



NTNU – Trondheim
Norwegian University of
Science and Technology

Advanced Positioning for Offshore Norway

Thomas Alexander Sahl

Petroleum Geoscience and Engineering

Submission date: June 2014

Supervisor: Sigbjørn Sangesland, IPT

Co-supervisor: Bjørn Brechan, IPT

Norwegian University of Science and Technology

Department of Petroleum Engineering and Applied Geophysics

Summary

When most people hear the word coordinates, they think of latitude and longitude, variables that describe a location on a spherical Earth. Unfortunately, the reality of the situation is far more complex. The Earth is most accurately represented by an ellipsoid, the coordinates are three-dimensional, and can be found in various forms. The coordinates are also ambiguous. Without a proper reference system, a geodetic datum, they have little meaning. This field is what is known as "Geodesy", a science of exactly describing a position on the surface of the Earth.

This Thesis aims to build the foundation required for the position part of a drilling software. This is accomplished by explaining, in detail, the field of geodesy and map projections, as well as their associated formulae. Special considerations is taken for the area offshore Norway. Once the guidelines for transformation and conversion have been established, the formulae are implemented in MATLAB. All implemented functions are then verified, for every conceivable method of operation. After which, both the limitation and accuracy of the various functions are discussed. More specifically, the iterative steps required for the computation of geographic coordinates, the difference between the North Sea Formulae and the Bursa-Wolf transformation, and the accuracy of Thomas-UTM series for UTM projections.

The conclusion is that the recommended guidelines have been established and implemented. The implementation has been verified, and the proper handling of information is suggested. Another conclusion is that all functions are more than accurate enough for drilling purposes, and that there is a clear difference between using the Bursa-Wolf transformation and the North Sea Formulae. The difference is discussed in detail, and a intersecting area near the 62°N is depicted, with 4 meters or less deviation between the two methods.

There are multiple areas on which to continue the work carried out on this Thesis. Within the field of Geodesy, it is suggested that work is done to map the transformations used worldwide, and implement these with the appropriate selection criteria for automation. Attempting to find a joint transformation for the North Sea and the Norwegian Sea would also be recommended. This transformation should replace both the North Sea Formulae and the Bursa-Wolf transformation.

Within the field of Map Projection, it is suggested that the conversion of coordinates is expanded to include the MGRS system, the UK grid system, and implementation of the Krüger-n series of 8th order for UTM coordinates.

Lastly, it is suggested that a graphical interface is developed for both current and future models. This interface should serve as a prototype for a final drilling software.

Sammendrag

De fleste vil tenke på lengdegrader og breddegrader når de får høre ordet koordinater. Variabler som beskriver en posisjon på en sfærisk klode. Dessverre er realiteten at situasjonen er mye mer komplisert. Jordkloden beskrives best med en ellipsoide, koordinater er tredimensjonale og de eksisterer i flere former. Koordinatene er også tvetydige. Uten et referansegrunnlag som et geodetisk datum, betyr koordinatene svært lite. Området som dreier seg om dette er kjent som "Geodesi", en vitenskap som dreier seg om å beskrive eksakte posisjoner på jordens overflate.

Denne oppgaven har som formål å bygge grunnlaget for å utvikle posisjonsbiten til et boreprogram. Dette blir gjort ved å forklare teorien samt formelverket innen både geodesi og kartprojeksjoner i detalj. Særs fokus blir lagt på området utenfor kysten til Norge. Formelverket blir implementert i MATLAB etter at retningslinjer for transformasjon og konvertering har blitt fastsatt. Deretter blir de implementerte funksjonene verifisert for alle tenkelige operasjoner. Til slutt drøftes både begrensingene og nøyaktigheten til de forskjellige funksjonene. Spesifikt; den iterative beregningen innen konvertering av geografiske koordinater, differansene mellom Nordsjø-formelen og Bursa-Wolf transformasjonen, samt presisjonen til Thomas-UTM serien for UTM kartprojeksjoner.

Opgaven konkluderer med at de anbefalte retningslinjene har blitt fastsatt og implementert. All matematisk implementering har blitt verifisert, og riktig behandling av informasjon har blitt foreslått. Det blir også konkludert at alle funksjonene er mer enn nøyaktig nok for boreoperasjoner, og at det eksisterer en klar forskjell mellom resultatet av Bursa-Wolf transformasjonen og Nordsjø-formelen. Forskjellen blir diskutert i detalj, og et område nærliggende 62°N breddegrader blir skildret. Dette området har mindre enn 4 meter i avvik mellom de forskjellige metodene.

Det er flere områder hvor arbeidet kan videreføres. Innen geodesi anbefales det at arbeid blir utført slik at alle transformasjonsmetoder blir kartlagt. Disse metodene bør implementeres sammen med valgkriterier slik at transformasjoner kan automatiseres. Det foreslås også at man prøver å finne en felles transformasjon som erstatter både Nordsjø-formelen og Bursa-Wolf transformasjonen.

Innen kartprojeksjoner foreslås det at konverteringen blir videreutviklet slik at det også inkluderer MGRS systemet, det britiske systemet, samt at Krüger-n serien blir implementert i orden 8 for UTM koordinater.

Til slutt anbefales det også at et grafisk brukersnitt blir utviklet for både nåværende samt fremtidige modeller. Dette brukersnittet bør fungere som en prototype for en endelig programvare.

Acknowledgements

This Master's Thesis would not be what you have in your hand, were it not for the assistance of numerous people. People who took their time to listen to my ideas and thoughts, and in turn give feedback, suggestions, and assistance. I am truly grateful for their help that allowed me to accomplish what I set out to do when I started this Thesis. With that being said, I would like to provide an extra thanks to the following:

Bjørn Brechan, STATOIL, for his supervision, Thesis proposal, invaluable help in locating resources, contacts, and providing valuable input. Also, for his keen interest in the subject which helped keep a high motivation throughout the semester.

Professor Sigbjørn Sangesland, for providing me the opportunity of working on this Thesis, and for his supervision and input when needed.

Erik Nyernes, STATOIL, for valuable input in the field of Geodesy.

Assistant Professor, Terje Skogseth, for quality controlling the Geodesy chapter and valuable input.


Bernt Nilsen, Schlumberger, for input and data.

Professor Arild Rødland, for his interest in the subject and general input.

University Lecturer, Erik Skogen, for his input in the field of Geodesy and help in locating references.

And last, I would like to thank my friends, coworkers, and family, for their continued support throughout my studies. None of this would have been possible were it not for you. It truly has been a great experience!

The author of this work hereby declares that the work is made independently and in accordance to the rules set down by "Examination regulations" at the Norwegian University of Science and Technology (NTNU), Trondheim.


Thomas A. Sinclair Sahl


Date

Contents

Summary	i
Sammendrag	iii
Acknowledgements	v
List of Figures	x
List of Tables	xii
Abbreviations	xiii
Symbols	xvii
1 Introduction	1
1.1 Motivation	1
1.2 Goal	1
1.3 Approach	2
1.4 Structure of Thesis	2
2 Geodesy	3
2.1 The Shape of the Earth	3
2.2 Ellipsoid of Revolution	4
2.3 Coordinates	5
2.3.1 Geographic Coordinates (ϕ, λ, H)	6
2.3.2 Cartesian Coordinates (X, Y, Z)	6
2.3.3 Orthometric Height	8
2.3.4 Grid Coordinates	8
2.4 Datums	11
2.4.1 Geodetic Datums	12
2.4.2 Common Geodetic Datums	12
2.4.3 Vertical Datums	13
2.4.4 Fixed Datums	13
2.5 Datum Transformation	13
2.5.1 Geocentric Transformations	14
2.5.2 Geographic Transformations	17
2.5.3 Transformations for Offshore Norway	19
3 Map Projections	23
3.1 Classification of Map Projections	23
3.1.1 Property Groups	23
3.1.2 Graticule Groups	24
3.2 Parameters related to Map Projections	25

3.3	The Mercator Projections	26
3.3.1	Mercator	26
3.3.2	Transverse Mercator	27
3.4	Universal Transverse Mercator	28
3.4.1	Converting from geographical coordinates to Northing and Easting (Thomas-UTM)	29
3.4.2	Converting from Northing and Easting to to geographical coordinates (Thomas-UTM)	30
4	Position Model	33
4.1	MATLAB as a Platform	33
4.2	Converting between Decimal Degrees and Degrees, Minutes, Seconds	33
4.3	Converting between Cartesian and Geographical Coordinates	34
4.4	Ellipsoid Parameters	35
4.5	Polynomials in the North Sea Formulae	35
4.6	Bursa-Wolf in the North Sea Formulae	36
4.7	Computing the Distance between two Coordinates	37
4.8	Computing UTM Coordinates	37
4.9	Modelled Example Case	38
4.9.1	Background	38
4.9.2	Wellhead Coordinates	38
4.9.3	Target Coordinates	39
4.9.4	Solution	39
5	Discussion	41
5.1	Iteration in the function for converting Cartesian to Geographic coordinates	41
5.2	Bursa-Wolf vs. The North Sea Formulae	42
5.3	Accuracy of Thomas-UTM	45
5.4	Offset for Verification	45
5.5	Quality Control	45
6	Conclusion	47
7	Future Work	49
	References	51
	Appendices	55
	Appendix A	55
	Appendix B	61
	Functions for Degrees, Minutes, Seconds and Decimal Degrees	61
	Cartesian and Geographic Coordinates	64
	Function for collecting Ellipsoid Parameters	67
	Reversibel Polynomial in the North Sea Formulae	71
	Bursa-Wolf Transformation	75

Scripts for testing complete transformations	78
Script for computing the deviation between the North Sea Formula and the Bursa-Wolf Transformation	81
Function for calculating the distance between two Geographical co- ordinates	85
Function for Converting from Geographical to UTM Coordinates using USGS	88
Function for converting from UTM to geographical coordinates us- ing USGS	90
Appendices	93
Appendix C	93

List of Figures

2.1	An ellipsoid of revolutuion with semi-major axis (a) and semi-minor axis (b) (Bowditch 1995)	4
2.2	The surface relation of geoidal surface, ellipsoid surface and topographic surface (Bowditch 1995)	5
2.3	The relation between latitude and longitude, and meridians and parallels (Strahler 1963)	6
2.4	The Cartesian axes and planes (OS 2013)	7
2.5	Three points with equal latitude and longitude in three different coordinate systems (OS 2013)	11
2.6	An overview of the various types of datum transformations, including coordinate transformations (Knippers 2009)	14
2.7	The area covered by the North Sea Formulae (Geodesidivisjon 1990)	22
3.1	Various projections of the Earth (Snyder 1987)	25
3.2	The Mercator projection (Snyder 1987)	26
3.3	The Transverse Mercator projection (Snyder 1987)	27
3.4	A map of the world showing the MGRS zones with UTM longitude bands (Snyder 1987)	29
5.1	The effect of decreasing the level of accuracy required for the iterative function "cart2geo".	42
5.2	The computational difference between Bursa-Wolf and The North Sea Formulae for a latitude interval of 48-55, offshore Norway. . . .	43
5.3	The computational difference between Bursa-Wolf and The North Sea Formulae for a latitude interval of 52-68, offshore Norway. . . .	43
5.4	The computational difference between Bursa-Wolf and The North Sea Formulae for a latitude interval of 55-62, offshore Norway. . . .	44
5.5	The computational difference between Bursa-Wolf and The North Sea Formulae for a latitude interval of 62-69, offshore Norway. . . .	44
5.6	A is WGS84 coordinates for Nidarosdomen, B is the same set of coordinates in ED50 (created using Google Maps).	46

List of Tables

2.1	Various ellipsoid of revolutions around the world and throughout time (Burkard 1959; NIMA 2000)	5
2.2	Parameters for transforming from ED87 to WGS84*SEA	20
2.3	Parameters for transforming from WGS84 to ED50 north of 62°N .	21
2.4	Parameters for a simplified transforming from WGS84 to ED50 south of 62°N	21
4.1	Deviation from test values found using the script "test_NorthSea1".	35
4.2	Deviation from test values found using the script "test_NorthSea2".	36
4.3	Deviation from test values found using the script "test_Northof62".	36
A.1	A detailed list of reference ellipsoid parameters (NGIA 2014)	55
A.2	The relation between ID and Reference Ellipsoid	56
A.3	Polynomial Constants for the transformation between ED50 and ED87 (Geodesidivisjon 1990).	57
A.4	Classification of conical map projections (Lee 1944)	58
A.5	Classification of non-conical map projections (Lee 1944)	58
A.6	Test values for the transformation between ED87 and ED50 using the 14-polynomial method (Geodesidivisjon 1990).	59
A.7	Test values for the transformation between ED87 and WGS84*SEA using the modified Bursa-Wolf (Geodesidivisjon 1990).	59
A.8	Test values for the transformation between ED50 and WGS84 (OGP 2001)	60

Abbreviations

NCS	Norwegian Continental Shelf
UKSC	U.K. Continental Shelf
WGS	World Geodetic System
GRS	Geodetic Reference System
OSGB	Ordnance Survey of Great Britain
ED	European Datum
NAD	North American Datum
IERS	International Earth Rotation Service
ETRF	European Terrestrial Reference System
EGM	Earth Gravitational Model
NN	normal null
GPS	Global Positioning System
GNSS	Global Navigation Satellite Systems
UTM	Universal Transverse Mercator
NIMA	National Imagery and Mapping Agency
DoD	U.S. Department of Defense
DMA	U.S. Defense Mapping Agency
EPSG	European Petroleum Survey Group
NGIA	National Geospatial-Intelligence Agency
OGP	International Association of Oil and Gas Producers
OS	Ordnance Survey
USGS	U.S. Geological Survey
UKOOA	U.K. Offshore Operators Association

Symbols

a	semimajor axis of an ellipse
b	semiminor axis of an ellipse
f	flattening of an ellipse
e	first eccentricity
e'	second eccentricity
ϕ	latitude
λ	longitude
H	ellipsoid height
x	Cartesian x-axis
y	Cartesian y-axis
z	Cartesian z-axis
ν	prime vertical radius of curvature
h	orthometric height
V	geoid undulation
N	Northing
E	Easting
$(X,Y,Z)_T$	target coordinates
$(X,Y,Z)_S$	source coordinates
$t(X,Y,Z)$	translation coordinates
$r(X,Y,Z)$	rotation coordinates
M	scale correction
δS	scale correction in ppm
FE	false easting
FN	false northing
k_0	scale factor at origin
Z	zone number
ϕ_0	latitude of natural origin
λ_0	longitude of natural origin

Chapter 1

Introduction

1.1 Motivation

Most of the computation involved in the planning of a new well is done by software. Software such as Landmark, Drilling Office X, and so forth. These softwares handle everything from the well placement, to analysis of hydraulics, torque, etc. for every drilled section of the well. The problem, however, is that they are closed systems with no insight to the procedure of the computation. And they are expensive, very expensive.

By creating an open-source drilling software, these problems can be eliminated. An open-source software could provide all users with an insight as to how computations are carried out, and what lies behind them. No longer restricting the ability to quality control the output to those with vast amounts of experience. An open-source software would provide others with the ability to improve the software, making it more reliable, better, and complete. It could provide researchers and academics a possibility of implementing theoretical ideas into a working model without having to create the framework themselves.

The ultimate motivation lies in the academic possibilities. An open software could improve the academic learning curve, by matching the theory learned in classes to the functionality of a software. It could also provide a platform, upon which cross-discipline and cross-university projects could be carried out. Where the goal could be to improve the software, evolve it into something more than just a drilling software. Perhaps, one day, a petroleum platform for all disciplines.

1.2 Goal

Although the ultimate goal is to build a complete drilling software, the goal of this Thesis is to build a part of it. More exact, the foundation for the positioning aspect of the software. The foundation should be built in such manner that it is easily understandable, altered, and quality controlled. It should also be built in such a way that alteration is possible. The foundation should consist of a theoretical part which explains both the theory and formulae involved in the computations, and the computational implementation for handling positions. In addition, emphasis

should be placed on the procedures for position handling offshore Norway.

1.3 Approach

The goal was achieved with a large emphasis on quality control. Every theoretical aspect has been verified using a plethora of references, including both written articles and professionals. The implementation of formulae have been designed in such a way that the structure is easily understandable, so that others can alter or continue the work without having to go to the same lengths of understanding. The implemented formulae have been verified for every conceivable purpose, using external and constructed test values. All transformations have been matched to external test values, and conversions have been run forwards and backwards with zero offset for a multitude of values.

1.4 Structure of Thesis

The Thesis first presents the theoretical foundation for understanding positions. The chapter “Geodesy” explains how positions on the surface of the Earth are related to coordinates through the use of reference systems known as geodetical datums. The chapter also details the procedures for transformations offshore Norway.

The next chapter goes into map projections. Map projections are the two-dimensional representation of three-dimensional coordinates onto a plane surface. The scope of this chapter revolves around UTM and the coordinates for this system.

The fourth chapter, “Position Model”, contains a written explanation of every function and script presented in Appendix B. The purpose and verification of every line of code is given here, along with the reason for choosing MATLAB as a programming platform. The chapter is finalized through a worked example, which details the exact procedure for transformations offshore Norway.

After all the code has been explained, the limits are discussed in the next chapter. This chapter looks at the differences between various methods of transformation, the importance of iterative steps, the result of verification, and the accuracy of the chosen map projection formulae.

Lastly, the most vital information is presented in the chapter “Conclusion”, and the suggested path ahead is described in the chapter “Future Work”. The information is presented as bullet-points.

Appendix A contains data information in tables. Data such as test values, reference parameters, and so forth, where as Appendix B contains every line of code for the implemented formulae and verification. The last appendice, Appendix C, consists of the script used for computing values for the worked example presented earlier. This script also contains comments which describe every step of the procedure.

