ROM SAF Report ??
Ref: SAF/ROM/DMI/REP/RSR/000
Web: www.romsaf.org
Date: 8 Nov 2019

The EUMETSAT Network of Satellite Application Facilities

ROM SAF
Radio Occultation Meteorology

ROM SAF Report ??

Forward model of Bending/Impact profile for reflected signals: `ropp` routines

Estel Cardellach[1]

[1] *IEEC (ICE/CSIC Unit), Barcelona, Spain*

SUBMITTED

Document Author Table

	Name	*Function*	*Date*	*Comments*
Prepared by:	E. Cardellach	ROM SAF Project Team	8 Nov 2019	
Reviewed by:			DD Mon YYYY	
Reviewed by:			DD Mon YYYY	
Approved by:	K. B. Lauritsen	ROM SAF Project Manager	DD Mon YYYY	

Document Change Record

Issue/Revision	*Date*	*By*	*Description*
-	8 Nov 2019	EC	Document presenting ropp routines for forward modelling of reflected bending/impact profiles

ROM SAF

The Radio Occultation Meteorology Satellite Application Facility (ROM SAF) is a decentralised processing centre under EUMETSAT which is responsible for operational processing of GRAS radio occultation data from the Metop satellites and RO data from other missions. The ROM SAF delivers bending angle, refractivity, temperature, pressure, and humidity profiles in near-real time and offline for NWP and climate users. The offline profiles are further processed into climate products consisting of gridded monthly zonal means of bending angle, refractivity, temperature, humidity, and geopotential heights together with error descriptions.

The ROM SAF also maintains the Radio Occultation Processing Package (ROPP) which contains software modules that will 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 project please go to: http://www.romsaf.org

Abstract

This document presents `ropp` routines to forward model the bending/impact profiles of the reflected signals, that is, impact heights below the impact of the Earth surface. The routines implement the equations in Aparicio et al., 2018, but assume that the simplifications applied in the original `ropp` routines do also apply in this case.

Contents

1 Chapter

1.1 Introduction

During the Continuous Development and Operations Phase 2 (CDOP2), the ROM SAF developed a forward model for the bending-impact profile of signals reflected off the Earth surface. The model, described in [1] and [2], is a modified form of the Abel transform formulation.

As part of Work Package 3420 in the CDOP3, the ROM SAF has implemented the forward model in `ropp` routines. These routines are presented in the following sections.

1.2 The forward model

The model to be implemented is valid for impact parameters below the impact corresponding to the Earth surface, a_s. The reflected bending, α_r, as a function of the reflected impact, a_r is [2]:

$$\alpha_r(a_r) = -2a_r \int_{a_s}^{\infty} \frac{d\ln(n)/da}{\sqrt{a^2 - a_r^2}} da - 2\arccos(\frac{a_r}{a_s}) \tag{1.1}$$

The Abel inverse transform for the bending-impact profile induced by the neutral atmosphere (originally implemented in `ropp`) is:

$$\alpha(a') = -2a' \int_{a'}^{\infty} \frac{d\ln(n)/da}{\sqrt{a^2 - a'^2}} da \tag{1.2}$$

Note that the only difference is the second term, and the lower limit of the integral.

The `ropp` forward model module, `ropp_fm` , codes Equation 1.2 under certain assumptions to simplify its numerical implementation. As reported in [3], the assumptions are

- the refractivity is small, so $dln(n)/da \sim 10^{-6} dN/da$ (equation 3.29 in [3]);
- the refractivity scale height is small compared to the radius of the Earth, so $\sqrt{a^2 - a'^2} \sim \sqrt{2a'(a - a')}$;
- different assumptions and options regarding the refractivity variation between the discrete set of levels.

These assumptions also hold for $a_r \sim a_s$, where the reflected bending-impact profile must be evaluated. Therefore, the implementation used in the original routine `ropp_fm_abel.f90` is mantained, and only the lower limit of the integral is replaced (a' in Equation 1.2 by a_s in Equation 1.1), and the second term of Equation 1.1 is added.

