DATA MINING IN GRACE MONTHLY SOLUTIONS

A thesis accepted by the Faculty of Aerospace Engineering and Geodesy of the University of Stuttgart in partial fulfillment of the requirements for the degree of Doctor of Engineering Sciences (Dr.-Ing.)

by

Muhammad Athar Javaid

Born in Lahore, Pakistan

Main referee:	Prof. Dr. sc. techn. Wolfgang Keller
Co-referees:	Prof. Dr.-Ing.habil. Dr.tech.h.c.mult Dr.-Ing.E.h.mult. Erik Grafarend
	Prof. Dr.-Ing. Stefanos Fasoulas
Defense date	25 February 2019

Institute for Geodesy

University Stuttgart

2019

Contents

NOMENCLATURE

$\mathbf{b}^l$	Bias vector for l^{th} layer, in chapter 5
b_j^l	Bias for j^{th} neuron in l^{th} layer, in chapter 5
$\mathbf{c}_i$	Centroid of the cluster in chapter 3
η	Learning rate, in chapter 5
$\mathbf{f}$	Activation function vector, in chapter 5
$\mathbf{g}$	Gradient of cost function, in chapter 5
$\mathbf{H}$	Hessian of cost function, in chapter 5
i	Iteration, in chapter 5
J^l	Number of neurons in l layer, in chapter 5
ℓ	Coefficient number in the vector of all sine and cosine coefficients
l	Layer in ANN, with $l = 0$ is input layer and L is the output layer, in chapter 5
$\nabla \mathbf{C}(\mathbf{x})$	Gradient of cost function, in chapter 5
$\nabla^2 \mathbf{C}(\mathbf{x})$	Hessian of cost function, in chapter 5

$\mathbf{P}_y$	A Data matrix with the order (12×8277) i.e. (number of months in a year $\times$ number of coefficients in a month) contains the SH coefficients for a year y, in chapter 2 and chapter 3
$\mathbf{P}$	Sample data matrix with κ features and Q samples, in chapter 5
$\mathbf{t}_q$	Target vector corresponds to sample data vector $\mathbf{p}_q$, in chapter 5
$\mathbf{p}_q$	Sample data vector, in chapter 5
$p^{mon}_{\ell,y}$	A variational level SH coefficient
$\mathbf{p}^{mon}_y$	A vector of all monthly variational level SH coefficients
$\mathbf{T}$	Target matrix with M targets and Q samples, in chapter 5
$\boldsymbol{\mu}_s$	Static field or long term mean field
$\boldsymbol{\mu}_y$	Yearly mean field
$\mathbf{W}^l$	Weight matrix connecting l and $l-1$ layer, in chapter 5
w^l_{jk}	Weight connecting the j^{th} neurons of the l^{th} and k^{th} neurons of the $l-1^{\text{th}}$ layer, in chapter 5
$\mathbf{x}$	A data point in chapter 3, a vector containing all unknown (weights and biases) of an ANN of size n, in chapter 5 and satellite position vector in chapter 8

Acronyms

*k*NN	*k* Nearest Neighbors
ANN	*Artificial neural networks*
BP	Backpropagation
EOF	*Empirical orthogonal function*
EWH	*Equivalent water height*
GD	Gradient Descent
LL-SST	Low-low Satellite To Satellite
LM	Levenberg - Marquardt
SH	Spherical Harmonics
SLR	*Satellite laser ranging*
SSE	*Sum of squares error*

ABSTRACT

The institutes of the GRACE science team recover monthly gravity field models in the form of spherical harmonic (SH) coefficients to study in particular the time-varying part of the gravity field. Typical models contain up to 8281 coefficients (degree and order 90/90), which are derived by inverting approximately one-month range-rate measurements of the GRACE satellites. Due to local and regional gravity variations, the coefficients vary by several orders of magnitudes; this is concerning the essential coefficients but also nonessential coefficients, which only vary by numerical, effects in minute ranges. This study shows that the same range-rate measurements when inverted to recover only the essential coefficients result in increasing the degree of freedom. Consequently, the recovered solution is more stable and have a less formal error.

Data mining methods have been utilized to segregate the SH coefficients into essential and nonessential coefficients. The process starts with k-means clustering, followed by a threshold, k-nearest neighbor and finally an artificial neural networks (ANN) based classification. Since SH coefficients of the GRACE models represent the seasonal and interannual variations of the gravity field, they are expected to have a periodic behavior, which is used to train the ANN. The study found that among 8281 SH coefficients 6490 coefficients are essential, and 1787 are nonessential.

ANN is also used to identify the predictable coefficients among the essential coefficients. The time series of coefficients are used to train separate ANNs, for each coefficient. Hence, the study found that out of 6490 essential coefficients only 245 coefficients are predictable and the rest, 6245 coefficients are unpredictable. Eventually, the list of essential coefficients reduces after removing the predicable coefficients from them. The gravity field recovered on the bases of the reduced list is called a reduced GRACE solution.

Variational equation based gravity recovery simulation is used to recover the gravity field from the range-rate observation. It turned out that the stability of the reduced solution improves significantly. However, it reduces the accuracy only within the formal error limits of the original solutions.

ZUSAMMENFASSUNG

Die Einrichtungen des GRACE-Science-Teams bestimmen monatliche Modelle des Erdschwerefeldes in der Form von sphärisch-harmonischen ((SH) Koeffizienten, insbesondere um die zeitlich-variablen Anteile des Schwerefeldes zu untersuchen. Typische Modelle enthalten 8281 Koeffizienten (Grad und Ordnung 90/90), welche aus der Inversion von circa einem Monat an Range-Rate Messungen der GRACE-Satelliten gewonnen werden. Aufgrund der lokalen und regionalen Variationen in der Gravitation variieren die Koeffizienten um mehrere Größenordnungen. Dies betrifft sowohl die wesentlichen Koeffizienten, als auch die nicht-wesentlichen Koeffizienten, wobei sich letztere nur durch numerische Effekte geringfügig ändern. Diese Arbeit zeigt, dass die Lösung der gleichen Range-Rate Beobachtungen nur für die wesentlichen Koeffizienten den Freiheitsgrad erhöht. Daraus ergibt sich, dass die bestimmte Lösung stabilier ist und kleinere formale Fehler aufweist.

Methoden der gezielten Datensuche (sogenanntes „data mining“) werden eingesetzt, um die SH-Koeffizienten in wesentliche und nicht-wesentliche Koeffizienten zu trennen. Der Prozess beginnt mit einem k-mean Cluster-Algorithmus, gefolgt von einer Schranke, einer k-Nächste-Nachbarschaft Klassifizierung und zuletzt einer Klassifzierung durch ein künstliches Neuronales Netz (ANN). Da die SH-Koeffizienten der GRACE Modelle saisonale und jährliche Variationen des Schwerefeldes repräsentieren, kann für diese ein periodisches Verhalten erwartet werden, welches genutzt wird um das ANN zu trainieren. Es stellt sich heraus, dass sich unter den 8281 SH-Koeffizienten 6490 wesentliche und 1787 nicht-wesentliche befinden.

Neuronale Netze werden auch eingesetzt, um die vorhersagbaren Koeffizienten unter den wesentlichen Koeffizienten zu identifizieren. Die Zeitreihen der Koeffizienten werden

eingesetzt, um getrennte ANN für jeden Koeffizienten zu trainieren. Dabei wird festgestellt, dass sich unter den 6490 wesentlichen Koeffizienten nur 245 vorhersagbar befinden und die verbleibenden 6245 sind nicht-vorhersagbar. Die vorhersagbaren Koeffizienten werden dann von der Klasse der wesentlichen Koeffizienten ausgeschlossen. Damit wird die Liste der wesentlichen Koeffizienten etwas verkleinert und die neue Lösung als „reduzierte GRACE-Lösung“ eingeführt.

In einer Simulation werden die Schwerefeldparameter aus den Range-Rate Beobachtungen bestimmt, wobei der Ansatz der Variationsgleichungen eingesetzt wird. Die Simulation zeigt, dass die Stabilität der reduzierten GRACE-Lösung deutlich verbessert wird, während die Genauigkeit nur innerhalb der Grenzen der formalen Fehler aus der ursprünglichen Lösung verringert wird.

ACKNOWLEDGMENTS

It was a long journey and without the support of several people, relations and institutes/organizations it would not be possible for me to reach that far. First of all, I am thankful to God for giving me health courage and strength that I am writing acknowledgments today.

I am thankful to my supervisor Mr Wolfgang Keller for all his support, encouragement, and discussions. It was his unshaken support which encourages me to move on. He was always there to discuss and interpret the results, and guide me towards the right path.

I am thankful to Mr Nico Sneeuw for all his support. I will always keep the memories of small but effective talks with him at coffee machine.

I am thankful to Mr Erik Grafarend, with whom I feel like a family member. His dedication to work motivates me to keep moving forward.

I am thankful to my colleagues here at the institute who gave me company at lunch breaks, helping me with understanding the challenging concepts, and suggesting me changes and improvements in my work in there own critical way without hesitating.

I would like to thank Higher Education Commission HEC, Pakistan for giving me last year stipend, which helps me a lot to concentrate on writeup and defense.

I am very thankful to my family including my father mother, brothers, lovely sister, their families and parents in law who loved me a lot. They all studded by me and took care of my family.

I am thankful to my wife who was taking care of my babies without me acting not only as mother but father as well. I found her a very supportive and strong lady. Without her emotional support, it would be tough for me to stay alone in Germany.

dedicated to my wife

1 INTRODUCTION

Scientific development is evidence of the evolution of human thinking. Although it is a long and slow journey, a rapid increase has been observed in the past few hundred years. When human beings are influenced by their surroundings and compelled by their needs, the first few disciplines about the universe which they began to think, geodesy is one of those.

Geodesy deals with the Earth's shape, size, its orientation in Space, nowadays in space-times, its Newtonian and relativistic gravity field in a corotating Earth. Nowadays, the temporal variations of the *Planet Earth* are described by the kinematics and dynamics, observed at the Earth surface. Unfortunately, we have only limited information about the Earth interior. *Plate tectonics* are observable on the continents, but only partly at the *Sea* and the *Atmosphere*. The complex "*System Earth*" is a well-posed in a review by many authors in Grafarend, Krumm, and Schwarze (eds) (2003). A breakthrough was the recent development to observe the whole arsenal of new terrestrial, airborne as well as satellite-borne measurement techniques for Earth Sciences. It made available *a broad spectrum of measurable Earth* parameters with surprising accuracy and precision, in particular, to resolve the time factor for a rotating-deforming Earth.

The Earth's shape and size for a rotating-deforming body are complex, and there are different acting fields. For example, for *Geometric Geodesy* a new standard tool is GPS / NavStar which helps to determine the position and its variation in time in *a rotating frame of reference defined as early as 1930*, namely the *International Reference Ellipsoid* in terms of the *Somigliana-Pizzetti reference gravity field*, for details see, (Grafarend and Ardalan, 1999b), (Ardalan et al., 2002) and (Grafarend, 2011). Unfortunately, the height of a point is not possible to determine by geometric means, but as *height above the Sea Level*. As

an *equipotential height* closest to *Mean Sea Level* it is well-defined by the *Gauss-Listing Geoid* (Grafarend et al., 2000), unfortunately under the continents, a central research object of *Geodetic Science*. The terrestrial gravity field is described well by *Somigliana-Pizzetti*-IAG/IUGG Resolution of the year 1930 balancing "*gravitation and rotation*" (Grafarend and Ardalan, 2001). Such a gravity field comprises a harmonic part as well as an anharmonic part and a much smaller time-variable part which is describable as a scalar potential field as well as a vector "*rot*" part. The time variable is of the order of millimeters to decimeters, namely expressed as *vertical deflections*, *geoidal heights* and *equivalent water height*. Mass redistribution under, on and above the Earth causes *gravitational and rotational effects* (Polar Motion(PM) and Length-of-Day (LOD)) variations in a deformable Earth (Grafarend and Ardalan, 1999a).

Water circulation and Mass transport in the atmosphere are the most substantial effects of gravity variations with seasonal to interannual periods (Peters et al., 2002) (Wahr et al., 1998). To obtain the global measurements of the gravitational potential satellite-based observations are used; this gives birth to an entire field called *Satellite Geodesy*. The most recent dedicated gravity satellite missions are the CHAllenging Minisatellite Payload (CHAMP) (Reigber et al., 2002), Gravity Recovery and Climate Experiment (GRACE) (Tapley et al., 2004) and Gravity field and steady state Ocean Circulation Explorer (GOCE) (Rummel et al., 2011). Specifically, the GRACE mission, accurately mapping variations in Earth's gravity field has been an overwhelming success. Usage of its data has marks on several research fields, for instance, some recent contributions include, ice-mass variation (Baur et al., 2009) groundwater mapping (Castellazzi et al., 2018), hydrology (Wang et al., 2018) and (Bai et al., 2018), atmosphere (Mehta and Linares, 2017) seismology (Xu et al., 2017) and (Fatolazadeh et al., 2017) where Tkachenko and Lygin (2017) enlist the applications of GRACE data for solving geological and geographic problems. A list of GRACE publications is available at https://grace.jpl.nasa.gov/publications.

1.1 THE GRACE MISSION

The GRACE was a dedicated gravity satellite mission, launched on 17 March 2002 and completed its science mission in October 2017, consists of two identical satellites, in near-

Table 1.1: Orbital parameters of the GRACE satellites

Parameter	Value
semi major axis [m]	6878000.0
eccentricity	0.002
inclination [°]	89
orbit type	near polar

circular, near polar and low earth orbits, separated from each other by approximately 220 km along-track, and linked by a highly accurate inter-satellite K-Band microwave ranging system. The important orbital parameters are given in Table 1.1.

The satellite altitude decays naturally (~30 m/day); thus the satellite does not have the same ground track pattern each month (Tapley et al., 2004). While at certain heights the satellite has repeating ground tracks which cause poor ground coverage and eventually degrade gravity solutions.

The changes in the gravity field perturb the inter-satellite distance caused by mass redistribution in the system Earth. The long term average of the mass distribution within the Earth system determines the static or mean gravity field (Meyer et al., 2012). The motion of water and air mostly causes mass redistribution. The period of redistribution ranges from several hours to several decades, causes temporal gravity variations. Mean, and the varying gravity field, both affect the motion of the satellite around the Earth. Since the GRACE satellites are rotating at different positions in the space, they are affected slightly differently, which causes a small relative motion between the satellites. Microwave signals measure the distance between the two GRACE satellites. The change in the distance between the satellites cases the phase change of the microwave signals. Continuous observation of the phase change gives the inter-satellite range-rate measurements. By precisely tracking two satellite, their positions can be inverted for gravity field mapping (Devaraju, 2015). The institutes of GRACE science team provide gravity field in the form of sets of spherical harmonics (SH) coefficients up to degree and order 90/90 once per month.

In 1924 *International Union of Geodesy and Geophysics* (IUGG) adopts the ellipsoid as the reference surface of the Earth Caputo (1967). Contrary to suggestion, the use of SH coefficients for the gravity field of Earth is ubiquitous. Since the computation of the SH

is relatively straightforward and simple, efficient algorithms are at hand and extensively studied. On the other hand, the reluctance for ellipsoidal harmonics likely arises in part from the overt computational complexity in the formulation of the series. Numerical instabilities exist with the general formulation of the series that currently limit the model resolution to low degrees, e.g. up to degree and order 14/14, which limits their use of higher-degree expansions (Hu, 2012). The application of the spherical harmonics coefficients for field modeling lost accuracy, but we have to live with available spherical data instead of recommended ellipsoidal harmonics.

Major sources of error in GRACE starts from downward continuation problem and includes system-noise errors in the satellite-to-satellite microwave ranging measurements, accelerometer and oscillator. The accuracy depends somewhat on the orbital configuration, for instance, on the altitude and spacecraft separation (Wahr et al., 1998). The altitude of GRACE satellites decay naturally (c.f. www2.csr .utexas.edu/grace) cause the satellites to fall into near *repeat orbit*, which causes a lack of spatial coverage and eventually degrade the gravity solution (Pini, 2012) (Keiko Yamamoto, 2005). Another problem is the presence of the correlated noise in GRACE SH coefficients that increase with the degree and causes the north-south striping in the spatial domain of variance level potential maps (Swenson and Wahr, 2006). The Gaussian filter is suggested to tackle this problem (Jekeli, 1981), which further cause loss of signal and leakage between the basin (Baur et al., 2009). Wouters and Schrama (2007), Bentel (2010) and (Forootan, 2016) presented alternating methods to remove the striping based upon the *empirical orthogonal function* (EOF) analysis. Due to orbital geometry and short separation between the satellites ($\sim$ 200 km), the change in the coefficient C_{20} are not well determined by GRACE, which affects the basin-scale water storage estimates (Chen et al., 2005). Instead, the first and second degree coefficients are better estimated from *satellite laser ranging* (SLR) and *Earth rotational* (EOP) data (Chen et al., 2004). Furthermore, some geophysical models of variability can be better determined from other techniques besides GRACE, such as solid, ocean and pole tides and atmospheric and ocean non-tidal variations, they are removed from the background using the best available external knowledge (Bettadpur, 2012).

Furthermore, due to orbital track coverage limitations, rapid variability in the gravity

field cannot be well determined from GRACE, though, if neglected, it has the potential to corrupt the GRACE estimate through aliasing.

Gravity variations can be obtained from the GRACE data after removing the static field form the monthly solutions (c.f. Section 2.2.2). Gravity varies a lot in some places, for example, the Amazon region and varies slightly in some other, for example, the Arabian peninsula, see Figure 2.1. In the spectral domain, the varying amplitude of the SH coefficients represents the gravity variation. Since we are interested in the gravity variation information, therefore, the coefficients which carry this information in the form of varying amplitude are the most important one and rest with slight amplitude variation are less critical, which raises two critical questions, stated in the next section.

1.2 RESEARCH QUESTIONS

i. Can we classify SH coefficients on the base of their information contents of gravity variations?

ii. Can we ignore SH coefficients with low information contents during the gravity recovery?

1.3 RESEARCH TASKS

To seek the answers of the research questions, stated above, the research tasks are,

i. find and separate different groups of GRACE SH coefficients using the data mining techniques, Such as,

- unsupervised classification
  - k-means clustering
  - threshold
- supervised classification
  - k nearest neighbor (kNN)
  - artificial neural networks (ANN)

ii. find out coefficients which we can predict using the data mining techniques,

iii. recover full spectrum of SH coefficients

iv. recover reduced spectrum of SH coefficients, use coefficients with high information contents of gravity variations only,

v. compare the recovered fields.

1.4 DATA MINING

Data mining is the process of discovering insightful, thought-provoking, and novel patterns, as well as descriptive, understandable and predictive models from large-scale data. The goal of data mining is to uncover the hidden information (Larose, 2004). It involves methods at the intersection of,

- artificial intelligence,
- machine learning,
- statistics,
- deep learning and
- database management.

The primary data mining tasks are data analysis, frequent pattern mining, clustering and classification using the statistical and probabilistic interpretation of data (Zaki and Jr., 2014). Data mining task can be of descriptive and predictive types. Clustering and association rules are the descriptive type while classification, regression and time series analysis is predictive type (Bala and Kumar, 2017). Many statistical and probability analysis comes under the definition of data mining.

There are several clustering and classification methods used in data mining (X. Wu, 2008). The study searches clusters using *k-mean clustering*, classifies the coefficients using *threshold k-nearest neighbor* and ANN, while predicts the future numerical values of the coefficients using *artificial neural networks*.

1.5 MOTIVATION

GRACE worked more than 15 years in the space and its science teams produce a dataset almost every month, which make it a large collection of SH coefficients containing the monthly variation of the gravity field of the Earth. The idea behind this study is to implement data mining techniques to explore its behavior and to look at these coefficients from a new point of view. Besides the formal techniques such as time series analysis and filtering, data mining equips us to analyze the dataset, suggest a procedure to reduce the noise and improve the quality of the recovered coefficients.

In this regard, Piretzidis et al. (2016) presents a very fine example. According to the details, the correlation error of the even and odd degree coefficients are investigated. Several uncorrelated and highly correlated coefficients are selected and a database of several features consists of geometric properties, such as the number of sign changes, total Euclidean length and convex hull area of the coefficients is created. This data is used to train, test and validate the *artificial neural networks* (ANN). Correlated errors free, mass change coefficients computed from GLDAS model are used as the external reference data. The features of all GRACE monthly solutions are extracted. These features are then fed into the trained ANN, which identifies and classifies the coefficients as correlated or uncorrelated.

During the study of the effects of gravity field on the satellite motion, Reigber (1974) discusses the behavior of the SH coefficients and divide them in the axial-symmetric (zonal) coefficients and length-dependent (tesseral) coefficients. The zonal components are more affected by the secular disturbances of nodular and perigee position and long-period disturbances of eccentricity and argument of perigee. Such effects could be relatively accurately separated from the analysis of orbital elements obtained during long-term orbit determination. This led to a reasonably good knowledge of the zonal coefficients. In contrast, the tesseral part, in particular, the high-frequency components, are not so readily observable because of the small size of the amplitudes occurring and the shortness of the individual periods. In the case of repeat orbit, he pointed out that the resonance occurs and amplitude of the specific coefficients of certain orders are affected.

Devaraju (2015) presents segregation among the coefficients on the base of their noise

contents. For a given (SH) degree, sectoral and the near-sectoral elements of the coefficients are more sensitive to the noise as compared to the noise levels of the tesseral and zonal harmonics.

1.6 ORGANIZATION OF CHAPTERS

Before implementing any clustering or classification technique GRACE data undergoes preprocessing steps, which includes formatting and long term mean removal processes.

Chapter 2 encompasses data preprocessing, data format and transformation issues and introduces the GRACE data visualization plot and data analysis tools.

Chapter 3 begins with an introduction and discussion on the unsupervised classification techniques to identify and separate different groups of coefficients.

Chapter 4 introduces the k nearest neighbor as the supervised classification technique.

Chapter 5 introduces ANN and states all aspects of training and testing processes. It also explains the feedforward algorithm, cost evaluation, and its minimization, backpropagation and hyperparameters.

Chapter 6 explores ANN as the supervised classification techniques and validates the results of the unsupervised classification.

Chapter 7 presents ANN application as a prediction tool and identify a group of predictable SH coefficients.

Data mining techniques have identified three groups of SH coefficients which have different characteristics on the base of their information contents or their behavior. They prove that only a fraction of all spherical harmonics up to degree and order 90 is necessary to resolve the monthly changes in the gravitational field (Keller, 2015).

Chapter 8 treats the gravity recovery technique using the variational equations method. It presents the cases of first all coefficients recovery second only essential coefficients recovery and third only unpredictable coefficients recovery.

Chapter 9 summarizes and the whole study and discuss the comparisons of gravity recovery and formal error; furthermore, it concludes the study with the remarks on the benefits of the classification and limitations of ANN prediction.

GRACE DATA 2

This chapter describes the GRACE dataset, its format for visualization, storage and pre-processing. The dataset used here is available at the *podaac* data server, at: *ftp://podaac-ftp.jpl.nasa.gov/allData/grace/*. The server hosts the data produced by three official centres of the GRACE science team, including,

1. GFZ (GeoforschungsZentrum Potsdam),
2. CSR (Center for Space Research at the University of Texas, Austin) and
3. JPL (Jet Propulsion Laboratory).

The dataset contains the gravity recovery solutions in file format, one file for each month, starting from April 2002. Every solution consists of SH coefficients, C_{lm} and S_{lm}, where l and m are the degree and order of a coefficient, and define the horizontal scale of the product as, $\sim 20,000/l$ km (Wahr et al., 2004). Though GRACE was initially launched on a five-year program, it lasts longer. This study uses the GRACE SH dataset from April 2002 to June 2017, from GFZ centre because it provides coefficients up to degree 90 and order 90 along with their standard deviation. The data calendar is given in Table 2.1, where the *equivalent water height* (EWH) maps of the available months are given in *Mollweide projection*. For detail on Mollweide projection, see (Grafarend and Heidenreich, 1995).

2.1 DATA VISUALIZATION

The numerical values of the SH coefficients are very small and the color plots of a special format named as SC format help us to display them for visualization. SC is a matrix format in which sine ($S_{l,m}$) and cosine ($C_{l,m}$) coefficients with degree l and order m are placed in a triangular format.

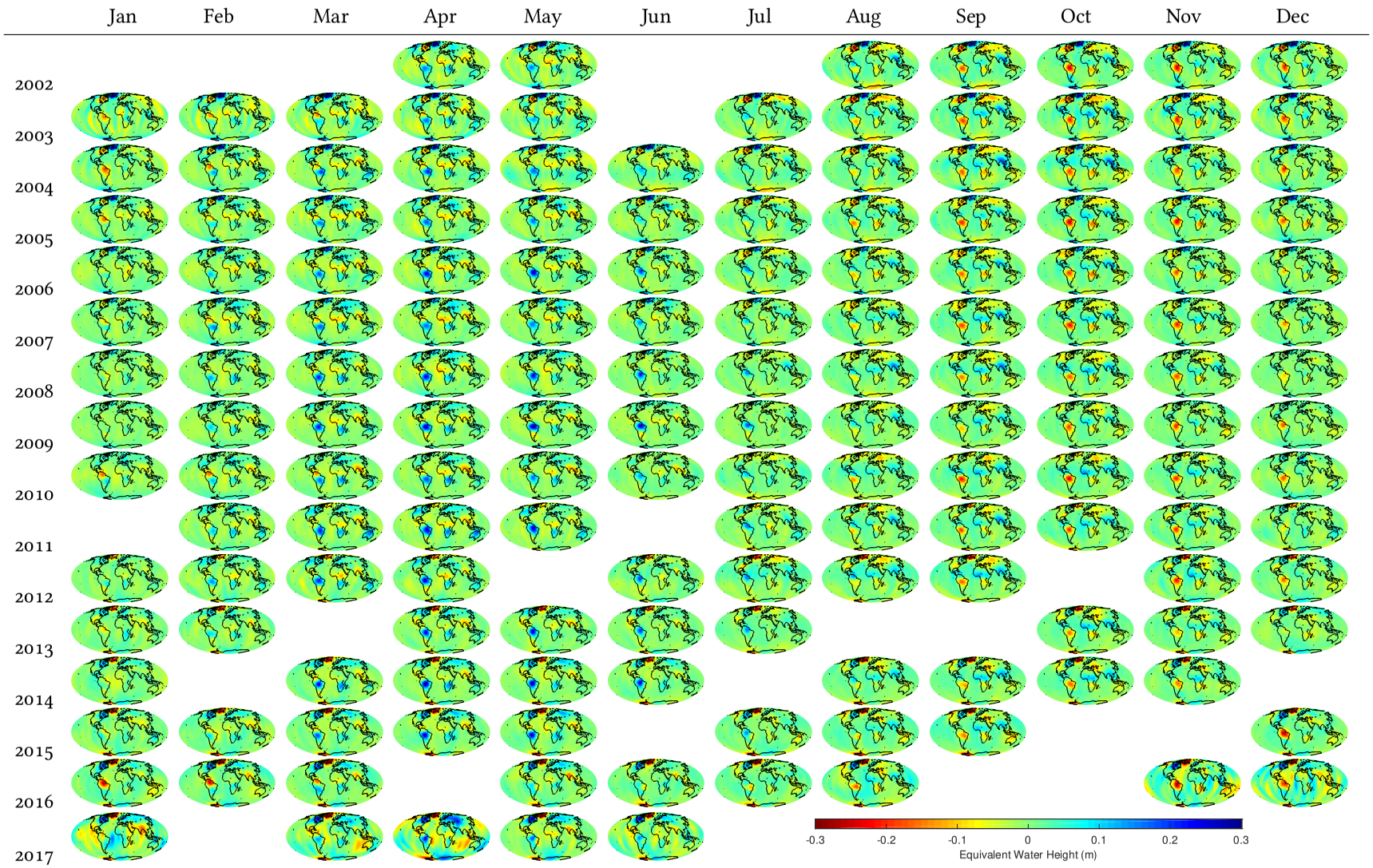


Table:2.1 GRACE calendar of GFZ dataset.

l increases from top to bottom of the matrix while m from center to the left for $S_{l,m}$ coefficients and to the right for $C_{l,m}$ coefficients, as illustrated in the following matrix.

$$\begin{pmatrix} & & & C_{0,0} & & & \\ & & S_{1,1} & C_{1,0} & C_{1,1} & & \\ & S_{2,1} & S_{2,1} & C_{2,0} & C_{2,1} & C_{2,2} & \\ \iddots & & \vdots & \vdots & \vdots & & \ddots \\ S_{l,m} & \cdots & S_{l,1} & C_{l,0} & C_{l,1} & \cdots & C_{l,m} \end{pmatrix}$$

Figure 2.1 represents a color display of the absolute values in logarithmic scale which gives best visualization to the SH coefficients in the SC matrix format. For this document, SC matrix is used to represent the SH coefficients, their formal errors and results of classification, clustering, prediction and gravity recovery processes. This study does not include the following five coefficients. One degree zero, i.e. $C_{0,0}$, three degree one, i.e. $S_{1,1}, C_{1,0}, C_{1,1}$ and a degree two coefficient, i.e. $C_{2,0}$, therefore the plot starts from $l = 2$. If l_{max} denotes the maximum degree of spherical harmonics then the total number of coefficient L, ignoring the five mentioned coefficients, is given by,

$$L = (l_{max} + 1)^2 - 5 \ . \tag{2.1}$$

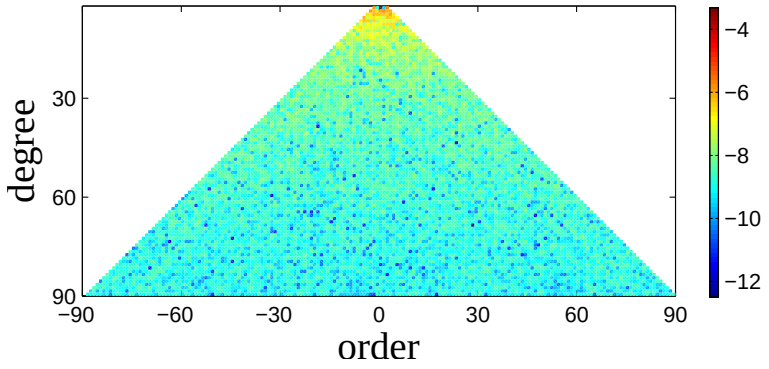


Figure 2.1: A colored representation of absolute values of SH coefficients in logarithmic scale.

2.2 DATA PREPARATION

This section first describes the data format used for storage and analysis and then states the pre-processing steps to remove the long-term mean or static mean to yield the variational level SH coefficients.

2.2.1 DATA FORMAT FOR STORAGE AND ANALYSIS

SC format appears to be the best choice for the visualization of GRACE data and related results, however, it is not good for data storage since it has higher % of sparsity i.e. 50%. This means that a SC matrix is only half filled with non zero numbers and half of it consists of zero or nonnumerical numbers, as a result, they occupy more space in the memory. Therefore, for data storage, processing and analysis some other formats are in use, the SC vector format, for instance, which has SH coefficients in increasing order m for each degree l of all sine coefficient first, followed by all cosine coefficients in a vector as,

$$D = \begin{bmatrix} S_{2,1} & S_{2,2} & S_{3,1} & \dots & S_{l_{max},l_{max}} & C_{2,1} & C_{2,2} & C_{3,0} & \dots & C_{l_{max},l_{max}} \end{bmatrix} \quad , \tag{2.2}$$

The collection of all twelve vectors, one for each month, of a year y becomes a matrix as,

$$\mathbf{D}_y = \begin{bmatrix} S^1_{2,1} & S^1_{2,2} & S^1_{3,1} & \dots & S^1_{l_{max},l_{max}} & C^1_{2,1} & C^1_{2,2} & C^1_{3,0} & \dots & C^1_{l_{max},l_{max}} \\ S^2_{2,1} & S^2_{2,2} & S^2_{3,1} & \dots & S^2_{l_{max},l_{max}} & C^2_{2,1} & C^2_{2,2} & C^2_{3,0} & \dots & C^2_{l_{max},l_{max}} \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & \vdots & & \vdots \\ S^{mon}_{2,1} & S^{mon}_{2,2} & S^{mon}_{3,1} & \dots & S^{mon}_{l_{max},l_{max}} & C^{mon}_{2,1} & C^{mon}_{2,2} & C^{mon}_{3,0} & \dots & C^{mon}_{l_{max},l_{max}} \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & \vdots & & \vdots \\ S^{12}_{2,1} & S^{12}_{2,2} & S^{12}_{3,1} & \dots & S^{12}_{l_{max},l_{max}} & C^{12}_{2,1} & C^{12}_{2,2} & C^{12}_{3,0} & \dots & C^{12}_{l_{max},l_{max}} \end{bmatrix} , \tag{2.3}$$

where $mon = 1, 2, \dots, 12$, is the month number in a year. Now after replacing both notions, i.e. $S_{l,m}$ for sine coefficients and $C_{l,m}$ for cosine coefficients by d_ℓ in a way that d_ℓ, with ℓ from 1 to 4096 represents the sine coefficients and d_ℓ, with ℓ from 4097 to 8276 represents the cosine coefficients, then the simpler form of the dataset emerges as,

$$\mathbf{D}_y = \begin{bmatrix} d^1_{1,y} & d^1_{2,y} & d^1_{3,y} & \dots & d^1_{\ell,y} & \dots & d^1_{L,y} \\ d^2_{1,y} & d^2_{2,y} & d^2_{3,y} & \dots & d^2_{\ell,y} & \dots & d^2_{L,y} \\ \vdots & \vdots & \vdots & & \vdots & & \vdots \\ d^{mon}_{1,y} & d^{mon}_{2,y} & d^{mon}_{3,y} & \dots & d^{mon}_{\ell,y} & \dots & d^{mon}_{L,y} \\ \vdots & \vdots & \vdots & & \vdots & & \vdots \\ d^{12}_{1,y} & d^{12}_{2,y} & d^{12}_{3,y} & \dots & d^{12}_{\ell,y} & \dots & d^{12}_{L,y} \end{bmatrix} , \tag{2.4}$$

where $\ell = 1, 2, \dots, L$, is the running coefficient number and L is the total number of the coefficients, given in (2.1). A monthly data vector $\mathbf{d}^{mon}_y$ for mon^{th} from data matrix (2.4) can be written as,

$$\mathbf{d}^{mon}_y = \begin{bmatrix} d^{mon}_{1,y} & d^{mon}_{2,y} & d^{mon}_{3,y} & \dots & d^{mon}_{\ell,y} \dots & d^{mon}_{L,y} \end{bmatrix} \quad . \tag{2.5}$$

2.2.2 LONG TERM MEAN REMOVAL

To get temporal variations, a static gravity field based on a long time period should be subtracted from the monthly solutions. The static field should consist of an integer number of years to avoid seasonal biases (Meyer et al., 2012). Therefore, the static field computed here consists of a 16-year data span starting from April 2002 to Jun 2017. Firstly the mean of all the months $\boldsymbol{\mu}^{mon}$ have been computed, with $mon = 1, 2, \ldots, 12$, for January, February up to December, respectively, followed by taking the mean of the monthly means $\boldsymbol{\mu}^{mon}$ which serve as the long term mean or the static field $\boldsymbol{\mu}_s$.

The following equation represents the mean of mon^{th} month of all years y from 2002 to 2017, with $mon = 1, 2, \ldots, 12$, for January, February up to December, respectively.

$$\boldsymbol{\mu}^{mon} = \frac{1}{M}\sum_{y=1}^{M}\mathbf{d}_y^{mon} = \frac{1}{M}\left[\sum_{y=1}^{M}d_{1,y}^{mon} \quad \sum_{y=1}^{M}d_{2,y}^{mon} \quad \sum_{y=1}^{M}d_{3,y}^{mon} \quad \ldots \quad \sum_{y=1}^{M}d_{\ell,y}^{mon} \ldots \quad \sum_{y=1}^{M}d_{L,y}^{mon}\right] \quad , \tag{2.6}$$

or

$$\boldsymbol{\mu}^{mon} = \left[\mu_1^{mon} \quad \mu_2^{mon} \quad \mu_3^{mon} \quad \ldots \quad \mu_\ell^{mon} \ldots \quad \mu_L^{mon}\right] \quad , \tag{2.7}$$

For each month , y ranges from 1 to M, where, only in this chapter, M denotes the total number of monthly dataset available for a certain month. From Table 2.1 M can be counted as,

$M =$ [14 13 14 15 15 12 13 14 13 12 14 14] for
$mon =$ [Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec], respectively.

