ROM SAF Report 16
Ref: SAF/ROM/METO/REP/RSR/016
Web: www.romsaf.org
Date: 1 August 2016

The EUMETSAT
Network of
Satellite
Application
Facilities

ROM SAF
Radio Occultation Meteorology

ROM SAF Report 16

Simplifying EGM96 undulation calculation in ROPP

Dave Offiler

Met Office, Exeter, UK

Document Author Table

	Name	*Function*	*Date*	*Comments*
Prepared by:	D Offiler	ROM SAF Project Team	1 August 2016	
Reviewed by:	I D Culverwell	ROM SAF Project Team	3 Oct 2013	
Reviewed by:	RW Saunders	Head, Satellite Applications	1 Aug 2016	
Approved by:	K B Lauritsen	ROM SAF Project Manager	3 Aug 2016	

Document Change Record

Issue/Revision	*Date*	*By*	*Description*
1st draft	18 Sep 2013	DO	First draft
2nd draft	26 Nov 2013	DO	Updates following review by IDC and comments from presentation at PT13
1st release	1 Aug 2016	DO	Minor corrections following review by RWS

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

Intellectual Property Rights

Abstract

This report reviews the calculation of *undulation* – the difference between the (WGS-84) ellipsoid and the (EGM96) geoid – in ROPP. The undulation value at the nominal RO profile location is subtracted from the *Impact Height* to obtain the height with respect to the geoid of any point in the profile. For practical purposes, the resulting *geoid height* can be considered to be the same as *height above mean sea level*. The current method of calculating undulations (or geoid heights in ROPP) uses a full *spherical harmonic* expansion which is both complex and cpu-intensive. An alternative *grid interpolation* method has been developed which is shown to be significantly simpler in code terms and so substantially quicker to run, and without any practical loss of accuracy when compared to the present full calculation method. The interpolation method is proposed to replace the current undulation code in ROPP.

Contents

1 Introduction

1.1 Background

Geodesy is the study of the measurement and representation of the Earth, including its gravitational field. A thorough review of the geodesy calculations implemented in ROPP has recently been undertaken (Culverwell, 2013 [1]) as a result of user concerns. The conclusion was that the current calculation details are consistent and quite sufficiently accurate for radio occulation applications.

This present report re-reviews the calculation of *undulation* (see Section 6 in Culverwell, 2013 [1]) to determine whether an alternative, simplified, method of calculation would be appropriate to implement in a future release of ROPP. The proposed method is already implemented for use in the EUMETNET E-GVAP project for converting the ellipsoidal heights of ground-based GNSS sites to heights above mean sea level for consistency with other meterological observations.

Locations relative to the Earth are commonly expressed in the well-known 3-D coordinate system latitude, longitude and height (LLH). However, to be precise, the *datum* to which this system is relative must be stated. The datum used for GNSS LLHs is usually an *ellipsoid* – a smooth approximation of the Earth's latitude-dependent shape – and is typically the WGS-84 ellipsoid datum (NIMA, 1997 [8], NGA, 2013c [7])[1]. In fact, this datum is so universally used for GNSS that it is commonly implied, rather than explicitly stated, when refering to GNSS LLHs. Further, GNSS latitudes are usually (but rarely stated) to be in *geocentric* measure; this means that the latitude value (in degrees) is the simple angle subtended between the point and the Equator at the Earth's centre of mass[2].

Many applications – including meteorology – require coordinates, and especially the height component, to be relative to *mean sea level* (MSL) which is technically the *orthometic height*[3] of a point. MSL varies depending on the local gravitational value, and so is a function of latitude and longitude[4]. Although not strictly true, the 'shape' of MSL around the Earth is often used synonymously with the *geoid* – a surface of constant gravitational potential. The geoid is the shape that the surface of the oceans would take under the influence of Earth's gravity, in the absence of other influences such as winds, tides and ocean currents[5] on a non-rotating Earth. The geoid is commonly modelled as an offset height from a reference ellipsoid; a commonly used geoid is EGM96 (NASA, 1997 [2], NASA, 1998, [3], NASA 2004 [4]) which is defined relative to the WGS-84 ellipsoid assuming *tide-free* conditions. See Appendix 1.

More recent geoids include EGM2000 and EGM2008 (see Appendix 2), but these are refinements on EGM96 which do little to benefit meteorological applications, at the expense of complexity[6].

The geoid is measured directly and indirectly by various methods incuding ground-, air- and satellite-based instruments (NASA, 1997 [2], Pavlis, 2012 [9]); the accuracy of the geoid varies according to

[1] Semi–major (equatorial) axis: $a = 6,378,137.0$m; Semi–minor (polar) axis: $b = 6,356,752.3142$m; Flattening parameter: $1/f = 298.257223563$ (where $f = (a-b)/a$); zero offset of origin from Earth's centre of mass.

[2] mapping coordinates usually imply *geodetic* latitudes, where the value is the angle between the point and Equator, for which the point is projected along the gravitational normal to the surface at that point; the intersection of this line and the Equatorial plane is in general not coincident with the centre of mass of the Earth.

[3] see, for instance, http://en.wikipedia.org/wiki/Orthometric_height.

[4] and height, but here we are concerned with points on or close to the Earth's surface, and gravity changes with height are relatively small

[5] http://en.wikipedia.org/wiki/Geoid

[6] but the increased accuracy is vital for certain scientific, commercial and military applications

the location – local gradients in gravity, sensor type and accuracy and density (resolution) of measurements. The geoid is in general more accurate over land than over sea, for instance (and is most accurate over the US and Europe), and has the largest error over large gravitational anomalies such as that near Hawaii and around the Pacific Rim. Typically, EGM96 is felt to have an accuracy of about 0.1–0.5 metres (1σ), but perhaps 1–5 m worst-case (NASA, 1997 [2]). Later versions of EGM aim to both improve on this accuracy and resolve smaller gravitational features, but for our purposes, an accuracy of 1m with a resolution of 100 km or so is sufficient when considering requirements for radio occulation applications (Culverwell, 2013 [1]).

For practical use, the geoid itself is not usually modelled directly, but it is the offset height beween geoid and ellipsoid which is modelled. The EGM96 'surface' is expressed in terms of *spherical harmonics* defined by a set of Legendre polynomials in latitude and sinusoids in longitude to degree and order 360 × 360, which is sufficient to capture the principle geoid 'shape' with a wavelength of 1 degree, or about 110 km at the equator. In comparison, EGM2008 is modelled to degree 2060 × 2060, or 18 km at the Equator (Pavlis, 2012 [9], NASA, 2013b [6]).

In order to convert heights (points in a radio occulation profile, or ground-based GNSS receiver heights, for instance) given on the WGS-84 ellipsoid datum, H_e, to geoidal (MSL) heights, H_g, one needs to calculate the undulation, U, (the EGM96 offset height from WGS-84) at the latitude and longitude of the point, then subtract this value from the WGS-84 ellipsoidal height, thus:

$$H_g = H_e - U(lat, lon) \tag{1.1}$$

1.2 Motivation

This report reviews the calculaton of U as implemented in ROPP, and compares this method to an alternative, simplified interpolation method. The performance of the latter, in terms of mathematical interpolation error, computation speed and consequences for applications, is presented and discussed, with a potential to replace the current calculation with the new one.

Note that we do not here discuss converting between geocentric and geodetic latitudes, nor the conversion of geoidal heights to geopotential heights. The latter is covered in Culverwell, 2013 [1].

The new, alternative grid interpolation method has been used in the E-GVAP software package to calculate GNSS station MSL heights from the WGS-84 LLH coordinates since November 2011 when it replaced the spherical harmonics method still used in ROPP. The motivation was to simplify the code and as a consequence – though this was not a driver – speed up the calculation. It was also required to have an equivalent pure Python-based function, and porting the Fortran spherical harmonics code was not justified when a simple interpolator would do the job.

While the new method (implemented in both Fortran 90 and Python) was fully validated at the time, and judged appropriate for the ground-based GNSS station heights for NWP applications, the nature of this software package[7] meant that this change was not formally documented. In part, this report is to remedy that omission, and partly to support the case for using the same undulation code in all aspects of GNSS data processing, including ROPP, so as to:

- contribute to simplifying the ROPP code-base (ROPP SVN Track Ticket #245) by reducing the total number of routines, source code and data files. Code consolidation, optimization and maintenance is the purpose of CDOP-2 WP3420 'Maintenance of Deliverable Software';

[7] the 'GWV' package is free to authorised E-GVAP participants, but comes 'as-is' and without any formal support. ROPP, on the other hand is fully supported, and as a consequence must be transparently documented and changes fully and openly validated.

- provide a single, unified code for consistent re-use between RO and ground-based GNSS modules
- simplfy the calculation method (and as a side-effect, speed it up) to aid future maintenance and provide flexibility to implement other geoid models merely by dynamically loading alternate grid-based files.

1.3 Report structure

The remainder of this report is organized as follows:
Section 2 summarizes the EGM96 undulation model and its uses;
Section 3 discusses the ’official’ NGA Fortran-77 code and data files for EGM96 and validates the performance for reference purposes;
Section 4 reviews the current calculation method in ROPP and its performance relative to the NGA implementation (Section 3);
Section 5 introduces the alternative method and tests its performance against the ROPP version (Section 4);
Section 6 provides a summary of results;
Section 7 gives an overall summary, conclusions and recommendations;
Appendix 1 (Section 8) includes some information about the EGM96 geoid model;
Appendix 2 (Section 9) includes some information about the new EGM2008 geoid model. Note that evaluation of EGM2008 is outside the scope of this report;
Appendix 3 (Section 10) provides documentation on the ROPP SH method;
Appendix 4 (Section 11) provides similar documentation for the proposed new method.

2 EGM96 Undulation

As described in the Introduction, the geoid — that is, the equipotential surface that best fits the Earth's mean sea level — is different from the ellipsoid datum – the latitudinal best-fit near-spherical surface – normally used for GNSS 3-D locations. The two differ because of localised variations in the Earth's gravitational field. The difference between the height of the geoid and the height of the ellipsoid is known as the *undulation*, U and is a function of latitude and longitude.

The undulation of the EGM96 geoid model (NASA, 1997 [2], NASA, 1998, [3], NASA 2004 [4]). is plotted in the top panel of Figure 2.1, and the magnitude of its gradient is plotted in the bottom panel (note the logarithmic colour scale). The undulation values were calculated using the full degree and order 360 spherical harmonic calculation (Section 4) on a 1/4° × 1/4° lat/lon grid.

The range of undulations are no greater than about ±100 m – the magnitude is typically less than 50 m — and with the limited resolution, appears relatively smooth. The highest gradients (and hence local variability) follow mountain ranges and undersea ridges; the high gradient values near Hawaii, the Pacific Rim, Central Asia and along the Andes may be noted.

Around half the points have undulation gradients greater than about $2{\times}10^{-5}$ m/m in magnitude. Over a 400 km drift in the tangent point during a radio occultation event, this could accumulate to a change in undulation of about 8 m, which is unlikely to be significant. In ROPP, therefore, a fixed undulation is presently assumed throughout the occultation, calculated at the 'nominal' latitude/longitude of the profile.

In most ROPP applications the undulation is given by the data provider as a single number, and for ground-based GNSS data provided under the EUMETNET E-GVAP project, the sites' heights are given (mostly, but not always) in both WGS-84 and EGM96 datums. Where an undulation or geoid value is not provided, it must be calculated prior to, for instance, BUFR encoding, so as to be consistent with heights used for other observational data in NWP models, etc.