As described in [3], `ropp_fm_abel.f90` implements two approaches to compute the integral, one when the refractivity gradient is positive with height and the other when the it is negative. These two approaches are also kept in the reflected signal implementation `ropp_fm_abel_refl.f90`, as they only affect the intra-layer interpolation process.

1.3 ropp_fm_abel_refl.f90

The most straightfoward approach to modify `ropp` for this job would have been to add a case in `ropp_fm_abel.f90`, to check whether each input impact parameter under analysis lays above or below the one corresponding to the surface level, a_s, and applying either Equation 1.2 or 1.1, respectively.

However, in order to fully separate the standard routines from the new routines developed for reflected signals, the approach finally implemented keeps the original routine for the non-reflected signals, and adds a second one that computes only the bending of reflected signals. They should be called in the correct order:

1. call the original abel transform, `ropp_fm_abel.f90`
2. call the new routine for reflected signals, `ropp_fm_abel_refl.f90`

The first call, to `ropp_fm_abel.f90`, will return the standard bending angles, while the second call, to `ropp_fm_abel_refl.f90`, will only compute and edit the bending angles corresponding to the input impact parameters below the surface one, a_s (Figure 1.1).

The impact parameter that corresponds to the surface level, a_s, is assumed to be $a_s = n(r_s)\ r_s$, where the radial distance of the surface is $r_s \sim (R_{roc} + undulation)$ and the refractive index at the surface is extrapolated from the state vector accordingly.

The calling syntax for the original and modified routines are shown in the table below, in red the new parameter to be passed:

`call ropp_fm_abel(in_nr, in_refrac, in_temp, in_roc, in_Tgrad_oper, inout_impact, inout_bangle)`
`call ropp_fm_abel_refl(in_nr, in_refrac, in_temp, in_roc, in_undulation, in_Tgrad_oper, inout_impact, inout_bangle)`

