinate system
ECMWF	The European Centre for Medium-Range Weather Forecasts
EGM–96	Earth Gravity Model, 1996. (US DoD)
EOP	Earth Orientation Parameters
EPS	EUMETSAT Polar System
ESA	European Space Agency
ESTEC	European Space Research and Technology Centre (ESA)
EU	European Union
EUMETSAT	European Organisation for the Exploitation of Meteorological Satellites
EUMETCast	EUMETSAT's primary dissemination mechanism for the NRT delivery of satellite data and products
FY-3C/D	GNSS radio occultation receivers (CMA)
GALILEO	European GNSS constellation project (EU)
GCM	General Circulation Model
GFZ	GFZ Helmholtz Centre (Germany)
GLONASS	Global Navigation Satellite System (Russia)
GNOS	GNSS Occultation Sounder (China)
GNSS	Global Navigation Satellite Systems (generic name for GPS, GLONASS, GALILEO and Beidou)
GPL	General Public Licence (GNU)
GPS	Global Positioning System (US)

GPS/MET	GPS Meteorology experiment, onboard Microlab-1 (US)
GPSOS	Global Positioning System Occultation Sensor (NPOESS)
GRACE–A/B	Gravity Recovery and Climate Experiment (US/Germany)
GRACE–FO	GRACE Follow-on experiment (US/Germany)
GRAS	GNSS Receiver for Atmospheric Sounding (onboard Metop)
GUI	Graphical User Interface
GTS	Global Telecommunications System
HIRLAM	High Resolution Limited Area Model
ICDR	Intermediate Climate Data Record
IERS	International Earth Rotation Service
ITRF	International Terrestrial Reference Frame
ITRS	International Terrestrial Reference System
IGS	International GPS Service
ISRO	Indian Space Research Organisation
JPL	Jet Propulsion Laboratory (NASA)
KMA	Korean Meteorological Agency
KOMPSAT–5	GNSS radio occultation receiver (KMA)
LAM	Local Area Model (NWP concept)
LEO	Low Earth Orbited
LGPL	Lesser GPL (*q.v.*)
LOS	Line Of Sight
Megha-Tropiques	Tropical water cycle (and RO) experiment (India/France)
METOP	Meteorological Operational polar satellites (EUMETSAT)
MKS	Meter, Kilogram, Second
MPEF	Meteorological Products Extraction Facility (EUMETSAT)
MSL	Mean Sea Level
N/A	Not Applicable or Not Available
NASA	National Aeronautics and Space Administration (US)
NMS	National Meteorological Service
NOAA	National Oceanic and Atmospheric Administration (US)
NPOESS	National Polar-orbiting Operational Environmental Satellite System (US)
NRT	Near Real Time
NWP	Numerical Weather Prediction
OI	Optimal Interpolation (NWP assimilation technique)
Operational ROM SAF	Team responsible for the handling of GRAS data and the delivery of meteorological products during the operational life of the instrument
PAZ	Spanish Earth Observation Satellite, carrying a Radio Occultation Sounder
PFS	Product Format Specifications
PMSL	Pressure at Mean Sea Level
POD	Precise Orbit Determination
Q/C	Quality Control

RO	Radio Occultation
ROC	Radius Of Curvature
ROM SAF	The EUMETSAT Satellite Application Facility responsible for operational processing of radio occultation data from the Metop satellites. Leading entity is DMI; collaborating entities are UKMO, ECMWF and IEEC.
ROPP	Radio Occultation Processing Package
ROSA	Radio Occultation Sounder for Atmosphere (on OceanSat-2 and Megha-Tropiques)
RMDCN	Regional Meteorological Data Communication Network
SAC–C	Satelite de Applicaciones Cientificas – C
SAF	Satellite Application Facility (EUMETSAT)
SAG	Scientific Advisory Group
SI	Système International (The MKS units system)
TAI	Temps Atomique International (International Atomic Time)
TanDEM–X	German Earth Observation Satellite, carrying a Radio Occultation Sounder
TBC	To Be Confirmed
TBD	To Be Determined
TDB	Temps Dynamique Baricéntrique (Barycentric Dynamical Time)
TDT	Temps Dynamique Terrestre (Terrestrial Dynamical Time)
TDS	True–of–date coordinate system
TerraSAR–X	German Earth Observation Satellite, carrying a Radio Occultation Sounder
TP	Tangent Point
UKMO	United Kingdom Meteorological Office
UML	Unified Modelling Language
UT1	Universal Time-1 (proportional to the rotation angle of the Earth)
UTC	Universal Time Coordinated
VAR	Variational analysis; 1D, 2D, 3D or 4D versions (NWP data assimilation technique)
VT	Valid or Verification Time
WEGC	Wegener Center for Climate and Global Change
WGS–84	World Geodetic System, 1984. (US DoD)
WMO	World Meteorological Organization
WWW	World Weather Watch (WMO)

1.4 Definitions, levels and types

RO data products from the Metop, Metop-SG and Sentinel-6 satellites and RO data from other missions are grouped in *data levels* (Level 0, 1, 2, or 3) and *product types* (NRT, Offline, NTC, 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:

[1] Note that the level definitions differ partly from the WMO definitions: http://www.wmo.int/pages/prog/sat/dataandproducts_en.php.

- **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, SNRs, 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 (ROM SAF Level 2 for EPS); (ii) 150 min after measurement (ROM SAF Level 2 for EPS-SG Global Mission); (iii) 125 min after measurement (ROM SAF Level 2 for EPS-SG Regional Mission); item
- **Offline and NTC products**: Data product delivered from about 5 days to up to 6 months after measurement, depending on the applicable 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 (ICDR) 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].

1.5 Structure of this document

Section 2 briefly describes ROPP and its documentation. Section 3 describes the theory behind the forward modelling of bending angles, refractivity and dry temperature starting from atmospheric temperature, pressure and humidity. Section 4 describes the ROPP 'one-dimensional' forward models — that is, those that result from assuming spherical symmetry in the atmospheric fields. Details are given of the pressure-based (e.g. ECWMF) and geopotential height-based (e.g. Met Office) models from which the ROPP forward model can start. Section 5 describes the ROPP 'two-dimensional' forward model — that is, one that allows horizontal gradients in the atmospheric fields. Section 6 describes how non-ideal gas effects can be included in the 1D and 2D forward models by means of 'compressibility factors'. Finally, Section 7 describes how

[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 (http://ceos.org/meetings/wgclimate-9)).

the non-ionospherically corrected L1 and L2 bending angles can be estimated by means of a simple model ionosphere.

Appendices give brief instructions on how to build ROPP, list the files in the `ropp_fm` module, list the 'extra diagnostic data' that is produced by the various ROPP tools (usually by means of a '`-d`' option), record useful ROPP and other ROM SAF documentation, list the principal authors of ROPP, and state the copyright information that applies to various parts of the code.

2 ROPP

2.1 ROPP introduction

The aim of ROPP is

> *. . . to provide users with a comprehensive software package, containing all necessary functionality to pre-process RO data from Level 1a (Phase), Level 1b (Bending Angle) or Level 2 (Refractivity) files, plus RO-specific components to assist with the assimilation of these data in NWP systems.*

ROPP is a collection of software modules (provided as source code), supporting data files and documentation, which aids users wishing to assimilate radio occultation data into their NWP models. It was originally designed to process data from the GRAS instrument on Metop-A and B, but the software should be adaptable enough to handle data from any other GNSS-LEO radio occultation mission.

The software is distributed in the form of a source code library written in Fortran 90. ROPP is implemented using Fortran modules and derived types, enabling the use of object oriented techniques such as the overloading of routines. The software is split into several modules. Figure 2.1 illustrates the interrelationships between each module. 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 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 release.

ROPP contains support for a generic data format for radio occultation data (`ropp_io`), one- and two-dimensional forward models (`ropp_fm`), routines for the implementation of 1D–Var retrievals, including quality control routines (`ropp_1dvar`), pre-processing and wave optics propagator routines (`ropp_pp`), and various standalone applications (`ropp_apps`). Utility routines used by some or all of the ROPP modules are provided in an additional module (`ropp_utils`). This structure (Figure 2.1) reflects the various degrees of interdependence of the difference ROPP modules. For example, the subroutines and functions in `ropp_io` and `ropp_fm` modules are mutually indepdendent, whereas routines in `ropp_1dvar` depend on `ropp_fm`. Sample standalone implementations of `ropp_pp`, `ropp_fm` and `ropp_1dvar` (which then require `ropp_io` for file interfaces, reading and writing data) are provided with those modules and documented in the relevant User Guides.

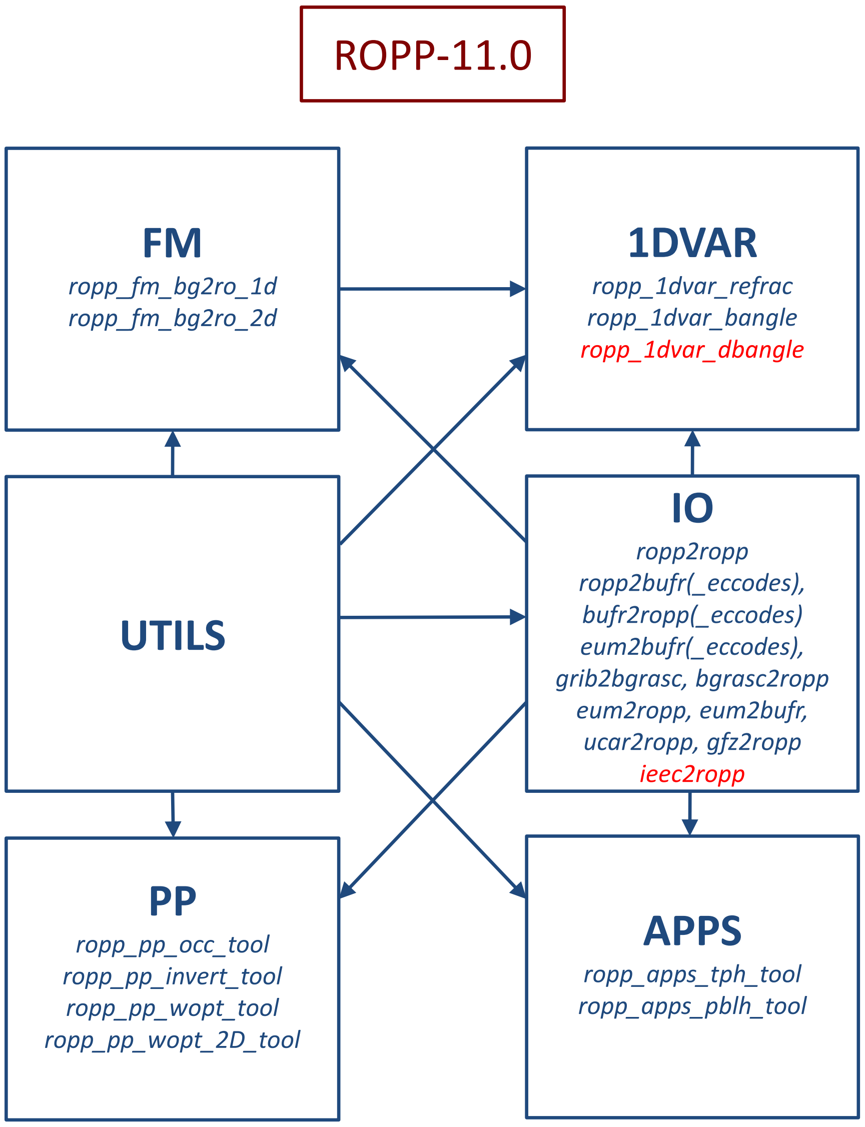


Figure 2.1: The **modules** and *tools* within ROPP-11.0. The module at the head of an arrow depends directly on the module at its tail.

2.2 User documentation

A full list of user documentation is provided in Tables D.1, D.2 and D.4. These documents are available via the ROM SAF website at http://www.romsaf.org.

The ROPP distribution website has a Release Notes file in the root directory which provides a 'Quick Start' guide to the package. This should be read before downloading the package files. Detailed build and install instructions are contained in the release notes of the individual ROPP software modules.

Module-specific user guides for the utilities (ROM SAF, 2021f), input/output (ROM SAF, 2021d), pre-processor (ROM SAF, 2021e), forward model (ROM SAF, 2021c), 1D–Var (ROM SAF, 2021a) and applications (ROM SAF, 2021b) modules describe the algorithms and routines used in those modules. These provide the necessary background and descriptions of the ROPP software for users to process radio occultation data from excess phase to bending angle or refractivity, to forward model background fields to refractivity and bending angle profiles, to simulate the propagation of GNSS radio waves through idealised atmospheric refractivity structures, and to perform 1D–Var retrievals of radio occultation data, as well as advice on how to implement ROPP in their own applications.

More detailed Reference Manuals are also available for each module for users wishing to write their own interfaces to the ROPP routines, or to modify the ROPP code. These are provided in the associated module distribution files.

Further documentation can be downloaded from the ROPP section of the ROM SAF web site http://www.romsaf.org. The full user documentation set is listed in Table D.1.

In addition to these PDF documents, most of the stand-alone application programs have Unix-style 'man page' help files which are installed during the build procedures. All such programs have summary help information which is available by running the command with the `-h` switch.

Any comments on the ROPP software should in the first instance be raised via the ROM SAF Helpdesk at http://www.romsaf.org.

References

ROM SAF, The Radio Occultation Processing Package (ROPP) 1D–Var module User Guide, SAF/ROM/METO/UG/ROPP/007, Version 11.0, 2021a.

ROM SAF, The Radio Occultation Processing Package (ROPP) Applications module User Guide, SAF/ROM/METO/UG/ROPP/005, Version 11.0, 2021b.

ROM SAF, The Radio Occultation Processing Package (ROPP) Forward model module User Guide, SAF/ROM/METO/UG/ROPP/006, Version 11.0, 2021c.

ROM SAF, The Radio Occultation Processing Package (ROPP) Input/Output module User Guide, SAF/ROM/METO/UG/ROPP/002, Version 11.0, 2021d.

ROM SAF, The Radio Occultation Processing Package (ROPP) Pre-processor module User Guide, SAF/ROM/METO/UG/ROPP/004, Version 11.0, 2021e.

ROM SAF, The Radio Occultation Processing Package (ROPP) Utilities module User Guide, SAF/ROM/METO/UG/ROPP/008, Version 11.0, 2021f.

3 Forward models

The ROPP Forward Model `ropp_fm` module provided as part of the ROPP software enables users to relate the bending angle, refractivity and dry temperature derived from GNSS-RO data to meteorological profiles of pressure, temperature and humidity. By using a forward model (or forward operator) information in the space of model variables can be mapped into that of the measured variables, and back, in a consistent manner (Eyre, 1997). A data assimilation system determines increments to NWP fields by comparing the NWP field, mapped into observation space using a forward model, with real observations and minimising the differences between the model and observations and between analysis and background NWP fields.

3.1 Radio occultation geometry

A detailed review of GNSS-RO measurements was provided by Kursinski et al. (1997). The typical radio occultation geometry is sketched in Figure 3.1. A ray passing from a GNSS to a LEO satellite through the atmosphere is refracted as a result of gradients in the refractive index, n. In general, the equations defining the the two-dimensional ray path, in circular polar co-ordinates (r and θ), are given by (page 149 Rodgers, 2000).

$$\begin{aligned}
\frac{dr}{ds} &= \cos\phi \\
\frac{d\theta}{ds} &= \frac{\sin\phi}{r} \\
\frac{d\phi}{ds} &= -\sin\phi\left[\frac{1}{r}+\frac{1}{n}\left(\frac{\partial n}{\partial r}\right)_\theta\right]+\frac{\cos\phi}{nr}\left(\frac{\partial n}{\partial\theta}\right)_r
\end{aligned} \tag{3.1}$$

where s is the distance along the ray-path, n is the refractive index, ϕ is angle between the local radius vector and the tangent to the ray-path. Under the assumption of spherical symmetry, where it is assumed $\left(\frac{\partial n}{\partial\theta}\right)_r = 0$, it can be shown that

$$a = nr\sin\phi = \text{constant} \tag{3.2}$$

along the ray-path. This is also known as Bouguer's formula (e.g., Born and Wolf, 1980). The constant, a, is called the impact parameter, and in the processing of GNSS radio occultation measurements it is assumed that the value of a at the GNSS satellite, a_G, and LEO satellite, a_L, are equal. The impact parameter also determines the height of the tangent point of the ray-path. If the tangent point is defined as that point on the ray where the angle ϕ between the position vector and the ray's tangent equals 90°, we obtain $\sin\phi = 1$, and therefore

$$a = n_t r_t \tag{3.3}$$

with r_t being the distance between the centre of curvature and the tangent point, and n_t being the refractive index at that point. This allows the calculation of the impact parameter from given refractivity

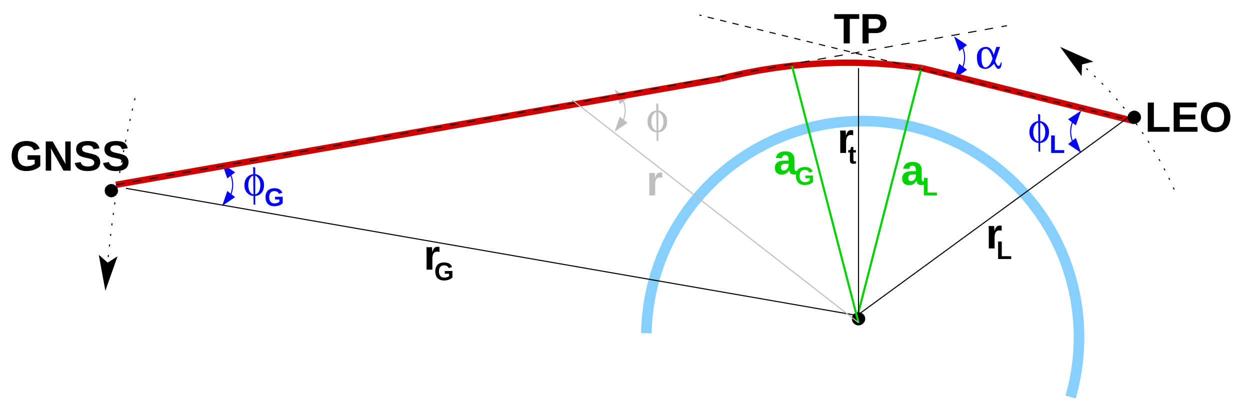


Figure 3.1: Radio occultation geometry; shown are the bending angle α, the GNSS and LEO side impact parameters (a_G and a_L), the GNSS and LEO coordinate vectors ($\mathbf{r}_G$, $\mathbf{r}_L$), the ray path (solid red line) and the satellite side asymptotes of the ray path (dashed). Radius r_t shows the radial distance between the centre of curvature and the ray tangent point.

n and geometric height h:

$$a = (h+R_c)\cdot n(h+R_c) \tag{3.4}$$

where, as indicated, n is evaluated at $(h+R_c)$ and R_c is the radius of curvature of the (assumed ellipsoidal) Earth at the tangent point. (h is the height above this ellipsoid.)

Note that the two satellite side impact parameters a_G and a_L will, in general, not be equal if the spherical symmetry assumption is not met. In addition, the impact parameter a calculated from (3.3) at some hypothetical tangent point is not necessarily the same as satellite side impact parameters in the non–symmetrical case.

3.2 Bending angle

The total amount of refraction is characterised by the bending angle α. This is the angle between the two satellite side ray asymptotes (Figure 3.1). Two-dimensional bending angle forward models in ROPP solve Eqns (3.1). However, the forward modelling can be simplified considerably by assuming spherical symmetry so that a is a constant along the ray path, In this case, it can be shown that $\alpha(a)$ is given by the integral (Fjeldbo et al., 1971; Melbourne et al., 1994; Kursinski et al., 1997)

$$\begin{aligned}\alpha(a) &= -2\int_{r_t}^{\infty} d\alpha = -2a\int_{r_t}^{\infty}\frac{1}{\sqrt{r^2n^2-a^2}}\frac{d\ln(n)}{dr}dr \\ &= -2a\int_{a}^{\infty}\frac{1}{\sqrt{x^2-a^2}}\frac{d\ln(n)}{dx}dx\ .\end{aligned} \tag{3.5}$$

The second expression is obtained by substituting $x = nr$. The negative sign in Eqn (3.5) follows the convention that bending towards the Earth's surface is positive.

The one-dimensional (1D) forward models provided in ROPP solve Eqn (3.5) using the algorithm is described by Healy and Thépaut (2006) and outlined in Chapter 4.

3.3 Refractivity

The amount of bending a ray of radio waves experiences on its way through the atmosphere depends on the local refractive index n of air along the ray path. For convenience, refractivity N is often used instead of the refractive index n. These are related as

$$N = (n-1) \times 10^6 \ . \tag{3.6}$$

At microwave frequencies in the Earth's atmosphere, N varies due to contributions from the dry neutral atmosphere, water vapour, free electrons in the ionosphere and particulates, thus Kursinski et al. (1997):

$$N = \kappa_1 \frac{P_d}{T} + \kappa_2 \frac{e}{T} + \kappa_3 \frac{e}{T^2} + \kappa_4 \frac{n_e}{f^2} + \kappa_5 W \ . \tag{3.7}$$

The first term is the dry neutral atmosphere contribution where P_d is the pressure of the dry air and T is the temperature. The second and third term is the water vapour contribution, where e is the partial water vapour pressure. The fourth ionospheric contribution results from free electrons in the ionosphere where n_e is the electron number density and f is the transmitter frequency of the radio signal. The final term results from scattering, mainly due to cloud droplets where W is the liquid water content.

For radio occultation soundings it is usually assumed that the scattering term is negligible, and that ionospheric effects have been removed during the pre-processing, such as by an ionospheric correction of the bending angles (Vorob'ev and Krasil'nikova, 1994). The refractivity forward models in ROPP therefore only deal with the neutral atmosphere.

Neglecting non-ideal gas effects, a 3-term expression relates refractivity to atmospheric parameters as

$$N = k_1 \frac{P_d}{T} + k_2 \frac{e}{T^2} + k_3 \frac{e}{T} \tag{3.8}$$

(Smith and Weintraub, 1953; Bevis et al., 1994) (Note that κ_1 (Eqn (3.7)) $= k_1$ (Eqn (3.8)), but that $\kappa_2 = k_3$ and $\kappa_3 = k_2$.) Further discussion of the refractivity formulation is provided by Rueger (2002), ROM SAF (2008) and ROM SAF (2009).

In the default ROPP settings we use a three term expression, but set $k_1 = k_3 = 77.60$ N-unit K hPa^{-1}, so it is consistent with the two term expression given in Smith and Weintraub (1953). However, it is easier to introduce non-ideal gas effects into the three term expression (see Chapter 6).

3.4 Dry temperature

The so-called dry temperature is obtained by ignoring the water vapour in Eqn (3.8) such that

$$N = \kappa_1 \frac{P}{T} \ . \tag{3.9}$$

With this assumption, refractivity is proportional to density $\rho = P/RT$, where R is the specific gas constant for dry air. Pressure can be obtained by assuming hydrostatic equilibrium $dP/dz = -\rho g$, where g is the gravitational acceleration.

The ropp_fm module includes an algorithm to compute the temperature with these assumptions. The resulting temperature is denoted *dry temperature* to distinguish it from real temperature. The dry temperature is practically identical to real temperature in the stratosphere and above, where the water vapour in Eqn (3.8) can be neglected, but it is significantly smaller than real temperature in moist regions of the troposphere.

Section 4.9 provides further details.

3.5 Ionospheric bending angles

The bending angles arising from two types of ionospheric model are included in ROPP: a Chapman layer, as described in Section 7, and a 'VaryChap' layer (a generalised Chapman layer), as described in Section 8. The former is activated by the -direct_ion options in the ROPP 1D forward model tool and the 1dvar bending angle retrieval tools. The bending angles calculated for a Chapman layer are calculated and added to the 'neutral' bending angles. The parameters that define the Chapman layer form part of the state vector, and are varied during the retrieval. In the latter case, the purely ionospheric bending angle difference $\alpha_{\mathrm{f2}} - \alpha_{\mathrm{f1}}$ is calculated for a VaryChap model ionosphere. A new differenced bending angle 1dvar retrieval tool uses this to infer the four parameters that define the VaryChap model. These are held in a new State0DFM fortran structure.

3.6 Summary

The ropp_fm module contains a one-dimensional operator to calculate refractivity and dry temperature as a function of geopotential height, $N(Z)$ and $T_{\mathrm{dry}}(Z)$, and both one- and two-dimensional operators to calculate bending angle as function of impact parameter, $\alpha(a)$. The one-dimensional refractivity operator evaluates Eqn (3.8) at the observation heights, and the dry temperature is computed at the same heights using Eqn (3.9) via hydrostatic integration. The one- and two-dimensional bending angles operators solve equations (3.5) and (3.1), respectively.

These forward models all include the following components:

- Mapping background information to pressure, temperature and humidity profiles.
- Integration of the hydrostatic equation to determine the geopotential heights of the model levels

The details of these tasks will depend on the format of the background data, and are they discussed in Chapters 4 and 5.

References

Bevis, M., Businger, S., Chiswell, S., Herring, T. A., Anthes, R. A., Rocken, C., and Ware, R. H., Mapping zenith wet delays onto precipitable water, *J. Appl. Meteor.*, *33*, 379–386, 1994.

Born, M. and Wolf, E., *Principles of Optics*, Pergamon Press, Oxford, 1980.

Eyre, J. R., Variational assimilation of remotely–sensed observations of the atmosphere, *J. Met. Soc. Jap.*, *75*, 331–338, 1997.

Fjeldbo, G., Kliore, G. A., and Eshleman, V. R., The neutral atmosphere of Venus as studied with the Mariner V radio occultation experiments, *Astron. J.*, *76*, 123–140, 1971.

Healy, S. B. and Thépaut, J.-N., Assimilation experiments with CHAMP GPS radio occultation measurements, *Quart. J. Roy. Meteorol. Soc.*, *132*, 605–623, 2006.

Kursinski, E. R., Hajj, G. A., Schofield, J. T., Linfield, R. P., and Hardy, K. R., Observing earth's atmosphere with radio occultation measurements using the Global Positioning System, *J. Geophys. Res.*, *102*, 23.429–23.465, 1997.

Melbourne, W. G., Davis, E. S., Duncan, C. B., Hajj, G. A., Hardy, K. R., Kursinski, E. R., Meehan, T. K., and Young, L. E., The application of spaceborne GPS to atmospheric limb sounding and global change monitoring, Publication 94–18, Jet Propulsion Laboratory, Pasadena, Calif., 1994.

Rodgers, C. D., *Inverse methods for atmospheric sounding: Theory and practice*, World Scientific Publishing, Singapore, New Jersey, London, Hong Kong, 2000.

Rueger, J. M., Refractive index formulae for electronic distance measurement with radio and millimetre waves, Unisurv Report S-68. School of Surveying and Spatial Information Systems, University of New South Wales, [Summary at http://www.fig.net/pub/fig_2002/Js28/JS28_rueger.pdf], 2002.

ROM SAF, Refractivity calculations in ROPP, SAF/GRAS/METO/REP/GSR/005, 2008.

ROM SAF, Refractivity coefficients used in the assimilation of GPS radio occultation measurements, SAF/GRAS/METO/REP/GSR/009, 2009.

Smith, E. K. and Weintraub, S., The constants in the equation for atmospheric refractivity index at radio frequencies, in *Proc. IRE*, vol. 41, pp. 1035–1037, 1953.

Vorob'ev, V. V. and Krasil'nikova, T. G., Estimation of the accuracy of the atmospheric refractive index recovery from doppler shift measurements at frequencies used in the NAVSTAR system, *USSR Phys. Atmos. Ocean, Engl. Transl.*, *29*, 602–609, 1994.

4 One-dimensional forward models

The ROPP forward model module (`ropp_fm`) includes routines to compute one-dimensional profiles of refractivity (`ropp_fm_refrac_1d`), dry temperature (`ropp_fm_tdry_1d`), and bending angle (`ropp_fm_bangle_1d`) from one-dimensional background pressure, temperature and humidity data. This is necessary for profile retrievals and the assimilation of radio occultation observations in NWP.

Figure 4.1 shows example refractivity and bending angle profiles computed from model background profiles of pressure, temperature and specific humidity. The results are computed for specified observation heights, enabling direct comparison with refractivity or bending angle data obtained by radio occultation in a data assimilation system.

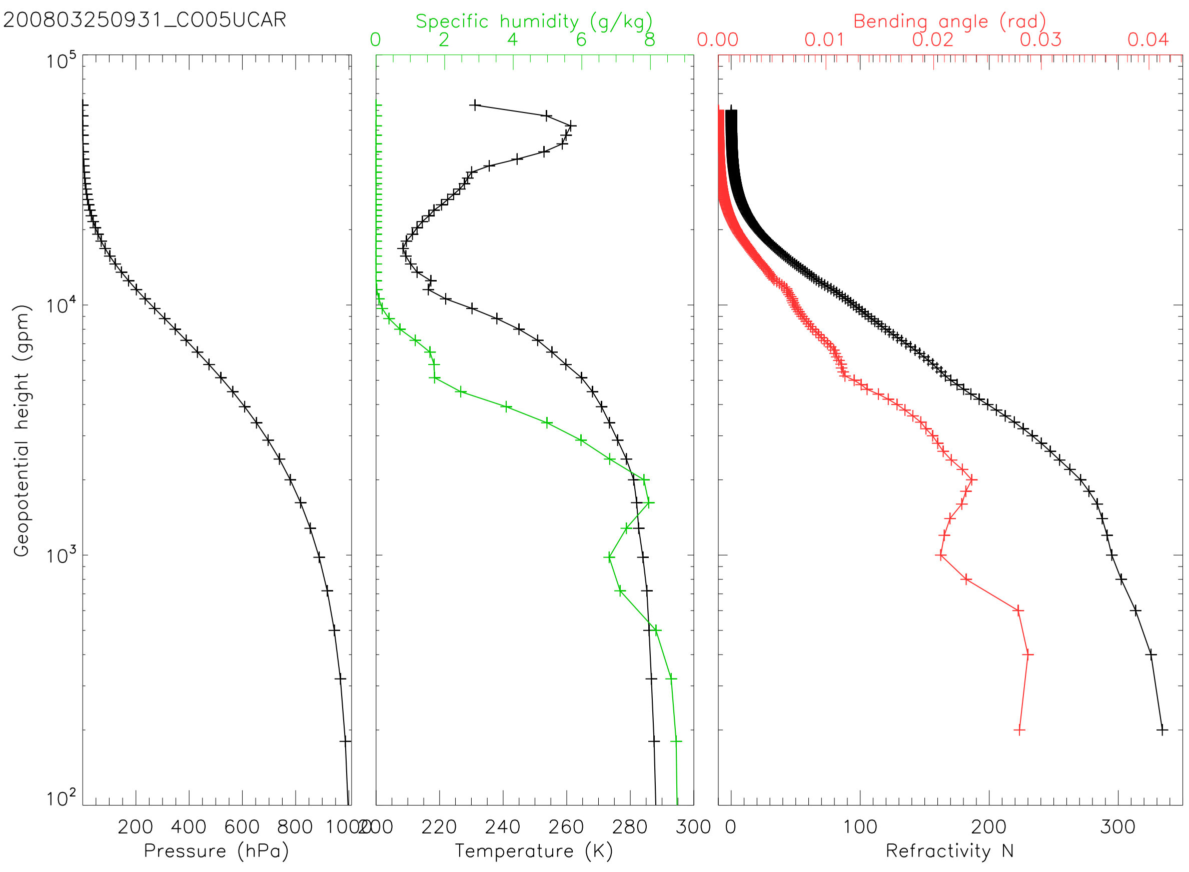


Figure 4.1: Example refractivity and bending angle profiles computed using the model pressure, temperature and specific humidity profiles plotted. Symbols show the vertical resolution of the input model data and the output observation heights.