The additive mean of these 12 monthly means represents the long term mean or the static field $\boldsymbol{\mu}_s$, such as,

$$\boldsymbol{\mu}_s = \frac{1}{12}\sum_{mon=1}^{12}\boldsymbol{\mu}^{mon} = \frac{1}{12}\left[\sum_{mon=1}^{12}\mu_1^{mon} \sum_{mon=1}^{12}\mu_2^{mon} \sum_{mon=1}^{12}\mu_3^{mon} \ldots \sum_{mon=1}^{12}\mu_\ell^{mon} \ldots \sum_{mon=1}^{12}\mu_L^{mon}\right] , \tag{2.8}$$

or

$$\boldsymbol{\mu}_s = \left[\mu_1 \quad \mu_2 \quad \mu_3 \quad \ldots \quad \mu_\ell \ldots \quad \mu_L\right] \quad . \tag{2.9}$$

Since the long term mean or the *static field* possesses the non-variational part of the gravity field its removal from the GRACE SH coefficients yields the *variational level coefficients* as,

$$\mathbf{p}_y^{mon} = \mathbf{d}_y^{mon} - \boldsymbol{\mu}_s = \left[p_{1,y}^{mon} \quad p_{2,y}^{mon} \quad p_{3,y}^{mon} \quad \ldots \quad p_{\ell,y}^{mon} \ldots \quad p_{L,y}^{mon}\right] \quad . \tag{2.10}$$

Now for a year y, with $mon = 1, 2, \ldots, 12$, for January, February up to December, respectively, the data matrix $\mathbf{P}_y$ is represented as,

$$\mathbf{P}_y = \begin{bmatrix} p^1_{1,y} & p^1_{2,y} & p^1_{3,y} & \cdots & p^1_{\ell,y} & \cdots & p^1_{L,y} \\ p^2_{1,y} & p^2_{2,y} & p^2_{3,y} & \cdots & p^2_{\ell,y} & \cdots & p^2_{L,y} \\ \vdots & \vdots & \vdots & & \vdots & & \vdots \\ p^{mon}_{1,y} & p^{mon}_{2,y} & p^{mon}_{3,y} & \cdots & p^{mon}_{\ell,y} & \cdots & p^{mon}_{L,y} \\ \vdots & \vdots & \vdots & & \vdots & & \vdots \\ p^N_{1,y} & p^N_{2,y} & p^N_{3,y} & \cdots & p^N_{\ell,y} & \cdots & p^N_{L,y} \end{bmatrix} , \tag{2.11}$$

and $\boldsymbol{\mu}_y$ represents the mean of the variational level monthly solutions for a year y as,

$$\boldsymbol{\mu}_y = \frac{1}{12}\sum_{mon=1}^{12} \mathbf{p}^{mon}_y = \frac{1}{12}\left[\sum_{mon=1}^{12} p^{mon}_{1,y} \sum_{mon=1}^{12} p^{mon}_{2,y} \sum_{mon=1}^{12} p^{mon}_{3,y} \cdots \sum_{mon=1}^{12} p^{mon}_{\ell,y} \cdots \sum_{mon=1}^{12} p^{mon}_{L,y}\right] , \tag{2.12}$$

or,

$$\boldsymbol{\mu}_y = \begin{bmatrix} \mu_{1,y} & \mu_{2,y} & \mu_{3,y} & \cdots & \mu_{\ell,y} \cdots & \mu_{L,y} \end{bmatrix} , \tag{2.13}$$

and the variance of each coefficients is,

$$\sigma^2_{\ell,y} = \frac{\sum_{mon=1}^{12}\left(p^n_{\ell,y} - \mu_{\ell,y}\right)^2}{12-1} , \tag{2.14}$$

therefore the variance vector of year y is,

$$\boldsymbol{\sigma}^2_y = \begin{bmatrix} \sigma^2_{1,y} & \sigma^2_{2,y} & \sigma^2_{3,y} & \cdots & \sigma^2_{\ell,y} \cdots & \sigma^2_{L,y} \end{bmatrix} . \tag{2.15}$$

Next chapter starts with the unsupervised classification of the monthly variational level datasets $\mathbf{p}^{mon}_y$.

UNSUPERVISED CLASSIFICATION 3

Every month a set of SH coefficients up to degree and order 90 is recovered by GFZ using range-rates. For this study, the dataset from April 2002 to June 2017 has been analyzed. Due to yearly mass redistribution, the time-varying part of the gravity field shows significant fluctuations. These fluctuations can be seen as the varying magnitudes of the SH coefficients. However, not all the SH coefficients show variations. The difference in behavior of SH coefficients is under investigation in this study. The aim is to separate two groups of the SH coefficients using different and independent techniques. The decision of the boundary between the two classes depends upon the clustering or classification methods. Respective sections describe the decision rules in details. In this chapter two unsupervised classification techniques, i.e. k-means clustering and threshold classification have been utilized to find the groups in the GRACE SH dataset.

Clustering and classification are among the primary analysis techniques a human brain uses for analysis. For instance, the grouping of things as living or nonliving is an example of clustering and assigning mammals class to whales is an example of classification. Clustering plays a vital role in every field of life. For example, there are many web pages on the internet, and for a particular search query, it may return a thousand pages. The grouping of the pages according to their contents can make the search algorithms effective and efficient. Eventuality smaller clusters result in more specific outcomes of the query. Similarly, the periodic table of elements where the elements are arranged in the order of their atomic numbers and the arrangement of books in a library according to their subject, are specific applications of clustering.

Good clusters have high intra-class similarity (Madigan, 2017), where the similarity is expressed a distance function, which is typically metric. There are many types of clustering. Most common are partitioning, hierarchical and density-based clustering. In this chapter a partitioning clustering technique, known as k-means clustering has been used to analyze the GRACE dataset. The details are given in the following section.

3.1 k-MEANS CLUSTERING

k-means clustering is an iterative unsupervised classification technique allows forming groups in the feature dataset before defining and labeling them, where k is the user-specified number of clusters representing centroid of clusters. Once k is fixed, discussed in Section 3.3, each feature or data point is then assigned to a closest centroid. The collection of data points assigned to a centroid is a cluster. The centroid is recomputed once the assignment is complete. The procedure of assigning the data points is repeated to redefine the clusters. The procedure continues until points does not change their cluster or the centroid remains the same. The proximity of the data point from the centroid is measured by L_2 distance in the Euclidean space. The quality of the clustering is measured as the *sum of squares error* (SSE) (Tan et al., 2005) as,

$$\text{SSE} = \sum_{i}^{k} \sum_{\mathbf{x} \in C_i} dist(\mathbf{c}_i, \mathbf{x})^2 \tag{3.1}$$

where, $\mathbf{x}$ is the data point, only in this chapter, C_i is the i^{th} cluster, $\mathbf{c}_i$ is the centroid of the cluster C_i, and k is the number of clusters. If the number of the data points in the i^{th} cluster are m_i, then its centroid $\mathbf{c}_i$ is defined by,

$$\mathbf{c}_i = \frac{1}{m_i} \sum_{\mathbf{x} \in C_i} \mathbf{x} \tag{3.2}$$

The major issues of the k-means clustering include choosing the number of centroids k and their initialization. The number of centroids can be specified by *elbow rule*, discussed in Section 3.3 and they are initialized randomly. The different sets of initialization results in different SSE minimization. It is possible that several runs of k-means clustering are not able to produce minimum SSE because of miss placed initial centroids. One way of selecting the initial centroid is to randomly select the first centroid and then for each next centroid select a point that is farthest from any of the initial centroids.

3.2 FEATURE DATASET

A data matrix $\mathbf{P}_y$ (c.f. 2.11) consists of rows equal to the number of monthly data available for a specific year y from 2002 to 2017, of GRACE monthly SH coefficients has been used as the feature dataset. Next section discusses the results using this data matrix. The $\mathbf{P}_y$ is given here for reference as,

$$\mathbf{P}_y = \begin{bmatrix} p^1_{1,y} & p^1_{2,y} & p^1_{3,y} & \cdots & p^1_{\ell,y} & \cdots & p^1_{L,y} \\ p^2_{1,y} & p^2_{2,y} & p^2_{3,y} & \cdots & p^2_{\ell,y} & \cdots & p^2_{L,y} \\ \vdots & \vdots & \vdots & & \vdots & & \vdots \\ p^{mon}_{1,y} & p^{mon}_{2,y} & p^{mon}_{3,y} & \cdots & p^{mon}_{\ell,y} & \cdots & p^{mon}_{L,y} \\ \vdots & \vdots & \vdots & & \vdots & & \vdots \\ p^N_{1,y} & p^N_{2,y} & p^N_{3,y} & \cdots & p^N_{\ell,y} & \cdots & p^N_{L,y} \end{bmatrix},$$

3.3 RESULTS

Deciding the number of k is the starting point for the k-means clustering. After an iterative process of clustering with different values of k, ranges from 1 to 20, the SSE is plotted against several values of k for all years. It is observed from the curve, given in Figure 3.1 that the error decreases as the k increases, sharply, in the beginning, represent *elbow rule*, because as the k increases the size of the cluster decreases and therefore the error within each cluster decreases. The optimal value of

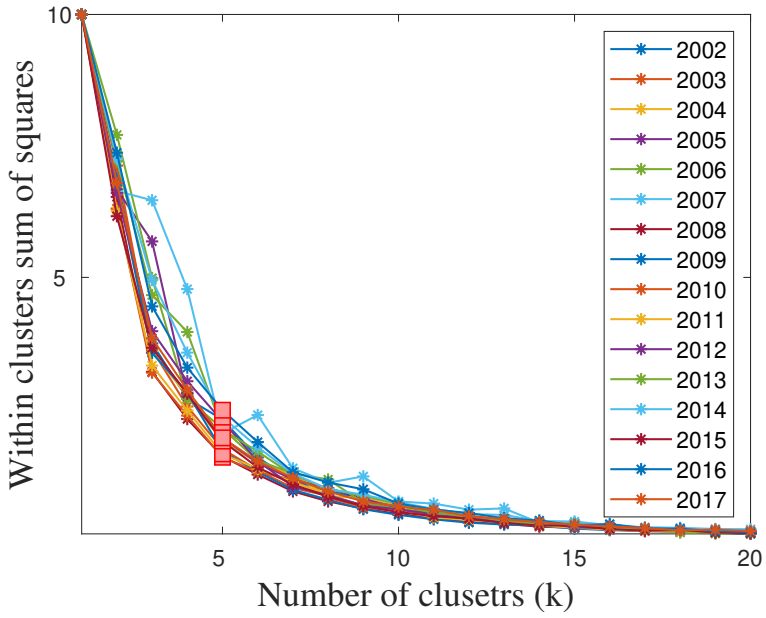
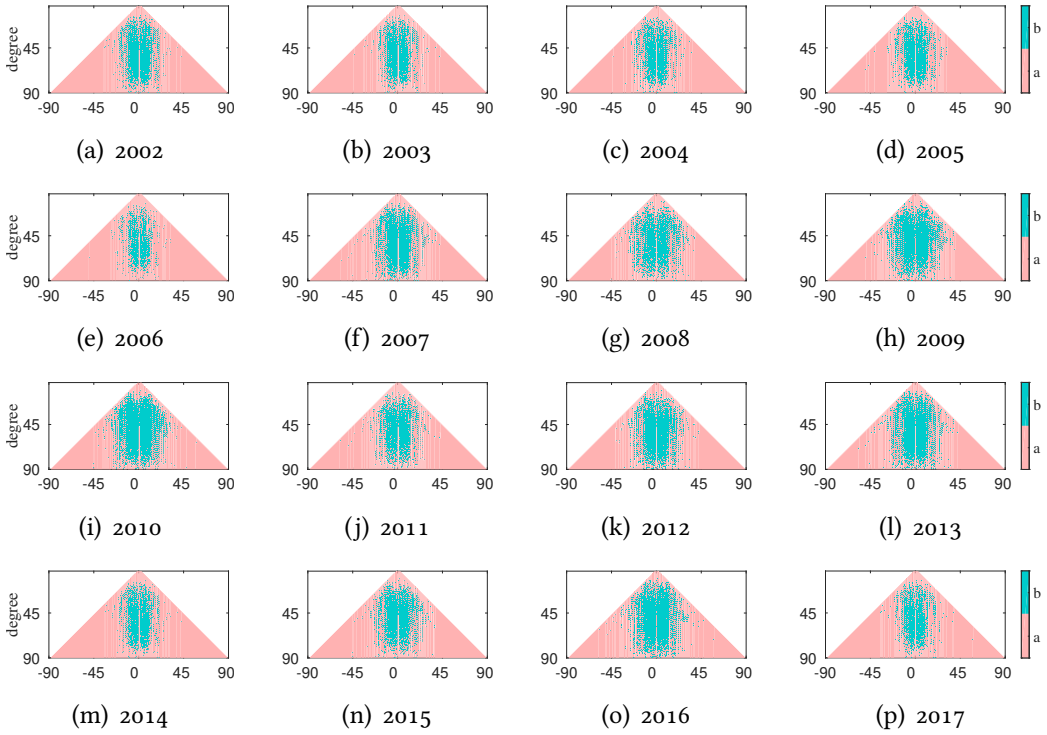


Figure 3.1: SSE minimization for years from 2002 to 2017. The optimal value of k is 5, since the error does not decrease afterwords significantly.

k is the point after which adding another cluster does not improve the model. Therefore $k = 5$ is selected for clustering. As $k = 5$ give five clusters, smaller clusters are merged in class a. Therefore, two clusters can be seen in the results. The results are given in Figure 3.2 for each year separately. It is observed from Sub-figures 3.2(a) to 3.2(p) that the number of SH coefficients in each cluster

changes. The reason lies in the changing quality of the gravity recovery products due to changing orbital track coverage of GRACE satellites.



(a) 2002 (b) 2003 (c) 2004 (d) 2005

(e) 2006 (f) 2007 (g) 2008 (h) 2009

(i) 2010 (j) 2011 (k) 2012 (l) 2013

(m) 2014 (n) 2015 (o) 2016 (p) 2017

Figure 3.2: k-means clustering identifies two clusters of different sizes each year.

Clustering is also used as the initial processing for other analysis techniques. In this chapter k-means provides the initial information about the natural partitioning exists in the dataset. Two distinct groups have been identified on the bases of distance between the data points and the centroids in the feature space. A higher density of data points shows that the points have similar data value and therefore stay closer to each other, whereas, the second group which has less density therefore needs bigger space in feature space. The obvious reasons behind the density difference of the two clusters are their variances. To verify this, in the next section *threshold classification* is used to treat the dataset, where the data will be classified on the bases of their yearly variance. An independent, unsupervised classification technique will help us to confirm the results of k-means clustering. Moreover, we will be able to label the clusters with meaning full names which exhibit their behavior.

3.4 CLASSIFICATION USING THRESHOLD

Threshold classification is used as the second unsupervised classification method. In the previous section, k-means clustering bifurcates the data into two groups on the bases of their density in the feature space. In this section *threshold classification* will be used to analyze the coefficients on the bases of their yearly variance. It is a simple way of data segregation, mostly used in image segmentation. According to Morse (2000), if $g(x)$ is a threshold version of $f(x)$ at some global threshold T then

$$g(x) = \begin{cases} 1 & \text{if } f(x) \geq T \\ 0 & \text{otherwise} \end{cases} . \tag{3.3}$$

In case if there exist two clear groups in a dataset, its histogram will be bimodal, and the threshold can be a value at the bottom of the valley (Kapur and Wong, 1985). In case of unclear separation of the modes, the threshold selection is a difficult task (Prasad et al., 2011). To resolve the issue, Otsu (1979) and Kapur and Wong (1985) proposed several methods. Sezgin and Sankur (2004) and Patra et al. (2011) present comprehensive reviews of several techniques.

The principle of threshold classification is to segregate SH coefficients on the bases of their yearly variance. For each year we have twelve datasets of monthly variational level SH coefficients, one for each month, consider variational level SH coefficients $p^{mon}_{\ell,y}$, with $y = 1, 2, 3, \ldots, 16$ for the years from 2002 to 2017, $mon = 1, 2, \ldots, 12$, for January, February up to December, respectively and $\ell = 1, 2, 3, \ldots, 8276$, coefficients, whereas the vector of monthly coefficients, as given in (2.10) as,

$$\mathbf{p}^{mon}_{y} = \begin{bmatrix} p^{mon}_{1,y} & p^{mon}_{2,y} & p^{mon}_{3,y} & \ldots & p^{mon}_{\ell,y} & \ldots & p^{mon}_{L,y} \end{bmatrix} .$$

During a year some of the coefficients vary a lot whereas some of them very minutely, as shown in Figure 3.3. Computing their variance $\sigma^2_{\ell,y}$ using (3.7) gives the minimum $min(\boldsymbol{\sigma}^2_y)$ and the max-

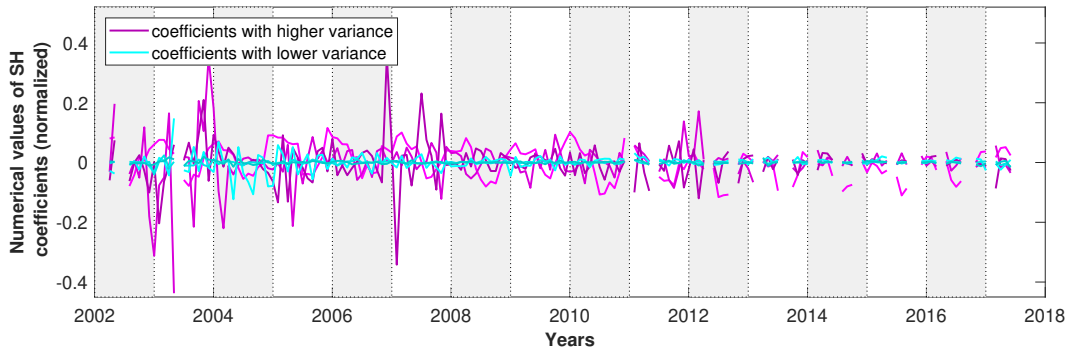


Figure 3.3: Time series of SH coefficients having different variance.

imum variance $max(\boldsymbol{\sigma}^2_y)$ limits. The maximum variance, $max(\boldsymbol{\sigma}^2_y)$, selected as the threshold T^{mon}_y,

can segregate the whole data into two classes. Coefficients with variance more than the threshold belong to the essential class whereas coefficients with variance less than the threshold belong to the nonessential class. The following points present the process of threshold classification for a monthly dataset of a specific year in precise steps.

1. load the variational level coefficients $p_{\ell,y}^{mon}$ of a year y for all months $mon = 1, 2, \ldots, 12$, and all coefficients $\ell = 1, 2, 3, \ldots, 8276$,
2. compute mean $\mu_{\ell,y}$ and variance $\sigma_{\ell,y}^2$ of each coefficient ℓ,

$$\boldsymbol{\mu}_y = \begin{bmatrix} \mu_{1,y} & \mu_{2,y} & \mu_{3,y} & \cdots & \mu_{\ell,y} \cdots & \mu_{L,y} \end{bmatrix} \quad , \tag{3.4}$$

$$\boldsymbol{\sigma}_y^2 = \begin{bmatrix} \sigma_{1,y}^2 & \sigma_{2,y}^2 & \sigma_{3,y}^2 & \cdots & \sigma_{\ell,y}^2 \cdots & \sigma_{L,y}^2 \end{bmatrix} \quad , \tag{3.5}$$

   where,

$$\mu_{\ell,y} = \frac{1}{N} \sum_{mon=1}^{N} p_{\ell,y}^{mon} \quad , \tag{3.6}$$

$$\sigma_{\ell,y}^2 = \frac{\sum_{mon=1}^{N} \left(p_{\ell,y}^{mon} - \mu_{\ell,y} \right)^2}{N-1} \quad . \tag{3.7}$$

   where, N is the number of months in the year.
3. note the minimum variance, $min(\boldsymbol{\sigma}_y^2)$,
4. set $max(\boldsymbol{\sigma}_y^2)$ as the initial thresholds T_y^{mon} for each mon,
5. compare T_y^{mon} with $\mu_{\ell,y}$ for each coefficient ℓ in (3.5) and classify them as,

$$\tilde{p}_{\ell,y}^{mon} = \begin{cases} \text{essential}, & \text{if } \sigma_\ell^2 \geq T_y^{mon} \text{ and} \\ \text{nonessential}, & \text{if } \sigma_\ell^2 < T_y^{mon} \quad , \end{cases} \tag{3.8}$$

   where, $\tilde{p}_y^{mon}$, is a classified coefficient, and

   a. **essential** are those coefficients whose values change significantly during a year, due to information contents of gravity variation,
   b. **nonessential** are these coefficients whose values don't change significantly during a year.

6. to authenticate T_y^{mon} the yearly mean values from the vector $\boldsymbol{\mu}_y$ replaces the nonessential coefficients of the data vector $\mathbf{p}_y^{mon}$. Replacement modifies the dataset therefore new dataset gets the name, modified dataset $\mathbf{p}_y^{'mon}$, which is constituted as,

$$p_{y,\ell}^{'mon} = \begin{cases} \mu_{\ell,y}, & \text{if } p_{\ell,y}^{mon} \text{ is nonessential} \\ p_{\ell,y}^{mon}, & \text{else} \quad , \end{cases} \tag{3.9}$$

7. replacement introduces *omission error*, $\delta \mathbf{P}_y^{mon}$, which is computed as,

$$\delta \mathbf{p}_y^{mon} =: |\mathbf{p}_y^{mon} - \mathbf{p}_y^{'mon}| \quad . \tag{3.10}$$

   For optimal threshold, the $\delta \mathbf{p}_y^{mon}$ must be smaller than the formal error Σ_y^{mon}, which are also available on GRACE science team data server (f.c. Chapter 2).

8. compute EWH field of omission error $\mathbf{M}(\delta \mathbf{p}_y^{mon})$ and find its maximum $max(\mathbf{M}(\delta \mathbf{p}_y^{mon}))$,

9. compute EWH field of the formal error $\mathbf{M}(\Sigma_y^{mon})$ of and find its maximum $max(\mathbf{M}(\Sigma_y^{mon}))$, for EWH synthesis see Appendix A

10. compare the two maximum values and verify that $max(\mathbf{M}(\delta \mathbf{p}_y^{mon})) < max(\mathbf{M}(\Sigma_y^{mon}))$, if not, decrease the threshold and repeat the process from step *5.*

An iterative program checks the condition given in the step *10.* for the threshold selection procedure and reaches the optimal threshold values for each month. It is shown in Figure 3.4 that after the selection of optimal threshold value for every month, the maximum value of the omission error field is less than the maximum value of the formal error field. The optimal threshold values of each month are given in Appendix B.

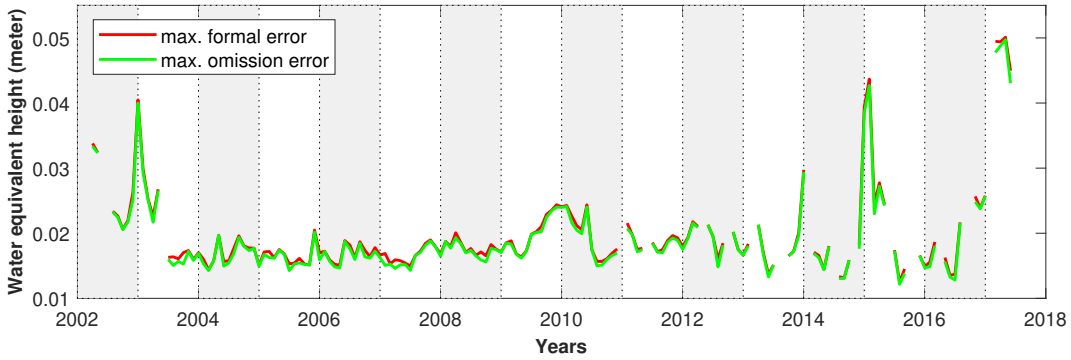


Figure 3.4: Formal error Σ_y^{mon} versus omission error $\delta \mathbf{P}_y^{mon}$ at filter cap 500 km. Omission error remains below the formal error.

3.5 RESULTS

GRACE monthly coefficients are classified into essential and nonessential classes using the threshold values given in Table B, Appendix B. Figure 3.5 shows the number of essential coefficients in each month for the period from April 2002 to June 2017. It is clear from the figure that the essential class have the most 7141 coefficients in May. 2003 and the least 6204 in Nov. 2008, or in other words there are 937 coefficients which change their class, called unclassified.

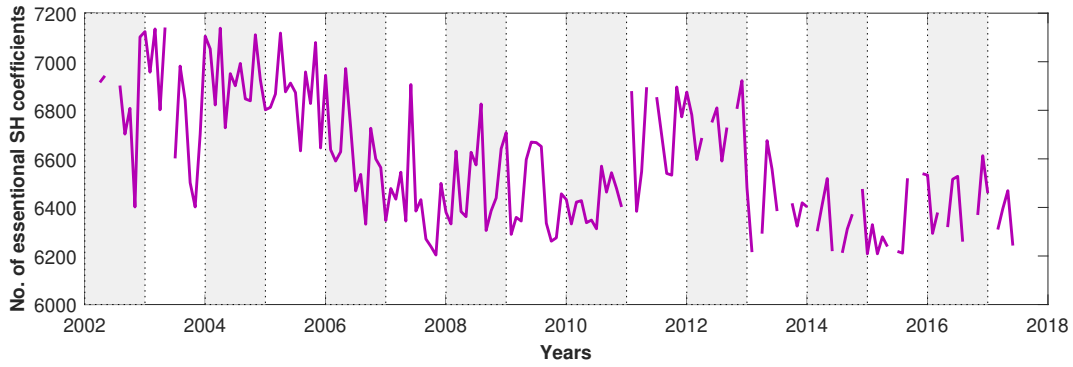
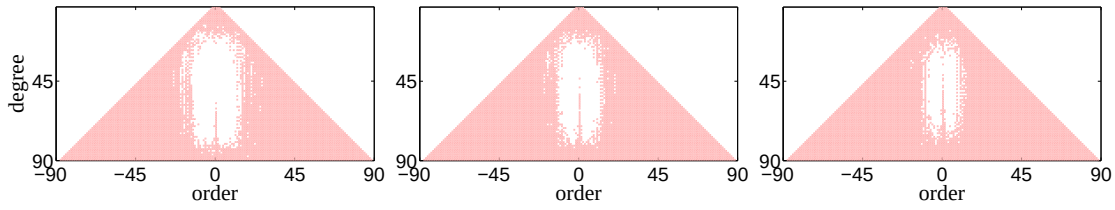


Figure 3.5: Number of essential coefficients in every month, from Apr. 2002 to Jun. 2017.

The plots in Figure 3.6 shows the essential coefficients in SC format for three different months, i.e. minimum to maximum, from left to right. It shows that a region around the zonal coefficients starting from the degree ~15 to ~75 of the order up to ~10 belong to the nonessential class, the coefficients with very minute information of the gravity variation. Rest of the coefficients around the nonessential class are essential coefficients, carrying the most information of the gravity variation.



(a) minimum (b) moderate (c) maximum

Figure 3.6: Essential class varying in size, from left to right, minimum to maximum.

Figure 3.7 shows the unclassified, essential and nonessential coefficients in SC format. Unclassified coefficients, in a year, belong to one group and in some other year belong to another group. The unclassified coefficients will be analyzed using the k nearest neighbor supervised technique in chapter 4. The coefficients in Clusters b. and c. does not change their clusters, whereas Figure 3.8 shows the number of coefficients 937, 6204 and 1136 in unclassified, essential and nonessential classes, respectively.

These results bring us to the end of the unsupervised classification, where k-means identifies the existence of groups in the dataset, threshold classification confirms the results and yearly variance of coefficients enables to label the classes on the base of their information contents. Meanwhile, a group of coefficients is also identified in the region between the two classes. These coef-

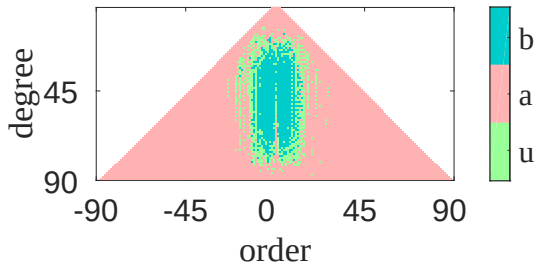


Figure 3.7: u) Unclassified, a) essential and b) nonessential classes in SC format

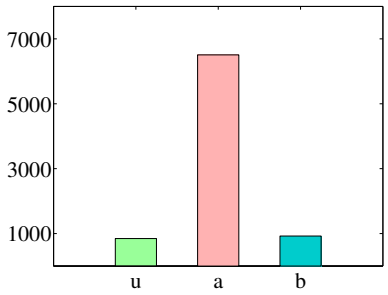


Figure 3.8: Number of coefficients, 937, 6204 and 1136 in u) Unclassified, a) essential and b) nonessential classes, respectively.

ficients change their classes when the classification results of different months are analyzed. In the next chapter, the k nearest neighbors (kNN) supervised classification method is discussed to bifurcate the unclassified coefficients into essential and nonessential classes.

SUPERVISED CLASSIFICATION 4

The objective of the classification process is to segregate the SH coefficients into two groups, i.e. essential and nonessential. In the previous chapter clustering and threshold classification are successfully utilized to analyze the behavior of the SH coefficients and to divide the coefficients into two groups. Meanwhile, we have found a group of SH coefficients changing their class, which means that during the analysis of one year they belong to one group and during the analysis of some other year they belong to the other group. This group is called as unclassified class. Essential and nonessential coefficients can be used to train the classifier to classify the unclassified coefficients too. This kind of classification is called *supervised classification*. In the following sections, supervised techniques are used to analyze the GRACE data. Followings are the supervised classification methods used in this study:

- ***k* nearest neighbor (*k*NN)** and
- **artificial neural networks (ANN).**

In this chapter, *k*NN segregates the unclassified coefficients into essential and nonessential classes and Chapter 6 using ANN.

4.1 *k* NEAREST NEIGHBOR (*k*NN)

If the classification problem belongs to a category where a sample of the classified dataset is available without any information about the distribution the best way to classify the candidate point is to look its neighborhood. This kind of problems belongs to the field of nonparametric statistics (Cover and Hart, 1967). It is reasonable to believe that the points which are close to each other belong to the same class. *k*NN has been successfully implemented for several classification tasks including pattern recognition, character recognition, and object and event recognition. Though *k*NN is very simple and efficient it has some limitation and disadvantages related to memory re-

quirements and complexities. Many techniques are developed to overcome these issues (Bhatia and Vandana, 2010).

To start the process, we need a training dataset which has several points related to each target class. Each candidate point is classified on the bases of its distance from all training points. The process starts by calculating the distance between all training points and the given candidate point, sorting the distance in increasing order and counting the number of majority class of the k nearest training points and finally giving the label of majority class to the candidate point. The algorithm as expressed by Kozma (2008), Mirkes (2011) and Sutto (2012) is given in the following list as,

a. a positive integer k is specified, along with a new sample,
b. select the k entries in the training database which are closest to the new sample,
c. find the majority or most common classification of these entries,
d. label the sample with the majority class.

The critical issue is to specify the number of k neighbors. To avoid the tie, k equals one more than the number of classes in the sample dataset is selected. In the following section, a brief description of the sample and target dataset is given; afterwards, in Section 4.3, results are presented.

4.2 DATASETS

To start with kNN, firstly, a training dataset is required which contains samples from the constituent classes, secondly the observation dataset which requires classification. Consider variational level SH coefficients $p_{\ell,y}^{mon}$, with $y = 1, 2, 3, \ldots, 16$ for the years from 2002 to 2017, $mon = 1, 2, \ldots, 12$, for January, February up to December, respectively and $\ell = 1, 2, 3, \ldots, 8276$, coefficients, whereas the vector of monthly coefficients, as given in (2.10), can be written as,

$$\mathbf{p}_y^{mon} = \begin{bmatrix} p_{1,y}^{mon} & p_{2,y}^{mon} & p_{3,y}^{mon} & \cdots & p_{\ell,y}^{mon} \cdots & p_{L,y}^{mon} \end{bmatrix} \quad .$$

The output of the threshold classification acts as the training dataset, but only those coefficients which belong to the essential and nonessential classes and never changed their groups, for instance, Figure 3.7, given here for quick reference, represents such SH coefficients. The essential and nonessential coefficients from April 2002 to June 2017 act as training dataset and used to classify the unclassified coefficients given in Figure 4.1.

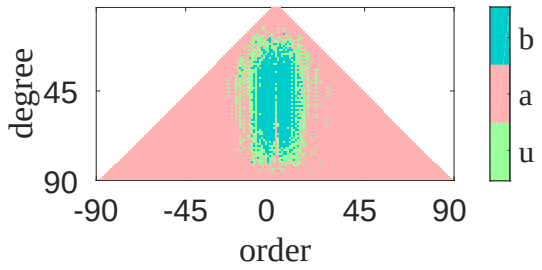


Figure 4.1: u) Unclassified, a) essential and b) nonessential classes in SC format

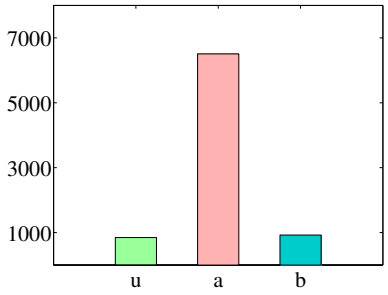


Figure 4.2: Number of coefficients, 937, 6204 and 1136 in u) Unclassified, a) essential and b) nonessential classes, respectively.

4.3 RESULTS

The essential and nonessential classes represented in Figure 4.1 acts as the source for the training process and the goal of the classification is to decide the association of the unclassified coefficients. Since the source dataset consists of two classes, therefore, $k = 3$. The number of k if set to odd helps to avoid the tie situation. Let us call the coefficients of the essential and nonessential classes as the training points and the unclassified coefficients as the candidate points. Then, the training points of 16 years GRACE data are stored in the memory and for each unclassified candidate coefficient, the distance between the candidate and the training points are measured. The training points are sorted on the bases of their distance from the candidate point. As the labels of the training points are also given therefor the candidate point will be classified as the majority class of the top three training points of the sorted list. The process is repeated for all the candidate points. At the end of the process, it is found that out of 937 unclassified points 286 points are close to the essential and 651 to the nonessential class, as shown in Figure 4.3. So the total number of essential and nonessential coefficients comes out to be 6490 and 1787, respectively, as given in Figure 4.4 and 4.5

These results bring us to the end of the first step of classifying the SH coefficients. The combinations of three independent algorithms help us to identify the two distinct classes in the dataset, i.e. essential and nonessential. The coefficients in essential class posses bigger place in the feature

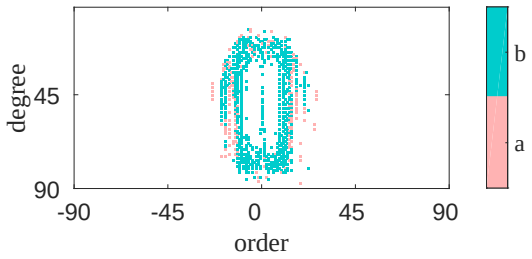


Figure 4.3: Classification of unclassified coefficients given in Figure 4.1 into a) essential and b) nonessential classes in SC format

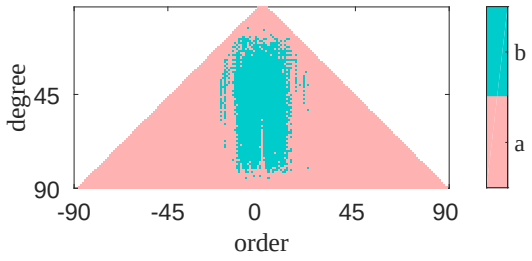


Figure 4.4: a) essential and b) nonessential classes in SC format

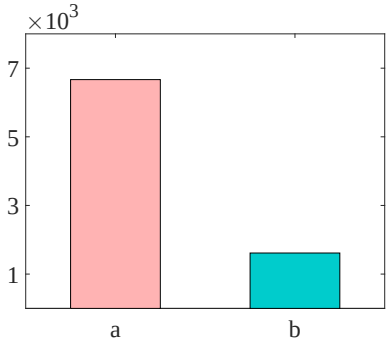


Figure 4.5: Number of coefficients, 6490 and 1787 in a) essential and b) nonessential classes, respectively.

space shows more yearly variance and therefore having the significant information of the gravity variation during the year. While the nonessential class is denser in feature space, therefore, possess a small place in the feature space, show less yearly variance and therefore having non-significant information of the gravity variation during the year. As the second step of machine learning, in the next chapter, another entirely different and independent machine learning technique, i.e. ANN, has been described along with its development. In the forthcoming chapters, ANN has been used as a *supervised classifier* and *prediction tool.*

ARTIFICIAL NEURAL NETWORKS (ANN) 5

A computer is a machine which works on instructions. Now, the demand includes that the computers must learn from the observations as humans do. This is the basic concept of machine learning. Since its invention, the computer is analyzing the data, which are collected by humans or captured by the sensors, using the known theoretical or mathematical models. Now there is a need to look at the data from another viewpoint. Hence, rather than theoretical or mathematical models, the data-driven analysis is in vogue. Human expects that the computer must look at the data in a way the humans have not looked at it and unveil the facts and hidden behavior or the relationships among its different parameters and finally make optimal decisions. ANN is one of the effective tools of machine learning. Web searching based on character, image or sound recognition, self driven vehicles using driver assistance systems, robots doing sorting, quality control, analyzing products and performing other importing jobs, face and object detection in cameras, text recognition form images and videos, online gaming, real time analysis of behavior, weather and natural hazard prediction, music composition, sound generation are some of its most common application fields. The salient feature of the ANN is learning. It learns from the historical data and improves its ability to do decisions. This study attempts to use the ANN for the *data mining* of the GRACE monthly datasets and uses it for two different purposes, i.e.

- **classification** and
- **prediction** of GRACE SH coefficients.

The objective of the study is to analyze the behavior of the GRACE monthly SH coefficients and find patterns, trends, classes or groups among them. In this chapter, the discussion starts from the introduction of ANN followed by its applications in the coming chapters for the classification and prediction of GRACE SH coefficients. The important steps towards the implementation of ANN are,

1. SOURCE AND TARGET DATASETS
2. NETWORK ARCHITECTURE

3. ACTIVATION FUNCTION
4. EVALUATION CRITERIA
5. FEEDFORWARD
6. OPTIMIZATION
7. ERROR BACKPROPAGATION

ANN is a data processing unit. The basic components of an ANN are *source dataset*, *target dataset* and a *network of neurons*. Requirements determine the size of the ANN, that is, the number of layers in the network and the number of neurons in each layer. The first is the input and the last is the output layer. Between the input and the output layers there exist inner layers. Except the neurons in the first layer, each neuron has a numerical value associated with it, named as *bias*, while the neurons in the first layer receive numerical values as input from the source dataset. Each neuron in a layer is connected to each neuron of the next layer. The connection of a neuron to the other neuron also has a numerical value, named as *weight*. The neurons, except the input layer, contain *activation functions*. During the learning process weights and biases get new values, due to which the output values changes. *Activation function* suppresses too small changes and ensures that only the essential information is passed on. All applications require a trained, tested and validated ANN. Source dataset is generally a large dataset and a big part ~60-70% of its trains, and two relatively small parts ~15-20% validates and ~15-20% tests the ANN, for instance. Target dataset is the required output and training, which is an iterative process, tunes the weights and biases of the network to produce an output like target dataset. The following section explains the components in details.