The `ROPP_PP` module, which generates bending angles and refractivities from level 1a excess phase data, the (single) undulation at the nominal tangent point location is calculated according to the EGM96 model. The future `ROPP_GBG` module will also need to calculate the EGM96 undulation so that all GNSS sites' geoid heights can be be provided to users. For futher use of undulation values in ROPP height conversions, see Sections 5 and 6 in Culverwell, 2013 [1].

If a better resolved undulation were desired at some point in the future, then it may become necessary to use a higher order expansion, such as EGM2008 (Pavlis, 2012 [9]), which is of order and degree 2159, and which has a minimum wavelength of about 18km at the Equator. However, EGM96 is felt to meet requirements for meterological applications of GNSS data at the present time[1].

[1] The ROM SAF PT11 minutes record that the SAF and EUMETSAT agreed to standardize on EGM96. This issue should be raised at IROWG with a view to gaining international agreement.

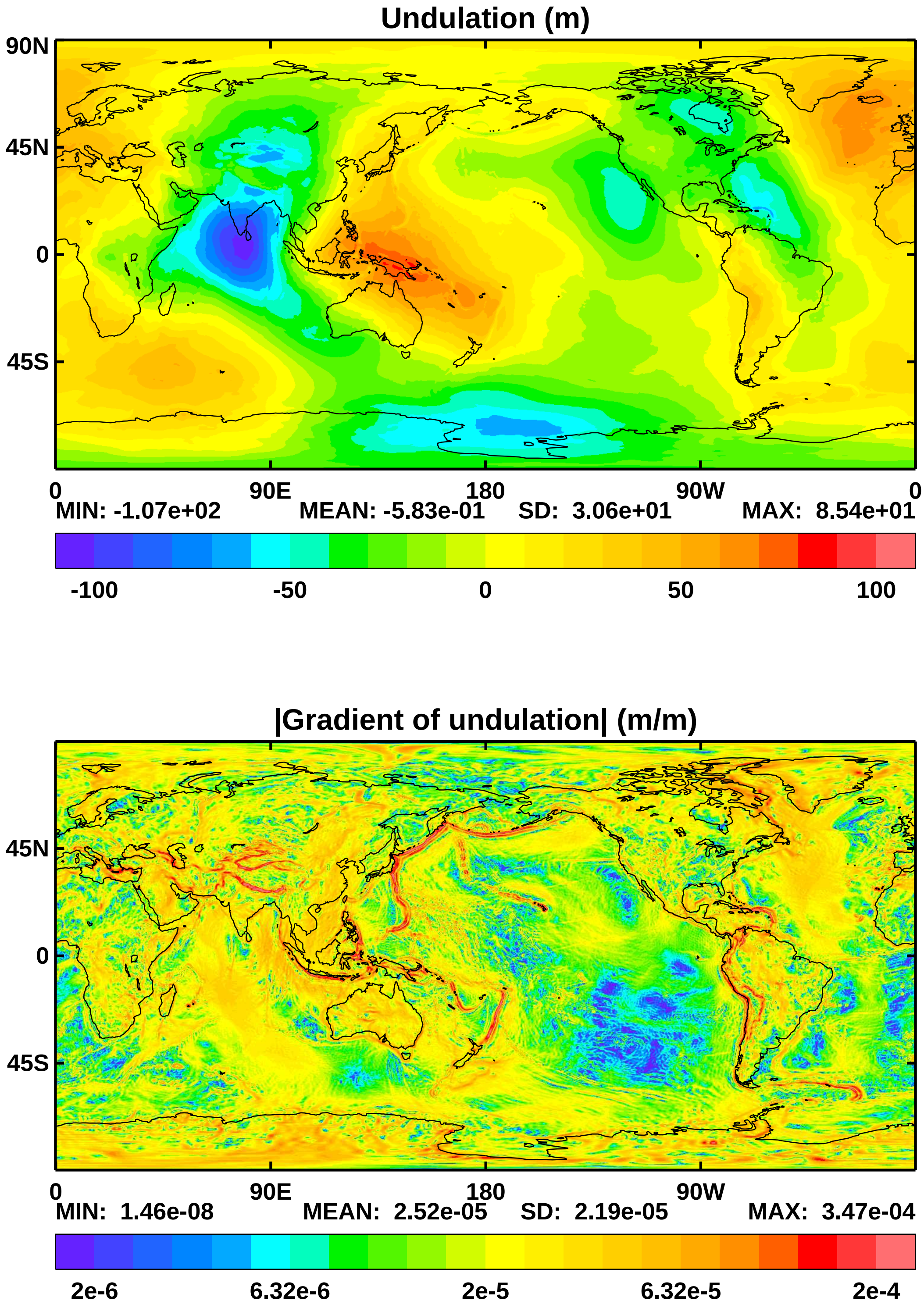

Figure 2.1: Spatial variation of: (top) undulation; and (bottom) magnitude of the gradient of undulation on a 1/4° × 1/4° grid. *(From Culverwell, 2013 [1].)*

3 Validation of the NGA implementations

Appendix 1 provides a summary description of the 'official' NGA implementation of EGM96 in the form of Fortran-77 programs and associated coeffcients and grid files ([6]). This section reports on the comparison of outputs using:

1. `F477.F` – Fortran-77 code implementing the full spherical harmonics (**N-SH**) method.
2. `INTPT.F` – Fortran-77 code implementing the grid interpolation (**N-GI**) method.

for a common set of input latitudes and longitudes.

Acknowleging that NGA are the experts, these programs can be assumed to be algorithically correct and the N-GI version is set to give optimum accuracy relative to the full N-SH version. Comparing the outputs from these two programs will give a good idea of the practical interpolation accuracy that NGA themselves feel is appropriate and so a goal for us to aim at.

In the following, the two F77 programs are compiled[1] as downloaded except:

- the output format statements have been modified to write out undulation values with four decimal places instead of three to reduce rounding errors of 1 mm due to limited numerical precision in the output files (but noting that the grid file itself is limited to 1 mm);
- Messages written to the termimal (*stdout*) relating to latitudes out of valid range and histograms are commented out of `INTPT.F`;
- the `IWINDO` parameter in `INTPT.F` is systematically modified to change the spline window size from 2×2 (actually a bilinear interpolation) up to 9×9 and the program re-run to test the interpolation accuracy of each option.

3.1 Grid Point vs. Spherical Harmonics files

The EGM96 undulations gridded file `WW15MGH.GRD` is pre-computed by NGA at each grid point using the SH method. However, it is not clear whether this has been done using the `F477.F` program (or similar F77 code based on it) or by some other implementation. So our first check is that the grid point values are consistent by re-generating the grid lat/lon points and calculating undulations with `F477.F` and comparing them with the `WW15MGH.GRD` file values.

With a simple statistical comparison (Table 3.1) we can confirm that the two sets of undulation values are identical to within ±0.3 mm (worst-case ±0.5 mm), which is consistent with the limited grid file's numerical precision of 1 mm. When plotted on a map (not shown), there is no systematic pattern to the differences, just a random horizontal distribution of values. Hence we can confidently state that the grid file accurately represents EGM96 *on the grid points, including the poles*, and when grid point values are given *to the nearest 1 mm*.

[1] In all cases, we use the Intel 'ifort' XE-2011 (v12.0) Fortran 90 compiler with default optimisation level, on a quad-core, desktop PC running 64-bit RHEL6 OS.

Npts	Mean	RMS	Minimum	Maximum
1,038,961	0.0	0.3	-0.5	0.5

Table 3.1: N-GI minus N-SH undulation difference statistics (mm) evaluated on grid file lat/lon points

3.2 Interpolation errors

In this Section we test the N-GI implementation on grid points (effictvely no interpolation) and at arbitary locations on the earth (i.e. not at grid point locations)[2]. Again, we assume that N-SH represents the 'true' EGM96 and we are evaluating N-GI differences (interpreted as 'interpolation error') from that reference.

The validation method involves the following steps:

1. generate a 2-D grid of latitudes and longitudes with a specified range and grid-point interval. For the purposes of this report, the same 1/4° grid as `WW15MGH.GRD` is generated (as done in Section 3.1);
2. the actual grid points are then offset by a fraction (zero, half or a random amount, case-by-case) of the grid interval in each direction. For the two offset cases, the grid is reduced by one row and column to keep the lat/lon within the usual ranges;
3. run the `F477.F` and `INTPT.F` programs, recording the elapsed times;
4. from the two output files, calculate statistics of the mean, RMS, minimum and maximum differences between N-SH and N-GI undulations, ignoring any invalid undulations near the poles (see below);
5. re-run `INTPT.F` with different values for `IWINDO`, the parameter controlling the spline window size.
6. convert the pairs of text file outputs to difference fields and save as netCDF for statistical analysis and plotting.

It should be noted that except for the smallest 2×2 spline window size (which is really a bilinear interpolation), `INTPT.F` cannot compute points close the poles and even the bilinear can only reach the South Pole; as the spline window is increased, the limit gets further from the poles. If the program cannot compute an undulation, it returns a value of `-999999.0` – these points are not included in the statistics, so the number of valid points may be less than the full test grid size of 721×1441=1,038,961 (no offset) or 720×1440=1,036,800 (offset grid) points.

3.2.1 No grid offset

Here, the lat/lon grid points are the same as used to generate the gridded undulations file. This checks that the basic algorithm is working and does not introduce any biases or other artifacts.

Table 3.2 shows the results for the various values of `IWINDO`. From this, we see that all window sizes 'interpolate' to the same (within 0.5 mm) value as the full spherical harmonic calculation on the grid points – at least for locations as close to the poles as the N-GI algorithm allows. A map of the

[2] our local R-SH (current ROPP) and G-GI (used in E-GVAP) implementations are evaluated in Sections 4 and 5, respectively.

IWINDO	Npts	Max. Lat	Mean	RMS	Minimum	Maximum	Notes
2	1,037,520	89.875	0.0	0.3	-0.5	0.5	bi-linear; can do S.Pole
3	1,037,520	89.875	0.0	0.3	-0.5	0.5	
4	1,034,668	89.625	0.0	0.3	-0.5	0.5	default
5	1,033,197	89.625	0.0	0.3	-0.5	0.5	
6	1,031,756	89.375	0.0	0.3	-0.5	0.5	
7	1,030,315	89.375	0.0	0.3	-0.5	0.5	
8	1,028,874	89.125	0.0	0.3	-0.5	0.5	
9	1,027,433	89.125	0.0	0.3	-0.5	0.5	

Table 3.2: N-GI minus N-SH difference statistics for various N-GI spline window sizes (IWINDO) at exact grid points (no lat/lon offsets). Undulation statistics are in millimetres. *Max. Lat* is the magnitude of the closest latitude to the poles that INTPT.F returned a valid undulation for the given spline window size. The full grid contains 1,038,961 points

undulation differences (not shown) confirms that there is no systematic pattern to the differences, just a random horizontal distribution of values, consistent with the numerical precision of the gridded undulations file.

3.2.2 Half grid offset

Here, the lat/lon grid points are offset by exactly half a grid interval in each direction – that is 0.125° for our usual 1/4° grid. This will test the interpolation furthest from all the surrounding grid points, which is likely to result in the largest errors. However, the maximum sampled latitude is only 89.875°.

Table 3.3 shows the results for the various values of IWINDO. From this, we see that IWINDO=6 generates the smallest overall RMS errors, but IWINDO=4 (default setting) best constrains the worst-case errors. Apart from the smallest two window sizes, typical errors look to be about 6 mm RMS, with worst-case errors up to 120 mm. This is consistent with [2] which claims differences should be no more than 1–2 cm.