Chapter 2

Geodesy

Geodesy is the science of finding exact positions along the surface of the Earth. It is further defined as that branch of mathematics which deals with the shape and area of the Earth, or large portions of it. Geodesy has evolved a lot throughout the last decades due to recent advances in military and computer technology. The increased accuracy requirements for military missiles combined with vast amounts of satellite data and improved computational handling has shaped geodesy into what it is today.

2.1 The Shape of the Earth

The shape of the Earth needs to be expressed mathematically before a coordinate system can be put into place. A simple approximation would be to define the Earth as spherical, which would yield a mathematical model that requires little effort for navigation. Such an approximation may prove sufficient for many astronomical and navigational computations. The reality is that the Earth resembles an oblate spheroid, with a shorter polar radius than the equatorial radius. The actual surface of the Earth is known as a topographic surface upon which geodetic measurements are made, and reduced into the geoid. The geoid is defined as "a surface along which the gravity potential is everywhere equal and to which the direction of gravity is always perpendicular" (Burkard 1959). This can be thought of as that surface which the oceans would conform into, if left free to adjust for the combined effect of the Earth's mass attraction and the centrifugal force due to the Earth's rotation. The uneven distribution of the Earth's mass results in an irregular geoidal surface which gives serious limitations as a mathematical model. In fact, the geoidal surface cannot be completely mathematically expressed. Another noteworthy fact is that crustal movement will induce measurement errors over time.

The solution to this is to approximate the surface. As previously mentioned, a sphere approximation may prove sufficient for simple navigational purposes, and for a small city survey, a flat Earth model can prove accurate enough. However, this approximation is not accurate enough for large scale calculations, nor exact enough to find the location of a well offshore. The oblate spheroid is best approximated using what is known as an ellipsoid of revolution.

2.2 Ellipsoid of Revolution

An ellipsoid of revolution is obtained by revolving an ellipse around its semi-minor axis (see Figure 2.1).

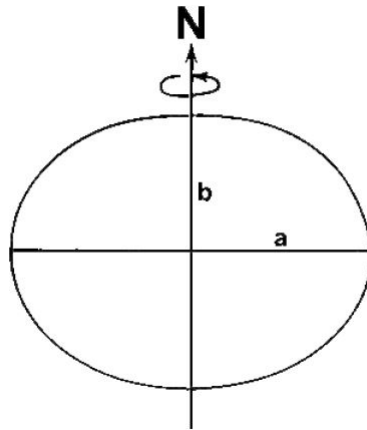


Figure 2.1: An ellipsoid of revolution with semi-major axis (a) and semi-minor axis (b) (Bowditch 1995)

An ellipsoid of revolution can be defined by its semi-major axis and its shape. The shape is often defined by flattening or inverse flattening, but can also be represented by the first and second eccentricity. The flattening is given by the following formulae:

$$f = \frac{a - b}{a} \tag{2.1}$$

The first eccentricity by:

$$e = \sqrt{\frac{a^2 - b^2}{a^2}} = 2f - f^2 \tag{2.2}$$

And the second eccentricity by:

$$e' = \sqrt{\frac{e^2}{1 - e^2}} \tag{2.3}$$

Various ellipsoids have been used throughout the years (Table 2.1) with different flattening and semi-major axis. This is because some ellipsoids are better for localized mapping than other. For instance, England still uses the Airy ellipsoid of 1830, whereas Ireland uses a slightly modified version of the same ellipsoid. A detailed list of ellipsoid parameters can be found in Appendix A (Table A.1).

Name	Year	a [m]	1/f	Where Used
Airy	1830	6,377,276	299.32	Great Britain
Everest	1830	6,377,276	300.80	India
Bessel	1841	6,377,397	299.15	Japan
Clarke	1866	6,378,206	294.98	North America
Clarke	1880	6,378,249	293.46	France, Africa
Krassowsky	1940	6,378,245	298.3	Russia
International	1942	6,378,388	297	Europe
WGS 66	1966	6,378,160	298.25	USA/DoD
GRS 67	1967	6,378,160	298.25	Australia, South America
WGS 72	1972	6,378,135	298.26	USA/DoD
GRS 80	1979	6,378,137	298.26	Australia, South America
WGS 84	1984	6,378,137	298.257	USA/DoD

Table 2.1: Various ellipsoid of revolutions around the world and throughout time (Burkard 1959; NIMA 2000)

The ellipsoid is an approximation of the Earth's surface and not a perfect representation. The variations between the ellipsoid and the geoid can be seen in Figure 2.2 where the angle between the plumb line, perpendicular to the geoid, and the line perpendicular to the ellipsoid is known as the deflection of the vertical. The difference in height between the geoid and the ellipsoid is known as the geoid undulation.

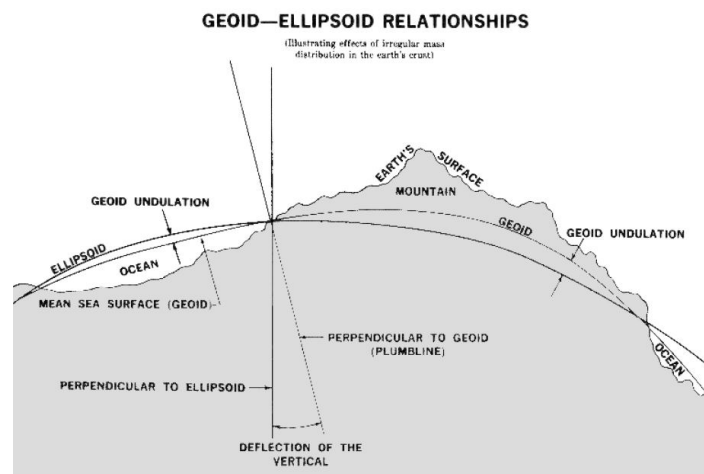


Figure 2.2: The surface relation of geoidal surface, ellipsoid surface and topographic surface (Bowditch 1995)

2.3 Coordinates

Coordinates are used to specify a geodetic position. They are combined with a coordinate system, which is a set of mathematical rules used to tie coordinates and position. This system can be further developed by realizing the coordinate

system through a datum, thus creating a coordinate based reference system, often referred to as a Terrestrial Reference Frame (TRF) (Geodesividsjonen 2009). Coordinates can be divided into three categories with respect to their nature; geographic, Cartesian, and grid based.

2.3.1 Geographic Coordinates (ϕ , λ , H)

Geographic coordinates consists of latitude (ϕ), longitude (λ), and ellipsoid height (H). Latitude and longitude are angles which represent a point on the surface of an ellipsoid. The angles are defined through the use of meridians and parallels (see Figure 2.3). Meridians are lines of constant longitude in the north-south direction, and parallels are lines of constant latitude in the east-west direction. One meridian is assigned the value of zero degrees longitude, and subsequently named the prime meridian. The ellipsoid is divided into two hemiellipsoids west and east of the prime meridian, where the longitude angles range from 0 to 180 degrees in either west or east respectively. The zero degrees reference for latitude is assigned to the equator of the ellipsoid, and the latitude angles range from 0 to 90 degrees in either north or south where 90 degrees is the pole of the ellipsoid.

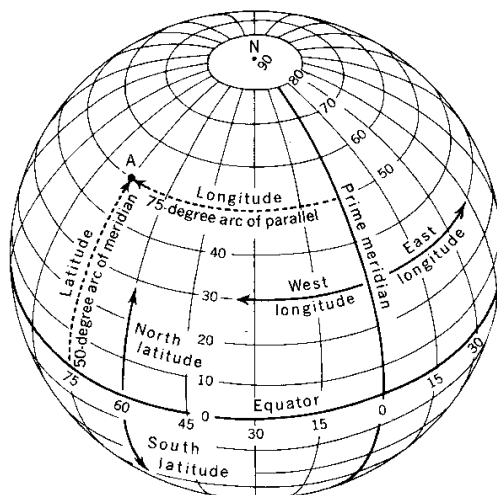


Figure 2.3: The relation between latitude and longitude, and meridians and parallels (Strahler 1963)

Latitude and longitude can be given in either decimal degrees, or in degrees, minutes, and seconds. In case of the latter, one degree is equal to 60 minutes, which in turn is equal to 3600 seconds. The ellipsoid height is the distance between the described point and the ellipsoid surface along a straight line perpendicular to the ellipsoid surface.

2.3.2 Cartesian Coordinates (X , Y , Z)

Cartesian coordinates, also known as geocentric coordinates, uses three perpendicular axes, X , Y , and Z to describe a position in three dimensions (Figure 2.4). This coordinate system conveys the same information as the latitude and longitude

system. The X axis is defined as positive on that side of the geocenter which passes through the prime meridian, the Z axis is defined as positive on that side of the geocenter which passes through the North Pole, and Y axis is defined as positive on that side of the geocenter which passes through 90 degrees west. Just as with latitude and longitude, a set of cartesian coordinates are bound to the reference ellipsoid and would have to be transformed if used for a different ellipsoid.

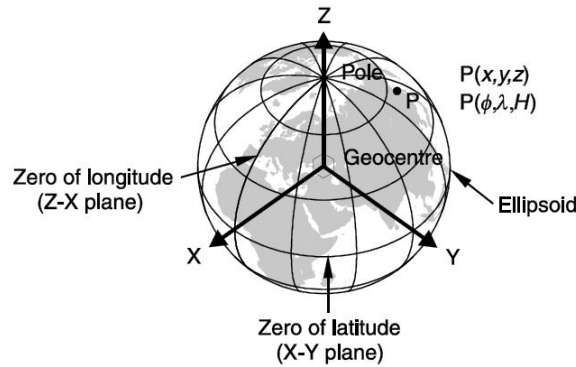


Figure 2.4: The Cartesian axes and planes (OS 2013)

Any point that can be represented by Cartesian coordinates can also be represented by latitude, longitude, and ellipsoid height. The only required information for the conversion between these two sets of data is the ellipsoid data.

Geographic coordinates to Cartesian coordinates

Cartesian coordinates can be calculated based on the following set of equations (OS 2013; Hofmann-Wellenhof et al. 2008):

$$\begin{aligned} x &= (v + H) \cos \phi \cos \lambda \\ y &= (v + H) \cos \phi \sin \lambda \\ z &= ((1 - e^2)v + H) \sin \phi \end{aligned} \quad (2.4)$$

Where e^2 is the eccentricity (Equation 2.2) and ν is the prime vertical radius of curvature at latitude ϕ and is given by:

$$\nu = \frac{a}{\sqrt{1 - e^2 \sin^2 \phi}} \quad (2.5)$$

Cartesian coordinates to Geographic coordinates

Geographic coordinates can be computed based on the following set of equations (OS 2013; Hofmann-Wellenhof et al. 2008):

$$\lambda = \arctan \frac{y}{x} \quad (2.6)$$

Initial latitude:

$$\phi_0 = \arctan \frac{z}{p(1 - e^2)} \quad (2.7)$$

Where:

$$p = \sqrt{x^2 + y^2} \quad (2.8)$$

The latitude is improved iteratively until desired precision:

$$\phi_n = \arctan \frac{z + e^2 \nu \sin \phi_{(n-1)}}{p} \quad (2.9)$$

Ellipsoidal height:

$$H = \frac{p}{\cos \phi} - \nu \quad (2.10)$$

2.3.3 Orthometric Height

Orthometric height is the distance between the Geoid and the ground. This parameter makes it possible to relate the height of one point to another, since the ellipsoid height alone is misleading. The orthometric height as presented assumes that the level passing through the position is parallel to the Geoid, which may not be true in all instances, but the difference is negligible. The relation between ellipsoid height and orthometric height is (OS 2013; Hofmann-Wellenhof et al. 2008):

$$h = V - H \quad (2.11)$$

Where h is the orthometric height, H is the ellipsoid height and V is the geoid undulation. The geoid undulation can be found for any longitude and latitude in what is known as a Geoid model. An important note is that the difference in h between two points is always more accurate than the h for one point alone, since the errors present in both points are removed (OS 2013).

2.3.4 Grid Coordinates

Grid coordinates are also known as plane coordinates, map coordinates, or more colloquial, Northings and Eastings. These coordinates are intended to be used with a map, which is a two-dimensional plane surface projection of an area. The map coordinates of a point are computed from ellipsoidal latitude and longitude by what is known as a map projection. The coordinates are based upon a simple two-dimensional Cartesian system which uses two axes known as northing and easting, where the distance from the axes are given in metres and referred to as Northings and Eastings.

Latitude and longitude to Eastings and Northings

In order to compute Eastings and Northings one need to know the latitude and longitude of the position, as well as the Northing and Easting of the true origin (N_0 and E_0), the scale factor on the central meridian (F_0 or k_0), the latitude and longitude of the true origin (ϕ_0 and λ_0), and finally the ellipsoid constants a , b and e^2 . All angles should be expressed in radians, and the converting is only

valid within the same datum. If all these parameters are known, the Eastings and Northings can be computed based on the following set of equations (OS 2013):

$$n = \frac{a - b}{a + b} \quad (2.12)$$

$$\nu = aF_0(1 - e^2 \sin^2 \phi)^{-0.5} \quad (2.13)$$

$$\rho = aF_0(1 - e^2)(1 - e^2 \sin^2 \phi)^{-1.5} \quad (2.14)$$

$$\eta^2 = \frac{\nu}{\rho} - 1 \quad (2.15)$$

$$\begin{aligned} M = bF_0 & \left[(1 + n + \frac{5}{4}n^2 + \frac{5}{4}n^3)(\phi - \phi_0) - \right. \\ & (3n + 3n^2 + \frac{21}{8}n^3) \sin(\phi - \phi_0) \cos(\phi + \phi_0) + \\ & (\frac{15}{8}n^2 + \frac{15}{8}n^3) \sin(2(\phi - \phi_0)) \cos(2(\phi + \phi_0)) \cdot \\ & \left. \frac{35}{24}n^3 \sin(3(\phi - \phi_0)) \cos(3(\phi + \phi_0)) \right] \end{aligned} \quad (2.16)$$

$$I = M + N_0 \quad (2.17)$$

$$II = \frac{\nu}{2} \sin \phi \cos \phi \quad (2.18)$$

$$III = \frac{\nu}{24} \sin \phi \cos^3 \phi (5 - \tan^2 \phi + 9\eta^2) \quad (2.19)$$

$$IIIA = \frac{\nu}{720} \sin \phi \cos^5 \phi (61 - 58 \tan^2 \phi + \tan^4 \phi) \quad (2.20)$$

$$IV = \nu \cos \phi \quad (2.21)$$

$$V = \frac{\nu}{6} \cos^3 \phi (\frac{\nu}{\rho} - \tan^2 \phi) \quad (2.22)$$

$$VI = \frac{\nu}{120} \cos^5 \phi (5 - 18 \tan^2 \phi + \tan^4 \phi + 14\eta^2 - 58(\tan^2 \phi)\eta^2) \quad (2.23)$$

$$N = I + II(\lambda - \lambda_0)^2 + III(\lambda - \lambda_0)^4 + IIIA(\lambda - \lambda_0)^6 \quad (2.24)$$

$$E = E_0 + IV(\lambda - \lambda_0) + V(\lambda - \lambda_0)^3 + VI(\lambda - \lambda_0)^5 \quad (2.25)$$

The formulae presented above are known as the Redfearn-Lee-OSGB series, and is further described in the chapter "Map Projections" below.

Eastings and Northings to latitude and longitude

Converting from Eastings and Northings to latitude and longitude is an iterative process which requires the same preliminary parameters as converting latitude and longitude to Eastings and Northings, namely a , b , e^2 , N_0 , E_0 , F_0 , ϕ_0 and λ_0 . The angles should be expressed in radians and the conversion is only valid within the same datum. With these preliminaries in place the conversion can be computed based on the following set of equations (OS 2013): Equation 2.26 calculates the initial latitude value:

$$\phi' = \frac{N - N_0}{aF_0} + \phi_0 \quad (2.26)$$

The value of M is obtained from Equation 2.16. As long as the absolute value of $(N - N_0 - M) \geq 0.01mm$ the value of ϕ' should be improved using the following equation:

$$\phi'_{new} = \frac{N - N_0 - M}{aF_0} + \phi' \quad (2.27)$$

Once the absolute value is less than 0.01mm the values of ν , ρ , and η^2 should be calculated using Equations 2.13, 2.14, and 2.15 respectively. The following equations should be calculated next:

$$VII = \frac{\tan \phi'}{2\rho\nu} \quad (2.28)$$

$$VIII = \frac{\tan \phi'}{24\rho\nu^3} (5 + 3 \tan^2 \phi' + \eta^2 - 9(\tan^2 \phi')\eta^2) \quad (2.29)$$

$$IX = \frac{\tan \phi'}{720\rho\nu^5} (61 + 90 \tan^2 \phi' + 45 \tan^4 \phi') \quad (2.30)$$

$$X = \frac{\sec \phi'}{\nu} \quad (2.31)$$

$$XI = \frac{\sec \phi'}{6\nu^3} \left(\frac{\nu}{\rho} + 2 \tan^2 \phi' \right) \quad (2.32)$$

$$XII = \frac{\sec \phi'}{120\nu^5} (5 + 28 \tan^2 \phi' + 24 \tan^4 \phi') \quad (2.33)$$

$$XIIIA = \frac{\sec \phi'}{5040\nu^7} (61 + 662 \tan^2 \phi' + 1320 \tan^4 \phi' + 720 \tan^6 \phi') \quad (2.34)$$

$$\phi = \phi' - VII(E - E_0)^2 + VIII(E - E_0)^4 - IX(E - E_0)^6 \quad (2.35)$$

$$\lambda = \lambda_0 + X(E - E_0) - XI(E - E_0)^3 + XII(E - E_0)^5 - XIIIA(E - E_0)^7 \quad (2.36)$$

2.4 Datums

A datum is defined as a set of numerical or geometrical quantities which serve as a reference point from which to measure other quantities (Bowditch 1995). A datum defines the placement of a coordinate system through defining the point of origin, the scale, and the orientation of the axis (Geodesividsjonen 2009). Any change in the datum values will result in a mismatch of coordinates and position. This is illustrated in Figure 2.5 which displays the same longitude and latitude realized with three different geodetic datums. The field of Geodesy recognizes three forms of datums; geodetic, vertical, and fixed.