5.1 SOURCE AND TARGET DATASETS

In ANN, learning is a collection of three processes i.e. training, validation and test. During the training, an ANN using source data matrix learns how to reach closer to the target value (Goodfellow et al., 2016). In this chapter $\mathbf{P}$ with order $\kappa \times Q$ denotes the source data matrix with κ is the number of features in the data and q are the data samples, ranging from 1 to Q, while $\mathbf{T}$ with order $M \times Q$ represents the target vector with M targets correspond to each sample in the source data matrix.

$$\begin{aligned} \mathbf{P} &= \begin{bmatrix} \mathbf{p}_1 & \mathbf{p}_2 & \dots & \mathbf{p}_q & \dots & \mathbf{p}_Q \end{bmatrix}_{\kappa \times Q}, \\ \mathbf{T} &= \begin{bmatrix} \mathbf{t}_1 & \mathbf{t}_2 & \dots & \mathbf{t}_q & \dots & \mathbf{t}_Q \end{bmatrix}_{M \times Q}. \end{aligned} \tag{5.1}$$

where $\mathbf{p}_q$ and $\mathbf{t}_q$ are the input sample vectors and corresponding target vectors, respectively.

A cost minimization algorithm works on the difference of the output of the ANN and the corresponding target value, and iterates to bring the cost close to zero. Before presenting the cost function in detail, the following section describes the architecture of ANN.

5.2 FEEDFORWARD NETWORK ARCHITECTURE

Neurons are the basic data processing unit in the ANN. They are arranged in layers in an ANN. To express the data flow directions and the connections between the neurons in different layers of ANN, literature uses different terms such as topology (Stathakis, 2009), network, paradigm (Kaastra and Boyd, 1996), design, and architecture (Hill et al., 1994). ANN, on the bases of the data flow within the network and network topology, have many types such as, *recurrent neural networks* (RNN), *completely linked network* and *feedforward* neural network (Kriesel, 2007) (Hassoun, 1995). For this study *feedforward* architecture is used. Figure 5.1 represents an example of feedforward architecture with 5, 4, 1 neurons in the input, inner and output layers, respectively. In a feedforward network, neurons are arranged in layers in such a way that each neuron of a layer is connected to each neuron of the next layer. This means there are no loops in the network. Information is always fed forward, never fed back and the output of the activation functions always depends upon the input (Nielsen, 2015).

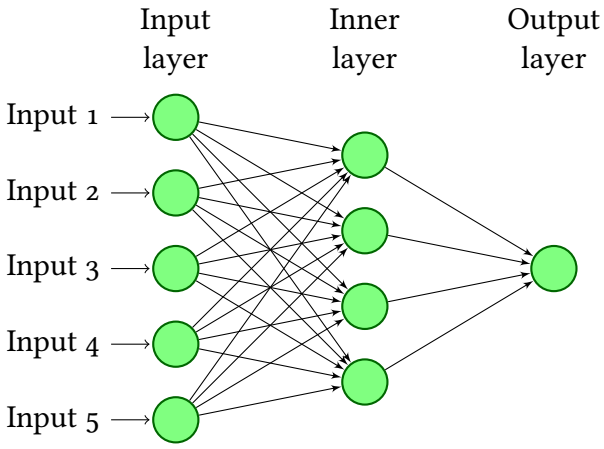


Figure 5.1: Simple feedforward ANN architecture. Neurons are arranged in layers.

The first layer in the ANN is the input layer and last is the output layer. Between input and output layers there exist inner layers. ANN without any inner layer is good to solve a linear regression model with linear activation function, for nonlinear regression models a multiple inner layers ANN is suggested (Kaastra and Boyd, 1996). l, in this chapter, denotes the layers of ANN. $l = 0$ stands

for the input layer, L represents the output layer and J^l denotes the number of neurons in a layer l. The output of a layer acts as the input of the next layer. Except for the first layer, each neuron in all other layers has an unknown variable called *bias*. b_j^l with j^{th} neuron of l^{th} layer, represents a bias. The number of neurons in the input layer is equal to the number of rows in the source dataset. The neurons in the input layer receive the source dataset. Each neuron in the first inner layer receives the output of each neuron of the input layer. Similarly, each neuron in the next layer receive the data from each neuron of the previous layer.

Data flows from one layer to the next layer through the connections between the neurons. Each connection has an unknown variable called *weight* and w_{jk}^l represents a weight of a connection from the k^{th} neuron in the $(l-1)^{\text{th}}$ layer to the j^{th} neuron in the l^{th} layer. Weights in matrix and biases in vector format for the inner layer of Figure 5.1 are shown in (5.2)

$$\mathbf{W}^l = \begin{Bmatrix} w_{1,1}^l & w_{1,2}^l & \dots & w_{1,\kappa}^l \\ w_{2,1}^l & w_{2,2}^l & \dots & w_{2,\kappa}^l \\ \vdots & \vdots & \vdots & \vdots \\ w_{j,1}^l & w_{j,2}^l & \dots & w_{j,\kappa}^l \end{Bmatrix}, \quad \mathbf{b}^l = \begin{Bmatrix} b_1^l \\ b_2^l \\ \vdots \\ b_j^l \end{Bmatrix} \quad . \tag{5.2}$$

If weights and biases from the above matrix are presented as the element x of a vector of unknowns $\mathbf{x}$ then $\mathbf{x}$ for the whole ANN can be written as,

$$\begin{aligned} \mathbf{x}^\top &= \begin{bmatrix} x_1 & x_2 & \dots & x_n \end{bmatrix} \\ &= \begin{bmatrix} w_{1,1}^1 & w_{1,2}^1 & \dots & w_{1,\kappa}^1 & \dots & w_{j,\kappa}^1 & b_1^1 & \dots & b_j^1 & w_{1,1}^2 & \dots & b_j^2 & \dots & w_{1,1}^L & \dots & b_j^L \end{bmatrix} , \end{aligned} \tag{5.3}$$

where total number of unknowns n in ANN is,

$$n = J^1(\kappa + 1) + J^2(J^1 + 1) + \ldots + J^l(J^{l-1} + 1) + \ldots + J^L(J^{L-1} + 1) \quad . \tag{5.4}$$

If a_j^l represents the input to a neuron then a_j^0 are the data values an ANN receives at the neurons in the input layer from the source data and $w_{jk}^l a_k^{l-1}$ are the inputs at the neurons in all layers $l > 1$. Now at a neuron, all the weighted inputs $w_{jk}^l a_k^{l-1}$ and the bias b_j^l are added together and the resultant z_j^l goes to the *activation function* f. (5.5) expresses the procedure and Section 5.3 express the activation function in detail.

$$a_j^l = f(z_j^l) \ , \text{ where } \ z_j^l = \sum_k w_{jk}^l a_k^{l-1} + b_j^l \quad , \tag{5.5}$$

and in matrix format,

$$\mathbf{a}^l = \mathbf{f}(\mathbf{z}^l) \ , \text{ where } \mathbf{z}^l = \mathbf{W}^l \mathbf{a}^{l-1} + \mathbf{b}^l \quad . \tag{5.6}$$

Consider Figure 5.1, a three-layer ANN with 5, 4 and 1 neurons in the input, inner and output layers, respectively. If matrix $\mathbf{W}^1$ of the order $(j, \kappa) = (4 \times 5)$ represents the weights connecting the 5 neurons of the input layer to the 4 of the inner layer, $\mathbf{W}^2$ of the order 1×4 connecting the 4 of the inner to the 1 neuron of output layer, $\mathbf{b}^1$ of the order 4×1 and $\mathbf{b}^2$ of the order 1×1 represents the vectors of biases for neurons of inner and output layer and $\mathbf{a}^0 = \{a_1 \;\; a_2 \;\; a_3 \;\; a_4 \;\; a_5\}^\top$ represents source data vector, the input layer receives, then the inner and the output layers i.e. $l = 1$ and $l = 2$, give the output vectors $\mathbf{a}^1$ and $\mathbf{a}^2$, respectively, as,

$$\mathbf{a}^1 = \mathbf{W}^1\mathbf{a}^0 + \mathbf{b}^1 = \begin{Bmatrix} w^1_{11} & w^1_{12} & w^1_{13} & w^1_{14} & w^1_{15} \\ w^1_{21} & w^1_{22} & w^1_{23} & w^1_{24} & w^1_{25} \\ w^1_{31} & w^1_{32} & w^1_{33} & w^1_{34} & w^1_{35} \\ w^1_{41} & w^1_{42} & w^1_{43} & w^1_{44} & w^1_{45} \end{Bmatrix} \begin{Bmatrix} a^0_1 \\ a^0_2 \\ a^0_3 \\ a^0_4 \\ a^0_5 \end{Bmatrix} + \begin{Bmatrix} b^1_1 \\ b^1_2 \\ b^1_3 \\ b^1_4 \end{Bmatrix} = \begin{Bmatrix} a^1_1 \\ a^1_2 \\ a^1_3 \\ a^1_4 \end{Bmatrix} , \tag{5.7}$$

$$\mathbf{a}^2 = \mathbf{W}^2\mathbf{a}^1 + \mathbf{b}^2 = \begin{Bmatrix} w^2_{11} & w^2_{12} & w^2_{13} & w^2_{14} \end{Bmatrix} \begin{Bmatrix} a^1_1 \\ a^1_2 \\ a^1_3 \\ a^1_4 \end{Bmatrix} + \begin{Bmatrix} b^2_1 \end{Bmatrix} = \begin{Bmatrix} a^2_1 \end{Bmatrix} . \tag{5.8}$$

and Figure 5.2 expresses the feedforward mechanism for the output layer explicitly.

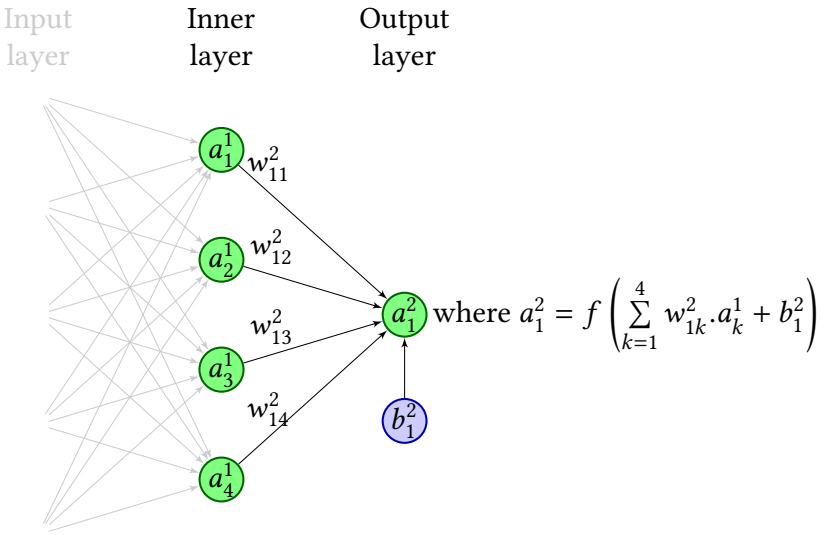


Figure 5.2: Feedforward mechanism in the output layer of an ANN with 4 neurons in inner and 1 in output layer.

The number of neurons in the input layer is equal to the number of features κ in the source data or rows of the source data matrix and the number of neuron in the output layer, for classification,

is equal to the number of classes in the target data and for prediction equals to one, for instance. Now the question is, how to decide the number of the inner layers and their neurons?

The decision usually based on trial and error, heuristics and some time include the *pruning*. Pruning is the method of ignoring the insignificant neurons during the training process (Stathakis, 2009). While discussing the required number of inner layers, McCullagh and Nelder (1989) suggest that ANN does not need any inner layer in a wide variety of linear and simple nonlinear applications. For complex nonlinear applications, Bishop (1995) suggests and discusses in detail the ANN with one and two inner layers. Özkan and Erbek (2003) report that the cost minimization or convergence is better in the ANN with one inner layer than the ANN with two inner layers. Hecht-Nielsen (1990) and Kaastra and Boyd (1996) advocate that the ANN with one inner layer can approximate any continuous function.

White (1992) argued that ANN are similar to linear and non-linear least squares regression and can be viewed as an alternative statistical approach to solving the least squares problem. Both ANN and conventional regression analysis attempt to minimize the sum of squared errors. Linear regression models may be viewed as a feedforward neural network with no hidden layers and one output neuron with a linear transfer function and the number of input neurons is equal to the number of independent variables while the output neuron(s) represent the dependent variable(s). The weights connecting the input neurons to the single output neuron are analogous to the coefficients in a linear least squares regression. Networks with one hidden layer resemble nonlinear regression models. The weights represent regression curve parameters.

During the experiment, it has been observed that ANN with one inner layer works perfectly for the classification as well as prediction while increasing the number of inner layers does not improve the results.

For finding the number of neurons, Hagan et al. (2014) states that if the number of neurons is too large, the network will *overfit* the training data. Overfit means the network memorize the data in the training set and fail to generalize to new situations. Kaastra and Boyd (1996) states that the greater the number of weights relative to the size of the training set, the greater the ability of the network to memorize idiosyncrasies of individual observation. Huang (2003) proposes the following for the number of neurons in inner layers N_h with,

Q is number of input samples from the source data

κ is number of neuron in the input layer.

M is number of targets.

$$N_h = 2\sqrt{(M+2)Q} \ , \tag{5.9}$$

Stathakis (2009), however, says that the number of neurons suggested by (5.9) causes overfit. Another suggestion, by Hagan et al. (2014), is,

$$N_h = \frac{Q}{(\alpha \times (\kappa + M))} \ , \tag{5.10}$$

with, α, an arbitrary scaling factor, usually 2-5, controls the overfitting. But, before the final decision, it is important to consider another fact about the relationship between the number of unknown variables n in a system and the number of samples in the source data. To determine n, consider (5.3), such as, for the network in Figure 5.1 where the number of inputs i.e $\kappa = 5$, the number of neuron in inner and output layers are $J^1 = 4$ and $J^2 = 1$, therefore $n = J^1(\kappa + 1) + J^2(J^1 + 1) = 4\times(5+1)+1\times(4+1) = 29$. In order to arrive at a regular linear system of equations we must have more than 29 data samples to solve the 29 unknowns, otherwise, if the number of the observation equations is less then the number of unknowns, we have an underdetermined system. We find a regular inverse only if the number of data samples is larger than the unknowns, an overdetermined system of equations (Grafarend and Awange, 2013). Therefore more the number of samples better would be the results. In our case where the source data size is very small, we have to be very careful and stay within the limits of the data size to set the number of layers and number of neurons in an ANN. The specific number of neurons in the inner layer are stated in the respective sections (c.f. Section (6.2) for classification and Section 7.2 for prediction). The following section introduces the activation functions and describes their merits and demerits and afterward, Section 5.4 discusses the cost function in detail.

5.3 ACTIVATION FUNCTION

ANN is a data processing unit which makes a computer capable of doing decision on the base of observations. This is machine learning. An activation function limits the amplitude of the output of a neuron. It squashes the permissible amplitude range of the output signal to some finite value (Haykin, 2011). The activation function suppresses too small changes in $\mathbf{z}^l$. In this way, it makes sure that only strong changes in $\mathbf{z}^l$ influence the new weights w^l_{jk} and biases b^l_j. The learning process, deep down in the ANN environment, changes the numerical values of the biases

b_j^l and weights w_{jk}^l. An activation function ensures that after each learning event, a small change in the biases and weights bring a small change in the output (Nielsen, 2015).

$$\Delta z_j^L \approx \sum_k \frac{\partial z_j^L}{\partial w_{jk}} \Delta w_{jk} + \sum_k \frac{\partial z_j^L}{\partial b_k} \Delta b_k \ , \tag{5.11}$$

where L represents the output layer. Multiplications and additions, two basic arithmetic operations, are operating in the core of the ANN where neurons are arranged in multiple layers. In the absence of an effective activation function, the output of the ANN would blow up. It is the nature of the activation function which keeps the output of the ANN to the limits, usually between -1 and +1. To keep the source data in the same range, in the literature it is suggested to use normalization of the source data (Larose, 2004), (Ioffe and Szegedy, 2015). Walia (2017) stressed the importance of the activation function. According to him, without activation function, an ANN would simply be a linear regression model, which has limited power and does not perform well most of the times. Without activation function ANN would not be able to learn and model complicated kinds of data such as images, videos, audio, speech etc. The list of possible activation functions is very long (Ramachandran et al., 2017), in the literature, however, on mathematical and empirical bases, the followings are discussed more than others:

- **linear function**
- **sigmoid function**

Sharma (2017) summarizes the advantages and disadvantages of these activation functions. The following sections discuss them one after the other.

5.3.1 LINEAR FUNCTION

A linear function is a straight line function where activation is proportional to the input. as,

$$a_j^l \propto z_j^l \ , \tag{5.12}$$

$$a_j^l = \text{const.} \times z_j^l \ . \tag{5.13}$$

It gives a range of activations, so it is not binary. Walia (2017), however, expresses his view about the limitations of the linear activation function. Firstly, a linear function is easy to solve but has limits to learn complicated functional mappings from data. Moreover, for *gradient descent*, a cost minimization algorithm (c.f. Section 5.6), uses the derivative of the activation function. The derivative of (5.13) with respect to z_j^l is constant. That means, the gradient has no relationship with input and descent is constant.

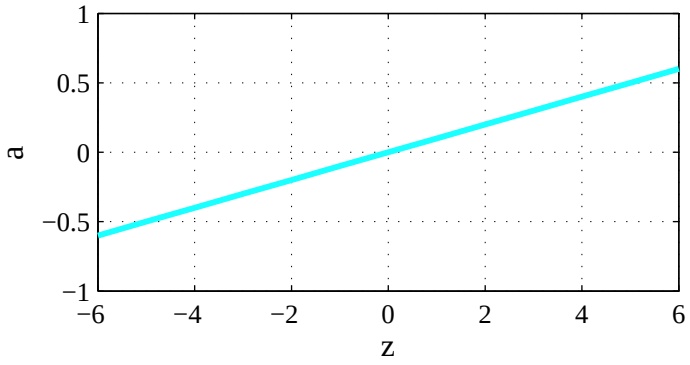


Figure 5.3: Linear function.

Secondly, ANN has connected layers. If each layer is activated by a linear function then it does not matter how many layers we have, a combination of linear functions in a linear manner is still another linear function. That means a single layer can replace all multiple layers. This way ANN lost the ability to stack layers. A nonlinear activation function can address this issue.

5.3.2 SIGMOID FUNCTION

A sigmoid or logistic function is an example of a nonlinear function as,

$$a_j^l = \frac{1}{1 + e^{-z_j^l}} \quad . \tag{5.14}$$

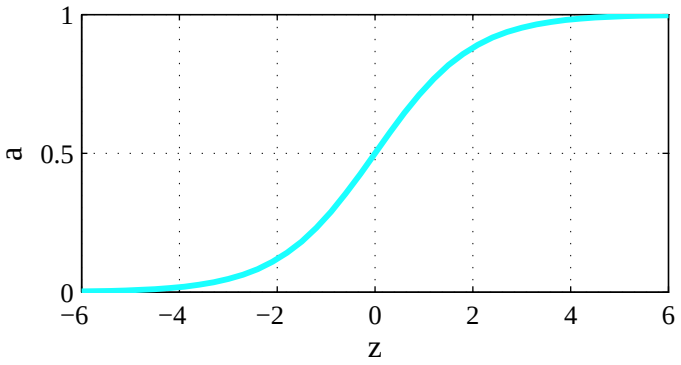


Figure 5.4: Sigmoid function.

It looks like a smooth step function. It is nonlinear in nature and has a smooth gradient. Combinations of this function are also nonlinear, in other words, it supports stacking layers. Moreover, z_j^l values range from -2 to 2, a_j^l values are very steep. Which means, any small changes in the values of z_j^l in that region will cause values of a_j^l to change significantly, that means this function

has a tendency to bring the a_j^l values to either end of the curve. Furthermore, unlike the linear function, the output of the activation function is always going to be in range (0,1) compared to (-inf, inf) of the linear function. So, in such a case we find a blow-up of the activations. Sigmoid functions are one of the most widely used activation functions today. This study uses sigmoid as the activation function.

Learning is an iterative process, and after each iteration, it is required to evaluate the ability of the ANN to perform the required task. The best way of evaluation is to compare the output of the ANN with externally available target data. The next section discusses the evaluation process and its criteria in detail.

5.4 EVALUATION CRITERIA

A cost function evaluates the ANN learning, specifically during the training process. The objective of the training process is to minimize the cost. Usually, a *quadratic cost* or *estimated mean squared error* acts as the cost function (Kaastra and Boyd, 1996). If all unknown of an ANN i.e weights and biases are arranged in a vector $\mathbf{x}$ as given in (5.3) then for the set of a source data vector and its corresponding target vector $\{\mathbf{p}_n, \mathbf{t}_n\}$ with $\mathbf{a}$ is the output of the ANN and $\mathbf{t}$ is the corresponding target, the algorithm minimizes the mean square error based cost function as,

$$\mathbf{C}(\mathbf{x}) \approx E\left\langle \mathbf{e}|\mathbf{e}\right\rangle \quad , \quad \text{where} \quad \mathbf{e} = (\mathbf{t} - \mathbf{a}) \quad , \tag{5.15}$$

where E is the expected value. This type of training is called as the performance learning (Hagan et al., 2014), a cost $\approx$ 0 means that the output of the network is close to the target, hence the training has achieved the goal. On the other hand, a large value of cost shows a big difference between output and target. The aim of the training is to minimize the cost. In other words, the objective of the training process is to set the weights and biases which make the cost as small as possible (Nielsen, 2015). This is called as performance optimizations. The goal of the optimization is to find $\mathbf{x}$ that minimize $\mathbf{C}(\mathbf{x})$ with each iteration, i.e.

$$\mathbf{C}(\mathbf{x}_{i+1}) < \mathbf{C}(\mathbf{x}_i) \tag{5.16}$$

If we assume that $\mathbf{C}(\mathbf{x})$ is an analytical function and all of its derivatives exist then for the $(i+1)^{\text{th}}$ iteration a Taylor series expansion around the $\mathbf{x}_i$ can be written as

$$\mathbf{C}(\mathbf{x}_{i+1}) \approx \mathbf{C}(\mathbf{x}_i) + \nabla\mathbf{C}(\mathbf{x}_i)^\top\Big|_{\mathbf{x}=\mathbf{x}_i} \Delta\mathbf{x}_i + \frac{1}{2}\Delta\mathbf{x}_i^\top \nabla^2\mathbf{C}(\mathbf{x}_i)\Big|_{\mathbf{x}=\mathbf{x}_i} \Delta\mathbf{x}_i + \cdots \quad , \tag{5.17}$$

where $\nabla \mathbf{C}(\mathbf{x})$ is the gradient and is defined as,

$$\nabla \mathbf{C}(\mathbf{x}) = \mathbf{g} = \begin{bmatrix} \frac{\partial}{\partial x_1}\mathbf{C}(\mathbf{x}) & \frac{\partial}{\partial x_2}\mathbf{C}(\mathbf{x}) & \cdots & \frac{\partial}{\partial x_n}\mathbf{C}(\mathbf{x}) \end{bmatrix}^\top , \tag{5.18}$$

and $\nabla^2 \mathbf{C}(\mathbf{x})$ is the Hessian and is defined as,

$$\nabla^2 \mathbf{C}(\mathbf{x}) = \mathbf{H} = \begin{Bmatrix} \frac{\partial^2 \mathbf{C}}{\partial^2 x_1} & \frac{\partial^2 \mathbf{C}}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 \mathbf{C}}{\partial x_1 \partial x_n} \\ \frac{\partial^2 \mathbf{C}}{\partial x_2 \partial x_1} & \frac{\partial^2 \mathbf{C}}{\partial^2 x_2} & \cdots & \frac{\partial^2 \mathbf{C}}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 \mathbf{C}}{\partial x_n \partial x_1} & \frac{\partial^2 \mathbf{C}}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 \mathbf{C}}{\partial^2 x_n} \end{Bmatrix} . \tag{5.19}$$

The sufficient condition for the point $\mathbf{C}(\mathbf{x})_i$ to be a definite minimum point is,

$$\nabla \mathbf{C}(\mathbf{x}_i)^\top \Big|_{\mathbf{x}=\mathbf{x}_i} = 0, \quad \text{and} \quad \nabla^2 \mathbf{C}(\mathbf{x}_i)^\top \Big|_{\mathbf{x}=\mathbf{x}_i} = \text{positive definite.} \tag{5.20}$$

In the following sections, three different iterative algorithms are introduced with the objective to optimize the cost. They begin with the initial guess i.e. $\mathbf{x}_0$. If i denotes the iteration then the expression,

$$\mathbf{x}_{i+1} = \mathbf{x}_i - \eta_i \mathbf{g}_i, \quad \text{or} \tag{5.21}$$

$$\Delta \mathbf{x}_i = -\eta_i \mathbf{g}_i \tag{5.22}$$

expresses the iterative process of cost minimization, where η_i is the learning rate which defines the step size towards the minimum and $\mathbf{g}_i$ gives the search direction of the cost function. Three different methods

1. GRADIENT DESCENT (GD)
2. NEWTON'S METHOD
3. LEVENBERG - MARQUARDT (LM)

are introduced in the following sections. GD uses gradient vector and Newton's method uses Hessian to optimize $\mathbf{C}(\mathbf{x})$. GD is slow and Newton's method needs modifications to fulfill the sufficient condition of positive definite. LM is an improved version of both method, therefore, it is used in this study for optimization. The discussion starts, with the introduction of GD, and the Newton's method followed by the LM algorithm.

5.5 GRADIENT DESCENT (GD)

Optimization means that the cost $\mathbf{C}(\mathbf{x}_i)$ decreases with each iteration, as given in (5.16). To minimize the cost for the $(i+1)^{\text{th}}$ iteration with a small learning rate, consider the first order Taylor

series expansion, around the $\mathbf{x}_i$

$$\mathbf{C}(\mathbf{x}_{i+1}) \approx \mathbf{C}(\mathbf{x}_i + \Delta\mathbf{x}_i) \approx \mathbf{C}(\mathbf{x}_i) + \mathbf{g}_i^\top \Delta\mathbf{x}_i \quad , \tag{5.23}$$

where gradient $\mathbf{g}_i \approx \nabla\mathbf{C}(\mathbf{x})\Big|_{\mathbf{x}=\mathbf{x}_i}$. By substituting the value of $\Delta\mathbf{x}$ from 5.22, we get,

$$\mathbf{C}(\mathbf{x}_{i+1}) \approx \mathbf{C}(\mathbf{x}_i) - \eta_i \mathbf{g}_i^\top \mathbf{g}_i \approx \mathbf{C}(\mathbf{x}_i) - \eta_i \left\|\mathbf{g}\right\|^2 \quad , \tag{5.24}$$

This shows that for every next iteration the cost decreases with positive and very small learning rate η_i. This method *converges very slowly* and η_i has great influence on the convergence behavior where η_i can be determined in two ways. Firstly, a constant value can be selected for all iterations. However, if it is very small, the method takes more iterations to converge. On the other hand, due to a large η_i, the algorithm becomes unstable and cost tends to increase instead of decreasing. Secondly, a dynamic learning rate can be used which changes after every iteration.

Slow convergence is the drawback of the GD method. It is recommended for a very big ANN since it stores only a gradient vector. Before moving forward to the other two optimization methods, the development of backpropagation (BP) algorithm for GD is presented, since the later algorithms are developed on its bases.

5.6 ERROR BACK PROPAGATION

The optimization of the ANN becomes more understandable when we consider them as the function appropriator. For example, in case of a linear function, a single layer ANN receives sample points and target data and compute the mean square error at the output neuron, as the error is a linear function of the weights, therefore its derivatives are simple to compute. However, for the multilayer ANN, the *error is not the linear function of the weights* and therefore its derivatives are not simple to compute and need optimization (Hagan et al., 2014). In this section, therefore a back propagation (BP) algorithm is developed, to handle the nonlinear functions, which back propagates the error sensitivities in case of GD. ANN trained by back prorogation is widely used in most of its applications after it is formally introduced by Rumelhart and McClelland (1986). Consider an ANN is provided with a sample dataset with corresponding target as,

$$\{\mathbf{p}_1, \mathbf{t}_1\}, \{\mathbf{p}_2, \mathbf{t}_2\}, \ldots, \{\mathbf{p}_Q, \mathbf{t}_Q\} \quad . \tag{5.25}$$

The algorithm for each input compares the output with the target data and adjust the vector of unknowns ANN parameters i.e weights and biases to minimize the cost. If $\mathbf{x}$ denotes the ANN unknown

parameters in a vector as given in (5.3), then the cost as the function of unknown parameters is written as,

$$\mathbf{C}(\mathbf{x}) \approx E\,\langle \mathbf{e}|\mathbf{e}\rangle \quad , \quad \text{where} \quad \mathbf{e} = (\mathbf{t} - \mathbf{a}) \quad , \tag{5.26}$$

where $\mathbf{a}$ is the output of the ANN and $\mathbf{t}$ is the corresponding target. BP is an iterative algorithm and cost is approximated by squares of error with each iteration, therefore, the expectation E is replaced by a squared error at iteration i

$$\hat{\mathbf{C}}(\mathbf{x}) \approx \left\langle \hat{\mathbf{e}}_i|\hat{\mathbf{e}}_i \right\rangle \quad , \quad \text{where} \quad \hat{\mathbf{e}}_i = \left(\mathbf{t}_i - \mathbf{a}_i\right) \quad , \tag{5.27}$$

Therefore from (5.24), we can write the equations for the *stochastic* GD or *incremental* GD as,

$$\begin{aligned} (w^l_{j,k})_{i+1} &= (w^l_{j,k})_i - \eta_i \frac{\partial \hat{\mathbf{C}}}{\partial w^l_{j,k}} \quad , \\ (b^l_j)_{i+1} &= (b^l_j)_i - \eta_i \frac{\partial \hat{\mathbf{C}}}{\partial b^l_j} \quad , \end{aligned} \tag{5.28}$$

where *stochastic* or *incremental* GD means that process encounter only one data sample with each iteration. This expression states the GD method for the development of the BP algorithm. To derive the partial derivatives of the cost function with respect to the vector of unknown parameters which is not a linear relation, a chain rule is introduced (Hagan et al., 2014). Consider (5.5), which shows that firstly the bias is added to the sum of weighted inputs, the resultant is called as z and then it goes to the activation function f, the output of which is compared with the target to compute error (c.f. section 5.2). In a similar way, we also relate the error to the initial product z^l_j and then z^l_j to the unknowns $w^l_{j,k}$ and b^l_j as,

$$\begin{aligned} \frac{\partial \hat{C}}{\partial w^l_{j,k}} &= \frac{\partial \hat{C}}{\partial z^l_j} \frac{\partial z^l_j}{\partial w^l_{j,k}} \quad , \\ \frac{\partial \hat{C}}{\partial b^l_j} &= \frac{\partial \hat{C}}{\partial z^l_j} \frac{\partial z^l_j}{\partial b^l_j} \quad . \end{aligned} \tag{5.29}$$

The derivative of the initial product z^l_j with respect to the unknowns $w^l_{j,k}$ and b^l_j is explicit as,

$$z^l_k = \sum_j w^l_{kj}\, a^{l-1}_j + b^l_k \quad . \tag{5.30}$$

and after taking the respective derivative,

$$\begin{aligned} \frac{\partial z^l_j}{\partial w^l_{j,k}} &= a^{l-1}_j \quad , \\ \frac{\partial z^l_j}{\partial b^l_j} &= 1 \quad . \end{aligned} \tag{5.31}$$

The derivative of the cost function with respect to the initial product z^l_j, however, get some new meanings. The expression,

$$\delta^l_j =: \frac{\partial \hat{C}}{\partial z^l_j} \quad , \tag{5.32}$$

defines the sensitivity of the cost function to the j^{th} element of the layer l. The (5.29) can now be rewritten in the simplified form as,

$$\begin{aligned} \frac{\partial \hat{C}}{\partial w_{j,k}^l} &= \delta_j^l a_j^{l-1} \quad , \\ \frac{\partial \hat{C}}{\partial b_j^l} &= \delta_j^l \quad , \end{aligned} \tag{5.33}$$

and hence, the expression for the approximated GD can be written as,

$$\begin{aligned} (w_{j,k}^l)_{i+1} &= (w_{j,k}^l)_i - \tfrac{\eta}{B} \delta_j^l a_j^{l-1} \quad , \\ (b_j^l)_{i+1} &= (b_j^l)_i - \tfrac{\eta}{B} \delta_j^l \quad , \end{aligned} \tag{5.34}$$

and in matrix format

$$\begin{aligned} (\mathbf{w}^l)_{i+1} = \quad & (\mathbf{w}^l)_i - \tfrac{\eta}{B} \boldsymbol{\delta}^l (\mathbf{a}^{l-1})^\top \quad , \\ (\mathbf{b}^l)_{i+1} = \quad & (\mathbf{b}^l)_i - \tfrac{\eta}{B} \boldsymbol{\delta}^l \quad , \end{aligned} \tag{5.35}$$

with B is the batch size and refer to the number of data sample used to update the vector of unknowns, discussed in Section 5.7, where $\boldsymbol{\delta}^l$ represents the sensitivity of the cost function due to all the elements J^l of the layer l,

$$\boldsymbol{\delta}^l \equiv \frac{\partial \hat{C}}{\partial \mathbf{z}^l} = \begin{bmatrix} \frac{\partial \hat{C}}{\partial z_1^l} & \frac{\partial \hat{C}}{\partial z_2^l} & \cdots \frac{\partial \hat{C}}{\partial z_{J^l}^l} \end{bmatrix}^\top \quad . \tag{5.36}$$

Now to solve the nonlinear relation of the derivative of the cost function with respect to the initial product z_j^l, chain rule is used again, therefore consider the relation,

$$\frac{\partial \hat{C}}{\partial \mathbf{z}^l} = \frac{\partial \hat{C}}{\partial \mathbf{z}^{l+1}} \frac{\partial \mathbf{z}^{l+1}}{\partial \mathbf{z}^l} \quad . \tag{5.37}$$

This means that the partial derivative of the cost function w.r.t initial product z_j^l is actually equal to two things, firstly, the derivative of the cost function with respect to the initial product of the proceeding layer i.e z_j^{l+1} and secondly, the partial derivative of the initial product of the proceeding layer i.e z_j^{l+1} w.r.t the initial product of the current layer i.e z_j^l, it is recursive relation, which leads us to the Jacobean,

$$\frac{\partial \mathbf{z}^{l+1}}{\partial \mathbf{z}^l} \equiv \begin{bmatrix} \frac{\partial z_1^{l+1}}{\partial z_1^l} & \frac{\partial z_1^{l+1}}{\partial z_2^l} & \cdots & \frac{\partial z_1^{l+1}}{\partial z_{J^l}^l} \\ \frac{\partial z_2^{l+1}}{\partial z_1^l} & \frac{\partial z_2^{l+1}}{\partial z_2^l} & \cdots & \frac{\partial z_2^{l+1}}{\partial z_{J^l}^l} \\ \vdots & \vdots & & \vdots \\ \frac{\partial z_{J^{l+1}}^{l+1}}{\partial z_1^l} & \frac{\partial z_{J^{l+1}}^{l+1}}{\partial z_2^l} & \cdots & \frac{\partial z_{J^{l+1}}^{l+1}}{\partial z_{J^l}^l} \end{bmatrix} \quad . \tag{5.38}$$

To get an expression for this matrix, consider its r, s element,

$$\begin{aligned} \frac{\partial z_r^{l+1}}{\partial z_s^l} &= \frac{\partial \left(\sum_{c=1}^{j} w_{r,c}^{l+1} a_c^l + b_r^{l+1} \right)}{\partial z_s^l} = w_{r,s}^{l+1} \frac{\partial a_s^l}{\partial z_s^l} \\ &= w_{r,s}^{l+1} \frac{\partial f^l(z_s^l)}{\partial z_s^l} = w_{r,s}^{l+1} \dot{f}^l(z_s^l) \quad , \end{aligned} \tag{5.39}$$

where

$$\dot{f}^l(z_s^l) = \frac{\partial f^l(z_s^l)}{\partial z_s^l} \quad . \tag{5.40}$$

and f^l is the *activation function*. Therefore the Jacobean matrix can be written as,

$$\frac{\partial \mathbf{z}^{l+1}}{\partial \mathbf{z}^l} = \mathbf{W}^{l+1}\dot{\mathbf{F}}^l(\mathbf{z}^l) \quad , \tag{5.41}$$

where,

$$\dot{\mathbf{F}}^l(\mathbf{z}^l) = \begin{bmatrix} \dot{f}^l(z_1^l) & 0 & \dots & 0 \\ 0 & \dot{f}^l(z_2^l) & \dots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \dots & \dot{f}^l(z_{J^l}^l) \end{bmatrix} \quad . \tag{5.42}$$

Now back to (5.37), and after inserting the value for the recurrence relation equation can be written as,

$$\begin{aligned} \boldsymbol{\delta}^l &= \frac{\partial \hat{C}}{\partial \mathbf{z}^l} = \left[\frac{\partial \mathbf{z}^{l+1}}{\partial \mathbf{z}^l}\right]^\top \frac{\partial \hat{C}}{\partial \mathbf{z}^{l+1}} = \dot{\mathbf{F}}^l(\mathbf{z}^l)(\mathbf{W}^{l+1})^\top \frac{\partial \hat{C}}{\partial \mathbf{z}^{l+1}} \\ &= \dot{\mathbf{F}}^l(\mathbf{z}^l)(\mathbf{W}^{l+1})^\top \boldsymbol{\delta}^{l+1} \quad , \end{aligned} \tag{5.43}$$

This relation takes us to the last layer L from where the sensitivities are propagated backward through the network to the first layer. Therefore consider the sensitivities in the last layer as,

$$\boldsymbol{\delta}_r^L = \frac{\partial \hat{C}}{\partial z_r^L} = \frac{\partial (\mathbf{t}-\mathbf{a})^\top (\mathbf{t}-\mathbf{a})}{\partial z_r^L} = \frac{\partial \sum_{c=1}^{j} (t_c - a_c)^2}{\partial z_r^L} = -2(t_r - a_r)\frac{\partial a_r}{\partial z_r^L} \quad . \tag{5.44}$$

Since,

$$\frac{\partial a_r}{\partial z_r^L} = \frac{\partial a_r^L}{\partial z_r^L} = \frac{\partial f^L(z_r^L)}{\partial z_r^L} = \dot{f}^L(z_r^L) \quad , \tag{5.45}$$

we can write,

$$\boldsymbol{\delta}_r^L = 2(t_r - a_r)\dot{f}^L(z_r^L) \quad , \tag{5.46}$$

and in the matrix form,

$$\mathbf{s}^L = 2\dot{\mathbf{F}}^L(\mathbf{z}^L)(\mathbf{t}-\mathbf{a}) \quad . \tag{5.47}$$

The GD, depending upon the learning rate, is capable of minimizing the cost function. The multilayer ANN actually possesses many minima, therefore, it is not always possible to achieve global minima (Kaastra and Boyd, 1996). Figure 5.5 shows a conceptual performance surface with local and global minima.

