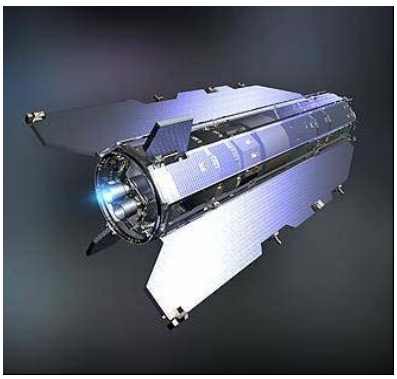
Universität Stuttgart

Geodätisches Institut

GOCE sensitivity studies in terms of cross-over analysis



Diplomarbeit im Studiengang

Geodäsie und Geoinformatik

an der Universität Stuttgart

Yang Xue

Stuttgart, September 2009

Betreuer: Dr.-Ing. Oliver Baur
Universität Stuttgart

Prof. Dr.-Ing. Nico Sneeuw
Universität Stuttgart

Selbständigkeitserklärung

Hiermit erkläre ich, Yang Xue, dass ich die von mir eingereichte Diplomarbeit zum Thema

GOCE sensitivity studies in terms of cross-over analysis

selbständig verfasst und ausschließlich die angegebenen Hilfsmittel und Quellen verwendet habe.

Ort, Datum Unterschrift

Acknowledgements

Here I would like to express my sincere appreciation to my supervisors Prof. Dr.-Ing. Nico Sneeuw and Dr.-Ing. Oliver Baur for offering this interesting topic, great supervision, patience, insightful discussions and comments.

I give my special thanks to Dr.-Ing. Matthias Weigelt for his very helpful advice, to Prof. Dr. Clark R. Wilson and Prof. Dr. Zong-liang Yang for the generous support at the University of Texas at Austin, to my parents and friends for encouraging and supporting me at every of my difficult times.

Abstract

The GOCE (Gravity field and steady-state Ocean Circulation Explorer) satellite, launched on 17 March 2009, for the first time applies satellite gravity gradiometry (SGG) to recover the Earth's gravity field with cm accuracy at a resolution of 100 km. To meet the envisaged accuracy, measurement validation at cross-over points (XOs) is necessary. Typically, validation is based on gravity gradients (GGs). However, the coefficient matrix of the gravitational tensor is dependent on orientation. In order to avoid matrix rotation, analysis based on orientation-independent invariants is possible. By applying various noise models, the goodness of XO-validation based on GGs and invariants will be studied in this thesis.

First, by determining the maximum of scalar products from two tracks, the XOs can be predicted. Next, using local polynomial approximation, the geographical coordinates of XOs are calculated by solving a system of equations. Due to the orbit drift, the interpolation of height is performed separately along ascending and descending track before final comparison.

Considering a sampling rate of 1 Hz, GGs and invariants in all points of a one-week orbit are simulated for the further interpolation at the XOs. To determine the goodness of the selected interpolation algorithm, a closed loop test with noise-free data is investigated first. Since signal to noise ratios of GGs and invariants are all above 70 dB, the same algorithm is applied in closed loop tests with noisy data.

Since GOCE can only provide high accuracy for the tensor components of $V_{\lambda\lambda}, V_{\phi\phi}, V_{rr}$ and $V_{\lambda r}$, various noise models, i.e. homogeneous and inhomogeneous white noise, as well as homogeneous and inhomogeneous colored noise, are added to the simulated values. The comparison of the goodness of GGs opposed to invariants is based on the signal to noise ratio (SNR). In this study, the second invariant demonstrates better SNR than GGs and the third invariant in the case of homogenous noise. However, due to the impact of the inaccurate GGs ($V_{\lambda\phi}, V_{\phi r}$), the SNR of invariants is poorer than the SNR of all GGs' in the case of inhomogeneous noise.

Keywords

Satellite gravity gradiometry, cross-over location, closed-loop test, invariants, GOCE

Contents

1 Introduction **1**

2 Simulation of GOCE orbit **3**
2.1 GOCE mission . . . 3
2.2 Orbit in the Earth-fixed system . . . 4
2.3 Saving structure . . . 4

3 Location of cross-over points **7**
3.1 Cross-over condition . . . 7
3.2 Data preparation . . . 8
3.3 XO prediction . . . 8
3.4 2D location of the actual XO . . . 9
3.5 Interpolation of height . . . 11

4 Simulation of gravity gradients and invariants **15**
4.1 Gravity gradients in the model system . . . 15
4.2 Two-step synthesis . . . 16
4.3 Gravity gradients in the Earth-fixed frame . . . 17
4.4 Gravity gradients in the local orbit reference frame . . . 19
4.5 Simulation of invariants . . . 19

5 Closed-loop test in the Earth-fixed frame **23**
5.1 Interpolation in the EFS . . . 23
5.2 Height reduction in EFS . . . 23
5.3 Residuals in EFS . . . 25

6 Closed-loop test in the local orbit reference frame (LORF) **29**

7 XO test with noisy data **33**
7.1 Test with white noise . . . 33
7.1.1 Test with homogeneous white noise . . . 34
7.1.2 Test with inhomogeneous white noise . . . 35
7.2 Test with homogeneous colored noise . . . 37
7.3 Test with inhomogeneous colored noise . . . 39

8 Summary and outlook **41**

List of Figures

1.1 GOCE mission (credits: ESA) 1

2.1 Ground track of the GOCE orbit (one week with sampling rate of 1Hz) 5

3.1 XO deviation caused by discrete sampling 8
3.2 Trajectory in near North Pole region 10
3.3 Deviation of cross-over location 11
3.4 Cross-over points within the one-week orbit simulation 12
3.5 Height difference between ascending and descending track at the same XO 13
3.6 Difference of height reduction between cubic interpolation and spline interpolation . . . 13

4.1 Absolute gravity gradients in the model system 17
4.2 Absolute gravity gradients in the Earth-fixed system 18
4.3 GGs in the Earth-fixed system reduced by GRS80 18
4.4 Absolute gravity gradients in the LORF 20
4.5 Gravity gradients in the LORF reduced by GRS80 20
4.6 Invariants reduced by GRS80 21

5.1 Spline and cubic interpolation of δV_{rr} 24
5.2 Spline and cubic interpolation of δI_2 24
5.3 Residuals of reduced GGs from cubic interpolation in the EFS 25
5.4 Residuals of reduced invariants from cubic interpolation in the EFS 26
5.5 Residuals of reduced GGs from spline interpolation in the EFS 26
5.6 Residuals of reduced invariants from spline interpolation in the EFS 27
5.7 Magnitude of GG's residuals ($\log_{10} dV_{rr}$) derived from spline interpolation in the EFS . . 27
5.8 Magnitude of invariant's residuals ($\log_{10} dI_3$) derived from spline interpolation in the EFS 28

6.1 The SNR of GGs from spline interpolation in the LORF with noise-free data 30
6.2 The SNR of invariants from spline interpolation in the LORF with noise-free data 30

7.1 Time series (left) and periodogram (right) of simulated homogeneous white noise with $\sigma = 6\,mE$ 33
7.2 The SNR of GGs in the the LORF with homogenous white noise 34
7.3 The SNR of invariants in the LORF with homogenous white noise 35
7.4 The SNR of GGs in the LORF with inhomogeneous white noise 36
7.5 The SNR of invariants in the LORF with inhomogeneous white noise 36
7.6 Times series and Periodograms of colored noise for diagonal GGs 37
7.7 The SNR of GGs in the LORF with homogeneous colored noise 38
7.8 The SNR of invariants in the LORF with homogeneous colored noise 38
7.9 The SNR of GGs in the LORF with inhomogeneous colored noise 39
7.10 The SNR of invariants in the LORF with inhomogeneous colored noise 40

8.1 Stabilities of GGs and invariants with respect to different types of noises 42

List of Tables

2.1 Initial values of Kepler elements for GOCE orbit simulation 3

5.1 Residuals from cubic and spline interpolation in the EFS 28

6.1 The SNR of GGs and invariants from spline interpolation in the LORF in closed loop test 31

7.1 The SNR of GGs and invariants in the LORF with homogenous white noise 34
7.2 Statistics of the simulated inhomogeneous white noise 35
7.3 The SNR of GGs and invariants in the LORF with inhomogeneous white noise 35
7.4 Statistics of the simulated homogeneous colored noise 37
7.5 The SNR of GGs and invariants in the LORF with homogeneous colored noise 39
7.6 Statistics of the simulated inhomogeneous colored noise 39
7.7 The SNR of GGs and invariants in the LORF with inhomogeneous colored noise 40

Chapter 1

Introduction

The GOCE (Gravity field and steady-state Ocean Circulation Explorer) satellite mission enables to model the geoid down to spatial wavelengths of 100 km with an accuracy of 1-2 cm. In order to meet the objectives for GOCE, validation plays an essential role. Therefore, GOCE sensitivity in terms of cross-over (XO) analysis will be studied in this thesis, which compares measurements in the same geographical positions, or the so called satellite ground track cross-overs. Due to the height differences at the same geographical positions caused by orbit eccentricity (about 0.001), height reduction using a consistent gravity field has to be applied. Therefore, the residuals after comparison indicate interpolation errors in the case of noise-free validation within a closed-loop test.

As it is well known that only within the measurement bandwidth (MBW) ranging from 5 mHz to 0.1 Hz high quality results can be provided by the gradiometer instrument; the accuracy of terrestrial gravity field recovery in the long-wavelength part is reduced by the colored noise characteristic. This study investigates both gravitational gradients (GGs) and gravitational tensor invariants. The latter due to their advantage of independency on orientation, i.e. they avoid errors caused by frame rotations.

After the location of XOs (chapter 3), studies of GGs and invariants at XOs in the closed loop test are investigated. Due to the same gravity field used for GOCE orbit generation (chapter 2), simulation of GGs and invariants (chapter 4) and height reduction (chapter 5), the residuals from the closed loop test are mainly caused by interpolation, including height interpolation (chapter 3) and interpolation of GGs and invariants (chapters 5 and 6). Finally, various observation noises are generated and adopted on both simulated GGs and invariants to derive their stochastic properties (chapter 7).

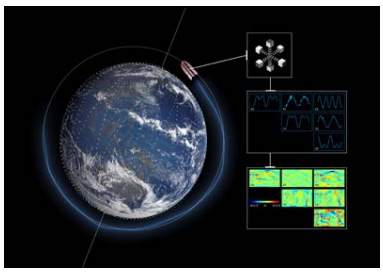


Figure 1.1: *GOCE mission (credits: ESA)*

Chapter 2

Simulation of GOCE orbit

This chapter presents a brief description of the simulated GOCE orbit, including transformation of coordinates into the Earth-fixed system and the saving structure of the coordinates.

2.1 GOCE mission

GOCE, launched on March 17, 2009, for the first time applies satellite gravity gradiometry (SGG). For this purpose, ultra-sensitive accelerometers are arranged pairwise in three dimensions, equidistant from the gradiometer center (figure 1.1). Different location leads to different gravitational acceleration. By measuring differential accelerations, i.e. second derivatives of Earth's gravitational potential, the Earth's gravity field can be recovered with high homogeneous accuracy. Second derivatives are denoted as gravity gradients (GGs). GGs are assembled in the gravitational tensor

$$\underline{\underline{V}} \quad = \quad \nabla(\nabla V(\mathbf{x}(\lambda,\phi,r))) = \mathbf{e}_i \otimes \mathbf{e}_j V_{ij}; \quad i,j = 1,2,3 \tag{2.1}$$

with V: gravitational potential, $\underline{\underline{V}}$: gravitational tensor, and V_{ij}: gravitational gradients.

The GOCE satellite is flying on a sun-synchronous orbit for at least 24 months, about 250 km above the surface of the Earth. The low altitude allows detecting sensitive gravitational signals. The initial values (Kepler elements) for GOCE orbit simulation are listed in Table 2.1.

Kepler element	Symbol	Initial value
Semi-major axis of orbit	a	6628 000 m
Eccentricity of orbit	e	0.001
Inclination of orbit	I	96.6°
Right ascension of the ascending node	Ω	0°
Argument of perigee	ω	0°
Mean anomaly	M	0°

Table 2.1: *Initial values of Kepler elements for GOCE orbit simulation*

This thesis is based on the analysis of simulated measurements of one week with the sampling rate of 1 Hz.

2.2 Orbit in the Earth-fixed system

The study is based on a simulated GOCE orbit generated with the program SOSP (Goetzelmann 2003). The EGM96 gravity field model (Lemoine 1998) up to degree and order 200 is adopted. The integrator provides the orbit coordinates in the inertial system. The cartesian coordinates have to be transformed to geographical coordinates with respect to the Earth-fixed system for XOs' location.

The first step is to transform the cartesian coordinates from the inertial system to the Earth-fixed system around the third coordinate axis with angle θ_{Gr} (McCarthy 1996):

$$\mathbf{x}(t) \;=\; \underline{\mathbf{R}}_3(\theta_{\mathrm{Gr}}(t)) \cdot \mathbf{X}(t) = \begin{pmatrix} \cos(\theta_{\mathrm{Gr}}(t)) & \sin(\theta_{\mathrm{Gr}}(t)) & 0 \\ -\sin(\theta_{\mathrm{Gr}}(t)) & \cos(\theta_{\mathrm{Gr}}(t)) & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \mathbf{X}(\mathbf{t}) \tag{2.2}$$

with $\mathbf{x_t}$: cartesian coordinates in the Earth-fixed system, $\mathbf{X_t}$: cartesian coordinates in inertial system and $\theta_{\mathrm{Gr}}(t)$: Greenwich Sidereal Time (GST). According to Escobal (1975), GST can be calculated with respect of $\theta_{\mathrm{Gr}}(0)$ at the time of $t_0 = 0.00\mathrm{h\,UT}$:

$$\theta_{\mathrm{Gr}}(t) = \theta_{\mathrm{Gr},0} + 0.25068447[^\circ/\mathrm{min}] \cdot (t - t_0) \tag{2.3}$$

$$\theta_{\mathrm{Gr},0} = 99.6909833^\circ + 36000.7689^\circ \cdot T_{\mathrm{u}} + 0.00038708^\circ \cdot T_{\mathrm{u}}^2 \tag{2.4}$$

$$T_u = (\mathrm{JD} - 2415020.0)/36525. \tag{2.5}$$

The next step is to transform the cartesian coordinates in geographical coordinates, or projection of the nominal ground track:

$$\mathbf{x}(\lambda, \phi, r) = \mathbf{e}_1 r \cos\phi \cos\lambda + \mathbf{e}_2 r \cos\phi \sin\lambda + \mathbf{e}_3 r \sin\phi \tag{2.6}$$

$$= \mathbf{e}_1 r \sin\theta \cos\lambda + \mathbf{e}_2 r \sin\theta \sin\lambda + \mathbf{e}_3 r \cos\theta \tag{2.7}$$

The result (ground track of the GOCE orbit) is presented in figure 2.1. The intersections of ascending and descending tracks indicate XO.

2.3 Saving structure

Descending points and ascending points can be separated by the sign of latitude difference between neighbor points. That means, $\mathrm{d}\theta(t_i) > 0$ for ascending points and $\mathrm{d}\theta(t_i) < 0$ for descending points with $\mathrm{d}\theta(t_i) = \theta(t_{i+1}) - \theta(t_i)$.

Then the same type of points can be ordered track by track by detecting the gap of their positions in the entire track. That means the position of the same type of points in the same track should be continuous.

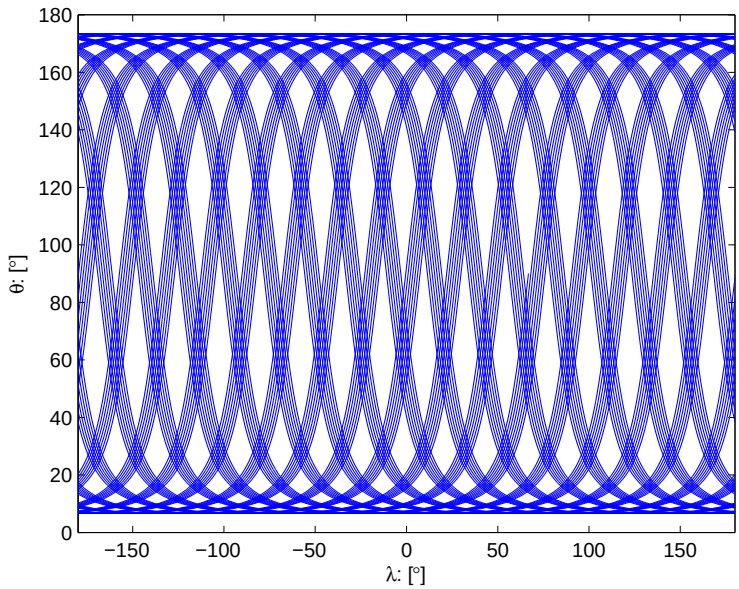


Figure 2.1: *Ground track of the GOCE orbit (one week with sampling rate of 1Hz)*

Since both type and track of points have been identified, their coordinates can be saved column-wise in a matrix. The same column indicates the same ascending and descending track, respectively. For example, the matrix for θ-values of ascending tracks becomes

$$\mathbf{T}_a = \begin{pmatrix} \theta_{11} & \theta_{21} & \dots & \theta_{n1} \\ \theta_{12} & \theta_{22} & \dots & \theta_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ \theta_{1m} & \theta_{2m} & \dots & \theta_{nm} \end{pmatrix}. \tag{2.8}$$

Matrix $\mathbf{T}_a$ consists of n ascending track with m points per track. Since different tracks could have different number of points, the maximal points-number is chosen as m, blanks are filled with `NaN` for convenient plotting and manipulation of the matrix.

Chapter 3

Location of cross-over points

A two step procedure for the location of cross-over points is investigated. The first step is to find the true points near the XO, or the so called analytical XO prediction. The purpose of it is to check the intersection-possibility of two tracks and to find the region containing the XO. The next step is the determination of XO by solving a system of equation based on the regional polynomial approximation.

3.1 Cross-over condition

An analytical condition for the intersection of two trajectories can be defined (Kim 1997),

$$\mathbf{x}(t_1) = \mathbf{x}(t_2) \tag{3.1}$$

$$\cos\psi(t_1,t_2) = \mathbf{x}(t_1)\cdot\mathbf{x}(t_2) = x(t_1)\cdot x(t_2) + y(t_1)\cdot y(t_2) + z(t_1)\cdot z(t_2) = 1. \tag{3.2}$$

Therein, $\mathbf{x}(t_i)$ denotes the geocentric unit vector pointing to a simulated data point at time tag t_i.