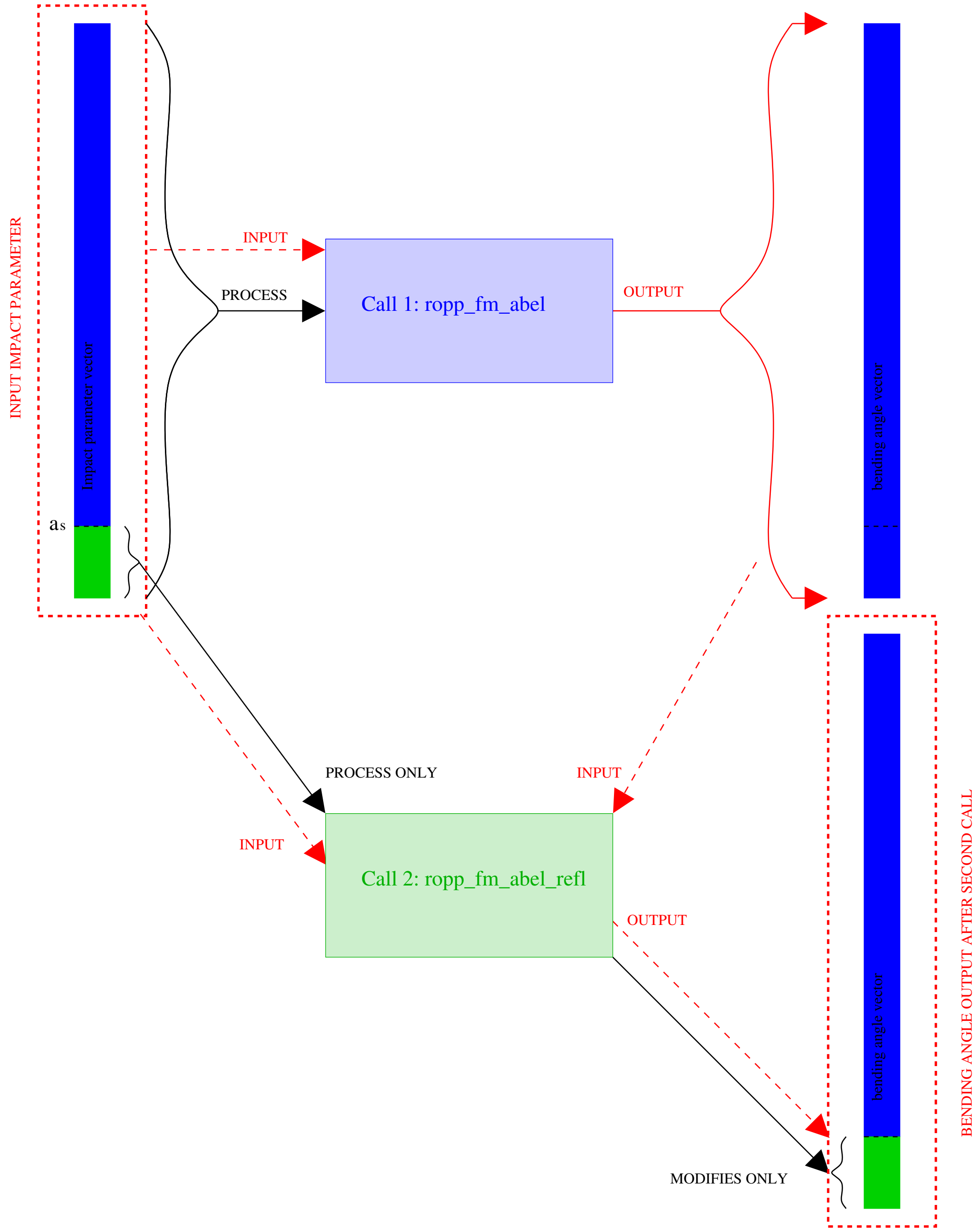

Figure 1.1: To separate the routines dealing with reflected signals from the original ones, the approach implemented is sketched in this figure: the orginal abel routine is applied to the input impact parameter vector, and it generates a bending angle output based on the standard abel model. Afterwards, the new reflectometry abel routine is called, that despite taking as input the whole impact parameter vector, it only computes bending angles for those impact parameters below the surface's one, a_s (green shaded part of the vector), and it keeps the bending angle array as it came from the first call, only modifying the bending angles that correspond to impact parameters below a_s.

1.4 Comparison beween `ropp_fm_abel.f90` and `ropp_fm_abel_refl.f90`

The code below corresponds to `ropp_fm_abel_refl.f90`, where the red lines are the ones added or modified in the original routine `ropp_fm_abel.f90`:

SUBROUTINE ropp_fm_abel_refl(nr, refrac, temp, roc, und, Tgrad_oper, impact, bangle)

```
!****s* BendingAngle/ropp_fm_abel_refl *
!
! NAME
! ropp_fm_abel_refl - Forward model calculating a one dimensional bending
! angle profile from refractivity / impact parameter profile
! at state vector levels using a modified Fast Abel Transform
! valid for impact parameters under the surface, which
! correspond to signals reflected off the surface.
!
! SYNOPSIS
! call ropp_fm_abel_refl(nr, refrac, temp, roc, und, Tgrad_oper, impact, bangle)
!
! DESCRIPTION
! This routine calculates bending angles for reflected signals, evaluated
! at a given set of impact parameters below the surface's impact parameter.
! It uses a vertical profile of rafractivity given at
! the state vector's set of
! x = nr levels.
!
! INPUTS
! real(wp), dimension(:) :: nr ! x = nr product
! real(wp), dimension(:) :: refrac ! Refractivity values
! real(wp), dimension(:) :: temp ! temperature values
! real(wp), :: roc ! radius of curvature
! real(wp), :: und ! undulation
! LOGICAL :: Tgrad_oper ! Use temp. gradient oper.
! real(wp), dimension(:) :: impact ! Impact parameters
!
! OUTPUT
! real(wp), dimension(:) :: bangle ! Forward modelled bending angles
!
! NOTES
! The model's equation can be found in Aparicio et al., 2018.
! The interpolation of bending angles calculated at the state vector's
! geopotential levels to the observation vector's impact parameters is
! carried out assuming that bending angle varies exponentially with
! impact parameter.
!
```

```
! SEE ALSO
! ropp_fm_types
! ropp_fm_bangle_1d_refl
! ropp_fm_bangle_1d
! ropp_fm_abel_ad
! ropp_fm_abel_tl
!
! AUTHOR
! Institute of Space Studies of Catalonia (IEEC), Barcelona, Spain, based on
! original code (no-reflection) by
! Met Office, Exeter, UK.
! Any comments on this software should be given via the ROM SAF
! Helpdesk at http://www.romsaf.org
!
! COPYRIGHT
! (c) EUMETSAT. All rights reserved.
! For further details please refer to the file COPYRIGHT
! which you should have received as part of this distribution.
!
!****

!------------------------------------------------------------------------------
! 1. Declarations
!------------------------------------------------------------------------------
USE typesizes, ONLY: wp =>EightByteReal
USE ropp_utils, ONLY: ropp_MDFV, ropp_ZERO, ropp_ZDTV
USE ropp_fm_constants, ONLY: pi, imp_ht_min

IMPLICIT NONE

REAL(wp), DIMENSION(:), INTENT(in) :: nr ! x = nr product
REAL(wp), DIMENSION(:), INTENT(in) :: refrac ! Refractivity
REAL(wp), DIMENSION(:), INTENT(in) :: temp ! Temperature
REAL(wp) , INTENT(in) :: roc ! Radius of curvature
REAL(wp) , INTENT(in) :: und ! Undulation
LOGICAL , INTENT(in) :: Tgrad_oper ! Use temp. gradient oper.

REAL(wp), DIMENSION(:), INTENT(in) :: impact ! Impact parameter
REAL(wp), DIMENSION(:), INTENT(out) :: bangle ! Bending angle

REAL(wp), DIMENSION(:), ALLOCATABLE :: kval ! Exponential decay rate
REAL(wp), DIMENSION(:), ALLOCATABLE :: beta ! Temperature gradient
REAL(wp), DIMENSION(:), ALLOCATABLE :: nr_mid ! Average nr product
REAL(wp), DIMENSION(:), ALLOCATABLE :: temp_mid ! Average temp. of two levels
REAL(wp), DIMENSION(:), ALLOCATABLE :: dval ! Useful in BA computation
REAL(wp) :: t_upper ! Upper bound integral
REAL(wp) :: t_lower ! Lower bound integral
REAL(wp) :: refrac_low ! Refrac at lower level