One way to avoid local minima is to try several sets of *initial weights*. Nielsen (2015) suggests to use the Gaussian random variables to initialize the weights and biases with mean = 0 and standard deviation = $1/\sqrt{n}$ with n = is the number of unknown parameters in the ANN. The next section discusses it in details.

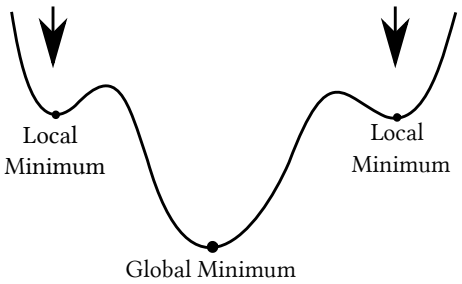


Figure 5.5: A conceptual figure of ANN error surface. Target of training is to reach the global minimum while escaping the local minima.

5.7 CONVERGENCE

The principle of the ANN training is to minimize the difference between the ANN output and target data, also known as convergence. Sometimes though the ANN parameters minimized the mean square error but the network response does not give the accurate approximation because of its architecture limitations (Hagan et al., 2014). The solution to this is to test the performance by changing the number of layers and the number of neurons in each hidden layer. On the other hand, a well-designed network architecture failed to provide the accurate approximation because the entire procedure is depending upon the initial values of the weights and biases, and the multilayer architecture might have more than one minima and the minimization process is stuck in one of them rather than searching for the global minimum. The suggested solution is to try several different sets of the initial weights.

learning rate η is an important factor while considering the optimization. The problem is that a constant η can slow down the convergence, while a large learning rate destabilizes the convergence on the steeper error surface, for instance. Therefore a variable learning rate is suggested in a way that it increases on flat surfaces and then decreases as the slope increased.

1. If the squared error increases by more than some set percentage after a weight update, then the weight update is discarded, the learning rate is reduced
2. If the squared error decreases after a weight update, then the weight update is accepted and the learning rate is increased.

Another parameter which plays an important role in the convergence is the way an ANN consume the source data, sequentially or using batches, for instance. During the sequential learning, the weights and biases are updated after each pass of a sample from source data, whereas in batch learning the weights and biases are updated after the pass of a complete batch of samples from

source data. The goal of learning is to reach the minimum possible cost. Nonlinear ANN have multiple minima. Batch method reaches the minimum whatever basin the initial weights are placed, however, due to individual samples having individual noise the weights can jump to another basin, possibly deeper minimum (LeCun et al., 1998).

An advantage of sequential learning is that we can use shuffled source data, which helps to learn the nonlinear trend in the time series. Shuffle means that the order of the appearance of the column changes, whereas, the order of the rows remains the same and the time behavior of the coefficient within a column remains intact. The target data vector gets the reshuffling in the similar order of the source data sets. The objective of the shuffling is to presents the most unfamiliar samples to ensure that the ANN is learning from the different classes and is not memorizing the values of source data sets. This is most recommended for classification.

Since GD is a slow method and depends upon the learning rate. LM is the ultimate solution to the learning rate problem, however before moving towards LM Newton's method for optimization is presented, since its processing steps are the base of LM optimization. Therefore the next section presents Newton's method in brief.

5.8 NEWTON'S METHOD

Newton's method is the second technique discussed in this study of cost optimization. The idea of Newton's method is to minimize the quadratic approximation iteratively, using the second order Taylor series expansion of the cost function around $\mathbf{x}_i$ as,

$$\mathbf{C}(\mathbf{x}_{i+1}) = \mathbf{C}(\mathbf{x}_i + \Delta\mathbf{x}_i) \approx \mathbf{C}(\mathbf{x}_i) + \mathbf{g}_i^\top \Delta\mathbf{x}_i + \frac{1}{2}\Delta\mathbf{x}_i^\top \mathbf{H}_i \Delta\mathbf{x}_i \quad . \tag{5.48}$$

The $\mathbf{H}_i$ is the $n \times n$ Hessian matrix of $\mathbf{C}(\mathbf{x}_i)$, evaluated at $\mathbf{C}(\mathbf{x})_i$, given in (5.19). $\mathbf{H}_i$ requires the cost function $\mathbf{C}(\mathbf{x}_i)$ to be twice continuously differentiable with respect to the elements of $\mathbf{x}$. Differentiating (5.48) with respect to $\Delta\mathbf{x}_i$ minimizes the resulting change $\Delta\mathbf{C}(\mathbf{x})_i$ if,

$$\mathbf{g}_i + \mathbf{H}_i \Delta\mathbf{x}_i = 0 \quad . \tag{5.49}$$

Solving this equation for $\Delta\mathbf{x}_i$ gives

$$\Delta\mathbf{x}_i = -\mathbf{H}_i \mathbf{g}_i \quad . \tag{5.50}$$

This defines Newton's method as,

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \Delta\mathbf{x} \tag{5.51}$$

$$= \mathbf{x}_i - \mathbf{H}_i^{-1}\mathbf{g}_i \quad , \tag{5.52}$$

where $\mathbf{H}_i^{-1}$ is the inverse of the Hessian of $\mathbf{C}$

GD finds the minimum cost function in small steps whereas, Newton's method is designed to approximate a function as quadratic and then locates the minima of the quadratic approximation in one step. However, if the function is not quadratic, then Newton's method will not generally converge in one step (Hagan et al., 2014). $\mathbf{H}$ has to be a positive definite matrix for all iterations, which may not happen always. Therefore, Newton's method needs modification (Powell, 1987) (Bertsekas, 1995).

5.9 LEVENBERG-MARQUARDT METHOD

The third optimization method presented in this study is the LM method. It gets its name from (Levenberg, 1944) and (Marquardt, 1963). It is an improved form of the GD and Newton's method (Haykin, 2011). In section 5.5 it is presented that GD converge slowly and its convergence depends upon a proper selection of learning rate, on the other hand, in 5.8 it is described that Newton's method converges rapidly near a local or global minimum, but may also diverge, because of the Hessian matrix $\mathbf{H}$, which may not be invertible for all iterations. LM solve this issue and presented a method which prevents the noninvertible situation. Let's start from Newton's method, consider the relation

$$\mathbf{x}_{i+1} = \mathbf{x}_i - \mathbf{H}_i^{-1}\mathbf{g}_i \quad , \tag{5.53}$$

where $H_i^{-1} \cong \nabla^2\mathbf{C}|_{\mathbf{x}=\mathbf{x}_i}$ and $\mathbf{g}_i \cong \nabla\mathbf{C}|_{\mathbf{x}=\mathbf{x}_i}$. Now for a multilayer ANN, consider the mean square error is the average of the error in all layers $l = 1, 2, \ldots, L$, and all the sample points $q = 1, 2, \ldots, \mathrm{Q}$, then we can rewrite (5.15) as

$$\mathbf{C}(\mathbf{x}) = \sum_{q=1}^{\mathrm{Q}} \langle \mathbf{e}|\mathbf{e}\rangle = \sum_{q=1}^{\mathrm{Q}} \sum_{j=1}^{J^L} (e_{j,q})^2 = \sum_{r=1}^{\mathrm{N}} (v_r)^2 = \langle \mathbf{v}|\mathbf{v}\rangle \quad , \tag{5.54}$$

where $e_{j,q}$ is the j^{th} element of the error for the q^{th} sample (input/target pair) and

$$\mathbf{v}^\top = [v_1, v_2, \ldots, v_{\mathrm{N}}] = \begin{bmatrix} e_{1,1} & e_{2,1} & \ldots & e_{J^1,1} & e_{1,2} & e_{2,2} & \ldots & e_{J^2,2} & \ldots & e_{L,1} & e_{L,2} & \ldots & e_{J^L,\mathrm{Q}} \end{bmatrix} \tag{5.55}$$

where, $\mathrm{N} = \mathrm{Q} \times J^L$, and $\mathbf{x}$ is the vectors of all unknowns of ANN given in (5.3), then the j^{th} element of the gradient can be written as,

$$[\nabla C(\mathbf{x})]_j = \frac{\partial C(\mathbf{x})}{\partial x_j} = 2\sum_{r=1}^{\mathrm{N}} v_r(\mathbf{x}) \frac{\partial v_r(\mathbf{x})}{\partial x_j} \quad , \tag{5.56}$$

and the gradient in the matrix form is

$$\nabla \mathbf{C} = 2\mathbf{J}^{\top}(\mathbf{x})\mathbf{v}(\mathbf{x}) \quad , \tag{5.57}$$

where $\mathbf{J}$ is the Jacobin and can be written as,

$$\mathbf{J}(\mathbf{x}) = \begin{bmatrix} \frac{\partial v_1(\mathbf{x})}{\partial x_1} & \frac{\partial v_1(\mathbf{x})}{\partial x_2} & \cdots & \frac{\partial v_1(\mathbf{x})}{\partial x_n} \\ \frac{\partial v_2(\mathbf{x})}{\partial x_1} & \frac{\partial v_2(\mathbf{x})}{\partial x_2} & \cdots & \frac{\partial v_2(\mathbf{x})}{\partial x_n} \\ \vdots & \vdots & & \vdots \\ \frac{\partial v_N(\mathbf{x})}{\partial x_1} & \frac{\partial v_N(\mathbf{x})}{\partial x_2} & \cdots & \frac{\partial v_N(\mathbf{x})}{\partial x_n} \end{bmatrix} \quad . \tag{5.58}$$

The second part of the (5.53) is the Hessian matrix, it's k, j element would be,

$$[\nabla^2 C(\mathbf{x})]_{k,j} = \frac{\partial^2 C(\mathbf{x})}{\partial x_k \partial x_j} = 2\sum_{r=1}^{N} \left\{ \frac{\partial v_r(\mathbf{x})}{\partial x_k}\frac{\partial v_r(\mathbf{x})}{\partial x_j} + v_r(\mathbf{x})\frac{\partial^2 v_r(\mathbf{x})}{\partial x_k \partial x_j} \right\} \quad . \tag{5.59}$$

If the second term in the braces considered being very small then the Hessian matrix can be approximated as

$$\nabla^2 \mathbf{C} \cong 2\mathbf{J}^{\top}(\mathbf{x})\mathbf{J}(\mathbf{x}) \quad . \tag{5.60}$$

No by substituting back the $\nabla \mathbf{C}$ from (5.57) and $\nabla^2 \mathbf{C}$ from (5.60) in (5.53), we get,

$$\begin{aligned} \mathbf{x}_{i+1} &= \mathbf{x}_i - \left[2\mathbf{J}^{\top}(\mathbf{x}_i)\mathbf{J}(\mathbf{x}_i)\right]^{-1} 2\mathbf{J}^{\top}(\mathbf{x}_i)\mathbf{v}(\mathbf{x}_i) \\ &= \mathbf{x}_i - \left[\,\mathbf{J}^{\top}(\mathbf{x}_i)\mathbf{J}(\mathbf{x}_i)\right]^{-1} \mathbf{J}^{\top}(\mathbf{x}_i)\mathbf{v}(\mathbf{x}_i) \quad . \end{aligned} \tag{5.61}$$

As discussed at the beginning of the section that the Hessian may not be invertible, therefore, we modify the matrix and make it sure that it inverts as,

$$\boxed{\mathbf{G} = \mathbf{H} + \mu\mathbf{I} \quad .} \tag{5.62}$$

Now suppose that the eigenvalues and eigenvector of $\mathbf{H}$ are $\{\lambda_1, \lambda_2, \ldots, \lambda_n\}$ and $\{\mathbf{y}_1, \mathbf{y}_1, \ldots, \mathbf{y}_n\}$ then

$$\mathbf{G}\mathbf{y}_j = [\mathbf{H} + \mu\mathbf{I}]\,\mathbf{y}_j = \mathbf{H}\mathbf{y}_j + \mu\mathbf{y}_j = \lambda_j\mathbf{y}_j + \mu\mathbf{y}_j = (\lambda_j + \mu)\mathbf{y}_j \quad . \tag{5.63}$$

In this way eigenvectors of $\mathbf{G}$ are the same as the eigenvectors of $\mathbf{H}$ and eigenvalues are $(\lambda_j + \mu)$. $\mathbf{G}$ would be definite positive by increasing μ until $(\lambda_j + \mu) > 0$ for all j, therefore, the matrix is invertible. This brings the definition of the *Levenberg-Marquardt method*

$$\begin{aligned} \mathbf{x}_{i+1} &= \mathbf{x}_i - \left[\mathbf{J}^{\top}(\mathbf{x}_i)\mathbf{J}(\mathbf{x}_i) + \mu_i\mathbf{I}\right]^{-1}\mathbf{J}^{\top}(\mathbf{x}_i)\mathbf{v}(\mathbf{x}_i) \quad , \\ &\text{or} \\ \Delta\mathbf{x}_i &= -\left[\,\mathbf{J}^{\top}(\mathbf{x}_i)\mathbf{J}(\mathbf{x}_i) + \mu_i\mathbf{I}\right]^{-1}\mathbf{J}^{\top}(\mathbf{x}_i)\mathbf{v}(\mathbf{x}_i) \quad . \end{aligned} \tag{5.64}$$

The algorithm starts with a small value of μ_i. If a step does not yield a smaller value for $C(\mathbf{x})$, then the step is repeated with μ_i multiplied by some factor, for example, $\theta > 1$. Eventually should decrease $C(\mathbf{x})$, since we would be taking a small step in the direction of steepest descent. If a step does produce a smaller value for $C(\mathbf{x})$, then μ_i is divided by θ for the next step, which should provide faster convergence. The algorithm provides a nice compromise between the speed of Newton's method and the guaranteed convergence of steepest descent.

In the following paragraphs it is described that how we can apply the LM to the multilayer ANN. The process starts with the evaluation of the Jacobean matrix $\mathbf{J}(\mathbf{x})$. First of all consider the definitions of $\mathbf{v}$, $\mathbf{x}$ and n in (5.55), (5.3) and (5.4), respectively and then substitute them in the Jacobean matrix in (5.58), we get,

$$\mathbf{J}(\mathbf{x}) = \begin{Bmatrix} \frac{\partial e_{1,1}}{\partial w^1_{1,1}} & \frac{\partial e_{1,1}}{\partial w^1_{1,2}} & \cdots & \frac{\partial e_{1,1}}{\partial w^1_{j,\kappa}} & \frac{\partial e_{1,1}}{\partial b^1_1} & \cdots \\ \frac{\partial e_{2,1}}{\partial w^1_{1,1}} & \frac{\partial e_{2,1}}{\partial w_{1,2}} & \cdots & \frac{\partial e_{2,1}}{\partial w^1_{j,\kappa}} & \frac{\partial e_{2,1}}{\partial b^1_1} & \cdots \\ \vdots & \vdots & \ddots & \vdots & \vdots & \\ \frac{\partial e_{j^1,1}}{\partial w^1_{1,1}} & \frac{\partial e_{j^1,1}}{\partial w^1_{1,2}} & \cdots & \frac{\partial e_{j^1,1}}{\partial w^1_{j,\kappa}} & \frac{\partial e_{j^1,1}}{\partial b^1_1} & \cdots \\ \frac{\partial e_{1,2}}{\partial w^1_{1,1}} & \frac{\partial e_{1,2}}{\partial w^1_{1,2}} & \cdots & \frac{\partial e_{1,2}}{\partial w^1_{j,\kappa}} & \frac{\partial e_{1,2}}{\partial b^1_1} & \cdots \\ \vdots & \vdots & & \vdots & \vdots & \end{Bmatrix} \quad . \tag{5.65}$$

To compute the terms of $\mathbf{J}(\mathbf{x})$ we use the relations in Section (5.6) with small modification. Consider the term in the Jacobean matrix,

$$[\mathbf{J}]_{h,n} = \frac{\partial v_h}{\partial x_n} = \frac{\partial \mathbf{e}_{i,q}}{\partial x_n} \quad , \tag{5.66}$$

with $h = (q-1)J^L + i$, from (5.29) we can write here as,

$$\frac{\partial \hat{C}}{\partial w^l_{j,k}} = \frac{\partial \hat{C}}{\partial z^l_j} \frac{\partial z^l_j}{\partial w^l_{j,k}} \quad , \tag{5.67}$$

with the first term is defined as sensitivity

$$\delta^l_j =: \frac{\partial \hat{C}}{\partial z^l_j} \quad , \tag{5.68}$$

using the same concept, we can define the sensitivity for the LM method as,

$$\tilde{\delta}^l_{j,h} =: \frac{\partial v_h}{\partial z^l_{j,q}} = \frac{\partial e_{i,q}}{\partial z^l_{j,q}} \quad , \tag{5.69}$$

elements of the Jacobean are computed as,

$$[\mathbf{J}]_{h,c} = \frac{\partial v_h}{\partial x_c} = \frac{\partial e_{i,q}}{\partial w^l_{j,k}} = \frac{\partial e_{i,q}}{\partial z^l_{j,q}} \times \frac{\partial z^l_{j,q}}{\partial w^l_{j,k}} = \tilde{s}^l_{j,h} \times \frac{\partial z^l_{j,q}}{\partial w^l_{j,k}} = \tilde{s}^l_{j,h} \times a^{l-1}_{k,q} \quad , \tag{5.70}$$

and for bias

$$[\mathrm{J}]_{h,c} = \frac{\partial v_h}{\partial x_c} = \frac{\partial e_{i,q}}{\partial b_j^l} = \frac{\partial e_{i,q}}{\partial z_{j,q}^l} \times \frac{\partial z_{j,q}^l}{\partial b_j^l} = \tilde{s}_{j,h}^l \times \frac{\partial z_{j,q}^l}{\partial b_j^l} = \tilde{s}_{j,h}^l \quad . \tag{5.71}$$

Now similar to the recursive relation (5.44) for the solution of the final layer, we can initiate the computation as,

$$\begin{aligned} \tilde{\delta}_{j,h}^L &= \frac{\partial v_h}{\partial z_{j,q}^L} = \frac{\partial e_{i,q}}{\partial z_{j,q}^L} = \frac{\partial (t_{i,q} - a_{i,q}^L)}{\partial z_{j,q}^L} = -\frac{\partial a_{i,q}^L}{\partial z_{j,q}^L} \\ &= \begin{cases} -\dot{f}^L(z_{j,q}^L) & \text{for} \quad j = i \quad , \\ 0 & \text{for} \quad j \neq i \quad . \end{cases} \end{aligned} \tag{5.72}$$

Therefore when the input has been applied to the network and the corresponding network output has been computed, the LM backpropagation is initialized with

$$\tilde{\boldsymbol{\delta}}_q^L = \dot{\mathbf{F}}^L(\mathbf{n}_q^L) \quad , \tag{5.73}$$

where is defined in Eq. (5.42). Each column of the matrix must be backpropagated through the network using Eq. (5.42) to produce one row of the Jacobean matrix. The columns can also be backpropagated together using

$$\tilde{\boldsymbol{\delta}}_q^l = \dot{\mathbf{F}}^l(\mathbf{n}_q^l)(\mathbf{W}^{m+1})^\top \tilde{\mathbf{S}}_q^{l+1} \quad . \tag{5.74}$$

The total Marquardt sensitivity matrices for each layer are then created by augmenting the matrices computed for each input is,

$$\tilde{\boldsymbol{\delta}}^l = \begin{bmatrix} \tilde{\boldsymbol{\delta}}_1^l & \tilde{\boldsymbol{\delta}}_2^l & \dots & \tilde{\boldsymbol{\delta}}_{\mathrm{Q}}^l \end{bmatrix} \quad . \tag{5.75}$$

This brings the discussion of the feedforward ANN optimization for cost minimization to an end. In the next chapters, the optimized feedforward networks are used for the classification and prediction of the GRACE datasets. In this chapter, the whole process of training the ANN is presented. In the next section, the whole learning process using ANN is summarized with a brief description of the validation and test process as well.

5.10 VALIDATION AND TEST

The training process starts with the flow of the source data sample from the input layer to the inner layer. It undergoes the following three steps. Firstly, from the input layer, after multiplication with the connecting weights, each value goes to each neuron of the inner layer. The magnitude of the weights decides how much impact of a certain component reduces or remains intact. Secondly,

at each neuron in the inner layer, the products of inputs and weights from all the neurons of the input layers are summed up and bias is added. Thirdly, the resultant value goes to the activation function. The output of the activation function is a value between 0 and 1. Here the flow of the data completes its journey between the input layer and the inner layer. The data from the inner layer to the output layer undergoes the same three steps. Eventually, At each neuron in the output layer, the resultant value is compared with the target data and the difference is sent to a cost minimization algorithm. The minimization process sets the new values of the weights and biases. The process goes on with the whole source data and for several iterations until the cost value does not change any further. Ideally, at this point, the training process achieves a minimum cost.

The validation process is the same as the training process. However, before the testing process, this process fine tunes the hyper-parameters such as, early stopping, in case if the cost remains constant for several iterations, without taking care that the minimum cost is achieved or not. Test process checks the ability of the trained ANN as a classifier. The data flow is the same as in the training step. One of the main differences between them is that the test process stops after the output compared with the target data. In short, the cost minimization does not follow the comparison step. In other words, the test process does not review the numerical values of the weights and biases. Comparison step reveals the quality of the trained ANN. It tells how many SH coefficients are correctly classified. Higher % of classification accuracy permits to use the ANN for classification, contrarily, lower % unveil the shortcomings in the cost minimization process. In such a case, the training process follows the test process with optimal values of cost minimization parameters. This brings the theoretical discussion of the ANN to an end. In the next two chapters, its application for classification and prediction are presented.

6 CLASSIFICATION USING ANN

ANN is a data processing unit which provides a modern data mining and machine learning tool. It helps to perform several tasks such as image, pattern and handwritten character recognition, artificial intelligence, web searching, and language translation. Use of ANN as a classifier is ubiquitous. ANN as a classifier is a supervised classification tool, in which an ANN is trained using samples and afterward used for classification of unknown dataset (Bischof et al., 1992) and (Heermann and Khazenie, 1992). Besides the conventional classification techniques, ANN is a data-driven self-adaptive method. It can adjust itself to the data without specification of the functional or distributional form of the underlying model, moreover, it can approximate any function with arbitrary accuracy (Zhang, 2000).

In Chapter 5 ANN has been introduced with details, where the activation function, cost function and, its evaluation are elaborated and all related parameters are discussed. In this chapter, ANN as a tool for classification of SH coefficients has been described.

For ANN based classification, we need a source and a target dataset. The source dataset must contain many samples of the constituting classes, whereas the target dataset must be a vector of the size, equals to the number of samples in the source dataset and contains the label of the corresponding class. The source and the target dataset and ANN architecture need detail description. Therefore, the following paragraphs first describe the preparation of both datasets followed by a discussion on the ANN architecture.

6.1 SOURCE AND TARGET DATASETS

GRACE monthly SH coefficients represent the seasonal and interannual variations in the gravity field. Hence one could assume SH coefficients posses a periodic behavior. Therefore the core assumption is the value of a specific coefficient depends upon its values in twelve previous epochs,

and consequently, the time series of the coefficient can be written as the sum of linear and periodic signal components. For instance, consider a time series like p includes linear and periodic behaviors, then we can presents this time series as:

$$p(i) = a + bi + c\sin(\omega i) + d\cos(\omega i) + e\sin(2\omega i) + f\cos(2\omega i) \quad , \tag{6.1}$$

with a, b, c, d, e, f are the coefficients of the signal components and i is the epoch. If one cycle completes in twelve epochs, i.e. $i = 12$, it results in angular frequency $\omega = 2\pi/12$ then one can formulate a system of observation equations as,

$$\begin{bmatrix} p(1) \\ p(2) \\ \vdots \\ p(12) \end{bmatrix} = \begin{bmatrix} 1 & 1 & \sin(\omega_1) & \cos(\omega_1) & \sin(2\omega_1) & \cos(2\omega_1) \\ 1 & 2 & \sin(\omega_2) & \cos(\omega_2) & \sin(2\omega_2) & \cos(2\omega_2) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & 12 & \sin(\omega_{12}) & \cos(\omega_{12}) & \sin(2\omega_{12}) & \cos(2\omega_{12}) \end{bmatrix} \begin{bmatrix} a \\ b \\ \vdots \\ f \end{bmatrix} \quad . \tag{6.2}$$

where, $\mathbf{x} = [a \;\; b \;\; c \;\; d \;\; e \;\; f]^\top$,only in this chapter, is the vector of unknowns. The least squares adjustment (LSA) solves the system of linear equations and yields the estimated values $\hat{\mathbf{x}}$.

Consider the GRACE monthly solutions with the uninterrupted monthly variational level SH coefficients time series, range from January 2004 to December 2010 i.e 84 months. Out of 84 monthly dataset 83 i.e. up to November 2010 are used for training and testing and validation, whereas December 2010 acts as the unseen dataset and will be classified using the trained ANN. If m_ℓ, in this chapter, represents a monthly value then $\mathbf{m}_\ell$, in this chapter, represents the vector of time series with 84 monthly values of a coefficient ℓ as,

$$\mathbf{m}_\ell = \begin{Bmatrix} m_\ell^1 & m_\ell^2 & m_\ell^3 & \dots & m_\ell^{83} \end{Bmatrix} \quad . \tag{6.3}$$

In the set of 83 months, there exists 71 sets of 12 consecutive months. Let τ denotes the consecutive month number from $1, 2, \dots, \mathrm{Q}$ and $\mathrm{Q} = 71$ is the total number of consecutive months, or number of samples in the source data, then $\mathbf{s}_\ell^\tau$ denotes one of the consecutive sets from the set $\mathbf{m}_\ell$ as,

$$\mathbf{s}_\ell^\tau = \begin{Bmatrix} m_\ell^\tau & m_\ell^{\tau+1} & m_\ell^{\tau+2} & \dots & m_\ell^{\tau+(12-1)} \end{Bmatrix}_{1\times 12}^\top \quad , \tag{6.4}$$

consider a matrix $\mathbf{S}_\ell$ consists of $\mathbf{s}_\ell^\tau$ as its columns, as,

$$\mathbf{S}_\ell = \begin{bmatrix} \mathbf{s}_\ell^1 & \mathbf{s}_\ell^2 & \mathbf{s}_\ell^3 & \dots & \mathbf{s}_\ell^\tau & \dots & \mathbf{s}_\ell^{71} \end{bmatrix}_{12\times 71} \quad , \tag{6.5}$$

Now if a set of 12 values of $\mathbf{s}_\ell^\tau$ estimates, using LSE, a vector of unknown $\hat{\mathbf{x}}_\ell^\tau = \begin{bmatrix} \hat{a}_\ell^\tau & \hat{b}_\ell^\tau & \hat{c}_\ell^\tau & \hat{d}_\ell^\tau & \hat{e}_\ell^\tau & \hat{f}_\ell^\tau \end{bmatrix}^\top$ as shown in the beginning of the section using the signal components estimation then the matrix

of all consecutive set $\mathbf{S}_\ell$ and the matrix $\hat{\mathbf{X}}_\ell$ containing the estimated vector $\hat{\mathbf{x}}_\ell^\tau$ can be written as,

$$\mathbf{S}_\ell = \begin{bmatrix} m_\ell^1 & m_\ell^2 & \dots & m_\ell^\tau & \dots & m_\ell^{71} \\ m_\ell^2 & m_\ell^3 & \dots & m_\ell^{\tau+1} & \dots & m_\ell^{72} \\ \vdots & \vdots & \dots & \vdots & \dots & \vdots \\ m_\ell^{11} & m_\ell^{12} & \dots & m_\ell^{\tau+10} & \dots & m_\ell^{81} \\ m_\ell^{12} & m_\ell^{13} & \dots & m_\ell^{\tau+11} & \dots & m_\ell^{82} \end{bmatrix}_{12\times 71}, \tag{6.6}$$

$$\hat{\mathbf{X}}_\ell = \begin{bmatrix} \hat{\mathbf{x}}_\ell^{\,1} & \hat{\mathbf{x}}_\ell^{\,2} & \dots & \hat{\mathbf{x}}_\ell^{\,\tau} & \dots & \hat{\mathbf{x}}_\ell^{71} \end{bmatrix}_{6\times 71}. \tag{6.7}$$

By inserting the estimated values $\hat{\mathbf{x}}_\ell^\tau$ back to (6.1) with $i = 13$, we get the predicted value for the proceeding SH coefficient, say $\underline{m}_\ell^\tau$. Repeating this for all τ sets, a vector of predicted values for a coefficient ℓ can be written as,

$$\underline{\mathbf{m}}_\ell = \begin{bmatrix} \underline{m}_\ell^1 & \underline{m}_\ell^2 & \underline{m}_\ell^3 & \dots & \underline{m}_\ell^\tau & \dots & \underline{m}_\ell^{71} \end{bmatrix}_{1\times 71}, \tag{6.8}$$

or in matrix format for all coefficients ℓ ranges from $1, 2, \dots, L$

$$\underline{\mathbf{M}} = \begin{bmatrix} \underline{m}_1^1 & \underline{m}_1^2 & \underline{m}_1^3 & \dots & \underline{m}_1^\tau & \dots & \underline{m}_1^{71} \\ \underline{m}_2^1 & \underline{m}_2^2 & \underline{m}_2^3 & \dots & \underline{m}_2^\tau & \dots & \underline{m}_2^{71} \\ \underline{m}_3^1 & \underline{m}_3^2 & \underline{m}_3^3 & \dots & \underline{m}_3^\tau & \dots & \underline{m}_3^{71} \\ \vdots & \vdots & \vdots & \dots & \vdots & \dots & \vdots \\ \underline{m}_\ell^1 & \underline{m}_\ell^2 & \underline{m}_\ell^3 & \dots & \underline{m}_\ell^\tau & \dots & \underline{m}_\ell^{71} \\ \vdots & \vdots & \vdots & \dots & \vdots & \dots & \vdots \\ \underline{m}_L^1 & \underline{m}_L^2 & \underline{m}_L^3 & \dots & \underline{m}_L^\tau & \dots & \underline{m}_L^{71} \end{bmatrix}_{L\times 71}, \tag{6.9}$$

where 71 columns represent the predicted monthly coefficients starting from January 2005 up to November 2010. Each superscript index τ represents a month *mon* of the year y from 2005 to 2010, such as, first twelve columns belong to 2005, the next twelve to 2006 and similarly last eleven belong to January 2010 to November 2010. If m is replaced by p and instead of month number, at superscript, ranging from 1 to 71 two indices i.e. *mon* in superscript and y in subscript is used in a way that the columns equivalence is given by,

$$\underline{m}_\ell^{\,1} \text{ to } \underline{m}_\ell^{12} = \underline{p}_{\ell,2005}^{Jan} \text{ to } \underline{p}_{\ell,2005}^{Dec},$$

$$\underline{m}_\ell^{13} \text{ to } \underline{m}_\ell^{24} = \underline{p}_{\ell,2006}^{Jan} \text{ to } \underline{p}_{\ell,2006}^{Dec},$$

$$\underline{m}_\ell^{25} \text{ to } \underline{m}_\ell^{36} = \underline{p}_{\ell,2007}^{Jan} \text{ to } \underline{p}_{\ell,2007}^{Dec},$$

$$\underline{m}_\ell^{37} \text{ to } \underline{m}_\ell^{48} = \underline{p}_{\ell,2008}^{Jan} \text{ to } \underline{p}_{\ell,2008}^{Dec},$$

$$\underline{m}_{\ell}^{49} \text{ to } \underline{m}_{\ell}^{60} = \underline{p}_{\ell,2009}^{Jan} \text{ to } \underline{p}_{\ell,2009}^{Dec},$$

$$\underline{m}_{\ell}^{61} \text{ to } \underline{m}_{\ell}^{71} = \underline{p}_{\ell,2010}^{Jan} \text{ to } \underline{p}_{\ell,2010}^{Nov},$$

a vector of monthly predicted coefficients, with elements instead of $\underline{m}_1^{\tau}$, is now denoted as $\underline{p}_{\ell,y}^{mon}$, can be written as,

$$\underline{\mathbf{p}}_{y}^{mon} = \begin{bmatrix} \underline{p}_{1,y}^{mon} & \underline{p}_{2,y}^{mon} & \underline{p}_{3,y}^{mon} & \dots & \underline{p}_{\ell,y}^{mon} \dots & \underline{p}_{\mathrm{L},y}^{mon} \end{bmatrix} \quad . \tag{6.10}$$

Here, a comparison of the original $\mathbf{p}_y^{mon}$ (c.f. (2.10)) and predicted $\underline{\mathbf{p}}_{y}^{mon}$ SH coefficients is vital. It reveals how well the frequency decomposition of the SH coefficients keep the original gravity variation information. The comparisons process segregate a group of SH coefficients whose values are relatively more preserved than the rest of coefficients. The comparison process consists of the following steps,

1. normalize both original GRACE $\mathbf{p}_y^{mon}$ and predicted $\underline{\mathbf{p}}_{y}^{mon}$ fields of December 2010,
2. compute their difference $\mathbf{q}_y^{mon}$ and note the maximum $max(\mathbf{q}_y^{mon})$ and minimum $min(\mathbf{q}_y^{mon})$ differences, where

$$\mathbf{q}_y^{mon} = \begin{bmatrix} q_{1,y}^{mon} & q_{2,y}^{mon} & q_{3,y}^{mon} & \dots & q_{\ell,y}^{mon} \dots & q_{L,y}^{mon} \end{bmatrix} \quad . \tag{6.11}$$

3. select a threshold value T_y^{mon} between $min(\mathbf{q}_y^{mon})$ and $max(\mathbf{q}_y^{mon})$ values,
4. compare the threshold to the differences in $\mathbf{q}_y^{mon}$ and segregate the coefficients into

$$\tilde{q}_y^{mon} = \begin{cases} \text{preserved class,} & \text{if } q_{\ell,y}^{mon} \leq T_y^{mon} \text{ and} \\ \text{Non-preserved class,} & \text{if } q_{\ell,y}^{mon} < T_y^{mon} \quad , \end{cases} \tag{6.12}$$

where, $\tilde{q}_y^{mon}$, denotes a classified coefficients.

Figure 6.1 presents the majority of the preserved coefficients belong to the same region in the SC matrix as the nonessential coefficients, (c.f. Figure 4.4). This shows the presence of two different kinds of coefficients in the original GRACE data. They differ not only on the bases of their

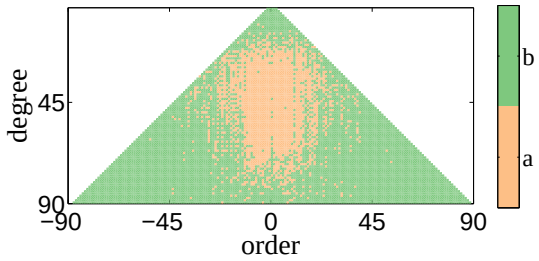


Figure 6.1: a)Preserved class b) non-preserved class.

information content but their behavior as a time series signal. The task of the ANN classification is to learn the behavioral properties of the coefficients and recognize the difference between them. After this test, the feature dataset which consists of signal components proves that it can represent the original SH coefficients in the ANN classification process as the source dataset.

The results of threshold classification, Figure 3.7 and Figure 3.8, show that there are 937 unclassified, 6204 essential and 1136 nonessential coefficients in GRACE data. The idea is to use the estimated signal components $\hat{\mathbf{X}}_\ell$ of the essential and nonessential coefficient, as samples for the ANN-based supervised classification and include them in the source dataset with labels '1' and 'o' in the corresponding target vector for the essential and nonessential coefficients, respectively. For a coefficient ℓ, the source and the target dataset are written as,

$$\hat{\mathbf{X}}_\ell = \begin{bmatrix} \hat{\mathbf{x}}_\ell^{\,1} & \hat{\mathbf{x}}_\ell^{\,2} & \dots & \hat{\mathbf{x}}_\ell^{\,\tau} & \dots & \hat{\mathbf{x}}_\ell^{\,71} \end{bmatrix}_{6\times 71} \quad . \tag{6.13}$$

$$\mathbf{t}_\ell = \begin{bmatrix} t_\ell^1 & t_\ell^2 & \dots & t_\ell^\tau & \dots & t_\ell^{71} \end{bmatrix}_{1\times 71} \quad , \text{ where } t_\ell^\tau \in \{0, 1\} \tag{6.14}$$

The coefficients from the unclassified class do not take any part as source dataset for ANN classification. If n coefficients (essential or nonessential) are selected for the training set then the source and corresponding target datasets for n coefficients can be written as:

$$\hat{\mathbf{X}} = \begin{bmatrix} \hat{\mathbf{X}}_1 & \hat{\mathbf{X}}_2 & \dots & \hat{\mathbf{X}}_n \end{bmatrix} \quad , \tag{6.15}$$

$$\mathbf{t} = \begin{bmatrix} \mathbf{t}_1 & \mathbf{t}_2 & \dots & \mathbf{t}_n \end{bmatrix} \quad . \tag{6.16}$$

After the description of the source and target datasets, the following section discusses the architecture and the working of the ANN in detail.