The reason for normalization of the geocentric vector $\mathbf{x}(t_i)$ is the inconsistency of radius caused by orbit drift. Specifically, different altitudes at the same geographical position are expected from the changing eccentricity of the GOCE orbit. Therefore, in this study XO demonstrates a pair of vectors - one from ascending track and one from descending track - sharing the same geographical coordinates (λ,ϕ). Their height difference will be calculated in section 3.5 and considered in the final comparison in section 5.2.

The crossing condition (Eq. 3.1) is used for both checking the intersection possibility by comparing the largest direction cosine $\cos\psi(t_1,t_2)$ with same threshold value and the location of two vectors with smallest spherical angle by finding the maximum direction cosine of two tracks. The reason that not the actual pair of XOs, but the pair with smallest spherical angle could be found with this condition is discretization, i.e. sampling with 1 Hz. Even the largest direction cosine of two tracks with intersection possibility could have slight deviation from 1. The situation indicating slight deviation is illustrated in figure 3.1, its maximal radial deviation Δr can be estimated with the mean flight velocity $\bar{v} \approx 7755\,\mathrm{m\,s^{-1}}$ in the Earth-fixed system to $\Delta r \approx \Delta t \cdot \frac{\sqrt{2}}{2} \cdot \bar{v}$. So the slight deviation between largest direction cosine and 1 can be estimated to

$$\frac{\Delta r}{R} = \frac{\sqrt{2}}{2}\,\mathrm{s} \cdot \frac{7755\,\mathrm{m\,s^{-1}}}{6370\,\mathrm{km}} \approx 0.00086. \tag{3.3}$$

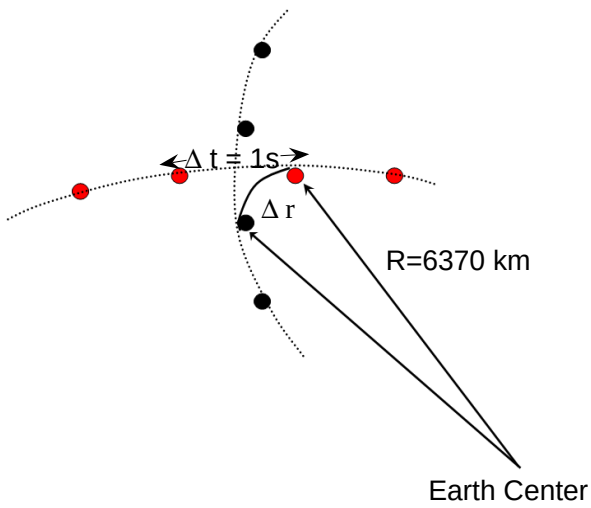


Figure 3.1: *XO deviation caused by discrete sampling*

The procedures of efficient calculation of every direction cosine between ascending and descending track will be demonstrated in the following sections.

3.2 Data preparation

In chapter 2, the coordinates of the simulated data points have been saved column-wise in matrices for ascending and descending tracks. Therefore, if XO_{ij} - XO from i-th ascending track and j-th descending track - is to be predicted, their corresponding coordinates can be found immediately in i-th column of "the ascending matrix" and j-th column of "the descending matrix".

3.3 XO prediction

In order to predict XOs with condition 3.1, all spherical angles between $\mathbf{x}(t_i)$ and $\mathbf{x}(t_j)$ are calculated. The largest direction cosine indicates two points with the smallest spherical angle or nearest the actual XO.

Since the crossing condition is based on geocentric unit vectors, the first step is its normalization,

$$\bar{\mathbf{x}} = (\bar{x}, \bar{y}, \bar{z}) = (x, y, z)/\sqrt{x^2 + y^2 + z^2}. \tag{3.4}$$

The second step is to calculate the spherical angle between each data point from one ascending track and one descending track. The coordinates from one ascending track with m_a points and one descending track with m_d points are saved in matrices $\mathbf{A}$ ($m_a \times 3$) and $\mathbf{D}$ ($m_d \times 3$), respectively:

$$\mathbf{A} = \begin{pmatrix} x(t_1) & y(t_1) & z(t_1) \\ x(t_2) & y(t_2) & z(t_2) \\ \vdots & \vdots & \vdots \\ x(t_{m_a}) & y(t_m) & z(t_{m_a}) \end{pmatrix} \quad \mathbf{D}^{\mathrm{T}} = \begin{pmatrix} x(t_1) & x(t_2) & \dots & x(t_{m_d}) \\ y(t_1) & y(t_2) & \dots & y(t_{m_d}) \\ z(t_1) & z(t_2) & \dots & z(t_{m_d}) \end{pmatrix} \tag{3.5}$$

$$\mathbf{A} \cdot \mathbf{D}^{\mathrm{T}} = \begin{pmatrix} \cos\psi(\mathbf{x}(t_1),\mathbf{x}(t_1)) & \cos\psi(\mathbf{x}(t_1),\mathbf{x}(t_2)) & \dots & \cos\psi(\mathbf{x}(t_1),\mathbf{x}(t_n)) \\ \cos\psi(\mathbf{x}(t_2),\mathbf{x}(t_1)) & \cos\psi(\mathbf{x}(t_2),\mathbf{x}(t_2)) & \dots & \cos\psi(\mathbf{x}(t_2),\mathbf{x}(t_n)) \\ \vdots & \vdots & \ddots & \vdots \\ \cos\psi(\mathbf{x}(t_m),\mathbf{x}(t_1)) & \cos\psi(\mathbf{x}(t_m),\mathbf{x}(t_2)) & \dots & \cos\psi(\mathbf{x}(t_m),\mathbf{x}(t_n)) \end{pmatrix} \tag{3.6}$$

By multiplication of matrix $\mathbf{A}$ and $\mathbf{D}^{\mathrm{T}}$, all spherical angle ψ are calculated at once. Furthermore, the saving-order of the results, e.g. component in p-th row and q-th column of the resulting matrix, indicates the spherical angle between vector at time t_p from the ascending track, denoted as p_a, and vector at time of t_q from the descending track, denoted as q_d.

After calculation of all spherical angles, a check of the intersection-possibility can be carried out by comparing the maximum direction cosine with the threshold value of $1 - \frac{\Delta r}{R} \approx 0.99914$ (Eq. 3.3). Intersection-possibility is denied in case of $\cos\psi < 0.99914$.

Once the intersection-possibility is evaluated, two XO candidates p_a and q_d with smallest spherical angle can be identified. Then two or six points near each candidate are selected for the regional polynomial approximation (two points for linear approximation and six points for cubic approximation).

3.4 2D location of the actual XO

As mentioned in section 3.1, a height difference from a pair of XOs is expected in 3D. Therefore, a well defined projection surface has to be defined before the determination the common geographical coordinates, which has already been done in chapter 2: conversion into nominal ground track, presented with the parameters λ, ϕ.

Regional polynomial approximation based on the selected points near p_a and q_d enables the determination of actual XO by solving a system of equations built from the ascending and descending tracks. The polynomial coefficients, which describe the dependency of θ on λ are derived with the Matlab function "`polyfit`".

The main problem is to define a suitable degree for polynomial approximation. Figure 2.1 indicates that the linear approach can only be used at low and middle latitudes, ranging from $40° < \theta < 140°$.

$$\theta = a_1\lambda + a_0 \tag{3.7}$$

$$\theta = d_1\lambda + d_0 \tag{3.8}$$

At higher latitudes, a polynomial with higher degree has to be applied. The empirical value of degree "3" is chosen here. Validation of the polynomial degree can be found below.

$$\theta = a_3\lambda^3 + a_2\lambda^2 + a_1\lambda + a_0, \quad \text{for the ascending track;} \tag{3.9}$$

$$\theta = d_3\lambda^3 + d_2\lambda^2 + d_1\lambda + d_0, \quad \text{for the descending track;} \tag{3.10}$$

Three solutions are expected from a third-order polynomial. But only solutions in the region of $\theta \in [0°, 180°]$ and $\lambda \in [-180°, 180°]$ make sense. Furthermore, tracks in polar regions with $\theta \in [170°, 180°]$ and $\theta \in [0°, 10°]$ are not considered for XO calculation due to the bad solution caused by their near-zero slope (figure 3.2)

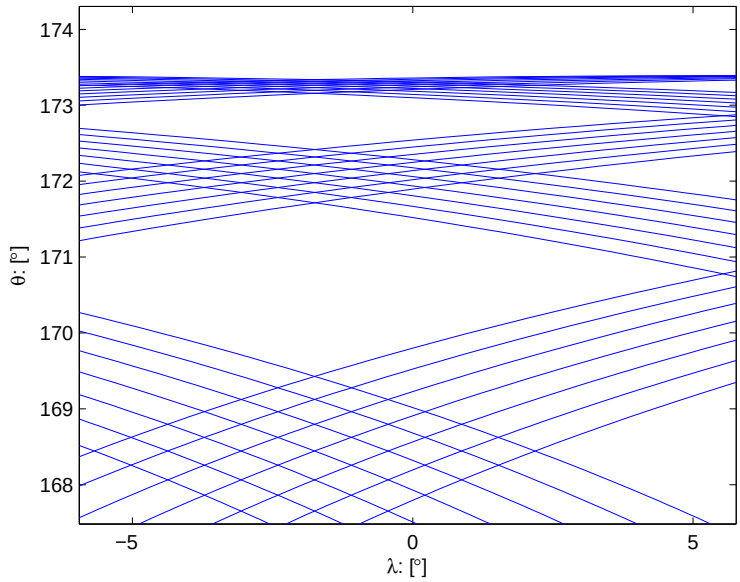


Figure 3.2: *Trajectory in near North Pole region*

In order to check the quality of polynomial approximation, figure 3.3 shows an example of an analytical XO with its corresponding visional XO.

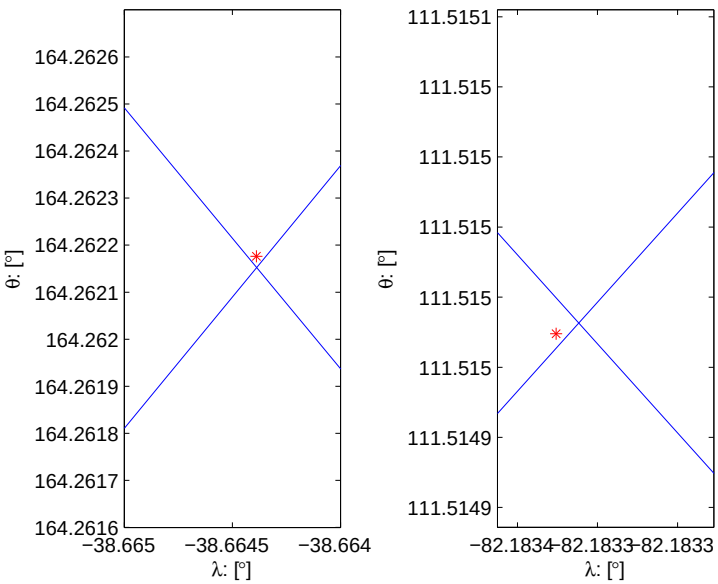


Figure 3.3: *Deviation of cross-over location*

About $2 \cdot 10^{-5^\circ}$ deviation in north direction can be seen in high latitude (θ near 164°) from the left part of figure 3.3. About $2 \cdot 10^{-5^\circ}$ deviation in west direction is visible in low latitude (θ near 111°) from the right part of figure 3.3. Both linear approximation and third-order polynomial approximation fit well in the regional approximation. The slight visional deviation is mainly caused by approximation errors. However, its effect can be ignored, since the same latitude and longitude is used for further interpolation along ascending track and descending track. The location error will drop out after the calculation of differences.

An overview of the XOs within the one-week orbit simulation together with the ground track is shown in figure 3.4.

3.5 Interpolation of height

After the location of XOs in two dimension, the interpolation of their height along ascending and descending track can be investigated. The Matlab rountine "`interp1`" is used in this thesis for along-track interpolation.

The first step is to select help points (HP) used for interpolation. Here along-track interpolation is investigated, which means only points within the same track were selected. The number of selected points should ensure "`interp1`" to work properly. Here points in the region of ±6° along north-south direction are selected.

Next, an appropriate method for Matlab routine "`interp1`" has to be chosen. Considering the complexity of the track's trend, two methods offered by "`interp1`" are applied: spline and cubic.

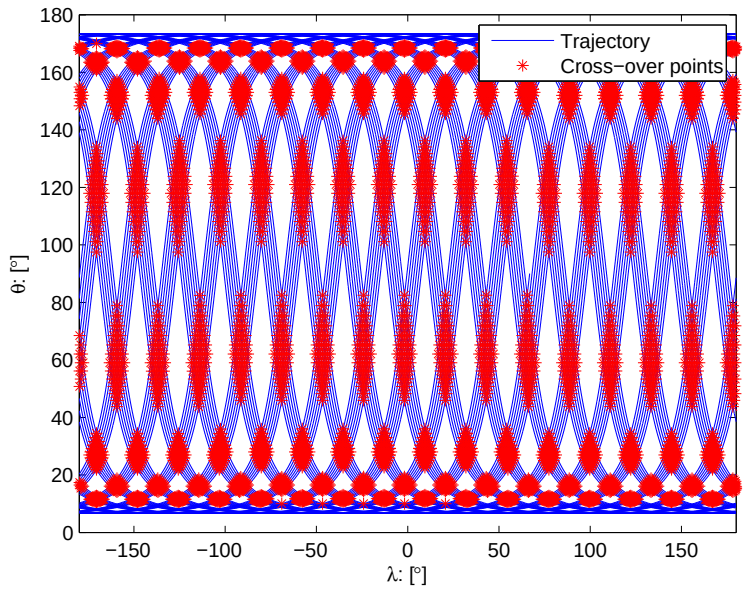


Figure 3.4: *Cross-over points within the one-week orbit simulation*

Spline method performs the cubic spline interpolation with piecewise polynomials. On the contrary, the cubic method (Piecewise Cubic Hermite Interpolating Polynomial), does not take the continuity of the second derivatives into account and oscillation effects are considered to be avoided. The differences between cubic- and spline-interpolation are below 0.8 mm (figure 3.6).

The height difference derived from spline-interpolation with respect of co-latitude is illustrated in figure 3.5. In the region of co-latitude around $\theta = 60°$ and $\theta = 120°$ the hight difference rises to 6 km.

From figure 3.6, we can see that significant deviation appears near the poles. Considering the polar gap of GOCE configuration (no measurements in the regions $\theta \in [0°, 6°]$ and $\theta \in [174°, 180°]$), interpolants near the poles may not have enough help points. Therefore, taken the continuity of the curve's second derivative into account the "spline" method seems to be superior to the "cubic" method. This statement will be verified in chapter 5.

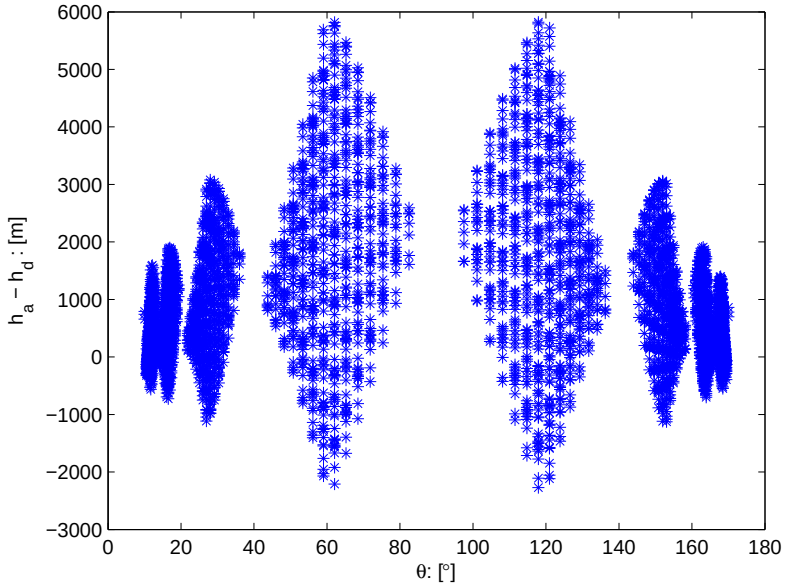


Figure 3.5: *Height difference between ascending and descending track at the same XO*

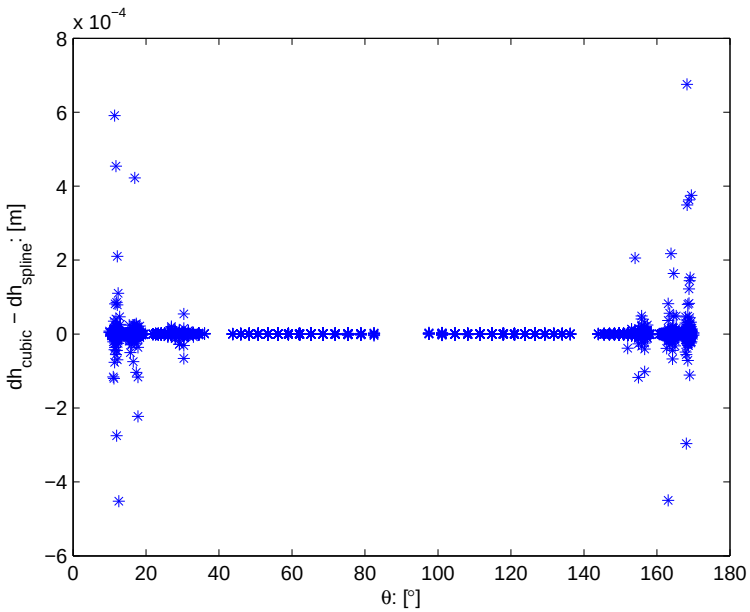


Figure 3.6: *Difference of height reduction between cubic interpolation and spline interpolation*

Chapter 4

Simulation of gravity gradients and invariants

In this chapter, gravity gradients (GGs) and invariants in each point of the GOCE orbit are calculated. They will be used for interpolation at XOs in chapter 5. In order to avoid errors caused by gravity field inconsistencies like the orbit itself, the simulation of GGs and invariants is based on EGM96 up to degree and order 200. Furthermore, the GGs have to be rotated from the model system into the Earth-fixed system, and finally to the local orbit reference frame (LORF) for interpolation and final XO comparison.