```

```
REAL(wp) :: nr_low ! x=nr at lower level
REAL(wp) :: zed ! Impact height

REAL(wp) :: integral_diff ! Integral approximation
REAL(wp) :: erf_up, erf_low
REAL(wp) :: int_up, int_low
REAL(wp) :: zt
REAL(wp) :: dn_dx,p1,p2,p3
REAL(wp) :: as ! surf. impact parameter
REAL(wp) :: r1,r2,AA,BB ! aux. variables

REAL(wp), PARAMETER :: a = 0.3480242_wp
REAL(wp), PARAMETER :: b = 0.0958798_wp
REAL(wp), PARAMETER :: c = 0.7478556_wp

INTEGER :: n_lev, n_lower, n_impact
INTEGER :: i, i_bot, l, n_top
!————————————————————————————-
! 2. Useful variables
!————————————————————————————-

n_lev = SIZE(nr)
n_impact = SIZE(impact)

ALLOCATE(kval(n_lev-1))
ALLOCATE(beta(n_lev-1))
ALLOCATE(nr_mid(n_lev-1))
ALLOCATE(temp_mid(n_lev-1))
ALLOCATE(dval(n_lev-1))

!————————————————————————————-
! 3. Calculate lowest usable level (because of superrefraction)
!————————————————————————————-
n_lower = 1
DO i = n_lev, 2, -1
IF (nr(i) - nr(i-1) <10.0_wp) THEN
n_lower = i
EXIT
ENDIF
ENDDO
IF (n_lower >1) THEN
print *, "WARNING SUPERREFRACTION: current implementation"
print *, "of ropp_fm_abel_refl might not work properly "
print *, "in superrefractive profiles."
ENDIF

!————————————————————————————-
! 4. Calculate exponential decay rate and temp. gradient between levels
```

```
!——————————————————————————————————————-

DO i = 1, n_lev - 1

kval(i) = LOG(refrac(i)/refrac(i+1)) / MAX(1.0_wp, (nr(i+1)-nr(i)))
kval(i) = MAX(1.0e-6_wp, kval(i))

! limit the maximum kval so that refractivity gradient is approx half critical value

kval(i) = MIN(kval(i),0.157_wp/refrac(i))

! beta is the temperature gradient

beta(i) = (temp(i+1) - temp(i)) / MAX(1.0_wp, (nr(i+1)-nr(i)))

! mean nr

nr_mid(i) = 0.5_wp*(nr(i) + nr(i+1))

! mean temp.

temp_mid(i) = 0.5_wp*(temp(i) + temp(i+1))

dval(i) = (nr(i) -nr_mid(i))**2
ENDDO

!——————————————————————————————————————-
! 5. Calculate bending angles for observational heights
!——————————————————————————————————————-

bangle(:) = ropp_MDFV

! 5.1 Find top output impact value (surface level or closest below)
! Extrapolating from bottom level of the state vector to surface

r1 = nr(1)/(refrac(1)*1E-6_wp+1.0_wp)
r2 = nr(2)/(refrac(2)*1E-6_wp+1.0_wp)
write(*,*) 'in abel_refl values of r1 and r2 ',r1, r2
write(*,*) 'in abel_refl values of refrac1 refrac2 ', refrac(1), refrac(2)
write(*,*) 'in abel_refl values of roc und', roc, und, roc+und
BB = (LOG(refrac(1))-LOG(refrac(2)))/(r2-r1)
!AA = EXP(LOG(refrac(1))+BB*(r2-roc-und))
AA = refrac(1) / (exp(-BB*(r1-roc-und)))
write(*,*) 'in abel_refl values of BB AA',BB,AA
! extrapolation of refractivity towards the surface:
! N=AA*exp(-BB*(r-rsurface))
! where AA is refractivity at the surface level.
! Then impact parameter at the surface is:
```

```
as = (AA*1.0E-6_wp+1.0_wp)*(roc+und)
write(*,*) 'in abel_refl value of impact at the surface ',as

! which limits the uppermost reflected impact parameter
! to be evaluated within the given impact vector

DO l = 1, n_lev
IF (nr(l) >as ) THEN
n_top = l-1
EXIT
ENDIF
ENDDO
write(*,*) 'n_top in abel and n_lev',n_top, n_lev
IF ( n_top <1 ) THEN
print *, "WARNING: impact parameters to be evaluated as reflected signals must be below the surface one."
ENDIF


! 5.1 Bottom state vector level is always the lowest one
! In fact, it should be the surface level, but here
! restricted to cases free of superrefractivity.