6.2 NETWORK ARCHITECTURE

A trained ANN is used to classify the dataset into essential and nonessential classes. The first step is to decide the size of the ANN, i.e. the number of inner layers and the number of neuron in

each layer. Since the basic unit of the source data is the set of six estimated coefficients of signal components, therefore the ANN has an input layer with six neurons and the target is to identify the two classes, i.e. essential and nonessential, presented by '1' and 'o' in one target vector, therefore, the output layer consists of one neuron. Regarding the number of the inner layers and the number of neurons in these layers, there exist several guidelines. Most of these guidelines are the outcomes of the heuristics. Therefore, different combinations of layers and neurons are tested. According to the observations, increasing the number of the inner layers decrease the relative accuracy of the classification, while only one inner layer gives the best results. Moreover, the literature suggests that the number of neurons in the inner layers must be integer multiple of the number of neurons in the input layers. It works up to a certain extent. The number of neurons in the inner layer, for this case, equals twice the number of neurons of the first layer, i.e. 12, gives the best relative accuracy. While, 3× and 4× neutrons of the number of neurons of first layers improve the relative accuracy, but not significantly and take longer processing times. In the end, however, hit and trial show that ten neurons in the inner layer give a fine tune to the classification results.

With six neurons in the input layer ten in the inner layer and one in the output layer, the total number of unknown variables is 81 using equation (5.4). On the other hand, the input data samples are in several thousand. Therefore it is an overdetermined system. Since it is nonlinear, we have to iterate it to optimize.

6.3 TRAINING THE ANN

LM algorithm has been used to training the ANN for classification, whereas the output of the threshold classification is used as the external validation or the target dataset. Figure 6.2 represents the target classes with a) nonessential and b) essential coefficients, where the ring-like white region in the center represents the unclassified SH coefficients which are not used as a sample in the source dataset. The source dataset composed of only essential and nonessential coefficients is used to train the ANN.

Eventually, 91.4% classification accuracy during the training was achieved, which means that we can still expect some miss-classification in the output, for instance, the salt-piper like a spread of green pixels in the essential region shown in the training output is given in Figure 6.3. A glance of the relative classification accuracy of the three processes given in Table 6.1 shows that how well the ANN is trained. Results are given in percentage. According to the details, during the training

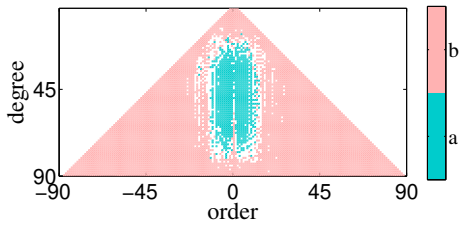


Figure 6.2: Regions of a) essential class and b) nonessential class used for the source data formulation. The ring like white space represents the unclassified coefficients and not used as the source data samples.

process, 91.4% of the coefficients are correctly classified and only 8.6% are misclassified. Similarly, in the validation and test process, 87% and 88% are correctly classified whereas 13% and 12% are misclassified, respectively.

Table 6.1: % classification accuracy of the training, validation and test processes. The higher accuracy suggests that the ANN is ready for the classification task.

	Training Process	Validation Process	Test Process	Total
correctly classified coefficients (%)	91.4	87	88	88
misclassified coefficients (%)	8.6	13	12	12

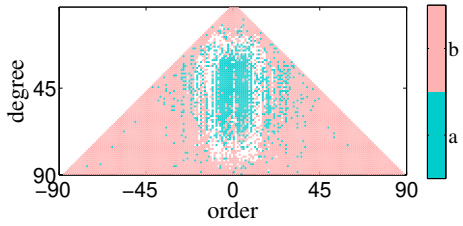


Figure 6.3: Training output for the SH coefficients, into a) essential class and b) nonessential class.

6.4 RESULTS

The trained ANN is now ready to classify the unseen dataset of December 2010. The ANN segregates the SH coefficients on the bases of what it learns during the learning process. Figure 6.4 shows that the ANN identifies a) essential and b) nonessential classes.

A trained ANN, using the uninterrupted dataset from January 2004 to November 2010, has classified the dataset of December 2010. The classification output shows that 1553 coefficients concentrated in the region around the zonal coefficients of SC format matrix from degree ~10 to ~75 and order ~15 of both sine and cosine coefficients belong to the nonessential class and 6724

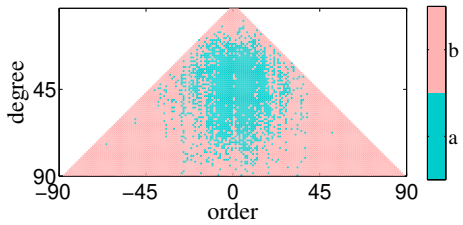


Figure 6.4: ANN classification output for the SH coefficients of December 2010, into a) essential class and b) nonessential class.

belong to the essential class. The comparison of the two outputs, i.e. from (kNN) (c.f. Figure 4.4) and ANN classification confirms the presence of two distinct groups in the dataset. Though these methods utilize different techniques reaches a similar result. The classes have a more fuzzy boundary in case of ANN classification because of less then 100% accuracy in the training phase. The source data preparation for ANN classification shows that the presentation of SH coefficients as the time series signal also preserve the time behavior of the coefficients. This concept is used in the prediction of SH coefficients using ANN in the next chapter.

PREDICTION USING ANN

7

The objective of the study is to analyze the behavior of the GRACE monthly SH coefficients and find patterns, trends, classes or groups among them. Chapter 3 finds clusters in the data using k means clustering and classifies the data using threshold method. Chapter 4 classifies the unclassified coefficients using k nearest neighbor algorithms. Chapter 6 also bifurcate the GRACE SH coefficients into two groups using ANN. All of these techniques point out the presence of two distinct classes, i.e. essential and nonessential, in the GRACE monthly SH data. Essential possess the most and nonessential possess very minute information of the gravity variations. The idea is to exclude the nonessential SH coefficients from the gravity recovery process, which decrease the formal error spectrum of the recovered SH coefficients. This chapter exploits the ANN to identify the predictable coefficients. The successful prediction enables us to reduce the number of GRACE coefficients during the gravity recovery process, which eventually reduces formal error spectrum, even further. Chapter 8 discusses the gravity recovery process and formal error of the recovered field, in details. The usage of ANN as a prediction tool is ubiquitous in forecasting and finance. Financial services have been the second largest sponsors of research in ANN applications (Kaastra and Boyd, 1996), (Trippi and Turban, 1992). In the following sections the discussion about prediction using ANN starts with the description of the source and target data formulation.

7.1 SOURCE AND TARGET DATASETS

The process of classifying SH coefficients into essential and non essential groups utilizes one ANN, as discussed in Section 6.2, however prediction process treats each SH coefficient separately. In other words, prediction process uses separate ANN for each SH coefficient.

Consider the GRACE monthly solutions with monthly variational level SH coefficients time series, range from April 2002 to June 2017 i.e 183 months. Chapter 2 discusses the GRACE data preprocessing and extraction of variational level coefficients in detail (c.f. 2.1). The GRACE SH data

range from April 2002 to November 2016 i.e. 176 data values, act as the source data to predict SH coefficients of December 2016 and the values from January 2017 to June 2017 are used for comparison with predicted values. The missing values are filled by interpolation. If m_ℓ, represents a coefficients ℓ then the time series of 176 monthly values in a vector can be written as,

$$\mathbf{m}_\ell = \left\{ \begin{array}{cccccc} m_\ell^1 & m_\ell^2 & m_\ell^3 & \dots & m_\ell^{176} \end{array} \right\} \ . \tag{7.1}$$

Since the variational level SH coefficients represent the seasonal and interannual variations in the gravity field, hence one could assume that variational level SH coefficients posses a periodic behavior. Therefore the core assumption is that the value of a monthly coefficient depends upon its instance in the last few, say κ months. Let τ denote the consecutive month number from $1, 2, \dots, \mathrm{Q}$ and Q is the total number of consecutive months, also the number of samples in the source data, then $\mathbf{s}_\ell^\tau$ denotes one of the consecutive sets from the set $\mathbf{m}_\ell$ as,

$$\mathbf{s}_\ell^\tau = \left\{ \begin{array}{cccccc} m_\ell^\tau & m_\ell^{\tau+1} & m_\ell^{\tau+2} & \dots & m_\ell^{\tau+(\kappa-1)} \end{array} \right\}_{1\times\kappa}^\top \ , \tag{7.2}$$

where κ denotes the size of the set. κ plays an important role in the formulation of the source data matrix and ANN design. Its value could be different for each coefficient. Rather than following any statistical property such as auto-correlation in monthly values of a coefficient, the ANN iterates the prediction process for κ in search of best prediction. The value ranges from 5 to 12. Failure to get a higher accuracy of prediction during this range means the time series of the coefficient is unpredictable. In this way, κ is one of the parameters to compose the list of predictable SH coefficients. For source data formulation for a coefficient ℓ, consider a matrix $\mathbf{X}_\ell$ consists of $\mathbf{s}_\ell^\tau$ as its columns, as,

$$\mathbf{X}_\ell = \left[\begin{array}{cccccccc} \mathbf{s}_\ell^1 & \mathbf{s}_\ell^2 & \mathbf{s}_\ell^3 & \dots & \mathbf{s}_\ell^\tau & \dots & \mathbf{s}_\ell^{\mathrm{Q}} \end{array} \right]_{\kappa\times\mathrm{Q}} \ , \tag{7.3}$$

with, Q represents the total number of $\mathbf{s}_\ell^\tau$ sets, or the samples in the source data. Firstly, a process calculates Q, depending upon κ, from the range of GRACE data. For instance, if $\kappa = 5$ then there exist 171 sets between January 2002 to November 2010 i.e. $\mathrm{Q} = 171$, with each set has 5 consecutive months that acts as the source data set and a proceeding month as its target value. Therefore for a SH coefficient ℓ, a matrix $\mathbf{X}_\ell$ of the order 5×171 is ready as source data matrix and a vector $\mathbf{t}_\ell$

of the order 1×171 as target data vector, one target point for one sample in source dataset, as,

$$\mathbf{X}_\ell = \begin{bmatrix} m^1 & m^2 & m^3 & \dots & m^{171} \\ m^2 & m^3 & m^4 & \dots & m^{172} \\ m^3 & m^4 & m^5 & \dots & m^{173} \\ m^4 & m^5 & m^6 & \dots & m^{174} \\ m^5 & m^6 & m^7 & \dots & m^{175} \end{bmatrix}_{5\times 171} , \tag{7.4}$$

$$\mathbf{t}_\ell = \begin{bmatrix} m^6 & m^7 & m^8 & \dots & m^{176} \end{bmatrix}_{1\times 171} \text{, or in general} \tag{7.5}$$

$$\mathbf{t}_\ell = \begin{bmatrix} t^1 & t^2 & t^3 & \dots & t^{171} \end{bmatrix}_{1\times 171} . \tag{7.6}$$

In ANN, learning is a collection of three processes i.e. training, validation and test. During the training an ANN using source data matrix, learns how to reach closer to the target value (Goodfellow et al., 2016). A cost minimization algorithm works on the difference of the output of the ANN and the corresponding target value, and iterates to bring the cost close to zero. Eventually, after the successful learning, the numerical values of the same coefficient ℓ from July 2016 to November 2016, considering $\kappa = 5$, act as the source data set i.e. $\tilde{\mathbf{s}}_\ell$ of size 5×1 as in (7.7) and the trained ANN predicts value of December 2016. In this way, the size of the source data for the prediction step is also same as κ.

$$\tilde{\mathbf{s}}_\ell = \begin{bmatrix} m^{172} & m^{173} & m^{174} & m^{175} & m^{176} \end{bmatrix}^\top_{1\times\kappa} . \tag{7.7}$$

The prediction process can be extended to the next epochs i.e. January 2017 to June 2017 by including the freshly predicted value in the source data as,

$$\underbrace{\tilde{\mathbf{s}}_\ell = \begin{bmatrix} m^{173} \\ m^{174} \\ m^{175} \\ m^{176} \\ m^{177} \end{bmatrix}, \mathbf{t}_\ell = \begin{bmatrix} m^{178} \end{bmatrix}}_{\text{predicting Jan. 2017}} , \quad \underbrace{\tilde{\mathbf{s}}_\ell = \begin{bmatrix} m^{174} \\ m^{175} \\ m^{176} \\ m^{177} \\ m^{178} \end{bmatrix}, \mathbf{t}_\ell = \begin{bmatrix} m^{179} \end{bmatrix}}_{\text{predicting Feb. 2017}} , \quad \cdots , \quad \underbrace{\tilde{\mathbf{s}}_\ell = \begin{bmatrix} m^{177} \\ m^{178} \\ m^{179} \\ m^{180} \\ m^{181} \end{bmatrix}, \mathbf{t}_\ell = \begin{bmatrix} m^{182} \end{bmatrix}}_{\text{predicting Jun. 2017}} .$$

During the learning, sequential and batch learning are two different ways a ANN can consume the source dataset. This study utilizes the batch methods for prediction. The batch method utilizes

the input data in small bunches. The batch size B means how many columns of the source data includes in each bunch while the rows remain intact.

In the example stated above related to the variable $\kappa = 5$, for instance for the batch size $B = 6$ means that the source data matrix of 5×171 with $\kappa = 9$ is now composed of 19 batches, each of size 5×9, and during the training process the system updates the weights and biases after learning from each six columns of a batch. For this study, for each value of κ i.e. $5, 6, \ldots, 12$ ANN iterates for B ranges from 1 to 8. Note that, $B = 1$ means sequential, incremental, stochastic (Hagan et al., 2014) or online learning (Stegemann, 1999). In the next section the architecture of the ANN for the prediction is described.

7.2 NETWORK ARCHITECTURE

The architecture of ANN for prediction is different from that for classification. According to the details, for classification the source dataset is composed of all SH coefficients, whereas for prediction, the study treats each SH coefficient separately. This means that here in prediction we have separate ANN for each coefficient and they could have different architecture. Furthermore, in case if they have same ANN architecture, for sure they must have different values of parameters specially biases and weights.

The feedforward network for prediction consists of three layer i.e input, inner and output layers. The number of neurons in the input layer is equal to the number of features, i.e. κ, in the source dataset. The neurons in the input layer receives the source dataset. Each neurons in the first inner layer receives the output of each neuron of the input layer. Similarly, each neuron in the next layer receive the data from each neuron of the previous layer.

Now back to the example of $\kappa = 5$ with batch size $B = 9$ from section 5.1, the input layer have 5 neurons and the output layer have one neuron. The number of neurons in the input layer is equal to the number of features in the source data or rows of the source data matrix which is controlled by κ and the number of neuron in the output layer is equal to the number of rows of the target data which is one, in our case of prediction. During the experiment, I observe that ANN with one inner layer works perfectly for the prediction, while increasing the number of inner layers does not improve the results, as suggested by literature mentioned in Section 5.2. Concerning the number of neurons in the inner layer N_h, the equation (5.10), i.e. $N_h = \mathrm{Q}/(\alpha \times (\kappa + M))$,

with, α, an arbitrary scaling factor, usually 2-5, controls the over fitting, Q is number of samples in source data, M number of target classes, suggests that if $\kappa = 5$, $Q = 171$, $\kappa + M = 5 + 1 = 6$, then the round off values of N_h are $17, 10, 7, 6$ for $\alpha = 2, 3, 4, 5$, respectively. But, before the final decision it is important to consider the relationship between the number of unknown variables n (weights and biases) in an ANN and the number of sample Q in source dataset. To determine total number of unknowns n, consider once again Figure 5.1, the ANN with 5,4,1 neurons in three layers. The number of unknown variables n is given by (5.4), i.e. $n = J^1(\kappa+1) + J^2(J^1+1) + \ldots + J^l(J^{l-1} + 1) + \ldots + J^L(J^{L-1}+1)$, where, κ is the number of neurons in the input layer and J^l is the number of neuron in the layer $l = 1, 2$. Therefore the number of unknown variables is 29. In order to arrive at a regular linear system of equations we must have more than 29 observations to solve the 29 unknowns. More the number of observations better would be the results. In our case where the source data size is very small, we have to be very careful and stay within the limits of the data size to set the number of neurons in the inner layer of an ANN.

The time series, for this study, consists of 171 data points i.e. from April 2002 to November 2017, one for each month. The learning process utilizes the time series not only for training, but also for validation and testing as well. Usually, the data split ratio is 70%, 15% and 15% for training, validation and testing, respectively. Which reduced the available data sample for training, even further. Kaastra and Boyd (1996) suggests that the training set is at least twice as large as the number of weights or preferably more. Therefore, this study restricts the size of ANN with maximum 85 number of unknowns n. With $n = 85$, suppose $\kappa = 12$ then the maximum number of the neuron in the inner layer can reach up to 6. To restrict the ANN up to $n = 85$ is a very strict limit, however keeping in view the size of the dataset, this is optimal. Table 7.1 presents the number of maximum neurons, an ANN possibly have in the inner layer against the different values of the κ and keeping the maximum number of $n \leq 43$.

In short, the input layer contains the neurons equal to the number of the input values from source dataset or in other words, the input layer contains the neurons equal to κ. The output layer have only one neuron. The ANN have one inner layer. The number of the neurons in the inner layer, ranges between 6-11, depends upon, firstly, the size of the data, specially κ, secondly the relative accuracy of the prediction. Sigmoid functions acts the role of the activation function. The output of the output layer i.e a^L is compared with the target data value t^τ for each data point in the cost function. The difference between the two determines the current status of the training process. The objective of the training process is to reduce the cost and bring it close to zero. Since,

Table 7.1: For different values of κ the maximum number of neurons in the inner layer of an ANN does not exceed the limit of 43

κ	size of inner layer (neuron)	n (bias + weights)
5	11	78
6	10	81
7	9	82
8	8	81
9	7	78
10	7	85
11	6	79
12	6	85

the time series of the coefficients have yearly periodic behavior, therefore before moving towards the prediction for SH coefficients lets consider the ANN prediction for an equivalent sine function.

7.3 EXAMPLE: SINE WAVE PREDICTION

The time series of SH coefficients posses an inter annual behavior. A prediction process, with the assumption that a sine wave can represent a time series of a SH coefficient, provides a starting point for the prediction of a SH coefficient time series. A sine wave having 16 cycles with 12 equidistant points in each cycle is shown in Figure 7.1. 16 cycles represent 16 years from 2002-2017 and 12 points in each cycle are equivalent to the 12 months of a year. In this example the prediction process uses the data from April 2002 to November 2016 as the source data and predict the coefficient for December 2016. Both, GD and LM algorithm are used for prediction. In the following paragraph firstly the procedure and outcomes of GD are presented, afterwards the description and results of LM are discussed.

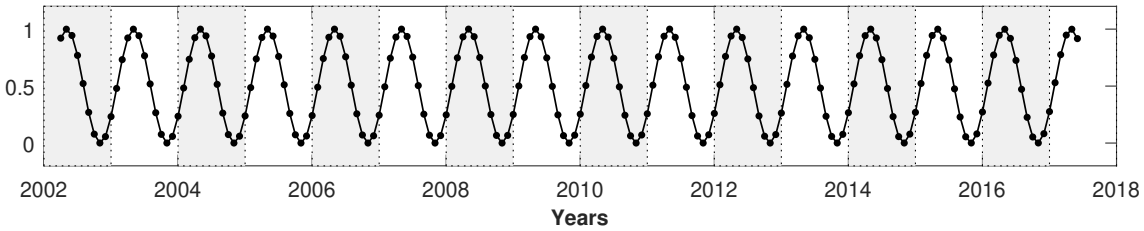


Figure 7.1: A sine wave, acting as a SH coefficient time series.

The prediction starts with a network of three i.e. input, inner and output layers, with one neuron in the output layer and four neurons in the inner layer, as discussed in the previous section.

The number of the neurons in the input layer κ, the learning rate η, number of iterations i, batch size B and most importantly the vector of initial values for biases and weights $\mathbf{x}$ are the variables which need fine tuning to get the best prediction. First of all, to get the initial guess for the vector of unknown variables $\mathbf{x}$ a source data and the target data with arbitrarily selected $\kappa = 8$ and $B = 4$ has been formulated, as described in Section 7.1. 70% of which is set as training and 30% as test data. Figure 7.2 represents the training and test targets using a thin and thick blue line, respectively. At the end of the thick line, a cyan circle represents the target for the prediction. The data is

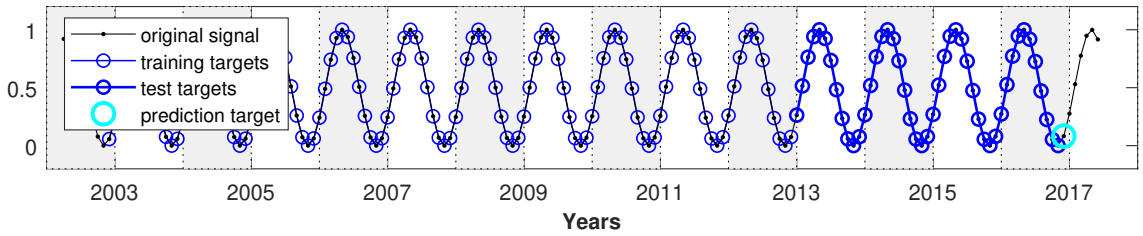


Figure 7.2: Target datasets for training process in thin blue line, for test process in thick blue line.

then used to train the ANN several times to test the randomly initialize the vector of unknown variables $\mathbf{x}$. The initial set which gives the best results has been saved and used in the rest of the process. The second parameter needed to be analyzed is the learning rate. Using the above-stated setting for κ, B and initial values of $\mathbf{x}$, different values of the learning rate has been investigated. The changing shape of cost minimization curve gives an insight into the learning process. For example, Figure 7.3 displays curves of cost minimization due to different learning rates η. It shows that a small η requires more iterations to minimize the cost whereas cost minimizes quickly as the η increases. On the other hand, Figure 7.4 displays the cost minimization curve is jumping around

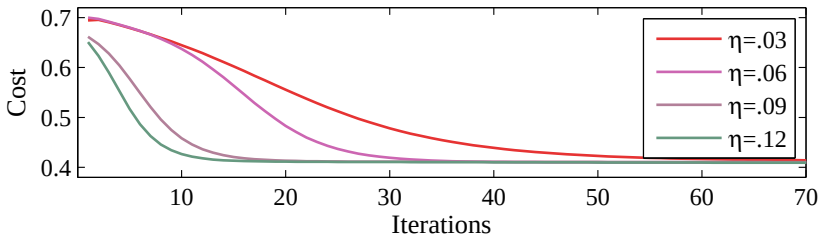


Figure 7.3: Cost minimization curves with different learning rates (η). A small η requires more iterations to minimize the cost.

to achieve the minimum because the η is too large. An optimal learning rate brings the cost to the minimum value quickly and therefore the process does not need many iterations. The plot in Figure 7.3 shows that the cost curve due to $\eta = .12$ does not decrease very slowly either jumping around the minimum, therefore, it is selected as an initial learning rate. The optimal learning rate

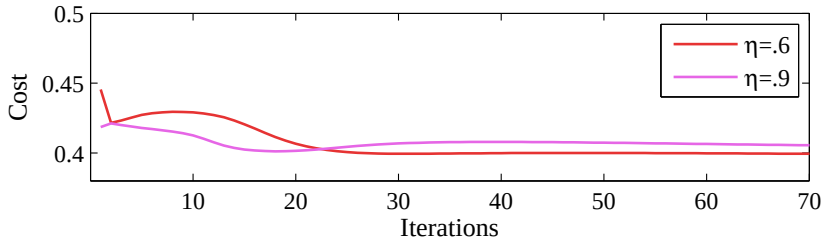


Figure 7.4: Cost minimization curves with large η jumping around the minimum.

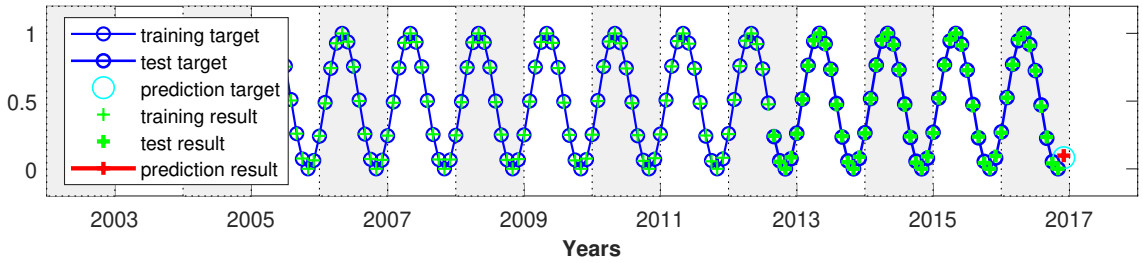
also helps us to set the number of iterations. Kaastra and Boyd (1996) discusses the issue of the number of optimal iterations in detail and suggest several checks to alter the number of iterations, such as

- if the cost remains constant for several iterations such as 20-30 iterations there is no need to iterate the process any further.
- If the cost decreases up to 50-60 % in the first few iterations,
  - reduce the number of iterations significantly,
  - reduce the learning rate to half.

In this way, by changing the learning rate in fewer iteration algorithm reaches the minimum, however, to start the process $i = 200$ is set as the initial value. κ and B are arbitrarily selected to initialize the other variable. Now by iterating the process for κ and the B bring us to an optimal combination which produces the best trained ANN. A flow chart in Figure 7.6 summarizes the iterating process. It generates the cost curves and the predicted values with combinations of κ, B and i. Loops iterate κ from 5 to 12, B from 1 to 8 and i 1 to 200.

A painstaking work, while observing the behavior of the cost minimization curve and changing the number of iteration and learning rate, leads to the outcome. A combination of $\kappa = 6$ and $B = 8$ with $\eta = 1.9$ and $i = 160$ produces the best results for the sine wave prediction. Figure 7.5(a) represents the final results. Thin green marks show that the trained ANN is able to imitate the target points of the training data matrix. Moreover, thick green marks and a red mark inside the cyan circle represent the accurate output of the test and prediction targets, respectively. To verify the prediction ability of the trained ANN the prediction process is extended up to May 2011. Figure 7.5(b) represents the predicted curve using a red line.

In the second part of the sine function example, the LM algorithm is used for optimization. In contrary to GD, LM is self-tuning optimizing algorithms as described in Section 5.9. Figure 7.5(c)



(a) Prediction results for the training, testing and prediction processes.

(b) Prediction results using GD extended up to June 2017.

(c) Prediction results using LM extended up to June 2017.

Figure 7.5: Prediction results: Sine function using GD and LM

represents the output. The blue color line in the background shows the targets and the green marks show the corresponding output, whereas the red thick line represents the outcome of the prediction step.

Using the LM algorithm, one has to iterate different values of the κ and B, however, there is no need to search for the learning rate. As the LM is the improved version of Newton's method and the GD and it operates with dynamic learning rate and reachs the global minimum in one step, therefore the difference between the two results is quite obvious. The following plot shows that the error of LM is less than that of GD.

The experience gained during the process of this example helps to initiate the ANN prediction process for the SH coefficients. The following section reports the results of the prediction using ANN.

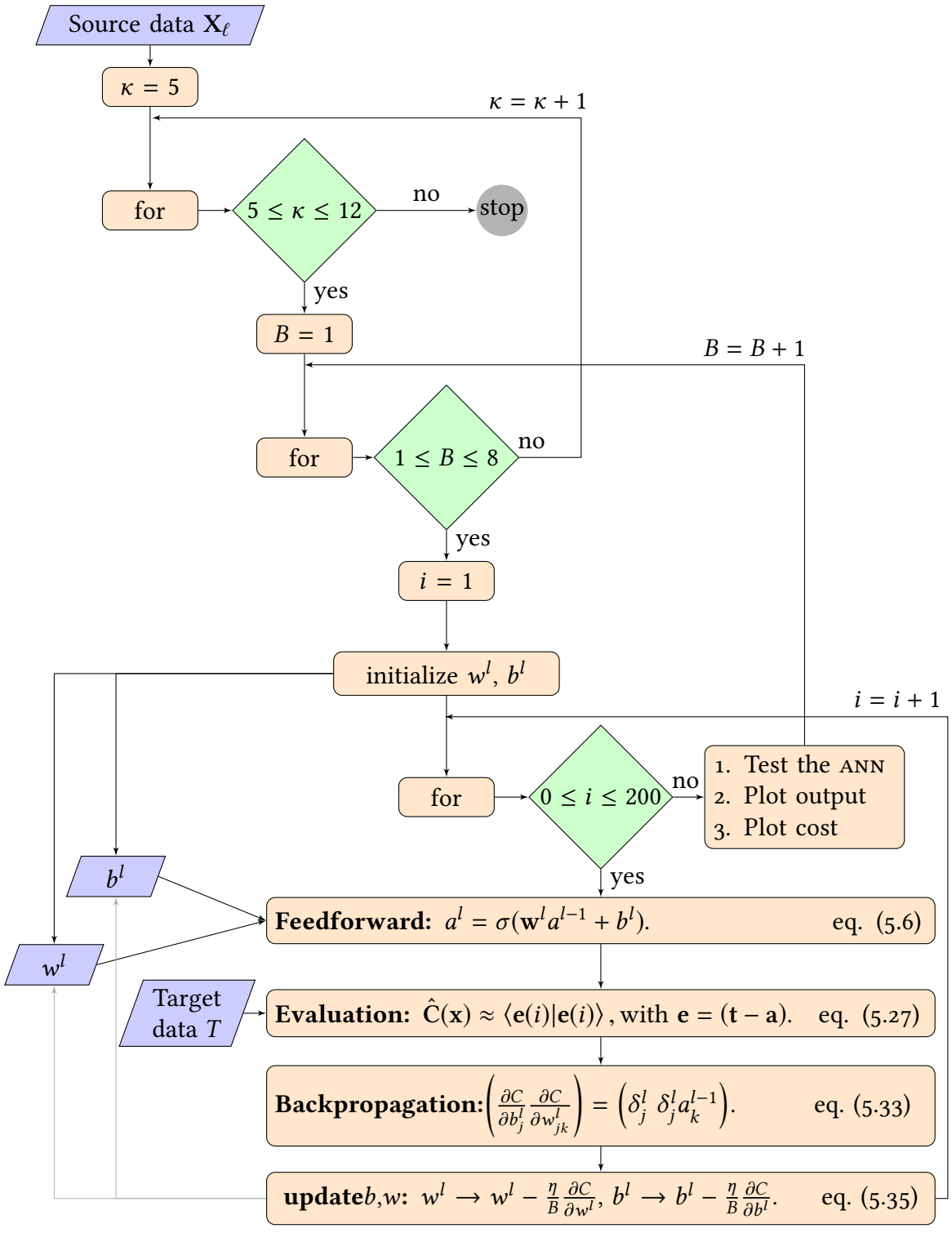


Figure 7.6: Flow chart summarizes the processing steps to train ANN using GD.

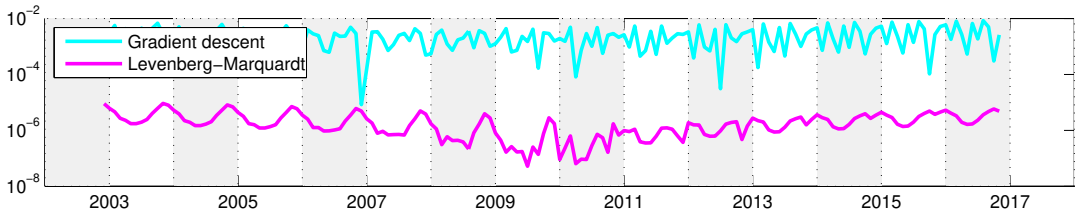


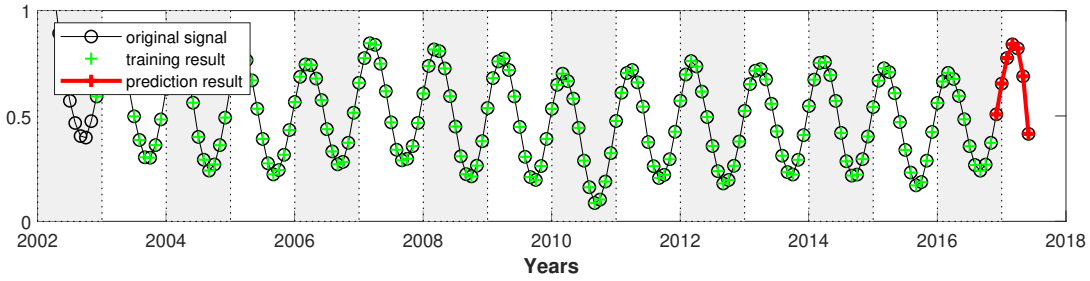
Figure 7.7: Relative error of GD is higher w.r.t. LM for training the ANN for sine function.

7.4 RESULTS

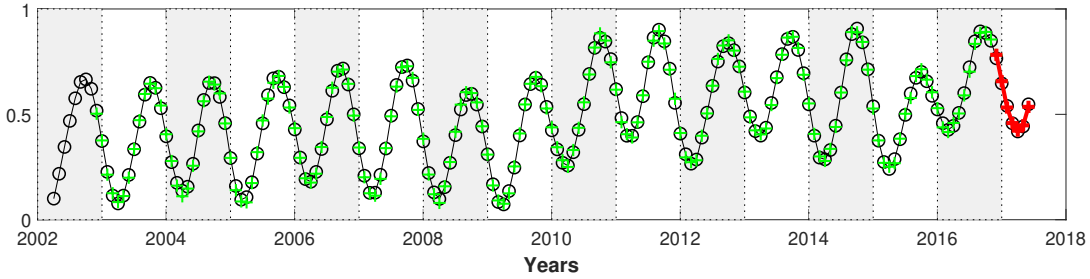
Section 7.3 present a simple case of sine wave prediction, which acts as a base for GRACE SH coefficient prediction. Selection of initial values for the parameters, analyzing the output of the cost function for all SH coefficients is a very time-consuming task, therefore LM which offer automatic adjustment of the number of iteration i and learning rate η is used. The process iterates the value of κ, B, and initial values of $\mathbf{x}$ to reach the minimum cost. Figure 7.8(a) and 7.8(b) present the outcome of the coefficient prediction of $C_{19,1}$ and $S_{13,7}$. The green marks show that the trained ANN successfully imitate the targets, while a thick red line represents the predicted values of the coefficient from December 2016 to June 2017.

Despite the hard work, there are many SH coefficients could not be predicted, for example, Figure 7.8(c) presents time series of $S_{3,3}$. The relative error of three cases, presented in Figure 7.8(d), show that the prediction of the $S_{3,3}$ given in Figure 7.8(c) is not acceptable. After analyzing the cost, error curves and predicted values, several coefficients are identified as the predictable coefficients. Figure 7.9 presents the predicted coefficients in the SC format. Each SH coefficient has its separate ANN with different sets of initial weights and biases. All of them have one inner layer which is composed of 4 to 6 neurons.

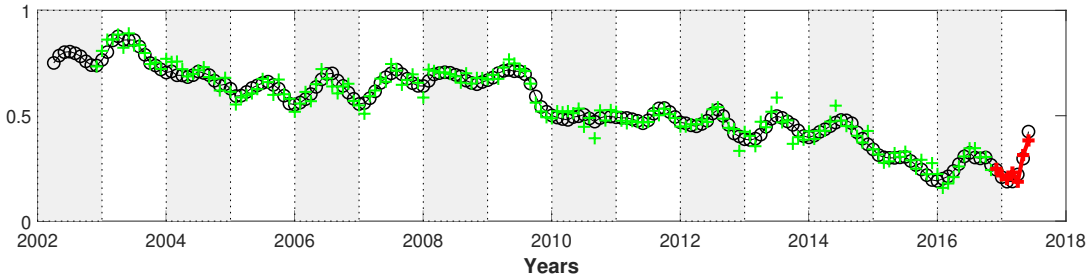
The lengthy process of ANN has predicted 245 coefficients successfully. The rest of essential coefficients are random and therefore even after tuning the parameters, such as changing the number of neurons in the inner layer, batch size, κ and initial weights, ANNs are not able to predict them. After removing the predictable coefficients from the essential coefficients, the class is called a reduced solution. According to the results, the final number count of the essential, nonessential and predictable classes of coefficients are 6245, 1787 and 245, respectively. Finally, the next section illuminates the comparison of prediction using ANN and polynomial fit.



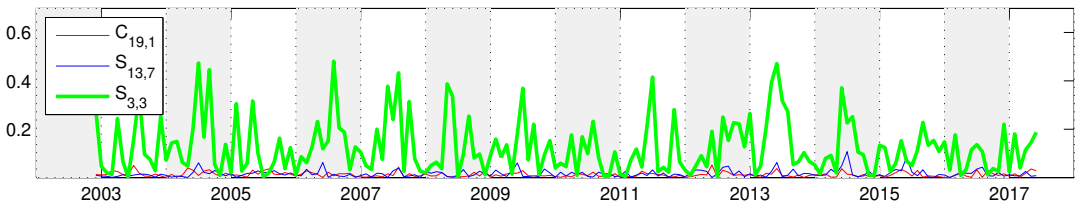
(a) prediction using ANN of $C_{19,1}$. The red curve show the higher prediction quality.



(b) prediction using ANN of $S_{13,7}$. The red curve show that the prediction quality is reliable.



(c) prediction using ANN of $S_{3,3}$. The red curve show that the prediction quality is not reliable.



(d) The relative errors of the predicted time series for $C_{19,1}$, $S_{13,7}$ and $S_{3,3}$.

Figure 7.8: Prediction results using LM for selected coefficients and their relative errors.

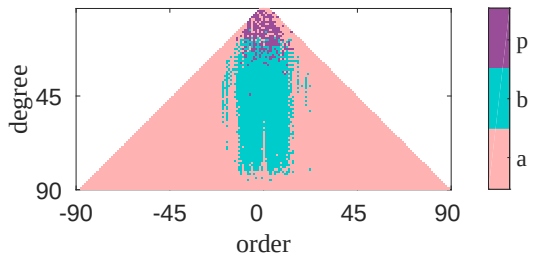


Figure 7.9: a) essential and b) nonessential classes in SC format

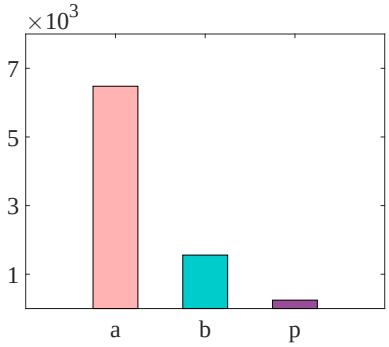


Figure 7.10: Number of coefficients, 6245, 1787 and 245 in a) essential, b) nonessential and p) predictable classes, respectively.