4.1 Gravity gradients in the model system

The GOCE gradiometer recovers the Earth's gravity field by full tensor gradiometry, i.e. by measuring the second derivatives of gravitational potential, denoted as gravity (or gravitational) gradients (GGs). In the model frame, the gravitational tensor becomes (Baur 2007),

$$
\begin{aligned}
\underline{\underline{V}} &= \nabla(\nabla V(\mathbf{x}(\lambda,\phi,r))) = \mathbf{e}_i \otimes \mathbf{e}_i V_{ij} \qquad (4.1)\\
&= \frac{GM_0}{R^3}\sum_{m=0}^{L}\sum_{l=0}^{m}\left(\frac{R}{r}\right)^{l+3}[\mathbf{e}_\phi \otimes \mathbf{e}_\phi(-(l+1)e_{lm}(\lambda,\phi)+\frac{\partial^2}{\partial^2\phi}e_{lm}(\lambda,\phi))\\
&\quad +\mathbf{e}_\lambda \otimes \mathbf{e}_\lambda(-(l+1)e_{lm}(\lambda,\phi)+\frac{1}{\cos^2\phi}\frac{\partial^2}{\partial^2\lambda}e_{lm}(\lambda,\phi)-\tan\phi\frac{\partial}{\partial\phi}e_{lm}(\lambda,\phi))\\
&\quad +\mathbf{e}_\lambda \otimes \mathbf{e}_\phi\frac{\partial}{\partial\phi}(\frac{1}{\cos\phi}\frac{\partial}{\partial\lambda}e_{lm}(\lambda,\phi))+\mathbf{e}_\phi \otimes \mathbf{e}_\lambda\frac{\partial}{\partial\phi}(\frac{1}{\cos\phi}\frac{\partial}{\partial\lambda}e_{lm}(\lambda,\phi))\\
&\quad -\mathbf{e}_\lambda \otimes \mathbf{e}_r(l+2)\frac{1}{\cos\phi}\frac{\partial}{\partial\lambda}e_{lm}(\lambda,\phi)-\mathbf{e}_r \otimes \mathbf{e}_\lambda(l+2)\frac{1}{\cos\phi}\frac{\partial}{\partial\lambda}e_{lm}(\lambda,\phi)\\
&\quad -\mathbf{e}_\phi \otimes \mathbf{e}_r(l+2)\frac{\partial}{\partial\phi}e_{lm}(\lambda,\phi)-\mathbf{e}_r \otimes \mathbf{e}_\phi(l+2)\frac{\partial}{\partial\phi}e_{lm}(\lambda,\phi)\\
&\quad +\mathbf{e}_r \otimes \mathbf{e}_r(l+2)(l+1)e_{lm}(\lambda,\phi)]v_{lm},
\end{aligned}
$$

with $\phi = 90^\circ - \theta$.

The coefficient matrix of $\underline{\underline{V}}$ is commonly represented as

$$
\underline{\underline{V}} = [V_{ij}] = \begin{bmatrix} V_{\lambda\lambda} & V_{\lambda\phi} & V_{\lambda r} \\ V_{\phi\lambda} & V_{\phi\phi} & V_{\phi r} \\ V_{r\lambda} & V_{r\phi} & V_{rr} \end{bmatrix}. \qquad (4.2)
$$

$\bar{P}_{lm}(\sin\phi)$ are the associated Legendre functions of the first kind. Its first and second derivatives are,

$$\begin{aligned}
\frac{\partial \bar{P}_{lm}(\sin\phi)}{\partial\phi} &= \frac{1}{\cos\phi}((l+1)\sin\phi\bar{P}_{lm}(\sin\phi) \\
&\quad -(l-m+1)\sqrt{\frac{(2l+1)(l+m+1)}{(2l+3)(l-m+1)}}\bar{P}_{l+1\,m}(\sin\phi)), \qquad (4.3)\\
\frac{\partial^2 \bar{P}_{lm}(\sin\phi)}{\partial\phi^2} &= \tan\phi\frac{\partial \bar{P}_{lm}(\sin\phi)}{\partial\phi} - [l(l+1) - \frac{m^2}{\cos^2\phi}]\bar{P}_{lm}, \qquad (4.4)\\
e_{lm}(\lambda,\phi) &= \begin{cases} \bar{P}_{lm}(\sin\phi)\cos m\lambda & 0 \le m \le l \\ \bar{P}_{l|m|}(\sin\phi)\sin|m|\lambda & -l \le m \le 0 \end{cases}, \qquad (4.5)\\
P_{lm}(\sin\phi) &= \frac{1}{2^l l!}(1-\sin^2\phi)^{m/2}\frac{1^{l+m}}{\mathrm{d}\sin\phi^{l+m}}(\sin^2\phi-1)^l, \qquad (4.6)\\
\bar{P}_{lm}(\sin\phi) &= \begin{cases} \sqrt{2(2l+1)\frac{(l-m)!}{(l+m)!}}P_{lm}(\sin\phi) & m>0 \\ \sqrt{2(2l+1)}P_l(\sin\phi) & m=0 \\ \sqrt{2(2l+1)\frac{(l-|m|)!}{(l+|m|)!}}P_{l|m|}(\sin\phi) & m<0 \end{cases}, \qquad (4.7)\\
\bar{v}_{lm} &= \begin{cases} \bar{c}_{lm} & 0 \le m \le l \\ \bar{s}_{l|m|} & -l \le m \le 0 \end{cases}. \qquad (4.8)
\end{aligned}$$

From equation 4.1, we see $V_{ij} = V_{ji}$ (symmetric). Due to the basic properties of the gravitational field (Rummel 1986), the Laplace equation $\Delta V = V_{\lambda\lambda} + V_{\phi\phi} + V_{rr} = 0$ holds true outside the mass distribution. Furthermore, the coefficient matrix of GGs is trace-free. Therefore, there are five independent GGs: $V_{\lambda\lambda}, V_{\lambda\phi}, V_{\lambda r}, V_{\phi\phi}, V_{\phi r}$.

4.2 Two-step synthesis

Two-step synthesis formulation (Sneeuw 1994) is applied for the calculation of GGs and invariants, in which the summation scheme is reordered and the latitude and longitude information is dealt with independently:

$$\sum_{l=0}^{\infty}\sum_{m=0}^{l} \rightarrow \sum_{m=0}^{\infty}\sum_{l=m}^{\infty}. \qquad (4.9)$$

Two-step synthesis (Sneeuw 1994):

$$\begin{aligned}
A_m(\phi) &= \sum_{l=m}^{\infty}\bar{P}_{lm}(\sin\phi)\bar{c}_{lm}, \qquad (4.10)\\
B_m(\phi) &= \sum_{l=m}^{\infty}\bar{P}_{lm}(\sin\phi)\bar{s}_{lm}, \qquad (4.11)\\
f(\phi,\lambda) &= \sum_{m=0}^{\infty}A_m(\phi)\cos m\lambda + B_m(\phi)\sin m\lambda. \qquad (4.12)
\end{aligned}$$

The GGs are calculated from equation 4.1 using the two-step harmonic spherical synthesis. Figure 4.1 shows that the main diagonal elements of the gravitational tensor are larger than others by three orders of magnitude.

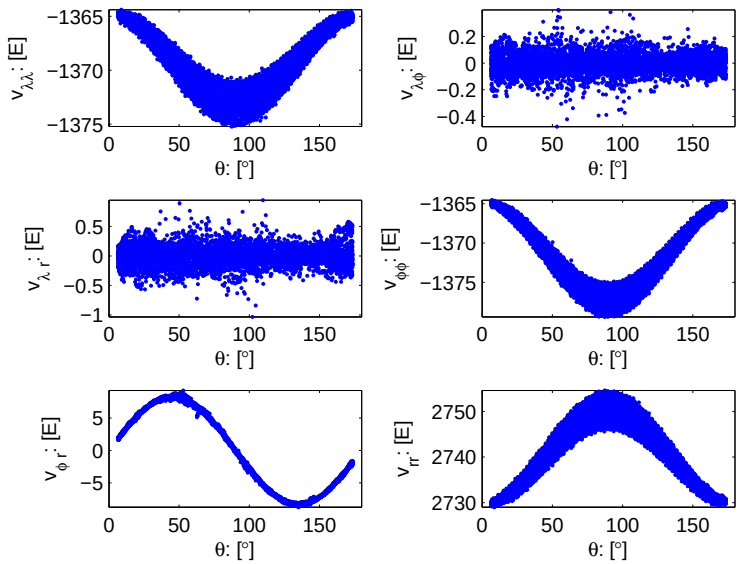


Figure 4.1: *Absolute gravity gradients in the model system*

4.3 Gravity gradients in the Earth-fixed frame

The GGs in the model system should be rotated into the Earth-fixed system with the orthogonal transformation in order to interpolate in a consistent frame.

$$\delta \underline{\underline{V}}^E = \mathbf{R}^{\mathrm{T}}_{M\to E}\,\delta \underline{\underline{V}}^M \mathbf{R}_{M\to E} \tag{4.13}$$

$$= \mathbf{R}_{E\to M}\,\delta \underline{\underline{V}}^M \mathbf{R}^{\mathrm{T}}_{E\to M} \tag{4.14}$$

$$\text{with} \quad \mathbf{R}_{E\to M} = \begin{pmatrix} -\sin\lambda & \cos\lambda & 0 \\ -\sin\phi\cos\lambda & -\sin\phi\sin\lambda & \cos\phi \\ \cos\phi\cos\lambda & \cos\phi\sin\lambda & \sin\phi \end{pmatrix}, \tag{4.15}$$

with $\delta \underline{\underline{V}}^{E(M)}$: reduced tensor by GRS80 in the Earth-fixed system (model system).

Figure 4.2 shows the absolute gravity gradients in the Earth-fixed system (EFS). In order to minimize the interpolation error, GGs and invariants will be reduced by the GRS80 reference field. The results after reduction are shown in figure 4.3.

Since GOCE provides highest accuracy in the components $V_{\lambda\lambda}, V_{\phi\phi}, V_{rr}, V_{\lambda r}$, transformation of the observed gravitational tensor should be avoided. Due to the independency on orientation, tensor invariants analysis is investigated.

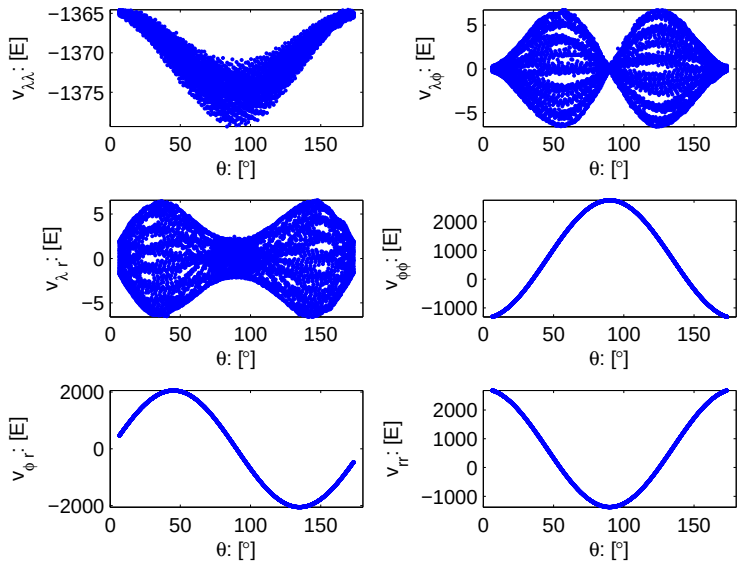


Figure 4.2: *Absolute gravity gradients in the Earth-fixed system*

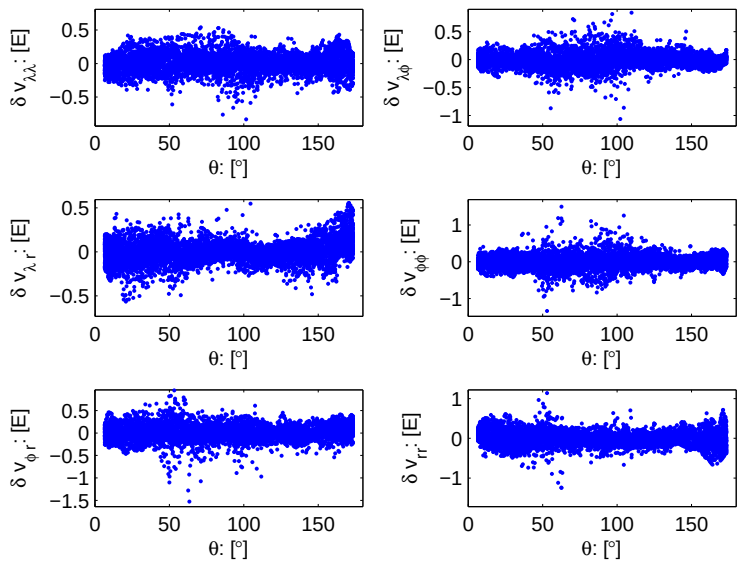


Figure 4.3: *GGs in the Earth-fixed system reduced by GRS80*

4.4 Gravity gradients in the local orbit reference frame

GOCE observations will be collected in the gradiometer frame, which is assumed here to coincide with the local orbit reference frame (LORF).

The relation between the Earth-fixed frame and the LORF is (Baur 2007),

$$
\begin{aligned}
\mathbf{e}^L &= \mathbf{R}(\mathbf{x},\dot{\mathbf{x}})\mathbf{e}^E \\
&= \begin{bmatrix} \frac{\dot{x}_1}{\sqrt{N_1}} & \frac{\dot{x}_2}{\sqrt{N_1}} & \frac{\dot{x}_3}{\sqrt{N_1}} \\ \frac{x_3\dot{x}_2 - x_2\dot{x}_3}{\sqrt{N_2}} & \frac{x_1\dot{x}_3 - x_3\dot{x}_1}{\sqrt{N_2}} & \frac{x_2\dot{x}_1 - x_1\dot{x}_2}{\sqrt{N_2}} \\ \frac{\dot{Z}_1}{\sqrt{N_3}} & \frac{\dot{Z}_2}{\sqrt{N_3}} & \frac{\dot{Z}_3}{\sqrt{N_3}} \end{bmatrix} \\
\text{with} \quad N_1 &= (\dot{x}_1)^2 + (\dot{x}_2)^2 + (\dot{x}_3)^2, \\
N_2 &= (x_2\dot{x}_3 - x_3\dot{x}_2)^2 + (x_3\dot{x}_1 - x_1\dot{x}_3)^2 + (x_1\dot{x}_2 - x_2\dot{x}_1)^2, \\
N_3 &= ((x_3\dot{x}_1 - x_1\dot{x}_3)\dot{x}_3 - (x_1\dot{x}_2 - x_2\dot{x}_1)\dot{x}_2)^2, \\
&+ ((x_1\dot{x}_2 - x_2\dot{x}_1)\dot{x}_1 - (x_2\dot{x}_3 - x_3\dot{x}_2)\dot{x}_3)^2, \\
&+ ((x_2\dot{x}_3 - x_3\dot{x}_2)\dot{x}_2 - (x_3\dot{x}_1 - x_1\dot{x}_3)\dot{x}_1)^2, \\
Z_1 &= (x_2\dot{x}_1 - x_1\dot{x}_2)\dot{x}_2 - (x_1\dot{x}_3 - x_3\dot{x}_1)\dot{x}_3, \\
Z_2 &= (x_3\dot{x}_2 - x_2\dot{x}_3)\dot{x}_3 - (x_2\dot{x}_1 - x_1\dot{x}_2)\dot{x}_1, \\
Z_3 &= (x_1\dot{x}_3 - x_3\dot{x}_1)\dot{x}_1 - (x_3\dot{x}_2 - x_2\dot{x}_3)\dot{x}_2.
\end{aligned}
\tag{4.16}
$$

From equation 4.16 we can see that Cartesian position and velocity coordinates in the Earth-fixed system are required for the transformation from the EFS to the LORF. Cartesian positions and velocities of all orbit points can be generated from the *sosp* software, however, with respect to the inertial system. The transformation to the EFS is achieved via equation 2.2 of both positions and velocities.

Finally, GGs in the LORF can be calculated,

$$
\underline{\underline{V}}^L = \mathbf{R}(\mathbf{x},\dot{\mathbf{x}})^{\mathrm{T}}\underline{\underline{V}}^E\mathbf{R}(\mathbf{x},\dot{\mathbf{x}}) \tag{4.17}
$$

Figures 4.4 and 4.5 illustrate absolute and reduced GGs in the LORF.

4.5 Simulation of invariants

Although the coefficient matrix of GGs is dependent on its orientation, the gravitational tensor is indeed invariant under rotation, which is presented by its invariants. According to Korn and Korn (2000), a tensor of order k holds $k(k+1)/2$ linear independent rotation invariants. In the case of GGs, a second-order ($k = 2$) tensor, I_1, I_2, I_3 compose a complete invariants system. "I_1 equals the trace and I_3 the determinant of the tensor coefficient matrix. I_2 is the sum of the coefficient matrix principal minor determinants by deleting one row and column." (Baur 2007).

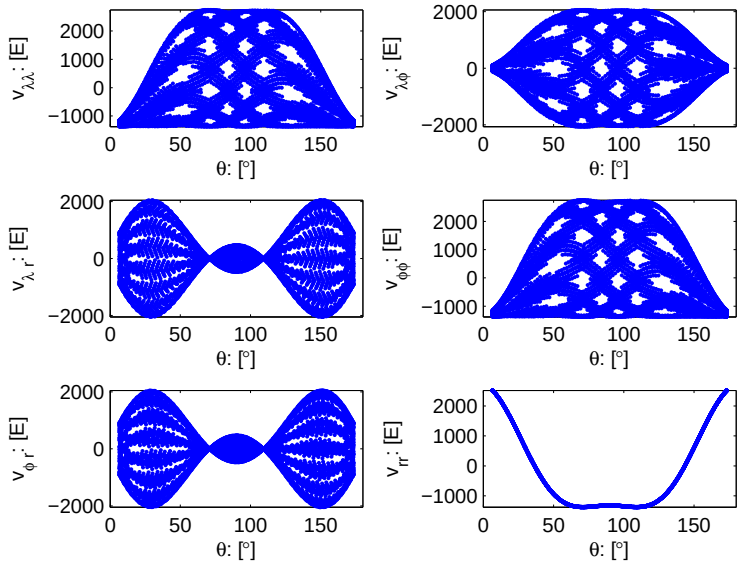


Figure 4.4: *Absolute gravity gradients in the LORF*

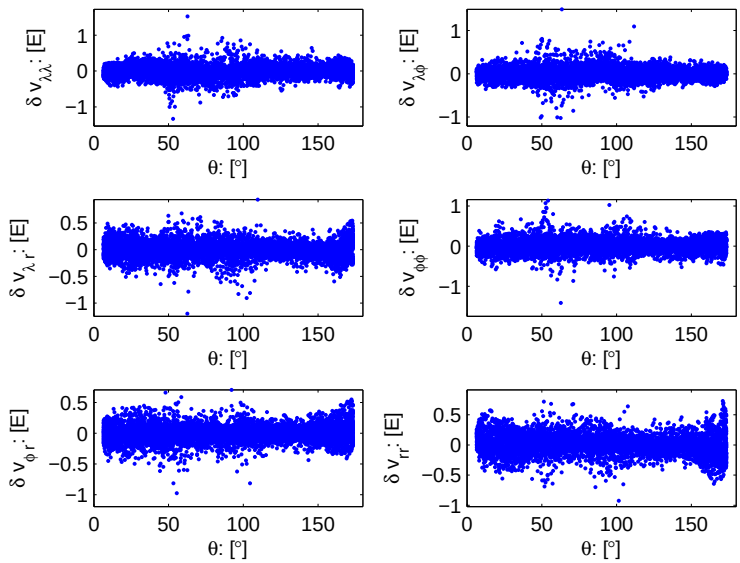


Figure 4.5: *Gravity gradients in the LORF reduced by GRS80*

$$I_1 = \mathrm{tr}\underline{\underline{V}} = 0, \tag{4.18}$$

$$I_2 = -\frac{1}{2}\mathrm{tr}\underline{\underline{V}}^2 = -\frac{1}{2}(V_{11}^2 + V_{22}^2 + V_{33}^2) - V_{12}^2 - V_{13}^2 - V_{23}^2, \tag{4.19}$$

$$I_3 = \det\underline{\underline{V}} = V_{11}V_{22}V_{33} + 2V_{12}V_{13}V_{23} - V_{11}V_{23}^2 - V_{22}V_{13}^2 - V_{33}V_{12}^2 \tag{4.20}$$

The first invariant is consistent with zero due to the trace-free character of the gravitational tensor. δI_2 and δI_3, i.e. I_2, I_3 reduced by the GRS80 reference field, are illustrated in figure 4.6.