! i_bot = n_lower which should be 1 except for
! superrefractive profiles
! ——————————————————

!i_bot = n_lower ! NO, now set to 1 always for reflected signals
i_bot = 1

! 5.2 Loop over all levels above
! ——————————
! DO l = 1, n_impact – now only evaluating impact parameters up to surface IF (n_top >= 1) THEN
DO l = 1, n_top

bangle(l) = ropp_ZERO


DO i = i_bot, n_lev - 1

! 5.2.1 Values of refractivity and impact parameter at lower level
! ———————————————————————-

refrac_low = refrac(i)
nr_low = nr(i)
```

```
IF (refrac(i+1)-refrac(i) > - ropp_ZDTV ) THEN

! 5.2.2 If the refractivity gradient is +ve with height
! -----------------------------------------------

! This will handle the cases where the refractivity goes up with height

dn_dx = (refrac(i+1)-refrac(i))/(nr(i+1)-nr(i))

t_upper = SQRT( nr(i+1)-impact(l))

t_lower = 0.0_wp

IF (i > i_bot) t_lower = SQRT( nr(i)-impact(l))

bangle(l) = bangle(l) - &
2.0E-6_wp*SQRT(2.0_wp*impact(l))* dn_dx*(t_upper-t_lower)

ELSE

! 5.2.3 Upper and lower bounds of the "normal" integral
! -----------------------------------------------

t_upper = SQRT(kval(i) * (nr(i+1) - impact(l)))

IF (i == i_bot) THEN
t_lower = 0.0_wp
ELSE
t_lower = SQRT(kval(i) * (nr(i) - impact(l)))
ENDIF

! 5.2.4 Error functions
! ----------

! Approximate error function with polynomial

zt = 1.0_wp / (1.0_wp + 0.47047_wp * t_lower)
erf_low = 1.0_wp - (a-(b-c*zt)*zt) * zt * EXP(-(t_lower*t_lower))
zt = 1.0_wp / (1.0_wp + 0.47047_wp*t_upper)
erf_up = 1.0_wp - (a-(b-c*zt)*zt) * zt * EXP(-(t_upper*t_upper))

IF (i == n_lev-1) erf_up = 1.0_wp ! limit at infinity


! 5.2.5 New terms for integral that now allows kval to vary within layer
! -----------------------------------------------

! These p1, p2,p3 values correspond the case where dT/dx = beta = 0, i.e kval is constant!
```

```
p1 = kval(i)
p2 = 0.0_wp
p3 = 0.0_wp

! impact height of level

zed = nr(i) - roc

IF ( i <n_lev-1 .AND. zed > imp_ht_min .AND. Tgrad_oper) THEN

! compute the "p" values for temp. gradient beta
p1 = kval(i)*(1.0_wp + beta(i)/temp_mid(i)* &
(0.5_wp*kval(i)*((impact(l)-nr_mid(i))**2 - dval(i)) - &
(impact(l)-nr_mid(i))))

p2 = kval(i)*beta(i)/temp_mid(i)* &
(kval(i)*(impact(l)-nr_mid(i))-1.0_wp)

p3 = 0.5_wp*kval(i)**2*beta(i)/temp_mid(i)

ENDIF

int_up = SQRT(pi/kval(i))* &
(p1 + 0.5_wp/kval(i)*(p2+1.5_wp*p3/kval(i)))*erf_up

int_up = int_up - EXP(-kval(i)*(nr(i+1)-impact(l)))* &
SQRT(nr(i+1)-impact(l))/kval(i)*(p2 + p3* ( &
(nr(i+1)-impact(l))+1.5_wp/kval(i)))

! lower limit of integral

int_low = 0.0_wp

IF (i > i_bot) THEN

int_low = SQRT(pi/kval(i))* &
(p1 + 0.5_wp/kval(i)*(p2+ 1.5_wp*p3/kval(i)))*erf_low

int_low = int_low - EXP(-kval(i)*(nr(i)-impact(l)))* &
SQRT(nr(i)-impact(l))/kval(i)*(p2 + p3* ( &
(nr(i)-impact(l))+1.5_wp/kval(i)))

ENDIF

integral_diff = int_up - int_low

! 5.2.6 Bending angle value
```

```
! ----------------------

bangle(l) = bangle(l) &
+ 1.0e-6_wp * SQRT(2.0_wp*impact(l)) &
* refrac_low * EXP(kval(i) * (nr_low - impact(l))) &
* integral_diff

ENDIF

ENDDO

! 5.3 Adding the term due to the reflection angle
! ----------------------------------------
bangle(l) = bangle(l) - 2.0_wp * ACOS(impact(l)/as)

ENDDO
ENDIF

DEALLOCATE(beta)
DEALLOCATE(nr_mid)
DEALLOCATE(temp_mid)
DEALLOCATE(dval)
DEALLOCATE(kval)

END SUBROUTINE ropp_fm_abel_refl
```