IWINDO	Npts	Max. Lat	Mean	RMS	Minimum	Maximum	Notes
2	1,036,800	89.875	0.0	51.4	-1152.2	786.9	bi-linear; can do S.Pole
3	1,035,360	89.875	0.0	23.4	-498.1	358.5	
4	1,033,920	89.625	0.0	6.0	-95.1	95.1	default
5	1,032,480	89.625	0.0	5.9	-120.1	125.8	
6	1,031,040	89.375	0.0	1.5	-116.2	119.3	
7	1,029,600	89.375	0.0	1.8	-119.4	119.4	
8	1,028,160	89.125	0.0	2.1	-120.8	121.4	
9	1,026,720	89.125	0.0	2.0	-120.6	121.4	

Table 3.3: As for Table 3.3 but using half-grid point interval offsets. The full grid contains 1,036,800 points

A map and histogram of the undulation differences is shown in Figure 3.1. The interpolation errors clearly form a pattern with the largest values aligned with the higest gradients (local variability) in gravitational anomalies associated with mountains and tectonic plate boundaries seen in Figure 2.1. The vast majority of the difference values are under 20 mm with no error above 100 mm.

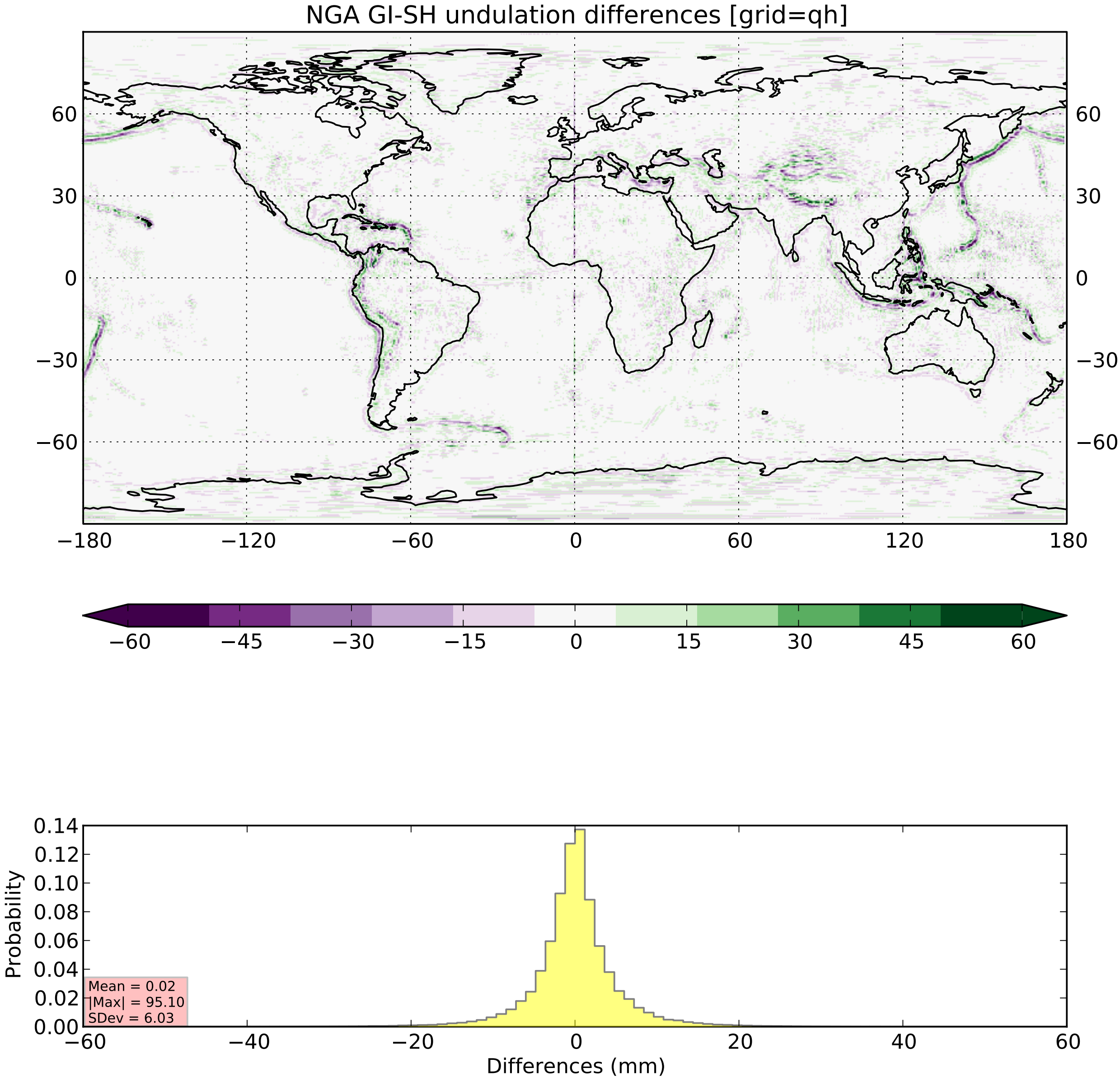

Figure 3.1: N-GI minus N-SH undulation differences (mm) for 1/4° × 1/4° grid, evaluated on half-grid point interval offset locations

3.2.3 Random grid offset

Here, the lat/lon grid points are offset by a random amount (uniform distribution) within each grid box in each direction – that is 0–0.249999° for our usual 1/4° grid. This represents a more real-life situation in which undulations are required at arbitrary lat/lon locations anywhere in the world. However, the maximum sampled latitude can be anything between 89.75° and 90.0°.

Table 3.4 shows the results for the various values of `IWINDO`. From this, we see that `IWINDO=6` again generates the smallest overall RMS errors, and `IWINDO=4` (default setting) best constrains the worst-case errors. Apart from the smallest two window sizes, typical errors look to be about 5 mm RMS, with worst-case errors up to 130 mm. While this latter figure appears higher than for the half-offset case, it may be that our random set happened to sample a particularly variable geoid zone which the fixed offset missed.

`IWINDO`	Npts	Max. Lat	Mean	RMS	Minimum	Maximum	Notes
2	1,038,240	90.000	0.0	37.0	-8764.6	746.7	bi-linear; can do poles
3	1,037,536	89.875	0.0	14.7	-336.6	279.5	
4	1,035,357	89.750	0.0	4.6	-82.8	95.9	default
5	1,033,893	89.625	0.0	3.7	-127.5	113.4	
6	1,032,473	89.500	0.0	1.2	-124.2	119.8	
7	1,031,069	89.375	0.0	1.3	-127.5	123.4	
8	1,029,584	89.250	0.0	1.5	-128.9	121.1	
9	1,028,151	89.125	0.0	1.4	-128.6	121.0	

Table 3.4: As for Table 3.3 but using random grid point offsets. Note that the statistics will vary a small amount from run-to-run as each generates a different set of random numbers.

A map of the undulation differences (not shown) is very similar in the error pattern to that in Figure 3.1 but with a slightly smaller spread of values.

3.3 Timings

The program elapsed (wall-clock) and CPU times[3] are shown in Table 3.5. These were for the no offset' case; the times for the 'half-grid offset' and 'random offset' points were very similar – see Tables 3.6 and 3.7.

`IWINDO`	Elapsed Time (secs)	CPU Time (secs)
SH	210.97	186.57
2	9.21	3.56
3	9.11	3.65
4	9.47	3.80
5	9.45	3.76
6	9.07	3.95
7	10.16	4.19
8	10.48	4.42
9	10.63	4.72

Table 3.5: Elapsed and CPU time to process on lat/lon grid points (N=1,038,961) using the N-SH and N-GI F77 programs.

As might be expected, the CPU times increase with the spline window size for N-GI[4], though the elapsed times are more variable. With `IWINDO=4`, a typical N-GI CPU timing is about 3.7 s per million points, compared with 181 s per million points for N-SH. Hence the N-GI method is apparently some 50 times faster than N-SH.

[3] using the GNU `time(1)` command-line utility 'real' and 'user' time values, averaged over five runs on a lightly-loaded desktop PC

[4] even allowing for N-GI skipping more near-pole calculations with larger window sizes.

IWINDO	Elapsed Time (secs)	CPU Time (secs)
SH	207.81	186.45
2	8.77	3.41
3	8.87	3.65
4	9.40	3.85
5	9.51	3.79
6	9.70	4.04
7	9.90	4.23
8	10.28	4.55
9	10.87	4.82

Table 3.6: As Table 3.5, but for half-grid offsets (N=1,036,800)

IWINDO	Elapsed Time (secs)	CPU Time (secs)
SH	1513.20	1504.81
2	8.77	3.46
3	9.67	3.62
4	9.42	3.78
5	9.32	3.80
6	9.55	3.97
7	10.21	4.22
8	10.73	4.49
9	11.33	4.86

Table 3.7: As Table 3.5, but for random offsets (N=1,036,800)

However, these times were measured ‘outside’ the exectuable programs at shell level, so will include the overheads of program startup and all I/O. This overhead will be relatively small for N-SH, but re-running N-GI with the interpolation subroutine call commented out (and replaced with a dummy fixed assignment for the undulation value) gave an average over five runs times of 9.0 s (3.5 s), as against 9.8 s (3.8 s) with the call. The difference therefore represents the undulation cpu-intensive calculation, so the N-SH/N-GI raw computation speed-up factor is more like 2,000). However approximate this timing is, it is clear that the N-SH is computationally-intensive and N-GI is very much faster – which after all, is one of the reasons for doing interpolation – while trading-off the slight loss of accuracy.

3.3.1 Summary

While there is some trade-off, these results confirm the default `IWINDO=4` (4×4 spline) in `INTPT.F` provides a near-optimum setting with minimum worst-case errors while keeping the RMS errors small. Smaller spline windows (and in particular the bilinear mode) do not fit well and larger windows perhaps over-fit with large worst-case errors — although the global RMS errors are smallest. Whether the 4×4 window size is optimum because this covers a $1^\circ \times 4^\circ$ area which happens to be the effective resolution of the gridded undulations is a coincidence or not remains open.

In summary, the best N-GI interpolation accuracy at random locations seems to give an absolute worse-case fit of <100 mm with an RMS interpolation error of around 5 mm (<5% of the assumed EGM96 model error) and zero bias, globally – at least when not closer than 0.25° to the poles when using a 4×4 spline window.

The N-SH method is shown to be significantly more expensive in CPU terms, with N-GI being some

50–2,000 times faster, depending on how this metric is measured.

4 Validation of the ROPP spherical harmonic implementation

Appendix 3 describes the history and documents the usage of the F90 **R-SH** implementation. When first written, this was validated by (limited) camparison to output from (a) the original NGA F77 code (N-SH) and (b) the official on-line calculator (NGA 2013a [5]) and found to give indentical spot results within the numerical precision of the calculator (0.01m) or better – but unfortunately this validation was never fully documented.

With some small updates, the author's orginal F90 EGM96 module was included in ROPP from version 5.0 for use in various `ROPP_PP` tools. (The same code is in the current release, ROPP v8.0.) As far as the author is aware, the modified ROPP version was not further validated comprehensively, only by spot-checking. This section remedies that deficiency.

4.1 Undulation differences

The ROPP SH undulations calculated with `Datum_HMSL()` (R-SH method) are compared to the NGA SH (N-SH) calculations using `F477.F` in a similar manner to that described in Section 3 – i.e. the R-SH subroutine is called from a stand-alone wrapper program which inputs the same set of pre-generated lat/lon points and the pair of output files are then compared.