4.1 ROPP forward model tool

A stand-alone tool `ropp_fm_bg2ro_1d` is provided in `ropp_fm` as an illustration of how the `ropp_fm` routines can be implemented to derive profiles of refractivity, dry temperature, and bending angle from background meteorological data. Figure 4.2 shows how the `ropp_fm` routines are integrated in the `ropp_fm_bg2ro_1d` code.

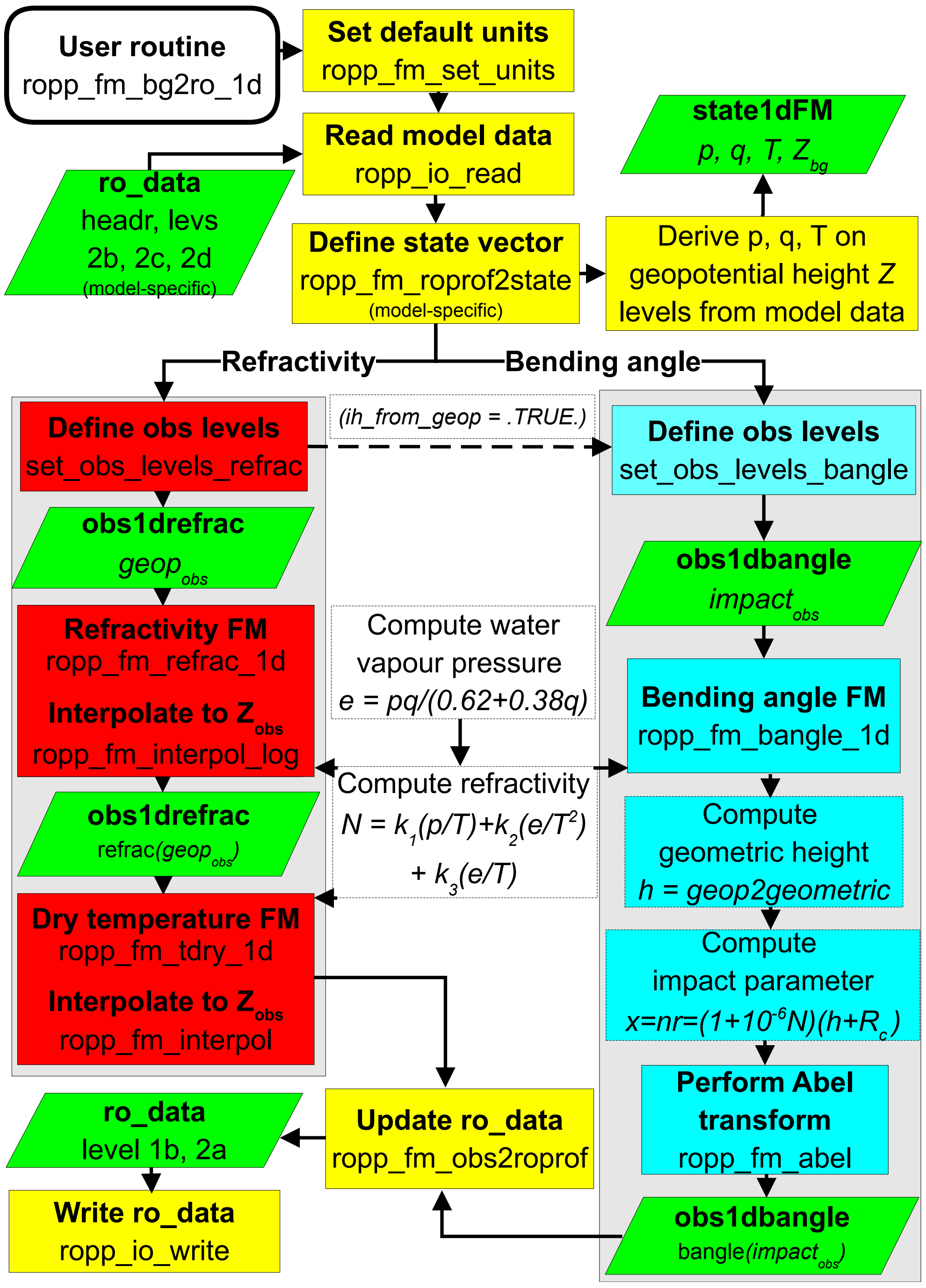


Figure 4.2: Flow chart illustrating calling tree of the ROPP forward model to compute refractivity, dry temperature, and bending angle profiles from input background model data.

4.1.1 Implementation

The ropp_fm_bg2ro_1d tool is run using the command

```
ropp_fm_bg2ro_1d <inputdatafile> -o <outputfile>
```

where <inputdatafile> is a ROPP netCDF file (ROM SAF, 2021b) containing the input model data and <outputfile> will contain the forward modelled refractivity, dry temperature, and bending angle profiles on pre-defined observation levels, also as a ROPP netCDF file. The output levels can be taken from an auxiliary 'levels file', or they can be the 'standard' 247 impact heights, or the user can specify a uniformly spaced set of refractivity levels (same for refractivity and dry temperature) and/or impact heights, whose minimum height, maximum height and number of levels is under the user's control. The default is 300 uniformly spaced refractivity geopotentials between 200 gpm and 60 000 gpm, and 300 impact heights corresponding to them.

The input file contains header, level 2b, level 2c and level 2d data, depending whether on the vertical coordinates of the source data. See Secs 4.5 and 4.6 for details. We would advise users to examine the example test files ropp_fm/data/bgr20090401_000601_M02_1200337800_N0007_XXXX.nc (pressure-based, ECMWF levels) or ropp_1dvar/data/IT-1DVAR-04_b.nc (height-based, UKMO levels), which are contained in the ROPP distribution, to see precisely what input data are needed. The output file contains level 2a fields (refractivity, dry temperature etc) on user-selectable levels, for which the geopotential heights (geop_refrac) and geometric heights (alt_refrac) are provided. The output file also contains level 1b fields (bending angles etc) on user-selectable levels, for which the impact parameters (impact) are provided.

If the input file is a multi-file containing more than one model profile, or more than one input file is specified, ropp_fm_bg2ro_1d computes the forward model for each profile in turn and an output file is generated containing all the output profiles.

These are the command line options available for the `ropp_fm_bg2ro_1d` tool:

`-o <outputfile>`	ROPP netCDF file for output retrieved profiles
`-l <levels_file>`	optional name of 'observation' file holding desired output levels
`-c <config_file>`	optional name of file containing the configuration namelist (overridden by command line options)
`-f`	forward model only, do not calculate gradients
`-use_logp`	use log(pressure) in forward model
`-use_logq`	use log(specific humidity) in forward model
`-comp`	use non-ideal gas compressibility options in forward model
`-check_qsat`	include check against saturation in forward model
`-nocheck_qmin`	do not include check against negative humidities in forward model
`-new_op`	use alternative (new) refractivity interpolation
`-direct_ion`	model L1 and L2 bending angles directly
`-247L`	output bending angles on standard 247 impact heights
`-best`	use the currently advised 'best' options
`-refrac_only`	do not generate bending angles
`-zmin <geop_min>`	minimum refractivity geopotential (gpm)
`-zmax <geop_max>`	maximum refractivity geopotential (gpm)
`-nz <n_geop>`	number of uniformly spaced refractivity geopotentials
`-bangle_only`	do not generate refractivities
`-ihmin <ih_min>`	minimum impact height (m)
`-ihmax <ih_max>`	maximum impact height (m)
`-nih <n_ih>`	number of uniformly spaced impact heights
`-d`	output additional diagnostic information (VerboseMode)
`-h`	give help menu
`-v`	output version information

The default options are given in the distributed configuration file `ropp_fm/config/default_fm.nml` (a Fortran 95 namelist), which is quoted here for reference:

```
! ========================================================
! Namelist of ROPP forward model configuration parameters
! ========================================================
&config_fm
!
! Refractivity options
! --------------------
refrac_wanted = .TRUE.,   ! Refractivities desired
geop_min = 200.0,         ! Minimum refrac geopotential (uniformly spaced)
geop_max = 60000.0,       ! Maximum refrac geopotential (uniformly spaced)
n_geop = 300,             ! No. of geopotential levels (uniformly spaced)
!
! Bending angle options
! ---------------------
bangle_wanted = .TRUE.,  ! Bending angles desired
use_247L = .FALSE.,      ! Use 'standard' 247 bending angle levels,
                         ! held in ropp_fm/common/ropp_fm_levels.f90
ih_from_geop = .TRUE.,   ! Derive impact heights from refrac geopotentials
                         ! (Not applied if use_247L = .TRUE.)
ih_min = 2000.0,         ! Minimum impact height (uniformly spaced)
ih_max = 60000.0,        ! Maximum impact height (uniformly spaced)
n_ih   = 291,            ! No. of impact heights (uniformly spaced)
!
! Levels file
! -----------
lfile = '',              ! Levels file, from which the level 1b
                         ! and/or level 2a output levels are to be drawn
! General options
! ---------------
calc_grad  = .TRUE.,     ! Calculate Jacobians drefrac, bangle/dstate?
use_logp   = .FALSE.,    ! Use log(pressure) in FM?
use_logq   = .FALSE.,    ! Use log(specific humidity) in FM?
compress   = .FALSE.,    ! Use non-ideal gas compressibility factors?
check_qsat = .FALSE.,    ! Forbid supersaturation?
check_qmin = .TRUE.,     ! Forbid superdryness?
new_op     = .FALSE.,    ! Use new interpolation scheme?
direct_ion = .FALSE.,    ! Model L1 and L2 bangles directly?
!
! Mandatory end of namelist delimiter
! -----------------------------------
/
```

Another namelist, `ropp_fm/config/best_fm.nml` is included in the distribution. This includes the currently advised 'best' options — that is, those that use the latest, validated, code developments and/or the most useful configuration specifications (eg output levels). The same options can be obtained by use of the -best option to `ropp_fm_bg2ro_1d`. At ROPP-9.0, the 'best' options are:

```
use_247L = .TRUE.,       ! Use 'standard' 247 bending angle levels,
ih_from_geop = .FALSE.,  ! Derive impact heights from refrac geopotentials.
calc_grad  = .FALSE.,    ! Calculate Jacobians d{refrac, bangle}/d{state}?
compress   = .TRUE.,     ! Use non-ideal gas compressibility factors?
new_op     = .TRUE.,     ! Use new interpolation scheme?
```

as can also be seen by direct examination of `ropp_fm/config/best_fm.nml`.

4.1.2 Code organisation

Figure 4.2 shows how the `ropp_fm_bg2ro_1d` tool is composed of the following stages:

- **Input data access and transformation to a generic state vector**

  Setup the input data arrays, read the input data and define a state vector structure `State1dFM` containing the background pressure p, temperature T and humidity q data as a function of geopotential height Z. If `check_qsat` is in force, q is limited by the saturation value at that temperature and pressure; otherwise, by default, supersaturation is allowed. See Section 4.4. If `nocheck_qmin` is in force, q is allowed to be negative; otherwise, by default, such 'superdryness' is forbidden.

- **Define the observation levels for which refractivity, dry temperature, and bending angles are to be calculated**

  There is no longer any requirement for the refractivity levels, on geopotential heights, to correspond to the banding angle levels, on impact heights. (Indeed, there is no need for both observation types to be forward modelled anyway, although this remains the default.) Instead, the various output levels are determined as follows.

  #### Level 2a fields

  The choice of the geopotential heights for the refractivities (and dry temperatures) is made in the following order (later options overriding earlier ones):

  1. The default values in the code, which match those in the namelist above, namely: 300 uniformly spaced levels between 200 gpm and 60 000 gpm;
  2. Any settings in a user-specified configuration namelist, if specified;
  3. Any specifications given at the command line (see above);
  4. The level 2a geopotential heights in the input file, if they exist;
  5. The level 2a geopotential heights in the 'levels' file (ie, the one specified by the `-l` option), if it is given and if they exist.

  #### Level 1b fields

  The choice of the impact heights for the bending angles is made in the following order (later options overriding earlier ones):

  1. The default values in the code, which match those in the namelist above. Since these include `ih_from_geop = .TRUE.`, it follows that the impact heights correspond to the refractivity geopotentials, which are derived first;
  2. Any settings in a user-specified configuration namelist, if specified;
  3. Any specifications given at the command line (see above);

4. If `use_247L = .TRUE.`, the 'standard' 247 impact heights, given in `ropp_fm/common/ropp_fm_levels.f90`, are used;
5. The level 1b impact parameters in the input file, if they exist;
6. The level 1b impact parameters in the 'levels' file (ie, the one specified by the `-l` option), if it is given and if they exist.

The levels in the observation vector structures `Obs1dRefrac` and/or `Obs1dBangle` are calculated in the subroutines `set_obs_levels_refrac`, `set_obs_levels_bangle` (if deriving impact heights from geopotential levels) or `set_obs_levels_bangle2` (if using 247 levels, or uniformly spaced impact heights).

- **Compute refractivity profile**

  `ropp_fm_refrac_1d` is the main refractivity forward model routine to compute refractivity N as a function of geopotential height Z. See Section 4.8.

- **Compute dry temperature profile**

  `ropp_fm_tdry_1d` is the main dry temperature forward model routine to compute dry temperature T_{dry} as a function of geopotential height Z. See Section 4.9.

- **Compute bending angle profile**

  `ropp_fm_bangle_1d` is the main bending angle forward model routine to compute bending angle α as a function of impact parameter a. See Section 4.10.

- **Write results to generic RO data structure and output file**

4.2 Data types

The observation state vector `y` and background state vector `x` are represented within ROPP by Fortran 90 derived types (or structures). These data structures are defined in `ropp_fm_types`.

Note that the ROPP forward model assumes data are in ascending height order. The input data are checked and reordered as part of the stand-alone tool processing (see Section 4.4). If required for 1D–Var, users must ensure that the error correlations used are appropriate for background and observation data in ascending height order.

4.2.1 Observation vector: `Obs1dRefrac`, `Obs1DBangle`

Refractivity and bending angle profiles are represented by the `Obs1dRefrac` and `Obs1dBangle` data structures respectively. Their elements are listed in Table 4.1.

For example, a structure holding data of an observed (or modelled) refractivity profile can be declared by

```
  USE ropp_fm
  TYPE(Obs1dRefrac) :: y
```

Refractivity observations are stored in the `y%refrac` element along with the corresponding geopotential heights `y%geop`. Similarly, a bending angle profile can be declared as `type(Obs1dBangle)` with bending angles stored in element `y%bangle` and the corresponding impact parameter values in `y%impact`. Dry temperature profiles are not yet included in the `Obs1dRefrac` structure.

The structure element `y%weights` is used by 1D–Var quality control routines provided by ROPP. Values range between 0 (data point rejected by QC) and 1 (data point passed by QC). This information can be used to downweight particular observation data points during the cost function minimisation of a variational analysis. The error covariance element `y%cov` is also required by the `ropp_1dvar` module. (See Secs 4.4 and 5.4 of the ROPP 1D–Var User Guide (2021a)).

The observation vector for bending angles also contains information on the surface gravity `y%g_sfc` and effective radius of Earth `y%r_earth`. The local radius of curvature `y%r_curve` is also included together with the undulation `y%undulation`, which is defined as the height of the EGM96 (NASA/NIMA) geoid above the WGS84 (NGA/WGS84) reference ellipsoid at the occultation point. If a height `h_Geoid` is expressed with respect to the geoid, then height `h_Ellipsoid` with respect to the ellipsoid is given by

$$\texttt{h\_Ellipsoid} = \texttt{h\_Geoid} + \texttt{y\%undulation} \tag{4.1}$$

4.2.2 State vector: `State1dFM`

Background meteorological data are represented by the `State1dFM` data structure. Its elements are listed in Table 4.2. Vertical profiles of temperature `x%temp`, pressure `x%pres` and specific humidity `x%shum` are stored together with the corresponding geopotential height levels `x%geop`. The number of background vertical levels is stored in the `x%n_lev` element. A structure holding background data can then be declared as

```
  USE ropp_fm
  TYPE(State1dFM) :: x
```

The state vector `x%state` is required for the `ropp_1dvar` processing. Its contents depends on the details of the background data. Although it is not directly used in `ropp_fm` processing its elements are updated using background-specific mapping between the state vector and conventional meteorological variables (as required if `ropp_fm` were implemented as part of `ropp_1dvar`). Further details are provided in Sec 4.4 and Secs 4 and 5 of ROPP 1D–Var UG. Elements `x%ak` and `x%bk` are required for this mapping if `State1dFM` is holding a state vector using hybrid vertical levels (e.g. ECMWF) as described in Section 4.5. The error covariance element `x%cov` is required by the `ropp_1dvar` module. (See Secs 4.5 and 5.5 of the ROPP 1DVAR UG (2021a.)

Obs1drefrac			
Structure element	Description	Typical range	Units
...%refrac	Refractivity values	0 to 500	N-units
...%geop	Geopotential height	-1000 to 10000	gpm
...%weights	Observation weights	0 to 1	
...%lon	Longitude	-180 to +180	° E
...%lat	Latitude	-90 to +90	° N
...%time	Time	–	Jul sec
...%cov	Error covariance for refractivity	0 to 10	N-units
...%obs_ok	Flag to specify observations ok	.FALSE. or .TRUE.	
...%cov_ok	Flag to specify error covariance ok	.FALSE. or .TRUE.	
Obs1dbangle			
Structure element	Description	Typical range	Units
...%bangle	Bending angle	0.0001 to 0.01	rad
...%impact	Impact parameter	(6.2 to 6.6) $\times 10^6$	m
...%weights	Observation weights	0 to 1	
...%rtan	Tangent point radius	(6.2 to 6.6) $\times 10^6$	m
...%a_path	$nr \sin\phi$ at ray endpoints	(6.2 to 6.6) $\times 10^6$	m
...%lon	Longitude	-180 to +180	° E
...%lat	Latitude	-90 to +90	° N
...%time	Time	–	Jul sec
...%g_sfc	Gravity at the surface	~ 9.8	ms^{-2}
...%r_earth	Effective earth radius	(6.2 to 6.6) $\times 10^6$	m
...%r_curve	Radius of curvature	(6.2 to 6.6) $\times 10^6$	m
...%r_leo	Radius of LEO orbit (nominal)	(6.7 to 8.0) $\times 10^6$	m
...%r_gns	Radius of GNSS orbit (nominal)	(25 to 45) $\times 10^6$	m
...%f1	Frequency of first bending angle	(1.0 to 2.0) $\times 10^9$	Hz
...%f2	Frequency of second bending angle	(1.0 to 2.0) $\times 10^9$	Hz
...%undulation	Undulation	-150 to +150	m
...%azimuth	Azimuthal angle	-180 to +180	° N
...%cov	Error covariance for bending angle	0 to 0.01	rad
...%obs_ok	Flag to specify observations ok	.FALSE. or .TRUE.	
...%cov_ok	Flag to specify error covariance ok	.FALSE. or .TRUE.	
...%n_L1	No. of L1 bending angles	100 to 300	
...%diff_bangle	Working on f2–f1?	.FALSE. or .TRUE.	

Table 4.1: Elements of the Obs1dBangle and the Obs1dRefrac observation vector structure. (Note that f1 and f2 are not necessarily equal to L1 and L2.)

State1dFM			
Structure element	Description	Typical range	Units
`...%state`	State vector data		
`...%temp`	Temperature	150 to 350	K
`...%shum`	Specific humidity	0 to 0.05	kg/kg
`...%pres`	Pressure	10 to 110000	Pa[a]
`...%geop`	Geopotential height	-1000 to 100000	gpm
`...%ak`	ECMWF-specific level coefficient	0 to 200000	Pa[a]
`...%bk`	ECMWF-specific level coefficient	0 to 2	
`...%n_lev`	Number of model levels	1 to 100	
`...%lon`	Longitude	-180 to +180	° E
`...%lat`	Latitude	-90 to +90	° N
`...%time`	Time	–	Jul sec
`...%cov`	Error covariance for state vector	0 to 2500	
`...%state_ok`	Flag to specify state vector ok	.FALSE. or .TRUE.	
`...%cov_ok`	Flag to specify error covariance ok	.FALSE. or .TRUE.	
`...%use_logp`	Flag to specify use log(p) in state	.FALSE. or .TRUE.	
`...%use_logq`	Flag to specify use log(q) in state	.FALSE. or .TRUE.	
`...%non_ideal`	Non-ideal gas flag	.FALSE. or .TRUE.	
`...%check_qsat`	Check against saturation	.FALSE. or .TRUE.	
`...%check_qmin`	Check against dryness	.FALSE. or .TRUE.	
`...%new_ref_op`	Alternative refractivity interpolation	.FALSE. or .TRUE.	
`...%new_bangle_op`	Alternative refractivity interpolation in 1d bending angle operator	.FALSE. or .TRUE.	
`...%direct_ion`	Direct modelling of L1 and L2 bending angles	.FALSE. or .TRUE.	
`...%Ne_max`	Max Chapman layer electron density	-10^{15} to $+10^{15}$	m^{-3}
`...%H_peak`	Height of Chapman layer peak	0 to 10^6	m
`...%H_width`	Width of Chapman layer	0 to 10^6	m

[a] Note that pressure variables in `State1dFM` are in units of Pa while standard units in the `ROprof` structure is hPa. See ROM SAF (2021b) for details. Users must specify pressure variables in Pa or ensure unit conversion is carried out before performing `ropp_fm` calculations if using `ropp_io` for reading input data. Routine `ropp_fm_set_units` is provided for this task.

Table 4.2: Elements of the State1dFM vector structure

4.2.3 State vector: `State0dFM`

Background ionospheric data are held in the `State0dFM` data structure. Its elements are listed in Table 4.3.

```
USE ropp_fm
TYPE(State0dFM) :: x
```

The state vector `x%state` is required for the `ropp_1dvar` processing. It contains parameters of the VaryChap electron density model that is used when 1dvar retrievals are carried out on the f2 – f1 differenced bending angles. The error covariance element `x%cov` is required by the `ropp_1dvar` module.

State0dFM			
Structure element	Description	Typical range	Units
...%state	0D state vector data		
...%n_lay	Number of VaryChap layers	1 to 5	
...%lon	Longitude	-180 to +180	° E
...%lat	Latitude	-90 to +90	° N
...%time	Time	–	Jul sec
...%cov	Error covariance for state vector	0 to 10^{15}	
...%state_ok	Flag to specify state vector ok	.FALSE. or .TRUE.	
...%cov_ok	Flag to specify error covariance ok	.FALSE. or .TRUE.	
...%ne_peak	Max VaryChap layer electron density	10^{10} to 10^{15}	m^{-3}
...%r_peak	Height of VaryChap layer peak	(50 to 1000) $\times 10^3$	m
...%h_zero	Scale height of VaryChap layer	(10 to 1000) $\times 10^3$	m
...%h_grad	Gradient of scale height of VaryChap layer	0 to 1	-

Table 4.3: Elements of the State0dFM vector structure

4.3 Defining default units: ropp_fm_set_units

The units listed in Tables 4.1 and 4.2 are the assumed default units for each variable in the ropp_fm module. Note these may differ from the units specified in a ROPP-format input file. Unit conversion together with transformation of the corresponding valid range parameters is performed before any other ropp_fm processing by calling subroutine ropp_fm_set_units. This utilises the ropp_io unitconvert library functions.

```
USE ropp_io
USE ropp_fm
TYPE(ROprof) :: ro_data
CALL ropp_fm_set_units(ro_data)
```

4.4 Defining the state vector: ropp_fm_roprof2state

The ropp_io module routine ropp_io_read reads a single profile of model background data from a netCDF ROPP format input file and fills the elements of the generic ROPP data structure type ROprof (ROM SAF, 2021b).

```
USE ropp_io
TYPE(ROprof) :: ro_data
CALL ropp_io_read(ro_data, inputfilename, rec=iprofile)
```

The relevant background model data are copied to the state vector x of data type State1dFM by calling the routine ropp_fm_roprof2state.

```
USE ropp_io
USE ropp_fm
TYPE(ROprof)    :: ro_data
TYPE(State1dFM) :: x
CALL ropp_fm_roprof2state(ro_data, x)
```

A number of checks on the input data are also conducted at this stage. The state vector `x%state_ok` flag is set to false if any test fails.

- Check longitude and latitude for background data are within range,
  set `x%lon = ro_data%georef%lon`
  set `x%lat = ro_data%georef%lat`
- Check that data arrays are in ascending order. Note that `ropp_fm` calculations assume that data array index 1 is closest to the surface. The input data are checked and reordered as required using
  `call ropp_io_ascend(ro_data)`
- Check that `ro_data` contains free atmosphere data (level 2b data)
  set `x%n_lev = ro_data%Lev2b%Npoints`

The state vector is required to contain temperature, specific humidity and pressure data as a function of geopotential height (Table 4.2). These data might be read in directly from the input file if the user provides the background data in this form. It is more likely however that elements of the state vector need to be calculated from the available background data provided by a user and the exact specification of the vertical level representation adopted in the NWP model. The type of model data contained in the input file is specified by the input variable `ro_data%Lev2d%level_type`. `ropp_fm` contains routines to derive the required generic state vector using background data from a NWP model where the vertical coordinate is based on pressure levels (e.g. ECMWF, Section 4.5) or on geopotential height (e.g. Met Office, Section 4.6). Users may prefer to implement their own functions to cater for the specific background vertical level type and available data provided.

Elements of the error covariance matrix `x%cov` used in `ropp_1dvar` are initialised in `ropp_fm_roprof2state` by setting its diagonal elements to the (square of the) standard deviation associated with each corresponding meteorological variable. These values should be specified in the input RO file (See Secs 4 and 5 of the ROPP 1D–Var UG.)

4.5 Hybrid sigma vertical levels - ECMWF model type: `ropp_fm_state2state_ecmwf`

For `ropp_fm` applications using background data from a model where the vertical coordinate is pressure-based (e.g. ECMWF), it is necessary to compute the pressure p on each model level. The geopotential height Z of each model level can then be calculated using the hydrostatic equation

$$Z(p) = Z(p_{\text{sfc}}) - \frac{R_{\text{dry}}}{g_{\text{wmo}}} \int_{p_{\text{sfc}}}^{p} T_v \, \mathrm{d} \ln p \tag{4.2}$$

where $Z(p_{\rm sfc})$ is the surface geopotential height, $R_{\rm dry}$ is the gas constant for dry air, $g_{\rm wmo}$ is the reference gravitational acceleration and T_v is the virtual temperature. This conversion from model-specific to generic ropp_fm variables is performed using a method which is consistent with the model dynamics in ropp_fm_state2state_ecmwf.

```
    USE ropp_fm
    TYPE(State1dFM) :: x
    CALL ropp_fm_state2state_ecmwf(x)
```

4.5.1 Vertical grid structure

Figure 4.3 shows a schematic illustration of the vertical model level and data structure of the ECMWF Integrated Forecasting System (IFS) (ECMWF, 2006). The atmosphere is divided into N layers, on which values of the temperature x%temp=T and humidity x%shum=q are defined. The levels are defined by the pressure at the interfaces (half-levels) between them. The pressure on each half-level is not stored directly, but is defined using the surface pressure $p_{\rm sfc}$ and level coefficients x%ak=$a_{k+1/2}$, x%bk=$b_{k+1/2}$ stored on each half level as

$$p_{k+1/2} = a_{k+1/2} + b_{k+1/2} p_{\rm sfc} \tag{4.3}$$

for $0 \le k \le N$. The pressure on each full-level is then calculated as

$$\texttt{x\%pres} = p_k = \frac{1}{2}(p_{k-1/2} + p_{k+1/2}) \tag{4.4}$$

for $1 \le k \le N$. The level coefficients are provided if available in the ROprof data structure as Level2d data (ROM SAF, 2021b).

The geopotential height of each full level x%geop=Z_k is computed following the ECMWF (2006) algorithms (Simmons and Burridge, 1981). The discrete version of the hydrostatic equation (Eqn 4.2) enables the geopotential height on a half-level to be computed as

$$Z_{k+1/2} = Z_{\rm sfc} + \sum_{j=1}^{k} \frac{R_{\rm dry}(T_v)_j}{g_{\rm wmo}} \ln\left(\frac{p_{k-1/2}}{p_{k+1/2}}\right) \tag{4.5}$$

for $1 \le k \le N$. Full-level values of the geopotential height are then given by interpolation as

$$Z_k = Z_{k-1/2} + \alpha_k \frac{R_{\rm dry}(T_v)_k}{g_{\rm wmo}} \tag{4.6}$$

where the interpolation coefficient α_k is

$$\alpha_k = 1 - \frac{p_{k+1/2}}{\Delta p_k} \ln\left(\frac{p_{k-1/2}}{p_{k+1/2}}\right) \tag{4.7}$$

for $1 \le k < N$ (with $\alpha_N = \ln 2$), and the pressure difference between half-levels is given by

$$\Delta p_k = p_{k-1/2} - p_{k+1/2} \tag{4.8}$$

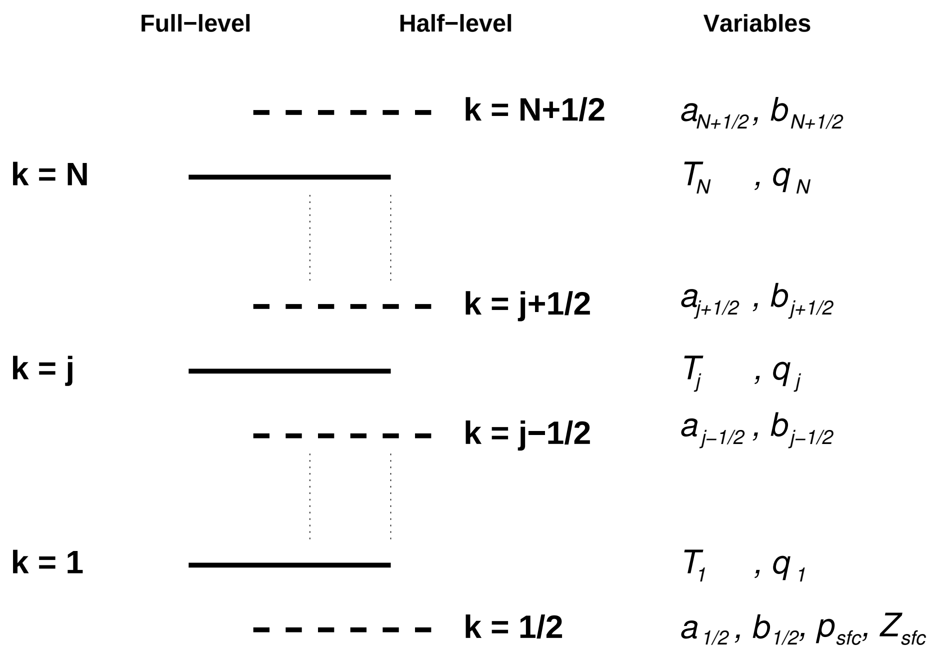


Figure 4.3: Schematic illustration of model level and data structure for a hybrid sigma vertical level model (e.g. ECMWF). The atmosphere is divided into N layers or full levels, defined by the pressures at the $N+1$ interfaces (half-levels) between them.