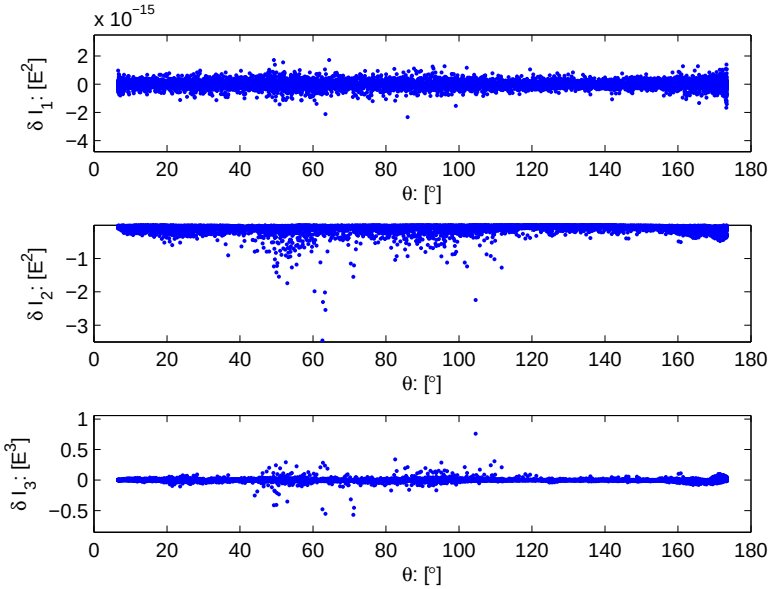


Figure 4.6: *Invariants reduced by GRS80*

Chapter 5

Closed-loop test in the Earth-fixed frame

The comparison of GGs and invariants at XOs in the case of noise free data is performed in terms of a closed loop test. No difference after comparison is expected. The remained residuals are mainly due to interpolation errors.

The closed loop test is at first based on the Earth-fixed system (EFS). The coefficient matrix of GGs at each orbit point has been transformed into the EFS in section 4.3.

5.1 Interpolation in the EFS

The GGs and invariants at each orbit point have been calculated in chapter 4. Due to the sampling rate of 1 Hz, interpolation at XOs along ascending and descending tracks is required. The same interpolation procedure (including selection of help points and using "`interp1`" routine based on both spline- and cubic-method) as interpolation of height (section 3.5) is applied here. Samples of δV_{rr} and δI_2 (reduced GG and invariant by GRS80) interpolation are illustrated in figures 5.1 and 5.2 respectively.

Figures 5.1 and 5.2 show that both the cubic and spline algorithm visionally fit well. The numerical comparison of both interpolation algorithms will be demonstrated in the end.

It is worth to mention that invariants and GGs should be interpolated independently to avoid accumulated errors due to the multiplication of GGs and frame rotation.

5.2 Height reduction in EFS

As indicated in chapter 3, height differences at the XOs with the same geographical coordinates rise to about 6 km due to the orbit drift. Therefore, height reduction (Eq. 5.1) is investigated in this section. In order to derive height reduction, GGs in EFS at each XO pair have to be calculated based on EGM96 up to degree and order 200. Since a consistent gravity field is used for height reduction and generation of GGs, no differences should show up after final comparison. Numerical inconsistencies are due to interpolation errors.

$$\Delta\delta V_{ij}^{E}(\lambda,\phi,\Delta h) = \delta V_{ij}^{E}(\lambda,\phi,r_a) - \delta V_{ij}^{E}(\lambda,\phi,r_d), \tag{5.1}$$

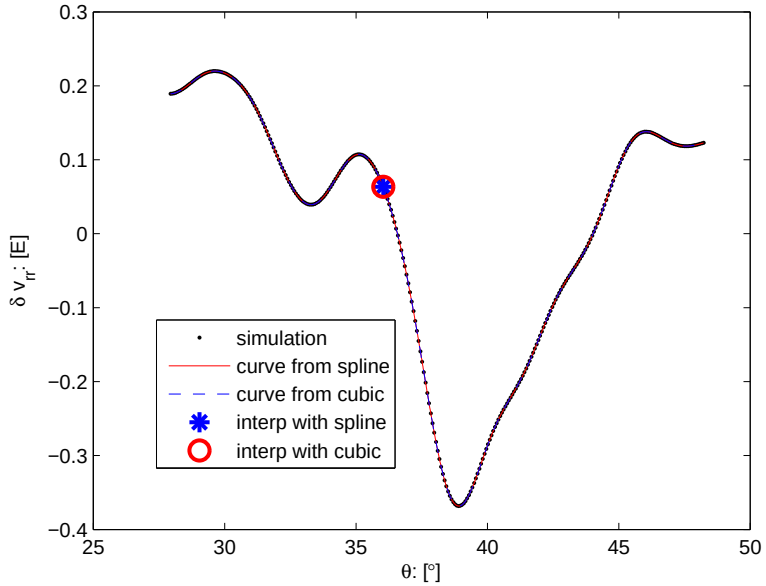


Figure 5.1: *Spline and cubic interpolation of* δV_{rr}

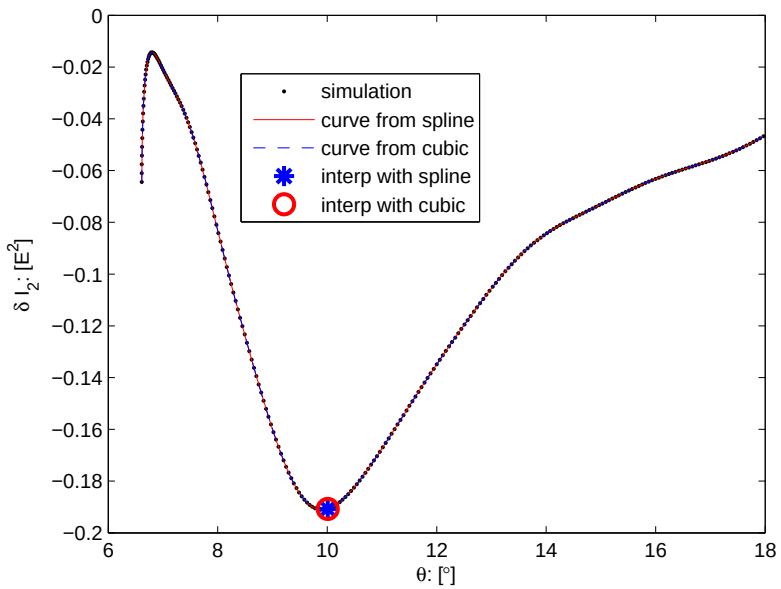


Figure 5.2: *Spline and cubic interpolation of* δI_2

with $\delta V_{ij}^{E}(\lambda,\phi,r_{a(d)})$: reduced GGs in EFS at the position of $\lambda,\phi,r_{a(d)}$ from the ascending (descending) track, and $\Delta\delta V_{ij}^{E}(\lambda,\phi,\Delta h)$: hight reduction, differences of reduced GGs between ascending and descending track with hight difference of Δh at the same geographical position of λ,ϕ .

5.3 Residuals in EFS

After interpolation and height reduction, residuals in the EFS from both spline and cubic interpolation can be generated from the closed loop test.

$$dV_{ij}(\lambda,\phi) = \delta V_{ij}^{*}(\lambda,\phi,r_a) - \delta V_{ij}^{*}(\lambda,\phi,r_d) - \Delta\delta V_{ij}(\lambda,\phi,\Delta h) \tag{5.2}$$

$\delta V_{ij}^{*}(\lambda,\phi,r_a)$: interpolated reduced GGs at XO on ascending track
$\delta V_{ij}^{*}(\lambda,\phi,r_d)$: interpolated reduced GGs at XO on descending track
$\Delta\delta V_{ij}(\lambda,\phi,\Delta h)$: height reduction based on EGM96

The residuals of the reduced GGs; dV_{ij}, and the residuals of the reduced invariants; $dI_{2,3}$, in the EFS derived from both methods with respect of the co-latitude θ are illustrated in figures 5.3 to 5.6. Figures 5.7 and 5.8 show dV_{rr} and dI_3 projected on nominal ground track.

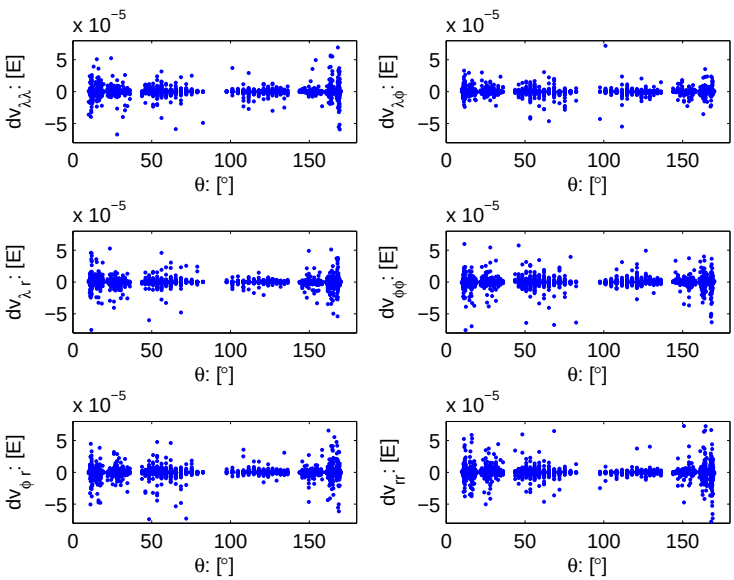


Figure 5.3: *Residuals of reduced GGs from cubic interpolation in the EFS*

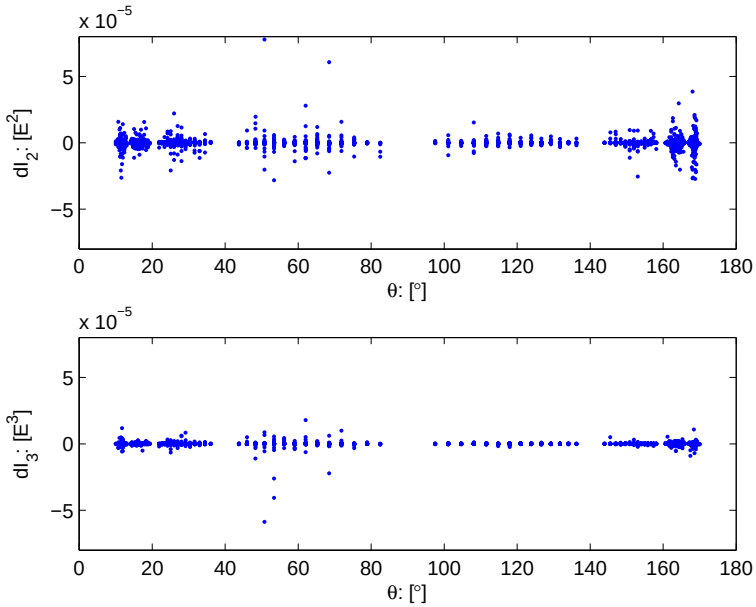


Figure 5.4: *Residuals of reduced invariants from cubic interpolation in the EFS*

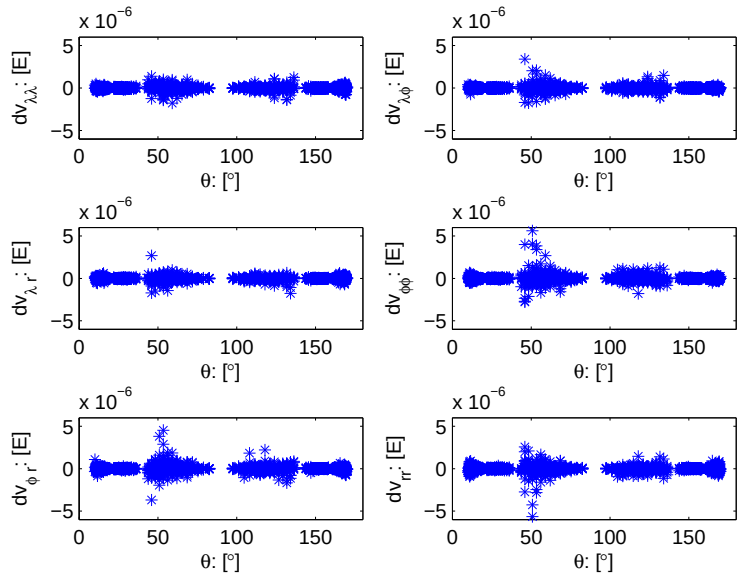


Figure 5.5: *Residuals of reduced GGs from spline interpolation in the EFS*

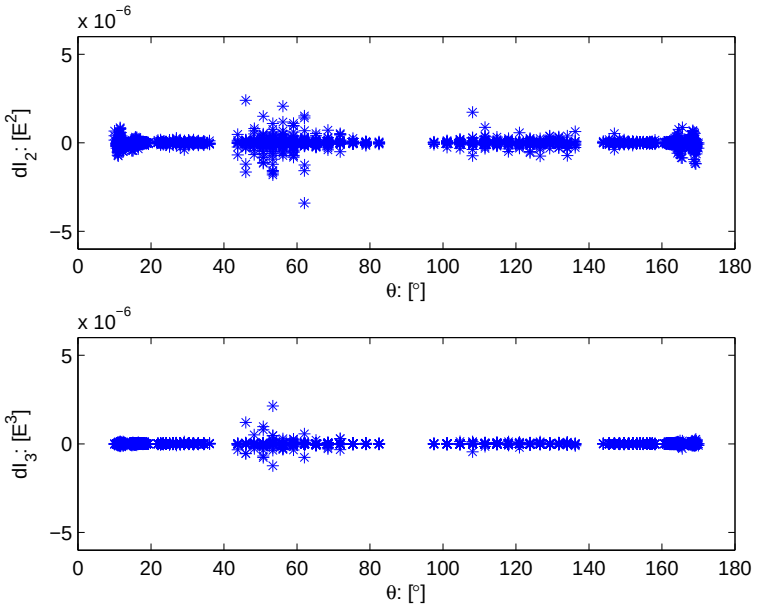


Figure 5.6: *Residuals of reduced invariants from spline interpolation in the EFS*

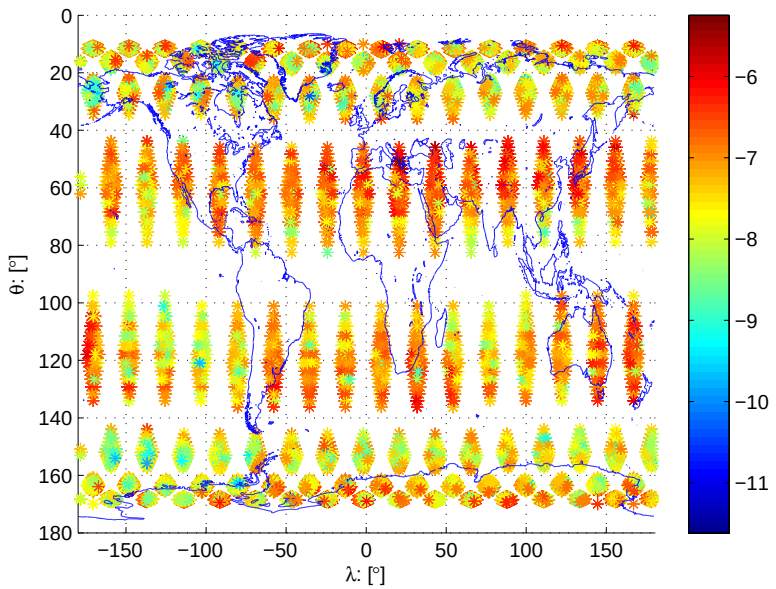


Figure 5.7: *Magnitude of GG's residuals* ($\log_{10} dV_{rr}$) *derived from spline interpolation in the EFS*

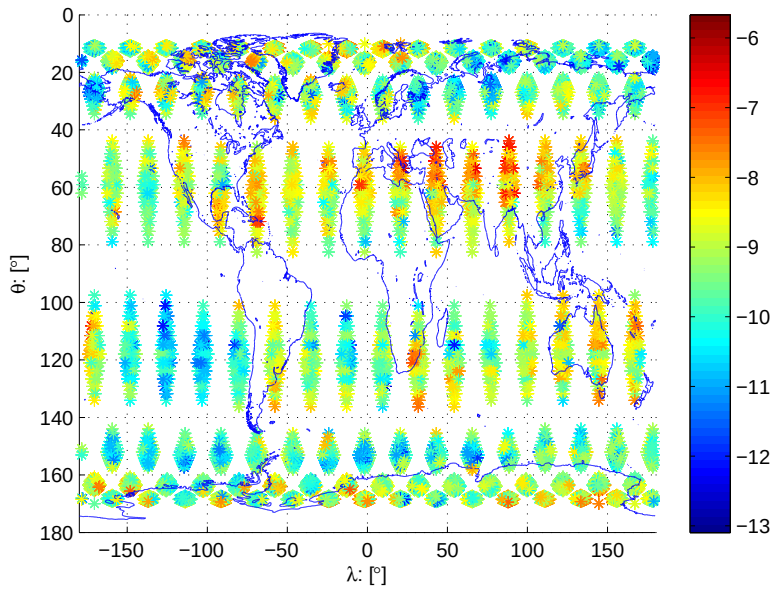


Figure 5.8: *Magnitude of invariant's residuals* ($\log_{10} dI_3$) *derived from spline interpolation in the EFS*

	$dV_{\lambda\lambda}$ $[\mu E]$	$dV_{\lambda\phi}$ $[\mu E]$	$dV_{\lambda r}$ $[\mu E]$	$dV_{\phi\phi}$ $[\mu E]$	$dV_{\phi r}$ $[\mu E]$	dV_{rr} $[\mu E]$	dI_2 $[\mu E^2]$	dI_3 $[\mu E^3]$
mean(cubic)	0.0270	-0.0235	-0.0562	0.0188	0.1219	0.0433	-0.0336	-0.0148
mean(spline)	0.0010	0.0014	0.0010	0.0015	0.0014	-0.0025	-0.0015	-0.0002
max(cubic)	69.229	94.165	253.30	180.74	208.2977	128.17	80.379	58.686
max(spline)	1.3409	3.2146	2.7353	5.6658	4.3686	2.2847	3.0984	2.2741
σ_{cubic}	4.6376	3.6928	5.9268	6.0047	6.6913	6.4607	2.8393	1.2062
σ_{spline}	0.1090	0.1244	0.0953	0.1978	0.1735	0.1683	0.1178	0.0509

Table 5.1: *Residuals from cubic and spline interpolation in the EFS*

Table 5.1 shows that residuals from spline interpolation are smaller than from cubic interpolation by one or two orders of magnitude. The maximum of GGs-residuals derived from spline interpolation is below $6\,\mu E$, whereas it can be above $100\,\mu E$ if derived from cubic interpolation. The most significant errors from cubic-interpolation appear near the Poles, if derived from spline interpolation. They mainly distribute around $45^\circ < \theta < 55^\circ$ (figures 5.7 and 5.8). Generally, invariants are more insensitive than GGs. Furthermore, the third invariant I_3 is more stable than the second one I_2. I_3 consists of one term purely composed of all diagonal elements (Eq. 4.20), which emphasizes the role of diagonal elements by multiplication. On the contrary, the strict separation of diagonal from off-diagonal elements is demonstrated in I_2 (Eq. 4.19).