In this case, both implementations can cover the poles.

Although the RMS differences were <0.5mm, again – again, consistent with the limited numerical precision of the output files – there was a (in this context) relatively large absolute maximum difference of almost 3 mm. This was at first attributed to numerical rounding errors in a very limited number of cases perhaps due to slightly different compiler optimisation between the F77 and F90 versions.

However, when plotted on a map – as in Figure 4.1 – it was clear that there was a systematic pattern around the equator. This pattern was only present in the grid offset cases, but not when undulations were calculated on the exact grid points; the SH wavelength is 1° and the error followed a sinusoid with zero crossing every 1/4°, with peak error half-way between grid points. The cause of this error pattern was tracked down to a long-standing bug in `Datum_HMSL()` which was reading, but not saving, the last pair of coefficients in each input file.

The remaining analysis is (re-)done having corrected the coding error in a working version of `earth.f90`[1]. Figure 4.2 confirms that the problem pattern has been completely removed.

The overall statistics are presented in Table 4.1. For all practical purposes then, the (corrected) R-SH code is validated to give the same undulation value as the 'official' NGA program (N-SH), so the former can be used as a more convenient reference when comparing the performance of the proposed alternative G-GI method (Section 5) since a callable API gives us more flexibility and control than a stand-alone program.

[1] See ROPP Trac Ticket #345. In the real world, a 3 mm error is insignificant, but any known systematic error should be eliminated.

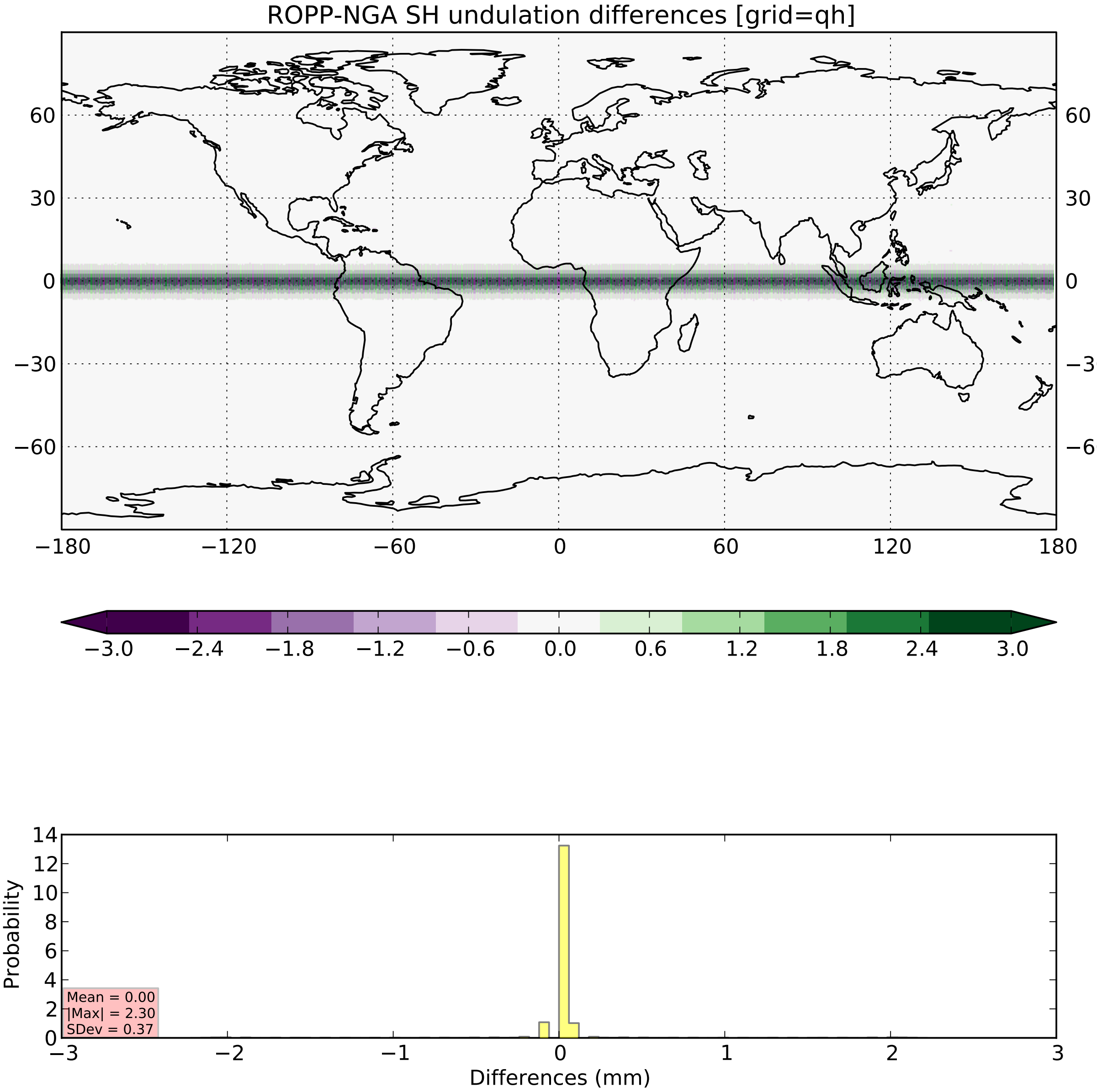

Figure 4.1: R-SH minus N-SH undulation differences (mm) for 1/4° × 1/4° grid, evaluated on grid-points, illustrating the 'equator bug'

Npts	Max. Lat	Mean	RMS	Minimum	Maximum
1,038,240	90.000	0.0	0.3	-0.5	0.5

Table 4.1: ROPP minus NGA SH undulation difference statistics. Values at the poles are included.

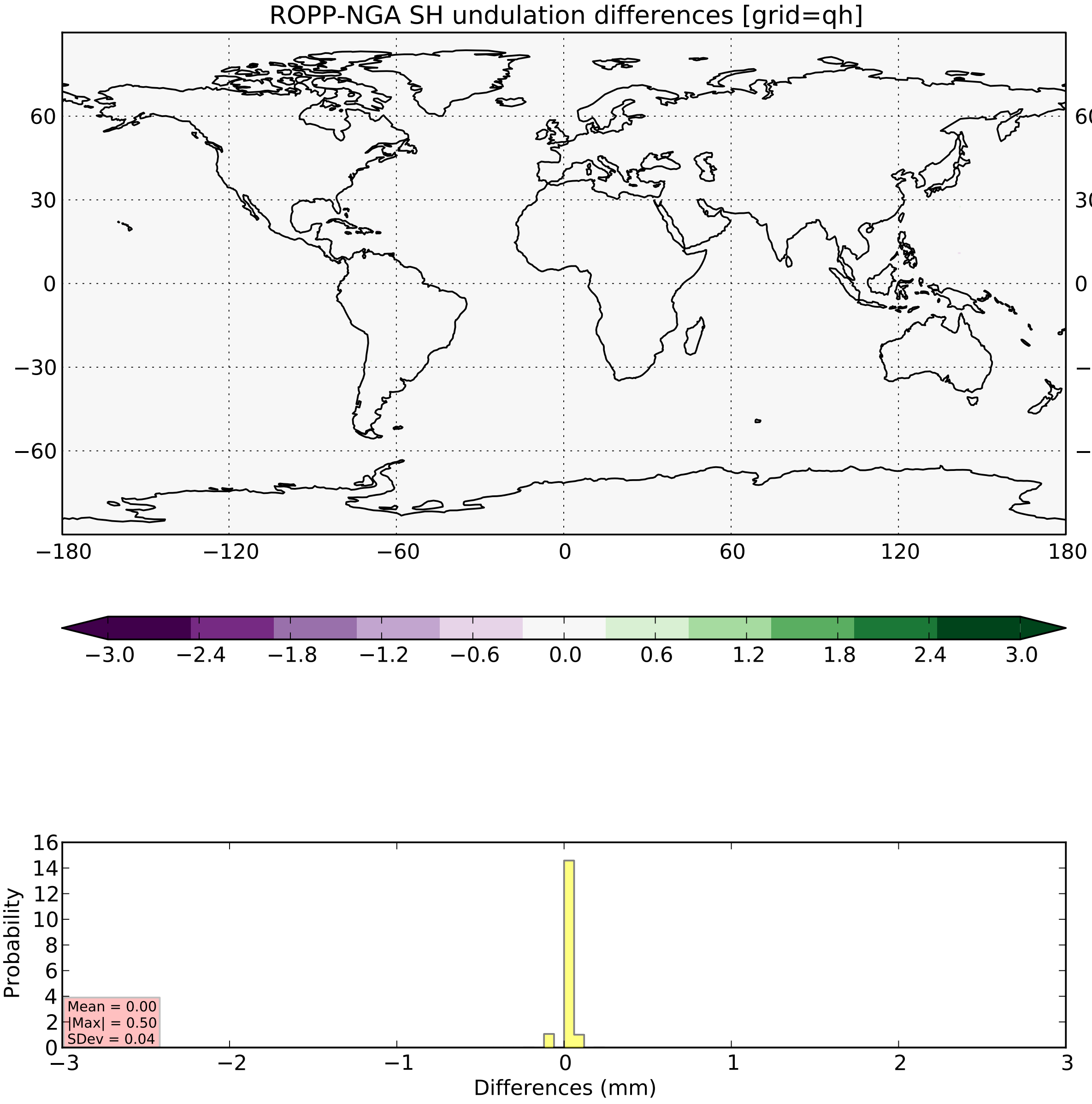

Figure 4.2: As Figure 4.1 but with ’equator bug’ fixed

4.2 Timings

Timings were measured as described in Section 3.3. In this case, the ROPP-based program executed in 1,490 s (24m50s), which is consistent with, if just slightly faster than, the N-SH program. While a difference of 20 s or so may be due to variations in the computer load at the time, the ROPP program was always slghtly faster than NGA when running the test program; it seems likely the (F90) compiler optimises the re-written F90 code more efficiently than the original F77 version.

5 Validation of the GWV grid interpolation implementation

Appendix 4 describes the history and documents the usage of the F90 and Python **R-SH** implementation. When first written, this code was validated by (limited) camparison to output from the original (pre-ROPP) `Datum_HMSL()` SH routine. and found to give indentical spot results within the expected numerical precision[1] – but unfortunately this validation was never fully documented, and only a summary of performance is quoted in the code header comments. This section remedies that deficiency.

The interpolation accuracy achieved by G-GI should ideally be no worse than the N-GI 4×4 spline window method implemented in `INTPT.F` (validated in Section 3) but in any case should attain an RMS interpolation error of less then 10% of the general EGM96 model accuracy itself (typically 100 mm or up to 5 m in the highest-variability areas), so an upper limit of 10 mm for RMS and 200 mm worst case error is our target. A speed-up factor (R-SH/G-GI) of more than 2,000 times would be nice too.

This section reports the results of basic testing in a similar manner to the previous sections, except that having validated R-SH (Section 4, we now use this F90 callable routine rather than the N-SH F77 program as our reference.

5.1 Undulation differences

For direct comparison, in this section we repeat the procedure described in Sections 3 and 4 but this time both the R-SH subroutine `Datum_HMSL()` and the G-GI function `EGM96_Undulations()` are called from separate stand-alone wrapper programs and the resulting pair of output files are compared as before.

Both implementations cover the poles.

The overall statistics for the 'no-offset' case are presented in Table 5.1, confirming that the basic interpolation algorithm is working as expected.