The virtual temperature T_v on each full level is derived from the background temperature x%temp$=T$ and specific humidity x%shum$=q$ as

$$T_v = T\left[1+\left(\frac{R_{\text{vap}}}{R_{\text{dry}}}-1\right)q\right] \tag{4.9}$$

where R_{vap} is the gas constant for water vapour. Note that all physical constants used in ropp_fm are defined in ropp_fm_constants.

4.6 Geopotential-based vertical levels - Met Office model type: ropp_fm_state2state_meto

For ropp_fm applications using background data from a model where the vertical coordinate is geopotential height-based (e.g. Met Office), it is necessary to know the pressure p on model levels where humidity information is stored. The temperature T at those locations can also be recovered using the hydrostatic equation (Eqn 4.2) assuming the temperature is constant across each model layer. These conversions from model-specific to generic ropp_fm variables are performed using a method which is consistent with the model dynamics in ropp_fm_state2state_meto.

```
USE ropp_fm
TYPE(State1dFM) :: x
CALL ropp_fm_state2state_meto(x)
```

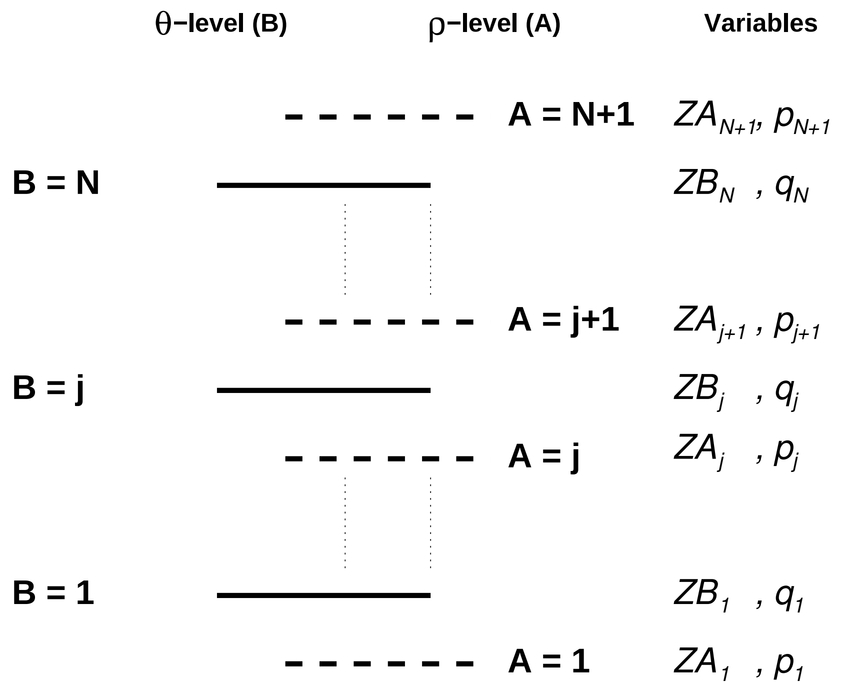


Figure 4.4: Schematic illustration of model level and data structure for a geopotential height based vertical level model (e.g. Met Office). Information is stored on a staggered height grid, with pressure and density on $N+1$ 'ρ' (or 'A') levels and potential temperature and humidity information on the N intermediate 'θ' (or 'B') levels.

4.6.1 Vertical grid structure

Figure 4.4 shows a schematic illustration of the vertical model level and data structure of the Met Office Unified Model (UM). Information is provided on a staggered vertical grid. Pressure and density are defined on 'ρ' or 'A' levels while humidity and potential temperature are defined on the intermediate 'θ' or 'B' levels.

The pressure on each 'B'-level `x%pres` is calculated from the available background data by first computing the Exner pressure Π on each 'A'-level as

$$\Pi_A = \left(\frac{p_A}{p_{\text{ref}}}\right)^{\kappa} \tag{4.10}$$

where $\kappa = R_{\text{dry}}/C_p$ and p_{ref} is the standard surface pressure (1000 hPa). The geopotential heights on the 'A'-levels are then estimated from those on the 'B'-levels by simple averaging/extrapolation, thus:

$$Z_A(1) = Z_{\text{sfc}} \tag{4.11}$$
$$Z_A(i) = \left(Z_B(i-1)+Z_B(i)\right)/2 \quad \text{for } i = 2,\ldots,N \tag{4.12}$$
$$Z_A(N+1) = \left(3Z_B(N)-Z_B(N-1)\right)/2. \tag{4.13}$$

The Exner value on the i^{th} 'B'-level is then calculated by assuming that it varies linearly with geopotential height, thus:

$$\Pi_B(i) = \left(\frac{Z_A(i+1) - Z_B(i)}{Z_A(i+1) - Z_A(i)}\right)\Pi_A(i) + \left(\frac{Z_B(i) - Z_A(i)}{Z_A(i+1) - Z_A(i)}\right)\Pi_A(i+1). \tag{4.14}$$

Finally, the pressure on each 'B'-level x%pres=p_B is then recovered from the Exner pressure via

$$p_B = p_{\text{ref}}(\Pi_B)^{1/\kappa}. \tag{4.15}$$

It is also possible to calculate the layer mean virtual temperature across level i given the pressure on each 'B'-level by using the hydrostatic equation to give

$$T_{vB}(i) = \frac{g_{\text{wmo}}}{C_p}\Pi_B(i)\left(\frac{Z_A(i+1) - Z_A(i)}{\Pi_A(i) - \Pi_A(i+1)}\right) \tag{4.16}$$

and the layer mean temperature x%temp=T_B is calculated using the specific humidity defined on 'B'-levels x%shum=q as

$$T_B = \frac{T_{vB}}{1 + (\varepsilon_w^{-1} - 1)q} \tag{4.17}$$

where ε_w is the ratio of molecular weight of water to that of dry air ($\approx$ 0.622).

4.6.2 Input data

The ropp_fm_roprof2state and ropp_fm_state2state_meto processing assume that the following variables are held in an input RO data structure containing background data from a geopotential height-based model.

- ro_data%Lev2b%shum — specific humidity values on each model 'B'-level.
- ro_data%Lev2b%geop — geopotential height values on each model 'B'-level.
- ro_data%Lev2c%geop_sfc — first element of Z_A, the geopotential height on 'A'-levels. Values of Z_A on other vertical levels are interpolated from Z_B in ropp_fm_state2state_meto.
- ro_data%Lev2c%press_sfc — first element of p_A, the pressure on 'A'-levels.
- ro_data%Lev2b%press — elements 2:x%n_lev+1 of p_A, the pressure on 'A'-levels.

Following the call to ropp_fm_state2state_meto elements x%temp, x%shum, x%pres and x%geop (those of the type State1dFM structure used by ropp_fm) are all co-located and correspond to data on the model 'B'-levels.

4.7 Defining the observation vector: ropp_fm_roprof2obs

When implemented as part of a data assimilation system, ropp_fm is required to compute refractivity or bending angle profiles from the background data on the observed height levels. These are unlikely to match the height levels on which the background data are defined.

If observations are read from a file into the `ROprof` structure, subroutines `ropp_fm_roprof2obs1drefrac` and `ropp_fm_roprof2obs1dbangle` can be used to define the observation variables of type `Obs1dRefrac` and `Obs1dBangle` respectively.

```
USE ropp_io
USE ropp_fm
TYPE(ROprof)       :: ro_data
TYPE(Obs1dRefrac) :: y
CALL ropp_fm_roprof2obs(ro_data, y)
```

A number of checks on the input data are conducted.

- Check longitude and latitude for background data are within range,
  set `y%lon = ro_data%georef%lon`
  set `y%lat = ro_data%georef%lat`
- Check that data arrays are in ascending order. Note that `ropp_fm` calculations assume that data array index 1 is closest to the surface. The input data are checked and reordered as required using
  `call ropp_io_ascend(ro_data)`
- Copy observation geopotential heights
  set `y%geop` = `ro_data%Lev2a%geop_refrac`
- Copy observed refractivity values
  set `y%refrac` = `ro_data%Lev2a%refrac`
- Set default observation weights values
  set `y%weights` = 1.0

Alternatively, for bending angle observations,

- Check longitude and latitude for background data are within range,
  set `y%lon = ro_data%georef%lon`
  set `y%lat = ro_data%georef%lat`
  `call ropp_io_ascend(ro_data)`
- Copy the radius of curvature
  set `y%rcurve` = `ro_data%georef%roc`
- Copy the undulation
  set `y%undulation` = `ro_data%georef%undulation`
- Copy the azimuth
  set `y%azimuth` = `ro_data%georef%azimuth`
- Check that data arrays are in ascending order. Note that `ropp_fm`calculations assume that data array index 1 is closest to the surface. The input data are checked and reordered as required using
- Copy observation impact parameters
  set `y%impact` = `ro_data%Lev1b%impact`
- Copy observed bending angle values
  set `y%bangle` = `ro_data%Lev1b%bangle`

Elements y%g_sfc and y%r_earth are also set using latitude-dependent routines based on Somagliana's equation Mahoney (2001) provided as part of ropp_utils (ROM SAF, 2007).

The diagonal elements of the observation error covariance (y%cov%d) may also be specified in ropp_fm_roprof2obs by the estimated error associated with the bending angle or refractivity observations.

$$\texttt{y\%cov\%d(i+i*(i-1)/2)} \quad = \texttt{ro\_data\%Lev2a\%refrac\_sigma(i)}^2 \quad \text{for each level } i$$

4.7.1 Set observation geopotential height levels set_obs_levels_refrac

The set_obs_levels_refrac subroutine provided as part of the stand-alone ropp_fm_bg2ro tool provides a simple way of defining the observation geopotential height levels for which the forward modelled refractivity profiles are to be computed. The observation vector geopotential height element y%geop defaults to 300 evenly spaced vertical levels between 200 and 60 000 gpm, although the user can alter this. The refractivity element y%refrac is initialised to zero and the weights y%weights are initialised to unity.

4.7.2 Set observation impact parameter levels set_obs_levels_bangle

By default, or if ih_from_geop has been set to .TRUE. by other means, the impact parameters for the bending angle observation vector y%impact are set to coincide with the same geopotential levels defined for the refractivity observation vector. Otherwise the bending angle levels are independent of the refractivity levels. The geopotential heights Z are converted into geometric heights h using the latitude-dependent conversion

$$h(Z,\phi) = \frac{R_{\text{eff}} Z}{(g/g_{\text{wmo}})R_{\text{eff}} - Z} \tag{4.18}$$

where R_{eff} is the Earth's effective radius and g is the surface gravity, both at latitide ϕ (Mahoney, 2001). This is implemented in the ropp_utils function geopotential2geometric.

Using Eqn (3.4) the impact parameter y%impact=a is then defined as

$$a = nr = (1+10^{-6}N)(h+R_c) \tag{4.19}$$

where R_c is the radius of curvature of Earth, corrected for the difference between the EGM-96 geoid (NASA/NIMA) and the WGS-84 ellipsoid (NGA/WGS84), and N is the refractivity at each observation height, computed by a prior call to the refractivity forward model ropp_fm_refrac_1d.

For convenience, other geodetic parameters are also defined in the bending angle observation vector (Table 4.1). These latitude-dependent variables are computed with reference to the WGS-84 ellipsoid using expressions derived from Somagliana's equation (see Mahoney (2001) for more details). The following functions provided as part of the ropp_utils module are used.

Gravity at surface:	obs_bangle%g_sfc	=	gravity(ro_data%GEOref%lat)
Effective Earth radius:	obs_bangle%r_earth	=	r_eff(ro_data%GEOref%lat)
Radius of curvature:	obs_bangle%r_curve	=	ro_data%GEOref%roc

(Users should be aware of the important physical difference between the effective radius, which is used in expressions involving the variation of gravity with height such as (4.18), and the radius of curvature, which is used in expressions involving the ray path geometry such as (4.19). ROM SAF Scientific Report 14 (ROM SAF, 2013b) explains the point in detail. The effective radius is computed within ROPP from the latitude (see the ROPP UTILS User Guide (ROM SAF, 2021d)). It is not part of the `ROprof` data structure and is therefore never output from ROPP. The radius of curvature, on the other hand, can be read in or calculated if necessary, and will be written out (as the variable '`roc`'), because it is held as part of the `ROprof%GEOtype` data substructure.)

4.8 Refractivity forward model `ropp_fm_refrac_1d`

`ropp_fm` computes the refractivity N on observation geopotential height levels by applying Eqn (3.8) to background profiles of pressure, humidity and temperature and interpolating the results onto observation heights (with the alternative forward model the background values are interpolated separately to the observation heights, where the refractivity is calculated). Constants k_1, k_2 and k_3 are defined in `ropp_fm_constants`. The input background data are contained in the state vector `x` of type `State1dFM` (Section 4.4) and the output refractivity values are contained in an element of an observation vector `y` of type `Obs1dRefrac` (Section 4.2.1).

```
USE ropp_fm
TYPE(State1dFM)   :: x
TYPE(Obs1dRefrac) :: y
CALL ropp_fm_refrac_1d(x, y)
```

4.8.1 Update variables in State1dFM

Elements of the `State1dFM` structure are computed from state vector `x%state` every time `ropp_fm_refrac_1d` is called to ensure that the temperature, pressure and humidity variables `x%temp`, `x%pres` and `x%shum` are consistent with `x%state`. This becomes necessary when `ropp_fm_refrac_1d` is implemented within `ropp_1dvar` (See Secs 4 and 5 of ROPP 1D–Var UG). The computation depends on the details of the state vector and the background data. Subroutine `ropp_fm_state2state_ecmwf` is used if the background data are on pressure-based vertical levels (Section 4.5) while `ropp_fm_state2state_meto` should be used if the data are on geopotential height-based levels (Section 4.6).

4.8.2 Compute partial water vapour pressure

The water vapour pressure e on each background level is derived from the definition of specific humidity q as

$$q = \frac{r}{1+r}$$

where r is the dry mixing ratio of water vapour defined as

$$r = \varepsilon_w \left(\frac{e}{p-e} \right)$$

This leads to a relationship for e in terms of the background specific humidity x%shum=q and pressure x%pres=p as

$$e = p \left(\frac{q}{\varepsilon_w + (1-\varepsilon_w)q} \right) \tag{4.20}$$

4.8.3 Compute refractivity

Eqn (3.8) is applied to compute a refractivity profile $N(Z)$ on background vertical levels from background profiles of pressure x%pres, temperature x%temp and partial water vapour pressure.

4.8.4 Interpolate refractivity to measurement geopotential levels ropp_fm_interpol_log

The refractivity values N computed on the background geopotential height levels Z are interpolated to the required observation heights y%geop to enable direct comparison with the observations. This interpolation is calculated in ropp_fm_interpol_log assuming that refractivity varies exponentially with geopotential height between the background levels.

```
USE ropp_fm
TYPE(State1dFM)   :: x
TYPE(Obs1dRefrac) :: y
CALL ropp_fm_interpol_log (x%geop, y%geop, N, y%refrac)
```

At each observation height k, the output refractivity y%refrac is calculated as

$$\ln(\mathtt{y\%refrac}(k)) = \ln(N_j) + \frac{\mathtt{y\%geop}(k) - \mathtt{x\%geop}(j)}{\mathtt{x\%geop}(j-1) - \mathtt{x\%geop}(j)}(\ln(N_{j-1}) - \ln(N_j)) \tag{4.21}$$

where observation level k lies between background model levels $j-1$ and j.

4.8.5 Alternative refractivity forward model

The alternative refractivity interpolation scheme has been shown to reduce biases in the innovations, which are largest in magnitude between the model levels (ROM SAF, 2013a). This can be seen at heights of 30 to 40km in Figure 4.5(a), where the Met Office model levels are overlaid. Figure 4.5(b) shows the smoother statistics obtained using the alternative interpolation. Note that for models with higher vertical resolution, the original biases may be small enough not to be problematic.

This option is selected using the '-new_op' command line flag when calling the ropp_fm_bg2ro_1d and ropp_1dvar_refrac tools.. Note that the first of these calls both the refractivity and bending angle forward models, whereas the second only calls the refractivity forward model.

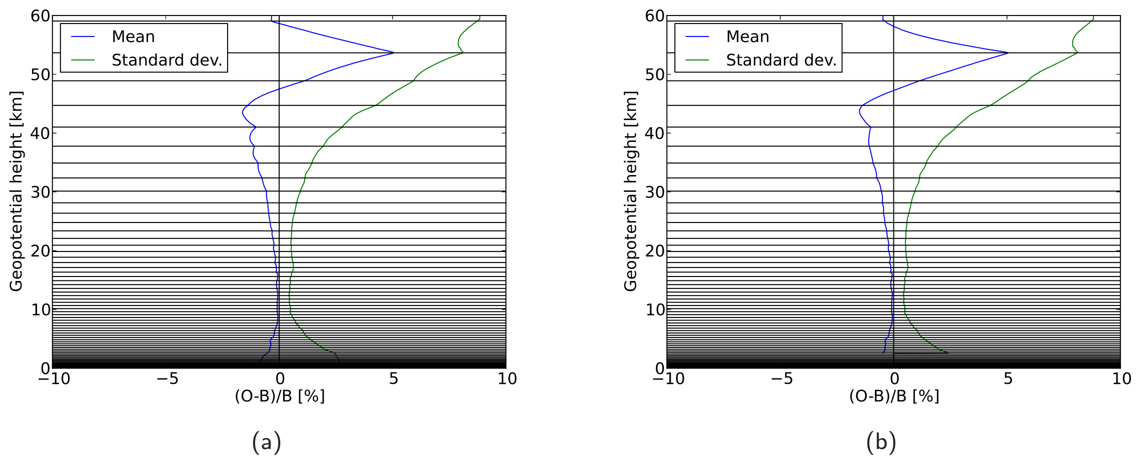


Figure 4.5: Refractivity innovation statistics, (observation-background)/background, using 1000 UCAR-processed profiles and co-located backgrounds from the 70-level Met Office model. The forward modelled refractivities are calculated using the standard forward model in (a) and the alternative forward model in (b). The horizontal lines correspond to the geopotential heights of the Met Office model levels over sea. Note that the large biases above ˜45km are due to temperature biases in the Met Office backgrounds.

The alternative refractivity forward model option is based on the approach described in ROM SAF (2013a). It involves interpolating separately the background temperature, pressure and humidity to the observation levels. We first calculate the temperature at the observation height by assuming that the temperature varies linearly between the surrounding model levels:

$$T(z) = T_j + \beta_j (z - z_j) \tag{4.22}$$

where

$$\beta_j = \frac{T_{j+1} - T_j}{z_{j+1} - z_j} \tag{4.23}$$

Based on this variation of $T(z)$, the hydrostatic equation is solved to compute the pressure at the observation height:

$$P(z) = P_j \left(\frac{T(z)}{T_j} \right)^{-\frac{g}{R\beta_j}} \tag{4.24}$$

Note that in the isothermal case, i.e. when $\beta_j = 0$, the pressure calculation will result in a singularity, but in the infinitesimal limit, the expression tends to exponentially varying pressure:

$$\lim_{\beta_j \to 0} P(z) = \lim_{\beta_j \to 0} P_j \left(1 + \frac{\beta_j (z - z_j)}{T_j} \right)^{-\frac{g}{R\beta_j}} = P_j \exp\left(-\frac{g}{RT} (z - z_j) \right) \tag{4.25}$$

To avoid such problems in practice, if the absolute temperature difference is less than ropp_ZDTV, i.e.

10^{-10}, (this has been known to occur with L137 ECMWF background profiles and standard atmospheres), then the exponential form of the pressure is used, but the scale height is replaced as follows to ensure the pressure is continuous at the top of the layer:

$$P(z) = P_j \exp\left(-\frac{\ln(P_j/P_{j+1})}{z_{j+1}-z_j}(z-z_j)\right) \tag{4.26}$$

For the usual, non-isothermal case, continuity of the pressure is enforced by replacing β_j with a value that forces continuity, γ_j, i.e.

$$P(z) = \left(\frac{T(z)}{T_j}\right)^{-\frac{g}{R\gamma_j}} \tag{4.27}$$

where

$$\gamma_j = -\frac{g}{R}\frac{\ln(T_{j+1}/T_j)}{\ln(P_{j+1}/P_j)} \tag{4.28}$$

The specific humidity is assumed to vary exponentially within each layer:

$$q(z) = q_j \exp(-\alpha_j(z-z_j)) \tag{4.29}$$

where

$$\alpha_j = \frac{\ln(q_j/q_{j+1})}{z_{j+1}-z_j} \tag{4.30}$$

If a negative humidity value is encountered in the refractivity forward model, the negative value is ignored and a small positive value (10^{-6} kg/kg) is used instead.

The refractivity is then calculated using $T(z)$, $P(z)$ and $q(z)$ and, optionally, the 'compressibility' factors, which are themselves functions of $T(z)$, $P(z)$ and $q(z)$ — see Sec 6. If an observation height is below the lowest model level, then information from the bottom layer is used to extrapolate the refractivity exponentially below the lowest level.

4.8.6 Refractivity forward model gradient

The `ropp_fm` module includes routines to compute the gradient (tangent linear) and adjoint of the refractivity forward model with respect to the state vector for use in `ropp_1dvar`. Corresponding tangent linear and adjoint codes exist for each `ropp_fm` routine. Further details are provided in Sec 4 of ROPP 1D–Var UG.

4.9 Dry temperature forward model `ropp_fm_tdry_1d`

For the purpose of comparing observations with model data, `ropp_fm` also computes a profile of dry temperature T_{dry} as a function of geopotential height Z. The height levels are identical to the heights used in the refractivity forward model. Dry temperature at a given geopotential height is derived according to Eqn (3.9) using the refractivity profile computed via Eqn (3.8). The input background data are contained in the state vector `x` of type `State1dFM` (Section 4.4) and the input refractivity data are contained in the observation vector `y` of type `Obs1dRefrac` (Section 4.2.1). Thus, a call to `ropp_fm_tdry_1d` must be preceded by a call to `ropp_fm_refrac_1d`. The dry temperature profile is returned in an allocated one-dimensional vector.

```
USE ropp_fm
TYPE(State1dFM)   :: x
TYPE(Obs1dRefrac) :: y
REAL(wp), DIMENSION(:), ALLOCATABLE :: tdry
CALL ropp_fm_refrac_1d(x, y)
ALLOCATE(tdry(size(y%refrac)))
CALL ropp_fm_tdry_1d(x, y, tdry)
```

4.9.1 Calculate dry temperature `ropp_fm_tdry`

The calculation of dry temperature in `ropp_fm` follows closely the calculation used in the inversion of real observations described in the User Guide for the `ropp_pp` module (2021c). The only difference is the handling of the upper boundary condition.

First the geopotential heights on the model levels are converted to geometric heights using Eqn (4.18) as implemented in the `ropp_utils` function `geopotential2geometric`. Using the equation of state for an ideal gas and assuming hydrostatic equilibrium, the dry pressure at each geometric height level, z, is obtained by solving

$$\frac{d\ln P_{\text{dry}}}{dz} = f(z, \ln P_{\text{dry}}(z)) = \frac{-g(z)N(z)}{R\kappa_1 \exp(\ln P_{\text{dry}}(z))} \tag{4.31}$$

using a fourth order Runge-Kutta method. The gravitational acceleration, g, is a function of z and also depends on the reference latitude of the occultation — see the Geodesy Section of (ROM SAF, 2021d) for details. The ideal gas refractivity constant $\kappa_1 = 77.60$ N-unit K hPa^{-1} in ROPP — see Chapter 3 for more. R is the dry gas constant (287.05 J kg^{-1} K^{-1} in ROPP).

For the Runge-Kutta integration in roppfm, the initial value of $\ln P_{\text{dry}}$ is given by the model pressure at the top altitude (controlled by the optional input variable `Zscale`). The top altitude in the Runge-Kutta integration is taken to be the second topmost level of the model. At the top level of the model, the dry temperature is set equal to the model temperature. Example use:

```
USE ropp_utils
USE ropp_fm
shum = 0.0_wp
```

```
CALL ropp_fm_tdry(lat, alt, refrac, shum, tdry_mod, pdry_mod, Zscale=Hscale)
```

The dry temperature T_{dry} at model levels below the top level is then derived from the dry pressure P_{dry} and the refractivity N by using (cf Eqn (3.9))

$$T_{\text{dry}} = \kappa_1 P_{\text{dry}}/N. \tag{4.32}$$

4.9.2 Interpolate dry temperature to measurement geopotential levels `ropp_fm_interpol`

The dry temperature values T_{dry} computed on the model geopotential height levels Z are interpolated to the required observation heights `y%geop` to enable direct comparison with the observations. This interpolation is calculated in `ropp_fm_interpol` assuming that dry temperature varies linearly with geopotential height between the model levels.

```
USE ropp_fm
TYPE(State1dFM)   :: x
TYPE(Obs1dRefrac) :: y
CALL ropp_fm_interpol(x%geop, y%geop, tdry_mod, tdry)
```

4.10 Bending angle forward model `ropp_fm_bangle_1d`

`ropp_fm` computes a profile of bending angles α as a function of impact parameter a corresponding to the observation geopotential heights used in the refractivity forward model. Bending angles at a given impact parameter are derived by evaluating Eqn (3.5) using the refractivity profile computed on background vertical levels using Eqn (3.8). The input background data are contained in the state vector `x` of type `State1dFM` (Section 4.4) and the output bending angle values are contained in an element of an observation vector `y` of type `Obs1dBangle` (Section 4.2.1).

```
USE ropp_fm
TYPE(State1dFM)   :: x
TYPE(Obs1dBangle) :: y
CALL ropp_fm_bangle_1d(x, y)
```

4.10.1 Update variables in State1dFM

As in `ropp_fm_refrac_1d` either `ropp_fm_state2state_ecmwf` or `ropp_fm_state2state_meto` is called to ensure than temperature, pressure and humidity variables `x%temp`, `x%pres` and `x%shum` are consistent with `x%state` before computing the forward model.

4.10.2 Compute refractivity profile

The refractivity N on each background level is calculated following the process described in Section 4.8. This involves the following stages:

- compute partial water vapour pressure on background geopotential height levels using Eqn (4.20)
- compute refractivity `refrac` on background geopotential height levels using Eqn (3.8)

4.10.3 Compute background impact parameter

The impact parameter corresponding to each background level is computed using the refractivity values N following Eqn (3.4)

$$\texttt{impact} = nr = (1 + 1 \times 10^{-6} N)(h + R_c) \tag{4.33}$$

The background geopotential height levels `x%geop` are converted to geometric altitude h using the `ropp_utils` function `geopotential2geometric`.

4.10.4 Calculate bending angles `ropp_fm_abel`

The evaluation of Eqn (3.5) in the bending angle forward model is performed in subroutine `ropp_fm_abel`. It computes bending angle `y%bangle` as a function of observation impact parameter `y%impact` from the profile of refractivity and temperature values on background impact parameter levels (Healy and Thépaut, 2006). *We restrict the calculation to non-super-refracting conditions.*

```
   USE ropp_fm
   TYPE(Obs1dBangle) :: y
   CALL ropp_fm_abel &
 (impact, refrac, x%temp, y%r_curve, x%new_bangle_op, y%impact, y%bangle)
```

Eqn (3.5) is simplified by approximating

$$\frac{d\ln(n)}{dx} \approx 10^{-6}\frac{dN}{dx} \tag{4.34}$$

which is valid because the refractivity is small, and

$$\sqrt{x^2 - a^2} \approx \sqrt{2a(x-a)} \tag{4.35}$$

This is valid because the refractivity scale height is small compared to the radius of the Earth. This gives

$$\alpha(a) = -\sqrt{2a}10^{-6}\int_a^\infty \frac{dN/dx}{\sqrt{x-a}}dx \tag{4.36}$$

The computation of the bending angle integral requires an assumption about the form of the refractivity variation between the model levels, because the NWP model only provides N on a finite set of levels. The assumption used in the routine depends on whether the refractivity, N, is increasing or decreasing with height.

When the refractivity is increasing with height, we assume that the refractivity gradient is constant between the levels. The gradient between the j and $(j+1)$ levels is approximated by

$$\left(\frac{dN}{dx}\right)_{av} = \frac{N_{j+1}-N_j}{x_{j+1}-x_j} \tag{4.37}$$

The bending angle computed between these levels is then

$$\alpha(a) = -2\sqrt{2a}10^{-6}\left(\frac{dN}{dx}\right)_{av}\left(\sqrt{(x_{j+1}-a)}-\sqrt{(x_j-a)}\right) \tag{4.38}$$

Simulations with the ECMWF model suggest that a positive refractivity gradient with height occur within a profile in around $\sim 7\%$ of cases.

The bending angle integration is more complicated in the more usual case where refractivity decreases with height. *In the default setting, it is still assumed that the refractivity varies exponentially between the model levels.* This is because by default `x%new_bangle_op = .FALSE.` when initialising the `State1dFM` structure `x` in the routine `ropp_fm_types.f90`. However, the new formulation of the bending angle integral can be tested using the '`-new_op`' command line flag when calling the `ropp_fm_bg2ro_1d` and `t_fascod` tools, and tested in a 1D-Var context using the flag '`-new_op`' when running `ropp_1dvar_bangle`.

Physically, the exponential decay is equivalent to assuming that the atmosphere is isothermal within the model layer. However, recent work has shown that this assumption can produce forward model biases of a few tenths of a percent in bending angle, when the model level separation gets large in the stratosphere. More details can be found in Burrows *et al.* (ROM SAF, 2013a), where alternative solutions are also given. The same report had led to a alternative formulation of the bending angle computation in ROPP. This impoves the bending angle departure for the Met Office model, as shown in (ROM SAF, 2013a).

When `x%new_bangle_op= .TRUE.` the new approach uses a functional form which approximates the effect of a linear temperature gradient between the model levels. The new approach is only valid for a dry atmosphere, hence it is only implemented above 12 km currently. The 12 km limit is set in `ropp_fm_constants.f90`, and is given by the parameter `imp_ht_min`. It can be adjusted if required. Below 12 km, we still assume an exponential decay between the model levels, but this is reasonable because the model level separations are smaller in the troposphere in most NWP systems, and so the interpolation between the model levels becomes less important.

The new computation proceeds as follows. We compute the inverse of a refractivity scale-height between the j and $j+1$ levels,

$$k_j = \frac{\ln(N_j/N_{j+1})}{(x_{j+1}-x_j)}. \tag{4.39}$$

The value of k_j is required to be positive for numerical reasons, so a minimum positive value of $k_j^{\min} = 10^{-6}$ m^{-1} is assumed. We also require $x_{j+1}-x_j$ to be at least 10 m to avoid problems with large refractivity gradients near super-refracting regions. In fact, a stronger restriction is also used so that k_j can be no larger than $0.157/N_j$. The maximum dN/dx is set to 0.157 N-units/m which is the curvature of the earth, so that dN/dr is about half the critical refraction value. This avoids problems encountered in a 4D-Var assimilation system with non-linearity when the atmosphere approaches super-refracting conditions. We note that this limit may not be necessary if the `ropp_fm_abel` routine is simply being used to *simulate* the measurements, but experience at the Met Office and ECMWF suggests it is necessary when the routine is used for assimilation purposes in incremental 4D-Var systems where linearity is assumed.