7.5 ANN VS POLYNOMIAL PREDICTION

Polynomial fitting is an alternative or equivalent to the ANN prediction. However, based on numerical simplicity the polynomial fitting is better than the ANN. Indeed, an ANN requires tough training phase and large computer memory to be available for prediction. However it outruns performance of polynomial fitting. Polynomial fitting performs well for interpolation within the data limits if it fits the data optimally. Even an optimal polynomial fit does not perform well for extrapolation. Either the values increase or decrease immediately outside the input data limits. Contrarily ANN has mechanisms of training and testing to check under and over-fit situations. An ANN is more flexible to mimic a time series.

Suppose N is the number of the given data point x, y then interpolation is defined as a way to find the data values at the unknown points within the data limits and extrapolation is to find the data point beyond the data limits. Trigonometric polynomial, given in the following equation, is one of many ways to extrapolate the data for prediction.

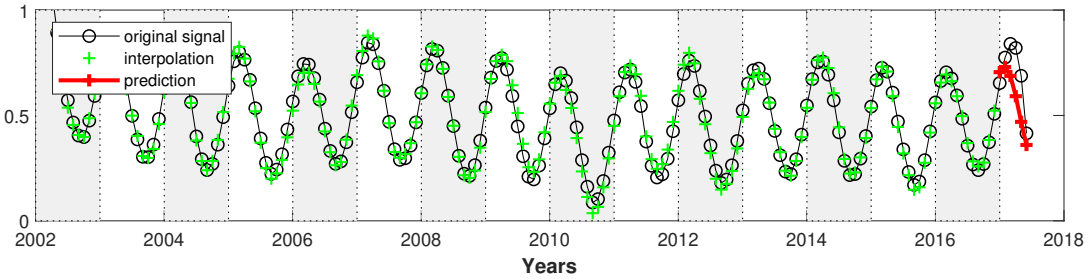
$$p(x) = a_0 + \sum_{k=1}^{K} a_k \cos(kx) + \sum_{k=1}^{K} b_k \sin(kx) \tag{7.8}$$

where $a_0, a_1, \ldots, a_k, b_1, \ldots, b_k$ are $2K+1$ unknown coefficients. After finding the coefficients, (c.f. for solution and other details Restrepo et al. (2001)), we get a function which passes through the N data points,

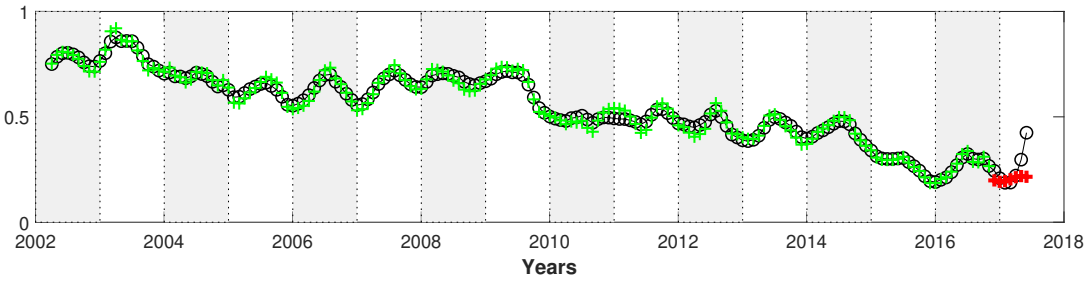
$$p(x_n) = y_n, \qquad n = 0, 1, 2, \ldots, N-1 \tag{7.9}$$

The plots in Figure 7.11 shows the result of prediction using trigonometric polynomial. The time series of the coefficients $C_{19,1}$ and $S_{3,3}$ are given in the Figure 7.11(a) and 7.11(b), respectively. The plots show that for both curves the interpolation using trigonometry give excellent results

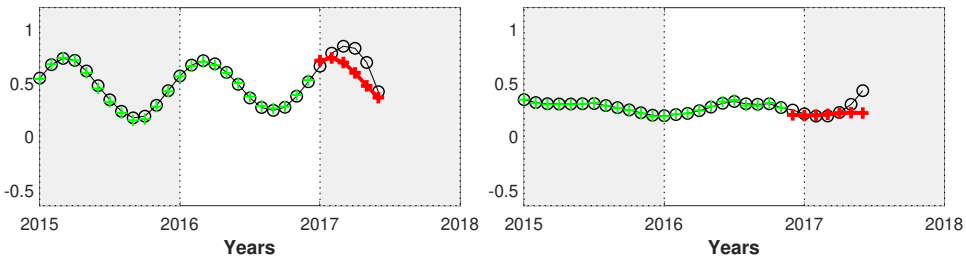
however for the prediction phase it fails badly. Figure 7.11(c) and 7.11(d) show the prediction results in closeup for coefficients $C_{19,1}$ and $S_{3,3}$, respectively.



(a) prediction of $C_{19,1}$. The red curve show bad prediction quality.



(b) prediction of $S_{3,3}$. The red curve show bad prediction quality.



(c) Closeup $C_{19,1}$.

(d) Closeup $S_{3,3}$.

Figure 7.11: Prediction results using trigonometric polynomial fit for selected coefficients.

The results in Figure 7.11 shows the inability of the trigonometric polynomials to predict the data value outside the given data limits. The comparison of prediction curves of $C_{19,1}$ in Figure 7.8(a) and 7.11 show that the ANN predicts far more batter than trigonometric polynomial. While in case of $S_{3,3}$ trigonometric polynomial is better in interpolation but both method fail to predict the random time series.

Next chapter presents a gravity recovery process based on the variational equation method and ends with the comparison of full and reduced solutions of the recovered gravity fields.

8 GRAVITY RECOVERY

The objective of the study is to identify classes in the GRACE monthly SH coefficients. Chapter 3 and Chapter 4 identify essential and nonessential classes while Chapter 7 identifies a group of predictable coefficients. The goal of the study is to show that if the gravity recovery process ignores nonessential and predictable classes during the recovery process, the quality of the recovered coefficients improves. This chapter proves this point with the help of a closed loop gravity recovery simulation. The process starts with the variational level monthly coefficients of Dec. 2016 denoted by $p^{Dec.}_{\ell,2016}$, which refers to the GRACE monthly coefficients after subtracting $\boldsymbol{\mu}_s$ the static field or long term mean field (Meyer et al., 2012). In vector form can be written as,

$$\mathbf{p}^{Dec}_{2016} = \left[p^{Dec.}_{1,2016}\ p^{Dec.}_{2,2016}\ p^{Dec.}_{3,2016}\ \cdots\ p^{Dec.}_{\ell,2016}\cdots\ p^{Dec.}_{L,2016}\right] \quad , \tag{8.1}$$

where $\ell = 1, 2, \ldots, 8276$ is the running coefficient number in the vector of all coefficients, for details see Chapter 2, (c.f. (2.10)).

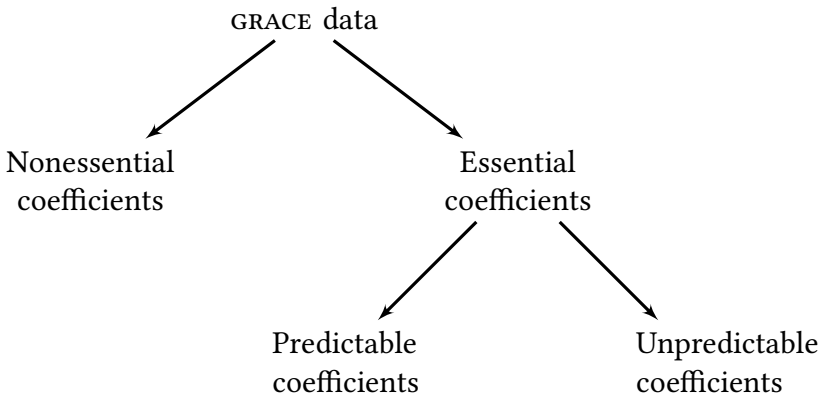


Figure 8.1: Flowchart of classification scheme.

Clustering and classification of $\mathbf{p}^{Dec.}_{2016}$ identify two classes of coefficients, i.e. essential and nonessential. Afterwards, ANN distinguishes predictable and unpredictable coefficients among the essential class. Flowchart in Figure 8.1 summarizes the classification scheme. In this chapter, $\boldsymbol{\mu}_{2016}$

(c.f. (2.13)) the mean field of the year 2016 acts as the reference field, hereafter called *a priori field* and $\mathbf{p}_{2016}^{Dec.}$ serves as the *true field* (Jäggi et al., 2006). The gravity recovery simulation using a priori and true fields recovers a set of SH coefficients, hereafter called *recovered fields*.

This chapter describes the details of the gravity recovery process. The process has its roots in the GRACE gravity recovery technique, for instance, the true range-rates $\dot{\rho}(t)$ are expressed as the truncated Taylor series with respect to unknown dynamic parameters $p_{\ell,2016}^{Dec.}$ about the a priori range-rates $\dot{\rho}_0(t)$ (Jäggi et al., 2010). GRACE, a constellation of two free falling low earth orbiting gravity recovery satellites, is an implementation of a low-low satellite to satellite (LL-SST) tracking system in which on-board sensors measure the inter-satellite range-rates (Rummel et al., 2002) (Freeden et al., 2002). The variation in inter-satellite range is due to the gravity field variation underneath the satellites. Regions of slightly stronger gravity affect the leading satellite first, accelerating it slightly stronger than the trailing satellite (NASA, 2002). There are several methods in practice to recover the gravity field from range-rates observation such as *variational equation approach* (Ballani, 1988) (Reigber, 1989) (Reigber et al., 2005) (Förste et al., 2008) (Tapley et al., 2005), *short arc approach* (Mayer-Gürr et al., 2005), *energy balance approach* (Weigelt, 2007) and *acceleration approach* (Rummel, 1979) (Reubelt, 2008) (Liu, 2008). *The handbook of Geodesy* summarizes all methods (Keller, 2013). In this study, the variational equation approach is utilized for gravity recovery.

Gravity field recovery from range-rates measurements can be considered as a differential orbit improvement process (Jäggi et al., 2006). Use of range, range-rates for the analysis of satellite based geodetic network is explained in details by Grafarend and Livieratos (1978). Grafarend and K.Heinz (1978) present the rank defect analysis of satellite geodetic network in the dynamic mode where the coefficients of SH representation of the gravity field are unknown is presented. This study uses numerical integration, firstly, to generate the position and velocity vectors of the two GRACE satellites and later to compute the partial derivatives of the position and velocity vectors with respect to a priori field. Section 8.1 introduces the observables of a LL-SST system. Section 8.2 states the mathematical details of the orbit determination. Section 8.3 formulates the variational equations for gravity recovery. Section 8.4 explains the process of computing the position and velocity vectors of the a priori satellites and function of partial derivatives of the position and velocity vectors with respect to the coefficients of a priori field using the *variation of constants* method. Section 8.5 shows how to solve the first order linear integral equation obtained by variation of constants method. Section 8.6 states the process to compute the range-rates using the

position and velocity vectors integrated earlier in Sections 8.2. Section 8.7 describes the process of coefficient estimation and evaluate the performance of the *variation of constants* method. Section 8.8 presents results, comparisons and analysis.

In the study, the gravity recovery process simulates the GRACE system with certain simplifications. It ignores all kinds of tides, the gravitational forces due to Sun, Moon and planets and all non gravitational accelerations during the orbit integration. Furthermore, it considers only the Greenwich Apparent Sidereal Time (GAST) to convert the coordinates of position and velocity vectors from the Earth-fixed system to the inertial system.

8.1 LL-SST OBSERVABLES

Inter-satellite range-rates are the key observations of the GRACE system. For the simulation, range-rates are computed using the position and velocity vectors generated by the orbit determination process (c.f. Section 8.2). Let $\mathbf{x}$ represents the position vectors of GRACE satellites in the Earth-fixed reference frame as,

$$\mathbf{x}_s = x_s\mathbf{e}_1 + y_s\mathbf{e}_2 + z_s\mathbf{e}_3, \tag{8.2}$$

where s = A, B, represents two satellites i.e. A is the leading and B is the trailing. and $\mathbf{e}_1, \mathbf{e}_2$ and $\mathbf{e}_3$ are the three unit vectors in the x, y and z directions. The differences between the respective position vectors are,

$$\delta\mathbf{x} = \mathbf{x}_\mathrm{B} - \mathbf{x}_\mathrm{A}. \tag{8.3}$$

Then the scalar inter-satellite range is

$$\rho = \sqrt{\langle\delta\mathbf{x}|\delta\mathbf{x}\rangle} \quad , \tag{8.4}$$

The unit vector in the direction of the inter-satellite range is

$$\mathbf{e} = \{\mathbf{e}_x,\ \mathbf{e}_y,\ \mathbf{e}_z\} = \frac{\delta\mathbf{x}}{\rho} \quad . \tag{8.5}$$

ρ and $\mathbf{e}$ constitutes the inter-satellite range vector as,

$$\delta\mathbf{x} = \rho\mathbf{e} \quad . \tag{8.6}$$

Differentiating (8.6) results in range-rates,

$$\delta\dot{\mathbf{x}} = \dot{\rho}\mathbf{e} \ + \ \rho\dot{\mathbf{e}} \quad , \tag{8.7}$$

where, $\delta\dot{\mathbf{x}}$ is the difference between the velocity vectors of GRACE satellites. $\mathbf{e}$ is a unit vector and therefore

$$\langle \mathbf{e}|\mathbf{e}\rangle = 1 \implies \langle \mathbf{e}|\dot{\mathbf{e}}\rangle = 0 \quad . \tag{8.8}$$

The time derivative $\dot{\mathbf{e}}$ of the line of sight vector (LOS) is perpendicular to the LOS vector itself. The vector $\dot{\mathbf{e}}$ itself is not a unit vector. Therefore (8.7) becomes

$$\langle \delta\dot{\mathbf{x}}|\mathbf{e}\rangle = \dot{\rho}\,\langle \mathbf{e}|\mathbf{e}\rangle + \rho\,\langle \dot{\mathbf{e}}|\mathbf{e}\rangle \quad , \tag{8.9}$$

or

$$\dot{\rho} = \langle \delta\dot{\mathbf{x}}|\mathbf{e}\rangle \quad . \tag{8.10}$$

which represents a projection of the velocity difference vector along the line-of-sight vector. It is noted that this quantity is not the magnitude of the velocity difference vector (Kim, 2000). Finally, using (8.5), the range-rates are equal to

$$\dot{\rho} = \left\langle \delta\dot{\mathbf{x}}|\frac{\delta\mathbf{x}}{\rho}\right\rangle \quad . \tag{8.11}$$

Mathematical derivation from (8.2) to (8.11) summarizes the computation of range-rates, provided the position and velocity vectors are available. In the following section, a detail discussion unveils the orbit determination process to compute the position and velocity vectors.

8.2 ORBIT DETERMINATION

The motion of a satellite around a celestial body can be expressed as a second order differential equation. From the *law of gravitation* we know that the gravitational force $\mathbf{F}$ between a satellite with mass m under the central gravitational force of a single point mass M is given by $\mathbf{F} = G\frac{Mm}{r^2}\mathbf{e}_r$, with the gravitational constant G, the distance between the satellite and the center of point mass r and the direction vector along the force of gravitation $\mathbf{e}_r$. Furthermore, from *Newton's second law of motion* $\mathbf{F} = m\ddot{\mathbf{x}}$, therefore, the equation of motion of a satellite under the influence of a isotropic central force can be written as,

$$m\ddot{\mathbf{x}} = -G\frac{Mm}{r^2}\mathbf{e}_r \quad . \tag{8.12}$$

As $\mathbf{e}_r = \frac{\mathbf{x}}{|\mathbf{x}|}$, the (8.12) becomes,

$$\ddot{\mathbf{x}} = -\frac{GM}{|\mathbf{x}|^3}\mathbf{x} \quad , \tag{8.13}$$

In case of Earth as the central gravitational body, there are number of additional forces acting on the satellite due to which the satellite experience addition acceleration, such as, acceleration due to non-spherically and inhomogeneous mass distribution within Earth, neighboring celestial bodies,

Ocean and earth tides, atmospheric drag and solar radiation. These forces are called perturbed forces (Seeber, 2003). For this simulation, only the acceleration $\mathbf{f}(\mathbf{x})$ due to non-spherically and inhomogeneous mass distribution within the Earth is considered. It can be added into the equation of motion as addition term as,

$$\ddot{\mathbf{x}} = -\frac{GM}{|\mathbf{x}|^3}\mathbf{x} + \mathbf{f}(\mathbf{x}) \quad . \tag{8.14}$$

It is a non-linear second order differential equation with initial state vector $\{\mathbf{x}(t_0) \neq \mathbf{0}, \dot{\mathbf{x}}(t_0) \neq \mathbf{0}\}$. An alternative form of motion of equation is

$$\ddot{\mathbf{x}} = \text{grad}\, V = \nabla V \quad , \tag{8.15}$$

where V is the gravitational potential and is given by,

$$V(r,\theta,\lambda) = \frac{GM}{r}\left(1 + \sum_{l=2}^{\infty}\left(\frac{R}{r}\right)^l \sum_{m=0}^{l} \overline{P}_{lm}(\cos\theta)\left[\overline{C}_{lm}\cos(m\lambda) + \overline{S}_{lm}\sin(m\lambda)\right]\right) \quad . \tag{8.16}$$

The first term $\frac{GM}{r}$ describe the potential due to the isotropic center force while the remaining part represents the finite estimation of the disturbing potential, estimated up to certain degree l_{max} of the SH.

$$T(r,\theta,\lambda) = \frac{GM}{r}\sum_{l=2}^{l_{max}}\left(\frac{R}{r}\right)^l \sum_{m=0}^{l} \overline{P}_{lm}(\cos\theta)\left[\overline{C}_{lm}\cos(m\lambda) + \overline{S}_{lm}\sin(m\lambda)\right] \quad , \tag{8.17}$$

where R is the semi-major axis of a reference ellipsoid of the earth, $\overline{S}_{l,m}$ and $\overline{C}_{l,m}$ with l is SH degree, m is SH order, are fully normalized spherical harmonics (SH) coefficients for the disturbing potential, and $\overline{P}_{lm}(\cos\theta)$ are the fully normalized associated Legendre functions, for detail, see (Heiskanen and Moritz, 1967). The disturbing potential T is observed in local north oriented frame, which if defined as,

- 1st axis is pointing in radial direction
- 2nd axis is oriented towards north
- 3rd axis points eastwards

then the partial derivatives of 8.17 or the gravitational potential gradients (Liu, 2008) with respect to r, θ, λ are,

$$\begin{aligned}
\frac{\partial T}{\partial r} &= -\frac{GM}{r^2}\sum_{l=2}^{l_{max}}\left(\frac{R}{r}\right)^l \sum_{m=0}^{l} \overline{P}_{lm}(\cos\theta)\left[\ \overline{C}_{lm}\cos(m\lambda) + \overline{S}_{lm}\sin(m\lambda)\right](l+1) \quad , \\
\frac{1}{r}\frac{\partial T}{\partial \theta} &= -\frac{GM}{r^2}\sum_{l=2}^{l_{max}}\left(\frac{R}{r}\right)^l \sum_{m=0}^{l} \overline{P}'_{lm}(\cos\theta)\left[\ \overline{C}_{lm}\cos(m\lambda) + \overline{S}_{lm}\sin(m\lambda)\right]\sin\theta \quad , \\
\frac{1}{r\sin\theta}\frac{\partial T}{\partial \lambda} &= \frac{GM}{r^2}\sum_{l=2}^{l_{max}}\left(\frac{R}{r}\right)^l \sum_{m=0}^{l} \overline{P}_{lm}(\cos\theta)\left[-\overline{C}_{lm}\sin(m\lambda) + \overline{S}_{lm}\cos(m\lambda)\right] m\frac{1}{\sin\theta} \quad .
\end{aligned} \tag{8.18}$$

If n, e and i, in this chapter only, represent local north oriented, earth fixed and inertial frame of references, respectively then the transformation matrix $\mathbf{R}_n^e$ in 8.19 converts gradient vector from north oriented frame to earth fixed frame.

$$\mathbf{R}_n^e = \begin{bmatrix} \sin\theta\cos\lambda & -\cos\theta\cos\lambda & -\sin\lambda \\ \sin\theta\sin\lambda & -\cos\theta\sin\lambda & \cos\lambda \\ \cos\theta & \sin\theta & 0 \end{bmatrix} \quad . \tag{8.19}$$

Afterwards, $\mathbf{R}_e^i = \mathbf{R}_3(-\text{GAST})$ rotation matrix rotates the vector from earth fixed frame to inertial frame. The complete transformation turns out to be

$$\mathbf{R}_n^i = \mathbf{R}_e^i \mathbf{R}_n^e \ , \tag{8.20}$$

and (8.21) shows the complete coordinate transformation with $\left[g_x, g_y, g_z\right]_i^\top$ as estimated gradient vector of the force field $\mathbf{f}(\mathbf{x})$ in inertial coordinate frame, as,

$$\mathbf{f}(\mathbf{x}) = \begin{bmatrix} g_x \\ g_y \\ g_z \end{bmatrix}_i = \mathbf{R}_n^i \begin{bmatrix} \mathbf{e}_r \frac{\partial T}{\partial r} \\ \mathbf{e}_\theta \frac{1}{r} \frac{\partial T}{\partial \theta} \\ \mathbf{e}_\lambda \frac{1}{r\sin\theta} \frac{\partial T}{\partial \lambda} \end{bmatrix}_n \quad , \tag{8.21}$$

where, $\mathbf{e}_r$, $\mathbf{e}_\theta$ and $\mathbf{e}_\lambda$ are the base vector of gradient in local frame. The integration of (8.14) for the perturbed motion of equation is worked out numerically. A multi-step integrator ode113 is used to solve it for the position and velocity vectors of the satellite orbit. The position vectors of the satellites are used to simulate the range-rates measurements. The details are given in Section 8.6. In the next section, the gravity recovery using variational equation is introduced.

8.3 VARIATIONAL EQUATIONS

Variational equations method is used to recover the gravity field. Variational equations are always linear (Riley et al., 1967). Consider the following equation of motion,

$$\ddot{\mathbf{x}} = \nabla U(\mathbf{x}) + \nabla T(\mathbf{x}) \ , \tag{8.22}$$

where U represents the isotropic part of the gravitational force as given by (8.13) and T represents the anisotropic part of the gravitational force as given by (8.17) with $\mathbf{x}$ is the position vector of the satellite. To study the impact of slightly changing gravity parameters such as SH coefficients, i.e. C_{lm} and S_{lm} on the position and velocity of satellites, the partial derivatives of the position and velocity with respect to these parameters are required (Ballani, 1988) (Montenbruck and Gill,

2000). For simplicity C_{lm} and S_{lm} are denoted as p_ℓ, where ℓ is the running coefficient number from 1 to 4096 represents the sine coefficients and from 4097 to 8276 represents cosine coefficients in the monthly data vector (c.f. Section 2.2), Therefore after differentiating (8.22) with respect to p_ℓ, the equation becomes,

$$\frac{\partial \ddot{\mathbf{x}}}{\partial p_\ell} = \frac{\partial \nabla U(\mathbf{x})}{\partial p_\ell} + \frac{\partial \nabla T(\mathbf{x})}{\partial p_\ell} \quad . \tag{8.23}$$

Since T is much smaller than U and therefore the orbit $\mathbf{x}$ can be replaced by the reference orbit $\mathbf{x}_0$. Then the term $\frac{\partial \nabla T(\mathbf{x}_0)}{\partial p_\ell}$ does net depend upon $\mathbf{x}$, now by applying the chain rule only on the first term of the right hand side, the equation becomes,

$$\frac{\partial \ddot{\mathbf{x}}}{\partial p_\ell} = \frac{\partial \nabla U}{\partial \mathbf{x}} \frac{\partial \mathbf{x}}{\partial p_\ell} + \frac{\partial \nabla T(\mathbf{x}_0)}{\partial p_\ell} \quad . \tag{8.24}$$

Now after considering the definition,

$$\boldsymbol{\xi}_\ell = \frac{\partial \mathbf{x}}{\partial p_\ell} \quad , \tag{8.25}$$

the (8.24) becomes,

$$\ddot{\boldsymbol{\xi}}_\ell = \nabla^2 U \boldsymbol{\xi}_\ell + \frac{\partial \nabla T(\mathbf{x}_0)}{\partial p_\ell} \quad , \tag{8.26}$$

where $\nabla^2 U$ initially computed in LNOF and then rotated to the inertial frame using $\mathbf{R}_n^i$ from the left and its transpose, i.e. $\mathbf{R}_i^n$ from the right. Its components in LNOF are given as,

$$\begin{aligned}
U_{xx} &= \tfrac{\partial}{\partial r}\left(\tfrac{\partial U}{\partial r}\right) \quad , \\
U_{yy} &= \tfrac{1}{r}\tfrac{\partial}{\partial \theta}\left(\tfrac{1}{r}\tfrac{\partial U}{\partial \theta}\right) + \tfrac{1}{r}\tfrac{\partial U}{\partial r} \quad , \\
U_{zz} &= \tfrac{1}{r\sin\theta}\tfrac{\partial}{\partial \lambda}\left(\tfrac{1}{r\sin\theta}\tfrac{\partial U}{\partial \lambda}\right) + \tfrac{1}{r}\tfrac{\partial U}{\partial r} - \tfrac{1}{r^2\tan\theta}\tfrac{\partial U}{\partial \theta} \quad , \\
U_{xy} &= \tfrac{\partial}{\partial r}\left(\tfrac{1}{r}\tfrac{\partial U}{\partial \theta}\right) - \tfrac{1}{r^2}\tfrac{\partial U}{\partial \theta} \quad , \\
U_{xz} &= \tfrac{1}{r\sin\theta}\tfrac{\partial}{\partial \lambda}\left(\tfrac{\partial U}{\partial r}\right) - \tfrac{1}{r\sin\theta}\tfrac{\partial U}{\partial \lambda} \quad , \\
U_{yz} &= \tfrac{1}{r\sin\theta}\tfrac{\partial}{\partial \lambda}\left(\tfrac{1}{r}\tfrac{\partial U}{\partial \theta}\right) \quad ,
\end{aligned}$$

where $U_{xy} = U_{yx}, U_{xz} = U_{zx}$ and $U_{yz} = U_{zy}$ see, (Wermuth, 2008). The system in (8.26) is a linear inhomogeneous differential equation system with $\boldsymbol{\xi}_\ell(t)$ to be its particular solution, where the initial state vector can be set to zero i.e. $\{\boldsymbol{\xi}_\ell(t_0) = \mathbf{0}, \dot{\boldsymbol{\xi}}_\ell(t_0) = \mathbf{0}\}$, since the initial state does not depend upon the dynamic force field. The solution vector $\boldsymbol{\xi}_\ell(t)$ consists of partial derivatives of the position and velocity vectors with respect to gravity parameters p_ℓ (c.f. (8.25)), one solution vector for each of gravity parameter p_ℓ of a priori coefficients. As the general solution of the inhomogeneous differential equation is the sum of the general solution of the corresponding homogenous part and the particular solution of the inhomogeneous (Kreyszig, 2000). Therefore,

at first, consider the homogeneous part of the variational equation, i.e.

$$\ddot{\boldsymbol{\xi}}_\ell(t) = \nabla^2 U(t)\boldsymbol{\xi}(t)_\ell \quad . \tag{8.27}$$

It is a second order differential equation with two fundamental solutions, both of them having three components k, this makes it a system of equations with six unknowns which needs six solutions. Let the partial derivatives $\zeta_{k,q}(t)$, and $\dot{\zeta}_{k,q}(t) \quad q = 1, 2, \ldots, 6$, with respect to the initial values are the six fundamental solutions. The six different solution starts with six different initial conditions, as,

$$\zeta_{k,q}(t_0) = \begin{cases} \mathbf{0} \ , & k \neq q \\ 1 \ , & k = q \end{cases} \quad \text{and} \quad \dot{\zeta}_{k,q}(t_0) = \begin{cases} \mathbf{0} \ , & k \neq q \\ 1 \ , & 3 + k = q \end{cases} \tag{8.28}$$

The solutions $\boldsymbol{\zeta}_q(t)$ and $\dot{\boldsymbol{\zeta}}_q(t)$ along with initial conditions given in (8.28) defines a complete system of solutions of coupled homogeneous equations (8.27). The linear combinations of the q solutions of the homogeneous system

$$\begin{aligned} \boldsymbol{\xi}_\ell(t) &= \textstyle\sum_{q=1}^{6} \boldsymbol{\zeta}_q \boldsymbol{\alpha}_q(t) \quad , \\ \dot{\boldsymbol{\xi}}_\ell(t) &= \textstyle\sum_{q=1}^{6} \dot{\boldsymbol{\zeta}}_q \boldsymbol{\alpha}_q(t) \quad , \end{aligned} \tag{8.29}$$

with constant coefficients, $\boldsymbol{\alpha}_q$ is also a solution of the homogeneous system, which may be considered as the general solution of the homogeneous system (8.27). The solution vector $\boldsymbol{\xi}_\ell(t)$ for the inhomogeneous system (8.26) is obtained by *variation of constants* using 8.29 as the general solution of the homogeneous equation. Next section describes the usage of variation of constants method in detail.

8.4 VARIATION OF CONSTANTS

Solving the variational equations, for each satellite and each parameter p_ℓ up to degree and order 90, i.e. 8276 gravity parameters, requires much computational time. The numerical effort of the solution of the variational equations dominates the computational load of the whole gravity field determination process (Antoni and Keller, 2013). Beutler (2005) and Jäggi (2007), therefore, present the *variation of constants* method to solve the inhomogeneous system of equations (8.26).

The solution vector $\boldsymbol{\xi}_\ell(t)$ and its first derivative $\dot{\boldsymbol{\xi}}_\ell(t)$ are obtained through the method of variation of constant as a linear combination of $q = 6$ linear homogeneous solutions.

$$\begin{aligned} \boldsymbol{\xi}_\ell(t) &= \textstyle\sum_{q=1}^{6} \boldsymbol{\zeta}_q \boldsymbol{\alpha}_q(t) \quad , \\ \dot{\boldsymbol{\xi}}_\ell(t) &= \textstyle\sum_{q=1}^{6} \dot{\boldsymbol{\zeta}}_q \boldsymbol{\alpha}_q(t) \quad , \end{aligned} \tag{8.30}$$

where the functions $\boldsymbol{\zeta}_q(t)$ and $\dot{\boldsymbol{\zeta}}_q$ are solutions of the homogeneous system (8.27), the coefficients $\boldsymbol{\alpha}_q$ are functions of time t and can be written as $\boldsymbol{\alpha}^\top(t) = \begin{bmatrix} \alpha_1(t) & \alpha_2(t) \dots \alpha_q(t) \end{bmatrix}$. Therefore we can write (8.30) in matrix form

$$\begin{aligned} \boldsymbol{\xi}_\ell(t) &= \mathbf{Z}(t)\boldsymbol{\alpha}(t) \quad , \\ \dot{\boldsymbol{\xi}}_\ell(t) &= \dot{\mathbf{Z}}(t)\boldsymbol{\alpha}(t) \quad , \end{aligned} \tag{8.31}$$

where $\mathbf{Z}(t)$ and $\dot{\mathbf{Z}}(t)$ are the rectangular matrices with q columns and k rows, in which column contains the elements of the solution $\zeta_{k,q}(t)$ and $\dot{\zeta}_{k,q}(t)$ of the homogeneous system. Now according to variation of constant method, (c.f. Appendix C), a system of linear algebraic equations from the system of solutions (8.30) can be written with conditions as,

$$\frac{d}{dt}(\mathbf{Z}\boldsymbol{\alpha}) = \dot{\mathbf{Z}}\boldsymbol{\alpha} \qquad \longrightarrow \quad \mathbf{Z}\dot{\boldsymbol{\alpha}} = 0 \quad , \tag{8.32a}$$

$$\frac{d^2}{dt^2}(\mathbf{Z}\boldsymbol{\alpha}) = \ddot{\mathbf{Z}}\boldsymbol{\alpha} + \dot{\mathbf{Z}}\dot{\boldsymbol{\alpha}} \qquad \longrightarrow \quad \dot{\mathbf{Z}}\dot{\boldsymbol{\alpha}} = \mathbf{f} \quad , \tag{8.32b}$$

where $\mathbf{f} = \frac{\partial \nabla T}{\partial p_\ell}$. Because the columns of $\mathbf{Z}$ fulfills the homogenous equation, we have,

$$\ddot{\mathbf{Z}}\boldsymbol{\alpha} = \nabla^2 U \cdot \mathbf{Z}\boldsymbol{\alpha} \quad , \tag{8.33}$$

and $\mathbf{Z}\boldsymbol{\alpha}$ has to be the solution of the non-homogeneous equation, we can conclude,

$$\ddot{\mathbf{Z}}\boldsymbol{\alpha} + \dot{\mathbf{Z}}\dot{\boldsymbol{\alpha}} = \frac{d^2}{dt^2}(\mathbf{Z}\boldsymbol{\alpha}) = \nabla^2 U \mathbf{Z}\boldsymbol{\alpha} + \frac{\partial \nabla T(\mathbf{x}_0)}{\partial p_\ell} \quad , \tag{8.34}$$

which leads to

$$\dot{\mathbf{Z}}\dot{\boldsymbol{\alpha}} = \frac{\partial \nabla T(\mathbf{x}_0)}{\partial p_\ell} \quad . \tag{8.35}$$

Combining the last solution with 8.32a, we get,

$$\tilde{\mathbf{Z}}\dot{\boldsymbol{\alpha}} = \mathrm{Y}(t) \quad , \tag{8.36}$$

$$\text{where, } \tilde{\mathbf{Z}} = \begin{bmatrix} \mathbf{Z} \\ \dot{\mathbf{Z}} \end{bmatrix} \text{ and } \mathrm{Y}(t) = \begin{bmatrix} 0 \\ \frac{\partial \nabla T(\mathbf{x}_0)}{\partial p_\ell} \end{bmatrix} \quad .$$

After rearranging the equation becomes

$$\dot{\boldsymbol{\alpha}}(t) = \tilde{\mathbf{Z}}(t)^{-1}\mathrm{Y}(t) \quad , \tag{8.37}$$

while integration of $\dot{\boldsymbol{\alpha}}(t)$ leads us to $\boldsymbol{\alpha}(t)$ as,

$$\boldsymbol{\alpha}(t) = \int \dot{\boldsymbol{\alpha}} dt \quad . \tag{8.38}$$

Simpson's rule, a numerical integration method is used to solve the first order differential equation (8.38). The following section presents the details.

8.5 NUMERICAL INTEGRATION

In this section the method of numerical integration is presented, to solve the problem like (8.38). The idea of a numerical evaluation is to solve the integral,

$$I = \int_a^b f(t)dt \quad , \tag{8.39}$$

with, a and b are the given integration limits and f is a given *analytical or empirical function.* Numerical integration helps to solve the integrals whose analytical evaluation would be difficult or impossible or whose integrand, as in (8.38) is an empirical function given by numeric values (Kreyszig, 2000). Most popular numerical integration methods are *rectangular rule, trapezoidal rule, Simpson's rule* and *Gauss quadrature.* Since the numerical values in (8.38) are given at an equal interval, therefore Gauss quadrature method can not be used. First three methods evaluate at equal intervals, with constant approximation results in the Rectangular rule, linear in the trapezoidal rule and quadratic results in Simpson's rule. Simpson's rule is computationally time consuming however provide sufficiently accurate results with respect to former two methods.

Simpson's rule divides the interval of integrations $a \leq t \geq b$ into even numbers of equal sub-intervals, suppose $n = 2m$ and the interval between them is $h = (b-a)/2m$ which starts from a or t_0 and goes up to b or t_m as, $t_0, \quad t_1, \quad t_2, \quad \ldots, \quad t_{2m-2}, \quad t_{2m-1}, \quad t_m$, then for the first two intervals, i.e. from t_0 to t_2 the integral can be written as,

$$\int_{t_0}^{t_2} f(t)dt \approx h\left(\frac{1}{3}f_0 + \frac{4}{3}f_1 + \frac{1}{3}f_2\right) \quad . \tag{8.40}$$

A similar expression can be written for the next two sub-intervals, i.e. from t_2 to t_4 and so on till t_{2m-2} to t_{2m} (c.f. (Kreyszig, 2000)). So the summation of all m formulas lead to the Simpson's rule,

$$\int_a^b f(t)dt \approx \frac{h}{3}\left(f_0 + 4f_1 + 2f_2 + 4f_3 + \cdots + 2f_{2m-2} + 4f_{2m-1} + f_{2m}\right) \quad . \tag{8.41}$$

Inserting the values of $\boldsymbol{\alpha}(t)$ back to the (8.31) yields the solutions $\{\boldsymbol{\xi}(t), \dot{\boldsymbol{\xi}}(t)\}$ as,

$$\begin{aligned} \boldsymbol{\xi}_\ell(t) &= \mathbf{Z}(t)\boldsymbol{\alpha}(t) = \begin{bmatrix}\xi_x(t) & \xi_y(t) & \xi_z(t)\end{bmatrix}^\top \quad , \\ \dot{\boldsymbol{\xi}}_\ell(t) &= \dot{\mathbf{Z}}(t)\boldsymbol{\alpha}(t) = \begin{bmatrix}\dot{\xi}_x(t) & \dot{\xi}_y(t) & \dot{\xi}_z(t)\end{bmatrix}^\top \quad . \end{aligned} \tag{8.42}$$

Above relations leads us to the end of the theoretical background of the variational equation. The solution $\{\boldsymbol{\xi}(t), \dot{\boldsymbol{\xi}}(t)\}$ of the variational equations 8.26 along with the simulated range-rates are used to formally write the observation equation of the gravity recovery process in Section 8.7. The process to simulate the range-rates is described in the following section.