Npts	Max. Lat	Mean	RMS	Minimum	Maximum	Notes
1,038,961	90.000	0.0	0.3	-0.7	0.7	including poles

Table 5.1: G-GI minus R-SH undulation difference statistics (millmetres).

A more detailed comparison of the new G-GI method versus the reference R-SH method is shown in Section 6.

5.2 Timings

Timings were measured as described in Section 3.3. In this case, the G-GI program executed in 10.6 s[2], which is consistent with, if just slightly slower than, the N-GI program at 9.8 s[3].

[1] the bug noted in Section 4 was not detected here either

[2] both single- and double-precision APIs ran in the same average times

[3] both averaged over 5 runs. Note that the N-GI 4×4 spline method skips more than 4,000 points as being too close to the poles, so it should run more quickly.

By comparison, the R-SH program was timed at 1501.7 s, giving G-GI an apparant (wall-clock) speed-up factor of 140 over R-SH. Clearly, either of the GI methods out-performs either SH implementation in terms of execution speed by a large margin.

More useful timings at routine-, rather than shell-level, are presented in Section 6.

6 Performance results

In Section 4 we showed that the ROPP SH (R-SH) method produces undulations which are entirely consistent with the ’official’ NGA SH (N-SH) program. Since the ultimate objective is to replace the current R-SH code, this section reports on the results of comparing the new G-GI method against the R-SH method in more detail.

We stress that any differences reported here should be interpreted only as mathematical interpolation error; the R-SH undulations are treated as reference but we should bear in mind that the EGM96 model has a general accuracy of around 1 m (1σ) relative to the true geoid; some areas will be better, some worse (NASA 1997 [2]). As a goal, the G-GI method should not introduce interpolation errors more than 10% of the underlying EGM96 accuracy – i.e. SH-GI differences should be less than 10 mm (1σ or RMS) on average, or 200 mm worst case where the EGM96 gradients or local variabilities are large (Figure 2.1).

The validation method involves running a Fortran wrapper program which performs the following steps:

1. set up 2-D grids of latitudes and longitudes with a specified range and grid-point separation; the same 1/4°×1/4° grid as used in Sections 3 and 4 was employed here (although lower-resolution grids were used for quicker initial testing). The same three ‘exact’, ’half-grid offset’ and ’random-offset’ cases were generated in turn.
2. call `Datum_HMSL()` to pre-load the EGM96 coeffcients and corrections data files into memory and perform any internal set-up calculations;
3. intitialise a timer (system clock, resolution 1 millisecond);
4. call `Datum_HMSL()` in a double-loop over the latitude rows and longitude columns to generate a 2-D grid of R-SH undulation values;
5. stop the timer;
6. repeat steps 2–5 with `EGM96_Undulations()` for the G-GI calculations;
7. calculate the SH–GI difference grid;
8. save the latitude, longitude and undulation R-SH, G-GI and difference grids to a netCDF file;
9. calculate statistics of the mean, minimum, maximum and RMS of the R-SH, G-GI undulations and their differences over the whole grid; record out these statistics together with the loop calculaton elapsed times and the relative speed-up factor.
10. the difference field saved in the netCDF file is later visualised in map and histograms forms[1].

Figure 6.1 is a plot of the 1/4° G-GI minus R-SH undulations evaluated on the grid points (no offset); this tests the basic algorithm with effectively no interpolation, so differences should (ideally) be zero, limited only by the internal single-precision G-GI calculations.

Figure 6.2 uses same grid spacing, but undulations are evaluated half-way between the grid-points; this tests the basic interpolation procedure, and any differences are likely to be (but not always) maximal in regions where the undulation gradients are very non-linear.

[1] If you’re really interested, the graphics were generated using a Python-based script with matplotlib, Cartopy and Iris

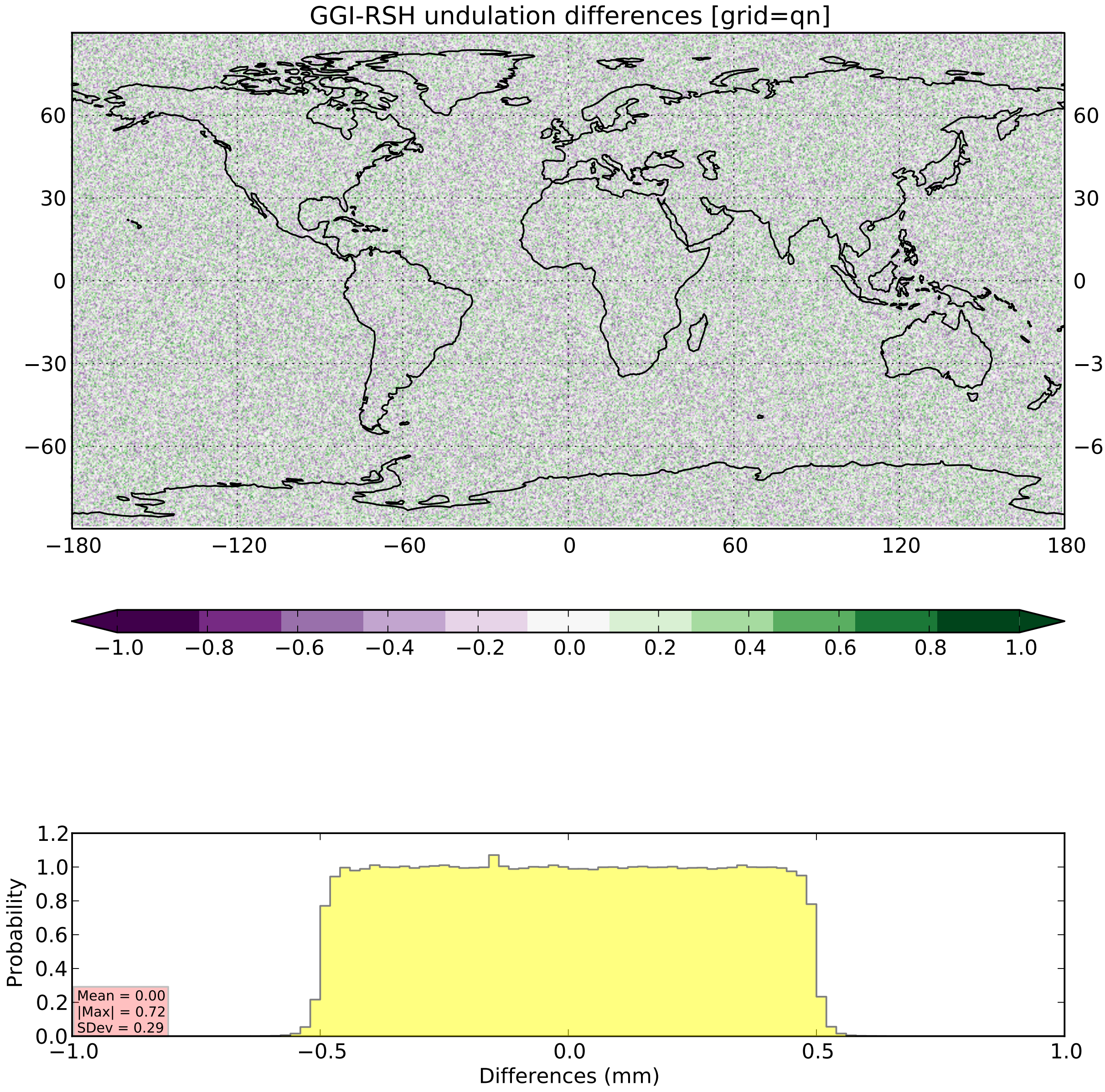

Figure 6.1: G-GI minus R-SH undulation differences (mm) for 1/4° × 1/4° grid, evaluated on grid-points.

Figure 6.3 again uses same grid spacing, but undulations are evaluated at a set of radom offsets between grid-points; this tests the performance in more real-world applications where undulations are required at arbitrary locations[2].

The statistical results are reported in Tables 6.1 (no offset), 6.2 (half-grid offset) and 6.3 (random

[2] this also the grid used in earlier sections of this report, except that the actual lat/lon values vary for each run as the random number seed is reset at the start of the wrapper program

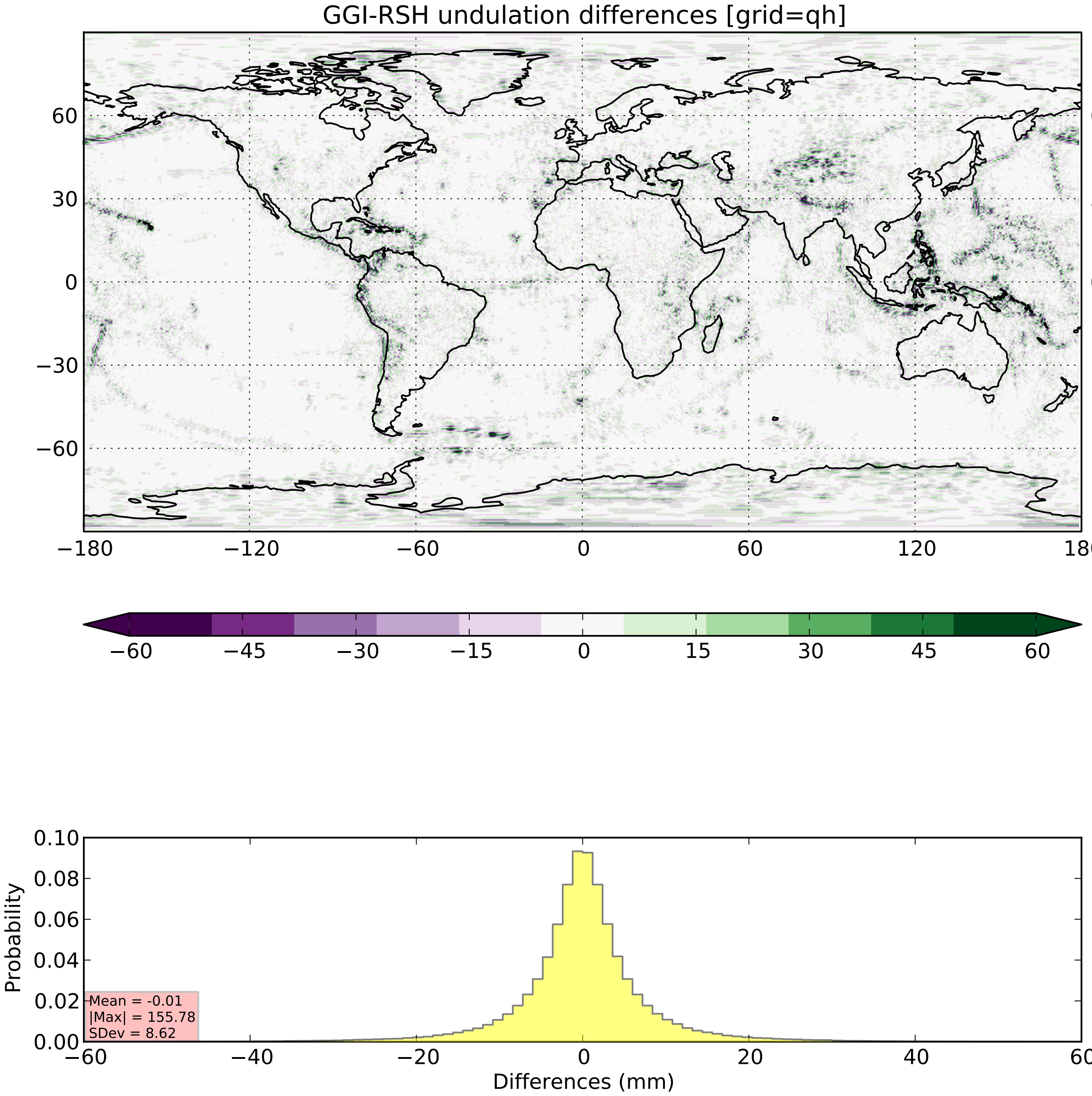

Figure 6.2: As for Figure 6.1 but evaluated half-way between grid-points.

offset). In each table, undulations and their differences are in metres, with elapsed times in seconds; the difference times are shown as the relative speed-up factor.