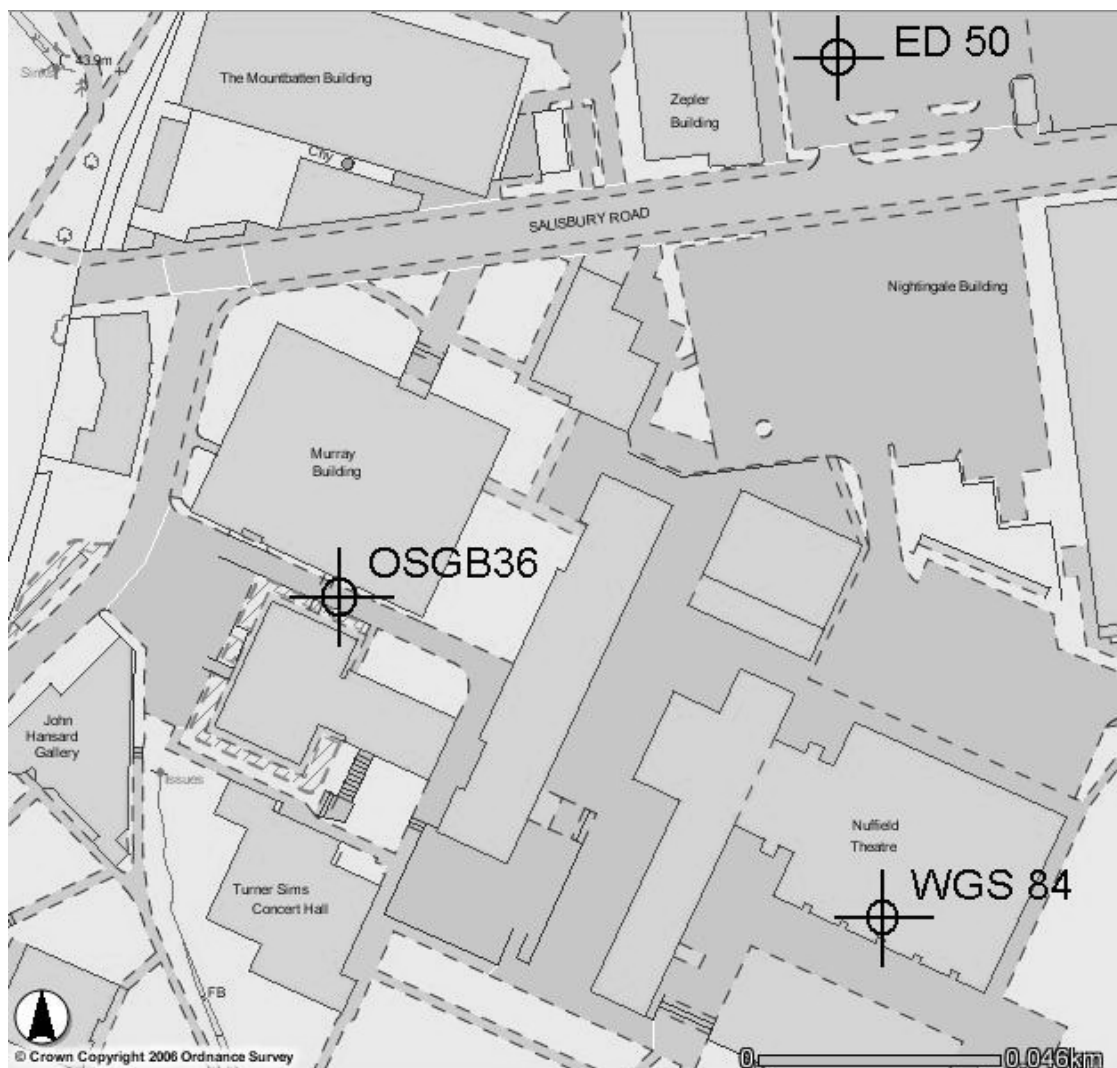


Figure 2.5: Three points with equal latitude and longitude in three different coordinate systems (OS 2013)

2.4.1 Geodetic Datums

A geodetic datum usually consists of both the size and shape of a reference ellipsoid, along with its placement and orientation in relation to the Earth. Any height is given as an ellipsoidal height in geodetic datums. There are two distinct forms of geodetic datums, topocentric and geocentric (Geodesividsjonen 2009).

Topocentric datums are the traditional form of geodetic datums. They are tied to the surface of the Earth and usually covers one or several nations. They are tied to a fundamental point, normally an observatory. The fundamental point is accompanied by geoidal height and deviation from plumb line.

Geocentric datums were made possible due to satellite technology. They assume a three-dimensional coordinate system is placed with the point of origin at the Earth's centre of mass. The axis are located such that one falls through the reference pole dictated by the International Earth Rotation Service (IERS) and lies quite close to the Earth's axis of rotation. Another axis falls through the point where the IERS reference pole and the equatorial plane crosses. This point is quite close to where the Greenwich meridian crosses the equator. The last, and final axis, is perpendicular to the aforementioned axes.

2.4.2 Common Geodetic Datums

There is a plethora of geodetic datums used around the world, but only a few will be covered here. They are the International Terrestrial Reference Frame (ITRF), World Geodetic System of 1984 (WGS84), and the European Datum of 1950 (ED50). ITRF is considered the most accurate geodetic datum to which all other geodetic datums matches themselves with. WGS84 was developed specifically for the use of Global Positioning System (GPS), and is widely used throughout the world. ED50 is the most common geodetic datum in the European oil and gas industry, and is considered the standard at both Norwegian and British side of the North Sea (UKOOA 1999; EPSG 2001).

International Terrestrial Reference Frame

The ITRF is realized for a given year in order to account for the tectonic plate movement. The referred year is denoted either through two digits for any year before 2000, or by four digits for the year 2000 and any year following it (Geodesividsjonen 2009). A version of the ITRF is known as the European Terrestrial Reference System (ETRF). This is used by the European countries. Norway decided to use this as a norm for all positions in the mainland in 1990, thus ETRF89 is the standard (Geodesividsjonen 2009).

World Geodetic System of 1984

The WGS84 is the latest generation of WGS datums. The WGS datums origin was a joint effort by American Army, Navy, and Air Force, led by the US Department of Defense in order to fulfil global military navigational requirements. This led to the first generation of WGS datums, namely the World Geodetic System of 1960

(WGS60). The next instalment was the World Geodetic System of 1966, which included large amount of data from Doppler and optical satellites which were now available (Bowditch 1995). The third generation, the World Geodetic System of 1972, was a product of computer evolution, which had produced better methods for handling and combining data. WGS84 is considered the standard reference datum for the Global Positioning System (Welch and Homsey 1997) and closely corresponds to the North American Datum of 1983 (NAD83). WGS 84 uses the Earth Gravitational Model of 1996 (EGM96) as a geopotential model, replacing the previous WGS84 geoid, and the WGS84 reference ellipsoid (NIMA 2000).

European Datum of 1950

The ED50 was developed after World War II as a result of incompatible latitude and longitude positioning at the borders of Germany, Netherlands, Belgium, and France. The datum uses the International Ellipsoid of 1924 as the reference ellipsoid, has its origin in Potsdam, Germany, and uses the Greenwich meridian as its prime meridian. The modern usage of ED 50 is limited to offshore applications in western Europe, mainly in the Oil & Gas industry (OGP 2001). As mentioned earlier, ED50 is considered the standard for both the NCS and the UKSC.

2.4.3 Vertical Datums

Vertical datums form the basis for gravity based heights, i.e., heights affected by variations in both the strength and the direction of the gravity. The datums consist of three parameters, the reference area, the fundamental point, and the time. The reference area is most commonly the Geoid. The Geoid is affected by forces from both the sun and the moon. The fundamental point is the visible point on the surface which corresponds to the Geoid. This point defines the zero level of height. Both the Geoid and the visible surface are affected by time, through erosion and other factors which need to be accounted for. The official vertical datum of Norway is NN1954, where the fundamental point lies near Tregde, Mandal (Geodesividsjonen 2009).

2.4.4 Fixed Datums

A fixed datum is a very simplified datum which only yields unique positions in a limited area. These types of datums are commonly used in construction sites, survey boats, and so forth, where everything is related to a fixed point (Geodesividsjonen 2009).

2.5 Datum Transformation

Datum transformation is the process where a set of coordinates are transformed from one datum system to another. This is a necessity since the horizontal datums which form the basis of the geographic coordinate system vary from system

to system. A unique set of latitude and longitude coordinates could indicate different positions for different datums as seen in Figure 2.5. Datum transformations can be divided into a subset of two categories, namely geocentric transformations and geographic transformations. The transformations can be realized by a direct approach, where the geographic coordinates of both datums are related, or by an indirect approach, where the geocentric coordinates of both datums are related. An overview of the two methods and the transformation approaches related to each method can be seen in Figure 2.6.

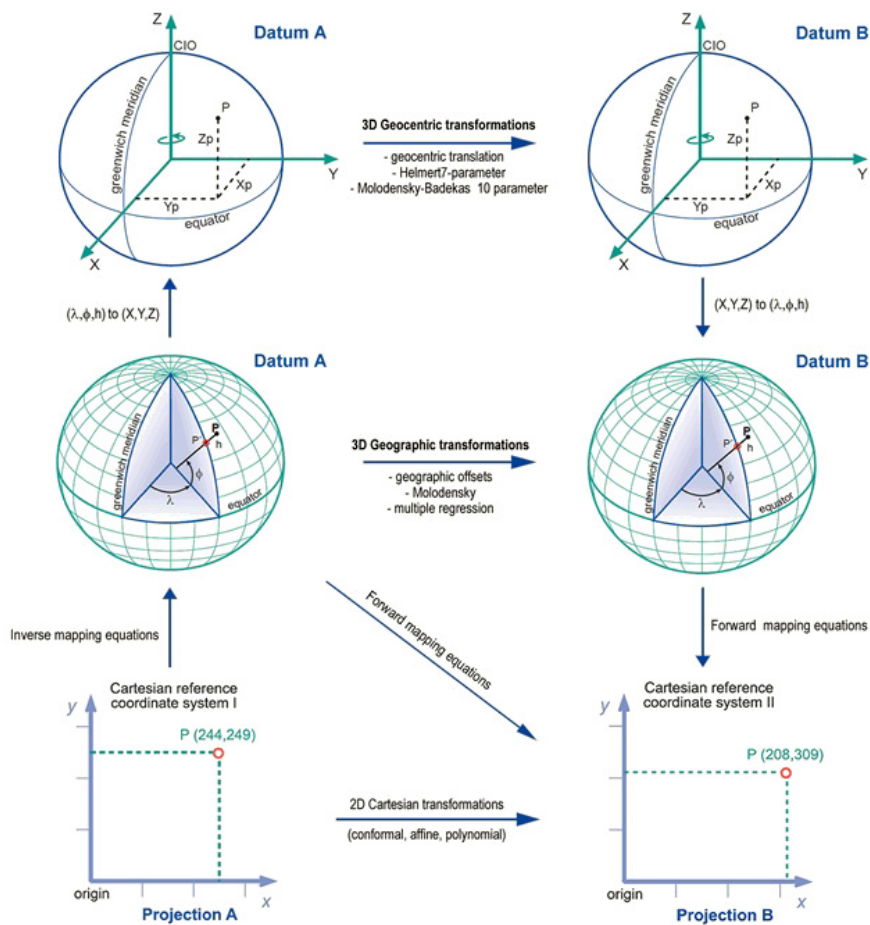


Figure 2.6: An overview of the various types of datum transformations, including coordinate transformations (Knippers 2009)

2.5.1 Geocentric Transformations

Transformations between geocentric coordinate systems are known as three-dimensional similarity transformations. These transformations are frequently used as a link in transforming geographic systems to one another by first converting the geographic coordinates to geocentric coordinates in the first datum, then transforming geocentric coordinates, and lastly converting them back to geographic coordinates

in the second datum. There are three main methods of transforming geocentric coordinates:

1. the geocentric translation
2. the Helmert 7-parameter transformations
3. the Molodensky-Badekas 10-parameter transformation

Geocentric Translation

The geocentric translation is the simplest form of transforming geocentric coordinates. The method assumes that the axes of both ellipsoids are parallel, the prime meridian is Greenwich, and that there is no scale difference between the two datums. If all the assumptions hold true, the geocentric coordinates of the target datum (X_T, Y_T, Z_T) can be found by adding three translations known as shifts (tX, tY, tZ) to the source coordinates (X_S, Y_S, Z_S) (OGP 2013):

$$\begin{aligned} X_T &= X_S + tX \\ Y_T &= Y_S + tY \\ Z_T &= Z_S + tZ \end{aligned} \tag{2.37}$$

Conventional Helmert 7-parameter

The Helmert 7-parameter transformation expands upon the simple geocentric translation by accounting for the rotation and scale differences between the source and target datum. The transformation is considered to be reversible and can be expressed as either a position vector transformation or as a coordinate frame transformation, where the difference lies in the definition of the rotation parameters. The Position Vector Transformation is also known as the Bursa-Wolf transformation, and assumes that the rotations are applied to the vector of the point and is expressed by the following matrix equation (OGP 2013):

$$\begin{pmatrix} X_T \\ Y_T \\ Z_T \end{pmatrix} = M \cdot \begin{pmatrix} 1 & -rZ & rY \\ rZ & 1 & -rX \\ -rY & rX & 1 \end{pmatrix} \cdot \begin{pmatrix} X_S \\ Y_S \\ Z_S \end{pmatrix} + \begin{pmatrix} tX \\ tY \\ tZ \end{pmatrix} \tag{2.38}$$

The Helmert 7-parameter transformation is considered reversible for practical purposes if the transformation parameters are small compared to the magnitude of the geocentric coordinates. The parameters (tX, tY, tZ) are known as the translation vector, which is the same as the coordinates of the origin of the source datum expressed in the coordinates of the target datum. The parameters (rX, rY, rZ) are the rotations which needs to be applied to the vector of the point. The rotations are defined as positive in the clockwise direction when viewed from the origin of the geocentric coordinate system. Equation 2.38 requires the rotation angles to be input in radians. The last parameter, M , is the scale correction which is a function of δS , the scale correction in parts per million:

$$M = (1 + \delta S \cdot 10^{-6}) \quad (2.39)$$

Equation 2.38 is commonly used in the European E&P industry, however the position vector and transformation sign are not universally accepted. The USA E&P industry uses a variation of the Position Vector Transformation known as the Coordinate Frame Rotation and is given as (OGP 2013):

$$\begin{pmatrix} X_T \\ Y_T \\ Z_T \end{pmatrix} = M \cdot \begin{pmatrix} 1 & rZ & -rY \\ -rZ & 1 & rX \\ rY & -rX & 1 \end{pmatrix} \cdot \begin{pmatrix} X_S \\ Y_S \\ Z_S \end{pmatrix} + \begin{pmatrix} tX \\ tY \\ tZ \end{pmatrix} \quad (2.40)$$

The only difference from the European method is in the rotation sign convention.

Time-dependent Helmert 7-parameter

The Helmert 7-parameter transformation can be modified to account for the tectonic plate motion. The result is a 15-parameter transformation where the additional parameters are the rates of the original 7 parameters, and the reference time of the rates. The Time-dependent Position Vector Transformation is given by the following equations (OGP 2013):

$$\begin{pmatrix} X_T \\ Y_T \\ Z_T \end{pmatrix} = M' \cdot \begin{pmatrix} 1 & -rZ' & rY' \\ rZ' & 1 & -rX' \\ -rY' & rX' & 1 \end{pmatrix} \cdot \begin{pmatrix} X_S \\ Y_S \\ Z_S \end{pmatrix} + \begin{pmatrix} tX' \\ tY' \\ tZ' \end{pmatrix} \quad (2.41)$$

Where ' indicates time-adjusted parameters:

$$\begin{aligned} tX' &= tX + \delta tX(t - t_0) \\ tY' &= tY + \delta tY(t - t_0) \\ tZ' &= tZ + \delta tZ(t - t_0) \\ rX' &= rX + \delta rX(t - t_0) \\ rY' &= rY + \delta rY(t - t_0) \\ rZ' &= rZ + \delta rZ(t - t_0) \\ M' &= 1 + dS' \cdot 10^{-6} \\ dS' &= dS + \delta dS(t - t_0) \end{aligned} \quad (2.42)$$

Where t is the epoch of the dynamic coordinates and t_0 is the transformation reference epoch. In order to reverse the transformation, all the sign conventions need to be reversed with the only exception being the reference epoch, t_0 . The Time-dependent Helmert 7-parameter transformation is also modified for the USA E&P industry and is known as the Time-dependent Coordinate Frame Rotation (OGP 2013):

$$\begin{pmatrix} X_T \\ Y_T \\ Z_T \end{pmatrix} = M' \cdot \begin{pmatrix} 1 & rZ' & -rY' \\ -rZ' & 1 & rX' \\ rY' & -rX' & 1 \end{pmatrix} \cdot \begin{pmatrix} X_S \\ Y_S \\ Z_S \end{pmatrix} + \begin{pmatrix} tX' \\ tY' \\ tZ' \end{pmatrix} \quad (2.43)$$

Molodensky-Badekas 10-parameter

The Molodensky-Badekas 10-parameter transformation is almost identical to the Helmert 7-parameter transformation. The difference between the two is that the Molodensky-Badekas transformation derives the rotation at a location within the points used in the determination of the parameters, rather than around the geocentric origin. This is done in order to eliminate high correlation between the translations and the rotations, and yields a transformation that is considered to be a better approximation. The Molodensky-Badekas 10-parameter is given by the following equation (OGP 2013):

$$\begin{pmatrix} X_T \\ Y_T \\ Z_T \end{pmatrix} = M \cdot \begin{pmatrix} 1 & rZ & -rY \\ -rZ & 1 & rX \\ rY & -rX & 1 \end{pmatrix} \cdot \begin{pmatrix} X_S - X_P \\ Y_S - Y_P \\ Z_S - Z_P \end{pmatrix} + \begin{pmatrix} tX \\ tY \\ tZ \end{pmatrix} \quad (2.44)$$

Where X_P , Y_P , and Z_P , are the geocentric coordinates of the point about which the coordinate reference frame is rotated, given in the source coordinate system. If these values are set equal to zero, the Molodesky-Badekas transformation becomes identical to the Coordinate Frame Rotation transformation. The Molodensky-Badekas 10-parameter transformation is not considered reversible.