The new operator accounts for the variation of k within a model layer as a result of a change in temperature. The approach is based on the fact that for a dry atmosphere the inverse scale height, $k = 1/H$, can be written as,

$$k(z) = \left(\frac{g}{RT(z)} + \frac{\beta}{T(z)} \right) \tag{4.40}$$

where $T(z)$ is the temperature at height z, g = 9.80665 ms^{-2} is the standard acceleration due to gravity value, R is the ideal gas constant, and $\beta = dT/dz$ is the temperature gradient. The gradient of k with respect to temperature, T, is then

$$\frac{dk}{dT} = -\frac{k}{T} \tag{4.41}$$

Using the chain rule for differentiation, the gradient of k with respect to z is

$$\frac{dk}{dz} = -\frac{k}{T}\beta \tag{4.42}$$

It is assumed that the computed value of k_j is a good approximation for the inverse scale-height near the centre of the layer, $x_m = (x_j + x_{j+1})/2$. The Mean Value Theorem of Calculus tells us that k_j will be correct somewhere between the model levels, and it has been verified numerically this is near the centre of the model layer. Based on Eqn (4.42), but noting the use of vertical variable x, we estimate the gradient in the layer,

$$\frac{dk}{dx} \simeq -\frac{k_j}{T_m}\beta_j \tag{4.43}$$

where $T_m = (T_j + T_{j+1})/2$ is the mean temperature of layer, and the temperature gradient with respect to x is given by (as in Eqn (4.23)),

$$\beta_j = \frac{T_{j+1} - T_j}{x_{j+1} - x_j} \tag{4.44}$$

This expression can then be integrated to give the change in k between the j and $j+1$ model levels,

$$k(x) \simeq k_j - \frac{k_j \beta_j}{T_m}(x - x_m) \tag{4.45}$$

We then obtain the refractivity for this $k(x)$ by integrating:

$$\int \frac{dN}{N} = -\int \left(k_j - \frac{k_j \beta_j}{T_m}(x - x_m) \right) dx \tag{4.46}$$

This gives,

$$\ln(N) = -\left(k_j (x - x_j) - \frac{k_j \beta_j}{2T_m}(x - x_m)^2 \right) + c \tag{4.47}$$

where c is a constant of integration. The refractivity can be written as

$$N(x) = N_j \exp\left(-k_j(x-x_j) + \frac{k_j\beta_j}{2T_m}\left((x-x_m)^2 - d\right)\right) \tag{4.48}$$

where d is chosen to satisfy the model refractivity values at x_j and x_{j+1} and is given by,

$$d = (x_j - x_m)^2 = (x_{j+1} - x_m)^2 \tag{4.49}$$

If the second term in the exponential is small, we can approximate the refractivity within the layer using the expansion,

$$N(x) \simeq N_j \exp(-k_j(x-x_j)) \times \left(1 + \frac{k_j\beta_j}{2T_m}\left((x-x_m)^2 - d\right)\right) \tag{4.50}$$

Note that this introduces the largest changes at the centre of the model layer when compared with the pure exponential decay. The refractivity is approximated with the product of an exponential multiplied by a quadratic, but it reverts to the pure exponential expression when $\beta_j = 0$, as expected.

The vertical gradient of N is

$$\frac{dN}{dx} = -N_j \exp(-k_j(x-x_j))\left(k_j + \frac{k_j^2\beta_j}{2T_m}\left((x-x_m)^2 - d\right) - \frac{k_j\beta_j}{T_m}(x-x_m)\right) \tag{4.51}$$

After writing $(x-x_m) = (x-a) + (a-x_m)$ the gradient can be re-written as,

$$\frac{dN}{dx} = -N_j \exp(-k_j(x-x_j))\left(P_1 + P_2(x-a) + P_3(x-a)^2\right) \tag{4.52}$$

where,

$$\begin{aligned} P_1 &= k_j + \frac{k_j^2\beta_j}{2T_m}\left((a-x_m)^2 - d\right) - \frac{k_j\beta_j}{T_m}(a-x_m) \\ P_2 &= \frac{k_j^2\beta_j}{T_m}(a-x_m) - \frac{k_j\beta_j}{T_m} \\ P_3 &= \frac{k_j^2\beta_j}{2T_m} \end{aligned} \tag{4.53}$$

Note again that if $\beta_j = 0$ then $P_1 = k_j$ and $P_2 = P_3 = 0$, and the problem simplifies to the pure exponential solved previously in ROPP. In fact, we set $P_1 = k_j$ and $P_2 = P_3 = 0$ to compute the bending angle contributions below 12 km, or when `x%new_bangle_op = .FALSE.`.

Substituting Eqn 4.52 into Eqn 4.36 leads to an expression for the bending between the j and $j+1$ levels,

$$\Delta\alpha \quad = 10^{-6}\sqrt{2a}\, N_j \exp(k_j\,(x_j-a)) \quad \left[\mathrm{erf}\left(\sqrt{k_j\,(x-a)}\right)\sqrt{\pi}\left(\frac{P_1}{k_j^{1/2}}+\frac{P_2}{2k_j^{3/2}}+\frac{3P_3}{4k_j^{5/2}}\right)- \sqrt{x-a}\ \exp(-k_j\,(x-a))\left(\frac{P_2}{k_j}+\frac{P_3\,(2k_j\,(x-a)+3)}{2k_j^2}\right)\right]_{x_j}^{x_{j+1}} \tag{4.54}$$

When $P_1 = k_j$ and $P_2 = P_3 = 0$, this simplifies to (Healy and Thépaut, 2006),

$$\Delta\alpha_j = 10^{-6}\sqrt{2\pi a k_j} N_j \exp(k_j(x_j-a)) \left[\mathrm{erf}\left(\sqrt{k_j(x-a)}\right)\right]_{x_j}^{x_{j+1}}$$

The error function `erf` is defined as

$$\mathrm{erf}(x) = \frac{2}{\sqrt{\pi}}\int_0^x \exp(-t^2)\,\mathrm{d}t \tag{4.55}$$

The error function terms are computed in `ropp_fm_abel` using the following polynomial approximation to `erf` (Abramowitz and Stegun, 1965):

$$\mathrm{erf}(x) \approx 1-(a_0+a_1t+a_2t^2)t\exp(-x^2) \tag{4.56}$$

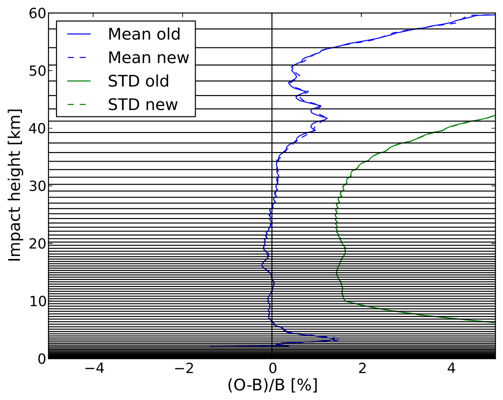
where $t = 1/(1+bx)$ and

$$a_0 = 0.3480242; \quad a_1 = -0.0958798; \quad a_2 = 0.7478556; \quad b = 0.47047.$$

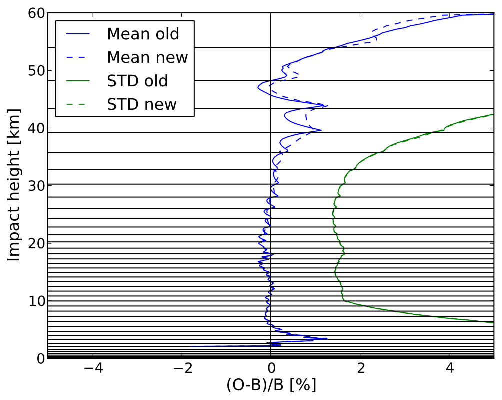
The total bending angle `y%bangle` for a given impact parameter `y%impact`$=a$ is then found in `ropp_fm_abel` by summing contributions calculated using Eqn 4.55 between a and the top of the background profile. Ray bending above the model top is accounted for by extrapolating assuming an exponential when $j+1$=`x%n_lev` and evaluating

$$\Delta\alpha_{top} = 10^{-6}\sqrt{2\pi a k_j} N_j \exp(k_j(x_j-a)) \left[1-\mathrm{erf}\left(\sqrt{k_j(x_j-a)}\right)\right] \tag{4.57}$$

The importance of the new bending angle forward model depends on the vertical spacing of the model levels. Figure 4.6(a) shows effect of using the new forward model on ECMWF innovation statistics (91-level model). It can be seen that the level of improvement is modest. Figure 4.6(b) shows the effect of the forward model when the background profiles are thinned vertically by a factor of 2. The improvement is much more significant, and the reduction in the oscillatory bias is similar to the improvement seen using 70-level Met Office backgrounds (in the upper stratosphere, the Met Office levels have similar vertical spacing to the thinned ECMWF case). This demonstrates that the alternative forward model can provide significantly improved bias characteristics for models with low vertical resolution in the stratosphere.



(a)



(b)

Figure 4.6: Bending angle innovation statistics, (observation-background)/background, using profiles from all available operational instruments over 30 days. (a) shows the innovation statistics using co-located backgrounds from the 91-level ECMWF model, using both the standard and alternative forward models (labeled 'old' and 'new' respectively). (b) shows the same statistics, but with alternate model levels omitted completely, thus halving the effective vertical resolution. The horizontal lines correspond to the heights of the model levels used in each case.

4.10.5 Bending angle forward model gradient

The `ropp_fm` module includes routines to compute the gradient (tangent linear) and adjoint of the bending angle forward model with respect to the state vector for use in the `ropp_1dvar` 1D–Var module. Corresponding tangent linear and adjoint code exist for each `ropp_fm` routine. Further details on this code is provided in Sec 5 of ROPP 1D–Var UG. If `ropp_fm_bg2ro_1d` is called without the `-f` option, it outputs the forward model gradients $\partial N_i/\partial x_j$ and $\partial \alpha_i/\partial x_j$, where $\mathbf{x}$ is the state vector (see Chapter 5 of ROPP 1D–Var UG). (These are large matrices which take a long time to generate, so most users will probably want to call `ropp_fm_bg2ro_1d -f`.)

4.11 Copy simulated observations to RO structure `ropp_fm_obs2roprof`

The computed refractivity and bending angle observation vectors are copied to the Level2a and Level1b parts of the `ROprof` data structure respectively for writing to the output netCDF data file using `ropp_fm_obs2roprof`. The dry temperature, sharing the altitude and geopotential height levels with the refractivity, is simply included in the Level2a part.

```
USE ropp_io
USE ropp_fm
TYPE(State1dFM)  :: ro_data
TYPE(Obs1dRefrac) :: obs_refrac
TYPE(Obs1dBangle) :: obs_bangle
CALL ropp_fm_obs2roprof(obs_refrac, ro_data)
CALL ropp_fm_obs2roprof(obs_bangle, ro_data)
```

```
ro_data%Lev2a%dry_temp = tdry
```

The results are written to the specified output file using the ropp_io module routine ropp_io_write.

4.12 Plotting tools

The directory tests/ contains the IDL procedure plot_fm.pro which gives an example of how to read and plot refractivity and bending angle profiles computed by running the forward model.

An example of its implementation, using archive data available from the ROM SAF archive (http://www.romsaf.org), is provided by running the test script test_fm_GRAS.sh.

4.13 Test software tools

In addition to the main stand-alone simulation tool ropp_fm_bg2ro_1d, we provide three additional simple test programs that the user may find instructive. These are in the directory tests/ and are called:

- t_fascod: a simple call to the 1D refractivity and bending angle operators and comparison with pre-calculated profiles.
- t_fascod_tl: a test of the 1D operator tangent linear code.
- t_fascod_ad: a test of the 1D operator adjoint code.

Note these programs use the ropp_io module for reading the test data.

4.13.1 t_fascod

This simple program reads meteorological and observation data from the file data/FASCOD_Scenarios.nc, simulates refractivity and bending angle profiles with the 1D operators, ropp_fm_refrac_1d and ropp_fm_bangle_1d, and compares the results with pre-calculated profiles. If the fractional differences are greater than 0.0001 a failure message is written to stdout.

4.13.2 t_fascod_tl

This program provides an example of how to call and test the tangent linear of the 1D refractivity and bending angle codes, ropp_fm_refrac_1d_tl and ropp_fm_bangle_1d_tl. For example, the 1D operator is called to compute simulated observations. A set of random perturbations between 0 and 1 are stored in a x_tl variable structure of type Obs1dBangle (the geopotential perturbation is increased a factor of 100). The code then loops through the following procedures 15 times. Firstly, the perturbations are added to the original state:

```
x_new%temp(:) = x%temp(:) + x_tl%temp(:)
x_new%pres(:) = x%pres(:) + x_tl%pres(:)
```

```
   x_new%geop(:) = x%geop(:) + x_tl%geop(:)
   x_new%shum(:) = x%shum(:) + x_tl%shum(:)
```

and the 1D operator is called with the perturbed state vector. The tangent linear routine is then called with the original vector and the perturbation

```
   CALL ropp_fm_bangle_1d(x_new,y_new)
   CALL ropp_fm_bangle_1d_tl(x,x_tl,y,y_tl)
```

The tangent linear test routine `t_fascod_tl` then compares the bending angle produced by the tangent linear routine, `y_tl%bangle` $= \mathbf{K}\Delta\mathbf{x}$ (where $\mathbf{K} = \partial\mathbf{H}/\partial\mathbf{x}$), with the finite difference approximation `y_new%bangle(:) - y_save%bangle(:)` $= \mathbf{H}(\mathbf{x}+\Delta\mathbf{x}) - \mathbf{H}(\mathbf{x})$, by calculating the cosine of the angle between them,

$$\cos\theta = \frac{(\mathbf{K}\Delta\mathbf{x})\cdot(\mathbf{H}(\mathbf{x}+\Delta\mathbf{x})-\mathbf{H}(\mathbf{x}))}{|\mathbf{K}\Delta\mathbf{x}||\mathbf{H}(\mathbf{x}+\Delta\mathbf{x})-\mathbf{H}(\mathbf{x})|}, \tag{4.58}$$

and the relative error,

$$\texttt{rel\_err} = 100\,\frac{\mathbf{H}(\mathbf{x}+\Delta\mathbf{x})-\mathbf{H}(\mathbf{x})-\mathbf{K}\Delta\mathbf{x}}{|\mathbf{H}(\mathbf{x}+\Delta\mathbf{x})-\mathbf{H}(\mathbf{x})|}. \tag{4.59}$$

These quantities are then reported to stdout.

The procedure is repeated, with the perturbations being reduced by a factor of 10 at each iteration. Initially, $\cos\theta$ increases (tends to unity), and `rel_err` decreases (tends to zero) as the size of the perturbation decreases, but numerical rounding error eventually causes them to diverge from these ideals. Nonetheless, their closest approach to them is a good test of the operator/tangent linear consistency. A failure message is written to stdout if $\cos\theta$ and `rel_err` have significantly different convergence characteristics.

4.13.3 `t_fascod_ad`

The `t_fascod_ad` program tests the consistency of the tangent linear and adjoint codes (`ropp_fm_refrac_1d_tl` and `ropp_fm_refrac_1d_ad`, and `ropp_fm_bangle_1d_tl` and `ropp_fm_bangle_1d_ad`).

Initially (in the case of bending angle, for example) the non-linear 1D bending angle operator is called:

```
   CALL ropp_fm_bangle_1d(x,y)
```

A set of random perturbations $\Delta\mathbf{x}$ between 0 and 1 are stored in a `x_tl` variable structure of type `Obs1dBangle` (the geopotential perturbation is increased a factor of 100) and the tangent linear routine is called, thus:

```
   CALL ropp_fm_bangle_1d_tl(x, x_tl, y, y_tl)
```

The norm of the linearised perturbation in observation space, `norm1`, is defined as

$$\texttt{norm1} = \Delta\mathbf{y}\cdot\Delta\mathbf{y} = \mathbf{K}\Delta\mathbf{x}\cdot\mathbf{K}\Delta\mathbf{x} = (\mathbf{K}\Delta\mathbf{x})^T\mathbf{K}\Delta\mathbf{x} = \Delta\mathbf{x}^T\mathbf{K}^T\mathbf{K}\Delta\mathbf{x} \tag{4.60}$$

where $\Delta\mathbf{y} =$ y_tl%bangle is output by ropp_fm_bangle_1d_tl, and $\mathbf{K} = \partial\mathbf{H}/\partial\mathbf{x}$ again.

An adjoint of the state variables x_ad of type State1dFM is defined and initialised to 0. An adjoint of the bending angle value y_ad is initialised to the output of the tangent linear code, $\mathbf{K}\Delta\mathbf{x}$. The adjoint of the 1D operator routine is then called, thus:

```
y_ad%bangle(:) = y_tl%bangle(:)
CALL ropp_fm_bangle_1d_ad(x, x_ad, y, y_ad)
```

The norm of the linearised perturbation in state space, norm2, is defined as

$$\texttt{norm2} = \mathbf{K}^T\mathbf{K}\Delta\mathbf{x}\cdot\Delta\mathbf{x} = (\mathbf{K}^T\mathbf{K}\Delta\mathbf{x})^T\Delta\mathbf{x} = \Delta\mathbf{x}^T\mathbf{K}^T\mathbf{K}\Delta\mathbf{x} \tag{4.61}$$

where $\mathbf{K}^T\mathbf{K}\Delta\mathbf{x} = \{$x_ad%temp, x_ad%pres, x_ad%geop, x_ad%shum$\}$ is output by ropp_fm_bangle_1d_ad.

If the adjoint and tangent linear routines are correctly coded, then, as Eqns (4.60) and (4.61) show, norm1 and norm2 both equal $\Delta\mathbf{x}^T\mathbf{K}^T\mathbf{K}\Delta\mathbf{x}$. The adjoint test routine t_fascod_ad outputs norm1, norm2 and (norm1-norm2)/norm2 to stdout, as well as a ‘FAIL’ message if the fractional difference between the two norms exceeds 10^{-9}.

References

Abramowitz, M. and Stegun, I. A., eds., *Handbook of mathematical functions*, Dover, 1965.

ECMWF, IFS documentation - Cy31r1. Part III: Dynamics and numerical procedures, IFS documentation, ECMWF, https://www.ecmwf.int/en/elibrary/17735-part-iii-dynamics-and-numerical-procedures, 2006.

Healy, S. B. and Thépaut, J.-N., Assimilation experiments with CHAMP GPS radio occultation measurements, *Quart. J. Roy. Meteorol. Soc.*, *132*, 605–623, 2006.

Mahoney, M. J., A discussion of various measures of altitude, https://wahiduddin.net/calc/refs/measures_of_altitude_mahoney.html, 2001.

NASA/NIMA, NASA and NIMA joint geopotential model EGM96 website, http://cddis.nasa.gov/926/egm96/egm96.html.

NGA/WGS84, National Geospatial-Intelligence Agency WGS84 website, https://earth-info.nga.mil/GandG/update/index.php.

ROM SAF, Geodesy calculations in ROPP, SAF/GRAS/METO/REP/GSR/002, 2007.

ROM SAF, Improvements to the ROPP refractivity and bending angle operators, SAF/ROM/METO/REP/RSR/015, 2013a.

ROM SAF, A review of the geodesy calculations in ROPP, SAF/ROM/METO/REP/RSR/014, 2013b.

ROM SAF, The Radio Occultation Processing Package (ROPP) 1D–Var module User Guide, SAF/ROM/METO/UG/ROPP/007, Version 11.0, 2021a.

ROM SAF, The Radio Occultation Processing Package (ROPP) Input/Output module User Guide, SAF/ROM/METO/UG/ROPP/002, Version 11.0, 2021b.

ROM SAF, The Radio Occultation Processing Package (ROPP) Pre-processor module User Guide, SAF/ROM/METO/UG/ROPP/004, Version 11.0, 2021c.

ROM SAF, The Radio Occultation Processing Package (ROPP) Utilities module User Guide, SAF/ROM/METO/UG/ROPP/008, Version 11.0, 2021d.

Simmons, A. J. and Burridge, D. M., An energy and angular-momentum conserving vertical finite-difference scheme and hybrid vertical coordinates, *Mon. Wea. Rev.*, *109*, 758–766, 1981.

5 Two-dimensional forward model

5.1 Background

The bending angle and refractivity forward models described in Chapter 4 are one-dimensional. This means that the simulated observations are computed using NWP information at a single location and this can introduce "horizontal gradient errors". The use of two-dimensional operators can reduce these forward model errors, particularly in the lower troposphere. The ROPP forward model module, `ropp_fm`, contains a two-dimensional (2D) bending angle forward model, (`ropp_fm_bangle_2d`). This model simulates bending angles from co-located NWP profiles of pressure, temperature, humidity and geopotential height, extracted at a series of points within a "2D occultation plane". The 2D occultation plane is defined in terms of the latitude and longitude of the occultation point, and an azimuthal angle relative to north, which is provided with the observation. The geometry of the measurement within the 2D occultation plane is illustrated in Figure 5.1.

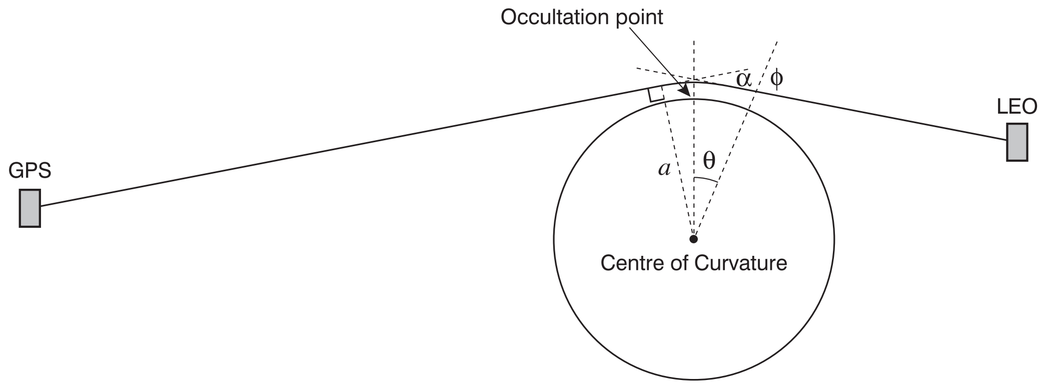


Figure 5.1: Illustrating the geometry of the GNSS-RO measurement.

The 2D forward model has been tested in extended forecast impact experiments with the ECMWF NWP assimilation system. Results at ECMWF suggest that the standard deviations of observed minus background bending angle departures in the lower troposphere can be reduced by around 7 % with a 2D operator. Further details are provided by Healy et al. (2007).

5.2 Theoretical basis of 2D operator

The 2D bending angle operator is more complex than the 1D version because the radial refractivity gradients, $\frac{\partial n}{\partial r}$, vary as a function of horizontal position, θ, within the 2D occultation plane. In this more general case, the ordinary differential equations defining the two-dimensional ray-path in circular polar co-ordinates (r and θ) are given by (page 149, Rodgers, 2000) by

$$\frac{dr}{ds} = \cos\phi \tag{5.1}$$

$$\frac{d\theta}{ds} = \frac{\sin\phi}{r} \tag{5.2}$$

$$\frac{d\phi}{ds} = -\sin\phi\left[\frac{1}{r} + \frac{1}{n}\left(\frac{\partial n}{\partial r}\right)_\theta\right] + \frac{\cos\phi}{nr}\left(\frac{\partial n}{\partial\theta}\right)_r \tag{5.3}$$

where s is the distance along the ray-path, n is the refractive index, ϕ is angle between the local radius vector and the tangent to the ray-path. The ray path equations are integrated numerically.

The ROPP 2D bending angle operator applies a 4th order Runge-Kutta approach to solve the ray path equations where the gradients along the ray path are expected to be large in the lower and mid-troposphere, but reverts to a 1D calculation from the upper-troposphere upwards. This "hybrid" approach reduces the computational costs, but note that the user is able to perform a fully 2D calculation if required.

The integration of the ray path equations starts from the assumed tangent point location, rather than one of the satellites. Furthermore, the user should be aware of two approximations currently used in the 2D bending angle calculation:

1. Tangent point drift is neglected, meaning the tangent points of all the bending angles are assumed to have the same horizontal location.

2. The observed impact parameter is used to determine the tangent point height.

These will be addressed in future releases of the 2D operator.

5.3 ROPP 2D forward model tool

A stand-alone tool `ropp_fm_bg2ro_2d` is provided in `ropp_fm` as an illustration of how the `ropp_fm` routines can be implemented to derive profiles of bending angle with the 2D operator from background meteorological data. This is based on the `ropp_fm_bg2ro_1d` stand-alone tool and it provides a convenient template for describing the 2D operator code. Figure 5.2 shows how the `ropp_fm` routines are integrated in the `ropp_fm_bg2ro_2d` code.

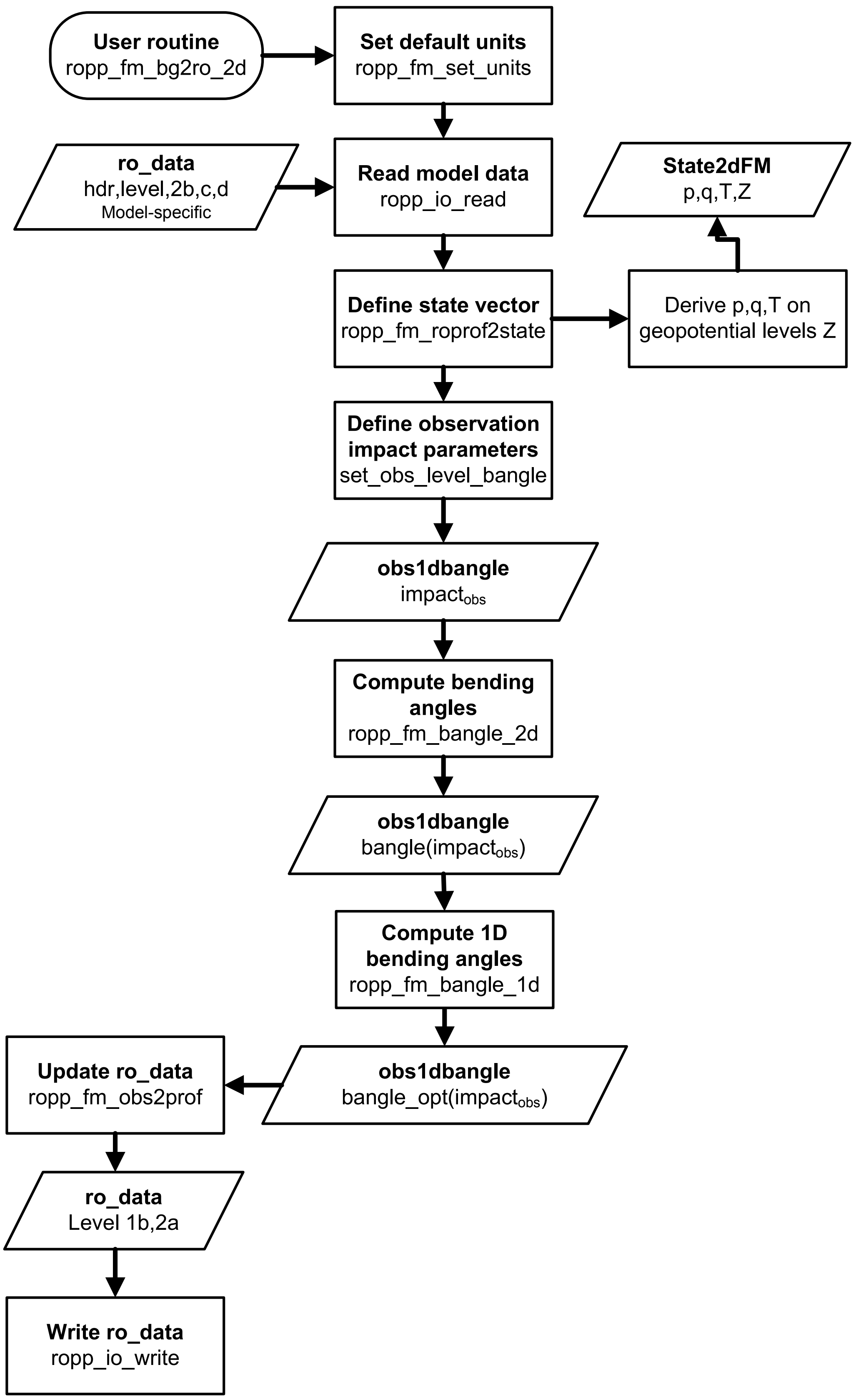


Figure 5.2: Flow chart illustrating calling tree of the ROPP forward model to compute bending angle profiles from input background model data using the 2D operator.

5.3.1 Implementation

The ropp_fm_bg2ro_2d tool is run using the command

```
ropp_fm_bg2ro_2d <inputdatafile> -o <outputfile>
```

where <inputdatafile> is a ROPP netCDF file (ROM SAF, 2021) containing the input model data and <outputfile> will contain the forward modelled bending angle profiles on pre-defined observation levels.

The input file contains header, level 2b and level 2c data, the last two being extended in the horizontal, which is defined by the netCDF dimension dim_horiz. The locations of the level 2b profiles that comprise the 'slice' of input data are specified by the variables lat_2d and lon_2d. See Sec 5.5 for more details of the 2D input data. We would advise users to examine the example test file ropp_fm/data/ECMWF_10_OCCS.nc, which is contained in the ROPP distribution, to see precisely what input data are needed. The output variables are the same as in the 1D forward model (see Sec 4).

If the input file is a multi-file containing more than one model profile, or more than one input file is specified, ropp_fm_bg2ro_2d computes the forward model for each profile in turn and an output file is generated containing all the output profiles.

The following command line options can be used with the ropp_fm_bg2ro_2d tool:

-o <outputfile>	ROPP netCDF file for output retrieved profiles
-comp	use non-ideal gas compressibility options in forward model
-check_qsat	include check against saturation in forward model
-nocheck_qmin	do not include check against negative humidities in forward model
-d	output additional diagnostic information (VerboseMode)
-h	give help menu
-v	output version information

5.3.2 Code organisation

Figure 5.2 shows how the ropp_fm_bg2ro_2d tool is composed of the following stages:

- **Input data access and transformation to a generic state vector**

  Setup the input data arrays, read the input data and define a 2D state vector structure State2dFM containing the background pressure p, temperature T and humidity q data as a function of geopotential height Z. If check_qsat is in force, q is limited by the saturation value at that temperature and pressure; otherwise, by default, supersaturation is allowed. If nocheck_qmin is in force, q is allowed to be negative; otherwise, by default, such 'superdryness' is forbidden. Section 5.5.

- **Define the observation levels for which the bending angles are to be calculated**

  This stage follows from the requirement of the forward model to compute results on levels corresponding to the observation heights for a given occultation. The observation vector Obs1dBangle is either defined using the observation heights specified in the input file, if these exist, or with default levels specified using subroutine set_obs_levels_bangle.