Method	Minimum	Maximum	Mean	RMS	Time
R-SH	-106.992	85.391	-1.465	29.274	3,070.453
G-GI	-106.991	85.391	-1.465	29.274	0.113
SH-GI	-4	4	0	0	27,163x

Table 6.1: Statistics for 1/4° × 1/4° grid with no offset. Npts=1,036,800 (720 × 1440)

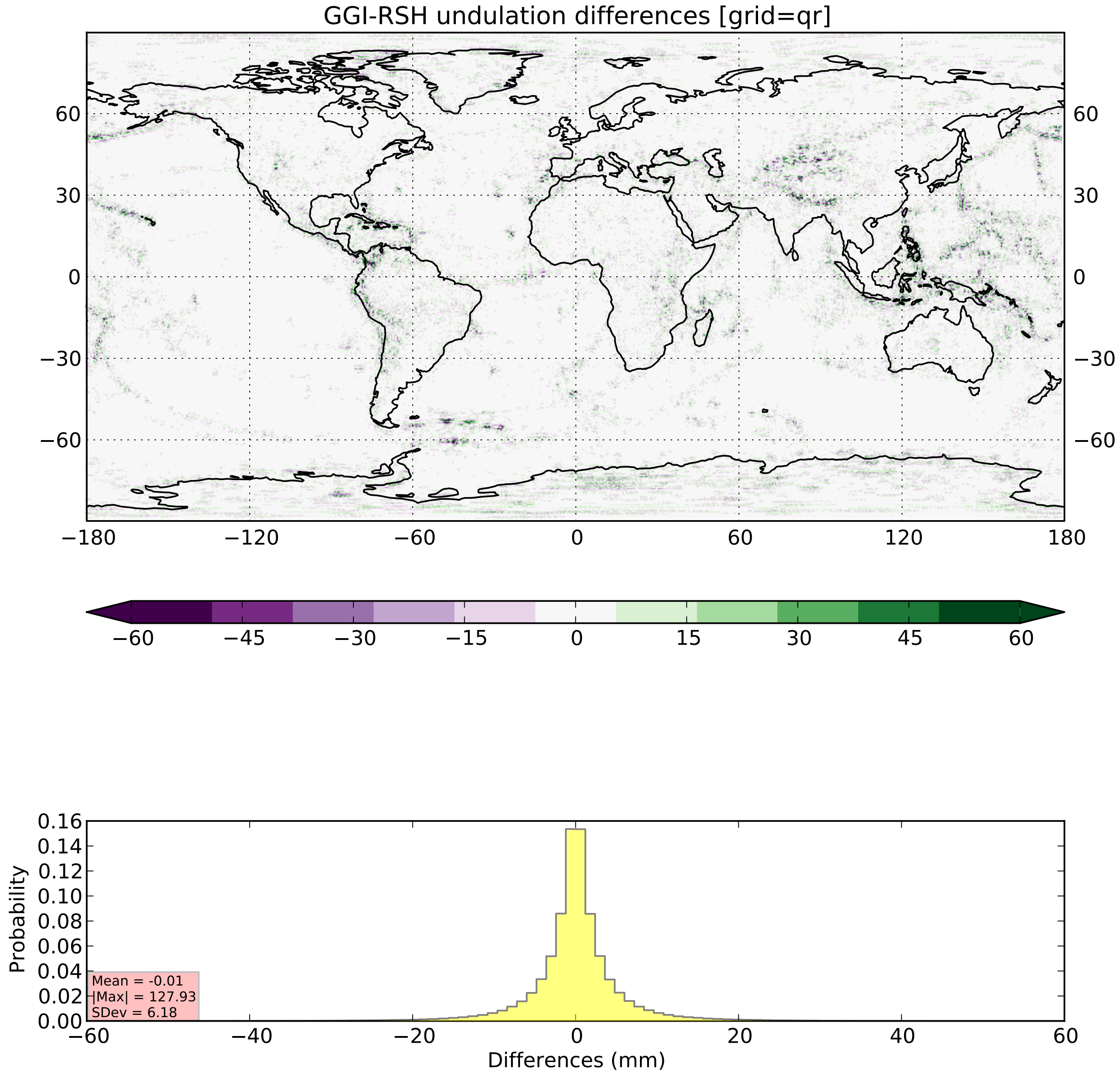

Figure 6.3: As for Figure 6.1 but evaluated at random grid-point offsets.

Method	Minimum	Maximum	Mean	RMS	Time
R-SH	-106.992	85.391	-1.405	29.257	3,081.372
G-GI	-106.991	85.391	-1.405	29.257	0.114
SH-GI	-4	4	0	0	27,026x

Table 6.2: As Table 6.1, but with half-grid offset

Given that the actual grid points evaluated for Table 6.1 are sampling randomly different locations (within the same grid boxes) from those in Table 5.1, the two sets of results are very similar.

Method	Minimum	Maximum	Mean	RMS	Time
R-SH	-106.985	85.327	-1.435	29.265	3,084.877
G-GI	-106.978	85.303	-1.435	29.265	0.114
SH-GI	-122	113	0	6	27,115x

Table 6.3: As Table 6.1, but with random grid offset

This analysis confirms that the G-GI method – implemented as a bi-cubic interpolation at random locations from a 1/4° × 1/4° pre-calculated undulation grid – easily meets our critera for RMS errors of less than 10 mm (6 mm achieved) and worst-case errors less than 500 mm (130 mm achieved) when compared with the R-SH method as reference.

The speed-up factor of the G-GI undulation calculations over R-SH (i.e. neglecting I/O) is approximately 27,000 times – for O(1 million) points, cpu time is well under 0.1 seconds against over 50 minutes (on a typical destop computer). However, the impact on ROPP tools – in which the undulaton calulation is only one small part, and which only need to calculate one undulation per profile — will be minimal.

7 Summary, conclusions and recommendations

7.1 Summary

This report has reviewed the EGM96 undulation calculations in ROPP. The key findings are as follows:

- The EGM96 model is thought to be generally accurate to around 1 m relative to the true geoid. Where there is a dense sensor network with good-quality observations and little variation in the geoid, the accuracy will be somewhat better (perhaps as good as 0.1 m), but in poorly observed areas and those with strong gravitational gradients (usualy due to high/low topography particularly under the oceans), worst-case accuracy may be nearer 5 m (NASA, 1997 [2]).
- EGM96 is implemented in ROPP following the spherical harmonic (SH) expansion in terms of Legendre Polynomials (NASA, 2004 [4], NGA 2013b [6]). This implementation (R-SH) has previously been validated against the original NGA F77 (SH) code and on-line calculator (NGA, 2013a [5]) and found to agree to better than 1 mm (or within the numerical precision of the on-line calculator). The [R-]SH method can be considered to be the definitive EGM96 model for determining undulation values at any arbitrary lat/lon location on the Earth. However, the calculation algorithm is quite complex and cpu-intensive.
- A simpler grid interpolation (G-GI) method has been implemented, designed to use a pre-calculated grid of undulation values (NGA 2013b [6]). A bicubic interpolation algorithm is found to be an optimal compromise between code simplicity (and hence speed of calculation and ease of maintenance) and minimising interpolation errors. Disk and core memory requirements for this method are similar to those of the R-SH method but cpu requirements are vastly reduced (by a factor of approching 30,000x in elapsed run-time and neglecting I/O overheads).
- A target interpolation error of less than 10% of the underlying EGM96 model accuracy is demonstrated. Typical interpolation errors are under 5-mm (1σ) and worst-case errors are less than 120 mm. This level of accuracy does not materially degrade the underlying EGM96 model error against the true geoid.

7.2 Conclusions

The overall conclusion is that the G-GI method has acceptable interpolation errors which do not degrade the basic EGM96 model accuracy when implemented using the full R-SH method. Geoid heights (and heights derived from them, such as geotential heights) will remain within user requirement, as previous established (Culverwell, 2013 [1]). There is no significant impact on disk space or run-time memory, but the cpu requirements can be very significantly reduced – though the impact on the run-time of ROPP tools will be very small.

7.3 Recommendations

1. Given the much simpler code base and vastly faster computation (not that the undulation calculation is seen as a bottle-neck in ROPP), it is recommended that the G-GI method replaces

the current R-SH code in ROPP, as it has in the GWV package for E-GVAP.

Since the API of the new code is quite different (and returns the undulation value itself for the caller to apply – rather than doing this within the routine as now, necessitating recovery of the undulation if this value is required), minimal changes to two tools in `ROPP_PP` and the library configuration file reader routine will also be necessary.

2. Whether the `egm96.f90` source code file should be in the `coodinates` sub-directory (where `earth.f90` now is) or under `geodesy` is less obvious, though the latter would seem to be more logical for this specialist function.

3. It would also be more logical for the the grid file be packaged in ROPP alongside the code which uses it (i.e. both in `ROPP_UTILS` – the two coefficents and corrections files are currently in `ROPP_PP`)

4. The GWV package does not pre-compress the grid file; this is left to the tarball compression stage. It is suggested to leave this file uncompressed in the ROPP source tree.

8 Appendix 1: The EGM96 model

8.1 Background

The abstract from NASA, 1998 ([3]) reads:

> The NASA Goddard Space Flight Center (GSFC), the National Imagery and Mapping Agency (NIMA), and the Ohio State University (OSU) have collaborated to develop an improved spherical harmonic model of the Earth's gravitational potential to degree 360. The new model, Earth Gravitational Model 1996 (EGM96) incorporates improved surface gravity data, altimeter-derived anomalies from ERS-1 and from the GEOSAT Geodetic Mission (GM), extensive satellite tracking data - including new data from Satellite laser ranging (SLR), the Global Positioning System (GPS), NASA's Tracking and Data Relay Satellite System (TDRSS), the French DORIS system, and the US Navy TRANET Doppler tracking system - as well as direct altimeter ranges from TOPEX/POSEIDON (T/P), ERS-1, and GEOSAT.
>
> The final solution blends a low-degree combination model to degree 70, a block-diagonal solution from degree 71 to 359, and a quadrature solution at degree 360. The model was used to compute geoid undulations accurate to better than one metre (with the exception of areas void of dense and accurate surface gravity data) and realizes EGM96 as a true three-dimensional reference system.
>
> Additional results from the EGM96 solution include models of the dynamic ocean topography to degree 20 from T/P and ERS-1 together, and GEOSAT separately, and improved orbit determination for Earth-orbiting satellites.

8.2 Files

Full information on EGM96 and related implementation in the form of Fortran code and data files can be found at: http://earth-info.nima.mil/GandG/wgs84/gravitymod/egm96/egm96.html.

Source code – written in Fortran 77 as stand-alone programs – is available to calculate undulations at arbitrary locations using either the full spherical harmonics (SH) method (`F477.F`) or by interpolating a grid (GI) of pre-computed undulations (`INTPT.F`).