1.5 Preparing the forward model

`ropp_fm_abel_refl.f90` computes the reflected bending angles that correspon to impact parameters below the one corresponding to the Earth surface.

When the forward model of reflected signals needs to be obtained from the state vector alone, or from the state vector and driven by the sampled impact parameters in an observable vector that does not contain impact parameters below a_s, it is required to generate an observables vector that contains impact parameters below a_s. This is done in the routine `ropp_fm_bangle_1d_refl.f90`.

Acknowledgements

This work was carried out as part of EUMETSAT's Radio Occultation Meteorology Satellite Application Facility (ROM SAF) which is a decentralised operational RO processing center under EUMETSAT.

Bibliography

[1] E. Cardellach and S. Oliveras Assessment of a potential reflection flag product , ROM SAF Report N.23, http://www.romsaf.org/general-documents/rsr/rsr_23.pdf, 2016.

[2] J.M. Aparicio, E. Cardellach and H. Rodriguez Information content in reflected signals during GPS Radio Occultation observations, Atmos. Meas. Tech., 11, 1883-1900, https://doi.org/10.5194/amt-11-1883-2018, 2018.

[3] I. Curverwell The Radio Occultation Processing Package (ROPP) User Guide, Part IIa: Forward model module, Version 9.0, 31 December 2016.

ROM SAF (and earlier GRAS SAF) Reports

SAF/GRAS/METO/REP/GSR/001	Mono-dimensional thinning for GPS Radio Occulation
SAF/GRAS/METO/REP/GSR/002	Geodesy calculations in ROPP
SAF/GRAS/METO/REP/GSR/003	ROPP minimiser - minROPP
SAF/GRAS/METO/REP/GSR/004	Error function calculation in ROPP
SAF/GRAS/METO/REP/GSR/005	Refractivity calculations in ROPP
SAF/GRAS/METO/REP/GSR/006	Levenberg-Marquardt minimisation in ROPP
SAF/GRAS/METO/REP/GSR/007	Abel integral calculations in ROPP
SAF/GRAS/METO/REP/GSR/008	ROPP thinner algorithm
SAF/GRAS/METO/REP/GSR/009	Refractivity coefficients used in the assimilation of GPS radio occultation measurements
SAF/GRAS/METO/REP/GSR/010	Latitudinal Binning and Area-Weighted Averaging of Irregularly Distributed Radio Occultation Data
SAF/GRAS/METO/REP/GSR/011	ROPP 1dVar validation
SAF/GRAS/METO/REP/GSR/012	Assimilation of Global Positioning System Radio Occultation Data in the ECMWF ERA-Interim Re-analysis
SAF/GRAS/METO/REP/GSR/013	ROPP PP validation
SAF/ROM/METO/REP/RSR/014	A review of the geodesy calculations in ROPP
SAF/ROM/METO/REP/RSR/015	Improvements to the ROPP refractivity and bending angle operators
SAF/ROM/METO/REP/RSR/016	Simplifying EGM96 undulation calculations in ROPP
SAF/ROM/METO/REP/RSR/017	Simulation of L1 and L2 bending angles with a model ionosphere
SAF/ROM/ECMWF/REP/RSR/018	Single Frequency Radio Occultation Retrievals: Impact on Numerical Weather Prediction
SAF/ROM/ECMWF/REP/RSR/019	Implementation of the ROPP two-dimensional bending angle observation operator in an NWP system

ROM SAF Reports are accessible via the ROM SAF website: http://www.romsaf.org