From the results we can see that the maximum of residuals derived from spline-interpolation is far below GOCE's measurement sensitivity and smaller than cubic-interpolation. Therefore, the goodness of spline-interpolation is proved and will be applied in further tests.

Chapter 6

Closed-loop test in the local orbit reference frame (LORF)

In preparation for XO tests with noisy data in the local orbit reference frame (LORF) in chapter 7, the quality of the interpolation algorithms is first evaluated with noise-free data in the LORF.

Since the final residuals from closed-loop test in the EFS are in the level of μm, the same algorithm ("`interp1`" with spline method) is investigated in the closed-loop test based on the LORF. Compared with the EFS, the LORF is an inconsistent frame due to its dependency on the flight orientation. Since XOs from ascending track and descending track are accompanied by their individual LORFs, the residuals are generated separately along ascending track and descending track. The signal of GGs and invariants at XOs can be simulated based on EGM96. The residuals can be estimated by comparing the interpolation and simulation.

$$dV_{ij_a} = \delta V^{L*}_{ij_a} - \delta V^{L}_{ij_a} \tag{6.1}$$

$$dV_{ij_d} = \delta V^{L*}_{ij_d} - \delta V^{L}_{ij_d} \tag{6.2}$$

The GGs δV^{L*}_{ij} are derived from along-track interpolation. The δV^{L}_{ij} are simulated values. Both are in the LORF and reduced by GRS80.

Since the residuals from ascending and descending tracks are based on different reference frames, analysis using signal to noise ratios is more appropriate than mixed residuals. The signal to noise ratio $\frac{\delta V_{ij}}{dV_{ij}}$ consists of ratios from ascending and descending tracks.

$$\frac{\delta V_{ij}}{dV_{ij}} = |\frac{\delta V_{ij_a}}{dV_{ij_a}}| + |\frac{\delta V_{ij_d}}{dV_{ij_d}}| \tag{6.3}$$

The signal to noise ratio (SNR) of invariants is calculated in the same way.

The SNR of GGs and invariants with respect to the co-latitude θ are illustrated in figures 6.1 and 6.2. Relative large interpolation error occur in the region near $\theta \approx 50°, 120°$. The statistic properties are listed in Table 6.1. (SNR in unit of dB: $10\log_{10}\frac{\delta V_{ij}}{dV_{ij}}$ and $10\log_{10}\frac{\delta I_i}{dI_i}$)

From Table 6.1 we can see that SNR(I_2) > SNR(GGs) > SNR(I_3) in average. The SNR of all GGs and invariants are above 70 dB in average. Since the good performance of one-dimensional interpolation using the spline method is proved in the LORF as well, the same interpolation algorithm will be further applied in chapter 7 with noisy data.

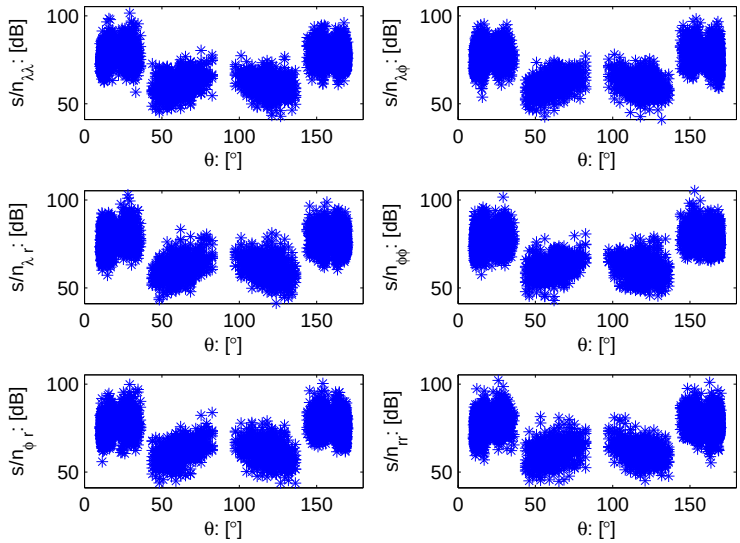


Figure 6.1: *The SNR of GGs from spline interpolation in the LORF with noise-free data*

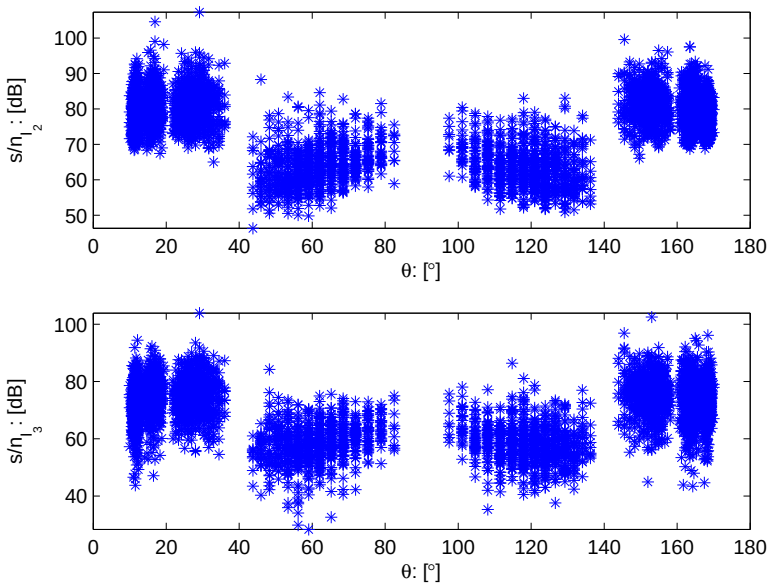


Figure 6.2: *The SNR of invariants from spline interpolation in the LORF with noise-free data*

	$\frac{\delta V_{\lambda\lambda}}{dV_{\lambda\lambda}}$	$\frac{\delta V_{\lambda\phi}}{dV_{\lambda\phi}}$	$\frac{\delta V_{\lambda r}}{dV_{\lambda r}}$	$\frac{\delta V_{\phi\phi}}{dV_{\phi\phi}}$	$\frac{\delta V_{\phi r}}{dV_{\phi r}}$	$\frac{\delta V_{rr}}{dV_{rr}}$	$\frac{\delta I_2}{dI_2}$	$\frac{\delta I_3}{dI_3}$
mean(SNR): [dB]	73.32	72.88	73.18	73.63	73.18	73.34	75.87	70.02
min(SNR): [dB]	42.91	41.12	41.39	42.93	43.71	44.79	46.31	28.39
max(SNR): [dB]	101.57	98.28	102.99	105.34	100.44	102.17	107.32	103.88

Table 6.1: *The SNR of GGs and invariants from spline interpolation in the LORF in closed loop test*

Chapter 7

XO test with noisy data

In order to determine the sensitivity of both GGs and invariants towards the limited GOCE measurement bandwidth (MBW) from 0.5 mHz to 0.1 Hz, simulated noise with adequate stochastic properties is added to the observations for cross-over analysis. The same process of interpolation of the GGs and invariants at XOs is applied as in Chapter 6. In order to test the interpolation performance both white noise and colored noise are envisaged. Furthermore, homogeneous and inhomogeneous noise will be applied. All GGs scenario share the same level of accuracy in the case of homogeneous noise. However, in the more realistic scenario the accuracy of the components $V_{\lambda\phi}$ and $V_{\phi r}$ is reduced by a factor of 100. This is designated as the inhomogeneous case.

7.1 Test with white noise

White noise, or random signal, has a flat power spectral density within a fixed bandwidth. It (figure 7.1) can be generated with Matlab function "`randn`".

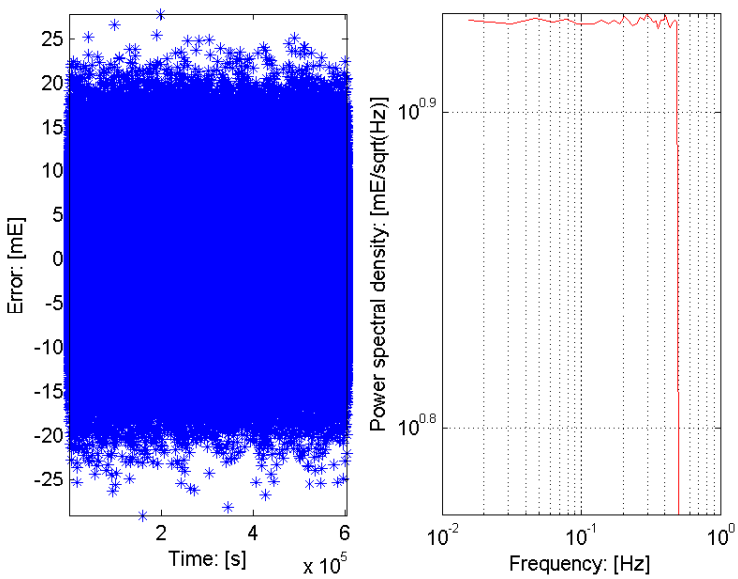


Figure 7.1: *Time series (left) and periodogram (right) of simulated homogeneous white noise with* $\sigma = 6\,mE$

7.1.1 Test with homogeneous white noise

In the case of homogeneous white noise, all GGs components share the same standard deviation (std) of 6 mE. Six series of random noise are added on the simulated GGs ($\delta V^L_{\lambda\lambda}, \delta V^L_{\lambda\phi}, \delta V^L_{\lambda r}, \delta V^L_{\phi\phi}, \delta V^L_{\phi r}, \delta V^L_{rr}$) in the LORF. Interpolation and height reduction for GGs and invariants are investigated independently using the same algorithm as in chapter 6. The SNR of GGs and invariants with respect to co-latitude θ are illustrated in figures 7.2 and 7.3.

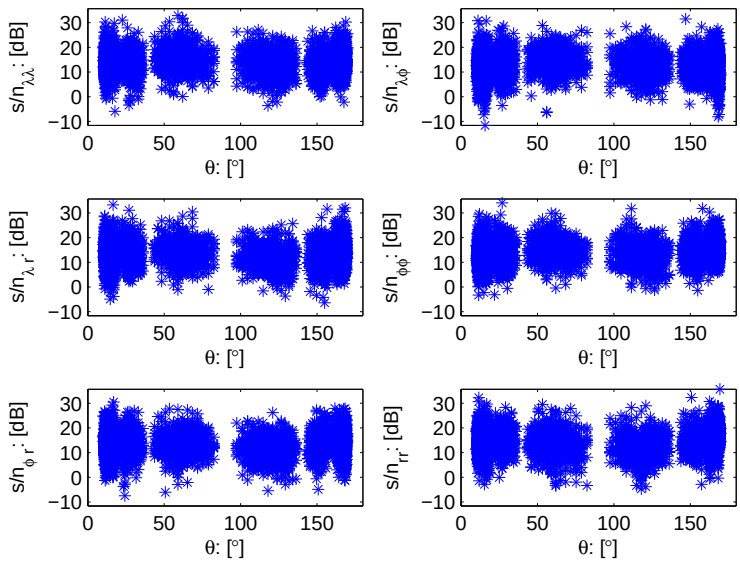


Figure 7.2: *The SNR of GGs in the the LORF with homogenous white noise*

	$\frac{\delta V_{\lambda\lambda}}{dV_{\lambda\lambda}}$	$\frac{\delta V_{\lambda\phi}}{dV_{\lambda\phi}}$	$\frac{\delta V_{\lambda r}}{dV_{\lambda r}}$	$\frac{\delta V_{\phi\phi}}{dV_{\phi\phi}}$	$\frac{\delta V_{\phi r}}{dV_{\phi r}}$	$\frac{\delta V_{rr}}{dV_{rr}}$	$\frac{\delta I_2}{dI_2}$	$\frac{\delta I_3}{dI_3}$
mean(SNR): [dB]	13.97	12.59	13.70	13.95	13.62	14.69	15.79	9.75
min(SNR): [dB]	-5.87	-11.57	-6.39	-3.27	-7.49	-4.64	1.78	-21.05
max(SNR): [dB]	32.88	31.50	33.28	34.09	30.52	35.69	33.30	28.58

Table 7.1: *The SNR of GGs and invariants in the LORF with homogenous white noise*

The SNR of GGs and invariants are presented in dB ($10log_{10}\frac{\delta V_{ij}}{dV_{ij}}$ and $10log_{10}\frac{\delta I_i}{dI_i}$). According to Table 7.1, the SNR of GGs ranges from [-12dB, 36dB]. The SNR with negative sign in dB unit means the noise to be larger than the signal. The $\delta V_{\lambda\phi}$ shows the poorest SNR in average among all GGs. The SNR of the second invariant is better than the value of the third invariant as well as all GGs in average.

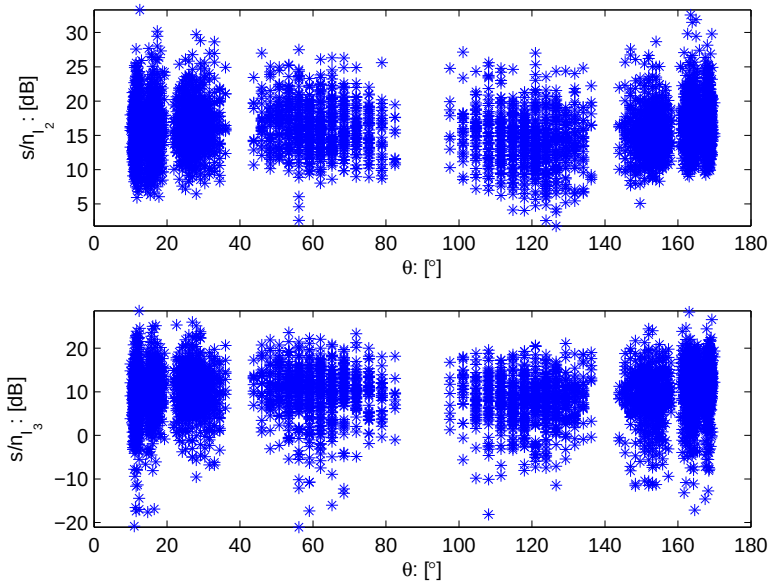


Figure 7.3: *The SNR of invariants in the LORF with homogenous white noise*

7.1.2 Test with inhomogeneous white noise

GOCE can provide high accuracy only in the components $V_{\lambda\lambda}, V_{\lambda r}, V_{\phi\phi}$ and V_{rr}. The other two gravity gradients $V_{\phi r}$ and $V_{\lambda\phi}$ are assumed to be less accurate by a factor of 100 with standard deviation 600 mE. The properties of inhomogeneous white noise analyzed here are shown in Table 7.2.

	$\mathrm{wn}_{\lambda\lambda}$	$\mathrm{wn}_{\lambda\phi}$	$\mathrm{wn}_{\lambda r}$	$\mathrm{wn}_{\phi\phi}$	$\mathrm{wn}_{\phi r}$	wn_{rr}
σ_{wn}:[mE]	6	600	6	6	600	6
$\mathrm{mean}_{\mathrm{wn}}$:[mE]	0.0126	0.3654	0.0126	0.0126	0.3654	0.0126

Table 7.2: *Statistics of the simulated inhomogeneous white noise*

The SNR of GGs and invariants with respect to co-latitude θ are illustrated in figures 7.4 and 7.5.

	$\frac{\delta V_{\lambda\lambda}}{dV_{\lambda\lambda}}$	$\frac{\delta V_{\lambda\phi}}{dV_{\lambda\phi}}$	$\frac{\delta V_{\lambda r}}{dV_{\lambda r}}$	$\frac{\delta V_{\phi\phi}}{dV_{\phi\phi}}$	$\frac{\delta V_{\phi r}}{dV_{\phi r}}$	$\frac{\delta V_{rr}}{dV_{rr}}$	$\frac{\delta I_2}{dI_2}$	$\frac{\delta I_3}{dI_3}$
mean(SNR): [dB]	13.97	-7.41	13.77	14.03	-6.31	14.74	-8.98	-12.56
min(SNR): [dB]	-5.87	-31.57	-5.12	-3.18	-26.59	-4.38	-36.79	-55.56
max(SNR): [dB]	32.88	11.50	29.89	33.06	12.49	34.38	18.44	16.67

Table 7.3: *The SNR of GGs and invariants in the LORF with inhomogeneous white noise*

From Table 7.3 we can see that the SNR of the two off-diagonal components $\delta V_{\lambda\phi}, \delta V_{\phi r}$ are about 20 dB poorer than the others in average. The average SNR of δI_2 and δI_3 are, however, even poorer than the SNRs of all GGs'.