The plain-text EGM96 gravitational anomaly spherical harmonics coefficients file (`EGM96`) and the companion height corrections file (`CORRCOEF`) are each 5.3 Mbytes[1] or 10.6 Mbytes in total. The version for interpolation is provided as a plain-text file (`WW15MGH.GRD`) which contains pre-computed undulations on a 15′×15′ (1/4°×1/4°) grid; the file size being 9.6 Mbytes. These three data files on the website are compressed using the standard Unix `compress` utility (Windows/DOS versions are also available), and need to be uncompressed locally before use.

Example input (`INPUT.DAT`) and output data files (`OUTF477.DAT` and `OUTINTPT.DAT`) are also provided for checking the correct running of the above programs.

[1] filesizes on disk after decompression

8.3 Calculation

The calculation of the geoid undulation, U, values using spherical harmonics takes place using the following procedure:

$$U(\theta,\lambda) = \zeta_z + \zeta^\star(r_E,\theta,\lambda) + C(\theta,\lambda)$$

where ζ_z = -0.53 m, $C(\theta,\lambda)$ is the sum of three correction terms and $\zeta^\star(r_E,\theta,\lambda)$ is:

$$\zeta^\star(r_E,\theta,\lambda) = \frac{GM}{r_E\gamma_E}\sum_{n=2}^{360}\left(\frac{a}{r_E}\right)^n \sum_{m=-n}^{n} C_{nm}Y_{nm}(\theta,\lambda)$$

where subscript E indicates an evaluation at an (ellipsoid) geocentric latitude corresponding to 90°–θ and at longitude λ. The even-degree zonal value of C_{nm} represents the difference between the EGM96 coefficient and the WGS-84 reference value. The values are calculated for the even zonal coefficients J_{2n} by:

$$C_{2n}(ref) = -J_{2n}/\sqrt{(2n+1)}$$

with J_{2n} pre-computed as:

$$J_2 = 0.108262982131\times10^{-2}$$
$$J_4 = -0.237091120053\times10^{-5}$$
$$J_6 = 0.608346498882\times10^{-8}$$
$$J_8 = -0.142681087920\times10^{-10}$$
$$J_{10} = 0.121439275882\times10^{-13}$$

For the full set of equations, see Chapter 11 of [3] or reverse-engineer `F477.F`.

The gridded file mentioned above was populated with undulation values calculated using this SH method over the specified latitude and longitude grid interval (grid size 721×1441=1,038,961 points) covering 90°S to 90°N and 0°–360°E.

8.4 Accuracy

NASA, 1998 [3] discusses in detail the difficulty in assessing the accuracy of the various candidate models for the final version of EGM96, and several figures are quoted depending on varous scenarios and assumptions. Results are generally set against other (older) goid models such as JGM-2 and OSU91A and how EGM96 better fits satellite orbit arcs and by calculating formal errors. Results suggest EGM96 undulation RMS errors are typically of order 0.4–0.5 m (see Chapters 10 and 11 of [3]).

The accuracy of EGM96 against the ‘true’ geoid remains uncertain, and in any case, cannot be encompassed by a single figure; the accuracy varies significantly by geographic location and is a function of the underlying local (small-scale) gravitational anomaly variablilty, observation density and instrumental accuracy. The *zero*-th order term ζ_z alone is suggested to have a 1σ error of ±0.1 m (p.11–4).

Generally, it can be said that the EGM96 model undulation values are most accurate over the sea and continental USA and Europe, but even here large gravitational variations in the horizontal can reduce the overall accuracy, largely due to the limited (harmonic) resolution. Summarising the results, EGM96 can be expected to be accurate over most of the earth to around 0.5 m (1σ) or better, but certain areas may be less accurate, perhaps 1–2 m with worst-case errors up to 5 m.

The F77 program SH and GI methods “should be the same within 1 or 2 cm” (p.11–5), implying an acceptable 10% or so interpolation error.

9 Appendix 2: The EGM2008 model

9.1 Background

The abstract from Pavlis, 2012 ([9]) reads:

> EGM2008 is a spherical harmonic model of the Earth's gravitational potential, developed by a least squares combination of the ITG-GRACE03S gravitational model and its associated error covariance matrix, with the gravitational information obtained from a global set of area-mean free-air gravity anomalies defined on a 5 arc-minute equiangular grid. This grid was formed by merging terrestrial, altimetry-derived, and airborne gravity data. Over areas where only lower resolution gravity data were available, their spectral content was supplemented with gravitational information implied by the topography. EGM2008 is complete to degree and order 2159, and contains additional coefficients up to degree 2190 and order 2159.
>
> Over areas covered with high quality gravity data, the discrepancies between EGM2008 geoid undulations and independent GPS/Leveling values are on the order of $\pm$5 to $\pm$10 cm. EGM2008 vertical deflections over USA and Australia are within $\pm$1.1 to $\pm$1.3 arc-seconds of independent astrogeodetic values. These results indicate that EGM2008 performs comparably with contemporary detailed regional geoid models. EGM2008 performs equally well with other GRACE-based gravitational models in orbit computations.
>
> Over EGM96, EGM2008 represents improvement by a factor of six in resolution, and by factors of three to six in accuracy, depending on gravitational quantity and geographic area.
>
> EGM2008 represents a milestone and a new paradigm in global gravity field modeling, by demonstrating for the first time ever, that given accurate and detailed gravimetric data, a single global model may satisfy the requirements of a very wide range of applications.

9.2 Files

Full information on EGM2008 and related implementation in the form of Fortran code and data files can be found at: http://earth-info.nima.mil/GandG/wgs84/gravitymod/egm2008/egm08_wgs84.html.

As for EGM96, source code standalone programs are freely available: `hsynth_WGS84.f` for calculation using the spherical harmonic coeficients and `interp_1min.f` and `interp_2p5min.f` for interpolating a 1-minute or 2.5-minute grid, respectively[1].

The plain-text EGM2008 coefficients file (`EGM2008_to2190_TideFree`) is over 242 Mbytes[2], and the corrections file (`Zeta-to-N_to2160_egm2008`) is 142 Mbytes (364 Mbytes in total).

The gridded version is provided in two resolutions:

- $2.5' \times 2.5'$ (`Und_min1x1_egm2008_isw=82_WGS84_TideFree`, 149 Mbytes) or

[1] despite the file extention **`.f`**, the source code is actually Fortran 90

[2] filesizes on disk after decompression

- $1' \times 1'$ (`Und_min2.5x2.5_egm2008_isw=82_WGS84_TideFree`, 933 Mbytes)

Both are available in small- or big-endian internal binary format.

Apart from noting the above basic facts, the author of this report has done no practical assessment of EGM2008.

9.3 Accuracy

From the above abstract, and our assessment of EGM96 (Appendix 2), we might conclude that in general EGM2008 would be typically accurate to around 0.1 m and no worse than 1 m anywhere. This improvement over EGM96 is due in part to the much higher resolution as well as the significantly better quality and coverage of the observational data, particularly the use of recent satellite gravity missions.

9.4 Future?

While it seems that EGM96 is still quite adequate for current meteorological applications, were EGM2008 to be suggested as the new geoid standard for GNSS use in the future – for instance a recommendation by IROWG – and especially if it were to be implemented in ROPP in place of EGM96, then the significant additional resources in terms of disk space and run-time memory would first need to be factored in as trade-off with the higher horizontal resolution and accuracy. A similar review of EGM2008 to this report would then need to be undertaken before any change to ROPP could be considered.

10 Appendix 3: The ROPP SH code

10.1 History

The undulation calculation implemented in ROPP (**R-SH**) is a slightly modifed version of a port of the original NGA `F477.F` F77 code into F90 made some years ago by the present author. The code was adapted as a callable API rather than a stand-alone program.

While the underlying mathematical algorithm (see Appendix 1) is the same – though brought up to modern coding standards – the 'housekeeping' is much improved. For instance, finding the required pair of EGM96 data files is flexible instead of using fixed names. Rather than returning the undulation, the subroutine actually converts a given height (usually on the WGS-84 datum, but other datums are supported) to height above MSL, which is more often required. The undulation value can be recovered by subtracting the original ellipsoid height, if that is required.

The resulting 'HMSL' code was included within a larger F90 module containing many other Earth-coordinate related routines as `earth.f90`. At the time, the subroutine code was validated against the NGA code and spot-checked against the on-line checker ([5]), but this procedure was never separately documented – merely noted in the standard code comment headers.

The undulation calculation in ROPP is implemented in module `ROPP_UTILS` as source code file `coordinates/earth.f90`. Although some routines have been removed from the authors' original module[1], this file still contains several earth-coordinate transform routines; the relevant one here at user-level is `DATUM_HMSL()`. As noted, this subroutine is essentially a Fortran 90 port of the NGA program `F477.F`, so implementing the spherical harmonics calculation of undulation (see Appendix 1).

Since the F90 module was developed outside of (and well before) the ROM (GRAS) SAF context, this code has *Pre-Existing Software* (PES) status in which copyright and IPR remain with the Met Office[2][3].

In this report, we refer to the ROPP implementation as **R-SH**.

10.2 Call signature

The signature of this subroutine is:

```
USE EarthMod
DOUBLE PRECISION :: GEO(3), HMSL
CHARACTER (LEN=200) :: CoefFile, CorrFile
GEO(:) = (/ <lat>, <lon>, <ht> /) ! on WGS-84 ellipsoid
CoefFile = <path-to-egm96-coefficients-file>  ! or blank
CorrFile = <path-to-egm96-corrections-file>   ! or blank
CALL Datum_HMSL("WGS84", GEO, HMSL, Fcoeff=CoefFile, Fcorr=CorrFile)
```

[1] *inter alia* those relating to the UK National Grid

[2] Strictly, Crown Copyright

[3] Before any lawyers start jumping, the orginal codebase has been virtually re-written and (as far as we know) the underlying algorithms are not under patent or any licence terms, so while the original source is gratefully acknowledged, the F90 code used in ROPP is ours.

10.3 Files

The two file arguments `Fcoeff` and `Fcorr` are optional or their values may be blank strings; their default location may be defined external to the application program via envionment variables; for instance at Bash shell level:

```
export GEOPOT_COEF=<path-to-egm96-coefficients-file>
export GEOPOT_CORR=<path-to-egm96-corrections-file>
```

If these are not defined, the routine will look in various places within the current directory tree for files `egm96.dat` and `corrcoef.dat`. If the file arguments are present, the actual files found will be returned to the caller, so the actual arguments must not be 'read-only' or literal strings.

In all cases, the 'final' file names could even be soft-links to the actual files themselves.

10.4 Algorithm

1. On first invocation, the routine first attempts to locate the two coefficients and corrections files – as described above – and loads the coeffcients from the first found files. The actual file names used are passed back to the caller via the argument list if these were present (Tip: the arguments' values may be set to blank strings in order to use the internal defaults and on exit contain the found file names.)
2. If the file(s) cannot be found or the coeffcients cannot be loaded, this status is flagged via the returned geoid height argument having a value of `-9999999.0d0`. Subsequent calls will also return this value. It is recommended that the routine be called once initially in order to pre-load the coefficients and to test that this was sucessful before actual calculations are performed. For instance:

```
CALL Datum_HMSL("WGS84", (/0.0d0, 0.0d0, 0.0d0/), hmsl[, Fcoeff=..., Fcorr=...])
IF ( hmsl < -999.0d0 ) THEN
  PRINT *, "ERROR: EGM96 file(s) not loaded!"
  CALL EXIT(1)
END IF
```

3. Thereafter, the file argument are ignored, so simply call as:

```
CALL Datum_HMSL("WGS84", (/wgslat, wgslon, wgsht/), hmsl)
```

4. If the datum (argument #1) is supported and is not `WGS84`, the given LLH values are transformed to WGS-84.
5. The SH undulation calculation is performed as per Appendix 1 using the WGS-84 latitude and longitude values.
6. The H_g value is calculated by subtracting the undulation value from the WGS-84 height and returns it to the caller via the argument `hmsl`. Not that the undulation value itself is not directly returned.