2.5.2 Geographic Transformations

Transformations through geographic coordinates directly relate the latitude, longitude, and ellipsoidal height of one datum system to another. There are three main applicable methods for geographic transformations (Knippers and Hendrikse 2001):

1. the geographic offsets
2. the Molodensky and Abridged Molodensky transformation
3. the multiple regression transformation

Geographic Offsets

Geographic offsets is the simplest method which only requires two parameters, the difference in geographic latitude and longitude ($\Delta\phi$ and $\Delta\lambda$). The ellipsoidal height is omitted in most cases, and the method is only used when low accuracy is tolerated. The coordinates of the target datum are calculated using the following equations:

$$\begin{aligned} \phi_T &= \phi_S + \Delta\phi \\ \lambda_T &= \lambda_S + \Delta\lambda \\ H_T &= H_S + \Delta H \end{aligned} \quad (2.45)$$

This method is generally only used for two-dimensional transformations (ϕ and λ), but can in very rare circumstances be expanded to include ellipsoidal height (H).

Molodensky and Abridged Molodensky

The Molodensky and Abridged Molodensky transformation formulae are what is known as curvilinear transformation formulae (Deakin 2004). The Abridged Molodensky is the same as the standard Molodensky, except that the ellipsoidal height is ignored, and any terms of second order in small parameters are dropped. The method assumes that the Cartesian axes of both datums are parallel and the Cartesian coordinate difference between the two origins are known (ΔX , ΔY , and ΔZ), along with the defining reference ellipsoid parameters (a and f) of both datums. The method uses ϕ , λ , and H , as input variables and calculates the changes, $\Delta\phi$, $\Delta\lambda$, and ΔH . The main problem with the Molodensky transformation is the significant errors produced by the limited amount of used parameters for large countries or continents. The errors could be in the range of tens of meters (Knippers and Hendrikse 2001).

Standard Molodensky Procedure

The Standard Molodensky transformation is given by (Deakin 2004):

$$\begin{aligned} \Delta\phi = \frac{1}{\rho + H} & [-\Delta X \sin\phi \cos\lambda - \Delta Y \sin\phi \sin\lambda + \\ & \Delta Z \cos\phi + \frac{ve^2 \sin\phi \cos\phi}{a} \Delta a + \\ & \sin\phi \cos\phi (\frac{\rho}{1-f} + v(1-f)\Delta f] \end{aligned} \quad (2.46)$$

$$\Delta\lambda = \frac{1}{(v + H) \cos\phi} (-\Delta X \sin\lambda + \Delta Y \cos\lambda) \quad (2.47)$$

$$\begin{aligned} \Delta H = \Delta X \cos\phi \cos\lambda + \Delta Y \cos\phi \sin\lambda \\ + \Delta Z \sin\phi - \frac{a}{v} \Delta a + v(1-f) \sin^2\phi \Delta f \end{aligned} \quad (2.48)$$

Where ΔX , ΔY , and ΔZ , is the difference in datum origin, Δf and Δa are the difference in ellipsoidal parameters, ν is the prime vertical radius of curvature (Equation 2.5), and ρ and is given by the following formula:

$$\rho = \frac{a(1 - e^2)}{1 - e^2 \sin^2\phi} \quad (2.49)$$

The new coordinates are found by adding the calculated changes in geographical coordinates.

Abridged Molodensky Procedure

The Abridged Molodensky transformation uses the following set of equations (Deakin 2004):

$$\Delta\phi = \frac{1}{\rho}[-\Delta X \sin \phi \cos \lambda - \Delta Y \sin \phi \sin \lambda + \Delta Z \cos \phi + (f\Delta a + a\Delta f) \sin 2\phi] \quad (2.50)$$

$$\Delta\lambda = \frac{1}{v \cos \phi}(-\Delta X \sin \lambda + \Delta Y \cos \lambda) \quad (2.51)$$

$$\Delta H = \Delta X \cos \phi \cos \lambda + \Delta Y \cos \phi \sin \lambda + \Delta Z \sin \phi - \Delta a + (f\Delta a + a\Delta f) \sin^2 \phi \quad (2.52)$$

Multiple Regression

The multiple regression method is also known as polynomial transformation. The method consists of a series of best-fit equations which provide the local latitude and longitude shifts for two specified datums as a function of position. The local datums are generally tied to the WGS84 datum or ED50 datum depending on the location. The transformations can involve polynomial expressions in the 9th order and are approximated based on a series of selected points with known coordinates in both datums. The multiple regression transformation can achieve a better fit over continental size areas than the Molodensky transformation. A general equation for polynomial transformations can be written as (Knippers and Hendrikse 2001):

$$\begin{aligned} \Delta\phi &= f(\phi, \lambda, a_1, a_2, a_3, \dots) \\ \Delta\lambda &= f(\phi, \lambda, b_1, b_2, b_3, \dots) \\ \Delta H &= f(\phi, \lambda, c_1, c_2, c_3, \dots) \end{aligned} \quad (2.53)$$

2.5.3 Transformations for Offshore Norway

The oil and gas industry have to report all operations in the ED50 datum for offshore Norway. This has caused the need for a transformation between WGS84 and ED50 since the use of GPS has become the most common practice. The first method of transformation was established in 1990 by Statens Kartverk and is best known as “The North Sea Formulae” (Geodesidivisjonen 1990).

The North Sea Formulae

The North Sea Formulae was derived with ED50 data from western Europe and Norwegian data as far north as 64°N. The lack of data further north resulted in a boundary at 62°N north for the North Sea Formulae. The formulae consists

of a two-step approach where the first part is transforming ED50 to ED87. This transformation calculates the change in latitude and longitude based on a set of 14 polynomial constants. The backwards transformation is obtained by using the negative value of the same set of polynomial constants. The change in latitude and longitude, given in degrees, are computed with the following equations (Geodesidivisjonen 1990):

$$\begin{aligned} \Delta = & A_0 + A_1U + A_2V + A_3U^2 + A_4UV \\ & + A_5V^2 + A_6U^3 + A_7U^2V + A_8UV^2 \\ & + A_9V^3 + A_{10}U^4 + A_{11}U^3V + A_{12}U^2V^2 \\ & + A_{13}UV^3 + A_{14}V^4 \end{aligned} \quad (2.54)$$

$$\begin{aligned} U &= (\textit{latitude(degrees)} - 55) \\ V &= (\textit{longitude(positivedegreeseastGreenwich)}) \end{aligned} \quad (2.55)$$

Where the origin is at ED50: 55°N and 0°E (Greenwich). The polynomial constants A_n can be found in Table A.3.

The second half of the transformation is from ED87 to WGS84*SEA. The term *SEA is to avoid ambiguity, i.e., to clarify that the WGS84 coordinates are derived with the North Sea Formulae. This is achieved through a Bursa-Wolf transformation where the shifts, rotations, and scale change are the following (Geodesidivisjonen 1990):

Shifts		Rotations	
tX	-82.981 m	rX	-0.5076E-6 rad
tY	-99.719 m	rY	0.1503E-6 rad
tZ	-110.709 m	rZ	0.3898E-6 rad
Scale change			
δS	-0.3143 ppm		

Table 2.2: Parameters for transforming from ED87 to WGS84*SEA

The formulae is backwards compatible and can be used by reversing the signs. It should also be noted that the North Sea Formulae only covers the area limited by the 62°N line in the north, the tripoint GB/NL/B in the south, 9°E Greenwich in the east, and a meridian adjacent to the Shetland Islands in the west as seen in Figure 2.7.

North of 62°N

The first transformation method for the area North of 62°N was given by Statens Kartverk in 1991 as a Bursa-Wolf transformation which was to be used north of 65°N, and advocated the use of linear interpolation between 62°N and 65°N (EPSG

2001). This method has since been discarded as of 2001, and the recommended practice is now to use a Bursa-Wolf from 62°N rather than interpolating. The following parameters should be used for the transformation from WGS84 to ED50 (EPSG 2001):

Shifts		Rotations	
tX	+116.641 m	rX	-4.327E-6 rad
tY	+56.931 m	rY	-4.464E-6 rad
tZ	+110.559 m	rZ	+4.444E-6 rad
Scale change			
δS	+3.520 ppm		

Table 2.3: Parameters for transforming from WGS84 to ED50 north of 62°N

When transforming the other way, from ED50 to WGS84 the signs should be reversed.

Simplified transformation for south of 62°N

The same note which details the Bursa-Wolf transformation for north of 62°N also suggests a Bursa-Wolf transformation for the area covered by the North Sea Formulae. This transformation yields an approximation to about 1 metre and uses the following parameters (EPSG 2001):

Translations		Rotations	
tX	+90.365 m	rX	-1.614E-6 rad
tY	+101.130 m	rY	-0.373E-6 rad
tZ	+123.384 m	rZ	-4.334E-6 rad
Scale change			
δS	-1.994 ppm		

Table 2.4: Parameters for a simplified transforming from WGS84 to ED50 south of 62°N

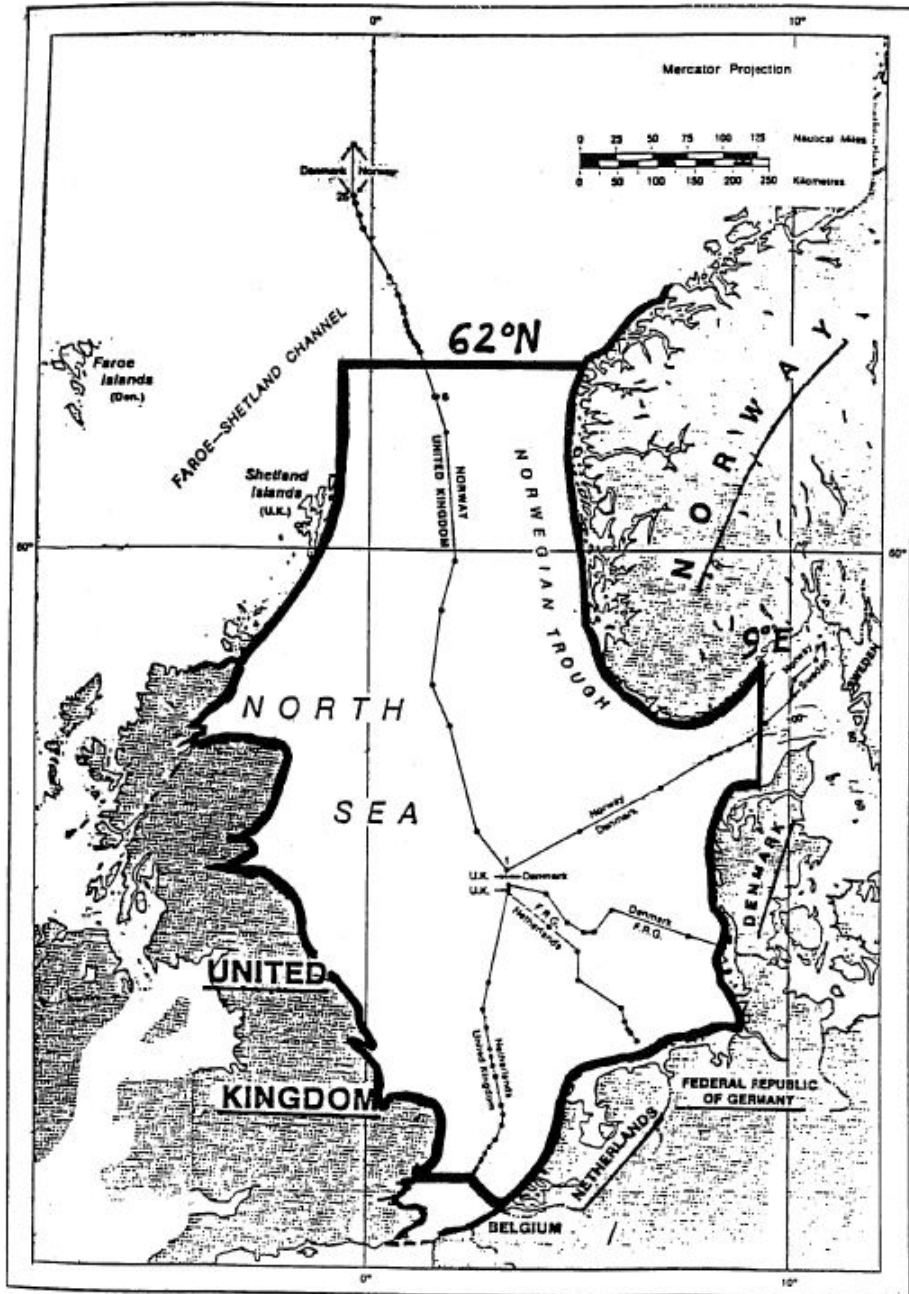


Figure 2.7: The area covered by the North Sea Formulae (Geodesidivisjon 1990)

Chapter 3

Map Projections

A map projection is a transformation of three-dimensional coordinates representing a location on e.g., an ellipsoid, into a two-dimensional location on a plane surface. Map projection can be used for large scale purposes such as atlas maps, world maps, or continental maps. In the oil & gas industry however, map projections are mostly limited to orthomorphic or conformal projections. These map projections are used for topographic and exploration mapping, where it is important to maintain accuracy when scaling positions and distances. (OGP 2013)

3.1 Classification of Map Projections

Map projections can be classified by using both their properties of representation and their graticule groups (Lee 1944). A complete table of classifications by Lee can be seen in Appendix A (Table A.4 and A.5).

3.1.1 Property Groups

The property groups used to classify map projections are mutually exclusive and are known as either conformal, authalic, or aphyllactic. Conformal projections are “projections in which, at any point, the scales in any two orthogonal directions are equal” (Lee 1944). Conformal projections preserve both the shape and angles of elementary areas. This is achieved because the scale at any point is equal in all directions around that point. Conformal projections are also sometimes referred to as orthomorphic or autogonal projections.

Authalic projections are “projections in which, at any point, the scales in two orthogonal directions are inversely proportional” (Lee 1944). These projections preserve a constant area scale, and are sometimes referred to by the colloquial term equal-area, or equivalent.

The last property group, aphyllactic, are projections which are neither conformal nor authalic. This group consists of an infinite amount of possible map projections, but the noteworthy projections are those that are known as balance of errors, or minimum error projections. These projections minimize the sum of the square of the errors in scale, in both the specified orthogonal directions over a mapped area.

3.1.2 Graticule Groups

The graticule groups are distinguished by the pattern formed by the meridians and the parallels. The definitions are as follows (Lee 1944):

- “Cylindric: Projections in which the meridians are represented by a system of equidistant parallel straight lines, and the parallels by a system of parallel straight lines at right angles to the meridians.”
- “Pseudocylindric: Projections in which the parallels are represented by a system of parallel straight lines, and the meridians by concurrent curves.”
- “Conic: Projections in which the meridians are represented by a system of equally inclined concurrent straight lines, and the parallels by concentric circular arcs, the angle between any two meridians being less than their true difference of longitude.”
- “Pseudoconic: Projections in which the parallels are represented by concentric circular arcs, and the meridians by concurrent curves.”
- “Poylconic: Projections in which the parallels are represented by a system of non-concentric circular arcs with their centers lying on the straight line representing the central meridian.”

A subset of the graticule groups can be found by accounting for the aspect of which the projections axis of symmetry coincides with the Earth’s axis. The standard, or normal, projection is when the projections axis of symmetry coincides with the Earth’s axis. The projection is defined as transverse if the axis of symmetry is at right angles to the Earth’s axis. Any angle in between standard and transverse is known as oblique (Figure 3.1) (Lee 1944; Snyder and Voxland 1989).

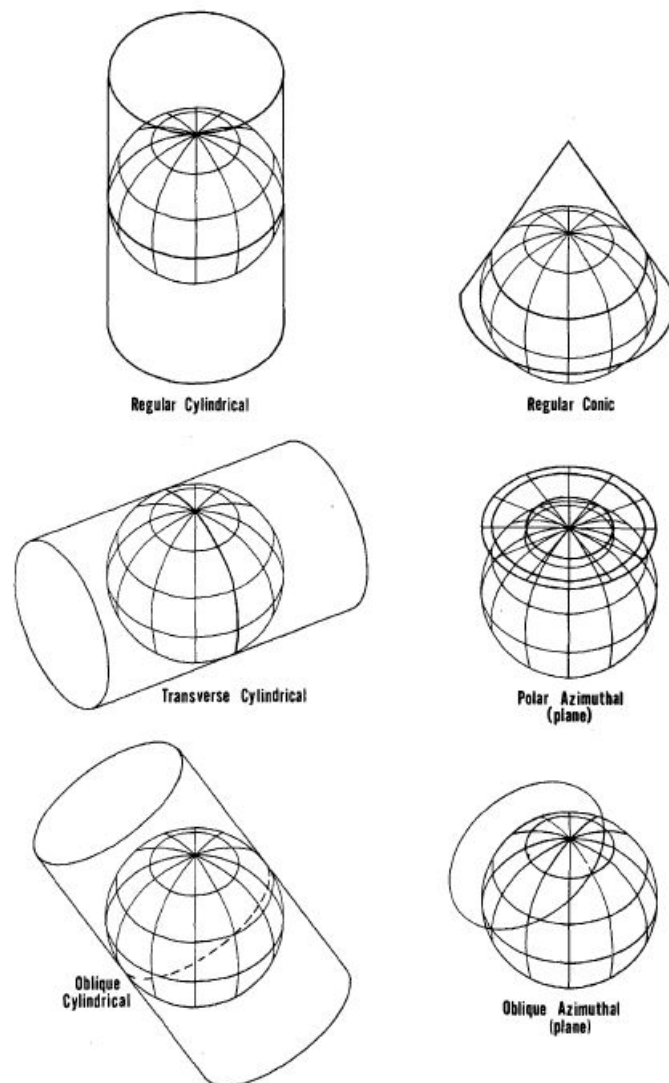


Figure 3.1: Various projections of the Earth (Snyder 1987)

3.2 Parameters related to Map Projections

Both a conversion method and associated parameters are required in order to relate a map projection grid to a geographical graticule. There are many different parameters depending on which conversion method is used, and a complete table can be seen in Appendix A (Tables A.4 & A.5).