- **Compute bending angle profile**

  ropp_fm_bangle_2d is 2D forward model routine to compute bending angle α as a function of impact parameter a. See Section 5.7.

- **Write results to generic RO data structure and output file**

5.4 Data types

The observation state vector y and background state vector x are represented within ROPP by Fortran 90 derived types (or structures). These data structures are defined in ropp_fm_types.

5.4.1 Observation vector: Obs1DBangle

Bending angle profiles are represented by the Obs1DBangle data structure. This structure has been appended to include new variables required for the 2D code. The elements are listed in Table 4.1 (Section 4.2.1).

5.4.2 State vector: State2dFM

Background meteorological data are represented by the State2dFM data structure. Its elements are listed in Table 5.1. Note that x%lat and x%lon are arrays (rather than single values as in the 1D case) and they contain the locations of x%n_horiz profiles in the 2D occultation plane. The x%n_horiz profiles are equally spaced in the plane, with an angular spacing of x%dtheta radians. x%n_horiz should be an odd number, with the central location corresponding to the location that would be used in a 1D calculation. ropp_fm includes a tool ropp_2d_plane for calculating the latitude and longitude of a series of equally spaced points in a plane which has been defined by y%lat, y%lon and y%azimuth. The tool can be called with

```
USE ropp_fm
INTEGER :: ierror
TYPE(State2dFM) :: x
TYPE(Obs1DBangle) :: y
CALL ropp_2d_plane(y%lat,y%lon,y%azimuth,x%dtheta,x%n_horiz,x%lat,x%lon,ierror)
```

ierror is a returned error flag, with a non-zero value denoting a failure in the routine.

Two-dimensional arrays of temperature x%temp(:,:), pressure x%pres(:,:) and humidity x%shum(:,:) are stored together with the corresponding geopotential height levels x%geop(:,:). The number of background vertical levels is stored in the x%n_lev element and, as noted, the number of horizontal locations in the plane is x%n_horiz. So, for example the temperature information in the occultation plane is stored as a 2D array of size x%temp(n_lev,n_horiz).

A structure holding the 2D background data can be declared as

```
USE ropp_fm
TYPE(State2dFM) :: x
```

State2dFM			
Structure element	Description	Typical range	Units
...%temp	Temperature	150 to 350	K
...%shum	Specific humidity	0 to 0.05	kg/kg
...%pres	Pressure	10 to 110000	Pa[a]
...%geop	Geopotential height	-1000 to 100000	gpm
...%ak	ECMWF-specific level coefficient	0 to 200000	Pa[a]
...%bk	ECMWF-specific level coefficient	0 to 2	
...%n_lev	Number of vertical model levels	1 to 100	
...%n_horiz	Number of horizontal locations	1 to 101	
...%dtheta	Angular separation of profiles	0 to $\pi/2$	Radians
...%lon	Longitude	-180 to +180	° E
...%lat	Latitude	-90 to +90	° N
...%time	Time	–	Jul sec
...%state_ok	Flag to specify state vector ok	.FALSE. or .TRUE.	
...%non_ideal	Non-ideal gas flag	.FALSE. or .TRUE.	
...%check_qsat	Check against saturation	.FALSE. or .TRUE.	
...%check_qmin	Check against dryness	.FALSE. or .TRUE.	

[a] Note that pressure variables in State2dFM are in units of Pa while standard units in the ROprof structure is hPa. See ROM SAF (2021) for details. Users must specify pressure variables in Pa or ensure unit conversion is carried out before performing ropp_fm calculations if using ropp_io for reading input data. Routine ropp_fm_set_units is provided for this task.

Table 5.1: Elements of the State2dFM vector structure

Further details are provided in Section 4.4. Elements x%ak and x%bk are required for this mapping if State2dFM is holding a state vector using hybrid vertical levels (e.g. ECMWF) as described in Section 4.5.

5.5 Defining the state vector: ropp_fm_roprof2state

The ropp_io module routine ropp_io_read reads a single profile of model background data from a netCDF ROPP netCDF input file and fills the elements of the generic ROPP data structure type ROprof2d (ROM SAF, 2021).

```
USE ropp_io
TYPE(ROprof2d) :: ro_data
CALL ropp_io_read(ro_data, inputfilename, rec=iprofile)
```

The relevant background model data are copied to the state vector x of data type State2dFM by calling the routine ropp_fm_roprof2state.

```
USE ropp_io
USE ropp_fm
TYPE(ROprof2d)  :: ro_data
TYPE(State2dFM) :: x
CALL ropp_fm_roprof2state(ro_data, x)
```

A number of checks on the input data are also conducted at this stage. The state vector `x%state_ok` flag is set to false if any test fails. See Section 4.4 for further details.

The state vector is required to contain temperature, specific humidity and pressure data as a function of geopotential height (Table 5.1). These data might be read in directly from the input file if the user provides the background data in this form. It is more likely however that elements of the state vector need to be calculated from the available background data provided by a user and the exact specification of the vertical level representation adopted in the NWP model. The type of model data contained in the input file is specified by the input variable `ro_data%Lev2d%level_type`. `ropp_fm` contains 2D routines to derive the required generic state vector using background data from a NWP model where the vertical coordinate is based on pressure levels (e.g. ECMWF, Section 4.5). Users may prefer to implement their own functions to cater for the specific background vertical level type and available data provided, as these routines are usually part of a data assimilation system.

5.6 Defining the observation vector: `ropp_fm_roprof2obs`

When implemented as part of a data assimilation system, `ropp_fm` is required to compute bending angle profiles from the background data on the observed impact parameter levels.

If observations are read from a file into the `ROprof2d` structure, the routine `ropp_fm_roprof2obs` can be used to define the observation variables of type `Obs1dBangle`. See Section 4.7 for further details.

5.6.1 Set observation impact parameter levels `set_obs_levels_bangle`

The `set_obs_levels_bangle` subroutine provided as part of the stand-alone `ropp_fm_bg2ro_2d` tool provides a simple way of defining the observation impact parameters for which the forward modelled bending angle parameters profiles are to be computed. The observation vector impact parameter element `y%impact` is defined by 250 evenly spaced vertical levels. The lowest impact height is 2700 m and the spacing is 200 m. The bending angle values `y%bangle` are initialised to zero.

5.7 Bending angle forward model `ropp_fm_bangle_2d`

The `ropp_fm` module computes a profile of bending angles α as a function of impact parameter a using a 2D observation operator (Healy et al., 2007).

The input background data are contained in the state vector `x` of type `State2dFM` (Section 4.4) and the output bending angle values are contained in an element of an observation vector `y` of type `Obs1dBangle` (Section 4.2.1).

```
USE ropp_fm
TYPE(State2dFM)   :: x
TYPE(Obs1dBangle) :: y
CALL ropp_fm_bangle_2d(x, y)
```

The main steps of 2D the operator are illustrated in Figure 5.3 and they can be summarised as follows:

1. Compute the water-vapour partial pressure on the (2D grid) model levels.
2. Compute the refractivity, N, on the model levels.
3. Compute the geometric height, h, and radius, r, values the model levels.
4. Compute the product of refractive index and radius, nr, on the model levels.
5. Compute the bending angles, α, as a function of observed impact parameter, a.

Note that steps 1 to 4 are a straightforward generalisation of the 1D code. Essentially, the 1D profile calculations are performed at the `n_horiz` horizontal locations in the plane. However, the details of the calculations will be repeated here for clarity.

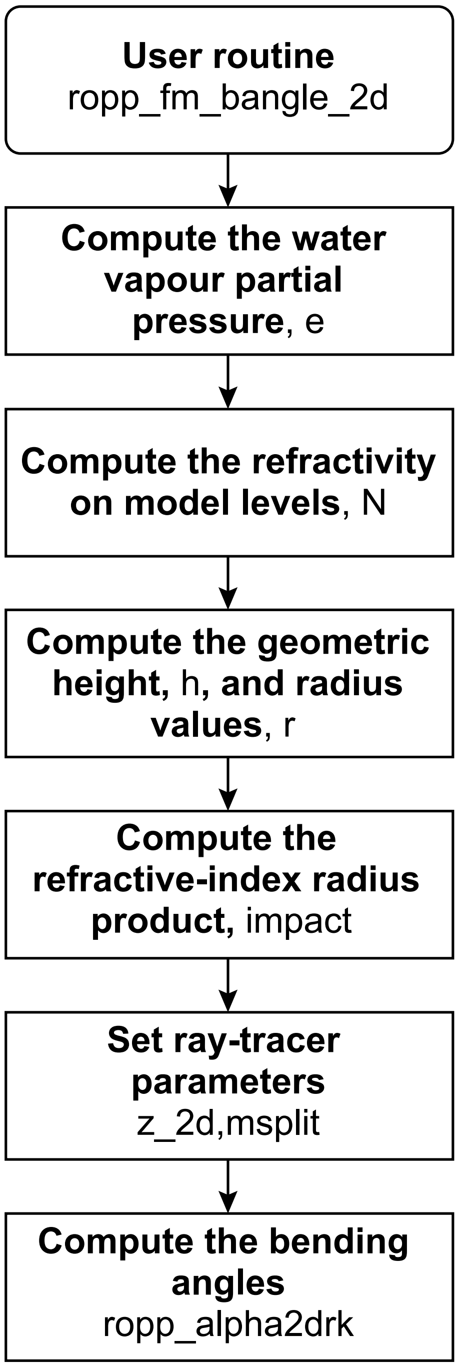

User routine
ropp_fm_bangle_2d
Compute the water vapour partial pressure, e
Compute the refractivity on model levels, N
Compute the geometric height, h, and radius values, r
Compute the refractive-index radius product, impact
Set ray-tracer parameters z_2d,msplit
Compute the bending angles ropp_alpha2drk

Figure 5.3: Flow chart illustrating the main components of the 2D bending angle operator.

5.7.1 Compute partial water vapour pressure

The water vapour pressure e on each model level in the 2D plane is derived from the definition of specific humidity q as

$$q = \frac{r}{1+r}$$

where r is the dry mixing ratio of water vapour defined as

$$r = \varepsilon_w \left(\frac{e}{p-e} \right)$$

This leads to a relationship for e in terms of the background specific humidity `x%shum`$=q$ and pressure `x%pres`$=p$ as

$$e = p \left(\frac{q}{\varepsilon_w + (1-\varepsilon_w)q} \right) \tag{5.4}$$

5.7.2 Compute refractivity

The refractivity N on the model levels from background profiles of pressure `x%pres`, temperature `x%temp` and partial water vapour pressure using

$$N = k_1 \frac{P}{T} + k_2 \frac{e}{T^2} + k_3 \frac{e}{T} \tag{5.5}$$

The refractivity constants k_1, k_2 and k_3 are defined in `ropp_fm_constants`.

5.7.3 Compute geometric heights and radius values

The geopotential heights Z are converted into geometric heights h using the latitude-dependent conversion

$$h(Z,\phi) = \frac{R_{\text{eff}} Z}{(g/g_{\text{wmo}})R_{\text{eff}} - Z} \tag{5.6}$$

where R_{eff} is the Earth's effective radius and g is the surface gravity, both at latitide ϕ (Mahoney, 2001). The radius values are given by

$$r = h + u + R_c \tag{5.7}$$

where u is the undulation and R_c is the radius of curvature.

5.7.4 Compute background impact parameter

The impact parameter (or refractive-index radius product) corresponding to each model level in the 2D plane is computed using the refractivity values N

$$\texttt{impact} = nr = (1 + 1 \times 10^{-6} N) r \tag{5.8}$$

5.7.5 Calculate bending angles `ropp_fm_alpha2drk`

Two parameters are currently "hardwired" in the `ropp_fm_bangle_2d` code. The integer `msplit=4` is used to govern the step-size used in the ray-tracer. Increasing `msplit` reduces the step size and increases the cost of the operator. The current setting appears to reasonable with ECMWF and Met Office vertical grids. The real `z_2d` defines the height above which the bending is calculated with a 1D approximation. This is currently set to 10 km.

The solution of Eqns (5.1)–(5.3) in the bending angle forward model is performed in subroutine `ropp_fm_alpha2drk`. It computes bending angle `y%bangle` as a function of observation impact parameter `y%impact` from the 2D planar refractivity values. This is called within `ropp_fm_bangle_2d`

```
USE ropp_fm
TYPE(State2DFM) :: x
TYPE(Obs1dBangle) :: y
CALL ropp_fm_alpha2drk(y%nobs, x%n_lev, x%n_horiz, msplit, x%dtheta, y%impact, &
                       refrac, rad, impact, y%r_curve, z_2d, a_path, y%bangle, y%rtan)
```

The variables `a_path` and `y%rtan` are the simulated values of $nr\sin\phi$ values at the ray end-points and the tangent radius values, respectively.

As noted earlier, the user should be aware of two approximations currently used in the 2D bending angle calculation:

1. Tangent point drift is neglected, meaning the tangent points of all the bending angles are assumed to have the same horizontal location.
2. The observed impact parameter is used to determine the tangent point height.

The computation of the bending angles within `ropp_fm_alpha2drk` proceeds as follows. The horizontal location of the tangent point is initialised to the middle profile of the 2D plane (`ikcen`). The radius of the ray tangent point is derived from the observed impact parameter, a, and the background (model) refractivity values at the central profile using $r_t = a/(1+10^{-6}N)$. Note that if the tangent point of the ray is above user-prescribed `z_2d`, the 2D ray tracer defaults to a pure 1D calculation of the bending angle. However, if the tangent point of the ray is below the user-prescribed `z_2d`, we estimate the ray bending for the section of path from the tangent point up to a height of `z_2d` with a numerical, raytracing solution of the of the differential equations that define the ray path. The ray tracer starts the calculation at the assumed tangent point, rather than one of the satellites. To evaluate the total bending, we have to compute the bending associated with the ray path on both sides of the tangent point and then add the two values together.

The bending along either side of the tangent point is calculated as follows (Figure 5.4). A four element

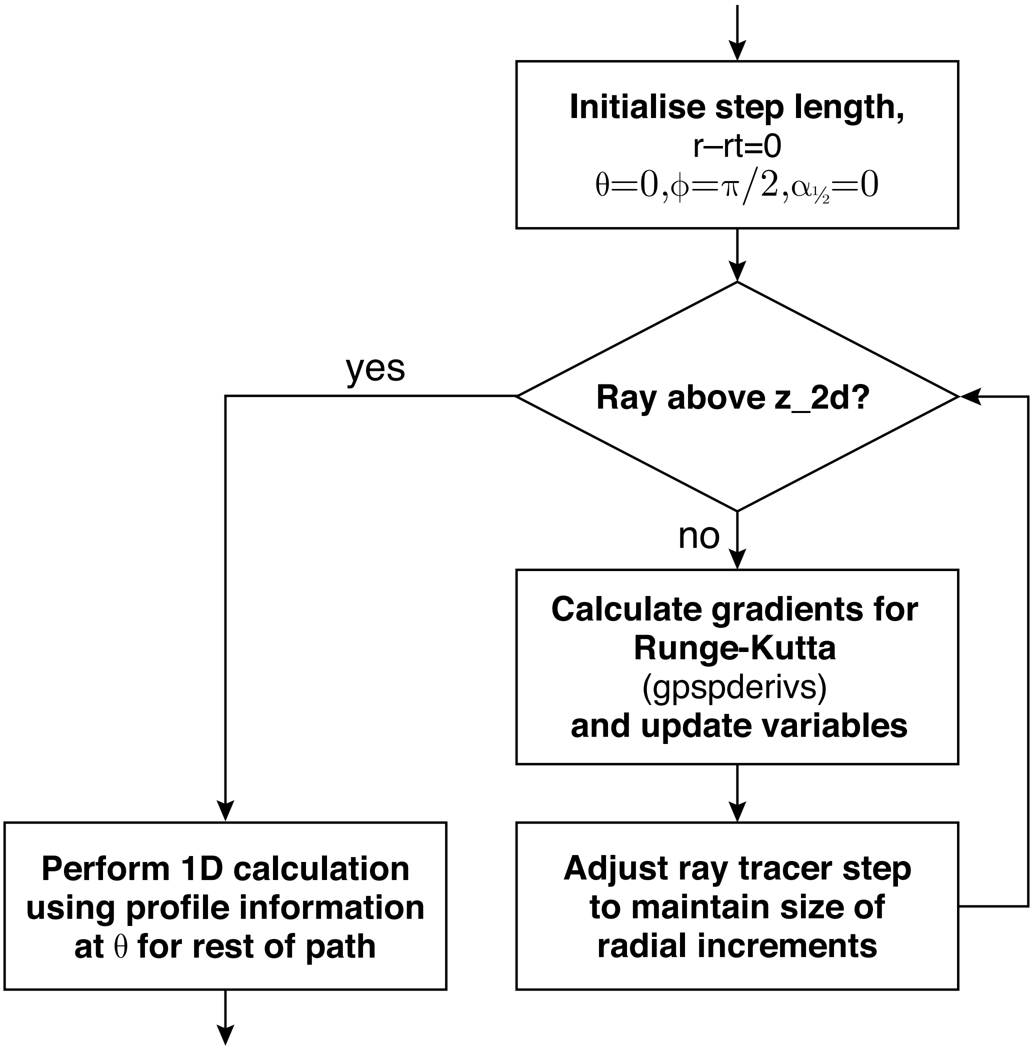

Initialise step length,
r–rt=0
θ=0,φ=π/2,α½=0
Ray above z_2d?
yes
no
Calculate gradients for Runge-Kutta (gpspderivs) and update variables
Adjust ray tracer step to maintain size of radial increments
Perform 1D calculation using profile information at θ for rest of path

Figure 5.4: Flow chart illustrating the calculation of the bending angles with the Runge-Kutta.

vector $\mathbf{zy}(:)$ containing the following variables $(r-r_t, \theta, \phi, \alpha_{1/2})$ is initialised as

$$\begin{aligned} r - r_t &= 0 \\ \theta &= 0 \\ \phi &= \frac{\pi}{2} \\ \alpha_{1/2} &= 0 \end{aligned} \tag{5.9}$$

where $r-r_t$ is the height above the tangent point, θ is the angular position in the plane, ϕ is the angle between the local radius vector and the ray-path and $\alpha_{1/2}$ is the bending angle along the section of ray path. The subscript "$1/2$" indicates that this is bending from just one side of the path. Note that we adopt the convention $\theta = 0$ at the horizontal location of the tangent point.

The numerical solution of the ray path equations is performed with a 4th order Runge-Kutta routine (RK4) (e.g., section 16.1, Press et al., 1992). The derivatives of the four variables, at an arbitrary point in the plane (r, θ) are evaluated in `gpspderivs`. This routine calculates the following gradients:

$$\begin{aligned}
\frac{d(r-r_t)}{ds} &= \cos\phi \\
\frac{d\theta}{ds} &= \frac{\sin\phi}{r} \\
\frac{d\phi}{ds} &\simeq -\sin\phi\left[\frac{1}{r}+\left(\frac{\partial n}{\partial r}\right)_\theta\right] \\
\frac{d\alpha_{1/2}}{ds} &= -\sin\phi\left(\frac{\partial n}{\partial r}\right)_\theta
\end{aligned} \tag{5.10}$$

where s is the distance along the ray-path and n is the refractive index. In gpspderivs we linearly interpolate the profile information to θ and then assume that refractivity varies exponentially between the model levels to calculate the radial gradient. A maximum refractivity gradient of 0.15 N units/m is also imposed.

As is standard for 4th order Runge-Kutta integration schemes, there are four calls to gpspderivs per integration step (eq. 16.1.3, Press et al., 1992), and then a weighted average of the gradient is calculated. This weighted average gradient is used to project the zy(:) vector forward one step, using Eqns 5.10. The code is set up so that there are msplit=4 steps between vertical model levels, and the step lengths zh are adjusted to achieve roughly equal radial increments per step between model levels. The iterations are repeated, with the local bending angle with respect to the horizontal, $\alpha_{1/2}$ being steadily accumulated. When the ray height exceeds z_2d, it is assumed that the refractivity field is spherically symmetric, so that the remaining bending can be calculated by the familiar 1D bending angle equation,

$$\Delta\alpha_{1d}(a) = -a\int_{R_c+z_{2d}}^{\infty} \frac{\mathrm{d}\ln n/\mathrm{d}x}{(x^2-a^2)^{1/2}}\,\mathrm{d}x \tag{5.11}$$

This integral is evaluated using the nearest model profile to θ, the angular position where the ray height equals z_2d. The solution is given in terms of the Gaussian error function (Eqn 4.57 in Section 4.10).

The whole procedure is repeated for the other 'half' of the occultation, with a ray starting from the the tangent point but going in the opposite direction. The sum of the two bending angles gives the total calculated bending angle.

5.7.6 Bending angle forward model gradient

The ropp_fm module includes routines to compute tangent linear and adjoint of the 2D bending angle forward model with respect to the state vector. These tangent linear and adjoint routines are called ropp_fm_bangle_2d_tl and ropp_fm_bangle_2d_ad, respectively. Some examples of their implementation are outlined in section 5.12.

5.8 Copy simulated observations to RO structure ropp_fm_obs2roprof

The computed bending angle observation vectors are copied to the Level2a and Level1b parts of the ROprof data structure respectively for writing to the output netCDF data file using ropp_fm_obs2roprof (Section 4.11).

```
USE ropp_fm
TYPE(State2dFM)  :: ro_data
TYPE(Obs1dBangle) :: obs_bangle
CALL ropp_fm_obs2roprof(obs_bangle, ro_data)
```

5.9 Comparing with a 1D bending angle operator

The `ropp_fm_bg2ro_2d` tool also calculates bending angles with the 1D bending angle operator to enable comparison between the two forward models. `ropp_fm_roprof2state` is used to copy the centre profile of the 2D section of level2a data into a `state1dFM` structure. The 1D bending angles are then calculated with `ropp_fm_bangle_1d` (Section 4.10). The resulting profiles are written into the `ro_data%lev1b%impact_opt` and `ro_data%lev1b%bangle_opt` elements of the output `ROprof` structure.

5.10 Writing out data

The results are written to the specified output file using the `ropp_io` module routine `ropp_io_write`.

5.11 Plotting tools

The directory `tests/` contains the IDL procedure `plot_fm_twod.pro` which gives an example of how to read and plot refractivity and bending angle profiles computed by running the forward model.

An example of its implementation is provided by running the test script `test_fm_2D.sh`.

5.12 Test software tools

In addition to the main stand alone simulation tool `ropp_fm_bg2ro_2d`, we provide three additional simple test programs that the user may find instructive. These are in the directory `tests/` and are called:

- `t_twodop`: a simple call to the 2D operator and comparison with bending angles computed at ECMWF.
- `t_twodtl`: a test of the 2D operator tangent linear code.
- `t_twodad`: a test of the 2D operator adjoint code.

These programs do not use the `ropp_io` modules.

5.12.1 `t_twodop`

This simple program reads meteorological and observation data from a text file, simulates bending angles with the 2D operator, `ropp_fm_bangle_2d`, and compares the results with bending angles simulated at ECMWF with the same data and 2D operator. Test data are provided in the text file `data/ECMWF_2D_DATA.DAT`.

If the fractional differences are greater than 0.0001 a failure message is written to screen, along with the simulated bending angle values that have failed the test.

5.12.2 t_twodtl

This program provides an example of how to call and test the tangent linear of the 2D bending angle code, ropp_fm_bangle_2d_tl. The 2D operator is called to compute simulated bending angles, which are saved to a new structure.

```
CALL ropp_fm_bangle_2d(x,y)
y_save%bangle(:) = y%bangle(:)
```

A set of random perturbations between 0 and 1 are stored in a x_tl variable structure of type Obs1dBangle (the geopotential perturbation is increased a factor of 100). The code then loops through the following procedures 15 times. Firstly, the perturbations are added to the original state:

```
x_new%temp(:,:) = x%temp(:,:) + x_tl%temp(:,:)
x_new%pres(:,:) = x%pres(:,:) + x_tl%pres(:,:)
x_new%geop(:,:) = x%geop(:,:) + x_tl%geop(:,:)
x_new%shum(:,:) = x%shum(:,:) + x_tl%shum(:,:)
```

and the 2D operator is called with the perturbed state vector. The tangent linear routine is then called with the original vector and the perturbation

```
CALL ropp_fm_bangle_2d(x_new,y_new)
CALL ropp_fm_bangle_2d_tl(x,x_tl,y,y_tl)
```

The code then compares the change in bending angle produced with of the tangent linear routine, y_tl%bangle with the finite difference approximation y_new%bangle(:) - y_save%bangle(:), calculating the cosine of the angle between the vectors (the dot product divided by the magnitude of the vectors) and evaluating the relative error. These are then output to screen.

The perturbations are then reduced by a factor of 10 and the procedure is repeated. Ideally, the cosine of the angle between the vectors should tend to unity as the size of the perturbation is reduced, but it does not reach unity because of numerical error. However, how close it gets to unity is a good test of the operator/tangent linear consistency. A failure message is written to screen if the tangent linear test does not converge towards unity.

5.12.3 t_twodad

The t_twodad program tests the consistency of the tangent linear and adjoint code, routines ropp_fm_bangle_2d_tl and ropp_fm_bangle_2d_ad, respectively. The 2D operator is called to compute simulated bending angles.

```
CALL ropp_fm_bangle_2d(x,y)
```

A set of random perturbations between 0 and 1 are stored in a x_tl variable structure of type Obs1dBangle (the geopotential perturbation is increased a factor of 100) and the tangent linear routine is called.

```
CALL ropp_fm_bangle_2d_tl(x,x_tl,y,y_tl)
```

We compute the matrix expression $\Delta\mathbf{y}^{\mathbf{T}}\Delta\mathbf{y}$, where $\Delta\mathbf{y} = \mathbf{H}\Delta\mathbf{x}$, given $\mathbf{H}$ is the matrix representing the linearised operator and $\Delta\mathbf{x}$ is the perturbation to the state.

```
norm1 = DOT_PRODUCT(y_tl%bangle(:),y_tl%bangle(:))
```

An adjoint of the state variables x_ad of type State2dFM is defined and initialised to 0. An adjoint of the bending angle value y_ad is initialised to the output of the tangent linear code. The adjoint of the 2D operator routine is then called.

```
y_ad%bangle(:) = y_tl%bangle(:)
CALL ropp_fm_bangle_2d_ad(x,x_ad,y,y_ad)
```

We then define norm2 in terms of x_ad and the original perturbation x_tl. Mathematically, norm2 is equivalent to the matrix expression $\mathbf{x}_{ad}^{\mathbf{T}}\Delta\mathbf{x}$, where $\mathbf{x}_{ad} = \mathbf{H}^{\mathbf{T}}\Delta\mathbf{y}$ and is the output of the adjoint routine. If the adjoint and tangent linear routines are consistent, by definition, norm1=norm2. The routine outputs norm1, norm2 and norm1/norm2 to screen. A failure message is written to screen if the ratio of norms is not close to unity.

References

Healy, S. B., Eyre, J. R., Hamrud, M., and Thépaut, J.-N., Assimilating GPS radio occultation measurements with two-dimensional bending angle observation operators, *Quart. J. Roy. Meteorol. Soc.*, *133*, 1213–1227, 2007.

Mahoney, M. J., A discussion of various measures of altitude, https://wahiduddin.net/calc/refs/measures_of_altitude_mahoney.html, 2001.

Press, W., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P., *Numerical recipes in C – The Art of Scientific Computing*, Cambridge University Press, Cambridge, New York, 2nd edn., 1992.

Rodgers, C. D., *Inverse methods for atmospheric sounding: Theory and practice*, World Scientific Publishing, Singapore, New Jersey, London, Hong Kong, 2000.

ROM SAF, The Radio Occultation Processing Package (ROPP) Input/Output module User Guide, SAF/ROM/METO/UG/ROPP/002, Version 11.0, 2021.

6 Including non-ideal gas effects

6.1 Background

In their default configuration, the ROPP 1D and 2D forward operators assume an ideal gas when computing the hydrostatic integration and the refractivity values from pressure, temperature and humidity. However, Aparicio et al. (2009) have recently demonstrated that neglecting the non-ideal gas effects can be a source of forward model error.

We have introduced an non-ideal gas option in all of the 1D and 2D forward models. The inclusion of non-ideal gas effects is inextricably linked to the choice of refractivity coefficients used in the forward calculation. We have chosen the "best average" refractivity coefficients provided by Rueger (2002), but with a k_1 coefficient adjusted for use in a refractivity expression that includes non-ideal gas compressibility (Thayer, 1974). A discussion on the uncertainty of the coefficients can be found in Cucurull (2010) and Healy (2011).

6.2 Effects of non-ideal gas compressibility

The ROPP refractivity and bending angle observation operator contains essentially three major components. The first is the integration of the hydrostatic equation, to compute the height of the model levels. The second is the evaluation of refractivity on the model levels, and the third is either interpolating the model refractivity to the observed refractivity height, or the evaluation of the bending angle integral, for the refractivity and bending angle operators, respectively.

Non-ideal gas effects should be included in both the integration of the hydrostatic equation and the evaluation of the refractivity on the model levels. Note that Aparicio et al. (2009) do not include the impact of non-ideal gas effects on the evaluation of the refractivity on the model levels.

Following Aparicio et al. (2009) the non-ideal gas equation of state can be written as,

$$P = \rho R T_v Z \tag{6.1}$$

where P is the total pressure, ρ is the density, R is the gas constant for dry air, T_v is the virtual temperature and Z is the compressibility of moist air. The compressibility accounts for non-ideal gas effects, such as the finite size of the molecules and mutual attraction, and it is a function of pressure, temperature and water vapour pressure. For an ideal gas $Z = 1$, but for air in the troposphere typically $Z \sim 0.9995$, so the departure is 0.05 %. Picard et al. (2008) provide a polynomial expansion for Z, which is straightforward to implement in the ROPP observation operators.

If we include the compressibility in the hydrostatic integral, the geopotential height, h, becomes

$$h = -\int \left(\frac{ZRT_v}{g_o P} \right) dp \tag{6.2}$$

where $g_o = 9.80665\text{ms}^{-2}$. Given that $Z < 1$, it is clear that including compressibility reduces the height of model levels. In the ROPP operators, the thickness between the ith and $(i+1)th$ model levels initially calculated neglecting non-ideal gas effects, denoted by $\Delta h'_{i,i+1}$, is scaled by the average of the compressibility at ith and $(i+1)th$ levels, to give $\Delta h_{i,i+1} = 0.5 \times (Z_i + Z_{i+1}) \times \Delta h'_{i,i+1}$. This approach is equally valid for ECMWF and Met Office height based vertical grids.

To put these adjustments in some context, in simulations we have found that the height of model levels near 100 hPa can be reduced by $dh \sim 7\text{m}$. Assuming a refractivity scale height of 7 km, $\Delta N/N = -dh/7\text{km}$, this translates into a reduction in the forward modelled refractivity or bending angle value of -0.1%. that Rüeger's k_1 value is 0.115 % larger than Smith and Weintraub (1953). should also be included in the expression of refractivity.