Figure 2.1 was generated by calling `Datum_HMSL()` from an F90 wrapper program in a double-loop to generate a grid of values running in N-S rows of E-W columns with a grid spacing of 0.25° in each direction. The grid values were saved in a netCDF file which was visualised using IDL.

10.5 EGM96 data files in ROPP

As used in ROPP, the same two coefficents and corrections files as described in Appendix 1 are used, but they are packaged in `ropp_pp/data` as GZIP-ed files `egm96.dat.gz` and `corrcoef.dat.gz`; the ROPP installer uncompresses these files *in situ* to `egm96.dat` and `corrcoef.dat`. Apart from the compression method and names, the content and formatting of both files are identical to the originals.

The ROPP preprocessor tools find these files via their full paths being specified in the tools' configuration files, as detected within the common routine `ropp_pp_read_config()`. Internally, this routine defaults to the above (uncompressed) file names in the current working directory. Hence the file names from the configuration file or default names are explicitly passed to the `DATUM_HMSL()` routine and any environment variable settings (see Appendix 3) are ignored.

10.6 Resources

The plain-text EGM96 gravitational anomaly spherical harmonics coefficients file (`egm96.dat`) and the companion height corrections file (`corrcoef.dat`) are each 5.3 Mbytes[4] or 10.6 Mbytes in total.

The run-time memory requirement is dominated by the coefficients arrays (five arrays each of $(361 \times 362) / 2 = 65{,}341$) double-precision (64-bit) elements plus three pre-computed sine and cosine arrays and a Legendre Function evaluation array, each of 361 d.p. elements. This totals roughly 2.7 Mbytes – challenging for computers when EGM96 was developed but rather modest now, even for a modern smartphone!

[4] filesizes on disk after decompression

11 Appendix 4: The GWV GI code

11.1 History

While we could have ported the NGA GI (N-GI) code implemented in F77 program `INITPT.F` to an equivalent F90 subroutine (or function), was is desirable to be able to calculate a valid undulation everywhere, including close to the poles[1] which the N-GI (spline) algorthm cannot do.

Further, the NGA spline interpolation method is considered to be more complex than is necessary, and a simpler (and even quicker) algorithm might be implemented, while still meeting the accuracy requirement for GNSS applications. Finally, it was desired to implement a version in pure Python, and the simpler the algorithm the better! The simplest N-GI bilinear algorithm was clearly not accurate enough, so higher order interpolations were prototyped. The resulting optimal bicubic interpolation algorithm is described below.

In this report, we refer to the GWV implimentation as **G-GI**.

11.2 Call signature

The undulation calculation in the GWV package for E-GVAP is implemented as source code file `egm96.f90` containing module `EGM96`. This module currently contains a single user-level routine `EGM96_Undulation()`. The signature of this function is:

```
USE EGM96
DOUBLE PRECISION :: wgslat, wgslon, Und
CHARACTER (LEN=200) :: GridFile
GridFile = <path-to-egm96-grid-file>  ! or blank
Und = EGM96_Undulation (wgslat, wgslon, File=GridFile)
```

The function API will also accept single-precision lat/lon arguments and will then return a single-precision undulation value.

11.3 Files

The file argument `File` is optional or its value may be a blank string; the default location may be defined external to the application program via an envionment variable; for instance at Bash shell level:

```
export EGM96_GRID=<path-to-egm96-grid-file>
```

If this is not defined, the routine will look for file `egm96_grid.dat` in various places such as `~/data/egm96`, the current working directory and `../data`. If the file argument is present, the actual file found will be returned to the caller, so the actual argument must not be ‘read-only’ or a literal string.

In all cases, the ‘final’ file name could even be a soft-link to the actual file itself.

[1] while being exceedingly rare very close to or at the poles, the GPS–LEO geometry does not make it impossible for RO profiles to occur here. Also, it is quite likely that at some point, near-real time data from a permanent ground-based GNSS receiver sited at the US Amundsen-Scott South Pole Station will become available.

11.4 Algorithm

1. On first invocation, the routine first attempts to locate the grid file – as described above – and loads the undulation values from the first found file. The actual file name used is passed back to the caller via the argument list if this was present (Tip: the argument's value may be set to a blank string in order to use the internal default and on exit contain the found file name.)

2. If the file cannot be found or the grid cannot be loaded, this status is flagged via the returned undulation variable having a value of `-999.0`. Subsequent calls will also return this value. It is recommended that the routine be called once initially in order to pre-load the grid and to test that this was sucessful before actual calculations are performed. For instance:

```
Undu = EGM96_Undulation(0.0, 0.0, File=GridFile) ! dummy call to load grid
IF ( undu < -900.0 ) THEN
  IF ( GridFile /= " " ) THEN
     PRINT *, "Error loading EGM96 grid from file "//TRIM(GridFile)
  ELSE
     PRINT *, "Failed to find EGM6 grid file"
  ENDIF
ELSE
 ... get undulations with real lat/lon locations...
ENDIF
```

3. Additional 'halo' rows and columns are created around the grid edge. The 'over-the-pole' rows are merely copies of the pole row values; the longitude columns wrap across 0° (360°).

4. Thereafter, the file argument are ignored, so simply call as:

```
Undu = EGM96_Undulation(wgslat, wgslon)
```

5. The G-GI undulation calculation is performed using a bicubic interpolation from the pre-computed grid points to the required WGS-84 latitude and longitude.

   a) the lat/lon is normalised to the grid index and the nearest 4×4 grid points extracted;

   b) for each of the four 'y' (latitude) rows, the four 'x' (longitude) column points are fitted to an exact cubic; with u_x being the grid undulation value at the relative grid point x and $x = 0$ the nearest 'floor' grid point:

$$b = (u_{-1} - 2u_0 + u_1)/2$$

$$a = (u_2/2 - u_1 + u_0/2 - b)/3$$

$$c = u_1 - u_0 - a - b$$

$$d = u_0$$

   and the undulation row-wise interpolated as: $U_y = ax^3 + bx^2 + cx + d$.

   c) and finally, the four interpolated U_y column points are similarly fitted and interpolated in the row direction to yield the required $U(x,y)$.

If the user requires the geoid height, H_g, the value can be calculated simply by subtracting the undulation value from the WGS-84 height (Equation 1.1).

11.5 Python

The author has also developed a Python-based importable module `egm96.py` providing a class `EGM96` and method `undulation()` which employs the same file-finding and bicubic interpolation algorithms as in the above Fortran routine. This version is used as part of a Python-based GNSS meta-data database system for E-GVAP GNSS stations.

The Python usage is similar to the Fortran version, except that instatiating the `EGM96` class pre-loads the grid file which can be interpolated with the `undulation()` method. For example:

```
import egm96
egm = egm96.EGM96([file=...])
if egm.loaded:
    undu = egm.undulation(lat, lon)
else
    if egm.gridfile is None:
        print 'EGM96 grid file not found'
    else
        print 'EGM96 grid file', gridfile, not loaded'
```

The grid file default search is identical to the Fortran one noted above. If the grid file failed to load but the status is not checked, the returned undulation value will be `-999.9`.

11.6 Status

Please note that the G-GI code described here is not (at the time of writing) part of ROPP and is subject to modification by the author at any time. If incorporated into ROPP, the code will be included in the ROPP distribution with pre-existing software (PES) status and kept under the usual ROPP code versioning system.

Bibliography

[1] Culverwell I.C. (2013). A review of the geodesy calculations in ROPP. ROM SAF Report 14, SAF/ROM/METO/REP/GSR/014, 2013.

[2] F. G. Lemoine, D. E. Smith, L. Kunz, R. Smith, E. C. Pavlis, N. K. Pavlis, S. M. Klosko, D. S. Chinn, M. H. Torrence, R. G. Williamson, C. M. Cox, K. E. Rachlin, Y. M Wang, S. C. Kenyon, R. Salman, R. Trimmer, R. H. Rapp, and R. S. Nerem (1997). The Development of the NASA GSFC and NIMA Joint Geopotential Model. Gravity, Geoid and Marine Geodesy, Vol. 117, International Association of Geodesy Symposia, J. Segawa, H. Fujimoto, and S. Okubo (editors), pp 461-469, 1997. Copy available from:
ftp://cddis.gsfc.nasa.gov/pub/egm96/general_info/tokyo_grageomar96_paper_web.ps
(visited October 2013).

[3] F. G. Lemoine, S. C. Kenyon, J. K. Factor, R.G. Trimmer, N. K. Pavlis, D. S. Chinn, C. M. Cox, S. M. Klosko, S. B. Luthcke, M. H. Torrence, Y. M. Wang, R. G. Williamson, E. C. Pavlis, R. H. Rapp and T. R. Olson (1998). The Development of the Joint NASA GSFC and NIMA Geopotential Model EGM96. NASA Goddard Space Flight Center, Greenbelt, Maryland, 20771 USA, July 1998. NASA/TP-1998-206861. Available from:
http://cddis.gsfc.nasa.gov/926/egm96/nasatm.html
dated January 2006 (visited October 2013).

[4] NASA (2004). The NASA GSFC and NIMA Joint Geopotential Model.
http://cddis.gsfc.nasa.gov/926/egm96/egm96.html
dated November 2004 (visited October 2013).

[5] NGA (2013a). The NGA on-line EGM96 calculator.
http://earth-info.nga.mil/GandG/wgs84/gravitymod/egm96/intpt.html.

[6] NGA (2013b). The EGM96 height (grid) and coefficients/corrections, Fortran-77 source code and other gravity model files (e.g. for EGM2008) can be downloaded from:
http://earth-info.nga.mil/GandG/wgs84/gravitymod/index.html.

[7] NGA (2013c). World Geodetic System 1984 (WGS 84).
http://earth-info.nga.mil/GandG/wgs84/index.html
dated April 2013 (visited October 2013).

[8] NIMA (1997). Department of Defense World Geodetic System 1984, Its Definition and Relationships With Local Geodetic Systems NIMA Technical Report TR8350.2, Third Edition, 4 July 1997. Copy available in PDF from:
http://earth-info.nga.mil/GandG/publications/tr8350.2/tr8350_2.html
(visited October 2013).

[9] Pavlis, NK, Holmes, SA, Kenyon, SC and Factor, JK (2012). The development and evaluation of the Earth Gravitational Model 2008 (EGM2008). Journal of Geophysical Research: Solid Earth (1978-2012) Volume 117, Issue B4, April 2012. Article first published online: 19 April 2012. DOI: 10.1029/2011JB008916

[10] Stoer, Joseph (1972). Fast one-dimensional equidistant spline interpolation function.
Einfuhrung in die Numerische Mathematik I, Springer 1972, p81. C

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/METO/REP/RSR/018	Single Frequency Radio Occultation Retrievals: Impact on Numerical Weather Prediction
SAF/ROM/METO/REP/RSR/019	Implementation of the ROPP two-dimensional bending angle observation operator in an NWP system
SAF/ROM/METO/REP/RSR/020	Interpolation artefact in ECMWF monthly standard deviation plots
SAF/ROM/METO/REP/RSR/021	5th ROM SAF User Workshop on Applications of GPS radio occultation measurements
SAF/ROM/METO/REP/RSR/022	The use of the GPS radio occultation reflection flag for NWP applications

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