Natural Origin is the shared point of both the map and the ellipsoid surface. This point would have the grid coordinates of 0,0 in the absence of false coordinates. The Natural Origin is located in the centre of the map projection (OGP 2013). Defining the centre of the map projection as coordinate 0,0 would lead to the required use of negative coordinates when moving away from the centre. Map projections are often assigned values of False Easting (FE) and False Northing (FN) to counter the need for negative coordinates. Adding false values to the map

projection leads to a shift of the 0,0 coordinate point. This point is known as the Grid Origin when false values are in place.

Another common parameter is the scale factor at the origin. The function of the scale factor is to limit the scale distortion within the projected area. This is achieved by reducing the nominal scale of the map at the origin, and achieving nominal scale at a distance away from the origin. The scale factor is usually denoted as k_0 (OGP 2013).

3.3 The Mercator Projections

3.3.1 Mercator

The Mercator projection is most likely the first named projection and was developed by Gerardus Mercator in 1569. The projection is cylindrical and maintains conformality. It is constructed of equally spaced meridians of longitude, cut by unequally spaced straight parallels. The spacing of the parallels increases towards the poles and is proportional to the secant of the latitude (Snyder 1987). The fact that it is a cylindrical projection along the Equator will lead to great distortion when nearing the poles. In fact, both the North and South Pole cannot be seen on the projection, since they are spaced infinitely away from the parallels. The distortion is best seen when comparing Greenland and South America, as seen in Figure 3.2. Greenland appears to be larger than South America when seen on the Mercator projection, but the reality of the situation is that Greenland is closer to one-eighth of the size of South America.

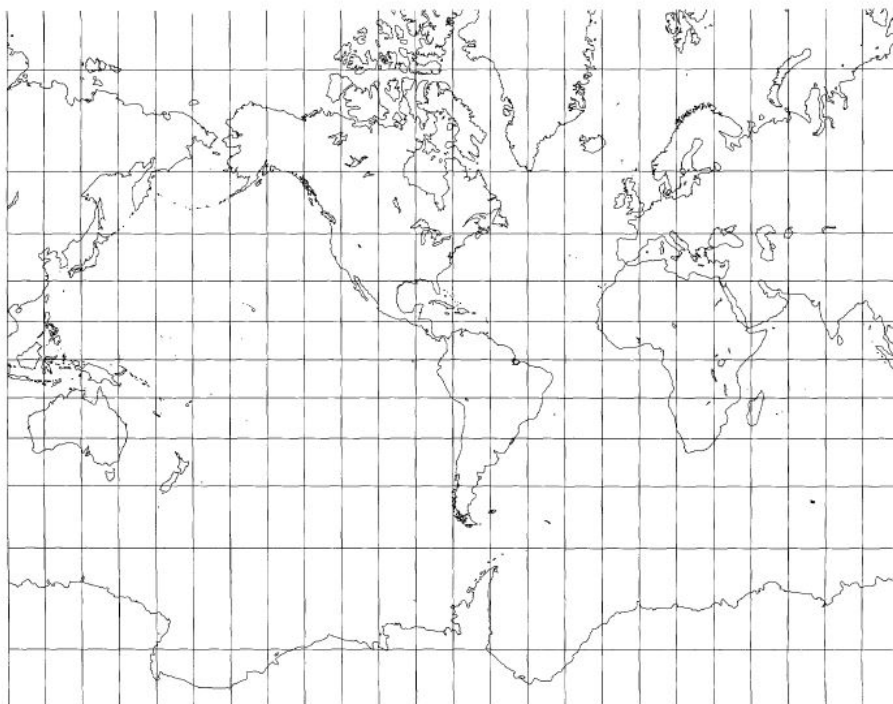


Figure 3.2: The Mercator projection (Snyder 1987)

3.3.2 Transverse Mercator

The Transverse Mercator is one of the most common projections for large-scale maps, and is also known as the Gauss-Krüger projection (Karney 2011). The projection is, as the name indicates, the transverse form of the Mercator projection. Rather than wrapping a cylinder along the Equator, the Transverse Mercator coincides with the central meridian throughout its length. Unlike the Mercator projection, the meridians and parallels now form complex curves rather than straight lines (Snyder 1987). The projection is still conformal, and the distortion increases with the distance away from the central meridian. The projection can be seen in Figure 3.3.

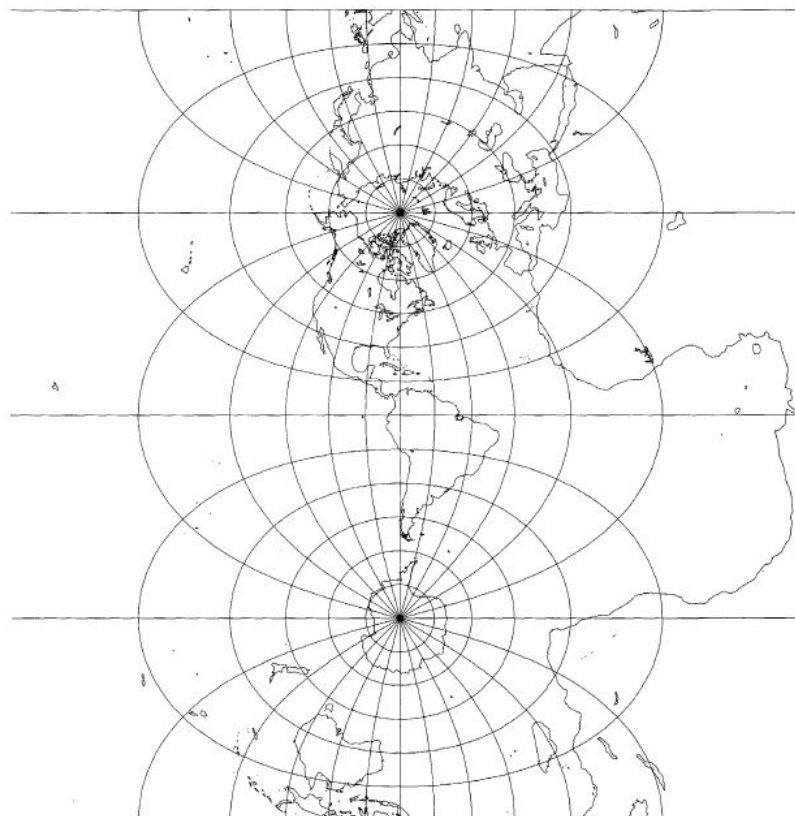


Figure 3.3: The Transverse Mercator projection (Snyder 1987)

Mathematical Formulae

The first mathematical projection of the Transverse Mercator was suggested by Johann Heinrich Louis Krüger in 1912 (Krüger 1912). Krüger suggested two truncated series based on the work of Johann Carl Friedrich Gauss. The suggested series are today known as the Krüger- n series, and the Krüger- λ series.

The Krüger- λ series are expansion series expressing the longitude difference from the central meridian. The series was further developed by L.P. Lee in 1946, and extended to the eighth order by Redfearn in 1948, becoming what is now known as the Lee-Redfearn-OSGB series (OS 2013). This series is the basis of the grid map of Great Britain and is given by Equations 2.12 - 2.36. The original

series was also recalculated by Paul Thomas in 1952, and is today known as the Thomas-UTM series (Snyder 1987; OGP 2013). The Thomas-UTM series was chosen as the basis for the UTM system by the United States Defence Mapping Agency (DMA 1989), which can be found in the UTM section below.

The n-series is often referred to as the JHS formulae, and is currently used by France, Finland, Sweden, and Japan, of the fourth order (OGP 2013). The proposed n-series by the Finnish Geodetic Institute in 2006 has been further developed by Karney in 2011 (Karney 2011). Karney proves and concludes that the Thomas series is unnecessarily inaccurate for Transverse Mercator projections, leading to maximum errors of over 1 km when far away from the central meridian. In comparison, the n-series of the fourth order yields errors of less than 1 μm (Karney 2011) .

3.4 Universal Transverse Mercator

The Universal Transverse Mercator (UTM) was developed by the U.S. Army in 1947 for accurate rectangular coordinates on large-scale military maps (Snyder 1987). The projection divides the world into 60 longitude zones, where each zone is 6° wide (Figure 3.4). These zones are valid between 84° N and 80°S (Snyder 1987). The zone number can be calculated based on the following formula (DMA 1989)

$$Z = 1 + INT\left(\frac{\lambda + 180}{W}\right) \quad (3.1)$$

Where Z is the zone number in UTM, INT means the whole integer, rounded down, λ is the longitude in degrees, 180 is the amount of degrees on each hemisphere, and W is the width of a zone in degrees.

The position is given by values of Easting and Northing along with the zone number. Both Easting and Northing are given in meters. The UTM system utilizes false values of Easting and Northing to avoid negative values. The false Easting (FE) is set as 500 000 m west of the zones central meridian. This means that Easting values are given as meters east from a line which is located 500 000 m west of the zones central meridian (Snyder 1987; DMA 1989).

Northings are given as meters north of Equator for positions in the Northern Hemisphere. For the Southern Hemisphere, a false Northing (FN) of 10 000 000 m is applied. This means that the negative values of Northings south of the equator are subtracted from the FN (Snyder 1987; DMA 1989).

U.S. Military Grid Reference System

Latitude zones of alphabetic distinction are not a part of the UTM system. They are, however, a part of the U.S. Military Grid Reference System (MGRS). This system is defined as an alphanumeric adaption of the UTM system, and was created by the U.S. Army (DMA 1990). In addition to the latitude bands, exceptions for zone widths for mainland Norway and regions around Svalbard have been added. Zone 32 is, at the expense of zone 31, widened to 9° between latitude 56° and 64°.

For the Svalbard region, zones 33 and 35 are, between 72° and 84°, widened to 12°. This is compensated by widening zones 31 and 37 to 9°, and eliminating zones 32, 34, and 36 (DMA 1990).

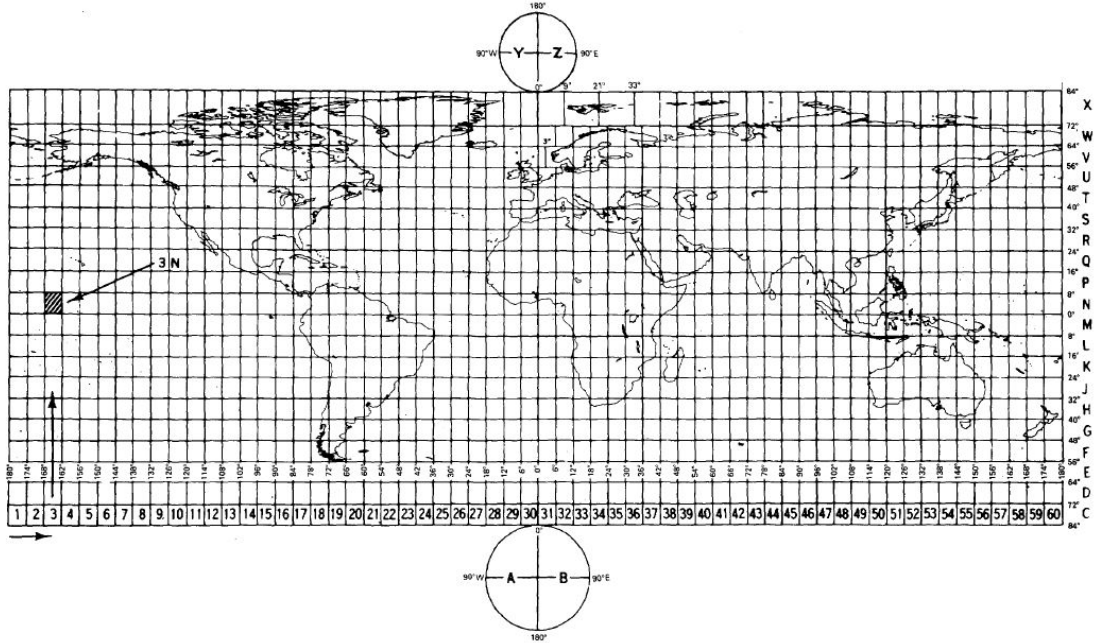


Figure 3.4: A map of the world showing the MGRS zones with UTM longitude bands (Snyder 1987)

3.4.1 Converting from geographical coordinates to Northing and Easting (Thomas-UTM)

The Northing and Easting values can be calculated using the Thomas-UTM series with accurate precision (Karney 2011; Snyder 1987; OGP 2013; DMA 1989):

$$E = FE + k_0\nu\left[A + (1 - T + C)\frac{A^3}{6} + (5 - 18T + T^2 + 72C - 58e'^2)\frac{A^5}{120}\right] \quad (3.2)$$

$$N = FN + k_0\left[M - M_0 + \nu \tan \phi\left(\frac{A^2}{2} + (5 - T + 9C + 4C^2)\frac{A^4}{24} + (61 - 58T + T^2 + 600C - 330e'^2)\frac{A^6}{720}\right)\right] \quad (3.3)$$

where all angles are expressed in radians, A, T, C, M, and M_0 are auxillary parameters given below, FE and FN are the false Easting and false Northing respectively, e' is the second eccentricity (Equation 2.3), ν is the prime vertical

radius of curvature using radians (Equation 2.5), and k_0 is the scale factor of the central meridian.

The auxillary parameters are given as:

$$T = \tan^2 \phi \quad (3.4)$$

$$C = \frac{e^2 \cos^2 \phi}{1 - e^2} \quad (3.5)$$

$$A = (\lambda - \lambda_0) \cos \phi \quad (3.6)$$

$$\begin{aligned} M = & a \left(\left(1 - \frac{e^2}{4} - \frac{3e^4}{64} - \frac{5e^6}{256} - \dots \right) \phi \right. \\ & - \left(\frac{3e^2}{8} + \frac{3e^4}{32} + \frac{45e^6}{1024} + \dots \right) \sin(2\phi) \\ & + \left(\frac{15e^4}{256} + \frac{45e^6}{1024} + \dots \right) \sin(4\phi) \\ & \left. - \left(\frac{35e^6}{3072} + \dots \right) \sin(6\phi) + \dots \right) \end{aligned} \quad (3.7)$$

$$\begin{aligned} M_0 = & a \left(\left(1 - \frac{e^2}{4} - \frac{3e^4}{64} - \frac{5e^6}{256} - \dots \right) \phi_0 \right. \\ & - \left(\frac{3e^2}{8} + \frac{3e^4}{32} + \frac{45e^6}{1024} + \dots \right) \sin(2\phi_0) \\ & + \left(\frac{15e^4}{256} + \frac{45e^6}{1024} + \dots \right) \sin(4\phi_0) \\ & \left. - \left(\frac{35e^6}{3072} + \dots \right) \sin(6\phi_0) + \dots \right) \end{aligned} \quad (3.8)$$

where the angles are still expressed in radians, e is the first eccentricity (Equation 2.2), ϕ_0 and λ_0 is the latitude and longitude of natural origin respectively:

$$\phi_0 = 0 \quad (3.9)$$

$$\lambda_0 = (3 + W(Z - 1) - 180) \quad (3.10)$$

3.4.2 Converting from Northing and Easting to to geographical coordinates (Thomas-UTM)

The latitude and longitude values can be calculated using the Thomas-UTM series (Karney 2011; Snyder 1987; OGP 2013; DMA 1989):

$$\begin{aligned} \phi = \phi_1 - \frac{\nu_1 \tan \phi_1}{\rho_1} & \left[\frac{D^2}{2} - (5 + 3T_1 + 10C_1 - 4C_1^2 - 9e'^2) \frac{D^4}{24} \right. \\ & \left. + (61 + 90T_1 + 298C_1 + 45T_1^2 - 252e'^2 - 3C_1^2) \frac{D^6}{720} \right] \end{aligned} \quad (3.11)$$

$$\lambda = \lambda_0 + \frac{1}{\cos(\phi_1)} \left[D - (1 + 2T_1 + C_1) \frac{D^3}{6} + (5 - 2C_1 + 28T_1 - 3C_1^2 + 8e^2 + 24T_1^2) \frac{D^5}{120} \right] \quad (3.12)$$

Where ϕ and λ are latitude and longitude respectively, in radians, ϕ_1 , D , T_1 , C_1 , ν_1 , and ρ_1 are auxillary parameters given below along with μ_1 , e_1 , M_1 , and M_0 :

$$T_1 = \tan^2 \phi_1 \quad (3.13)$$

$$C_1 = e'^2 \cos^2 \phi_1 \quad (3.14)$$

$$\nu_1 = \frac{a}{\sqrt{1 - e^2 \sin^2 \phi_1}} \quad (3.15)$$

$$\rho_1 = \frac{a(1 - e^2)}{(1 - e^2 \sin^2 \phi_1)^{1.5}} \quad (3.16)$$

$$\begin{aligned} \phi_1 = \mu_1 + & \left(e_1 \frac{3}{2} + e_1^3 \frac{27}{32} + \dots \right) \sin(2\mu_1) \\ & + \left(e_1^2 \frac{21}{16} - e_1^4 \frac{55}{32} + \dots \right) \sin(4\mu_1) \\ & + \left(e_1^3 \frac{151}{96} + \dots \right) \sin(6\mu_1) \\ & + \left(e_1^4 \frac{1097}{512} + \dots \right) \sin(8\mu_1) + \dots \end{aligned} \quad (3.17)$$

$$e_1 = \frac{1 - \sqrt{1 - e^2}}{1 + \sqrt{1 + e^2}} \quad (3.18)$$

$$\mu_1 = \frac{M_1}{a \left(1 - e^2 \frac{1}{4} - e^4 \frac{3}{64} - e^6 \frac{5}{256} - \dots \right)} \quad (3.19)$$

$$M_1 = M_0 + \frac{N - FN}{k_0} \quad (3.20)$$

$$\lambda_0 = (3 + 6(Z - 1) - 180) \quad (3.21)$$

$$\phi_0 = 0 \quad (3.22)$$

where angles are still expressed in radians, e and e' are the first and second eccentricity respectively, a is the major-axis of the ellipsoid, M_0 is given by Equation 3.8), FN and FE stands for false Northing and false Easting respectively, k_0 is the scale factor of the central meridian, and Z indicates the zone number.