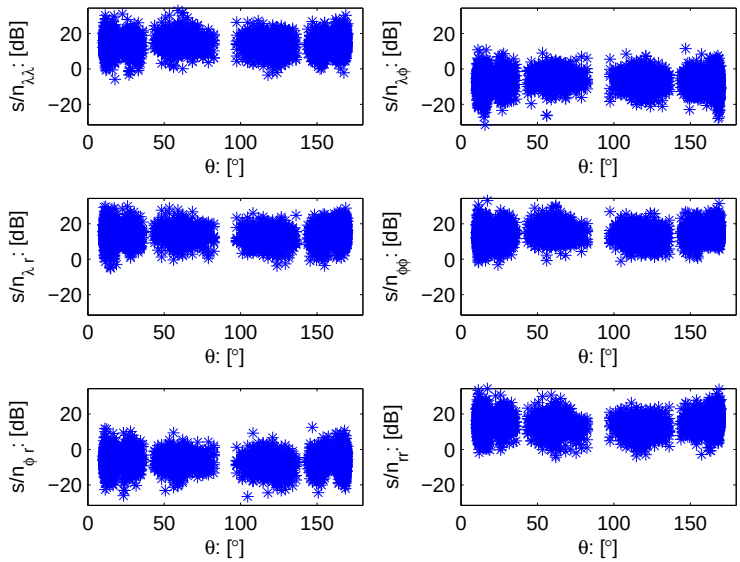


Figure 7.4: *The SNR of GGs in the LORF with inhomogeneous white noise*

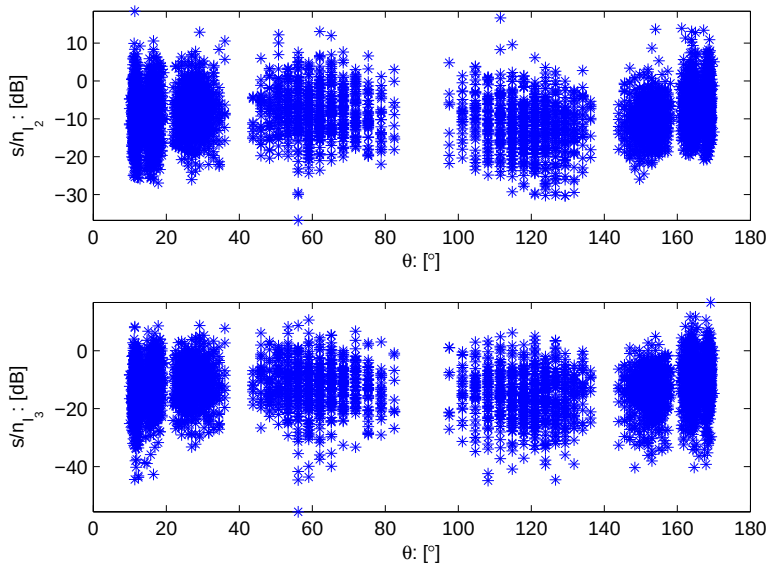


Figure 7.5: *The SNR of invariants in the LORF with inhomogeneous white noise*

7.2 Test with homogeneous colored noise

Since the gradiometer instrument only provides high quality results within the bandwidth ranging from 5 mHz to 0.1 Hz, colored noise model is more realistic for GOCE sensitivity studies. Colored noise can be generated from autoregressive (AR) and moving average (MA) filter applying on white noise (figure 7.6). The first set of homogeneous colored noise can be generated by ARMA coefficients, provided by Bonn University (Shuh 1996), which are applied on homogeneous white noise with standard deviation of 6 mE. The ARMA coefficients for δV_{rr} are further applied on other reduced GGs based on different time series of white noise, because only ARMA coefficients for diagonal GGs are provided.

$$\begin{aligned} y_n = \quad & a_1 y_{n-1} + a_2 y_{n-2} + \cdots + a_p y_{n-p} \\ & b_1 \varepsilon_{n-1} + b_2 \varepsilon_{n-2} + \cdots + b_q \varepsilon_{n-q} + \varepsilon_n \end{aligned} \tag{7.1}$$

with y_n simulated colored noise and ε_n white noise. The ARMA coefficient are denoted as a_i, b_i.

The properties of homogeneous colored noise are listed in Table 7.4

	$cn_{\lambda\lambda}$	$cn_{\lambda\phi}$	$cn_{\lambda r}$	$cn_{\phi\phi}$	$cn_{\phi r}$	cn_{rr}
σ_{cn}:[mE]	35.85	60.12	60.00	44.02	60.60	59.60
$mean_{cn}$:[mE]	2.76	-2.67	-2.83	0.75	-4.39	-2.34

Table 7.4: *Statistics of the simulated homogeneous colored noise*

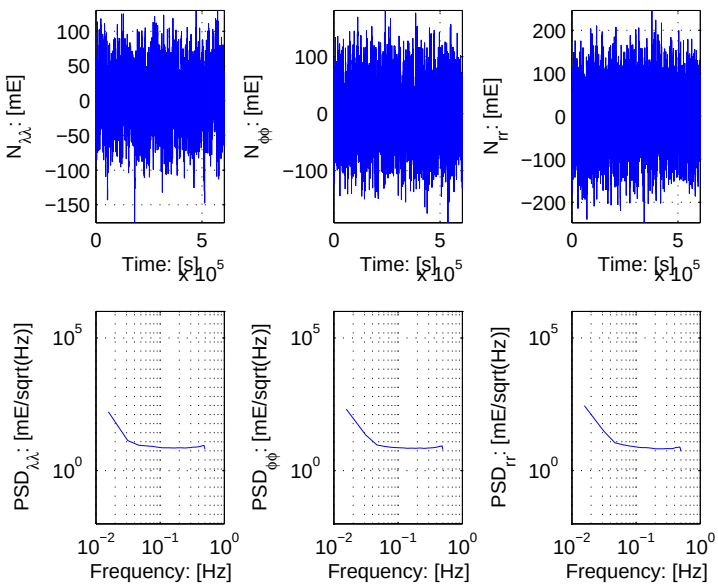


Figure 7.6: *Times series and Periodograms of colored noise for diagonal GGs*

The SNR of GGs range from -20 dB to 25 dB (figures 7.7 and 7.8). The average SNR of the third invariant is below zero (Table 7.5), much poorer than the second one and all GGs. Although there is not distinguished difference between the second invariant's SNR and GGs' SNR in average, the second invariant demonstrates the largest maximum of 29.638 dB as well as the largest minimum of -13.277 dB.

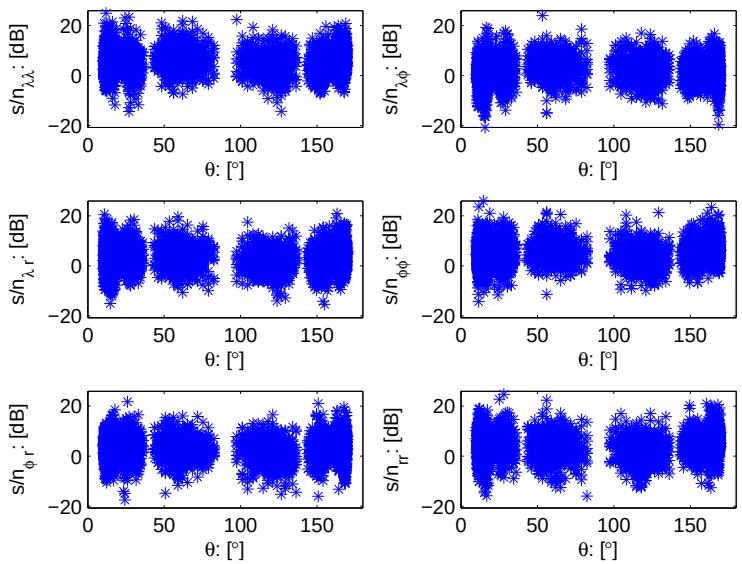


Figure 7.7: *The SNR of GGs in the LORF with homogeneous colored noise*

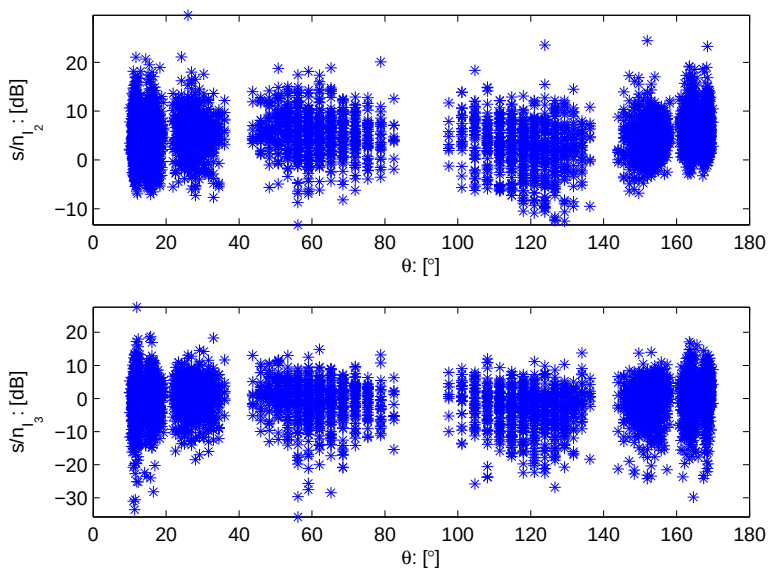


Figure 7.8: *The SNR of invariants in the LORF with homogeneous colored noise*

	$\frac{\delta V_{\lambda\lambda}}{dV_{\lambda\lambda}}$	$\frac{\delta V_{\lambda\phi}}{dV_{\lambda\phi}}$	$\frac{\delta V_{\lambda r}}{dV_{\lambda r}}$	$\frac{\delta V_{\phi\phi}}{dV_{\phi\phi}}$	$\frac{\delta V_{\phi r}}{dV_{\phi r}}$	$\frac{\delta V_{rr}}{dV_{rr}}$	$\frac{\delta I_2}{dI_2}$	$\frac{\delta I_3}{dI_3}$
mean(SNR): [dB]	5.84	2.17	3.45	5.10	3.28	4.32	4.99	-0.47
min(SNR): [dB]	-14.36	-20.74	-15.41	-14.32	-17.53	-16.01	-13.28	-35.76
max(SNR): [dB]	24.93	23.96	20.92	25.97	21.77	24.71	29.64	27.62

Table 7.5: *The SNR of GGs and invariants in the LORF with homogeneous colored noise*

7.3 Test with inhomogeneous colored noise

To generate inhomogeneous colored noise, the same ARMA coefficient will be applied on inhomogeneous white noise from section 7.2. The properties of inhomogeneous colored noise are listed in Table 7.6. The standard deviation of $\delta V_{\lambda\phi}$ and $\delta V_{\phi r}$ go up to 6E.

	$\mathrm{cn}_{\lambda\lambda}$	$cn_{\lambda\phi}$	$cn_{\lambda r}$	$cn_{\phi\phi}$	$cn_{\phi r}$	cn_{rr}
σ_{cn}:[mE]	35.85	6012.39	60.00	44.02	6060.00	59.60
$\mathrm{mean}_{\mathrm{cn}}$:[mE]	2.76	-266.93	-2.83	0.75	-438.84	-2.34

Table 7.6: *Statistics of the simulated inhomogeneous colored noise*

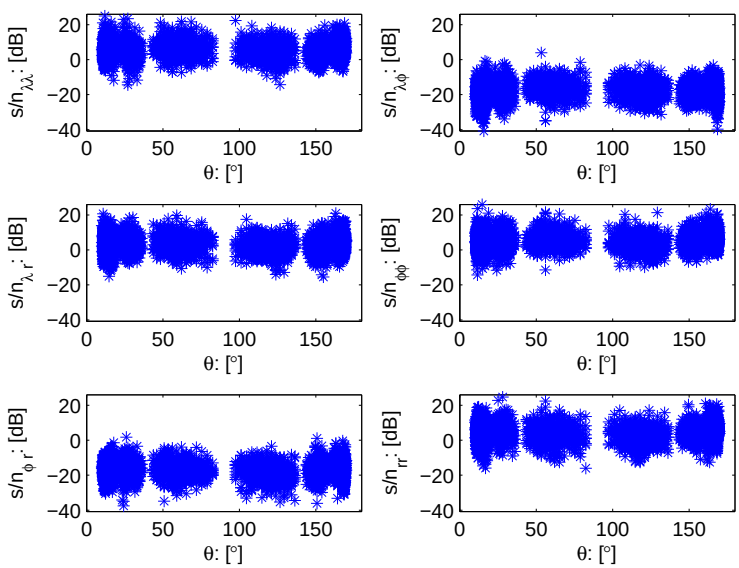


Figure 7.9: *The SNR of GGs in the LORF with inhomogeneous colored noise*

From Table 7.7 and figures 7.9 and 7.10, we can see that the SNR of invariants are more sensitive towards the inhomogeneous colored noise than all GGs, even the maximum of both invariants' SNR are still below -7 dB.

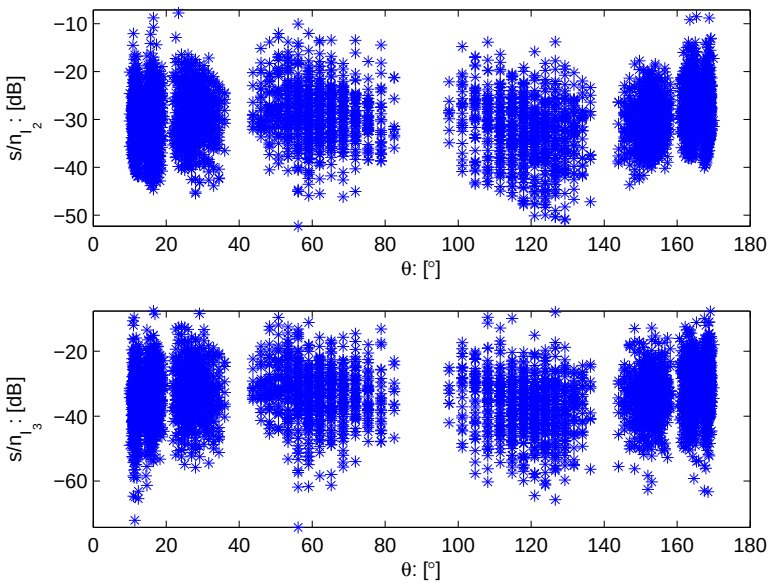


Figure 7.10: *The SNR of invariants in the LORF with inhomogeneous colored noise*

	$\frac{\delta V_{\lambda\lambda}}{dV_{\lambda\lambda}}$	$\frac{\delta V_{\lambda\phi}}{dV_{\lambda\phi}}$	$\frac{\delta V_{\lambda r}}{dV_{\lambda r}}$	$\frac{\delta V_{\phi\phi}}{dV_{\phi\phi}}$	$\frac{\delta V_{\phi r}}{dV_{\phi r}}$	$\frac{\delta V_{rr}}{dV_{rr}}$	$\frac{\delta I_2}{dI_2}$	$\frac{\delta I_3}{dI_3}$
mean(SNR): [dB]	5.84	-17.83	3.45	5.10	-16.72	4.32	-29.43	-33.41
min(SNR): [dB]	-14.36	-40.74	-15.41	-14.32	-37.53	-16.01	-52.32	-74.24
max(SNR): [dB]	24.93	3.96	20.92	25.97	1.77	24.71	-7.13	-7.62

Table 7.7: *The SNR of GGs and invariants in the LORF with inhomogeneous colored noise*

Chapter 8

Summary and outlook

The purpose of this thesis is to compare the sensitivity of GGs and invariants regarding to different noise models in close-loop tests at XOs. Since the coefficient matrix of GGs is orientation dependent, analysis based on invariants is investigated.

At first, XOs can be located in 2D by selecting the maximum of scalar products for XO-prediction and using local polynomial approximation. Matlab routine `interp1` with spline method is investigated for height interpolation and interpolation of GGs as well as invariants at XOs.

Considering the reduced accuracy of the GGs components $V_{\lambda\phi}$ and V_{rr}, various noise types (homogeneous and inhomogeneous white noise, homogeneous and inhomogeneous colored noise) are applied in the closed-loop tests.

In the case of white homogeneous and color homogeneous noise, the SNR of the second invariant (δI_2) is about 3dB stronger than the poorest SNR of GGs ($\delta V_{\lambda\phi}$) in average and the SNR of the third invariant (δI_3) is about 3dB poorer than the SNR of $\delta V_{\lambda\phi}$ in average.

In the case of white inhomogeneous noise, the SNR of δI_2 and δI_3 are about 2dB to 5dB poorer than the poorest SNR of GGs ($\delta V_{\lambda\phi}$) in average, respectively. In the case of colored inhomogeneous noise, the SNR of δI_2 and δI_3 are about 12dB to 15dB poorer than the SNR of $\delta V_{\lambda\phi}$ in average, respectively.

Judging from signal to noise ratio (SNR) at XOs, the conclusion can be made that the second invariant shows less sensitivity in case of homogeneous noise, while the GGs are more stable than invariants with inhomogeneous noise (figure 8.1).

To take the advantage of orientation independency from invariants, instead of the direct measurements from GOCE, simulated GGs components $V_{\lambda\phi}, V_{rr}$ should be investigated in the closed-loop test in order to reduce the impact of the poorly observed components. In this way, validation based on the second invariant is comparable with GGs or even more stable.

White homogeneous	I_2, GGs, I_3
White inhomogeneous	GGs, I_2, I_3
Colored homogeneous	I_2, GGs, I_3
Colored inhomogenous	GGs, I_2, I_3

Figure 8.1: *Stabilities of GGs and invariants with respect to different types of noises*

Bibliography

[1] Baur O., Sneeuw N., Grafarend E.W. (2007), Methodology and use of tensor invariants for satellite gravity gradiometry, Journal of Geodesy 82, pp. 279-293

[2] Escobal P.D. (1975), Methods of Orbit Determination, Robert E. Krieger Publishing Comp., Malabar (Florida)

[3] Goetzelmann M. (2003), Simulation of Satellite Orbits taking into account direct tidal effects, Study Report, online available: http://www.uni-stuttgart.de/gi/education/studienarbeiten/goetzelmann.pdf

[4] Jarecki F., Mueller F. (2003), Validation of GOCE gradients using crossovers, in GEOTECHNOLOGIEN: Observation of the System Earth from Space, GEOTECHNOLOGIEN Science Report No. 3.