8.6 RANGE-RATES

This section states the sequential processing to simulate the range-rates observations through orbit determination. The simulation starts with the transformation of Osculating Kepler elements given in Table 8.1 using the equations given in Box 1 in (Austen and Grafarend, 2001) into the position $\mathbf{x}_A(t_0), \mathbf{x}_B(t_0)$ and velocity $\dot{\mathbf{x}}_A(t_0), \dot{\mathbf{x}}_B(t_0)$ vectors, which act as the initial values for the numerical orbital integration of satellites A and B, respectively. Orbital integration, firstly, yields true orbit $\mathbf{x}_A(t), \dot{\mathbf{x}}_A(t), \mathbf{x}_B(t), \dot{\mathbf{x}}_B(t)$ for GRACE satellites A and B for (t) epochs, using the true field followed by the integration of a priori orbit $\mathbf{x}_{0A}(t)$, $\dot{\mathbf{x}}_{0A}(t)$, $\mathbf{x}_{0B}(t)$, $\dot{\mathbf{x}}_{0B}(t)$ using the a priori field. The simulation yields the orbit in arcs of 90 min with the frequency of tan sec. The last position and velocity values, of the previous arc, act as the initial values for the next arc. Therefore, each arc is composed of 540 observations. While 480 such arcs, to solve 8277 unknown SH coefficients, make this problem an overdetermined system of equations. In this way, the orbits are computed for one month period.

True Position and velocity vectors of both satellites yield true $\dot{\rho}(t)$ range-rates. Similarly, a priori Position and velocity vectors of both satellites yield a priori $\dot{\rho}_0(t)$ range-rates. The process for computing range-rates from position and velocity vector is given in Section 8.1. Figure 8.2 summarizes the process as a flow chart. The difference between the two range-rates i.e $\delta\dot{\rho}(t) = \dot{\rho}(t) - \dot{\rho}_0(t)$ acts as the observation y of the least square adjustment process, as discussed in Section 8.7. The noise of the level of 1×10^{-7}m is added to the error-free simulated range-rate observations to imitate GRACE like situation. The next task, discussed in the next section, is to formulate an expression for the partial derivatives of position and velocity with respect to the a priori SH coefficients. The expression gives birth to the design matrix and the vector of unknown coefficients for the least squares adjustment.

Table 8.1: Initial osculating KE of the two GRACE satellites for simulation

KE	Sat. A	Sat. B
semi major axis [m]	6838136.6	6838136.6
eccentricity	0.002	0.002
inclination [°]	89	89
right ascension of ascending node [°]	0	0
argument of perigee [°]	0	0
mean anomaly [°]	+1	−1

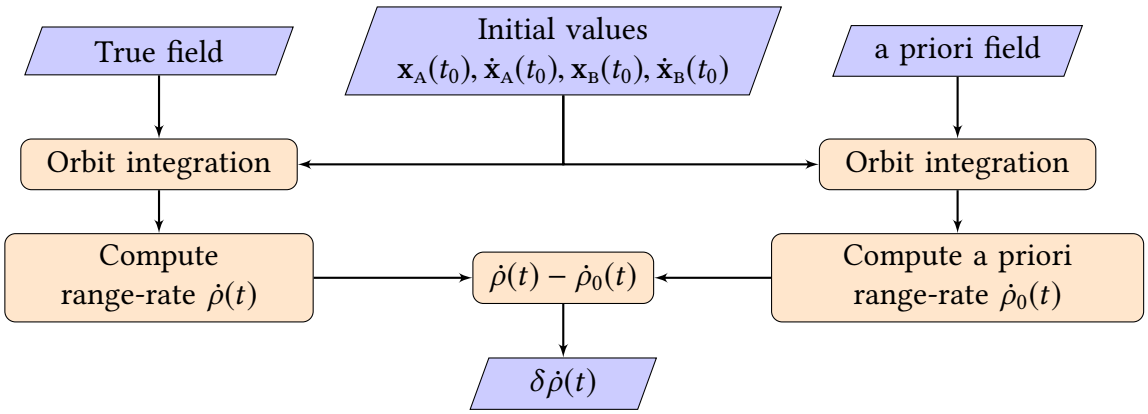


Figure 8.2: Flow chart for computing range-rates.

8.7 COEFFICIENT ESTIMATION

Gravity field recovery from range-rates measurements can be considered as a differential orbit improvement process. Therefore, $\dot{\rho}(t)$ is expressed as truncated Taylor series with respect to p_ℓ about $\dot{\rho}_0(t)$. (Jäggi et al., 2010). Let us formally express it as,

$$\dot{\rho}(t) = \dot{\rho}_0(t) + \left[\frac{\partial\dot{\rho}_0}{\partial p_1}(t),\ \ldots,\ \frac{\partial\dot{\rho}_0}{\partial p_L}(t)\right]\begin{bmatrix}\Delta p_1\\ \vdots\\ \Delta p_L\end{bmatrix}\quad , \tag{8.43}$$

so the final observation equation becomes,

$$\delta\dot{\rho}(t) = \left[\frac{\partial\dot{\rho}_0}{\partial p_1}(t),\ \ldots,\ \frac{\partial\dot{\rho}_0}{\partial p_L}(t)\right]\begin{bmatrix}\Delta p_1\\ \vdots\\ \Delta p_L\end{bmatrix}\quad , \tag{8.44}$$

where $\delta\dot{\rho}(t) = \dot{\rho}(t) - \dot{\rho}_0(t)$. For simplicity, let us drop (t) and the index 0, for each p_ℓ we write

$$\frac{d\dot{\rho}}{dp_\ell} = \left\langle\frac{\partial\dot{\rho}}{\partial\mathbf{x}_A}|\frac{\partial\mathbf{x}_A}{\partial p_\ell}\right\rangle + \left\langle\frac{\partial\dot{\rho}}{\partial\dot{\mathbf{x}}_A}|\frac{\partial\dot{\mathbf{x}}_A}{\partial p_\ell}\right\rangle + \left\langle\frac{\partial\dot{\rho}}{\partial\mathbf{x}_B}|\frac{\partial\mathbf{x}_B}{\partial p_\ell}\right\rangle + \left\langle\frac{\partial\dot{\rho}}{\partial\dot{\mathbf{x}}_B}|\frac{\partial\dot{\mathbf{x}}_B}{\partial p_\ell}\right\rangle\quad , \tag{8.45}$$

or after using definition $\boldsymbol{\xi} = \frac{\partial\mathbf{x}}{\partial p_\ell}$ from (8.25)

$$\frac{d\dot{\rho}}{dp_\ell} = \left\langle\frac{\partial\dot{\rho}}{\partial\mathbf{x}_A}|\boldsymbol{\xi}_A\right\rangle + \left\langle\frac{\partial\dot{\rho}}{\partial\dot{\mathbf{x}}_A}|\dot{\boldsymbol{\xi}}_A\right\rangle + \left\langle\frac{\partial\dot{\rho}}{\partial\mathbf{x}_B}|\boldsymbol{\xi}_B\right\rangle + \left\langle\frac{\partial\dot{\rho}}{\partial\dot{\mathbf{x}}_B}|\dot{\boldsymbol{\xi}}_B\right\rangle\quad , \tag{8.46}$$

where $\frac{\partial\dot{\rho}}{\partial\mathbf{x}_A}$, $\frac{\partial\dot{\rho}}{\partial\dot{\mathbf{x}}_A}$, $\frac{\partial\dot{\rho}}{\partial\mathbf{x}_B}$ and $\frac{\partial\dot{\rho}}{\partial\dot{\mathbf{x}}_B}$ are the partial derivatives of the range-rate, which is the function of both position and velocity (Kim, 2000). After inserting the value $\dot{\rho} = \langle\delta\dot{\mathbf{x}}|\mathbf{e}\rangle$ from (8.10) we can write,

$$\frac{d\dot{\rho}}{dp_\ell} = \left\langle\frac{\partial\langle\delta\dot{\mathbf{x}}|\mathbf{e}\rangle}{\partial\mathbf{x}_A}|\boldsymbol{\xi}_A\right\rangle + \left\langle\frac{\partial\langle\delta\dot{\mathbf{x}}|\mathbf{e}\rangle}{\partial\dot{\mathbf{x}}_A}|\dot{\boldsymbol{\xi}}_A\right\rangle + \left\langle\frac{\partial\langle\delta\dot{\mathbf{x}}|\mathbf{e}\rangle}{\partial\mathbf{x}_B}|\boldsymbol{\xi}_B\right\rangle + \left\langle\frac{\partial\langle\delta\dot{\mathbf{x}}|\mathbf{e}\rangle}{\partial\dot{\mathbf{x}}_B}|\dot{\boldsymbol{\xi}}_B\right\rangle\quad , \tag{8.47}$$

or

$$\frac{d\dot{\rho}}{dp_\ell} = \left\langle \delta\dot{\mathbf{x}} \frac{\partial}{\partial \mathbf{x}_A}(\mathbf{e}) \,|\boldsymbol{\xi}_A\right\rangle + \left\langle \mathbf{e}\frac{\partial}{\partial \dot{\mathbf{x}}_A}(\delta\dot{\mathbf{x}}) \,|\dot{\boldsymbol{\xi}}_A\right\rangle + \left\langle \delta\dot{\mathbf{x}} \frac{\partial}{\partial \mathbf{x}_B}(\mathbf{e}) \,|\boldsymbol{\xi}_B\right\rangle + \left\langle \mathbf{e}\frac{\partial}{\partial \dot{\mathbf{x}}_B}(\delta\dot{\mathbf{x}}) \,|\dot{\boldsymbol{\xi}}_B\right\rangle \quad , \tag{8.48}$$

insert $\delta\dot{\mathbf{x}} = \dot{\mathbf{x}}_B - \dot{\mathbf{x}}_A$ from (8.3)

$$\frac{d\dot{\rho}}{dp_\ell} = \left\langle \delta\dot{\mathbf{x}} \frac{\partial}{\partial \mathbf{x}_A}(\mathbf{e}) \,|\boldsymbol{\xi}_A\right\rangle + \left\langle \mathbf{e}\frac{\partial}{\partial \dot{\mathbf{x}}_A}(\dot{\mathbf{x}}_B - \dot{\mathbf{x}}_A) \,|\dot{\boldsymbol{\xi}}_A\right\rangle + \left\langle \delta\dot{\mathbf{x}} \frac{\partial}{\partial \mathbf{x}_B}(\mathbf{e}) \,|\boldsymbol{\xi}_B\right\rangle + \left\langle \mathbf{e}\frac{\partial}{\partial \dot{\mathbf{x}}_B}(\dot{\mathbf{x}}_B - \dot{\mathbf{x}}_A) \,|\dot{\boldsymbol{\xi}}_B\right\rangle \quad . \tag{8.49}$$

For simplification, derive the terms $\frac{\partial}{\partial \mathbf{x}_A}(\mathbf{e})$, $\frac{\partial}{\partial \dot{\mathbf{x}}_A}(\dot{\mathbf{x}}_B - \dot{\mathbf{x}}_A)$, $\frac{\partial}{\partial \mathbf{x}_B}(\mathbf{e})$, and $\frac{\partial}{\partial \dot{\mathbf{x}}_B}(\dot{\mathbf{x}}_B - \dot{\mathbf{x}}_A)$. Firstly, consider $\frac{\partial}{\partial \mathbf{x}_A}(\mathbf{e})$, where $\mathbf{e} = \delta\mathbf{x}/\rho$, from (8.5),

$$\frac{\partial}{\partial \mathbf{x}_A}(\mathbf{e}) = \frac{\partial}{\partial \mathbf{x}_A}\left(\frac{\delta\mathbf{x}}{\rho}\right) = \frac{\rho\frac{\partial}{\partial \mathbf{x}_A}(\delta\mathbf{x}) - \left\langle \delta\mathbf{x}|\frac{\partial\rho}{\partial \mathbf{x}_A}\right\rangle}{\rho^2} \quad , \tag{8.50}$$

inserting the value $\delta\mathbf{x} = \mathbf{x}_B - \mathbf{x}_A$ from (8.3)

$$= \frac{1}{\rho^2}\left(\rho\frac{\partial}{\partial \mathbf{x}_A}(\mathbf{x}_B - \mathbf{x}_A) - \left\langle \delta\mathbf{x}|\frac{\partial\rho}{\partial \mathbf{x}_A}\right\rangle\right) \quad . \tag{8.51}$$

Since from (8.4) ρ is $= \sqrt{\langle\delta\mathbf{x}|\delta\mathbf{x}\rangle}$, therefore,

$$\frac{\partial\rho}{\partial \mathbf{x}_A} = -\frac{1}{\rho}(\mathbf{x}_B - \mathbf{x}_A) = -\frac{\delta\mathbf{x}}{\rho} = -\mathbf{e}, \quad \text{as } \mathbf{e} = \delta\mathbf{x}/\rho, \text{ from (8.5)}, \tag{8.52}$$

$$\frac{\partial\mathbf{x}_A}{\partial\mathbf{x}_A} = 1 \quad \text{and} \quad \frac{\partial\mathbf{x}_B}{\partial\mathbf{x}_A} = 0 \quad , \tag{8.53}$$

by inserting these results back into (8.51) we get,

$$\frac{\partial}{\partial \mathbf{x}_A}(\mathbf{e}) = \quad \frac{1}{\rho^2}\left(-\rho + \langle\delta\mathbf{x}|\mathbf{e}\rangle\right) \tag{8.54}$$

multiply both side by $\delta\dot{\mathbf{x}}$,

$$\delta\dot{\mathbf{x}}\frac{\partial}{\partial \mathbf{x}_A}(\mathbf{e}) = -\frac{1}{\rho}\left(\delta\dot{\mathbf{x}} - \left\langle \delta\dot{\mathbf{x}}|\frac{\delta\mathbf{x}}{\rho}\right\rangle \mathbf{e}\right) \quad , \tag{8.55}$$

insert $\dot{\rho} = \left\langle \delta\dot{\mathbf{x}}|\frac{\delta\mathbf{x}}{\rho}\right\rangle$ from (8.11) and finally

$$\boxed{\delta\dot{\mathbf{x}}\frac{\partial}{\partial \mathbf{x}_A}(\mathbf{e}) = -\frac{1}{\rho}\left(\delta\dot{\mathbf{x}} - \dot{\rho}\mathbf{e}\right) \quad .} \tag{8.56}$$

Secondly, consider $\frac{\partial}{\partial \dot{\mathbf{x}}_A}(\dot{\mathbf{x}}_B - \dot{\mathbf{x}}_A)$ from (8.49). Since, $\frac{\partial\dot{\mathbf{x}}_A}{\partial\dot{\mathbf{x}}_A} = 1, \quad \frac{\partial\dot{\mathbf{x}}_B}{\partial\dot{\mathbf{x}}_A} = 0$,

$$\boxed{\frac{\partial}{\partial \dot{\mathbf{x}}_A}(\dot{\mathbf{x}}_B - \dot{\mathbf{x}}_A) = -1 \ .} \tag{8.57}$$

Thirdly, consider $\frac{\partial}{\partial \mathbf{x}_B}(\mathbf{e})$, where $\mathbf{e} = \delta\mathbf{x}/\rho$, from (8.5),

$$\frac{\partial}{\partial \mathbf{x}_B}(\mathbf{e}) = \frac{\partial}{\partial \mathbf{x}_B}\left(\frac{\delta\mathbf{x}}{\rho}\right) = \left(\frac{\rho\frac{\partial}{\partial \mathbf{x}_B}(\delta\mathbf{x}) - \left\langle \delta\mathbf{x}|\frac{\partial\rho}{\partial \mathbf{x}_B}\right\rangle}{\rho^2}\right) \quad , \tag{8.58}$$

inserting the value $\delta\mathbf{x} = \mathbf{x}_B - \mathbf{x}_A$ from (8.3)

$$\frac{\partial}{\partial \mathbf{x}_B}(\mathbf{e}) = \frac{1}{\rho^2}\left(\rho\frac{\partial}{\partial \mathbf{x}_B}(\mathbf{x}_B - \mathbf{x}_A) - \left\langle \delta\mathbf{x}|\frac{\partial\rho}{\partial \mathbf{x}_B}\right\rangle\right) \quad . \tag{8.59}$$

Since from (8.4) ρ is $= \sqrt{\langle\delta\mathbf{x}|\delta\mathbf{x}\rangle}$, therefore,

$$\frac{\partial\rho}{\partial \mathbf{x}_B} = \frac{1}{\rho}(\mathbf{x}_B - \mathbf{x}_A) = \frac{\delta\mathbf{x}}{\rho} = \mathbf{e}, \quad \text{as } \mathbf{e} = \delta\mathbf{x}/\rho, \text{ from (8.5)}, \tag{8.60}$$

$$\frac{\partial \mathbf{x}_A}{\partial \mathbf{x}_B} = 0 \quad \text{and} \quad \frac{\partial \mathbf{x}_B}{\partial \mathbf{x}_B} = 1 \quad , \tag{8.61}$$

by inserting these results back into (8.59) we get,

$$\frac{\partial}{\partial \mathbf{x}_B}(\mathbf{e}) = \quad \frac{1}{\rho^2}(\rho - \langle\delta\mathbf{x}|\mathbf{e}\rangle) \tag{8.62}$$

multiply both side by $\delta\dot{\mathbf{x}}$,

$$\delta\dot{\mathbf{x}}\frac{\partial}{\partial \mathbf{x}_B}(\mathbf{e}) = \frac{1}{\rho}\left(\delta\dot{\mathbf{x}} - \left\langle \delta\dot{\mathbf{x}}|\frac{\delta\mathbf{x}}{\rho}\right\rangle\mathbf{e}\right) \quad , \tag{8.63}$$

insert $\dot{\rho} = \left\langle \delta\dot{\mathbf{x}}|\frac{\delta\mathbf{x}}{\rho}\right\rangle$ from (8.11) and finally

$$\boxed{\delta\dot{\mathbf{x}}\frac{\partial}{\partial \mathbf{x}_B}(\mathbf{e}) = \frac{1}{\rho}(\delta\dot{\mathbf{x}} - \dot{\rho}\mathbf{e}) \quad .} \tag{8.64}$$

Lastly, consider $\frac{\partial}{\partial \dot{\mathbf{x}}_B}(\dot{\mathbf{x}}_B - \dot{\mathbf{x}}_A)$ from the right hand side of (8.49). Since, $\frac{\partial \dot{\mathbf{x}}_A}{\partial \dot{\mathbf{x}}_A} = 1, \quad \frac{\partial \dot{\mathbf{x}}_B}{\partial \dot{\mathbf{x}}_A} = 0$,

$$\boxed{\frac{\partial}{\partial \dot{\mathbf{x}}_B}(\dot{\mathbf{x}}_B - \dot{\mathbf{x}}_A) = 1 \ .} \tag{8.65}$$

by inserting the four terms form 8.56, 8.57, 8.64 and 8.65 in (8.49) we can write

$$\frac{d\dot{\rho}}{dp_\ell} = -\left\langle \frac{1}{\rho}(\delta\dot{\mathbf{x}} - \mathbf{e}\dot{\rho})\,|\boldsymbol{\xi}_A\right\rangle - \left\langle \mathbf{e}|\dot{\boldsymbol{\xi}}_A\right\rangle + \left\langle \frac{1}{\rho}(\delta\dot{\mathbf{x}} - \mathbf{e}\dot{\rho})\,|\boldsymbol{\xi}_B\right\rangle + \left\langle \mathbf{e}|\dot{\boldsymbol{\xi}}_B\right\rangle \quad , \tag{8.66}$$

Note that the quantities for satellite A and satellite B are same but with opposite signs. Solving (8.66) for each p_ℓ leads to the formulation of the design matrix $\mathbf{A}$. The orbit integration in Section 8.6 provides the observation $\mathbf{y}$ in the form of $\delta\dot{\rho}(t)$. From equation 8.1, $\mathbf{p}^{Dec}_{2016}$ represents the vector of unknown SH coefficients and here for simplicity replaced by $\mathbf{p}$ So, the final observation equation is,

$$\mathbf{y} = \mathbf{A}\mathbf{p} + \epsilon, \text{where } \epsilon \text{ is the error vector} \quad , \tag{8.67}$$

which leads us to the following system of normal equations (Koch and Kusche, 2002) (Elsaka et al., 2014)

$$\mathbf{N}\mathbf{p} = \mathbf{b} \;\text{ with }\; \mathbf{N} = \mathbf{A}^\top\mathbf{W}\mathbf{A} \;\text{ and }\; \mathbf{b} = \mathbf{A}^\top\mathbf{W}\mathbf{y} \quad , \tag{8.68}$$

where $\mathbf{W}$ is the weight matrix and here it is identity matrix. The normal equations are directly accumulated from the individual arc-wise blocks as,

$$\mathbf{N} = \sum_{i=1}^{m} \mathbf{A}_i^\top \mathbf{W}_i \mathbf{A}_i \quad \text{and} \quad \mathbf{b} = \sum_{i=1}^{m} \mathbf{A}_i^\top \mathbf{W}_i \mathbf{y} \quad . \tag{8.69}$$

where, $m = 480$ is the number of 90 min arcs in 30 days. The solution of the normal equations yields the estimation of the unknown parameters as

$$\mathbf{p} = \mathbf{N}^{-1}\mathbf{b} = (\mathbf{A}^\top\mathbf{W}\mathbf{A})^{-1}\mathbf{A}^\top\mathbf{W}\mathbf{y} \quad . \tag{8.70}$$

The output is then a set of estimated SH coefficients. Whereas, adding them back to the a priori SH coefficients, gives the recovered field. In the next section a detailed analysis of the three different recovered fields are presented.

8.8 RESULTS

In this chapter, a closed loop simulation for gravity recovery has been presented. The simulation aims to show that if we recover a reduced field, the recovered coefficients are more stable since they have a less formal error. To establish this point gravity recovery simulation is used in three different scenarios, discussed in the following paragraphs.

The process starts with the original GRACE field for December 2016. Clustering and classification segregate the data into two classes, nonessential and essential. Afterwards, a prediction process bifurcates the essential class into predictable and unpredictable coefficients using ANN. For the first scenario, a complete set of SH coefficients is recovered. In the second and third scenarios, only essential coefficients and unpredictable coefficients are recovered, respectively. At the end of three scenarios, a comparison of output unveils the improvement in the recovered field. Formal error plots, curves and their relative improvement, EWH maps, their differences and some statistics based on their standard deviation are used as the comparison tools.

For the first scenario, the simulation recovers the complete field up to degree 90 and order 90. A flowchart in Figure 8.3 describes the process.

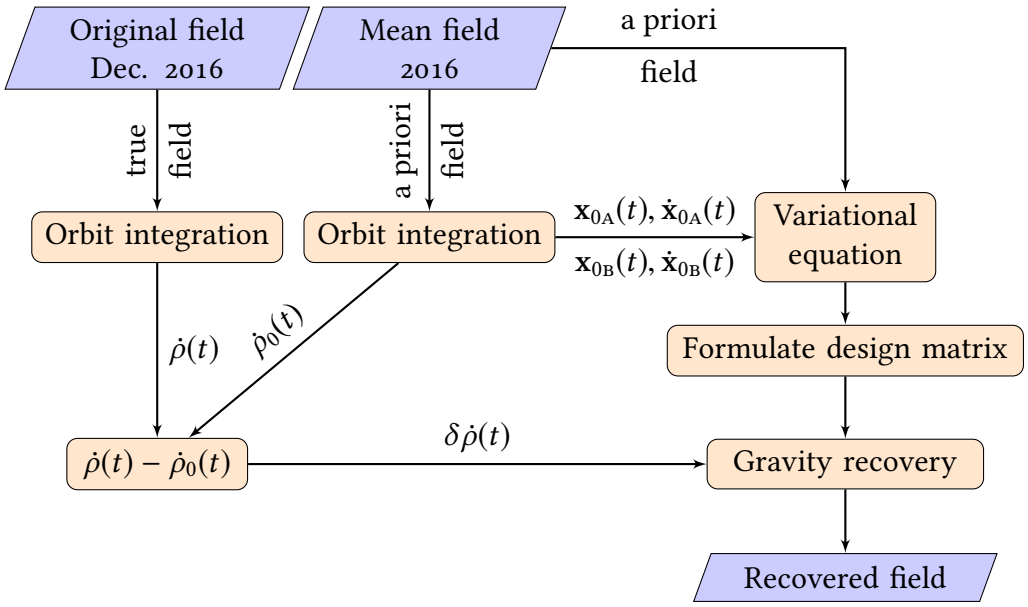


Figure 8.3: Flowchart of gravity recovers simulation. Case 1. recovery of complete SH spectrum up to degree 90 and order 90.

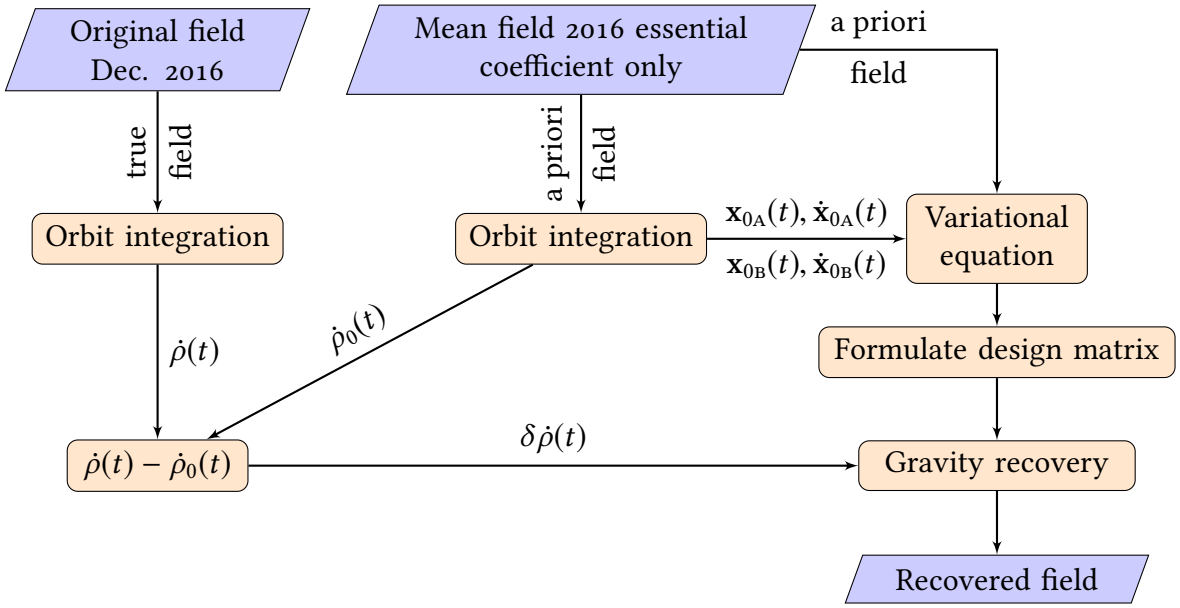
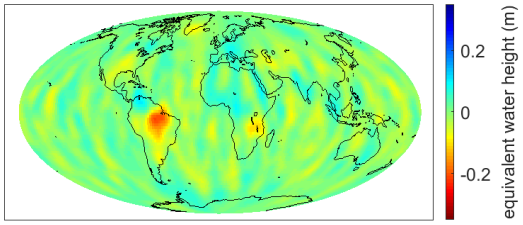


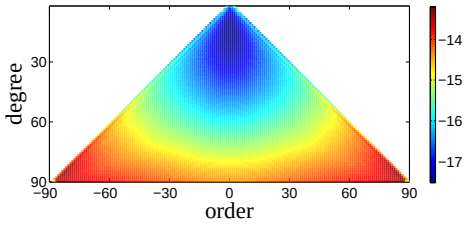
Figure 8.4: Flowchart of gravity recovers simulation. Case 2. recovery of essential coefficients only.

First of all the orbit integration using the true field gives the true range-rates. On the other hand, the integration process using the a priori field gives a priori range and partials derivatives of the orbit. The difference between the two range-rates gives the observation vector. Using this observation vector and the matrix of the partial derivatives which acts as design matrix, least squares adjustment estimates the vector of unknown coefficients or recovered field. Figure 8.5(a)

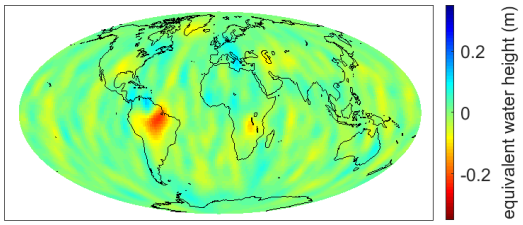
and 8.5(b) represent the EWH map and the formal error in SC format of the recovered field, respectively where Figure 8.6 displays the degree variance of the recovered field in red color, which acts as the reference point to measure the improvement in the next two scenarios.



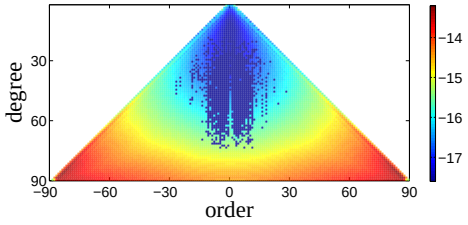
(a) EWH, full spectrum recovered.



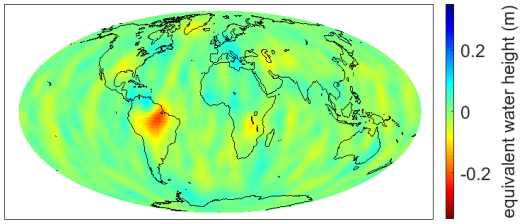
(b) formal error, full spectrum recovered.



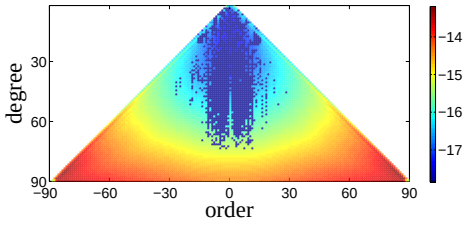
(c) EWH, only essential recovered.



(d) formal error, only essential recovered



(e) EWH, only unpredictable recovered + predictable through ANN.



(f) formal error, only unpredictable recovered.

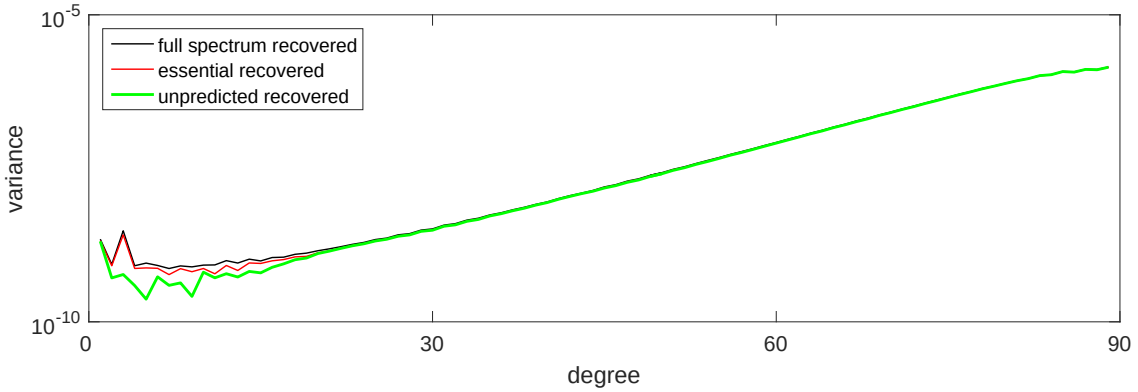
Figure 8.5: Comparisons of three scenarios using EWH maps and formal errors in meter

For the second scenario, the simulation recovers only the essential coefficients. A flowchart in Figure 8.4 represents the process, according to which besides getting the true range-rates the nonessential coefficients of the a priori field are removed and the orbit and the partials are integrated from the remaining essential coefficients of the a priori field. The simulation in the end, therefore, recovers the essential coefficients. Figure 8.5(c) and 8.5(d) display the EWH map and the formal error in the SC format of the recovered field, respectively for the second scenario and the blue curve in Figure 8.6 represents degree variance of second scenario.

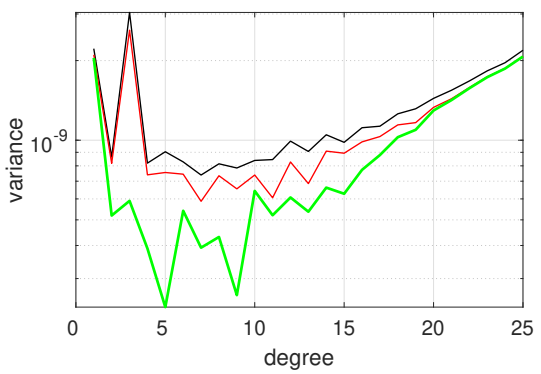
For the third scenario, the simulation follows the same procedure as the second scenario but recovers only the unpredictable coefficients. Figure 8.5(e) and 8.5(f) display the EWH map and

the formal error in SC format of the recovered field, respectively for the third scenario, where the green curve in Figure 8.6 is presenting its degree variance. Where, Figure 8.7 shows the difference of EWH map and recovered field of true field and all three scenarios.

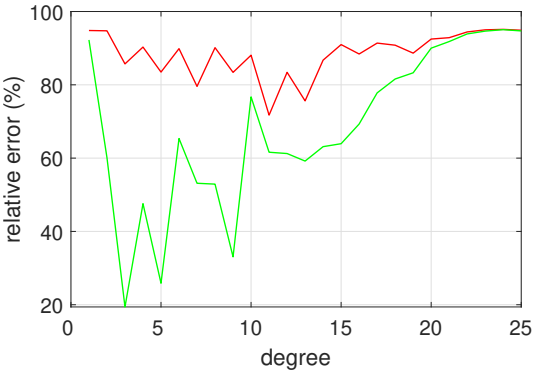
Three EWH maps given in Figure 8.5(a), 8.5(c) and 8.5(e), are identical. For example, the red feature which shows gravity decreases due to water drainage out of the Amazon basin is almost the same. Which shows in case of reduced field recovery the information is not lost.



(a) Degree variance up to 90 degree.



(b) Degree variance up to 25 degree (closeup).

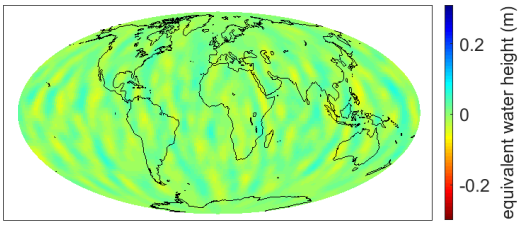


(c) Relative error up to 25 degree.

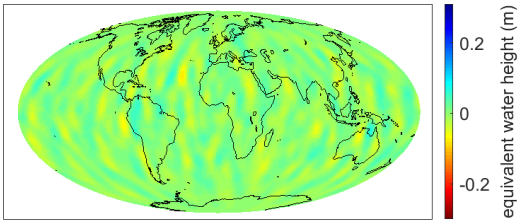
Figure 8.6: Degree variance of the three cases. Firstly, full spectrum is recovered, secondly only the essential coefficients are recovered and thirdly only the unpredictable coefficients are recovered.

Three curves in Figure 8.6(a) show that the formal error reduces if the simulation has to recover fewer coefficients during the gravity recovery process. Here the benefit of the increase of the degree of freedom improves and stabilizes the output. The effect is also clear from Figure 8.5 in three formal error SC plots, where the area covered by blue has extended towards the lower frequency values. Figure 8.6(c) shows that the error percentage for each degree has been decreased in the third scenario where only unpredictable coefficients are recovered. The decrease in the formal error in the low degree range as shown in Figure 8.6(c) is an excellent sign for recovered

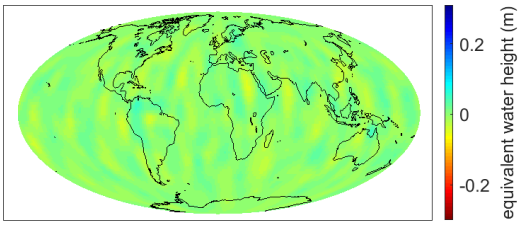
fields since the gravity signal of all important river basins such as Amazon, California, Colorado, Congo and Danube lie in this region.



(a) Change in EWH, full spectrum recovered.



(b) Change in EWH, only essential recovered.



(c) Change in EWH, only unpredictable recovered + predictable through ANN.

Figure 8.7: EWH difference maps between true field and the recovered fields in the three scenarios

Difference between the EWH maps of the true field and the recovered fields of the three scenarios given in Figure 8.7 show that the difference decreases as the number of coefficients in the recovered field decreases.

The statistical comparison of the two cases, i.e. full recovery and the reduced recovery shows that the ratio of the row-wise mean to the column-wise mean is equal to one. The ratio of the row-wise standard deviation to the column-wise standard deviation of the full recovery is 1.325, whereas in case of the reduced solution it is 1.304. While the norm of the full recovery field is 0.1216 and the reduced field recovery field is 0.1204. In conclusion, the comparison shows that the gravity variation information does not lose in case of reduced field recovery and we can ig-

nore the nonessential and predictable coefficients during the recovery process. The next chapter summarizes the whole process of data mining for GRACE monthly solutions.

SUMMARY

9

GRACE, a tandem constellation, was launched in March 2002, since then the range-rates between the two satellites have been inverted to recover the gravity field in the form of SH coefficients, once per month, by GFZ. The gravity field in the form of SH coefficients are available from three official data centres, i.e.

1. GFZ (GeoforschungsZentrum Potsdam),
2. CSR (Center for Space Research at University of Texas, Austin) and
3. JPL (Jet Propulsion Laboratory).

In this study, the data form GFZ is used because it provides coefficients up to degree 90 and order 90 along with their standard deviation. The first data file has SH coefficients of April 2002 and the last June 2017, i.e. 183 months, however, data for 20 different months are missing due to some technical failures and batteries issues. Thus the data files for 163 months are available. Each file has SH coefficients up to degree and order 90/90, so the total number of coefficients, in each file, is 8281. GRACE dataset unveils the monthly variation in the gravity field of the Earth. Due to gravity variation, the magnitude of several coefficients changes significantly, called *essential coefficients*. On the other hand, the magnitude of several others do not change significantly, called *nonessential coefficients*. Succinctly stated, essential possess the most and nonessential possess very little information of the gravity variations. If we recover only the essential coefficients using the same one-month range-rates dataset, we can reduce the formal error and get more stable gravity field solutions. In other words, we get the benefit of increasing the degree of freedom while reducing the number of recoverable coefficients. In the study, several data mining methods are presented to identify essential and nonessential coefficients.