Chapter 4

Position Model

4.1 MATLAB as a Platform

The task of the Thesis was to develop a model/calculator for handling coordinates within geodetic systems and map projections. The calculations for these conversions and transformations require many sub-calculations which are often the same for different methods. Thus, the most logical approach was to develop the models in a system that supported the use of auxiliary functions which could be called upon when needed. MATLAB is capable of doing just that, as well as having built-in matrix handling capabilities which was essential for transformations such as the Bursa-Wolf transformation. MATLAB is by default a very accurate program, with a variable-precision accuracy of 32 digits (MATLAB 2014b). The variable-precision accuracy can be manually specified to any integer between 1 and $2^{29} + 1$ (roughly 0.5 billion) with the Symbolic Math Toolbox, which unfortunately is not a part of the NTNU licence. An added benefit is that the programming language of MATLAB is a fairly intuitive one. The benefit of using an intuitive programming language is that it makes it easy for others to understand, and then make use of or continue the work. The second choice for a platform was Excel. Excel would provide a decent graphical overview, but all functions have to either be made through macros, or in each cell. This could potentially lead to a model which would be hard, or impossible, to alter in a later stage. This would also make it difficult for others to continue the work. The precision of computation is also vastly smaller in Excel. Excel is designed around IEEE 754 when it comes to floating-point numbers, and thus has a max value of 15 significant digits of precision (Excel 2014).

4.2 Converting between Decimal Degrees and Degrees, Minutes, Seconds

Degrees, Minutes, and Seconds (DMS), are by far the most common way of expressing geographical coordinates. They are, however, not used for computations. All transformations and coordinate conversions make use of decimal degrees (DD). A full circle consists of 360 degrees, one degree consists of 60 minutes, and one

minute consists of 60 seconds. The implementation of these relations in MATLAB are fairly straightforward, but requires a lot of conditional statements which handles the cases of negative number inputs. The functions can be seen in Appendix B as “coord2dec” and “dec2coord”. When going from DMS to DD it is important to maintain the sign notation of the largest number. This was implemented as a series of conditional IF statements checking the value of degrees, minutes, and seconds in turn, in order to properly add the values to DD with the correct sign notation. Going back from DD to DMS is a slightly more complicated ordeal when programming. The function utilizes the built-in floor function in order to round down positive numbers to the nearest low integer, as well as the built-in fix function to round down negative numbers to the integer closest to 0. Combining this with numerous IF statements makes it possible to account for the size of the DD number, in order to correctly relate it to either negative degrees, minutes, or seconds. It should also be noted that MATLAB has built-in functions which does exactly this, known as degrees2DMS and DMS2degrees. Unfortunately, they are a part of the Mapping Toolbox, which is not a part of licence available to NTNU. However, MATLAB has given example values for all possible input and output variations in DMS and DD, which the built functions have been tested against and proved to be completely accurate (MATLAB 2014a).

4.3 Converting between Cartesian and Geographical Coordinates

Coordinates are normally presented in their geographical form, as latitude, longitude, and ellipsoidal height. However, in order to transform between WGS84 and ED50 offshore Norway, the coordinates have to be in Cartesian form, as X, Y, and Z values. This is because both methods utilizes the Bursa-Wolf transformation, which is a geocentric transformation.

The forward conversion from geographic to Cartesian is a straight forward computation which uses the radius of curvature in the prime vertical to relate the latitude, longitude, and height, to X, Y, and Z parameters in combination with the ellipsoid parameters a and b. This can be seen in Appendix B as ”geo2cart”. The only computational issue lies in the backwards conversion, from Cartesian to geographic. As seen in “cart2geo”, from Appendix B, the computation is an iterative one. This is because both the latitude and ellipsoid height are dependent on the same equation. This is solved by using a WHILE condition that loops the calculations until the precision of the computation is sufficient. In “cart2geo” the tolerated difference between one latitude and the next has been set to 10^{-20} . In terms of meters, this would relate to a difference in the order of approximately 10^{-15} , varying slightly depending on the position. The functions have been verified by computing both forward and backwards for a large set of coordinates and calculating the difference between the original input and final output. The difference was non-existent with 32 digit precision.

4.4 Ellipsoid Parameters

The function “ellipsoid” is a database constructed of IF statements, which checks an input, or ID code, and returns the parameters related to that ID code. It will also return an error message if the ID code is invalid or unknown. This function exists because the ellipsoid parameters are involved in many computations, and it creates a better overview and functionality than merely defining the relevant ellipsoid parameters for every unique case. The ellipsoid parameters are taken from the National Geospatial-Intelligence Agency of the United States, and matches those found from other sources.

4.5 Polynomials in the North Sea Formulae

The polynomial function “northsea1” calculates the changes for latitude and longitude when transforming between ED87 and ED50 in either direction. The function takes in an input “A”, which should be set equal to 1 when going from ED50 to ED87, and as -1 when going the other way. It uses two sets of 14 polynomials for either latitude or longitude respectively given by Statens Kartverk as described in Chapter 2. The function has been verified using a set of test values which describes the transformation from ED87 to ED50. They can be found in Table A.6 (Geodesidivisjonen 1990). The deviation between the computed values and the given values are:

Latitude				Longitude				Distance
DD	D	M	S	DD	D	M	S	meter
-5.94E-09	0	0	-2.14E-05	1.15E-08	0	0	4.15E-05	0.00143697
7.02E-09	0	0	2.53E-05	-5.66E-09	0	0	-2.04E-05	0.00099866
1.09E-08	0	0	3.92E-05	1.47E-08	0	0	5.28E-05	0.00202552
-3.52E-10	0	0	-1.27E-06	-6.70E-09	0	0	-2.41E-05	0.00074383
-1.23E-08	0	0	-4.42E-05	-7.05E-09	0	0	-2.54E-05	0.00156582
-1.54E-08	0	0	-5.53E-05	2.82E-09	0	0	1.01E-05	0.00172731
1.15E-08	0	0	4.15E-05	-1.03E-09	0	0	-3.73E-06	0.00128006
1.52E-08	0	0	5.46E-05	-4.80E-09	0	0	-1.73E-05	0.00175922
9.53E-10	0	0	3.43E-06	-4.00E-09	0	0	-1.44E-05	0.00045662
2.61E-09	0	0	9.41E-06	2.15E-09	0	0	7.74E-06	0.00037477
-9.90E-09	0	0	-3.56E-05	1.04E-08	0	0	3.74E-05	0.00159035

Table 4.1: Deviation from test values found using the script “test_NorthSea1”.

4.6 Bursa-Wolf in the North Sea Formulae

The “helmert7” function is a Bursa-Wolf transformation. The function uses the geocentrical source coordinates along with the 7 parameters needed for computation as input. The output is the geocentrical target coordinates. The function has been verified using two sets of test values. The first set describes the transformation from ED87 to WGS84*SEA, found in Table A.7 (Geodesidivisjonen 1990). The computed deviation from these values are:

Latitude				Longitude				Distance
DD	D	M	S	DD	D	M	S	meter
2.40E-08	0	0	8.64E-05	1.58E-08	0	0	5.70E-05	0.00318313
1.81E-08	0	0	6.53E-05	2.24E-08	0	0	8.07E-05	0.00320149
1.24E-08	0	0	4.47E-05	2.34E-08	0	0	8.43E-05	0.00294672
3.85E-12	0	0	1.39E-08	3.09E-08	0	0	0.00011127	0.00344067
9.73E-09	0	0	3.50E-05	1.91E-08	0	0	6.89E-05	0.00238619
1.12E-08	0	0	4.04E-05	2.58E-08	0	0	9.30E-05	0.00313059
1.62E-10	0	0	5.83E-07	1.67E-08	0	0	5.99E-05	0.00184757
2.12E-08	0	0	7.62E-05	1.17E-08	0	0	4.22E-05	0.0026764
2.52E-08	0	0	9.06E-05	2.27E-08	0	0	8.16E-05	0.00375692
6.27E-09	0	0	2.26E-05	1.40E-08	0	0	5.05E-05	0.00170454
1.15E-08	0	0	4.16E-05	2.51E-08	0	0	9.04E-05	0.00307306

Table 4.2: Deviation from test values found using the script ”test_NorthSea”.

The final set of test values are for a complete transformation from ED50 to WGS84 using the Bursa-Wolf transformation with parameters for areas north of 62°N. The values can be found in Table A.8 (EPSG 2001) and the deviation is:

Latitude				Longitude				Distance
DD	D	M	S	DD	D	M	S	meter
-3.92E-08	0	0	-0.00014108	2.89E-07	0	0	0.00104102	0.03248062
1.13E-07	0	0	0.00040761	2.90E-07	0	0	0.00104528	0.03466205
1.59E-07	0	0	0.00057187	4.66E-07	0	0	0.00167811	0.05478244
1.81E-07	0	0	0.00065166	4.15E-07	0	0	0.00149222	0.05029649
1.80E-07	0	0	0.00064651	4.13E-07	0	0	0.00148635	0.05006677
1.54E-07	0	0	0.00055594	4.61E-07	0	0	0.00165919	0.05407195
1.05E-07	0	0	0.00037948	2.80E-07	0	0	0.00100946	0.03331935
3.24E-08	0	0	0.00011666	4.27E-07	0	0	0.00153586	0.0476264
-6.47E-08	0	0	-0.00023299	3.44E-07	0	0	0.00123709	0.03891651
9.17E-08	0	0	0.00033008	3.09E-07	0	0	0.00111183	0.03584356
-5.41E-08	0	0	-0.00019459	3.22E-07	0	0	0.00115878	0.03632654
9.79E-08	0	0	0.0003526	3.19E-07	0	0	0.00114833	0.03712327

Table 4.3: Deviation from test values found using the script ”test_Northof62”.

4.7 Computing the Distance between two Coordinates

The distance in meters between two sets of geographical coordinates can be computed by using the function “vdist”. The function was created by Michael Kleder (Kleder n.d.) and is built upon algorithms described by Vincenty (Vincenty 1975). The function is precise to within 0.01 mm for any position on a WGS84 ellipsoid, with the exception of the poles. At the poles, a position is shifted by 0.6 mm in order to be computed.

4.8 Computing UTM Coordinates

There are two functions which deal with UTM. The first, “coord2grid_USGS”, takes geographical coordinates, along with ellipsoid parameters, and computes the Northing, Easting, zone number, and a parameter indicating whether the position is located on the Northern or Southern hemisphere (indicated by 1 for north and -1 for south). The computation is achieved by using the Thomas-UTM formulae. The function will first check if the given geographical coordinates are valid, and promptly produce an error code if they are not. It will then compute the false Northing along with the hemisphere indicator. Next, the zone number is computed, and along with it the longitude of natural origin which is used in the final computation. The last part of the function is the computation of the auxiliary parameters T , C , A , ν , M , and M_0 , which are all used for computing the Easting, E , and Northing, N . The auxiliary parameters are all truncated series where all coordinates are expressed in radians.

The second function, “grid2coord_USGS”, is the reverse calculation. It takes in the Northing, Easting, zone number, hemisphere indicator, along with the ellipsoid parameters, and computes the longitude and latitude. The reverse function is constructed much like the forward function, with a validity check at first, followed by an IF statement which dictates the false Northing depending on the hemisphere indicator. The function then proceeds to compute the auxiliary parameters M_0 , M_1 , μ_1 , e_1 , T_1 , C_1 , ν_1 , ρ_1 , and D , which are subsequently used to compute the longitude and latitude in radians. The function then computes the latitude and longitude degrees and return them as output.

The functions have been verified by running forwards and backwards computations for a multitude of values both for the Northern and Southern Hemisphere. The differences between computed values and real values were zero. The functions have also been tested against several online UTM calculators (Dutch 20014; Taylor 2003). The computed result matched that of the online calculators, albeit more precise.

4.9 Modelled Example Case

A worked example has been created in order to exemplify the formulae and functions presented in the Thesis. The example aims to describe the exact locations of wells in the North Sea and the Norwegian Sea without the loss of information when converting between different coordinate forms, and transforming between different geodetic systems. The coordinates are also projected into UTM coordinates as the end result.

4.9.1 Background

A rig is located south of the 62°N latitude band, and a target exists north of the 62°N latitude band. The coordinates for both the rig and the target have been given as geographical coordinates for the ED50 system. The target should be expressed in terms of ΔN , ΔE , and ΔTVD away from the wellhead.

4.9.2 Wellhead Coordinates

The rig is located at 61° 59' 30.45"N and at 02° 30' 20.10"E. The rig is assumed to be at MSL and the water depth is 180 m. The first step is to convert the DMS coordinates into DD coordinates by using the function "coord2dec" for both latitude and longitude. This results in 61.99179°N and 2.50558°E.

The next step is to find the geographic offset required to transform the coordinates from ED50 to ED87. This is achieved by using Equation 2.54 which is implemented as the function "northsea1". The function produces a latitude offset of $2.1767^\circ \cdot 10^{-6}$ and a longitude offset of $-2.4563^\circ \cdot 10^{-5}$.

The ED87 coordinates are found by applying the geographic offset as described by Equation 2.45. It is assumed that ED50 and ED87 ellipsoids have no difference in ellipsoidal height.

The ED87 coordinates will have to be converted into geocentric coordinates before they can be transformed further. The geocentric coordinates are computed using Equation 2.4. This set of equations require the appropriate ellipsoid parameters, which in this case is the ellipsoid for ED87. This ellipsoid is known as International 1924, and the parameters are found by using the function "ellipsoid" for the ID code 17. The coordinates are then converted using the function "geo2cart". This results in the geocentric coordinates (X, Y, Z) 3000286.16, 131286.96, and 5608208.94.

Now that the geocentric coordinates have been found, the next step is to transform the ED87 coordinates into WGS84*SEA coordinates using the Bursa-Wolf transformation. The Bursa-Wolf transformation is given by Equation 2.38, and the appropriate parameters are found in Table 2.2. The transformation is computed by using the function "helmert7". The geocentric WGS84*SEA coordinates then becomes 3000203.02, 131191.21, and 5608095.95.

Since the UTM formulae only accepts geographical coordinates, the geocentric coordinates will have to be converted into geographical coordinates. The

geographical coordinates are found using the function “cart2geo” which is an implementation of Equations 2.6 - 2.10. The geographical WGS84*SEA coordinates are 61.99132°N, 2.50380°E, and 39.11 m below ellipse surface.

The Thomas-UTM series described by Equations 3.2 - 3.10 is implemented as the function “coord2grid_USGS”. This function also requires the ellipsoid parameters in addition to the longitude and latitude. The ellipsoid for WGS84 is given by the ID code 23. Running the function yields 6873313.43 Northing, 474003.35 Easting, and the zone number 31.

Going from rig coordinates to wellhead coordinates is straightforward. The 39.11 m ellipsoidal height is subtracted from the water depth (180 m) which produces a TVD of 140.88 m below ellipse surface. The Northing, Easting, and zone number is the same.

4.9.3 Target Coordinates

The target is given in geographical ED50 coordinates as 62° 01' 04.30"N, 02° 30' 40"E and H is 2300 m below ellipse surface. Just as for the rig, the coordinates will have to be converted from DMS to DD, and then into geocentric coordinates. The geocentric coordinates then becomes 2996632.50, 131418.03, and 5607541.53.

The geocentric ED50 coordinates can now be transformed into WGS84 coordinates using the Bursa-Wolf transformation, but with different parameters than before. The parameters for the area north of 62° are found in Table 2.3. The transformation is computed using the function “helmert7” with the specified parameters. This results in the geocentric WGS84 coordinates 2996530.93, 131323.06, and 5607398.42. The major difference between the target and the rig is that this transformation does not go through the ED87 datum, but is a direct transformation.

The UTM coordinates for the target are found in the same manner as that of the rig coordinates. The result is 6876217.28 Northing, 474317.69 Easting, zone number 31, and a TVD of 2296.14 m.

4.9.4 Solution

Calculating the change in Northing, Easting, and TVD between the wellhead and the target is straightforward. The wellhead is given the position 0, 0, 0 by subtracting the wellhead coordinates. The wellhead coordinates are also subtracted from the target coordinates, producing 2903.84 Northing, 314.34 Easting, and 2155.25 TVD.

The example case can be found as a script in Appendix C with all relevant data.

Chapter 5

Discussion

5.1 Iteration in the function for converting Cartesian to Geographic coordinates

Geographic coordinates are computed from Cartesian coordinates using the function “cart2geo”. The function uses iterative mathematics, where a WHILE loop checks the difference between step 1 and step 2, and stops only when a certain level of accuracy has been reached. The level of accuracy was set to $1 \cdot 10^{-25}$ for all verification and computation, which proved to be more than sufficient. However, it is important to understand the limits of all functions. A script was made, which alters the level of accuracy 10% at a time in order to fully comprehend the importance of the accuracy level. The script runs this alteration from $1 \cdot 10^{-25}$ to $1 \cdot 10^3$, and computes the deviation from true value and computed value, as well as monitoring the amount of iterative steps required to reach the accuracy.