[5] Jarecki F., Wolf K.I., Denker H., and Mueller J. (2006), Quality assessment of GOCE gradients; in: Observation of the System Earth from Space, Ed. Flury, Rummel, Reigber, Rothacher, Boedecker, Schreiber, Springer Verlag

[6] Kim M.C. (1997), Theory of satellite ground-track crossovers, Journal of Geodesy 71, pp. 749-767

[7] Korn G.A., Korn T.M. (2000), Mathematical handbook for scientists and engineers, Dover Publications NewYork

[8] Lemoine F.G., Kenyon S.C., Factor J.K., Trimmer R.G., Pavlis N.K., Chinn D.S., Cox C.M., Klosko S.M., Luthcke S.B., Torrence M.H., Wang Y.M., Williamson R.G., Pavlis E.C., Rapp R.H., Olson T.R. (1998), The Development of the Joint NASA GSFC and NIMA Geopotential Model EGM96, NASA Goddard Space Flight Center, Greenbelt, Maryland, USA

[9] McCarthy D.D. (1996), IERS Conventions, IERS Technical Note, No. 21

[10] Mueller J., Dener H., Jarecki F., Wolf K.I. (2004), Computation of calibration gradients and methods for in-orbit validation of gradiometric GOCE data; in: Proc. of the 2nd Int. GOCE User Workshop: GOCE, The Geoid and Oceanography, ESA SP-569

[11] Rummel R. (1986), Satellite Gradiometry, Mathematical and Numerical Techniques in Physical Geodesy, Lecture Notes in Earth Sciences 7, pp. 318-363

[12] Schuh W.D. (1996), Tailored numerical solution strategies for the global determination of the Earth's gravity field, Mitteilungen der Universität Graz, 81

[13] Sneeuw N. (1994), Global spherical harmonic analysis by least squares and numerical quadrature methods in historical perspective, Geophysical Journal International 118, pp. 707-716

Appendix Matlab Programs

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% main_xo_best.m file:main program (For Chapter 2 and Chapter 3 of the Diplom thesis)
% Functions including:
% 1. Coordinates Transformation
% 2. Cross-over Location
% 3. Interpolation of heights and velocities at XOs
% uses: if2efmJd.m, jump2nan.m, brk.m, crossover2.m, finterp_a.m,finterp_d.m, M2V.m
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

[t  X Y Z vx vy vz ax ay az] = textread(’output.txt’,’%f %f %f %f %f %f %f %f %f %f’);
t1 = 3600*7*24;
x = X(1:t1); y = Y(1:t1); z = Z(1:t1);
[xe, ye, ze] = if2efmJd(x,y,z,t(1:t1));
% Transformation of coordinates from inertial system to earth fixed system
[vxe, vye, vze] = if2efmJd(vx,vy,vz,t(1:t1));
% Transformation of velocities from inertial system to earth fixed system
[lam,phi,r] = cart2sph(xe,ye,ze);
% Transformation into geographical coordinates
lam = lam * 180/pi;
% lam, th, phi in [grad]!!!!!
phi = phi * 180/pi;
th = 90-phi;
[lam2,th2] = jump2nan(lam,180,th);
% avoiding jump-problem by plotting

% Separating ascending and descending track
dth = th(2:end)-th(1:end-1);
num_as = find(dth>0);
% "_as" for ascending track
num_ds = find(dth<0);
% "_ds" for descending track
lam_as = lam(num_as);lam_ds = lam(num_ds);
r_as = r(num_as); r_ds = r(num_ds);
th_as = th(num_as); th_ds = th(num_ds);
xe_as = xe(num_as);ye_as = ye(num_as);ze_as = ze(num_as);
xe_ds = xe(num_ds);ye_ds = ye(num_ds);ze_ds = ze(num_ds);
vxe_as = vxe(num_as); vye_as = vye(num_as); vze_as = vze(num_as);
vxe_ds = vxe(num_ds); vye_ds = vye(num_ds); vze_ds = vze(num_ds);

% Finding break-point to identify tracks-begin and -end
dnum_as = num_as(2:end)-num_as(1:end-1);
dnum_ds = num_ds(2:end)-num_ds(1:end-1);
brk_as = find(dnum_as>50);
% either 2 or 50 or even 2000 is ok, for the continuity ground track is continuous
brk_ds = find(dnum_ds>50);

% Using break-point to save ascending and descending vector in matrix, track by track
lam_mas = brk(lam_as,brk_as);    %"NaN" used for NA
lam_mds = brk(lam_ds,brk_ds);
th_mas = brk(th_as,brk_as);
```

```
th_mds = brk(th_ds,brk_ds);
r_mas = brk(r_as,brk_as);
r_mds = brk(r_ds,brk_ds);
xe_mas = brk(xe_as,brk_as);
xe_mds = brk(xe_ds,brk_ds);
ye_mas = brk(ye_as,brk_as);
ye_mds = brk(ye_ds,brk_ds);
ze_mas = brk(ze_as,brk_as);
ze_mds = brk(ze_ds,brk_ds);
vxe_mas = brk(vxe_as, brk_as); vye_mas = brk(vye_as, brk_as); vze_mas = brk(vze_as, brk_as);
vxe_mds = brk(vxe_ds, brk_ds); vye_mds = brk(vye_ds, brk_ds); vze_mds = brk(vze_ds, brk_ds);

% Normalization of geocentric coordinates
[num_apoint num_aline] = size(ze_mas);      % "num_point" for number of points per track
[num_dpoint num_dline] = size(ze_mds);      % "num_line"  for number of tracks
for i = 1:num_aline
    norm_mas(:,i) = sqrt(xe_mas(:,i).^2 + ye_mas(:,i).^2 + ze_mas(:,i).^2);
    n_xemas(:,i) = xe_mas(:,i)./norm_mas(:,i);
    n_yemas(:,i) = ye_mas(:,i)./norm_mas(:,i);
    n_zemas(:,i) = ze_mas(:,i)./norm_mas(:,i);
end
for i = 1:num_dline
    norm_mds(:,i) = sqrt(xe_mds(:,i).^2 + ye_mds(:,i).^2 + ze_mds(:,i).^2);
    n_xemds(:,i) = xe_mds(:,i)./norm_mds(:,i);
    n_yemds(:,i) = ye_mds(:,i)./norm_mds(:,i);
    n_zemds(:,i) = ze_mds(:,i)./norm_mds(:,i);
end

% 2. Cross-over Location
% Using results from cxy_all.mat (including cxall & cyall)
mcx = cxall(1:113,1:113); mcy = cyall(1:113,1:113);
cx = zeros(num_aline,num_dline); cy = zeros(num_aline,num_dline);  %"0" used for NA

for i = 1:num_aline
    for j = 1:num_dline
        [cxp cyp]=  crossover2(i,j,n_xemas,n_yemas,n_zemas,...
        ... n_xemds,n_yemds,n_zemds,lam_mas, th_mas, lam_mds,th_mds);
        cx(i,j)=cxp; cy(i,j) = cyp;
    end
end

save('Z:\users\c108\GOCE\GOCE\crossover_oneweek2','cx','cy');

% 3. Interpolation of heights and velocities at XOs

% Interpolation of heights
mcx = cx; mcy = cy;
dr_mas = r_mas - mean(r_as); dr_mds = r_mds - mean(r_ds);
% Using reducted height for interpolation
dmcr_int_d1 =  finterp_d(lam_mds, th_mds, dr_mds, mcx, mcy, 'cubic');
dmcr_int_d2 =  finterp_d(lam_mds, th_mds, dr_mds, mcx, mcy, 'spline');
dmcr_int_a1 =  finterp_a(lam_mas, th_mas, dr_mas, mcx, mcy, 'cubic');
dmcr_int_a2 =  finterp_a(lam_mas, th_mas, dr_mas, mcx, mcy, 'spline');
vcr_afull = M2V(dmcr_int_a2) + mean(r_as);
vcr_dfull = M2V(dmcr_int_d2) + mean(r_ds);

% Interpolation of velocities
mcvxe_int_a =  finterp_a(lam_mas, th_mas, vxe_mas, mcx, mcy, 'spline');
mcvxe_int_d =  finterp_d(lam_mds, th_mds, vxe_mds, mcx, mcy, 'spline');
```

```
mcvye_int_a =  finterp_a(lam_mas, th_mas, vye_mas, mcx, mcy, 'spline');
mcvye_int_d =  finterp_d(lam_mds, th_mds, vye_mds, mcx, mcy, 'spline');
mcvze_int_a =  finterp_a(lam_mas, th_mas, vze_mas, mcx, mcy, 'spline');
mcvze_int_d =  finterp_d(lam_mds, th_mds, vze_mds, mcx, mcy, 'spline');

% Save results in vectors
vcvxe_int_a = M2V(mcvxe_int_a);
% "M2V" for Matrix to Vector
vcvxe_int_d = M2V(mcvxe_int_d);
vcvye_int_a = M2V(mcvye_int_a);
vcvye_int_d = M2V(mcvye_int_d);
vcvze_int_a = M2V(mcvze_int_a);
vcvze_int_d = M2V(mcvze_int_d);

% Calculation of cartesic coordinates at XOs based on their geographical
% cooridnates and interpolated heights
vcxfull = M2V(mcx);
vcyfull = M2V(mcy);
vcxlam = vcxfull*pi/180; vcyphi = pi/2 - vcyfull*pi/180;
[vcx_xa vcx_ya vcx_za] = sph2cart(vcxlam, vcyphi, vcr_afull);
[vcx_xd vcx_yd vcx_zd] = sph2cart(vcxlam, vcyphi, vcr_dfull);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% mian_simulationGG.m file: main program for Chapter 4
% Including:
% 1. Simulation GGs and invariants at XOs
% 2. Simulation GGs and invariants at orbits
% uses: vgg3.m
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

load CS1.mat   % EGM96field in CS format
vcxfull = M2V(mcx);
vcyfull = M2V(mcy);

[vmc2_a vec2_a Imc2_a Iec2_a]=vgg3(CS1(1:201,1:201),vcyfull,vcxfull,vcr_afull,1);
[vmc2_d vec2_d Imc2_d Iec2_d]=vgg3(CS1(1:201,1:201),vcyfull,vcxfull,vcr_dfull,1);

% Converging vector of simulation to matrix, NA filled with zeros.
mcr_a = zeros(size(mcx)); mcr_d = mcr_a;
mvc_lamlam = mcr_a; mvc_lamphi = mcr_a; mvc_lamr = mcr_a; mvc_phiphi = mcr_a;
mvc_phir = mcr_a; mvc_rr = mcr_a;
for i = 1:length(vcxfull)
    [po_a po_d] = find(vcxfull(i)==mcx);
    mcx(po_a,po_d) = vcxfull(i);  mcy(po_a,po_d) = vcyfull(i);
    mcr_a(po_a,po_d) = vcr_afull(i); mcr_d(po_a,po_d) = vcr_dfull(i);
    mvcd_lamlam(po_a,po_d)=vec2_d(i,1);  mvcd_lamphi(po_a,po_d)=vec2_d(i,2);
    mvcd_lamr(po_a,po_d)=vec2_d(i,3);
    mvcd_phiphi(po_a,po_d)=vec2_d(i,4); mvcd_phir(po_a,po_d)=vec2_d(i,5);
    mvcd_rr(po_a,po_d)=vec2_d(i,6);
    mvca_lamlam(po_a,po_d)=vec2_a(i,1);  mvca_lamphi(po_a,po_d)=vec2_a(i,2);
    mvca_lamr(po_a,po_d)=vec2_a(i,3);
    mvca_phiphi(po_a,po_d)=vec2_a(i,4); mvca_phir(po_a,po_d)=vec2_a(i,5);
    mvca_rr(po_a,po_d)=vec2_a(i,6);
end

 [vmc1_a vec1_a Imc1_a Iec1_a]=vgg3(CS1(1:201,1:201),vcyfull,vcxfull,vcr_afull,0);
 [vmc1_d vec1_d Imc1_d Iec1_d]=vgg3(CS1(1:201,1:201),vcyfull,vcxfull,vcr_dfull,0);

nn = length(x); sample = 20000;
```

```
for i = 1:nn/sample
    n1 = (i-1)*sample + 1; n2 = i*sample;
    [vm80(n1:n2,:) ve80(n1:n2,:) Im80(n1:n2,:) Ie80(n1:n2,:)]
    =vgg3(CS1(1:201,1:201),th(n1:n2),lam(n1:n2),r(n1:n2),1);

    save('Z:\users\c108\GOCE\GOCE\GG','vm80','ve80','Im80','Ie80');
end

% vm80 = [vm_lamlam vm_lamphi vm_lamr vm_phiphi vm_phir vm_rr];

 [vm09 ve09 Im09 Ie09]=vgg3(CS1(1:201,1:201),th(1:100:end),lam(1:100:end),r(1:100:end),0);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% main_interp_earthfixedsystem.m file:
% Closed loop test in earth-fixed-system with noise-free data
% Program for Chapter 5
% uses: vm2ve.m, E2L.m, brk.m, finterp_a(d).m, M2V.m, vgg3.m
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

ve90 = vm2ve(vm90,th,lam);
vg90 = E2L(xe,ye,ze,vxe,vye,vze,ve90);

% % %%%%%%%% First Step: Change vectors to Matrix for interpolation!
vg_lamlam_a = ve90(num_as,1); vg_lamlam_d = ve90(num_ds,1);
vgm_lamlam_a = brk(vg_lamlam_a,brk_as); vgm_lamlam_d = brk(vg_lamlam_d,brk_ds);
vg_lamphi_a = ve90(num_as,2); vg_lamphi_d = ve90(num_ds,2);
vgm_lamphi_a = brk(vg_lamphi_a,brk_as); vgm_lamphi_d = brk(vg_lamphi_d,brk_ds);
vg_lamr_a = ve90(num_as,3); vg_lamr_d = ve90(num_ds,3);
vgm_lamr_a = brk(vg_lamr_a,brk_as); vgm_lamr_d = brk(vg_lamr_d,brk_ds);
vg_phiphi_a = ve90(num_as,4); vg_phiphi_d = ve90(num_ds,4);
vgm_phiphi_a = brk(vg_phiphi_a,brk_as); vgm_phiphi_d = brk(vg_phiphi_d,brk_ds);
vg_phir_a = ve90(num_as,5); vg_phir_d = ve90(num_ds,5);
vgm_phir_a = brk(vg_phir_a,brk_as); vgm_phir_d = brk(vg_phir_d,brk_ds);
vg_rr_a = ve90(num_as,6); vg_rr_d = ve90(num_ds,6);
vgm_rr_a = brk(vg_rr_a,brk_as); vgm_rr_d = brk(vg_rr_d,brk_ds);

% %%%%%%%% Second Step: interpolation with results in Matrix
% % %%%%%%interpolation of ascending track
vcgm_lamlam_int_a = finterp_a(lam_mas, th_mas, vgm_lamlam_a, mcx, mcy, 'spline');
clear vgm_lamlam_a
vcgm_lamphi_int_a = finterp_a(lam_mas, th_mas, vgm_lamphi_a, mcx, mcy, 'spline');
clear vgm_lamphi_a
vcgm_lamr_int_a = finterp_a(lam_mas, th_mas, vgm_lamr_a, mcx, mcy, 'spline');
clear vgm_lamr_a
vcgm_phiphi_int_a = finterp_a(lam_mas, th_mas, vgm_phiphi_a, mcx, mcy, 'spline');
clear vgm_phiphi_a
vcgm_phir_int_a = finterp_a(lam_mas, th_mas, vgm_phir_a, mcx, mcy, 'spline');
clear vgm_phir_a
vcgm_rr_int_a = finterp_a(lam_mas, th_mas, vgm_rr_a, mcx, mcy, 'spline');
clear vgm_rr_a

% %%%%%% Third Step: Change Matrix results to Vectors
vcgc_lamlam_int_a = M2V(vcgm_lamlam_int_a);  clear vcgm_lamlam_int_a
vcgc_lamphi_int_a = M2V(vcgm_lamphi_int_a); clear vcgm_lamphi_int_a
vcgc_lamr_int_a = M2V(vcgm_lamr_int_a);  clear vcgm_lamr_int_a
vcgc_phiphi_int_a = M2V(vcgm_phiphi_int_a); clear vcgm_phiphi_int_a
vcgc_phir_int_a = M2V(vcgm_phir_int_a);  clear vcgm_phir_int_a
vcgc_rr_int_a = M2V(vcgm_rr_int_a); clear vcgm_rr_int_a
```

```
vcgc_a = [vcgc_lamlam_int_a vcgc_lamphi_int_a vcgc_lamr_int_a ...
...vcgc_phiphi_int_a vcgc_phir_int_a vcgc_rr_int_a];
clear vcgc_lamlam_int_a vcgc_lamphi_int_a vcgc_lamr_int_a :
clear vcgc_phiphi_int_a vcgc_phir_int_a vcgc_rr_int_a;

% % % interpolation of descending track
% %%%%%%% Second Step: interpolation with results in Matrix
vcgm_lamlam_int_d = finterp_d(lam_mds, th_mds, vgm_lamlam_d, mcx, mcy, 'spline');
clear vgm_lamlam_d
vcgm_lamphi_int_d = finterp_d(lam_mds, th_mds, vgm_lamphi_d, mcx, mcy, 'spline');
clear vgm_lamphi_d
vcgm_lamr_int_d = finterp_d(lam_mds, th_mds, vgm_lamr_d, mcx, mcy, 'spline');
clear vgm_lamr_d
vcgm_phiphi_int_d = finterp_d(lam_mds, th_mds, vgm_phiphi_d, mcx, mcy, 'spline');
clear vgm_phiphi_d
vcgm_phir_int_d = finterp_d(lam_mds, th_mds, vgm_phir_d, mcx, mcy, 'spline');
clear vgm_phir_d
vcgm_rr_int_d = finterp_d(lam_mds, th_mds, vgm_rr_d, mcx, mcy, 'spline');
clear vgm_rr_d

%%%%%% Third Step: Change Matrix results to Vectors
vcgc_lamlam_int_d = M2V(vcgm_lamlam_int_d); clear vcgm_lamlam_int_d
vcgc_lamphi_int_d = M2V(vcgm_lamphi_int_d); clear vcgm_lamphi_int_d
vcgc_lamr_int_d = M2V(vcgm_lamr_int_d); clear vcgm_lamr_int_d
vcgc_phiphi_int_d = M2V(vcgm_phiphi_int_d); clear vcgm_phiphi_int_d
vcgc_phir_int_d = M2V(vcgm_phir_int_d); clear vcgm_phir_int_d
vcgc_rr_int_d = M2V(vcgm_rr_int_d); clear vcgm_rr_int_d

vcgc_d = [vcgc_lamlam_int_d vcgc_lamphi_int_d vcgc_lamr_int_d vcgc_phiphi_int_d ...
... vcgc_phir_int_d vcgc_rr_int_d];
clear vcgc_lamlam_int_d vcgc_lamphi_int_d vcgc_lamr_int_d :
clear vcgc_phiphi_int_d vcgc_phir_int_d vcgc_rr_int_d;

%%%%% Height and Orientation reduction
load CS1.mat;