The second step where non-ideal effects should be included is the computation of refractivity on the model levels. Using a general expression that includes non-ideal gas compressibility (Thayer, 1974)

$$N = \frac{k_1 P_d}{Z_d T} + \frac{k_2 e}{Z_w T^2} + \frac{k_3 e}{Z_w T} \tag{6.3}$$

where Z_d and Z_w are the compressibility of the dry air and water vapour, respectively. Again we use the Picard et al. (2008) polynomial to determine the dry and wet compressibilities. Note that the introduction of compressibility in this expression increases the refractivity value, whereas introducing compressibility in the hydrostaic integration reduces the value.

When running the ROPP observation operators with non-ideal gas effects included, we use the "best average" coefficients given by Rueger (2002). However, we adjust the best average k_1 value to account for its use in a refractivity expression that includes compressibility (Healy, 2011). This reduces the value by around 0.05 % to $k_1 = 77.643\ \text{KhPa}^{-1}$.

6.3 Forward model code structure

The introduction of non-ideal gas effects has required some modification of the forward operators. The 1D and 2D statevectors `State1dFM` and `State2dFM` contain a logical switch `x%non_ideal`. (These are set to .FALSE. in `ropp_fm_types.f90` by default.)

We can use sections of the `ropp_fm_refrac_1d` code to demonstrate the structure, since the same approach is used in both `ropp_fm_bangle_1d` and `ropp_fm_bangle_2d`:

```
! set inverse of compressibilities

  zcomp_dry_inv(:) = 1.0_wp
  zcomp_wet_inv(:) = 1.0_wp
```

```
! initialise geopotential heights

  z_geop(:) = x%geop(:)

  IF (x%non_ideal) THEN

! if non ideal gas calculation, use adjusted coefficients

     kap1 = kappa1_comp
     kap2 = kappa2_comp
     kap3 = kappa3_comp

!    calculate compressibility and adjust geopotential heights in z_geop

     CALL ropp_fm_compress(x,z_geop,zcomp_dry_inv,zcomp_wet_inv)

  ELSE

     kap1 = kappa1
     kap2 = kappa2
     kap3 = kappa3

  ENDIF
```

To summarize the above, the inverse of the dry and wet compressibility values are initialised to unity, and we copy the model level geopotential height values into `z_geop(:)`. The code then tests `x%non_ideal` and if `.TRUE.` selects the (modified) Rueger (2002) coefficients and calls `ropp_fm_compress`. This routine reduces the model geopotential height values in `z_geop(:)`, and calculates the inverse dry and wet compressibilities.

Since the operators use a three term refractivity expression, we then calculate the dry air and vapour pressure,

```
  pwvp = x%pres * x%shum / (epsilon_water + (1.0_wp - epsilon_water)*x%shum)

  pdry = x%pres - pwvp
```

The refractivity on the model levels is then calculated with the three term expression

```
  refrac = kap1 * pdry * zcomp_dry_inv/ x%temp +  &
           kap2 * pwvp * zcomp_wet_inv/ x%temp**2 + &
           kap3 * pwvp * zcomp_wet_inv/ x%temp
```

As noted, this expression defaults to the standard two-term Smith and Weintraub (1953) formula when `x%non_ideal=.FALSE.` since `kap1=kap3` and the inverse compressibility factors are set to unity.

For the alternative refractivity interpolation (Section 4.8.5), a new subroutine, `ropp_fm_compress_single_level` has been created to calculate the compressibility factors at the interpolated heights.

Note that the corresponding tangent linear and adjoint routines have been modified to be consistent with these changes in the forward model.

6.4 Running ROPP FM routines with non-ideal effects

The ROPP forward model tools `ropp_fm_bg2ro_1d` and `ropp_fm_bg2ro_2d` have been modified to include a new command line argument `-comp`. So, for example, the 2D bending angle operator tool, `ropp_fm_bg2ro_2d`, can be run using the command:

```
ropp_fm_bg2ro_2d <inputdatafile> -comp -o <outputfile>
```

where `<inputdatafile>` is a ROPP netCDF file (ROM SAF, 2021) containing the input model data and `<outputfile>` will contain the forward modelled bending angle profiles on pre-defined observation levels, computed with non-ideal gas effects included in the calculation.

The following test routines also run with a `-comp` command-line option:

- `t_fascod`
- `t_fascod_tl`
- `t_fascod_ad`
- `t_twodop`
- `t_twodad`
- `t_twodtl`

The test scripts

- `ropp_fm/test/test_fm_1D.sh`
- `ropp_fm/test/test_fm_2D.sh`
- `ropp_fm/test/test_fm_TWOD.sh`
- `ropp_fm/test/test_fm_FASCOD.sh`

have been extended to test this functionality.

References

Aparicio, J., Deblonde, G., Garand, L., and Laroche, S., The signature of the atmospheric compressibility factor in COSMIC, CHAMP and GRACE radio occultation data, *J. Geophys. Res.*, p. doi:10.1029/2008JD011156, 2009.

Cucurull, L., Improvement in the use of an operational constellation of GPS radio-occultation receivers in weather forecasting, *Weather and Forecasting*, *25*, 749–767, 2010.

Healy, S., Refractivity coefficients used in the assimilation of GPS radio occultation measurements, *J. Geophys. Res.*, *116*, D01 106, doi:10.1029/2010JD014 013, 2011.

Picard, A., Davis, R., Glaser, M., and Fujii, K., Revised formula for the density of moist air (CIPM-2007), *Metrologia*, *25*, 149–155, 2008.

Rueger, J. M., Refractive index formulae for electronic distance measurement with radio and millimetre waves, Unisurv Report S-68. School of Surveying and Spatial Information Systems, University of New South Wales, [Summary at http://www.fig.net/pub/fig_2002/Js28/JS28_rueger.pdf], 2002.

ROM SAF, The Radio Occultation Processing Package (ROPP) Input/Output module User Guide, SAF/ROM/METO/UG/ROPP/002, Version 11.0, 2021.

Smith, E. K. and Weintraub, S., The constants in the equation for atmospheric refractivity index at radio frequencies, in *Proc. IRE*, vol. 41, pp. 1035–1037, 1953.

Thayer, G. D., An improved equation for the refractive index of air, *Radio Sci.*, *9*, 803–807, 1974.

7 Modelling L1 and L2 bending angles

7.1 Background

ROPP has the facility to forward model L1 and L2 bending angles, instead of the ionospherically corrected neutral bending angle which is used elsewhere throughout the `ropp_fm` and `ropp_1dvar` modules. It does this by assuming a simple model ionosphere comprising a single Chapman layer (Chapman, 1931).

This facility allows the user to:

- Interpolate or extrapolate L1 and L2 using a simple physical model of the difference between them;
- Examine the sensitivity of L1 and L2 bending angles to variations in ionospheric parameters;
- Make 1D–Var retrievals using the less heavily processed L1 and L2 bending angles directly, rather than the ionospherically corrected neutral bending angle.

The background theory will be discussed first, followed by the modifications to the `ropp_fm` modules.

7.2 Theory

ROM SAF report 17 (ROM SAF (2015)) gives details of the calculation of the bending induced by a spherically symmetric single Chapman layer electron density distribution. In summary it says the following.

As in Sec 3.2, the bending angle α at impact height a in a spherically symmetric medium is given by

$$\alpha(a) = -2a \int_a^\infty \frac{1}{\sqrt{x^2 - a^2}} \frac{d\log(n)}{dx} \mathrm{d}x \tag{7.1}$$

where $x = nr$. In the ionosphere at RO frequencies, the dominant contribution to the refractive index is proportional to the electron density n_e and inversely proportional to the square of the frequency f, thus:

$$\log n \approx n - 1 \approx -k_4 n_e / f^2 \tag{7.2}$$

where $k_4 = 40.3$ m^{3}s^{-2} (eg Kursinski et al. (1997)).

In a Chapman layer peaking at height r_0, with maximum electron density n_e^{max} and width H, the electron density n_e as a function of height r is given by

$$n_e(r) = n_e^{max} \exp\left(\frac{1}{2}(1 - u - e^{-u})\right) \tag{7.3}$$

where $u = (r - r_0)/H$ (Chapman, 1931). The more familiar total electron content, TEC, for a Chapman layer is given by

$$\text{TEC} = \int n_e(r) dr = \sqrt{2\pi e} H n_e^{max}. \tag{7.4}$$

For typical ionospheric electron densities $n_e \sim 10^{11}$ m^{-3} at RO frequencies $f \sim 10^9$ Hz, Eqn (7.2) implies $n-1 \sim 10^{-6}$. We therefore replace x by r in the integrand of Eqn (7.1) and obtain, after substituting Eqn (7.2),

$$\begin{aligned} \alpha(a,f) &= -2a \int_a^\infty \frac{d\log n/dx}{\sqrt{r^2-a^2}} \mathrm{d}r & (7.5)\\ &= 2a\frac{k_4}{f^2} \int_a^\infty \frac{dn_e/dr}{\sqrt{r^2-a^2}} \mathrm{d}r & (7.6)\\ &\approx 2a\frac{k_4}{f^2}\frac{2r_0}{(r_0+a)^{3/2}} \int_a^\infty \frac{dn_e/dr}{\sqrt{r-a}} \mathrm{d}r & (7.7) \end{aligned}$$

where we approximately factorise $\sqrt{r^2-a^2}$ as $\sqrt{r-a}(r_0+a)^{3/2}/2r_0$ in the last step to ensure that Eqns (7.6) and (7.7) give the same result for an infinitely thin model ionosphere, $n_e(r) = \text{TEC}\,\delta(r-r_0)$, which is an appropriate limit for this problem. Ionospheric bending angles calculated this way differ from (exceed) those calculated according to the more usual factorisation $\sqrt{r^2-a^2} \approx \sqrt{2a(r-a)}$ (eg Eqn (4.35)) by no more than 1%.

Substituting Eqn (7.3) into Eqn (7.7) gives, eventually,

$$\alpha_i(a) = \alpha(a,f_i) = \frac{k_4}{f_i^2} n_e^{max} \sqrt{\frac{4er_0^2a^2}{H(r_0+a)^3}} Z\left(\frac{r_0-a}{H}\right) \tag{7.8}$$

where the dimensionless, order 1, function Z is defined by

$$Z(l) = \int_{-l}^{\infty} \frac{(e^{-3u/2}-e^{-u/2})\exp(-\frac{1}{2}e^{-u})}{\sqrt{u+l}} \mathrm{d}u. \tag{7.9}$$

The function Z describes most of the variation of bending angle with height, since the other factors in Eqn (7.8) only vary slowly over the range of impact parameters a of interest. Crucially, Z depends only on the parameter $l = (r_0-a)/H$, the distance from the peak of the Chapman layer expressed in widths of the layer. For practical RO applications, $2 \lesssim l \lesssim 10$.

$Z(l)$ is sketched in Fig 7.1, as are the following limits, which can be inferred from Eqn (7.9):

$$Z(l) \sim \begin{cases} -\sqrt{2\pi e^l} & \text{as } l \to -\infty \\ \sqrt{2\pi/l^3} & \text{as } l \to \infty \end{cases} \tag{7.10}$$

Neither limiting approximation is accurate enough for the required range of l (as is clear from Fig 7.1), but the following Padé approximation in the transformed variable θ is much quicker to evaluate than direct numerical integration and is accurate to within 2.2% for all l:

$$Z(l) \approx \tilde{Z}(\theta) = \sqrt{2\pi\theta}\, \frac{p_0+p_1\theta+p_2\theta^2+p_3\theta^3}{q_0+q_1\theta+q_2\theta^2+q_3\theta^3+q_4\theta^4+q_5\theta^5} \tag{7.11}$$

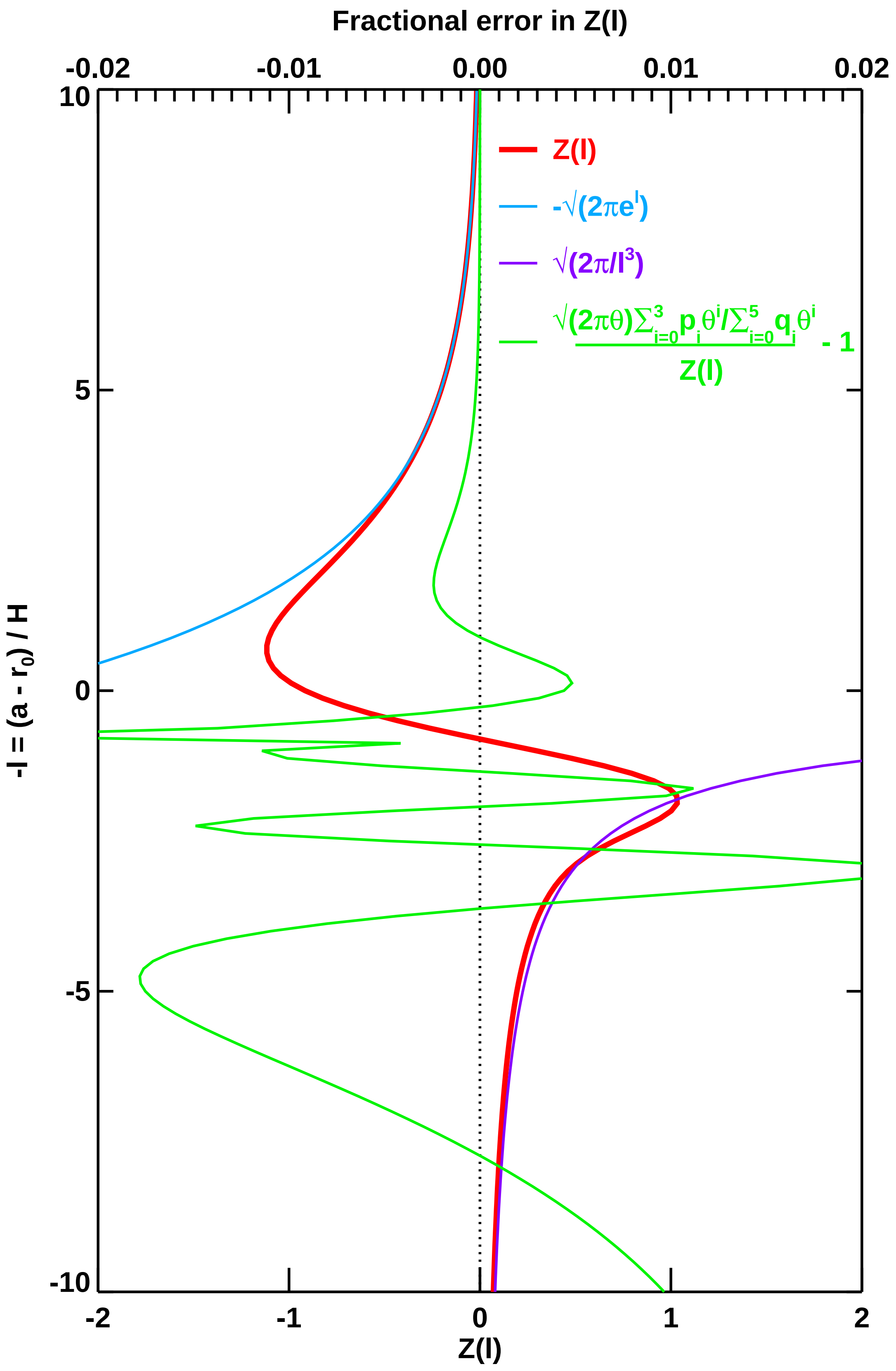


Figure 7.1: $Z(l)$ (calculated numerically at high precision), its asymptotic limits for large positive and negative l, and the fractional error in the Padé approximation to it.

where $\theta(l) = \sinh^{-1}(e^l/2)$ and the coefficients $p_0 \ldots q_5$ are listed in Table 7.1. The conditions $p_0 = -\sqrt{2}$ and $p_3 = q_5$ encapsulate the the correct asymptotic behaviours of $Z(l)$ as given in Eqn (7.10). This lends confidence to the integrity to the solution. The other coefficients are obtained by a least squares fitting procedure. Eqn (7.11) is easy to differentiate, which is useful when writing tangent linear and adjoint codes.

Padé coefficients	
p_0	-1.41421360 $(=-\sqrt{2})$
p_1	2.32540970
p_2	-1.11628850
p_3	0.23605387
q_0	1.00000000 $(=1)$
q_1	0.15210651
q_2	-0.76649105
q_3	1.26080520
q_4	-0.84687066
q_5	0.23605387 $(=p_3)$

Table 7.1: Padé coefficients in Eqn (7.11)

Eqn (7.11) is hard to distinguish by eye from Eqn (7.9) (which is calculated to high precision numerically), so the fractional difference between them is plotted in Fig 7.1. (The biggest error in the region not shown is about 1.8% at $l \approx 20$.)

7.2.1 Effects of finite electron density at LEO

Single frequency bending angle calculations should include the effects of a non-zero electron density at the LEO, because (see later) these can change the bending angles by up to 30% at CHAMP altitudes of around 450 km, and 5% even at COSMIC altitudes of 800 km. There are two aspects to these corrections: the removal of the contribution to the bending above the height of the LEO, which has been erroneously included in Eqn (7.9); and the effect of using the correct (non-unity) refractive index at the LEO when deriving bending angles from Doppler shifts. Both corrections are inversely proportional to the square of the radio frequency, which means that their effect on the neutral bending angle is eliminated by the usual ionospheric correction method.

Removal of bending angle above LEO

Appendix A.2 of ROM SAF report 17 (ROM SAF (2015)) describes a series approximation to $Z(l)$ (Eqn (7.9) above) that is valid for heights far above the peak of the ionospheric electron density. In fact it is valid at all heights, but is difficult to compute numerically for impact parameters far below the ionospheric peak, as in the troposphere and stratosphere, which is why the Padé approximation of Eqn (7.11) was developed. The approximation is, however, very suitable for this application, because the ionospheric electron density would probably peak at around 300 km for an F-layer, and lower for a D-layer. And the series is quite rapidly convergent above the electron density peak, so that only a few terms are needed to get an accurate approximation to the bending at the LEO. Full details of an extension of the theory of ROM SAF report 17, to accumulate just the bending produced by refractivity gradients above the LEO rather than the bending produced by refractivity gradients above the impact parameter, are provided in Section 4.2 of ROM SAF report 33 (ROM SAF (2018)). The ROPP routine `ropp_fm/iono/ropp_fm_iono_bangle_above_leo.f90` encodes the following three term expression — Eqn 4.18 of RSR 33 rewritten in the above notation — for the contribution to the bending angle above the LEO, which needs to be *subtracted* from Eqn (7.8) to account for the finite height of the LEO:

$$\Delta\alpha_i^{(1)}(a) = -\frac{k_4 n_e^{max} a}{f_i^2}\sqrt{\frac{e\pi g_{LEO}}{H(r_0+a)}}\sum_{r=0}^{2}\frac{(-g_{LEO}/2)^r\sqrt{r+1/2}}{r!}\mathrm{erfcx}\left(\sqrt{(r+1/2)(l-l_{LEO})}\right) \tag{7.12}$$

where $l_{LEO} = (r_0 - r_{LEO})/H < 0$ is the l-coordinate at the location of the LEO, $g_{LEO} = \exp(l_{LEO})$ and $\mathrm{erfcx}(x) = \exp(x^2)\mathrm{erfc}(x)$. Note the f_i^{-2} scaling, which ensures the effect is eliminated by the usual dual frequency ionospheric correction. Note too the rapid exponential decrease with r_{LEO}, for large r_{LEO}.

Corresponding tangent linear and adjoint codes also exist in the same directory.

For a typical ionospheric F-layer with a strength of 100 TECU ($= 10^{18}$ electrons/m^2) and $H = 75$ km, peaking at 300 km, Eqn (7.12) shows that $\Delta\alpha_i^{(1)}$ is almost constant at around -10 μrad from the surface up to 80 km, assuming the LEO to be sited at 800 km (e.g. COSMIC). This is around 2% of the total L1 or L2 bending at 80 km. Naturally, the effect would be larger if the LEO were lower down (e.g. CHAMP at around 450 km). Since $\Delta\alpha_i^{(1)}$ is negative, and has to be subtracted from the $r_{LEO} \to \infty$ bending angle, this correction increases the forward modelled bending angles.

Correction of refractive index at LEO

Section 4.1 of ROM SAF report 33 (ROM SAF (2018)) explains that, at least for circular orbits, the impact parameter $a = nr\sin\phi$ derived from Doppler shifts is correct even in the presence of finite electron density at the LEO. This constancy of a then implies a correction to ϕ, and therefore to the bending angle, when these are derived from Doppler shifts — see, for example, Sec 3.3.2 of the ROPP PP module User Guide (ROM SAF (2021)). Eqn 4.4 of ROM SAF report 33 shows that because we assume $n_{LEO} = 1$ in the observational processing, the derived bending angles are too small by an amount

$$\Delta\alpha_i^{(2)}(a) = \frac{k_4}{f_i^2}\frac{a}{\sqrt{r_{LEO}^2 - a^2}}n_e(r_{LEO}) \tag{7.13}$$

This correction appears in `ropp_fm/iono/ropp_fm_iono_bangle.f90`, and its tangent linear and adjoint counterparts are in the same directory.

For the typical ionospheric F-layer described above, Eqn (7.13) shows that $\Delta\alpha_i^{(2)}$ is constant at around $+10$ μrad up to 80 km, assuming the LEO to be sited at 800 km (e.g. COSMIC). Once again, therefore, this is a few per cent of the total L1 or L2 bending at 80 km. And, also once again, it would be larger for a lower LEO, such as CHAMP.

Because the observationally derived bending angles are too small by $\Delta\alpha_i^{(2)}$, we need to reduce the forward modelled bending angles by the same amount. The overall correction to the bending angles given by Eqn 7.8

is therefore

$$\Delta\alpha_i(a) = -\Delta\alpha_i^{(1)}(a) - \Delta\alpha_i^{(2)}(a) \tag{7.14}$$

For the typical ionospheric F-layer described above, the residual bending angle correction given by Eqn 7.14 is constant at around -0.5 μrad from the surface up to 80 km. Despite the near cancellation of $\Delta\alpha_i^{(1)}$ and $-\Delta\alpha_i^{(2)}$, it is worth retaining both contributions, in case one of the effects has already been accounted for in some way. The other would then need to be available to prevent a potentially significant systematic error from developing.

7.3 ropp_fm module

7.3.1 Implementation

The L1 and L2 forward modelled bending angles can be calculated by calling the ROPP one-dimensional forward modelling tool `ropp_fm_bg2ro_1d` with the `-direct_ion` flag, thus:

```
ropp_fm_bg2ro_1d -f -direct_ion bgr_input_file.nc -o output_file.nc
```

(`-f` prevents calculation of gradients, which cannot be generated in this case anyway.)

The bending angle theory described above is implemented in ROPP by extending the `Lev2c` subtructure of the `ROprof` data structure to include the ionospheric parameters n_e^{max}, r_0 and H, and their errors (needed by the 1D–Var module), thus:

```
TYPE L2ctype
  ...
  REAL(wp)  :: Ne_max         = ropp_MDFV ! Peak elec.density (m-3)
  REAL(wp)  :: Ne_max_sigma   = ropp_MDFV ! Error in peak elec.density (m-3)
  REAL(wp)  :: H_peak         = ropp_MDFV ! Height of peak elec. dens. (m)
  REAL(wp)  :: H_peak_sigma   = ropp_MDFV ! Error in height of peak elec. dens. (m)
  REAL(wp)  :: H_width        = ropp_MDFV ! Width of Chapman layer (m)
  REAL(wp)  :: H_width_sigma  = ropp_MDFV ! Error in width of Chapman layer (m)
  LOGICAL   :: direct_ion     = .FALSE.   ! Model L1 and L2 directly if .TRUE.
  ...
```

Note that the logical flag `ROprof%Lev2c%direct_ion` has also been introduced to record whether the direct modelling of L1 and L2 bending angles has been invoked.

The ionospheric parameters are held in the variables `Ne_max`, `H_peak` and `H_width` in the background netCDF files that form the input to the forward model. For example:

```
ncdump -vH_width,Ne_max,H_peak bgr_input_file.nc

  netcdf bgr_input_file {
```

```
...
      float H_width(dim_unlim) ;
              H_width:long_name = "Chapman layer width" ;
              H_width:units = "m" ;
              H_width:valid_range = 0., 1000000. ;
      float Ne_max(dim_unlim) ;
              Ne_max:long_name = "Peak Chapman layer electron density" ;
              Ne_max:units = "m-3" ;
              Ne_max:valid_range = 0., 1.e+15 ;
      float H_peak(dim_unlim) ;
              H_peak:long_name = "Height of Chapman layer peak" ;
              H_peak:units = "m" ;
              H_peak:valid_range = 0., 1000000. ;
...
data:
 H_width = 75000 ;
 Ne_max = 3e+11 ;
 H_peak = 300000 ;
}
```

If any of these parameters are unset or equal to `ropp_MDFV`, `ropp_fm/iono/ropp_fm_iono_set_default.f90` will set them equal to these corresponding default values:

```
REAL(wp), PARAMETER  :: ne_max_default          = 300.0e9_wp ! m-3
REAL(wp), PARAMETER  :: ne_max_sigma_default    = 200.0e9_wp ! m-3
REAL(wp), PARAMETER  :: h_peak_default          = 300.0e3_wp ! m
REAL(wp), PARAMETER  :: h_peak_sigma_default    = 150.0e3_wp ! m
REAL(wp), PARAMETER  :: h_width_default         =  75.0e3_wp ! m
REAL(wp), PARAMETER  :: h_width_sigma_default   =  25.0e3_wp ! m
```

If `-direct_ion` has been invoked, then `ropp_fm/bangle_1d/ropp_fm_bangle_1d.f90` calculates the ionospheric contributions to the total L1 and L2 bending angles, α_1 and α_2 respectively, by calling `ropp_fm/iono/ropp_fm_iono_bangle_1d.f90`, which evaluates Eqn (7.8) for the two frequencies. ('r_0' in this equation is found by adding the input peak height of the Chapman layer, `H_peak`, to the appropriate local radius of curvature R_c.) The two ionospheric bending angles are added to the neutral bending angle, which is calculated as before, and held internally in an *extended* observation vector: `bangle = (bangle_L1, bangle_L2)`. This concatenation of α_1 and α_2 is hidden from the user, because the extended bending angle observation vector is 'unpacked' (by `ropp_fm/iono/ropp_fm_iono_unpack_bangle.f90`) before being output into the `bangle_L1` and `bangle_L2` elements of the `Lev1b` substructure of the `ROprof` structure. The neutral bending angle, $\alpha_n = (f_1^2\alpha_1 - f_2^2\alpha_2)/(\alpha_1 - \alpha_2)$, is also returned (in `ROprof%Lev1b%bangle`), as usual.

7.3.2 Testing

The 'make test' step, as applied when building `ropp_fm` with `buildpack`, has been extended to test the `-direct_ion` functionality by means of four new tests:

- **t_iono**, which compares newly generated L1 and L2 bending angles against precomputed reference values (which are included in the distribution);
- **t_iono_tl**, which assesses the tangent linear code `ropp_fm_iono_bangle_tl.f90` by comparing the derivative of the forward model against the difference in the full non-linear operator at two states, in which the ionospheric parameters (only) are slightly perturbed, analogously to the procedure described in Sec 4.13.2;
- **t_iono_ad**, which assesses the adjoint code `ropp_fm_iono_bangle_ad.f90` by comparing the norms in model space and observation space of the difference which appears when (only) the ionospheric parameters are perturbed, analogously to the procedure described in Sec 4.13.3;
- **test_fm_iono.sh**, which runs `ropp_fm_bg2ro_1d` through 5 ECMWF background files with and without the `-direct_ion` option, and checks that the neutral bending angles in the files agree, that the L1 and L2 bending angles agree with those held in reference files (included in the ROPP distribution), and that the neutral bending angle inferred from L1 and L2 agrees with the neutral bending angle in the file.

7.3.3 Results

As an example, Fig 7.2 shows α_n, α_1 and α_2 calculated for typical daytime, solar maximum E-, F-, F1- and D-layers (all having the same ECMWF background fields). The parameters of the E- and F-layers are taken from Syndergaard (2000); those of the F1 layer from the Chiu ionospheric model (Chiu, 1975); those of the D-layer are estimated from Friedrich and Torkar (2001). There is clearly a wide range in size and shape of L2 – L1, which is of course independent of the neutral bending angle. The key factor determining the magnitude of $\alpha_2 - \alpha_1$ is probably n_e^{max}; the key factor determining the shape is probably H.

7.3.4 Summary

ROPP can forward model the L1 and L2 bending angles produced by a model Chapman layer ionosphere, whose parameters can be specified by the user. This paves the way for retrievals to be made using L1 and L2 directly. This feature of ROPP is described in the accompanying User Guide on the 1D–Var module.

References

Chapman, S., The absorption and dissociative or ionizing effect of monochromatic radiation of an atmosphere on a rotating earth, *Proc. Phys. Soc.*, *43*, 483–501, 1931.

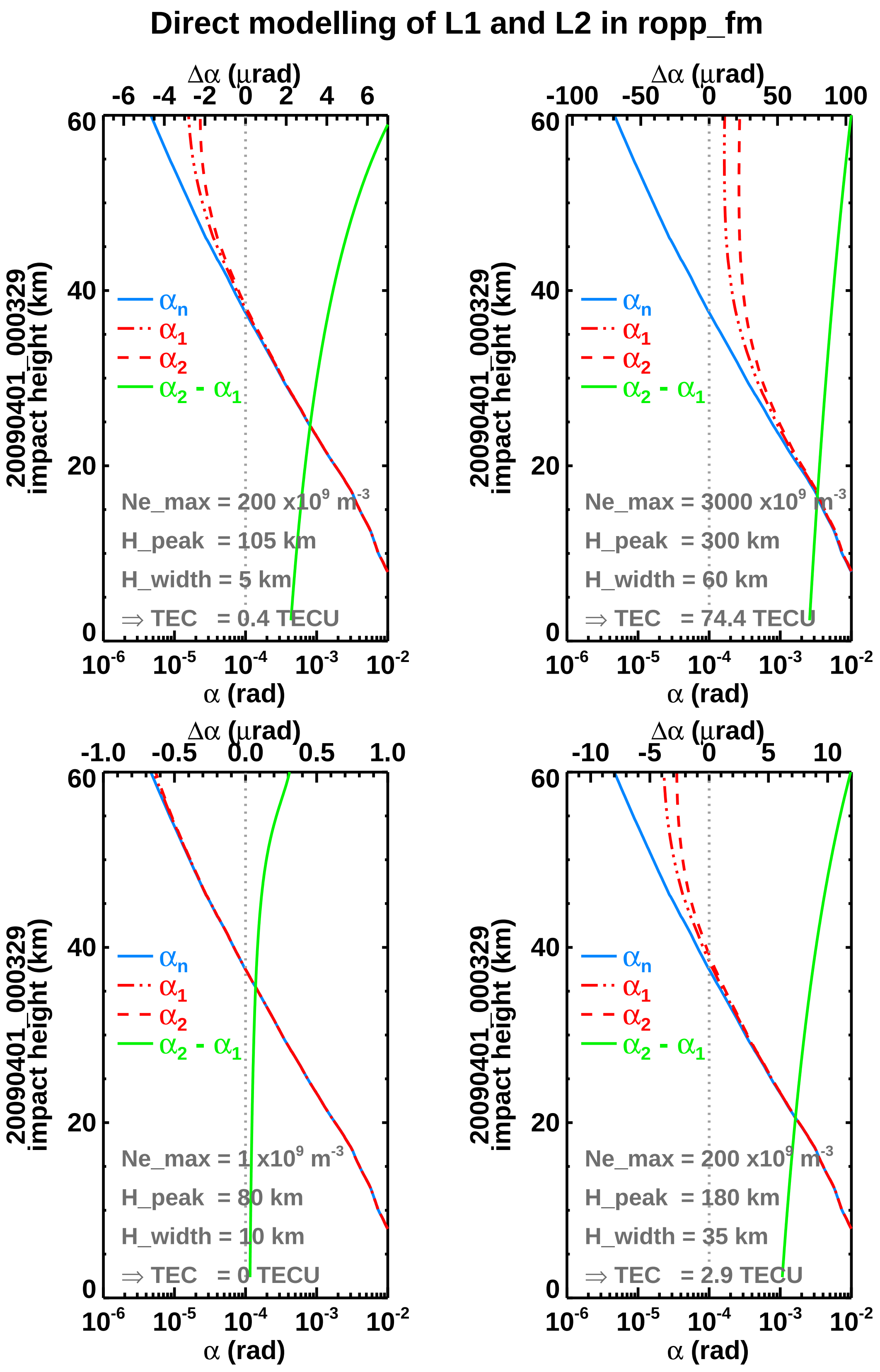


Figure 7.2: Example L1 and L2 bending angles calculated for (clockwise from top left) typical E-, F-, F1- and D-layers, having the indicated ionospheric parameters.