Figure 5.1 shows the graphical output of the 700 data-sets generated by the script. The figure shows that any accuracy level set higher than $6 \cdot 10^{-13}$ will give the least amount of computational error, and run 5 steps. Up until an accuracy level of $1.2 \cdot 10^{-10}$ the function will use 4 iterative steps. Both 5 and 4 steps yield the same deviation, 1.7 nm. 3 iterative steps yield a deviation of 64.6 nm. The next step-change occurs at an accuracy level of $2.67 \cdot 10^{-8}$. Now only 2 iterative steps are required, yielding a considerably higher deviation of 13.7 μm . The final step-change occurs at an accuracy level of $6.1 \cdot 10^{-6}$, after which only 1 step is required. At 1 step the function is at its most unreliable, yielding a deviation of 2.9 mm. Although all deviations can be said to be negligible in terms of drilling and positioning, the fact is that by merely increasing the iterative process from 1 step to 2 steps, the accuracy is increased by over 4700 times. For a single set of coordinates, the added computational effort for maximum accuracy is negligible.

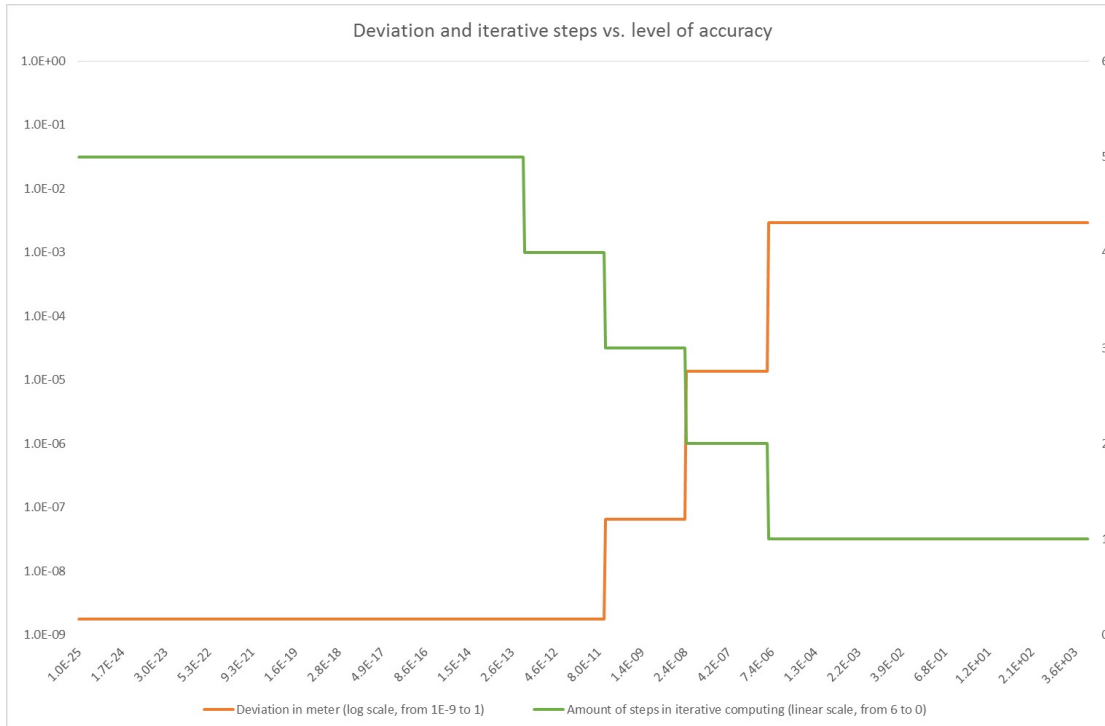


Figure 5.1: The effect of decreasing the level of accuracy required for the iterative function "cart2geo".

5.2 Bursa-Wolf vs. The North Sea Formulae

Although the official recommendation by Statens Kartverk is the use of a Bursa-Wolf transformation north of 62°N , and the North Sea Formulae south, there is a discrepancy between the computed results at the boundary. This difference is important to be aware of, especially when the target and platform are on separate sides of the border.

The script "test_62degext" computes the difference between transforming with either Bursa-Wolf or the North Sea Formulae, using the same set of coordinates. The longitude coordinates represent the area between the UK/Norwegian border and mainland Norway. The amount of elements between these two points can be altered by changing the parameter "n". Similarly, the latitude coordinates are given with a set of start and end coordinates that represent the area near the 62°N latitude band, where the amount of elements can be altered by changing the parameter "d". The script will then proceed to run both transformations for a matrix of n rows and d columns, and finally compute the difference between each point in both degrees, minutes, seconds, and in meters. The result is also converted into a colored image by using the bar3 function.

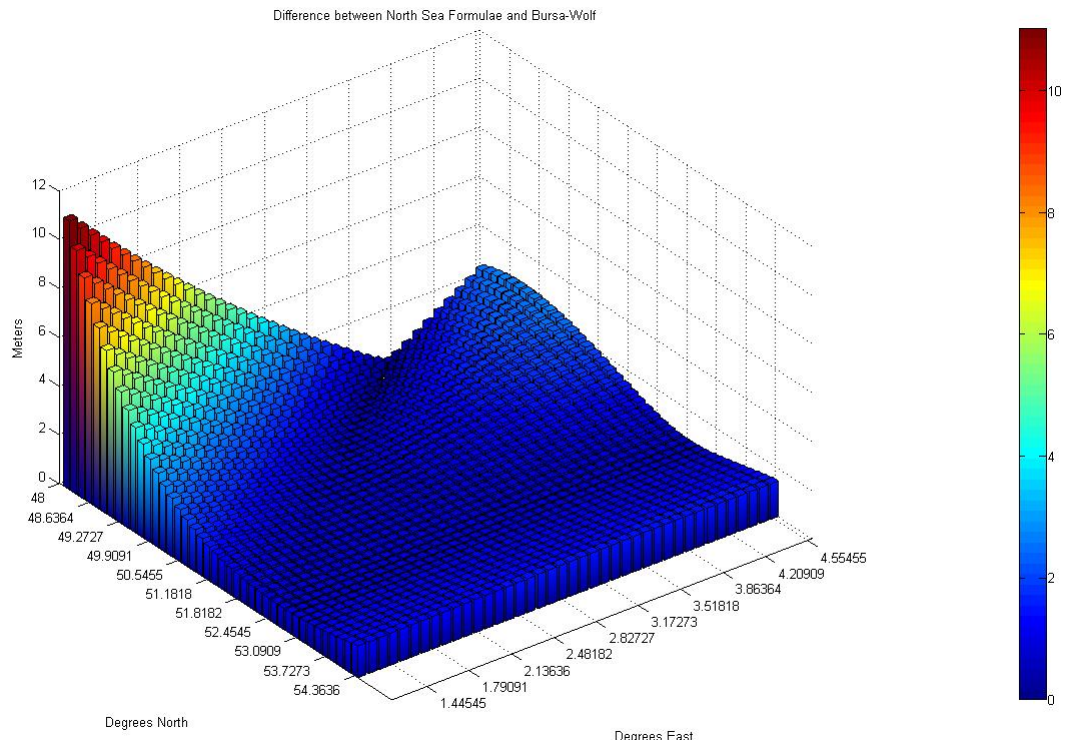


Figure 5.2: The computational difference between Bursa-Wolf and The North Sea Formulae for a latitude interval of 48-55, offshore Norway.

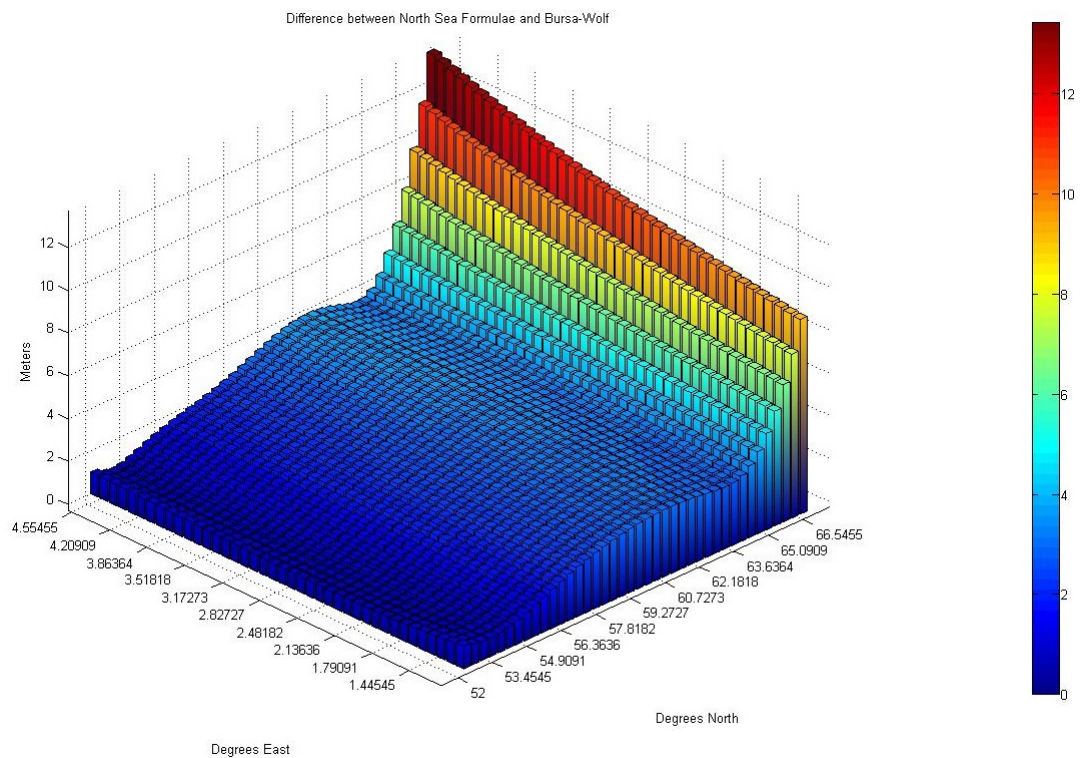


Figure 5.3: The computational difference between Bursa-Wolf and The North Sea Formulae for a latitude interval of 52-68, offshore Norway.

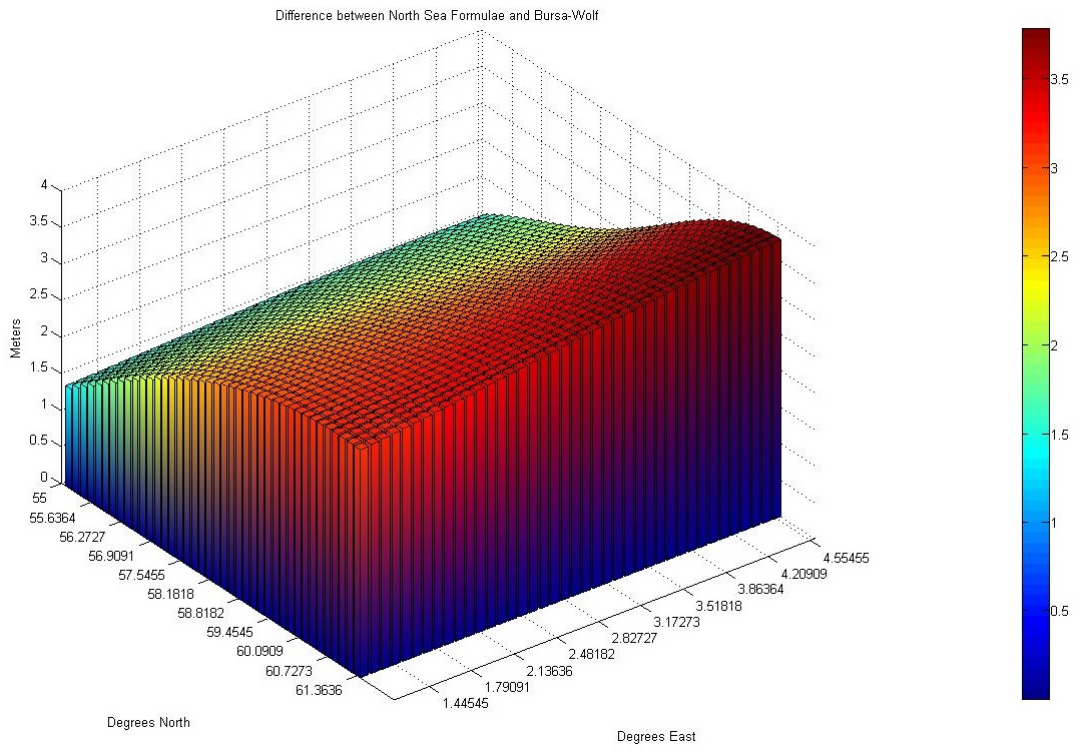


Figure 5.4: The computational difference between Bursa-Wolf and The North Sea Formulae for a latitude interval of 55-62, offshore Norway.

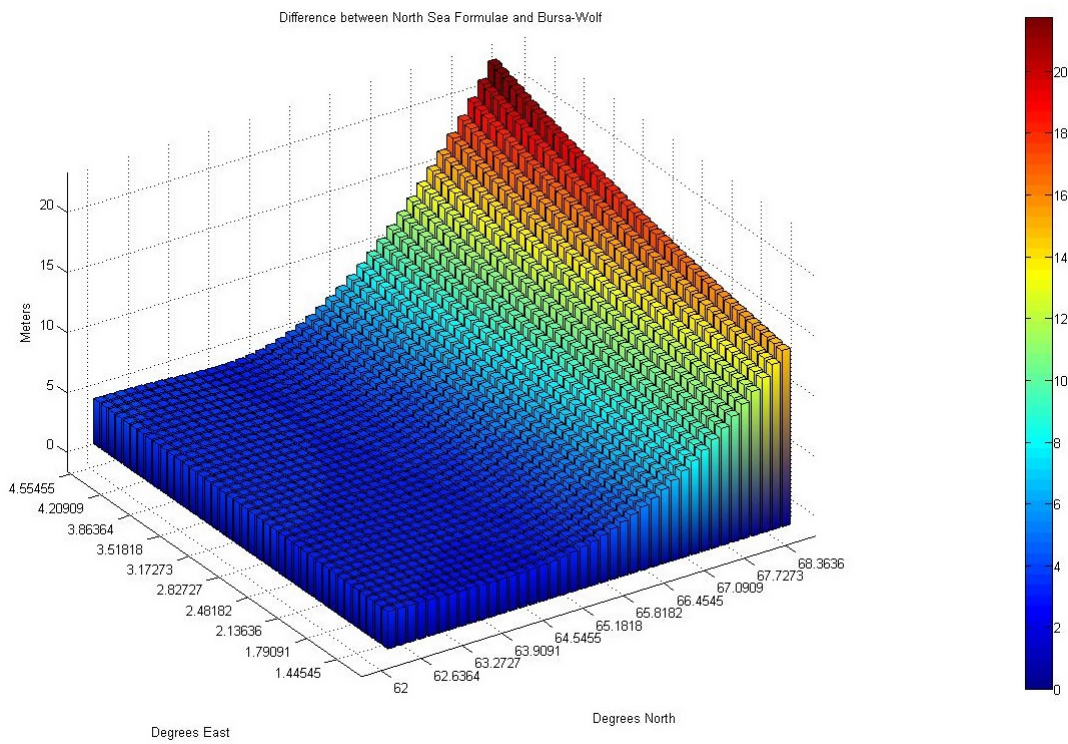


Figure 5.5: The computational difference between Bursa-Wolf and The North Sea Formulae for a latitude interval of 62-69, offshore Norway.

Figure 5.2 shows the lower extent where the Bursa-Wolf transformation becomes very unreliable. A large increase in computation difference is seen south of 51°N. The trend also builds as it progresses towards the UK boundary.

The next image, Figure 5.3, shows the area in close proximity to the 62°N boundary. The difference in computation is 4 meters at the boundary, and increases dramatically when progressing further north. Unlike the southern extent, the deviation trend builds towards mainland Norway.

Figure 5.4 is a close-up of the previous area. The close-up shows the 62°N boundary and the area south. The figure shows that the deviation decreases along southern progression, and continues to do so until 52°N as, seen in Figure 5.2.

Figure 5.5 shows the northern extent, and how rapidly the computational difference increases. The difference exceeds 5 meters at approximately 65.5°N, after which it follows an exponential trend.

Comparing all of the figures above will yield an area (50°N - 62°N) where the only difference between the methods is in the order of 2-4 m. This means that if the location is positioned within this field, and the method of which the coordinates were obtained is unknown, an uncertainty of the responding deviation can be added to the drillers target if possible. For any position beyond this field, it is vital to know the method used for obtaining the coordinates.

5.3 Accuracy of Thomas-UTM

Karney (Karney 2011) provides algorithms for computing Transverse Mercator positions, and also looks into the accuracy of the various formulae. He shows that the flattening series provide the most accurate result, with errors less than 1 μm . On the other hand, the Thomas-UTM series, derived from the expansion series, yield errors over 1 km in size for Transverse Mercator. These errors propagate with the distance from the central meridian. Since UTM is built of grids, each with its own central meridian, this problem is no longer a concern. Karney states that the errors for applying the Thomas-UTM formulae in a UTM system are less than 1 mm.

5.4 Offset for Verification

Tables 4.1, 4.2, and 4.3, all show offset values in the order of $1 \cdot 10^{-5}$ for the seconds. This is not a computational error, but a round off error as a result of accuracy limits in the test values. The test values, seen in Tables A.6, A.7, and A.8, are given with a precision of $1 \cdot 10^{-4}$ in seconds. This means that any computed value more precise than the reference value, will show as an offset for any higher order.

5.5 Quality Control

Maintaining quality control over the geodetic information is absolutely vital. Any set of coordinates are meaningless without the datum tying them as a position on

the surface of the Earth. If the coordinates have been transformed from a different datum, the source coordinates should always be available, if not, then at least the method of transformation.