Data mining is shaping up the relation between the data and the information. For the last hundred years, since the advent of sensors to measure the phenomenon, the size of available data is growing. It has been observed that in some cases the information is lost in such a significant

amount of data. Besides that, there is some related information hidden in the data which requires the efficient time analysis of the data. Data mining techniques have been successfully used to capture the information from large datasets. In the study, data mining methods are used for clustering, data mining and prediction of the grace datasets. The use of data mining techniques to study the behavior of the SH coefficients of the gravity solution is entirely new and presented firstly in the study. The following list states the details as,

i. the idea of classifying the coefficient on the bases of their behavior is entirely new. The earlier studies that represent the relationship between the coefficients in the bases of their behavior includes,
   a) while investigating the stripes problem in the grace data, the correlation between some coefficients has been found at higher orders (Swenson and Wahr, 2006),
   b) coefficients are classified on the bases of some geometric properties such as sign changes, total Euclidean length and convex hull area of the coefficients is created (Piretzidis et al., 2016),
   c) during several studies coefficients are classified on the bases of noise accumulation, such as (Reigber, 1974) and (Devaraju, 2015),

ii. present study exploits several different and independent ways to classify the grace dataset, including,
   a) k-means clustering, which utilizes the density of the data points by measuring the L_2 distance in Euclidean space or cosine similarity in vector space,
   b) threshold bifurcates the data on the bases of yearly variance,
   c) k neatest neighbor which works in the density measurement using the closest neighbor method,
   d) ANN deals with the historical data to train a network of neurons using weights and later perform as a supervised processing unit,

iii. the most exciting part of the study is the ANN-based prediction of SH coefficients. The most challenging part of the prediction is to optimize the output for this purpose three different methods are investigated.

To state the sequential process succinctly, consider the clustering. k-means clustering proves the underlying hypothesis that there exist two distinct groups. Threshold classification validates two classes and helps us to label them on the bases of yearly variance. The class which has higher variance contains the significant information of the gravity variance and vice versa. During the

threshold classification, the yearly variance has been computed, and an optimal value between the maximum and minimum limits of the yearly variance is selected as the threshold. The coefficients having the variance below the threshold value are the nonessential coefficients and coefficients having variance higher than the threshold value are essential.

Suppose we replace the nonessential coefficients with their yearly mean before using it as the a priori field in the gravity recovery process, are introducing omission error. The omission error must stay within the standard deviations limits of the original GRACE monthly solutions. If the replacement process fulfills this requirement, then we will not recover the identified nonessential coefficients during the gravity recovery process. During the 16 year analysis of coefficients through classification, it has been observed that there are some coefficients which belong to one class in one year and belong to other class in some other year. They have been classed as unclassified coefficients. k nearest neighbor is used to classify them. Afterwards, ANN has been used as an independent technique to verify the classification results.

ANN is a modern machine learning technique. It uses several primary mathematical techniques, including, linear algebra, probability, dynamic system, and optimization. Besides classification, it has been used for prediction purposes. A set of predictable coefficients has been identified among the essential coefficients. Hence the list of essential coefficients is reduced and therefore the recovered solution is called as the reduced GRACE solution.

At the end of the study, the variational equation method is used to recover the gravity field for three different cases. For the first case, a complete set of coefficients is recovered. In the second case only the essential coefficients are recovered and in the third case, a set of reduced essential coefficients is recovered. The results of the three recovered fields are compared using the degree variance curves of the formal error and the EWH maps and their changes. The results show that the reduced gravity field is more stable while having the low formal error and the statistical comparison of the recovered field show that the information is not lost.

In the end, it is worthwhile to state that the potential of the ANN can be further explored for SH coefficients prediction. Among 8277 coefficients, only 245 coefficients could be predicted, by the end of the study. The number of predictable coefficients can be increased. The benefits of prediction include,

i. filling the data gaps for the missing months,
ii. predicting the natural hazard such as drought or floods,

The major limitations in the way to predict are,

i. random behavior of the coefficients,

ii. limited historical data to train the ANN.

Since the GRACE-FO is in the space and hopefully extend the SH coefficients data for the gravity field, therefore the data availability to train the ANN is becoming easier by each passing month. On the other hand, the ANN to predict the random time series are becoming robust and more efficient since the trading stock market is growing every day and people are trying to predict the random time series of stocks. Therefore there are possibilities that the random behavior prediction can be used in the prediction of SH coefficients for gravity field.

EQUIVALENT WATER HEIGHT (EWH)

A

In this study, the geoidal variation is analyzed in term of EWH. To compute the EWH field from the GRACE data, consider C_{lm} and S_{lm} represent the spherical harmonic coefficients, given by the GRACE data product, and $\Delta\hat{C}_{lm}$ and $\Delta\hat{S}_{lm}$ represent their variation from the mean coefficients $\overline{C}_{lm}$ and $\overline{S}_{lm}$. Now, we assume that if time variable signal is concentrated near the surface of the earth, the spherical harmonic coefficients are converted to surface density by the following expression:

$$\Delta\sigma(\phi,\lambda) = R\rho_w \sum_{l=0}^{\infty}\sum_{m=0}^{l} \overline{P}_{lm}(\cos\phi)\left[\Delta\hat{C}_{lm}\cos(m\lambda) + \Delta\hat{S}_{lm}\sin(m\lambda)\right] \quad . \tag{A.1}$$

$\Delta\sigma$ is the *surface density*, R is the mean Earth radius, ρ_w is the density of water, $\overline{P}_{lm}(\cos\phi)$ are the normalized associated Legendre polynomials, is the latitude, λ is the longitude, with origin at center of the earth, l is the degree, m is the order and $\Delta\hat{C}_{lm}/\Delta\hat{S}_{lm}$ are rescaled spherical harmonic coefficients, given as,

$$\begin{aligned} \Delta\hat{C}_{lm} &= \frac{\rho_{ave}}{3\rho_w}\frac{2l+1}{1+k_l}\Delta C_{lm} \quad , \\ \Delta\hat{S}_{lm} &= \frac{\rho_{ave}}{3\rho_w}\frac{2l+1}{1+k_l}\Delta S_{lm} \quad , \end{aligned} \tag{A.2}$$

with, ρ_{ave} is the average density of the Earth (5517 kg/m^3), ρ_w is the density of water, l is the degree, k_l is the Love number of degree l used to account for the change in shape of the elastic Earth. The EWH is therefore given by,

$$\mathbf{M}(\phi,\lambda) = \frac{\Delta\sigma}{\rho_w} \quad , \tag{A.3}$$

for detail, see (Wahr et al., 1998), and (McCullough, 2013).

THRESHOLD VALUES

B

This appendix states the threshold values for each month from January 2005 to December 2010. Section 3.4 explains the process to get the optimal threshold values.

Month, year	Threshold
April, 2002	8.452e-24
May, 2002	8.197e-24
August, 2002	5.872e-24
September, 2002	5.178e-24
October, 2002	9.510e-24
November, 2002	4.829e-24
December, 2002	6.770e-24
January, 2003	6.900e-24
February, 2003	8.823e-24
March, 2003	6.735e-24
April, 2003	5.163e-24
May, 2003	6.707e-24
July, 2003	3.955e-24
August, 2003	4.824e-24
September, 2003	4.443e-24
October, 2003	3.706e-24
November, 2003	3.657e-24
December, 2003	3.647e-24
January, 2004	2.591e-24
February, 2004	2.791e-24

continued ...

Month, year	Threshold
March, 2004	2.4159e-24
April, 2004	2.4641e-24
May, 2004	4.2988e-24
June, 2004	3.2486e-24
July, 2004	3.5103e-24
August, 2004	3.0367e-24
September, 2004	3.7566e-24
October, 2004	3.7881e-24
November, 2004	2.5728e-24
December, 2004	3.3431e-24
January, 2005	2.1445e-24
February, 2005	3.8052e-24
March, 2005	3.6186e-24
April, 2005	2.6332e-24
May, 2005	3.5757e-24
June, 2005	3.4617e-24
July, 2005	3.5877e-24
August, 2005	4.6937e-24
September, 2005	3.2540e-24
October, 2005	3.7513e-24

continued ...

Month, year	Threshold
November, 2005	2.7854e-24
December, 2005	4.6166e-24
January, 2006	2.2512e-24
February, 2006	3.0433e-24
March, 2006	3.1791e-24
April, 2006	3.0603e-24
May, 2006	2.1876e-24
June, 2006	2.7849e-24
July, 2006	3.5237e-24
August, 2006	3.3424e-24
September, 2006	5.1801e-24
October, 2006	2.7984e-24
November, 2006	3.1571e-24
December, 2006	3.2731e-24
January, 2007	3.1612e-24
February, 2007	2.8217e-24
March, 2007	2.9202e-24
April, 2007	2.6832e-24
May, 2007	3.1639e-24
June, 2007	1.9626e-24
July, 2007	3.0278e-24
August, 2007	3.6950e-24
September, 2007	3.3104e-24
October, 2007	3.9731e-24
November, 2007	3.4464e-24
December, 2007	2.7898e-24
January, 2008	3.0420e-24
February, 2008	4.4418e-24
March, 2008	4.1806e-24
April, 2008	3.0336e-24
May, 2008	3.0827e-24

continued ...

Month, year	Threshold
June, 2008	2.5242e-24
July, 2008	2.6365e-24
August, 2008	2.1201e-24
September, 2008	3.2301e-24
October, 2008	3.0235e-24
November, 2008	3.8128e-24
December, 2008	2.4879e-24
January, 2009	2.5537e-24
February, 2009	3.5165e-24
March, 2009	4.1663e-24
April, 2009	3.3951e-24
May, 2009	2.7728e-24
June, 2009	2.6179e-24
July, 2009	2.6245e-24
August, 2009	3.9760e-24
September, 2009	3.4145e-24
October, 2009	3.5833e-24
November, 2009	3.5463e-24
December, 2009	3.0706e-24
January, 2010	5.2747e-24
February, 2010	5.2747e-24
March, 2010	5.3683e-24
April, 2010	3.7700e-24
May, 2010	4.1038e-24
June, 2010	4.0694e-24
July, 2010	2.8884e-24
August, 2010	3.3486e-24
September, 2010	3.6398e-24
October, 2010	3.4459e-24
November, 2010	3.6017e-24
December, 2010	1.7525e-24

continued ...

Month, year	Threshold
February, 2011	2.698e-24
March, 2011	5.027e-24
April, 2011	4.364e-24
May, 2011	2.912e-24
July, 2011	3.076e-24
August, 2011	3.698e-24
September, 2011	4.390e-24
October, 2011	4.412e-24
November, 2011	2.910e-24
December, 2011	3.443e-24
January, 2012	3.322e-24
February, 2012	3.702e-24
March, 2012	4.529e-24
April, 2012	4.071e-24
June, 2012	3.827e-24
July, 2012	3.594e-24
August, 2012	4.561e-24
September, 2012	3.897e-24
November, 2012	3.607e-24
December, 2012	3.126e-24
January, 2013	2.828e-24
February, 2013	3.511e-24
April, 2013	3.714e-24
May, 2013	2.382e-24
June, 2013	2.636e-24
July, 2013	3.101e-24
October, 2013	2.984e-24
November, 2013	3.280e-24
December, 2013	3.939e-24
January, 2014	3.366e-24
March, 2014	3.769e-24
April, 2014	3.389e-24
May, 2014	2.739e-24
June, 2014	2.433e-24
August, 2014	2.446e-24
September, 2014	2.720e-24
October, 2014	2.855e-24
November, 2014	2.720e-24
January, 2015	6.214e-24
February, 2015	1.536e-24
March, 2015	6.881e-24
April, 2015	7.496e-24
May, 2015	7.212e-24
July, 2015	2.429e-24
August, 2015	2.918e-24
September, 2015	1.962e-24
December, 2015	1.930e-24
January, 2016	2.928e-24
February, 2016	4.281e-24
March, 2016	6.991e-24
May, 2016	4.303e-24
June, 2016	2.970e-24
July, 2016	2.940e-24
August, 2016	5.401e-24
November, 2016	4.505e-24
December, 2016	2.677e-24
January, 2017	6.229e-24
March, 2017	3.585e-24
April, 2017	1.073e-24
May, 2017	3.839e-24
June, 2017	1.711e-24

continued ...

Table B.1: Threshold values from April 2002 and 2017

C

VARIATION OF CONSTANTS

This appendix briefly describes the derivation of a linear system of two algebraic equations for the unknown functions $u(t)$ and $v(t)$ using the *variation of constants* method, as a part of the solution of second order inhomogeneous differential equation. According to the *variation of constants* method, for a second order linear inhomogeneous system,

$$\ddot{y} + a(t)\dot{y} + b(t)y = f(t) \quad , \tag{C.1}$$

whose homogeneous part,

$$\ddot{y} + a(t)\dot{y} + b(t)y = 0 \quad , \tag{C.2}$$

have $y_h(t)$ as a particular solution,

$$y_h(t) = c_1 y_1(t) + c_2 y_2(t) \quad , \tag{C.3}$$

where y_1 and y_2 are the two general solutions of (C.2) and c_1 and c_2 are the constant coefficients which can be replaced by functions $u(t)$ and $v(t)$.

$$y_h(t) = u(t)y_1(t) + v(t)y_2(t) \quad , \tag{C.4}$$

To determine the varying constants we have to impose two conditions. Before moving towards the conditions, let us have the first derivative of the solution (C.4)

$$\dot{y}_h = \dot{u}y_1 + u\dot{y}_1 + \dot{v}y_2 + v\dot{y}_2 \quad , \tag{C.5}$$

Now to avoid the second derivative of the varying constant in the solution, the first condition state that

$$\dot{u}y_1 + \dot{v}y_2 = 0 \tag{C.6}$$

due to the first condition, the (C.5) reduces to

$$\dot{y}_h = u\dot{y}_1 + v\dot{y}_2 \quad , \tag{C.7}$$

Differentiating (C.7), we obtain

$$\ddot{y}_h = \dot{u}\dot{y}_1 + u\ddot{y}_1 + \dot{v}y_2 + v\ddot{y}_2 \quad , \tag{C.8}$$

Now substituting the y_p, $\dot{y}_h$ and $\ddot{y}_h$ from (C.4), (C.7) and (C.8), respectively, into (C.1). Collecting terms in u and terms in v, we obtain,

$$u(\ddot{y}_1 + a\dot{y}_1 + by_1) + v(\ddot{y}_2 + a\dot{y}_2 + by_2) + \dot{u}\dot{y}_1 + \dot{v}\dot{y}_2 = f(t) \quad , \tag{C.9}$$

since y_1 and y_2 are the general solution of (C.2), therefore $\ddot{y}_1 + a\dot{y}_1 + by_1$ and $\ddot{y}_2 + a\dot{y}_2 + by_2$, both are equal to zero, and the (C.9) reduces to

$$\dot{u}\dot{y}_1 + \dot{v}\dot{y}_2 = f(t) \quad , \tag{C.10}$$

which form the second condition, which along with (C.6) form the linear system of two algebraic equations for the unknown functions $\dot{u}$ and $\dot{v}$ and derivatives of general solutions y_1 and y_2. The system can be solved using several methods, elimination followed by Cramer's rule, for instance, where for the varying constants $\dot{u}$ and $\dot{v}$ we need to integrate to get u and v (Kreyszig, 2000).

Index

Bibliography

Antoni, Markus and Wolfgang Keller (2013), "Closed solution of the hill differential equation for short arcs and a local mass anomaly in the central body." *Celestial Mechanics and Dynamical Astronomy*, 115, 107–121.

Ardalan, A, E Grafarend, and J Ihde (2002), "Molodensky potential telluroid based on a minimum-distance map. Case study: the quasi-geoid of East Germany in the World Geodetic Datum 2000." *Journal of Geodesy*, 76, 127–138.

Austen, G. and Erik W. Grafarend (2001), "Fast inverse kepler transformation." *Artificial Satellites*, 36, 99–114.

Bai, Peng, Xiaomang Liu, and Changming Liu (2018), "Improving hydrological simulations by incorporating GRACE data for model calibration." *Journal of Hydrology*, 557, 291 – 304.

Bala, Rajni and Dr. Dharmender Kumar (2017), "Classification using ANN: A Review." *International Journal of Computational Intelligence Research*, 13.

Ballani, L. (1988), "Partielle Ableitungen und Variationsgleichungen zur Modellierung von Satellitenbahnen und Parameterbestimmung (Partial derivatives and variational equations for the modelling of satellite orbits and parameter estimation)." *Vermessungstechnik*, 36, 192–194.

Baur, O., M. Kuhn, and W. E. Featherstone (2009), "GRACE-derived ice-mass variations over Greenland by accounting for leakage effects." *Journal of Geophysical Research: Solid Earth*, 114.

Bentel, Katrin (2010), *Empirical Orthogonal Function Analysis of GRACE Gravity Data: A Method to Indentify Signals in Time Series of Gravity Changes*. VDM Verlag Dr. Müller.

Bertsekas, D.P. (1995), *Nonlinear Programming. Belmont.* MA:Athenas Scientific.

Bettadpur, Srinivas (2012), "Level-2 gravity field product user handbook, grace 327-734." *Center for Space Research, The University of Texas at Austin.*

Beutler, Gerhard (2005), *Methods of Celestial Mechanics: Volume I: Physical, Mathematical, and Numerical Principles*, chapter Variational Equations, 175–207. Springer Berlin Heidelberg, Berlin, Heidelberg.

Bhatia, Nitin and Vandana (2010), "Survey of Nearest Neighbor Techniques." *CoRR*, abs/1007.0085.

Bischof, H., W. Schneider, and A. J. Pinz (1992), "Multispectral classification of Landsat-images using neural networks." *IEEE Transactions on Geoscience and Remote Sensing*, 30, 482–490.

Bishop, C.M. (1995), *Neural networks for pattern recognition*. Oxford University Press, USA.

Caputo, M. (1967), *The gravity field of the Earth*. Academic Press New York, London.

Castellazzi, Pascal, Laurent Longuevergne, Richard Martel, Alfonso Rivera, Charles Brouard, and Estelle Chaussard (2018), "Quantitative mapping of groundwater depletion at the water management scale using a combined GRACE/INSAR approach." *Remote Sensing of Environment*, 205, 408 – 418.

Chen, J. L., Matt Rodell, C. R. Wilson, and J. S. Famiglietti (2005), "Low degree spherical harmonic influences on Gravity Recovery and Climate Experiment (GRACE) water storage estimates." *Geophysical Research Letters*, 32.

Chen, J. L., C. R. Wilson, B. D. Tapley, and J. C. Ries (2004), "Low degree gravitational changes from GRACE: Validation and interpretation." *Geophysical Research Letters*, 31.

Cover, T. and P. Hart (1967), "Nearest neighbor pattern classification." *IEEE Transactions on Information Theory*, 13, 21–27.

Devaraju, Balaji (2015), *Understanding filtering on the sphere: experiences from filtering GRACE data*. Ph.D. thesis, University Stuttgart.

Elsaka, Basem, Jean-Claude Raimondo, Phillip Brieden, Tilo Reubelt, Jürgen Kusche, Frank Flechtner, Siavash Iran Pour, Nico Sneeuw, and Jürgen Müller (2014), "Comparing seven candidate mission configurations for temporal gravity field retrieval through full-scale numerical simulation." *Journal of Geodesy*, 88, 31–43.

Fatolazadeh, Farzam, Mehdi Raoofian Naeeni, Behzad Voosoghi, and Armin Rahimi (2017), "Estimation of fault parameters using GRACE observations and analytical model: Case study: The 2010 chile earthquake." *Journal of Geodynamics*, 108, 26 – 39.

Forootan, E. (2016), *Statistical Signal Decomposition Techniques for Analyzing Time-Variable Satellite Gravimetry Data*. Number 763 in Deutsche Geod. Kommission, Reihe C, Bayer. Akademie der Wissenschaften, Munich.

Förste, Christoph, Roland Schmidt, Richard Stubenvoll, Frank Flechtner, Ulrich Meyer, Rolf König, Hans Neumayer, Richard Biancale, Jean-Michel Lemoine, Sean Bruinsma, Sylvain Loyer, Franz Barthelmes, and Saskia Esselborn (2008), "The geoforschungszentrum potsdam/groupe de recherche de gëodësie spatiale satellite-only and combined gravity field models: EIGEN-GLO4S1 and EIGEN-GLO4C." *Journal of Geodesy*, 82, 331–346.

Freeden, Willi, Volker Michel, and Helga Nutz (2002), "Satellite-to-satellite tracking and satellite gravity gradiometry (Advanced techniques for high-resolution geopotential field determination)." *Journal of Engineering Mathematics*, 43, 19–56.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville (2016), *Deep Learning*. MIT Press.

Grafarend, E (2011), "Space gradiometry: tensor-valued ellipsoidal harmonics, the datum problem and application of the lusternik-schnirelmann category to construct a minimum atlas." *GEM – International Journal on Geomathematics*, 1, 145–166.

Grafarend, E and A Ardalan (1999a), "A first test of $\dot{W}_0$, the time variation of W_0 based on three GPS campaigns of the baltic sea level project." In *Final Results of the Baltic Sea Level 1997 GPS Campaign* (M Poutanen and J Kakkuri, eds.), Report 99:4, 93–113, The Finnish Institute of Geodesy, Kirkonummi.

Grafarend, E and A Ardalan (1999b), "World geodetic datum 2000." *Journal of Geodesy*, 73, 611–623.

Grafarend, E and A Ardalan (2001), "Somigliana-pizzetti gravity: the international gravity formula accurate to the sub-nanogal level." *Journal of Geodesy*, 75, 424–437.

Grafarend, E and J Awange (2013), *Applications of linear and nonlinear models: Fixed effects, random effects, and total least squares*. Springer.

Grafarend, E, J Engels, and P Varga (2000), "The temporal variation of the spherical and cartesian multipoles of the gravity field." *Journal of Geodesy*, 74, 519–530.

Grafarend, E. and A. Heidenreich (1995), "The generalized mollweide projection of the biaxial ellipsoid." *Bulletin géodésique*, 69, 164.

Grafarend, E., F.W. Krumm, and V.S. Schwarze (eds), eds. (2003), *Geodesy - the Challenge of the 3rd Millennium*. Springer Berlin Heidelberg.

Grafarend, Erik W. and K.Heinz (1978), "Rank defect analyse of satellite geodetic network ii - dynamic mode -." *Manuscripta Geodaetica*, 3, 135–156.

Grafarend, Erik W. and E. Livieratos (1978), "Rank defect analyse of satellite geodetic network 1 - geometric and semi-dynamic mode -." *Manuscripta Geodaetica*, 3, 107–134.

Hagan, M.T., H.B. Demuth, and M.H. Beale (2014), *Neural network design*. Martin Hagan.

Hassoun, Mohamad H. (1995), *Fundamentals of artificial neural networks /*. Prentice Hall„ New Delhi,.

Haykin, Simon (2011), *Neural Networks and Learning Machines*. Pearson Education.

Hecht-Nielsen, Robert (1990), *Neurocomputing*. Addison-Wesley.

Heermann, P. D. and N. Khazenie (1992), "Classification of multispectral remote sensing data using a back-propagation neural network." *IEEE Transactions on Geoscience and Remote Sensing*, 30, 81–88.

Heiskanen, Weikko A. and Helmut Moritz (1967), *Physical Geodesy*. W. H. freeman and Co.

Hill, Tim, Leorey Marquez, Marcus O'Connor, and William Remus (1994), "Artificial neural network models for forecasting and decision making." *International Journal of Forecasting*, 10, 5 – 15.

Hu, Xuanyu (2012), *Comparison of Ellipsoidal and Spherical Harmonics for Gravitational Field Modeling of Non-Spherical Bodies*. Master's thesis, Geodatic Science and Surveyong, The Ohio State University.

Huang, Guang-Bin (2003), "Learning capability and storage capacity of two-hidden-layer feedforward networks." *Trans. Neur. Netw.*, 14, 274–281.

Ioffe, Sergey and Christian Szegedy (2015), "Batch normalization: Accelerating deep network training by reducing internal covariate shift." *CoRR*, abs/1502.03167.

Jäggi, A. (2007), *Pseudo-Stochastic Orbit Modeling of Low Earth Satellites Using the Global Positioning System*, volume 73. .

Jäggi, A., G. Beutler, and L. Mervart (2010), "Grace gravity field determination using the celestial mechanics approach-first results." In *Gravity, Geoid and Earth Observation* (Stelios P. Mertikas, ed.), volume 135 of *International Association of Geodesy Symposia*, 177–184, Springer Berlin Heidelberg.

Jäggi, A., U. Hugentobler, and G. Beutler (2006), "Pseudo-stochastic orbit modeling techniques for low-earth orbiters." *Journal of Geodesy*, 80, 47–60.

Jekeli (1981), "Alternative methods to smooth the Earth's gravity field,." *Rep. 327, Dep. of Geod. Sci. and Surv., Ohio State Univ., Columbus.*

Kaastra, Iebeling and Milton Boyd (1996), "Designing a neural network for forecasting financial and economic time series." *Neurocomputing*, 10, 215 – 236.

Kapur, P. K., J. N. Sahoo and A. K .C. Wong (1985), "A new method for gray-level picture thresholding using the entropy of the histogram." *Computer Vision, Graphics, and Image Processing*, 29, 273 – 285.

Keiko Yamamoto, et al. (2005), "A simulation study of effects of GRACE orbit decay on the gravity field recovery." *Earth Planets Space.*, 57, 291 – 295.

Keller, W. (2015), "Data mining in GRACE monthly solutions." In *EGU General Assembly Conference Abstracts*, volume 17 of *EGU General Assembly Conference Abstracts*, 15195.

Keller, Wolfgang (2013), "Satellite-to-satellite tracking (low–low/high–low sst)." In *Handbook of Geomathematics* (Willi Freeden, M. Zuhair Nashed, and Thomas Sonar, eds.), 1–36, Springer Berlin Heidelberg, Berlin, Heidelberg.

Kim, Jeongrae (2000), *Simulation Study of A Low-Low Satellite-to-Satellite Tracking Mission.* Ph.D. thesis, The University of Texas at Austin.

Koch, K.-R. and J. Kusche (2002), "Regularization of geopotential determination from satellite data by variance components." *Journal of Geodesy*, 76, 259–268.

Kozma, L. (2008), "k nearest neighbours algorithm." http://www.lkozma.net/knn2.pdf.

Kreyszig, Erwin (2000), *Advanced Engineering Mathematics: Maple Computer Guide*, 8th edition. John Wiley & Sons, Inc., New York, NY, USA.

Kriesel, David (2007), *A Brief Introduction to Neural Networks.* Book available at: http://www.dkriesel.com/.

Larose, Daniel T. (2004), *Discovering Knowledge in Data: An Introduction to Data Mining*. Wiley-Interscience, New York, NY, USA.

LeCun, Yann, Léon Bottou, Genevieve B. Orr, and Klaus-Robert Müller (1998), "Efficient BackProp." In *Neural Networks: Tricks of the Trade, This Book is an Outgrowth of a 1996 NIPS Workshop*, 9–50, Springer-Verlag, London, UK, UK.

Levenberg, K. (1944), "A method for the solution of certain non-linear problems in least squares." *Quart. Appl. Math.*, 12, 164–168.

Liu, Xianglin (2008), *Global gravity field recovery from satellite-to-satellite tracking data with the acceleration approach*. Ph.D. thesis, NCG Nederlandse Commissie voor Geodesie Netherlands Geodetic Commission, Delft.

Madigan, David (2017), "What is cluster analysis?" Online.

Marquardt, D.W. (1963), "An algorithm for least-squares estimation of nonlinear parameters." *J. Soc. Indust. Appli. Math.*, 11, 431–441.

Mayer-Gürr, T., K.H. Ilk, A. Eicker, and M. Feuchtinger (2005), "ITG-CHAMP01: a CHAMP gravity field model from short kinematic arcs over a one-year observation period." *Journal of Geodesy*, 78, 462–480.

McCullagh, P. and J.A. Nelder (1989), *Generalized linear models*, 2nd edition. Chapman and Hall, London.

McCullough, Christopher (2013), "Global trends in equivalent water height with GRACE and GLDAS." *GEO 386G*.

Mehta, Piyush M. and Richard Linares (2017), "A methodology for reduced order modeling and calibration of the upper atmosphere." *Space Weather*, 15, 1270–1287.

Meyer, U., A. Jäggi, and G. Beutler (2012), "Monthly gravity field solutions based on GRACE observations generated with the Celestial Mechanics Approach." *Earth and Planetary Science Letters*, 345, 72–80.

Mirkes, E. (2011), "Knn and potential energy (applet)." http://www.math.le.ac.uk/people/ag153/homepage/KNN/KNN3.html.

Montenbruck and Gill (2000), *Satellite orbits*. Springer.

Morse, S. (2000), "Lecture 4: Thresholding." Online: Brigham Young University.

Nielsen, Michael A. (2015), *Neural Networks and Deep Learning*. Determination Press.

Nasa, Press-Kit (2002), "grace launch." *National Aeronautics and Space Administration*.

Otsu, N. (1979), "A Threshold Selection Method from Gray-Level Histograms." *IEEE Transactions on Systems, Man, and Cybernetics*, 9, 62–66.

Özkan, Coskun and Filiz Sunar Erbek (2003), "The comparison of activation functions for multispectral landsat tm image classification." *Photogrametric Engineering and Remote Sensing*, 69, 1225–1234.

Patra, Swarnajyoti, Susmita Ghosh, and Ashish Ghosh (2011), "Histogram thresholding for unsupervised change detection of remote sensing images." *International Journal of Remote Sensing*, 32, 6071–6089.

Peters, T., J. Muller, and N. Sneeuw (2002), "Temporal variations in the Earth's gravity field with emphasis on atmospheric effects." In *Vortrage beim 4. DFG-Rundgespräch im Rahmen des Forschungsvorhabens 'Rotation der Erde'* (H. Schuh, M. Soffel, and H. Hornik, eds.), 133–140, .

Pini, Alex James (2012), *Investigation of the effect of repeat orbits on GRACE gravity recovery*. Master's thesis, The University of Texas at Austin.

Piretzidis, D., G. Sra, and M.G. Sideris (2016), "Identification of correlated grace monthly harmonic coefficients using pattern recognition and neural networks." In *AGU Fall Meeting, San Francisco,12- 16 December, 2016*.

Powell, M.J.D. (1987), "A review of algorithms for nonlinear equations and unconstrained optimization." In *First International Conference on Industrial and Applied Mathematics*, 220–264, ICIAM'87, Philadelphia, Society for Industrial and Applied Mathematics.

Prasad, M Seetharama, V Rama Krishna, and LSS Reddy (2011), "Ivestigation on entropy based threshold method." *Asian Journal of Computer Science and Information Technology*, 1.

Ramachandran, Prajit, Barret Zoph, and Quoc V. Le (2017), "Searching for activation functions." *CoRR*, abs/1710.05941.

Reigber, Ch., Georges Balmino, Peter Schwintzer, Richard Biancale, Albert Bode, Jean-Michel Lemoine, Rolf König, Sylvain Loyer, Hans Neumayer, Jean-Charles Marty, Franz Barthelmes, Felix Perosanz, and Shen Yuan Zhu (2002), "A high-quality global gravity field model from CHAMP GPS tracking data and accelerometry (EIGEN-1S)." *Geophysical Research Letters*.

Reigber, Christoph (1974), "Bestimmungsgleichungen für Resonanzparameter der Ordnung 13 aus der Analyse von Bahnen der Satelliten GEOS B, BEC und D1D." *Deutsche Geodätische Kommission, Report C 198*.

Reigber, Christoph (1989), "Gravity field recovery from satellite tracking data." In *Theory of Satellite Geodesy and Gravity Field Determination* (Fernando Sansò and Reiner Rummel, eds.), 197–234, Springer Berlin Heidelberg, Berlin, Heidelberg.

Reigber, Christoph, Roland Schmidt, Frank Flechtner, Rolf König, Ulrich Meyer, Karl-Hans Neumayer, Peter Schwintzer, and Sheng Yuan Zhu (2005), "An earth gravity field model complete to degree and order 150 from GRACE: EIGEN-GRACE02S." *Journal of Geodynamics*, 39, 1 – 10.

Restrepo, Juan M., Rob Indik, Emily Lane, and Rachel Labes (2001), "Introduction to scientific computing." online.

Reubelt, Tilo (2008), *Harmonic analysis of the Earth's gravitational field from kinematic orbits of a Low Earth Orbiting GPS tracked satellite of type CHAMP, GRACE and GOCE by means of an acceleration approach*. Ph.D. thesis, University Stuttgart.

Riley, James D., Morris M. Bennett, and Emily McCormickt (1967), "Numerical integration of variational equations." *Mathematics of computaion*, 21, 12–17.

Rumelhart, David E. and James L. McClelland, eds. (1986), *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1: Foundations*. MIT Press, Cambridge, MA, USA.

Rummel, R. (1979), "Determination of short-wavelength components of the gravity field from satellite-to-satellite tracking or satellite gradiometry an attempt to an identification of problem areas." *Manuscripta Geodaetica*, 4, 107–148.

Rummel, R., G. Balmino, J. Johannessen, P. Visser, and P. Woodworth (2002), "Dedicated gravity field missions - principles and aims." *Journal of Geodynamics*, 33, 3–20.

Rummel, Reiner, Weiyong Yi, and Claudia Stummer (2011), "GOCE gravitational gradiometry." *Journal of Geodesy*, 85, 777.

Seeber, Günter (2003), *Satellite Geodesy*. Walter de Gruyter Berlin, New York.

Sezgin, Mehmet and Bülent Sankur (2004), "Survey over image thresholding techniques and quantitative performance evaluation." *J. Electronic Imaging*, 13, 146–168.

Sharma, Avinash (2017), "Understanding activation functions in neural networks." Blog on Web page: understanding activation functions in neural networks: https://medium.com/the-theory-of-everything/understanding-activation-functions-in-neural-networks-9491262884e0.

Stathakis, D. (2009), "How many hidden layers and nodes?" *International Journal of Remote Sensing*, 30, 2133–2147.

Stegemann, N. R., J. A.and Buenfeld (1999), "A glossary of basic neural network terminology for regression problems." *Neural Computing & Applications*, 8, 290–296.

Sutto, Oliver (2012), "Introduction to k nearest neighbour classification and condensed nearest neighbour data reduction." http://www.math.le.ac.uk/people/ag153/homepage/KNN/OliverKNN_Talk.pdf.

Swenson, Sean and John Wahr (2006), "Post-processing removal of correlated errors in GRACE data." *Geophysical Research Letters*, 33.

Tan, Pang-Ning, Michael Steinbach, and Vipin Kumar (2005), *Introduction to Data Mining*. Addison Wesley.

Tapley, B., J. Ries, S. Bettadpur, D. Chambers, M. Cheng, F. Condi, B. Gunter, Z. Kang, P. Nagel, R. Pastor, T. Pekker, S. Poole, and F. Wang (2005), "GGM02 – An improved Earth gravity field model from GRACE." *Journal of Geodesy*, 79, 467–478.

Tapley, B. D., S. Bettadpur, M. Watkins, and C. Reigber (2004), "The gravity recovery and climate experiment: Mission overview and early results." *Geophysical Research Letters*.

Tkachenko, N. S. and I. V. Lygin (2017), "Application of the grace satellite mission for solving geological and geographic problems." *Moscow University Geology Bulletin*, 72, 159–163.

Trippi, Robert R. and Efraim Turban, eds. (1992), *Neural Networks in Finance and Investing: Using Artificial Intelligence to Improve Real World Performance*. McGraw-Hill, Inc., New York, NY, USA.

Wahr, John, Mery Molenaar, and Frank Bryan (1998), "Time variability of the Earth's gravity field: Hydrological and oceanic effects and their possible detection using grace." *JOURNAL OF GEOPHYSICAL RESEARCH*, 103, 30,205–30,229.

Wahr, John, Sean Swenson, Victor Zlotnicki, and Isabella Velicogna (2004), "Time-variable gravity from GRACE: First results." *Geophysical Research Letters*, 31.

Walia, Anish Singh (2017), "Activation functions and it's types-Which is better?" blog in web page: Towards data science: https://towardsdatascience.com/activation-functions-and-its-types-which-is-better-a9a5310cc8f.

Wang, Linsong, Chao Chen, Maik Thomas, Mikhail K Kaban, Andreas Güntner, and Jinsong Du (2018), "Increased water storage of lake qinghai during 2004-2012 from grace data, hydrological models, radar altimetry and in situ measurements." *Geophysical Journal International*, 212, 679–693.

Weigelt, Matthias Luigi (2007), *Global and Local Gravity Field Recovery from Satellite-to-Satellite Tracking*. Ph.D. thesis, University of Calgary, Calgary, Canada.

Wermuth, Martin K. (2008), *Gravity Field Analysis from the Satellite Missions CHAMP and GOCE*. Ph.D. thesis, Technischen Universität München.

White, Halbert (1992), *Artificial Neural Networks: Approximation and Learning Theory*. Blackwell Publishers, Inc., Cambridge, MA, USA.

Wouters, B. and E. J. O. Schrama (2007), "Improved accuracy of GRACE gravity solutions through empirical orthogonal function filtering on spherical harmonics." *Geophysical Research Letters, 34(L23711), . doi:10.1029/2007/GL032098.*

X. Wu, et al. (2008), "Top 10 algorithms in data mining." *Knowledge Inf. Syst.*, 14, 37.

Xu, Changyi, Xiaoning Su, Tai Liu, and Wenke Sun (2017), "Geodetic observations of the co- and post-seismic deformation of the 2013 okhotsk sea deep-focus earthquake." *Geophysical Journal International*, 209, 1924–1933.

Zaki, Mohammed J. and Wagner Meira Jr. (2014), *Data Mining and Analysis: Fundamental Concepts and Algorithms*. Cambridge University Press.

Zhang, Guoqiang Peter (2000), "Neural networks for classification: a survey." In *and Cybernetics - Part C: Applications and Reviews.*