Chiu, Y. T., An improved phenomenological model of ionospheric density, *J. Atmos. Sol.-Terr. Phys.*, *37*, 1563–1570, 1975.

Friedrich, M. and Torkar, K. M., FIRI: A semiempirical model of the lower ionosphere, *J. Geophys. Res.*, *106 (A10)*, 21 409–21 418, 2001.

Kursinski, E. R., Hajj, G. A., Schofield, J. T., Linfield, R. P., and Hardy, K. R., Observing earth's atmosphere with radio occultation measurements using the Global Positioning System, *J. Geophys. Res.*, *102*, 23.429–23.465, 1997.

ROM SAF, Simulation of L1 and L2 bending angles with a model ionosphere, SAF/ROM/METO/REP/RSR/017, 2015.

ROM SAF, Some science changes in ROPP-9.1, SAF/ROM/METO/REP/RSR/033, 2018.

ROM SAF, The Radio Occultation Processing Package (ROPP) Pre-processor module User Guide, SAF/ROM/METO/UG/ROPP/004, Version 11.0, 2021.

Syndergaard, S., On the ionospheric calibration in GPS radio occultation measurements, *Radio Sci.*, *35*, 865–883, 2000.

8 Modelling f2 – f1 bending angles

8.1 Background

ROPP has the facility to forward model the purely ionospheric f2 – f1 bending angle difference. (Note that f2 and f1 are not necessarily equal to L2 and L1.) It does this by assuming a simple model ionosphere comprising multiple 'VaryChap' layers (ROM SAF (2019)). At present this is only used by the 1dvar retrieval tool `ropp_1dvar_dbangle`.

8.2 Theory

The `ropp_fm` directory `ropp_fm/dbangle_1d` contains routines to calculate the electron density defined by a collection of 'VaryChap' model ionospheric layers, the bending angles induced by such a model ionosphere, and its vertically integrated electron content. For further details of the calculations, see (ROM SAF, 2021).

8.2.1 Calculation of electron density

The electron density profile $n_e(r)$ in a VaryChap layer is defined by the four parameters $\{n_e^{\mathsf{max}}, r_0, H_0, k\}$ as:

$$\begin{aligned} n_e(r) &= n_e^{\mathsf{max}} \exp\left((1-u-e^{-u})/2\right)\sqrt{H_0/H}, \quad \text{where} & (8.1)\\ u &= \log\left(H/H_0\right)/k \quad \text{and} & (8.2)\\ H &= H_0 + k(r-r_0). & (8.3)\end{aligned}$$

These equations apply if $r > r_0$. If $r \le r_0$, or if $k \le 10^{-3}$, the 'standard' Chapman layer approximation, which is given by the limit as $k \to 0$ of the above, applies — namely:

$$\begin{aligned} n_e(r) &= n_e^{\mathsf{max}} \exp\left((1-u-e^{-u})/2\right), \quad \text{where} & (8.4)\\ u &= (r-r_0)/H_0. & (8.5)\end{aligned}$$

See (ROM SAF, 2019) and (ROM SAF, 2021) for more details.

The `ropp_fm` routine `ropp_fm_ne_vchap.f90` calculates an electron density profile by summing these expressions for $n_e(r)$ over a number of layers, the parameters (n_e^{max}, H_0, etc) of each being contained in the `State0DFM` state vector structure. The output levels r are taken from the impact parameters in an accompanying `Obs1dBangle` bending angle structure. `ropp_fm` routine `ropp_fm_ne_vchap_tl.f90` is the tangent linear of this routine, i.e. it calculates the change in n_e resulting from a given change in the state vector x. `ropp_fm_ne_vchap_grad.f90` calculates the Jacobian of the transformation, i.e. $\partial \mathbf{n_e}/\partial \mathbf{x}$, by calling `ropp_fm_ne_vchap_tl.f90` multiple times.

8.2.2 Calculation of f2–f1 bending angle difference

The slantwise total electron content (STEC) at impact parameter a between the GNSS and the LEO in the presence of a spherically symmetrical electron density field $n_e(r)$ is given by:

$$\begin{aligned}
\mathsf{STEC}(a) &= \int_{\mathsf{GNSS}}^{\mathsf{LEO}} n_e \, \mathrm{d}s && (8.6)\\
&= \int_a^{r_{\mathsf{LEO}}} \frac{n_e r}{\sqrt{r^2-a^2}} \, \mathrm{d}r + \int_a^{r_{\mathsf{GNSS}}} \frac{n_e r}{\sqrt{r^2-a^2}} \, \mathrm{d}r && (8.7)\\
&= \sqrt{r_{\mathsf{LEO}}^2 - a^2}\, n_e(r_{\mathsf{LEO}}) - \int_a^{r_{\mathsf{LEO}}} \frac{\mathrm{d}n_e}{\mathrm{d}r} \sqrt{r^2-a^2}\, \mathrm{d}r - \int_a^{r_{\mathsf{GNSS}}} \frac{\mathrm{d}n_e}{\mathrm{d}r} \sqrt{r^2-a^2}\, \mathrm{d}r && (8.8)
\end{aligned}$$

upon integrating by parts, and assuming (reasonably) that $n_e(r_{\mathsf{GNSS}}) = 0$.

This implies that

$$\frac{\mathrm{d}\mathsf{STEC}}{\mathrm{d}a}(a) = -\frac{a}{\sqrt{r_{\mathsf{LEO}}^2 - a^2}} n_e(r_{\mathsf{LEO}}) + a \int_a^{r_{\mathsf{LEO}}} \frac{\mathrm{d}n_e/\mathrm{d}r}{\sqrt{r^2-a^2}} \, \mathrm{d}r + a \int_a^{r_{\mathsf{GNSS}}} \frac{\mathrm{d}n_e/\mathrm{d}r}{\sqrt{r^2-a^2}} \, \mathrm{d}r. \qquad (8.9)$$

The first term of Eqn (8.9) is proportional to the correction to the bending angles that results from the finite electron density at the LEO — Eqn (7.13) of Sec 7. (Recall that this bias needs to be *subtracted* from the true bending angles to get the observed bending angles — see (ROM SAF, 2021).) The sum of the second and third term of Eqn (8.9) are effectively proportional to Eqn (7.6) in the same Section, after it is recognised that $2\int_a^\infty$ in Eqn (7.6) is really an approximation to $\int_a^{r_{\mathsf{LEO}}} + \int_a^{r_{\mathsf{GNSS}}}$, which is the correct expression for the ionospheric bending when proper account is taken of the positions of the satellites. This means that the difference between the observed f1 and f2 bending angles, which have both neutral and ionospheric components, can be written as

$$\begin{aligned}
\alpha_{\mathsf{f2}} - \alpha_{\mathsf{f1}} &= (\alpha_{\mathsf{f2}}^{\mathsf{neut}} + \alpha_{\mathsf{f2}}^{\mathsf{iono}}) - (\alpha_{\mathsf{f1}}^{\mathsf{neut}} + \alpha_{\mathsf{f1}}^{\mathsf{iono}}) && (8.10)\\
&\approx \alpha_{\mathsf{f2}}^{\mathsf{iono}} - \alpha_{\mathsf{f1}}^{\mathsf{iono}}, \quad \text{assuming } \alpha_{\mathsf{f2}}^{\mathsf{neut}} \approx \alpha_{\mathsf{f1}}^{\mathsf{neut}} && (8.11)\\
&= k_4 (f_2^{-2} - f_1^{-2}) \frac{\mathrm{d}\mathsf{STEC}}{\mathrm{d}a}, && (8.12)
\end{aligned}$$

where $\mathrm{d}\mathsf{STEC}/\mathrm{d}a$ is given by Eqn (8.9), and after including the proportionality constants of k_4/f_i^2 from Eqns (7.6) and (7.13).

The `ropp_fm` routine `ropp_fm_dbangle_1d.f90` calculates the difference in bending angles according to Eqn 8.12. The integral is estimated by Simpson's rule, with at least 30 points between the tangent point and the LEO (counted twice), and 10 points between the LEO and the GNSS (counted once). The substitution $r = a\cosh(w)$ takes care of the integrable singularity in Eqn 8.9 at $r = a$. `ropp_fm_dbangle_1d_tl.f90` and `ropp_fm_dbangle_1d_grad.f90` respectively calculate the tangent linear and the Jacobian of the bending angle difference with respect to the state vector.

8.2.3 Calculation of vertically integrated total electron content

Eqn 2.11 of ROM SAF Report 31 (ROM SAF (2019)) shows that the vertically integrated total electron content of a 'full' VaryChap layer — that is, one for which Eqns 8.1–8.3 apply for $r > r_0 - H_0/k$ (where n_e vanishes), rather than just for $r > r_0$ — is given by

$$\begin{aligned} \text{VTEC} &= \int_{r_0-H_0/k}^{\infty} n_e(r)\,\mathrm{d}r && (8.13) \\ &= n_e^{\text{max}}\, H_0\, \sqrt{e}\, 2^{(1-k)/2}\, \Gamma\left(\frac{1-k}{2}\right) && (8.14) \end{aligned}$$

where Γ is the gamma function, whose logarithm is returned by the `ropp_utils` function `ropp_utils/math/lngamma.f90`. (Eqn 8.14 reduces to the familiar expression $n_e^{\text{max}} H_0 \sqrt{2\pi e}$ as $k \to 0$, i.e. for a standard Chapman layer.)

The 'standard' Chapman layer behaviour below $r = r_0$, as embodied in Eqns 8.4 and 8.5, complicates matters. It is handled by subtracting the difference between the VaryChap and the Chapman VTECs below r_0:

$$\int_{-\infty}^{\infty} n_e(r)\,\mathrm{d}r = \int_{r_0-H_0/k}^{\infty} n_e^{\text{VChap}}(r)\,\mathrm{d}r - \int_{r_0-H_0/k}^{r_0} n_e^{\text{VChap}}(r)\,\mathrm{d}r + \int_{-\infty}^{r_0} n_e^{\text{Chap}}(r)\,\mathrm{d}r. \qquad (8.15)$$

The first term on the right hand side is given by Eqn 8.14; the second is estimated numerically by means of a simple midpoint scheme with 1000 points; the last is easily shown to equal $n_e^{\text{max}} H_0 \sqrt{2\pi e}\,\text{erfc}(1/\sqrt{2})$.

The `ropp_fm` routine `ropp_fm_ne_vtec.f90` returns the VTEC calculated according to Eqn 8.15.

References

ROM SAF, Sensitivity of some RO measurements to the shape of the ionospheric electron density profile, SAF/ROM/METO/REP/RSR/031, 2019.

ROM SAF, A one-dimensional variational ionospheric retrieval for truncated GNSS Radio Occultation measurements, SAF/ROM/METO/REP/RSR/042, 2021.

A Installing and using ROPP

A.1 Software requirements

ROPP is written in standard Fortran 95. Thus, compilation and use of the routines forming ROPP require the availability of standard ISO-conforming compilers. Fortran 95 was preferred over Fortran 90 because it has a number of convenient features. In particular, it allows elemental functions and pointers can be nullified when they are declared.

A.2 Software release notes

The latest ROPP distribution is available for download via the ROM SAF website http://www.romsaf.org. The ROPP Release Notes available from the ROPP download page and provided with the main ROPP download tarfile gives instructions for unpacking and installing the complete ROPP package, or individual modules. Users are strongly recommended to refer to the ROPP Release Notes and use the build and configure tools described therein. The information contained here is intended to complement the ROPP Release Notes. Where any contradiction between the User Guide and ROPP Release Notes exist, the ROPP Release Notes page is considered to be the most up-to-date latest information.

A.3 Third-party packages

To fully implement ROPP, the code uses some standard third-party packages. These are all non-commercial and cost-free. Note that third-party codes are only needed by the `ropp_utils`, `ropp_io` and `ropp_pp` modules, so are optional if these modules are not required by the user.

All third-party code or packages used by ROPP are, by definition, classed as 'Pre-Existing Software' and all rights remain with the originators. Separate rights licences may be part of these distributions — some may have a licence which may impose re-distribution restrictions — and such licences must be adhered to by users.

If a third-party package is required, this must be built and installed before attempting to build the ROPP code. For convenience, these packages should be installed to the same root path as ROPP. It is highly recommended that the package is compiled using the same compiler and using the same compiler flags as will be used to build the ROPP code. Example configure scripts for supported compilers are provided in the `ropp_build` module available from the ROPP download website. See Section A.4 for further details.

A.3.1 NetCDF (optional in principle)

The input/output library `ropp_io` uses Unidata's `netCDF` data format. Thus, the `netCDF` library and its associated utility programs (like `ncdump`, `ncgen`) are required and must be properly installed on the user's system before the compilation of the `ropp_io` package can be attempted. `netCDF` may also be used for reading MSIS or BAROCLIM climatology data as part of the `ropp_pp` module.

The SAF provides versions of the `netCDF` distribution, which have been successfully integrated with ROPP, alongside the ROPP distribution. This may not be the most recent distribution. Latest versions are freely available from

http://www.unidata.ucar.edu/software/netcdf/

With effect from ROPP9.0, ROPP netCDF build support for 'classic' netCDF-4 has been dropped, which implies a need for HDF5 and, optionally, ZLIB libraries. These last two can be found at

https://support.hdfgroup.org/HDF5/

and

http://www.zlib.net/

respectively.

In addition, the supported versions of the netCDF library are now split into two parts: a netCDF-Core library, written in C, and a netCDF-Fortran interface. The ROPP `buildpack` script (see Sec A.4 for more details) allows installation of these libraries as follows:

```
> buildpack zlib <compiler>
> buildpack hdf5 <compiler>
> buildpack netcdf <compiler>    (the netCDF-Core library)
> buildpack netcdff <compiler>   (the netCDF-Fortran library)
```

These packages need to be installed in this order, since each depends on the previous one. Note, however, that the `zlib` and the `HDF5` libraries may already be installed as part of a standard Linux distribution, in which case, of course, the user need not build a local version.

Note that the `tests` subdirectory of the `ropp_io` distribution contains a simple test to check if the `netCDF` installation works; see Section A.7 for details.

A very useful complementary set of tools for handling and manipulating netCDF data files are the netCDF Operators `nco`[1]. While the latter are not required for using `ROPP` libraries and sample applications, we highly recommend them.

Some example and test programs provided with the `ropp_pp`, `ropp_apps`, `ropp_fm` and `ropp_1dvar` packages read data via `ropp_io`. A complete installation of the `ropp_io` library is therefore required if the test programs or one of the sample applications are to be run. As a consequence, the complete installation of these packages also requires the availability of `netCDF`. Note, however, that the libraries `libropp_pp.a`, `libropp_apps.a`, `libropp_fm.a` and `libropp_1dvar.a` can be compiled and installed without `ropp_io`

[1] See `http://nco.sourceforge.net/`.

and therefore without `netCDF`; the configuration script will recognise the absence of these libraries and only compile and install the core pre-processor, forward model or 1DVar routines (i.e. those with no dependencies on `netCDF` or `ropp_io`).

A.3.2 BUFR (optional)

The GNSS-RO BUFR encoder/decoder tools `ropp2bufr` and `bufr2ropp` in `ropp_io` require either the Met Office's 'MetDB' or the ECMWF BUFR library to be pre-installed. Alternatively, the BUFR encoder/decoder tools `ropp2bufr_eccodes` and `bufr2ropp_eccodes` can be used if the ECMWF ecCodes library is pre-installed. If no BUFR library is detected by the installation configure script, then these tools will not be built.

The tools to BUFR-encode EUMETSAT-format grouped netCDF data, `eum2bufr` and `eum2bufr_eccodes` in `ropp_io`, require the ECMWF BUFR library or ECMWF ecCodes library to be pre-installed, respectively.

The MetDB BUFR package is available without charge on request from the ROPP Development Team but with some licence restrictions. The ECMWF BUFR package is licensed under the GNU/GPL and can be downloaded from:

https://software.ecmwf.int/wiki/display/BUFR

The ECMWF ecCodes package is licensed under Apache (2.0), and can be downloaded from:

https://confluence.ecmwf.int/display/ECC/ecCodes+Home

Note that a small change has been made to the ecCodes tarball supplied with ROPP to suppress the warning message that is produced each time a missing data indicator is set. This change can be made to a user's own copy of the ecCodes library by using the patch provided at `ropp_io/tools/eccodes_patch`.

Both libraries generate essentially identical data when decoded (there may be non-significant round-off differences due to use of single– vs. double–precision interfaces). While the MetDB library is easier to install from a portability point of view, the ROPP `buildpack` script makes the ECMWF installation compatibly with ROPP more transparent. Therefore users can employ whichever BUFR package they prefer. Thus, the MetDB library could be built with

```
> buildpack bufr <compiler>
```

or

```
> buildpack mobufr <compiler>
```

while the ECMWF BUFR library would be be built with

```
> buildpack ecbufr <compiler>
```

and the ECMWF ecCodes library would be be built with

```
> buildpack eccodes <compiler>
```

In order to install BUFR tables and related files, and for the applications to find them at run-time, an environment variable must be pre-defined to the path to these files. For instance, for the MetDB library:

```
> export BUFR_LIBRARY=<path>/data/bufr/
```

or for the ECMWF BUFR library or ecCodes library:

```
> export BUFR_TABLES=<path>/data/bufr/
```

Note that in both cases, the path must currently be terminated with a '/' character, although this restriction has been relaxed for later (v20+) releases of the MetDB BUFR library. By default, the `buildpack` script will set `<path>` to be `ROPP_ROOT`.

A.3.3 GRIB (optional) - either GRIB_API or ecCodes

The GRIB background reading tool `grib2bgrasc` in `ropp_io` requires either the ECMWF GRIB_API library or the ECMWF ecCodes library to be pre-installed. If neither is detected by the installation configure script, then this tool will not be built.

The ECMWF GRIB_API package is licensed under Apache (2.0), and can be downloaded from:

https://software.ecmwf.int/wiki/display/GRIB/

The ROPP `buildpack` script allows installation of the GRIB_API by typing:

```
> buildpack grib <compiler>
```

The ECMWF ecCodes package is licensed under Apache (2.0), and can be downloaded from:

https://confluence.ecmwf.int/display/ECC/ecCodes+Home

The ROPP `buildpack` script allows installation of ecCodes by typing:

```
> buildpack eccodes <compiler>
```

A.3.4 SOFA (optional)

The routines in `ropp_utils` that transform coordinates between reference frames have the option of using the IAU Standards of Fundamental Astronomy (SOFA) library to convert between some frames. If this library is unavailable, less sophisticated formula-based versions of the routines will be used instead.

The SOFA libraries are freely available for use, provided the routines are not modified in any way. They can be downloaded from

http://www.iausofa.org/

The ROPP `buildpack` script allows installation of the SOFA library by typing:

```
> buildpack sofa <compiler>
```

A.3.5 RoboDoc (optional)

The ROPP Reference Manuals have been auto-generated using the RoboDoc documentation tool[2] All source code, scripts, etc. have standardised header comments which can be scanned by RoboDoc to produce various output formats, including LaTeX and HTML. If code (and in particular the header comments) is modified, RoboDoc can optionally be used to update the documentation. This tool is not required in order to build the ROPP software.

A.3.6 autoconf and automake (optional)

The `automake` and `autoconf` tools, common on most Linux and Unix systems, are not necessary to build the ROPP package as provided, but are useful if any modifications are made to the code or build systems to re-generate the package `configure` files. Versions at, or higher than, v1.9 are required to support some of the m4 macros defined in the ROPP build system.

A.4 BUILDPACK script

The ROPP package distribution includes a collection of configure and build scripts for a number of compilers and platforms suitable for ROPP and the dependency packages. A top-level BASH shell script `buildpack` is provided which may be used to automate the build of any ROPP module or dependency package in a consistent way, using the appropriate configure scripts. Use of `buildpack` is therefore highly recommended for first time build and less experienced users. Summary usage can be obtained using

```
> buildpack -h
```

In general, to build and install a package,

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

where `<package>` is one of the supported package names (e.g. `ropp_fm`, `ropp_io`, `netcdf`, `mobufr`, etc.) and `<comp>` is the required compiler (e.g. `ifort`, `gfortran`, etc.).

The `buildpack` script assumes that all tarball files and configure scripts provided with the ROPP distribution are placed in the same working directory. Packages will be decompressed here and installed to the `ROPP_ROOT/<comp>` target directory. The script automates the `configure` – `make` – `make install` build cycle described below. Further information on the `buildpack` script are provided in the ROPP Release Notes.

The shell scripts `build*_ropp`, `build_deps` and `build_ropp` have also been provided to help automate the build process by calling `buildpack` with a pre-determined sequence of packages or compilers, and to save a copy of all screen output to a disk log file. Users should review and edit these to suit their requirements. Using these tools, a complete check out of ROPP from scratch can be effected by running (in order):

[2] See http://rfsber.home.xs4all.nl/Robo/robodoc.html.

```
> buildzlib_ropp <compiler>
> buildhdf5_ropp <compiler>
> buildnetcdf_ropp <compiler> (note that this builds the core and Fortran libs)
> buildmobufr_ropp <compiler> or buildecbufr_ropp <compiler> or buildeccodes_ropp <compiler>
> buildgrib_ropp <compiler> or buildeccodes_ropp <compiler>
> buildsofa_ropp <compiler>
> build_ropp <compiler>
```

Or, even more quickly:

```
> build_deps <compiler> zlib hdf5 netcdf netcdff mobufr/ecbufr/eccodes grib/eccodes sofa
> build_ropp <compiler>
```

A.5 Building and installing ROPP manually

The low-level build sequence performed by `buildpack` may be implemented manually by more experienced users. After unpacking, all packages are compiled and installed following the `configure` – `make` – `make install` cycle.

1. First run the command `configure` to check for the availability of all required libraries. `configure` allows the user to specify compiler options, paths to libraries and the location where the software shall eventually be installed, on the command line or as environment variables. Based on this information, `configure` generates user specific `Makefiles`, allowing a highly customised configuration and installation of the software.
2. Compilation is then initiated with the command `make`.
3. If building the software was successful, a `make install` will install libraries, header and module files as well as any executables in the directories specified by the user via the configure step.

Note that the ROPP modules partially depend on each other. In particular, all packages require that `ropp_utils` has been installed successfully. This package therefore needs to be compiled and installed first. Most packages make use of the `ropp_io` package for sample applications and testing, and should therefore be installed next if these are required. Note that users wishing to use ROPP source code directly in their own applications need not install the `ropp_io` module. If the `ropp_io` module is not available at build time, only the source code libraries will be compiled. We thus recommend the following build order:

i) Third-party packages: `zlib`, `hdf5`, `netcdf`, `netcdff`, `mo/ecbufr`, `grib` (as required)
ii) `ropp_utils`
iii) `ropp_io` (if required)
iv) `ropp_pp` (if required)
v) `ropp_apps` (if required)
vi) `ropp_fm` (if required)
vii) `ropp_1dvar` (if required)

Note that *all* libraries need to be built with the same Fortran compiler, and preferably with the same version of the compiler as well.

Supported Fortran (and C) compilers are listed in the Release Notes distributed with the ROPP package.

A.5.1 Unpacking

Once the required third-party software packages have been installed successfully, the `ROPP` packages can be installed. The complete ROPP package and individual modules are distributed as gzipped tar (`.tar.gz`) files. The complete package file name consists of the version name (e.g. `ropp-11.0.tar.gz`). This file contains the complete ROPP distribution. The module file names consist of the package's name (e.g. `ropp_utils`) and version (e.g. 11.0), as in `ropp_utils-11.0.tar.gz`. If GNU tar is available (as on Linux systems), gzipped tar files can be unzipped with

```
> tar -xvzf ropp-11.0.tar.gz
```

Older, or non-GNU, versions of `tar` might need

```
> gunzip -c ropp-11.0.tar.gz | tar -xv
```

In all cases, a new subdirectory named (in the above example) `ropp-11.0` will be created which contains the source code of the complete package.

A.5.2 Configuring

Details on the installation procedure for the individual packages can be found in the files `README.unix` and `README.cygwin` for the installation under Unix and Windows (with Cygwin), respectively. Here, we provide a brief example for a Unix or Linux system.

Unpacking the `ropp_build` package will create the `configure/` sub-directory containing a number of mini-scripts for local build configuration. The files have names `<package>_configure_<compiler>_<os>` where `<package>` is the package name (ropp, netcdf), `<compiler>` is the compiler ID (ifort, nagfor, pgf95, ...) and `<os>` is the operating system ID, as output by the uname(1) command but entirely in lower case (linux, cygwin, ...). Note these configure mini-scripts are also used by the high-level `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.

The main configure scripts provided assume that the external libraries and individual ROPP modules are all installed under `$ROPP_ROOT`, i.e. the libraries can be found in the directory `$ROPP_ROOT/lib` and/or `$ROPP_ROOT/lib64`, and header and module files in `$ROPP_ROOT/include`. The `$ROPP_ROOT` location should be specified as an environment variable, e.g,

```
> export ROPP_ROOT=$HOME (for sh, ksh and bash users)
> setenv ROPP_ROOT $HOME (for csh and tcsh users)
```

For most compilers, this means that the two paths to the header and module files need to be specified via the proper compiler options — usually via the `-I` option. The linker also needs to know where libraries are located; on most Unix systems, this can be achieved by specifying the -L option at link time. Users are referred to the examples provided in the configure package for further details.

Running the appropriate script from configure/ will set the required compiler flags and specify the header, module and library paths before running the configure script. For example if the Fortran 95 compiler is named (say) ifort, the following command would be sufficient to configure a package for later compilation:

```
> cd ropp_<module>
> ../configure/ropp_configure_ifort_linux
```

The configure script will check for all required libraries and add the required options for the linker. If configure is not successful finding the required libraries, an error message will be produced, and further compilation will not be possible. Should the configuration step fail entirely, the file config.log created during the run of configure usually gives some clues on what went wrong; the most likely reason for failing is that compiler or linker options (and in particular paths to include files or libraries) are not set correctly.

Note that ropp_io may optionally use other external libraries in order to support additional features. For example, the ropp_io library will provide two conversion tools from ROPP to BUFR and back if a supported BUFR library is found. The existence of such additional libraries is also checked during configure. If these libraries are missing, however, the installation will proceed without building the parts related to the missing library. Should the build process fail to find usable BUFR libraries, for example, and therefore fail to build the BUFR tools, config.log should again provide evidence on what went wrong.

A.5.3 Compiling

If configuration was successful, the software can be built with the command

```
> make
```

This will compile all relevant source code, but may take several minutes. The resulting object library archive will be located in the build subdirectory. It will be named similar to the package following usual Unix conventions; for example, the ropp_utils library is named libropp_utils.a. Sample applications and test programs or scripts will also have been built in the relevant subdirectories. Sample and test runs can be performed without installing the software; for details on available test programs, see A.7.

Currently supported Fortran compilers include (on Linux unless otherwise stated): Intel's ifort (v16 and v17); NAG's nagfor (v6.1); Portland Group's pgf95 (v16); GNU gfortran (v4.8.5); Cray's ftn (v8.3.4). For the authoritative list please refer to the ROPP Release Notes and README files in each sub-package.

A.5.4 Installing

After building the software successfully, the command

```
> make install
```

will install libraries in {prefix}/lib, Fortran modules in {prefix}/include, and any application programs in {prefix}/bin. Here, {prefix} is the prefix directory given as argument to the --prefix option of the configure command. By default, this is $ROPP_ROOT. If no --prefix is given, the installation root directory defaults to /usr/local which would normally require root (sudo) privileges.

A.5.5 Cleaning up

The temporary files created during the compilation of any ROPP package can be removed from the package directory tree with

```
> make clean
```

Note that this will keep the information gathered during configuration as well as the build libraries and executables intact. Thus, a new build can be attempted using make without the need for another configure. To remove all data related to the build and install process, run

```
> make distclean
```

which will restore the original state of the unpacked package, but with all potential user modifications to the source code still in place.

If the software has been installed previously, but shall be removed from the user's computer, this can be accomplished with the command

```
> make uninstall
```

performed in the source code distribution directory. Note that this requires a configuration which is identical to the one used for the original installation of the software. It is not necessary to rebuild the software again before uninstalling it.

A.6 Linking

If one (or more) ROPP packages have been installed successfully, linking your application's code against the ROPP libraries requires the specification of all ROPP and all external libraries. For example, to create an executable from your own application.f90 and the ropp_io libraries, something like

```
> ifort -o application application.f90 -L/usr/local/lib -L$ROPP_ROOT/lib  \
        -L$ROPP_ROOT/lib64 -lropp_io -lropp_utils -lnetcdf (-lnetcdff)
```

will be required. (Since netCDF-4.1.1, the netCDF C and Fortran routines have been split, with the latter held in libnetcdff.a. Hence, if compiling Fortran routines against a recent version of netCDF, -lnetcdff must be included in the list of libraries to be linked. Note that the netCDF libraries recommended for use with ROPP are now split in this way.)

A.7 Testing

The ROPP software has undergone formal testing before distribution, as will all future modifications and improvements. A subset of the test procedures and some reference files are provided with the source code in order to facilitate quick tests whether the compilation was completed successfully. Users can run these tests to ensure that there are no major problems. It should be kept in mind, though, that not all of the functionality of the corresponding package is fully tested. Note also that several of the test scripts attempt to run IDL to generate output which can be compared against existing reference plots. Generally the user would only do this if one of the tests failed. If IDL is unavailable the tests will bypass this step.

A.7.1 ropp_utils

Tested as part of the other modules, mainly with ropp_io.

A.7.2 ropp_io

The subdirectory tests of the ropp_io distribution contains several test programs and scripts to test various aspects of the software. A test is provided to check the user's installation of the netCDF library. They can be run after a successful compilation of the ropp_io package with

```
> make test_netcdf
```

from within the tests subdirectory. The program executed for this test does not use ropp_io, but is exclusively based on the native Fortran 90 interfaces for netCDF. Failure of this test strongly indicates that there is a problem with the installation or setup of the external library, which needs to be fixed before ropp_io can be used.