[vmc2_a vec2_a Imc2_a Iec2_a]=vgg3(CS1(1:201,1:201),vcyfull,vcxfull,vcr_afull,1);
[vmc2_d vec2_d Imc2_d Iec2_d]=vgg3(CS1(1:201,1:201),vcyfull,vcxfull,vcr_dfull,1);

vgc2_a = E2L(vcx_xa, vcx_ya, vcx_za, vcvxe_int_a, vcvye_int_a, vcvze_int_a, vec2_a);
vgc2_d = E2L(vcx_xd, vcx_yd, vcx_zd, vcvxe_int_d, vcvye_int_d, vcvze_int_d, vec2_d);

dg_lamlam = vcgc_a - (vcgc_d + (vec2_a - vec2_d));
figure; plot(vcyfull, dg_lamlam,'*')

nn1 = find(dg_lamlam(:,1)>1e-5); nn2 = find(dg_lamlam(:,2)>1e-5);
nn3 = find(dg_lamlam(:,3)>1e-5);
nn4 = find(dg_lamlam(:,4)>1e-5); nn5 = find(dg_lamlam(:,5)>1e-5);
nn6 = find(dg_lamlam(:,6)>1e-5);
vcy_nn1 = vcyfull(nn1); vcx_nn1 = vcxfull(nn1); figure; plot(vcx_nn1, vcy_nn1,'*');
vcy_nn2 = vcyfull(nn2); vcx_nn2 = vcxfull(nn2); figure; plot(vcx_nn2, vcy_nn2,'*');
vcy_nn3 = vcyfull(nn3); vcx_nn3 = vcxfull(nn3); figure; plot(vcx_nn3, vcy_nn3,'*');
vcy_nn4 = vcyfull(nn4); vcx_nn4 = vcxfull(nn4); figure; plot(vcx_nn4, vcy_nn4,'*');
vcy_nn5 = vcyfull(nn5); vcx_nn5 = vcxfull(nn5); figure; plot(vcx_nn5, vcy_nn5,'*');

nn = find(vcxfull>-179&vcxfull<179);
dg_gg = dg_lamlam(nn,:);
figure; plot(vcyfull,dg_lamlam,'*');
figure; plot(vcyfull(nn),dg_gg,'*');
```

```
%%%%%%% interpolation of invariants
Ig90 = gg2I(ve90);
mIg_2a = brk(Ig90(num_as,2),brk_as); mIg_2d =  brk(Ig90(num_ds,2),brk_ds);
mIg_3a = brk(Ig90(num_as,3),brk_as); mIg_3d =  brk(Ig90(num_ds,3),brk_ds);

vcIm_2a = finterp_a(lam_mas, th_mas, mIg_2a, mcx, mcy, 'spline');
vcIc_2a = M2V(vcIm_2a); clear vcIm_2a
vcIm_2d = finterp_d(lam_mds, th_mds, mIg_2d, mcx, mcy, 'spline');
vcIc_2d = M2V(vcIm_2d); clear vcIm_2d
vcIm_3a = finterp_a(lam_mas, th_mas, mIg_3a, mcx, mcy, 'spline');
vcIc_3a = M2V(vcIm_3a); clear vcIm_3a
vcIm_3d = finterp_d(lam_mds, th_mds, mIg_3d, mcx, mcy, 'spline');
vcIc_3d = M2V(vcIm_3d); clear vcIm_3d

vcIg_a = gg2I(vgc2_a); vcIg_d = gg2I(vgc2_d);
vcIc_a = [vcIc_2a vcIc_3a]; vcIc_d = [vcIc_2d vcIc_3d];
clear vcIc_2a vcIc_3a vcIc_2d vcIc_3d
dIg = vcIc_a - (vcIc_d + (vcIg_a(:,2:3) - vcIg_d(:,2:3)));
figure; plot(vcyfull,dIg,'*')
dIgn = dIg(nn,:);
figure; plot(vcyfull(nn),dIgn,'*');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% main_colorednoise_generation.m file: Preparation for Chapter 7
% 4 Different Noise Typ: homogeneous & inhomogeneous white noise
%                        homogeneous & inhomogeneous colored noise
% uses: armafilter.m
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% 1.1 homogeneous white noise
randn('state',0)  ;  % Verwendung MatLab 5 Random Number Generator
randn('state');
number = 7*24*3600;
w_xyz = randn(number,6)*0.006;

% 1.2 inhomogeneous white noise
randn('state',0)  ;  % Verwendung MatLab 5 Random Number Generator
randn('state');
number = 7*24*3600;
w_xyzx = randn(number,4)*0.006;
w_yz = randn(number,2)*0.6;

% 2.1 homogeneous colored noise
% lesedatei
lesdat = 'filter_30_30.txt';

% grad und ordnung der filter
p = 30;
q = 30;

% anzahl der zeitpunkte (dt = 5s -> beobachtungszeitraum = number*dt)
number = 7*24*3600;
% laenge der aufwaermphase des filters
warmup = 15000;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% fileinput: phi, theta, sigma2
```

```
fid1=fopen(lesdat,'r');
data = fscanf(fid1,'%f %f %f %f %f %f',[6 p+q]);
fclose(fid1);
data=data';
phi_xx = data(1:p,1); phi_yy = data(1:p,4); phi_zz = data(1:p,6);
theta_xx= data(p+1:end,1);  theta_yy=data(p+1:end,4);  theta_zz=data(p+1:end,6);

sigma_xx = 6e-3;
sigma_yy = 6e-3;
sigma_zz = 6e-3;
sigma_xy = sigma_xx;
sigma_xz = sigma_zz;
sigma_yz = sigma_zz;

% white noise
randn('state',0)    % Verwendung MatLab 5 Random Number Generator
randn('state')
e_xx  = randn(warmup+number,1)*sigma_xx;
e_yy  = randn(warmup+number,1)*sigma_yy;
e_zz  = randn(warmup+number,1)*sigma_zz;
e_xz  = randn(warmup+number,1)*sigma_xz;
e_yz  = randn(warmup+number,1)*sigma_yz;
e_xy  = randn(warmup+number,1)*sigma_xy;


arma_xx = armafilter(e_xx,[1 -phi_xx'],[1 theta_xx']);
arma_yy = armafilter(e_yy,[1 -phi_yy'],[1 theta_yy']);
arma_zz = armafilter(e_zz,[1 -phi_zz'],[1 theta_zz']);
arma_xy = armafilter(e_xy,[1 -phi_zz'],[1 theta_zz']);
arma_xz = armafilter(e_xz,[1 -phi_zz'],[1 theta_zz']);
arma_yz = armafilter(e_yz,[1 -phi_zz'],[1 theta_zz']);

arma_xyz = [arma_xx(warmup+1:end) arma_xy(warmup+1:end) arma_xz(warmup+1:end) ...
...arma_yy(warmup+1:end) arma_yz(warmup+1:end) arma_zz(warmup+1:end)];

mean(arma_xyz)
std(arma_xyz)

N = 2^16;
[pxx,fx]=pwelch(arma_xyz(:,1)*1000,hanning(2*N),N,2*N,1);
loglog(fx,sqrt(pxx),'r*-'); grid on;

% 2.2 inhomogeneous colored noise
% lesedatei
lesdat = 'filter_30_30.txt';

% grad und ordnung der filter
p = 30;
q = 30;

% anzahl der zeitpunkte (dt = 5s -> beobachtungszeitraum = number*dt)
number = 7*24*3600;
% laenge der aufwaermphase des filters
warmup = 15000;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% fileinput: phi, theta, sigma2
fid1=fopen(lesdat,'r');
```

```
data = fscanf(fid1,'%f %f %f %f %f %f',[6 p+q]);
fclose(fid1);
data=data';
phi_xx = data(1:p,1); phi_yy = data(1:p,4); phi_zz = data(1:p,6);
theta_xx= data(p+1:end,1);  theta_yy=data(p+1:end,4);  theta_zz=data(p+1:end,6);

sigma_xx = 6e-3;
sigma_yy = 6e-3;
sigma_zz = 6e-3;
sigma_xy = sigma_zz*100; %sigma_xy = sigma_xx*100;
sigma_xz = sigma_zz;
sigma_yz = sigma_zz*100; %sigma_yz = sigma_zz*100;

% weisses rauschen
randn('state',0)    % Verwendung MatLab 5 Random Number Generator
randn('state')
e_xx  = randn(warmup+number,1)*sigma_xx;
e_yy  = randn(warmup+number,1)*sigma_yy;
e_zz  = randn(warmup+number,1)*sigma_zz;
e_xz  = randn(warmup+number,1)*sigma_xz;
e_yz  = randn(warmup+number,1)*sigma_yz;
e_xy  = randn(warmup+number,1)*sigma_xy;

arma_xx = armafilter(e_xx,[1 -phi_xx'],[1 theta_xx']);
arma_yy = armafilter(e_yy,[1 -phi_yy'],[1 theta_yy']);
arma_zz = armafilter(e_zz,[1 -phi_zz'],[1 theta_zz']);
arma_xy = armafilter(e_xy,[1 -phi_zz'],[1 theta_zz']);
arma_xz = armafilter(e_xz,[1 -phi_zz'],[1 theta_zz']);
arma_yz = armafilter(e_yz,[1 -phi_zz'],[1 theta_zz']);

arma_xyz = [arma_xx(warmup+1:end) arma_xy(warmup+1:end) arma_xz(warmup+1:end) ...
...arma_yy(warmup+1:end) arma_yz(warmup+1:end) arma_zz(warmup+1:end)];

mean(arma_xyz)
std(arma_xyz)

N = 2^16;
[pxx,fx]=pwelch(arma_xyz(:,1)*1000,hanning(2*N),N,2*N,1);
loglog(fx,sqrt(pxx),'r*-'); grid on;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% main_interp_gradiometersystem.m file:
% Test with noisy data in LORF
% Program for Chapter 7
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

ve90 = vm2ve(vm90,th,lam);
vg90 = E2L(xe,ye,ze,vxe,vye,vze,ve90);

vg90 = vg90 + w_xyz; % w_xyz presents generated noise from main_colorednoise_generation.m

% % %%%%%%%%% First Step: Change vectors to Matrix for interpolation!
vg_lamlam_a = vg90(num_as,1); vg_lamlam_d = vg90(num_ds,1);
vgm_lamlam_a = brk(vg_lamlam_a,brk_as); vgm_lamlam_d = brk(vg_lamlam_d,brk_ds);
vg_lamphi_a = vg90(num_as,2); vg_lamphi_d = vg90(num_ds,2);
vgm_lamphi_a = brk(vg_lamphi_a,brk_as); vgm_lamphi_d = brk(vg_lamphi_d,brk_ds);
vg_lamr_a = vg90(num_as,3); vg_lamr_d = vg90(num_ds,3);
vgm_lamr_a = brk(vg_lamr_a,brk_as); vgm_lamr_d = brk(vg_lamr_d,brk_ds);
vg_phiphi_a = vg90(num_as,4); vg_phiphi_d = vg90(num_ds,4);
```

```
vgm_phiphi_a = brk(vg_phiphi_a,brk_as); vgm_phiphi_d = brk(vg_phiphi_d,brk_ds);
vg_phir_a = vg90(num_as,5); vg_phir_d = vg90(num_ds,5);
vgm_phir_a = brk(vg_phir_a,brk_as); vgm_phir_d = brk(vg_phir_d,brk_ds);
vg_rr_a = vg90(num_as,6); vg_rr_d = vg90(num_ds,6);
vgm_rr_a = brk(vg_rr_a,brk_as); vgm_rr_d = brk(vg_rr_d,brk_ds);

% %%%%%%%% Second Step: interpolation with results in Matrix
% % %%%%%%interpolation of ascending track
vcgm_lamlam_int_a = finterp_a(lam_mas, th_mas, vgm_lamlam_a, mcx, mcy, 'spline');
clear vgm_lamlam_a
vcgm_lamphi_int_a = finterp_a(lam_mas, th_mas, vgm_lamphi_a, mcx, mcy, 'spline');
clear vgm_lamphi_a
vcgm_lamr_int_a = finterp_a(lam_mas, th_mas, vgm_lamr_a, mcx, mcy, 'spline');
clear vgm_lamr_a
vcgm_phiphi_int_a = finterp_a(lam_mas, th_mas, vgm_phiphi_a, mcx, mcy, 'spline');
clear vgm_phiphi_a
vcgm_phir_int_a = finterp_a(lam_mas, th_mas, vgm_phir_a, mcx, mcy, 'spline');
clear vgm_phir_a
vcgm_rr_int_a = finterp_a(lam_mas, th_mas, vgm_rr_a, mcx, mcy, 'spline');
clear vgm_rr_a

% %%%%%% Third Step: Change Matrix results to Vectors
vcgc_lamlam_int_a = M2V(vcgm_lamlam_int_a);  clear vcgm_lamlam_int_a
vcgc_lamphi_int_a = M2V(vcgm_lamphi_int_a); clear vcgm_lamphi_int_a
vcgc_lamr_int_a = M2V(vcgm_lamr_int_a);  clear vcgm_lamr_int_a
vcgc_phiphi_int_a = M2V(vcgm_phiphi_int_a); clear vcgm_phiphi_int_a
vcgc_phir_int_a = M2V(vcgm_phir_int_a);  clear vcgm_phir_int_a
vcgc_rr_int_a = M2V(vcgm_rr_int_a); clear vcgm_rr_int_a

vcgc_a = [vcgc_lamlam_int_a vcgc_lamphi_int_a vcgc_lamr_int_a vcgc_phiphi_int_a ...
...vcgc_phir_int_a vcgc_rr_int_a];
clear vcgc_lamlam_int_a vcgc_lamphi_int_a vcgc_lamr_int_a vcgc_phiphi_int_a ...
...vcgc_phir_int_a vcgc_rr_int_a;
% % % interpolation of descending track
% %%%%%%% Second Step: interpolation with results in Matrix
vcgm_lamlam_int_d = finterp_d(lam_mds, th_mds, vgm_lamlam_d, mcx, mcy, 'spline');
clear vgm_lamlam_d
vcgm_lamphi_int_d = finterp_d(lam_mds, th_mds, vgm_lamphi_d, mcx, mcy, 'spline');
clear vgm_lamphi_d
vcgm_lamr_int_d = finterp_d(lam_mds, th_mds, vgm_lamr_d, mcx, mcy, 'spline');
clear vgm_lamr_d
vcgm_phiphi_int_d = finterp_d(lam_mds, th_mds, vgm_phiphi_d, mcx, mcy, 'spline');
clear vgm_phiphi_d
vcgm_phir_int_d = finterp_d(lam_mds, th_mds, vgm_phir_d, mcx, mcy, 'spline');
clear vgm_phir_d
vcgm_rr_int_d = finterp_d(lam_mds, th_mds, vgm_rr_d, mcx, mcy, 'spline');
clear vgm_rr_d

%%%%%% Third Step: Change Matrix results to Vectors
vcgc_lamlam_int_d = M2V(vcgm_lamlam_int_d); clear vcgm_lamlam_int_d
vcgc_lamphi_int_d = M2V(vcgm_lamphi_int_d); clear vcgm_lamphi_int_d
vcgc_lamr_int_d = M2V(vcgm_lamr_int_d); clear vcgm_lamr_int_d
vcgc_phiphi_int_d = M2V(vcgm_phiphi_int_d); clear vcgm_phiphi_int_d
vcgc_phir_int_d = M2V(vcgm_phir_int_d); clear vcgm_phir_int_d
vcgc_rr_int_d = M2V(vcgm_rr_int_d); clear vcgm_rr_int_d

vcgc_d = [vcgc_lamlam_int_d vcgc_lamphi_int_d vcgc_lamr_int_d vcgc_phiphi_int_d ...
...vcgc_phir_int_d vcgc_rr_int_d];
clear vcgc_lamlam_int_d vcgc_lamphi_int_d vcgc_lamr_int_d vcgc_phiphi_int_d ...
```

```
...vcgc_phir_int_d vcgc_rr_int_d;

%%%%%% Height and Orientation reduction
load CS1.mat;

[vmc2_a vec2_a Imc2_a Iec2_a]=vgg3(CS1(1:201,1:201),vcyfull,vcxfull,vcr_afull,1);
[vmc2_d vec2_d Imc2_d Iec2_d]=vgg3(CS1(1:201,1:201),vcyfull,vcxfull,vcr_dfull,1);

vgc2_a = E2L(vcx_xa, vcx_ya, vcx_za, vcvxe_int_a, vcvye_int_a, vcvze_int_a, vec2_a);
vgc2_d = E2L(vcx_xd, vcx_yd, vcx_zd, vcvxe_int_d, vcvye_int_d, vcvze_int_d, vec2_d);

dg_lamlam = vcgc_a - (vcgc_d + (vgc2_a - vgc2_d));
% mixed residuals of GGs from ascending and descending track
dg_a = vcgc_a - vgc2_a;
dg_d = vcgc_d - vgc2_d;
dg_ad = abs(dg_a) + abs(dg_d);
vgc2_ad = abs(vgc2_a) + abs(vgc2_d);

% ascending signal to noise
rdg_ad = 10*log10(vgc2_ad./dg_ad);  % signal to noise ratio of GGs

%%%%%%%% interpolation of invariants
Ig90 = gg2I(vg90);
mIg_2a = brk(Ig90(num_as,2),brk_as); mIg_2d =  brk(Ig90(num_ds,2),brk_ds);
mIg_3a = brk(Ig90(num_as,3),brk_as); mIg_3d =  brk(Ig90(num_ds,3),brk_ds);

vcIm_2a = finterp_a(lam_mas, th_mas, mIg_2a, mcx, mcy, 'spline'); vcIc_2a = M2V(vcIm_2a);
clear vcIm_2a
vcIm_2d = finterp_d(lam_mds, th_mds, mIg_2d, mcx, mcy, 'spline'); vcIc_2d = M2V(vcIm_2d);
clear vcIm_2d
vcIm_3a = finterp_a(lam_mas, th_mas, mIg_3a, mcx, mcy, 'spline'); vcIc_3a = M2V(vcIm_3a);
clear vcIm_3a
vcIm_3d = finterp_d(lam_mds, th_mds, mIg_3d, mcx, mcy, 'spline'); vcIc_3d = M2V(vcIm_3d);
clear vcIm_3d

vcIg_a = gg2I(vgc2_a); vcIg_d = gg2I(vgc2_d);
vcIc_a = [vcIc_2a vcIc_3a]; vcIc_d = [vcIc_2d vcIc_3d];
clear vcIc_2a vcIc_3a vcIc_2d vcIc_3d
dIg = vcIc_a - (vcIc_d + (vcIg_a(:,2:3) - vcIg_d(:,2:3)));
% residuals of invariants from as- and descending track

dIg_a = vcIc_a - vcIg_a(:,2:3);
dIg_d = vcIc_d - vcIg_d(:,2:3);
dIg_ad = abs(dIg_a) + abs(dIg_d);
vcIg_ad = abs(vcIg_a(:,2:3)) + abs(vcIg_d(:,2:3));
rdIg_ad =10*log10(vcIg_ad./dIg_ad);
% signal to noise ratio of invariants
```