A second test can be run with

```
> make test_ropp
```

which runs a script performing several conversions between ROPP data files. Running this test through make has the advantage that the results of the conversions are interpreted properly and result in 'success' or 'failure' messages.

If a supported BUFR library is available, the tests subdirectory will also contain a test script for the two programs ropp2bufr and bufr2ropp which convert ROPP data files to and and from BUFR format data files. Issuing the command

```
> make test_bufr
```

will run a number of conversions and provide some verbose information on the content of the BUFR files and the encoding and decoding process. The script finally also compares the results. Its output should be self-explanatory. Note that due to limitations of the BUFR format, non-significant loss of precision may be detected and flagged as differences from the reference file; this is normal.

The gfz2ropp and ucar2ropp tools to convert GFZ native text files or UCAR netCDF files to ropp-standard netcdf are tested with the commands

```
> make test_gfz
> make test_ucar
```

The grib2bgrasc and bgrasc2ropp tools, which extract background profiles from GRIB-format gridded data and convert to ascii format, and then convert this to a ROPP-format netCDF file, are respectively tested with the commands

```
> make test_grib
> make test_bgrasc
```

The eum2ropp and eum2bufr tools to convert 'EUMETSAT-format' RO data into standard ROPP netCDF or BUFR files, are tested with the commands

```
> make test_eum
> make test_eumbufr
```

Finally, the command

```
> make test
```

will run all of the above described tests.

The test of the ropp_io library and tools can also be tested manually by running, for example,

```
> t_ropp2ropp -t -n
```

which will create a series of different files. These should be compared (e.g., using diff) according to the advice given through the program's execution. Users can safely ignore numerical differences in the order of the cutoff in the text representation of the ROPP data files. Also note that different file names will show up in the first line of the text representation of netCDF data files (files created by the test script with the extension .cdl) and can be ignored. The test_ropp target actually does the same, but interprets the differences between the files with the above issues in mind. Note that the output of t_ropp2ropp can be found in the file t_ropp2ropp.log when run through make.

A.7.3 ropp_pp

The subdirectory tests of the ropp_pp distribution contains testing software, to compare the geometric optic and wave optic processing with known output, check the consistency of the Abel integral routines and their inverses, and compare the ionospheric correction processing with known output. It also tests a low resolution of the wave optics propagator code, which resides in the ropp_pp module. Run

```
> make test
```

to check if solutions agree with precalculated solutions to within expected small tolerances. If IDL is available on the user's machine, plots of the results are made and can be compared against reference plots. A table summarising the results of the tests is written to stdout after they have all run.

A.7.4 `ropp_apps`

The subdirectory `tests` of the `ropp_apps` distribution contains testing software, to calculate tropopause height, and planetary boundary layer height, from a variety of profile data: bending angles, refractivities, background temperatures etc. Run

```
> make test
```

to check if solutions agree with precalculated solutions to within expected small tolerances. A table summarising the results of the tests is written to stdout after they have all run.

A.7.5 `ropp_fm`

The subdirectory `tests` of the `ropp_fm` distribution contains testing software. Run

```
> make test
```

to check if everything is working correctly. A series of tests are run to run the 1D and 2D operator applications to generate simulated refractivity and bending angle profiles, which are compared with precalculated data. Also included are tests of the consistency of the 1D and 2D tangent linear and adjoint routines. Warning messages are written to `stdout` if the operator, tangent linear and adjoint routines do not meet the expected (demanding) consistency checks. If IDL is available on the user's machine, plots of the results are made and can be compared against reference plots. A table summarising the results of the tests is written to stdout after they have all run.

A.7.6 `ropp_1dvar`

A simple test is provided to check the correct running of the 1D–Var stand-alone application. This inputs a file of 'observations' (refractivity profiles) simulated from a set of ECMWF model background profiles. The same backgrounds are used in the 1D–Var retrieval. Hence the expected retrieved output profiles should be identical to the background (within rounding errors).

Further tests are run of retrievals based on COSMIC observations (refractivities and bending angles) and co-located Met Office background profiles, and of retrievals based on GRAS observations (refractivities and bending angles) and co-located ECMWF background profiles. A simple test of a retrieval using L1 and L2 bending angles is also included.

The subdirectory `tests` of the `ropp_1dvar` distribution contains the testing software. Run

```
> make test
```

to check if everything is working correctly. The results of each test are numerically compared to reference results, and a PASS/FAIL message issued to stdout if the differences are smaller/greater than some small tolerance. If IDL is available on the user's machine, plots of the results are made and can be compared against reference plots. A table summarising the results of the tests is written to stdout after they have all run.

A.8 Troubleshooting

If something goes wrong during the configuration step, carefully check the full output of the last unsuccessful `configure` run to get an idea why the software could not be built; this can be found in the file `config.log`. This also applies if parts of ROPP are not built (e.g. the `BUFR` tools), even though the required additional libraries are available.

During compilation, warnings that indicate unused variables (e.g. with the NAG compiler) or the potential trimming of character variables (with Intel compilers) can safely be ignored. If the compilation is successful, but installation fails, make sure you have write permissions on the installation directories.

If linking against ROPP libraries fails because of unresolved externals, make sure that *all* relevant libraries – *including all external ones* – are specified in the correct order (some linkers are not able to recursively browse through several libraries in order to resolve externals) with lower-level libraries following higher-level (ROPP) ones.

If the BUFR encoding or decoding fail with messages about missing run-time BUFR tables, check that the appropriate environment variable `BUFR_LIBRARY` (for the MetDB library) or `BUFR_TABLES` (for the ECMWF library) have been correctly set to the path of the installed BUFR tables, and that the path ends with a '/' character.

Forward modelling of, and retrievals using, L1 and L2 bending angles impose heavier memory requirements than the more standard use of neutral bending angles. Users should therefore be prepared to increase the local memory available on their machines if using this feature.

If an ROPP module compiles and runs satisfactorily, but produces unexpected results, an easy first step in tracking down the problem is to print out extra diagnostic information. Most of the ROPP tools provide the facility to do this by means of the '`-d`' option. `ropp_pp`, `ropp_1dvar`, `ropp_apps` and `ropp_fm` also allow the user to add sets of pre-defined variables to the `ROprof` structure, which are written out in `netCDF` format with the usual variables. The first two modules do this by means of an option in a configuration file; the last two by means of a command line option in (some of) the tools. In fact, all ROPP modules allow the user to add specified variables to the `ROprof` structure in this way, by calling `ropp_io_addvar`, as described in the ROPP I/O user Guide. This obviously requires the code to be recompiled.

B `ropp_fm` program files

The `ropp_fm` module provides one-dimensional forward operators to compute vertical refractivity and bending angle profiles from background data on height-based and hybrid NWP model vertical grids. A two-dimensional forward operator to compute a plane bending angle profile from a two-dimensional plane of background data is also included. Tangent linear, adjoint and gradient codes to the forward operators are provided for use in assimilation processing.

As Fig 2.1 shows, `ropp_utils` and `ropp_io` are required for all `ropp_fm` tools. So is the `netCDF` library.

tools/

- **ropp_fm_bg2ro_1d**
- **ropp_fm_bg2ro_2d**

tests/

- **test_fm_FASCOD.sh**
- **test_fm_IONO.sh**
- **test_fm_TWOD.sh**
- **test_fm_1D.sh**
- **test_fm_2D.sh**
- **test_fm_iono.sh**
- `ropp_fm_compare.f90`
- `ropp_fm_summary.f90`
- `t_fascod.f90`
- `t_fascod_tl.f90`
- `t_fascod_ad.f90`
- `t_twoop.f90`
- `t_twotl.f90`
- `t_twoad.f90`
- `t_iono.f90`
- `t_iono_tl.f90`
- `t_iono_ad.f90`

- Integrated code

`bangle_1d/`

- `ropp_fm_bangle_1d.f90`
- `ropp_fm_bangle_1d_grad.f90`
- `ropp_fm_bangle_1d_tl.f90`
- `ropp_fm_bangle_1d_ad.f90`
- `ropp_fm_abel.f90`
- `ropp_fm_abel_tl.f90`
- `ropp_fm_abel_ad.f90`

`bangle_2d/`

- `ropp_fm_bangle_2d.f90`
- `ropp_fm_bangle_2d_tl.f90`
- `ropp_fm_bangle_2d_ad.f90`
- `ropp_fm_alpha2drk.f90`
- `ropp_fm_alpha2drk_ad.f90`

```
    ropp_fm_alpha2drk_td.f90
    ropp_fm_gpsderivs.f90
    ropp_fm_gpsderivs_ad.f90
    ropp_fm_gpsderivs_tl.f90

refrac_1d/
    ropp_fm_refrac_1d.f90
    ropp_fm_refrac_1d_ad.f90
    ropp_fm_refrac_1d_grad.f90
    ropp_fm_refrac_1d_new.f90
     ropp_fm_refrac_1d_new_ad.f90
    ropp_fm_refrac_1d_new_tl.f90
    ropp_fm_refrac_1d_tl.f90

math/matrix/
    matrix_types.f90

model_ecmwf/
    ropp_fm_state2state_ecmwf.f90
    ropp_fm_state2state_ecmwf_tl.f90
    ropp_fm_state2state_ecmwf_ad.f90

model_meto/
    ropp_fm_state2state_meto.f90
    ropp_fm_state2state_meto_tl.f90
    ropp_fm_state2state_meto_ad.f90

common/
    ropp_fm.f90
    ropp_fm_types.f90
    ropp_fm_constants.f90
    ropp_fm_free.f90
    ropp_fm_interpol.f90
    ropp_fm_interpol_tl.f90
    ropp_fm_interpol_ad.f90
    ropp_fm_interpol_log.f90
    ropp_fm_interpol_log_tl.f90
    ropp_fm_interpol_log_ad.f90
    ropp_fm_levels.f90
    ropp_fm_obs2obs.f90
    ropp_fm_state2state.f90
    ropp_fm_copy.f90
    ropp_fm_set_units.f90
    ropp_fm_roprof2obs.f90
    ropp_fm_roprof2state.f90
    ropp_fm_obs2roprof.f90
    ropp_fm_state2roprof.f90
    ropp_fm_compress_single.f90
    ropp_fm_compress_single_tl.f90
    ropp_fm_compress_single_ad.f90

iono/
    ropp_fm_iono.f90
    ropp_fm_iono_bangle.f90
    ropp_fm_iono_bangle_tl.f90
    ropp_fm_iono_bangle_ad.f90
    ropp_fm_asinh.f90
    ropp_fm_zorro.f90
    ropp_fm_dzorro_dlg.f90
    ropp_fm_iono_set_default.f90
    ropp_fm_iono_unpack_bangle.f90

dbangle_1d/
    ropp_fm_dbangle_1d.f90
    ropp_fm_dbangle_1d_tl.f90
    ropp_fm_dbangle_1d_grad.f90
    ropp_fm_vchap.f90
    ropp_fm_vchap_tl.f90
    ropp_fm_vchap_grad.f90
    ropp_fm_ne_vtec.f90
```

C ROPP extra diagnostic data

For reference and for completeness, the listings of the all ROPP modules' extra variables are listed below.

C.1 ropp_io_addvar

The general form of the extra data, appended to the `RO_prof` structure by `ropp_io_addvar`, is described in Table C.1.

ROprof (Additional variables requested by call to `ropp_io_addvar`, throughout ROPP)	
Structure element	Description
`...%vlist%VlistD0d%name`	Name of 1st 0D extra variable
`...%vlist%VlistD0d%long_name`	Long name of 1st 0D extra variable
`...%vlist%VlistD0d%units`	Units of 1st 0D extra variable
`...%vlist%VlistD0d%range`	Range of 1st 0D extra variable
`...%vlist%VlistD0d%DATA`	Value of 1st 0D extra variable
`...%vlist%VlistD0d%next%name (etc)`	Name (etc) of 2nd 0D extra variable
`...%vlist%VlistD0d%next%next%name (etc)`	Name (etc) of 3rd 0D extra variable
`...%vlist%VlistD1d%name`	Name of 1st 1D extra variable
`...%vlist%VlistD1d%long_name`	Long name of 1st 1D extra variable
`...%vlist%VlistD1d%units`	Units of 1st 1D extra variable
`...%vlist%VlistD1d%range`	Range of 1st 1D extra variable
`...%vlist%VlistD1d%DATA`	Value of 1st 1D extra variable
`...%vlist%VlistD1d%next%name (etc)`	Name (etc) of 2nd 1D extra variable
`...%vlist%VlistD1d%next%next%name (etc)`	Name (etc) of 3rd 1D extra variable
`...%vlist%VlistD2d%name`	Name of 1st 2D extra variable
`...%vlist%VlistD2d%long_name`	Long name of 1st 2D extra variable
`...%vlist%VlistD2d%units`	Units of 1st 2D extra variable
`...%vlist%VlistD2d%range`	Range of 1st 2D extra variable
`...%vlist%VlistD2d%DATA`	Value of 1st 2D extra variable
`...%vlist%VlistD2d%next%name (etc)`	Name (etc) of 2nd 2D extra variable
`...%vlist%VlistD2d%next%next%name (etc)`	Name (etc) of 3rd 2D extra variable

Table C.1: Additional elements of `ROprof` structure, available throughout ROPP

C.2 PPDiag

The extra data which are output to the netCDF file if `config%output_diag` is set to `.TRUE.` in `ropp_pp`, are described in Table C.2.

PPDiag (`config%output_diag` = `TRUE` in `ropp_pp`)	
Structure element	Description
`...%CTimpact`	CT processing impact parameter (m)
`...%CTamplitude`	CT processing amplitude
`...%CTamplitude_smt`	CT processing smoothed amplitude
`...%CTimpactL2`	CT processing L2 impact parameter (m)
`...%CTamplitudeL2`	CT processing L2 amplitude
`...%CTamplitudeL2_smt`	CT processing smoothed L2 amplitude
`...%ba_ion`	Ionospheric bending angle in L1 (rad)
`...%err_neut`	Error covariance of neutral bending angle (rad^2)
`...%err_ion`	Error covariance of ionospheric bending angle (rad^2)
`...%wt_data`	Weight of data (data:data+clim) in profile
`...%sq`	SO badness score: $\mathtt{MAX[err\_neut}^{1/2}/\alpha_N] \times 100\%$
`...%L2_badness`	L2 phase correction badness score
`...%L2_min_SLTA`	Lowest valid L2 SLTA (m)

Table C.2: Elements of `PPDiag` structure, available from `ropp_pp`

C.3 ropp_fm_bg2ro

The extra data which are appended to the ROprof structure if the `ropp_fm` tool `ropp_fm_bg2ro_1d` is called without the '`-f`' option, are described in Table C.3.

ROprof (Absence of '`-f`' option in call to `ropp_fm_bg2ro_1d`, in `ropp_fm`)	
Structure element	Description
`...%gradient_refrac`	$\partial N_i/\partial x_j$ matrix
`...%gradient_bangle`	$\partial \alpha_i/\partial x_j$ matrix

Table C.3: Additional elements of `ROprof` structure, available from `ropp_fm`. See Table C.1 for the detailed structure.

C.4 VarDiag

The extra data which are output to the netCDF file if `config%extended_1dvar_diag` is set to `.TRUE.` in `ropp_1dvar`, are described in Table C.4.

VarDiag (`config%extended_1dvar_diag` = `TRUE` **in** `ropp_1dvar`)	
Structure element	Description
`...%n_data`	Number of observation data
`...%n_bgqc_reject`	Number of data rejected by background QC
`...%n_pge_reject`	Number of data rejected by PGE QC
`...%bg_bangle`	Background bending angle
`...%bg_refrac`	Background refractivity
`...%OmB`	Observation minus background
`...%OmB_sigma`	OmB standard deviation
`...%pge_gamma`	PGE check gamma value
`...%pge`	Probability of Gross Error along profile
`...%pge_weights`	PGE weighting values
`...%ok`	Overall quality flag
`...%J`	Cost function value at convergence
`...%J_scaled`	Scaled cost function value $(2J/m)$
`...%J_init`	Initial cost function value
`...%J_bgr`	Background cost function profile
`...%J_obs`	Observation cost function profile
`...%B_sigma`	Forward modelled bg standard deviation
`...%n_iter`	Number of iterations to reach convergence
`...%n_simul`	Number of simulations
`...%min_mode`	Minimiser exit mode
`...%res_bangle`	Analysis bending angle
`...%res_refrac`	Analysis refractivity
`...%OmA`	Observation minus analysis
`...%OmA_sigma`	OmA standard deviation
`...%bg_ne`	Background electron density
`...%bg_ne_sigma`	Error in background electron density
`...%res_ne`	Analysis electron density
`...%res_ne_sigma`	Error in analysis electron density
`...%VTEC_bg`	VTEC of background electron density
`...%VTEC_an`	VTEC of analysis electron density

Table C.4: Elements of `VarDiag` structure, available from `ropp_1dvar`.

D ROPP user documentation

Title	Reference	Description
ROPP User Licence	SAF/ROM/METO/LIC/ROPP/002	Legal conditions on the use of ROPP software
ROPP Overview	SAF/ROM/METO/UG/ROPP/001	Overview of ROPP and package content and functionality
ROPP_IO User Guide	SAF/ROM/METO/UG/ROPP/002	Description of `ropp_io` module content and functionality
ROPP_PP User Guide.	SAF/ROM/METO/UG/ROPP/004	Description of `ropp_pp` module content and functionality
ROPP_APPS User Guide.	SAF/ROM/METO/UG/ROPP/005	Description of `ropp_apps` module content and functionality
ROPP_FM User Guide.	SAF/ROM/METO/UG/ROPP/006	Description of `ropp_fm` module content and functionality
ROPP_1DVAR User Guide.	SAF/ROM/METO/UG/ROPP/007	Description of `ropp_1dvar` module content and functionality
ROPP UTILS Reference Manual	SAF/ROM/METO/RM/ROPP/001	Reference manual for the `ropp_utils` module
ROPP IO Reference Manual	SAF/ROM/METO/RM/ROPP/002	Reference manual for the `ropp_io` module
ROPP FM Reference Manual	SAF/ROM/METO/RM/ROPP/003	Reference manual for the `ropp_fm` module
ROPP 1D–Var Reference Manual	SAF/ROM/METO/RM/ROPP/004	Reference manual for the `ropp_1dvar` module
ROPP PP Reference Manual	SAF/ROM/METO/RM/ROPP/005	Reference manual for the `ropp_pp` module
ROPP APPS Reference Manual	SAF/ROM/METO/RM/ROPP/006	Reference manual for the `ropp_apps` module
WMO FM94 (BUFR) Specification for Radio Occultation Data	SAF/ROM/METO/FMT/BUFR/001	Description of BUFR template for RO data

Table D.1: ROPP user documentation

Title	Reference	Description
Mono-dimensional thinning for GPS Radio Occultations	SAF/GRAS/METO/REP/GSR/001	Technical report on profile thinning algorithm implemented in ROPP
Geodesy calculations in ROPP	SAF/GRAS/METO/REP/GSR/002	Summary of geodetic calculations to relate geometric and geopotential height scales
ROPP minimiser - minROPP	SAF/GRAS/METO/REP/GSR/003	Description of ROPP-specific minimiser, minROPP
Error function calculation in ROPP	SAF/GRAS/METO/REP/GSR/004	Discussion of impact of approximating erf in ROPP
Refractivity calculations in ROPP	SAF/GRAS/METO/REP/GSR/005	Summary of expressions for calculating refractivity profiles
Levenberg-Marquardt minimisation in ROPP	SAF/GRAS/METO/REP/GSR/006	Comparison of Levenberg-Marquardt and minROPP minimisers
Abel integral calculations in ROPP	SAF/GRAS/METO/REP/GSR/007	Comparison of ‘Gorbunov’ and ‘ROM SAF’ Abel transform algorithms
ROPP thinner algorithm	SAF/GRAS/METO/REP/GSR/008	Detailed review of the ROPP thinner algorithm
Refractivity coefficients used in the assimilation of GPS radio occultation measurements	SAF/GRAS/METO/REP/GSR/009	Investigation of sensitivity of ECMWF analyses to empirical refractivity coefficients and non-ideal gas effects
Latitudinal Binning and Area-Weighted Averaging of Irregularly Distributed RO Data	SAF/GRAS/METO/REP/GSR/010	Discussion of alternative spatial averaging method for RO climate data
ROPP 1D–Var validation	SAF/GRAS/METO/REP/GSR/011	Illustration of ROPP 1D–Var functionality and output diagnostics
Assimilation of GPSRO Data in the ECMWF ERA-Interim Re-analysis	SAF/GRAS/METO/REP/GSR/012	Assimilation of GPSRO Data in the ECMWF ERA-Interim Re-analysis
ROPP_PP validation	SAF/GRAS/METO/REP/GSR/013	Illustration of ROPP_PP functionality and output diagnostics

Table D.2: GRAS SAF Reports

Title	Reference	Description
A review of the geodesy calculations in ROPP	SAF/ROM/METO/REP/RSR/014	Comparison of various potential geodesy calculations
Improvements to the ROPP refractivity and bending angle operators	SAF/ROM/METO/REP/RSR/015	Improved interpolation in ROPP forward models
Simplifying EGM96 undulation calculations in ROPP	SAF/ROM/METO/REP/RSR/016	Simplifying ROPP undulation calculations
Simulation of L1 and L2 bending angles with a model ionosphere	SAF/ROM/METO/REP/RSR/017	Simulating L1 and L2 bending angles in ROPP
Single Frequency Radio Occultation Retrievals: Impact on Numerical Weather Prediction	SAF/ROM/METO/REP/RSR/018	Potential impact of loss of L2 bending angle on NWP
Implementation of the ROPP two-dimensional bending angle observation operator in an NWP system	SAF/ROM/METO/REP/RSR/019	Implementation of ROPP 2D forward model at ECMWF
Interpolation artefact in ECMWF monthly standard deviation plots	SAF/ROM/METO/REP/RSR/020	Investigation into plot anomaly
5th ROM SAF User Workshop on Applications of GPS radio occultation measurements	SAF/ROM/METO/REP/RSR/021	Report on 5th ROM SAF User Workshop
The use of the GPS radio occultation reflection flag for NWP applications	SAF/ROM/METO/REP/RSR/022	Impact of reflected occultations at ECMWF
Assessment of a potential reflection flag product	SAF/ROM/METO/REP/RSR/023	Assessment of flagged COSMIC occultations
The calculation of planetary boundary layer heights in ROPP	SAF/ROM/METO/REP/RSR/024	Description of ROPP PBLH diagnostics
Survey on user requirements for potential ionospheric products from EPS-SG radio occultation measurements	SAF/ROM/METO/REP/RSR/025	Results of a ROM SAF survey of the interest in possible EPS-SG ionospheric products
Estimates of GNSS radio occultation bending angle and refractivity error statistics	SAF/ROM/METO/REP/RSR/026	RO error statistics as derived by forward modelling ECMWF model errors
Recent forecast impact experiments with GPS radio occultation measurements	SAF/ROM/METO/REP/RSR/027	Impacts in NWP of 2014–2015 RO data
Description of wave optics modelling in ROPP-9 and suggested improvements for ROPP-9.1	SAF/ROM/METO/REP/RSR/028	Wave optics propagator in ROPP-9.0 and 9.1

Table D.3: ROM SAF Reports

Title	Reference	Description
Testing reprocessed GPS radio occultation datasets in a reanalysis system	SAF/ROM/METO/REP/RSR/029	Impact of reprocessed RO data on reanalyses
A first look at the feasibility of assimilating single and dual frequency bending angles	SAF/ROM/METO/REP/RSR/030	Single and dual frequency assimilation
Sensitivity of some RO measurements to the shape of the ionospheric electron density profile	SAF/ROM/METO/REP/RSR/031	Ionospheric shape sensitivity
An initial assessment of the quality of RO data from KOMPSAT-5	SAF/ROM/METO/REP/RSR/032	KOMPSAT-5 quality assessment
Some science changes in ROPP-9.1	SAF/ROM/METO/REP/RSR/033	ROPP-9.1 science
An initial assessment of the quality of RO data from Metop-C	SAF/ROM/METO/REP/RSR/034	Metop-C quality assessment
An initial assessment of the quality of RO data from FY-3D	SAF/ROM/METO/REP/RSR/035	FY-3D quality assessment
An initial assessment of the quality of RO data from PAZ	SAF/ROM/METO/REP/RSR/036	PAZ quality assessment
6th ROM SAF User Workshop	SAF/ROM/METO/REP/RSR/037	ROM SAF–IROWG 2019 report
An initial assessment of the quality of RO data from COSMIC-2	SAF/ROM/METO/REP/RSR/038	COSMIC-2 quality assessment
Impacts of RO mission differences on trends in multi-mission data records	SAF/ROM/METO/REP/RSR/039	RO mission CDR differences
Anomalous GRAS radio occultations	SAF/ROM/METO/REP/RSR/040	Anomalous occultations
Assessment of sensitivity of the ROM SAF 1D-Var solutions to various error covariance choices	SAF/ROM/METO/REP/RSR/041	Sensitivity to error covariances
A one-dimensional variational ionospheric retrieval for truncated GNSS Radio Occultation measurements	SAF/ROM/METO/REP/RSR/042	Ionospheric 1dvar

Table D.4: ROM SAF Reports (continued)

Title	Reference	Description
CDOP-3 Proposal	SAF/ROM/DMI/MGT/CDOP3/001	Proposal for the Third Continuous Development and Operations Phase (CDOP-3) March 2017 – February 2022
Co-operation Agreement	EUM/C/85/16/DOC/19	C/A between EUMETSAT and DMI, Lead Entity for the CDOP-3 of the ROM SAF, signed at the 86th Council meeting on 7th December 2016
Product Requirements Document (PRD)	SAF/ROM/DMI/MGT/PRD/001	Detailed specification of the products of the ROM SAF
System Requirements Document (SRD)	SAF/ROM/DMI/RQ/SRD/001	Detailed specification of the system and software requirements of the ROM SAF

Table D.5: Applicable documents

E Authors

Many people, inside and outside the ROM SAF, have contributed to the development of ROPP. The principal authors are listed alphabetically in Table E.1. The ROM SAF extends its sincere gratitude for their efforts.

ROPP Authors

Name	Current institute	Contribution
Carlo Buontempo	Met Office	Savitzky-Golay thinner code.
Chris Burrows	ECMWF	2nd ROPP Test Manager. Test folder developments, improved FM vertical interpolation scheme.
Ian Culverwell	Met Office	2nd ROPP Development Manager. Documentation, testing, consolidation, IO development, GRIB2 reader, implementation of tropopause height diagnostics and planetary boundary layer height diagnostics, forward modelling of L1 and L2 bending angles, implementation of VaryChap f2–f2 FM and 1DVAR code
Axel von Engeln	EUMETSAT	Author of original Test Folder system and of EUMETSAT-formatted RO data reader.
Hans Gleisner	DMI	Elements of ropp_pp, prototype GRIB2 reader, ec{i/f}2ec{i/f} code.
Michael Gorbunov	Russian Academy of Sciences	Original pre-processor code.
Sean Healy	ECMWF	Original 1D FM code, 2D FM operator code, introduction of compressibility factors, improved FM vertical interpolation scheme, forward modelling of L1 and L2 bending angles, 1D and 2D wave optics propagators, prototype VaryChap f2–f2 FM and 1DVAR code.
Helge Jønch-Sørensen	DMI	BAROCLIM code.
Kjartan Kinch	DMI	Elements of ropp_pp.
Kent Bækgaard Lauritsen	DMI	Code reviews; liaison with EUMETSAT (licences, beta tester contracts).
Huw Lewis	Met Office	1st ROPP Development Manager, FM and 1D–VAR extensions. PP module.
Owen Lewis	Met Office	BUFR developments.
Christian Marquardt	EUMETSAT	Author of majority of ROPP-1 code in UTILS, IO, FM and 1DVAR modules, and much personal, pre-existing software.
Dave Offiler	Met Office	ROPP Project Manager, IO application code and IO extensions, BUFR format/template.
Michael Rennie	ECMWF	1st ROPP Test Manager. Test folder developments.
Barbara Scherllin-Pirscher	Wegener Center	BAROCLIM (3) dataset for statistical optimisation.
Torsten Schmidt	GFZ	Guidance on tropopause height diagnostics.
Stig Syndergaard	DMI	Original spectral version of MSIS model (expansion in spherical harmonics and Chebychev polynomials), PP module developments.
Francis Warrick	Met Office	Implementation of ecCodes lib; ROPP devt and testing.
Feiqin Xie	Texas A & M	Suggested boundary layer height diagnostic algorithms.

Table E.1: Contributors to ROPP

F Copyrights

The majority of ROPP code is

© Copyright 2009-2021, EUMETSAT, All Rights Reserved.

This software was developed within the context of the EUMETSAT Satellite Application Facility on Radio Occultation Meteorology (ROM SAF), under the Cooperation Agreement dated 29 June 2011, between EUMETSAT and the Danish Meteorological Institute (DMI), Denmark, by one or more partners within the ROM SAF. The partners in the ROM SAF are DMI, Met Office, UK, the Institut d'Estudis Espacials de Catalunya (IEEC), Spain and the European Centre for Medium-Range Weather Forecasts (ECMWF), UK

Some parts of the source code within this distribution were developed within the Met Office outside the context of the ROM SAF and represents pre-existing software (PES); this portion is

© Crown copyright 2018, Met Office. All rights reserved.

Use, duplication or disclosure of this code is subject to the restrictions as set forth in the contract. If no contract has been raised with this copy of the code, the use, duplication or disclosure of it is strictly prohibited. Permission to do so must first be obtained in writing from the Head of Satellite Applications at the following address:

Met Office, FitzRoy Road Exeter, Devon, EX1 3PB United Kingdom

This ROPP package also contains open source code libraries available through its author, Christian Marquardt. This is also PES, and is

© Copyright 2007 Christian Marquardt <christian@marquardt.sc>

All rights reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software as well as in supporting documentation.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT.

IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

This ROPP package may also contain open source code libraries available through its author, Michael Gorbunov. This is also PES, and is

© Copyright 1998-2010 Michael Gorbunov <michael.gorbunov@zmaw.de>

All rights reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software with the rights to use, copy, modify, merge copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software as well as in supporting documentation.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE. HOWEVER, ALL EFFORTS ARE BEING MADE BY THE AUTHOR IN ORDER TO FIND AND ELIMINATE ALL POSSIBLE ERRORS AND PROBLEMS. IN THIS CASE THE AUTHOR MAY PROVIDE AN UPDATE.

This ROPP package may also contain open source code libraries available through its author, Stig Syndergaard. This is also PES, and is

© Copyright 1998 Stig Syndergaard <ssy@dmi.dk>

All rights reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, and/or sublicense the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software as well as in supporting documentation.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT.

IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

This ROPP package may also contain a dataset available through its author, Barbara Scherllin-Pirscher, and is

© Copyright 2013-2014 Barbara Scherllin-Pirscher

All rights reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of the latest BAROCLIM (V3) dataset (the "Dataset") to use, copy, publish and merge copies of the Dataset for scientific and non-commercial purposes only and to permit persons to whom the Dataset is furnished to do so also for scientific but non-commercial purposes only, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Dataset as well as in supporting documentation.

THE DATASET IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDER BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE DATASET OR THE USE OR OTHER DEALINGS OF THE DATASET.