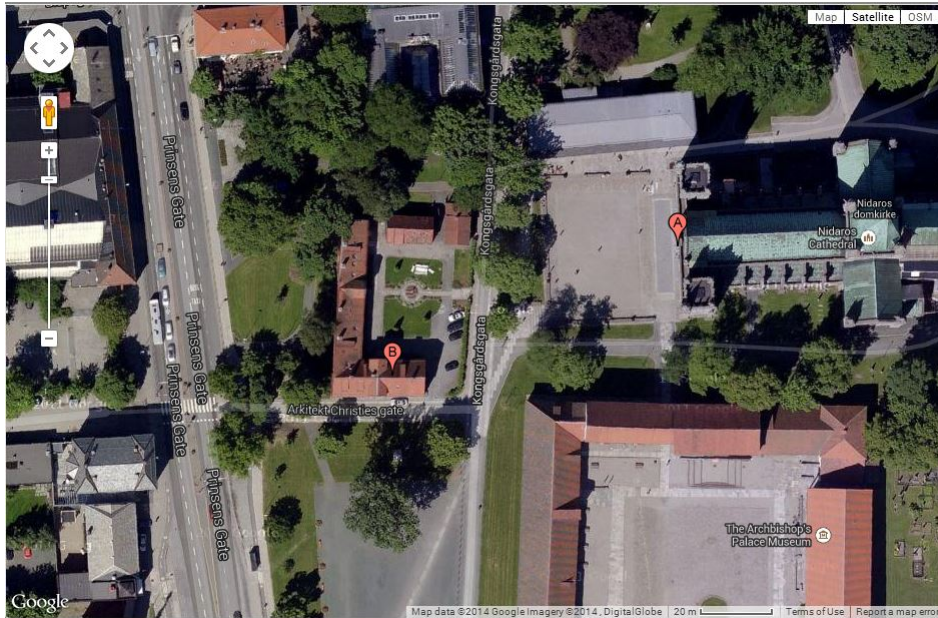


Figure 5.6: A is WGS84 coordinates for Nidarosdomen, B is the same set of coordinates in ED50 (created using Google Maps).

Figure 5.6 shows the WGS84 coordinates of Nidarosdomen plotted as point A. The same coordinates have been plotted for the ED50 system as point B. The distance between these points is 83.57 m. This might be a feasible walking distance, but when it comes to exploration and drilling, this would most likely mean a complete miss. The ED50 coordinates were plotted by transforming them into WGS84 coordinates using the Helmert-7 parameter transformation since Nidarosdomen is positioned north of 62°N . If the coordinates had been transformed using the North Sea Formulae, the position would have been shifted another 4 meters.

Chapter 6

Conclusion

- The mathematical formulae for converting between different coordinate types have been implemented in MATLAB and verified.
- The mathematical formulae for datum transformation, offshore Norway, have been implemented in MATLAB and verified.
- There is an overlapping area between 50°N And 62°N (WGS84) where the deviation between the two methods of datum transformation do not exceed 4 meters. The deviation grows with an exponential trend beyond this extent.
- The iterative function for converting from geocentric to geographic coordinates has been proven accurate even with just a single-step.
- The Thomas-UTM formulae for calculating UTM positions based on geographic coordinates have been implemented and verified. The errors for this set of formulae are less than 1 *mm*.
- Coordinates must always be given with their respective datum, preferably along with their source datum, and method of transformation. This prevents loss of key information, which could lead to grave errors for exact positioning.

CONCLUSION

Chapter 7

Future Work

- Geodesy: Attempt to find a method of transformation which replaces both the North Sea Formulae and the Bursa-Wolf Transformation for North of 62° . It is suggested that best-fit equations are used for computing parameters for a Bursa-Wolf Transformation.
- Geodesy: Expand the position model to cover the world. This model should include selection criteria, either based on a graphical selection, or an automatic selection based on coordinates. The model should be in accordance with current geo-political boundaries and best practice.
- Map Projection: Expand the current UTM model into a MGRS model, displaying the MGRS zone and coordinates as a secondary set of information.
- Software: Develop a graphical user interface for both the current and the future model. It is suggested that the interface is designed to be an intuitive prototype for a final software.

References

- Bowditch, Nathaniel (1995). *The American Practical Navigator*. US. Defense Mapping Agency.
- Burkard, Lt. Col. Richard K. (1959). *Geodesy for the Layman*. Fourth Ed. US. Department of Defense.
- Deakin, R. E. (2004). *The Standard and Abridged Moldensky Coordinate Transformation Formulae*. Department of Mathematical and Geospatial Sciences, RMIT University.
- DMA (1989). *TM8358.2: THE UNIVERSAL GRIDS: Universal Transverse Mercator (UTM) and Universal Polar Stereographic (UPS)*. U.S. Defense Mapping Agency.
- (1990). *TM8358.1: Datums, Ellipsoids, Grids, and Grid Reference Systems*. U.S. Defense Mapping Agency.
- Dutch, Steven (20014). *Convert Between Geographic and UTM Coordinates*. URL: <http://www.uwgb.edu/dutchs/usefuldata/ConvertUTMNoOZ.HTM>.
- EPSG (2001). *Guidance Note Number 10 - Geodetic Transformations Offshore Norway*. European Petroleum Survey Group.
- Excel (2014). *Floating-point arithmetic may give inaccurate results in Excel*. URL: <http://support.microsoft.com/kb/781113>.
- Geodesidivisjon (1990). *The transformation between ED 50 and WGS84 for exploration purposes in the North Sea*. Statens Kartverk.
- Geodesividsjonen (2009). *Koordinatbasert referansesystem, versjon 2.1*. Statens Kartverk.
- Hofmann-Wellenhof, Bernhard, Herbert Lichtenegger, and Elmar Wasle (2008). *GNSS - Global Navigation Satellite Systems*. SpringerWienNewYork.
- Karney, Charles F. F. (2011). “Transverse Mercator with an accuracy of a few nanometers”. In: *J. Geodesy* 8.85, pp. 475–485.

-
- Kleder, Michael. *Geodetic distance on WGS84 earth ellipsoid*. URL: <http://www.mathworks.com/matlabcentral/fileexchange/5379-geodetic-distance-on-wgs84-earth-ellipsoid/content/vdist.m>.
- Knippers, R. A. (2009). *Coordinate Transformations*. URL: <http://kartoweb.itc.nl/geometrics/>.
- Knippers, R. A. and J. Hendrikse (2001). *Coordinate Transformations*.
- Krüger, Prof. Dr. L. (1912). *Konforme Abbildung des Erdellipsoids in der Ebene*. Druck und Verlag von B.G. Teubner in Leipzig.
- Lee, L.P. (1944). “The Nomenclature and Classification of Map projections”. In: *Empire Survey Review* 7.51, pp. 190–200.
- MATLAB (2014a). *Convert degrees-minutes-seconds to degrees*. URL: <http://www.mathworks.se/help/map/ref/dms2degrees.html>.
- (2014b). *Variable-precision accuracy*. URL: <http://www.mathworks.se/help/symbolic/digits.html>.
- NGIA, National Geospatial-Intelligence Agency (2014). *Reference Ellipsoid Parameters*. URL: http://geoengine.nga.mil/geospatial/SW_TOOLS/NIMAMUSE/webinter/geotrans2/help/elliptab.htm.
- NIMA (2000). *Department of Defense: World Geodetic System 1984*. Third Ed. US. National Imagery and Mapping Agency.
- OGP (2001). *Survey & Positioning Guidance note 10*. International Association of Oil & Gas Producers.
- (2013). *Geomatics Guidance Note Number 7, part 2: Coordinate Conversions and Transformations including Formulas*. International Association of Oil & Gas Producers.
- OS (2013). *A guide to coordinate systems in Great Britain*. v. 2.2. Ordnance Survey.
- Snyder, John P. (1987). *Map Projections - A Working Manual*. U.S. Government Printing Office, Washington.
- Snyder, John P. and Philip M. Voxland (1989). *An Album of Map Projections*. U.S. Geological Survey.
- Strahler, Arthur Newell (1963). *Earth Sciences*. Harper & Row.
- Taylor, Charles (2003). *Geographic/UTM Coordinate Converter*. URL: <http://home.hiwaay.net/~taylorc/toolbox/geography/geoutm.html>.

UKOOA (1999). *Guidance Notes on the use of Co-ordinate Systems in Data Management on the UKCS*. U.K. Offshore Operators Association.

Vincenty, T. (1975). “Direct and Inverse Solutions of Geodesics on the Ellipsoid with Application of Nested Equations”. In: *Survey Review* 23.176, pp. 88–93.

Welch, R. and A. Homsey (1997). “Datum Shifts for UTM Coordinates”. In: *PE & RS* 63.4, pp. 371–375.



Appendix A

ID	Semi-Major	Semi-Minor	Flattening
Code	Axis a	Axis b	f
1	6377563.396	6356256.909	1/299.3249646
2	6378160	6356774.719	1/298.25
3	6377397.155	6356078.963	1/299.1528128
4	6377483.865	6356165.383	1/299.1528128
5	6378206.4	6356583.8	1/294.9786982
6	6378249.145	6356514.87	1/293.465
7	6377298.556	6356097.55	1/300.8017
8	6377276.345	6356075.413	1/300.8017
9	6377301.243	6356100.228	1/300.8017
10	6377309.613	6356109.571	1/300.8017
11	6377304.063	6356103.039	1/300.8017
12	6377295.664	6356094.668	1/300.8017
13	6378137	6356752.314	1/298.257222101
14	6378200	6356818.17	1/298.3
15	6378270	6356794.343	1/297
16	6378160	6356774.504	1/298.247
17	6378388	6356911.946	1/297
18	6378245	6356863.019	1/298.3
19	6377340.189	6356034.448	1/299.3249646
20	6378155	6356773.32	1/298.3
21	6378160	6356774.719	1/298.25
22	6378135	6356750.52	1/298.26
23	6378137	6356752.314	1/298.257223563

Table A.1: A detailed list of reference ellipsoid parameters (NGIA 2014)

Reference Ellipsoid	ID
	Code
Airy (1930)	1
Australian National	2
Bessel 1841	
Ethiopia, Indonesia, Japan, Korea	3
Namibia	4
Clarke 1866	5
Clarke 1880	6
Everest	
Brunei & E. Malasia (Sabah & Sarawak)	7
India 1830	8
India 1956*	9
Pakistan*	10
W. Malasia and Singapore 1948	11
W. Malasia 1969*	12
Geodetic Reference System 1980 (GRS 80)	13
Helmert 1906	14
Hough 1960	15
Indonesian 1974	16
International 1924	17
Krassovsky 1940	18
Modified Airy	19
Modified Fischer 1960 (South Asia)	20
South American 1969	21
World Geodetic System 1972 (WGS 72)	22
World Geodetic System 1984 (WGS 84)	23

Table A.2: The relation between ID and Reference Ellipsoid

Constants	From ED50 to ED87	
	Latitude	Longitude
A ₀	-.556098E-05	.148944E-04
A ₁	-.155391E-05	.268191E-05
A ₂	-.402620E-06	.245290E-05
A ₃	-.509693E-06	.294400E-06
A ₄	-.819775E-06	.152260E-05
A ₅	-.247592E-06	.910592E-06
A ₆	.136682E-06	-.368241E-06
A ₇	.186198E-06	-.851732E-06
A ₈	.123350E-06	-.566713E-06
A ₉	.568797E-07	-.185188E-06
A ₁₀	-.232217E-08	.284312E-07
A ₁₁	-.769931E-08	.684853E-07
A ₁₂	-.786953E-08	.500828E-07
A ₁₃	-.612216E-08	.415937E-07
A ₁₄	-.401382E-08	.762236E-08

Table A.3: Polynomial Constants for the transformation between ED50 and ED87 (Geodesidivisjonen 1990).

APPENDIX A

CONICAL PROJECTIONS			
CYLINDRIC	CONFORMAL Mercator	AUTHALIC Authalic Cylindric	APHYLACTIC Equidistant Cylindric Rectangular Cylindric Central Cylindric Gall
PSEUDOCYLINDRIC	...	Collignon Sinusoidal Mollweide Authalic Parabolic Prépéit-Foucault Eumorphic	Trapezoidal Apianus Loritz Orthographic Fournier II Arago
CONIC	Conformal conic with one standard parallel Conformal conic with two standard parallels (Lambert)	Authalic conic with one standard parallel Authalic conic with two standard parallels (Albers)	Equidistant conic with one standard parallel Equidistant conic standard parallels (Albers) with two standard parallels Murdoch I, II, III
PSEUDOCONIC	...	Bonne Sinusoidal Werner	...
POLYCONIC	Lagrange Stereographic	Authalic Polyconic	Equidistant Polyconic Rectangular Polyconic Fournier I Nicolosi Van der Grinten
AZIMUTHAL Perspective	Stereographic (Sphere)	...	Orthographic Gnomonic Clarke James La Hire Parent Lowry
Non-Perspective	Stereographic (Spheroid)	Authalic Azimuthal	Equidistant Azimuthal Airy Breusing

Table A.4: Classification of conical map projections (Lee 1944)

NON-CONICAL PROJECTIONS			
RETROAZIMUTHAL	CONFORMAL ...	AUTHALIC ...	APHYLACTIC Equidistant Retroazimuthal
ORTHOAPSIDAL	...	Authalic Orthoapsidal pp.	Orthoapsidal pp.
MISCELLANEOUS	Littrow August Peirce Guyou Adams pp. Laborde pp.	Aitoff	Schmidt Petermann Two-point Azimuthal (Orthodromic) Two-point Equidistant

Table A.5: Classification of non-conical map projections (Lee 1944)

ED87			ED50				
Latitude	Longitude		Latitude		Longitude		
deg.	deg.	deg.	min.	sec.	deg.	min.	sec.
52	2	52	0	0.0133	1	59	59.9699
53	4	53	0	0.0089	3	59	59.9402
54	0	54	0	0.0168	0	0	-0.0464
55	6	55	0	0.0353	5	59	59.8838
56	-2	56	0	0.023	-2	0	0.0504
57	8	57	0	0.0582	7	59	59.9366
58	8	58	0	0.0572	7	59	59.9777
59	4	59	0	0.049	3	59	59.9888
60	2	60	0	0.0404	2	0	0.0011
61	0	61	0	0.0242	0	0	0.004
62	2	61	59	59.9944	2	0	0.078

Table A.6: Test values for the transformation between ED87 and ED50 using the 14-polynomial method (Geodesidivisjon 1990).

ED87			WGS84*SEA				
Latitude	Longitude		Latitude		Longitude		
deg.	deg.	deg.	min.	sec.	deg.	min.	sec.
52	2	51	59	57.0927	1	59	55.14
53	4	52	59	57.2812	3	59	55.1916
54	0	53	59	57.2224	0	0	-5.2508
55	6	54	59	57.5876	5	59	55.1325
56	-2	55	59	57.3584	-2	0	5.6811
57	8	56	59	57.9079	7	59	55.0669
58	8	57	59	58.0334	7	59	54.9297
59	4	58	59	57.9938	3	59	54.3814
60	2	59	59	58.0339	1	59	54.0159
61	0	60	59	58.0722	0	0	-6.3666
62	2	61	59	58.2998	1	59	53.6259

Table A.7: Test values for the transformation between ED87 and WGS84*SEA using the modified Bursa-Wolf (Geodesidivisjon 1990).

ED50						WGS84 North of 62					
Latitude			Longitude			Latitude			Longitude		
deg.	min.	sec.	deg.	min.	sec.	deg.	min.	sec.	deg.	min.	sec.
62	0	0	1	22	22.769	61	59	58.343	1	22	16.425
62	0	0	1	40	0	61	59	58.355	1	39	53.689
62	0	0	2	0	0	61	59	58.369	1	59	53.726
62	0	0	2	20	0	61	59	58.383	2	19	53.764
62	0	0	2	40	0	61	59	58.397	2	39	53.802
62	0	0	3	0	0	61	59	58.411	2	59	53.84
62	0	0	3	20	0	61	59	58.425	3	19	53.879
62	0	0	3	40	0	61	59	58.439	3	39	53.917
62	0	0	4	0	0	61	59	58.453	3	59	53.956
62	0	0	4	20	0	61	59	58.466	4	19	53.995
62	0	0	4	40	0	61	59	58.48	4	39	54.034
62	0	0	4	52	45.24	61	59	58.488	4	52	39.299

Table A.8: Test values for the transformation between ED50 and WGS84 (OGP 2001)

Appendix B

Functions DMS and DD

Degrees, Minutes, Seconds to Decimal Degrees

```
1 function [dec]=coord2dec(d,m,s)
2
3 %Name: coord2dec.m
4 %Last-Updated: 15.05.2014
5 %Formulae:
6 %Notes: Takes into account negative values of either
7 %         degrees, minutes, and seconds. Verified.
8 %Name:
9
10 deg = abs(d);
11 min = abs(m/60);
12 sec = abs(s/3600);
13
14 dec = deg + min + sec;
15
16 if d == 0
17
18     if m < 0
19
20         dec = sign(m)*(min + sec);
21
22     elseif m == 0
23
24         if s < 0
25
26             dec = sign(s)*sec;
27
28         end
29
30     end
31
32 elseif d < 0
33
34     dec = sign(d)*(deg + min + sec);
35
36 end
37
38 return
```

Decimal Degrees to Degrees, Minutes, Seconds

```
1 function [d,m,s]=dec2coord(dec)
2
3 %Name: dec2coord.m
4 %Last-Updated: 15.05.2014
5 %Formulae:
6 %Notes: Takes into account negative values of either
7 %         degrees, minutes, and seconds.
8 if dec > 0
9     d = floor(dec);
10    dm = (dec-d)*60;
11    m = floor(dm);
12    ms = (dm-m)*60;
13    s = ms;
14
15 elseif dec == 0
16    d = 0;
17    m = 0;
18    s = 0;
19
20 elseif dec < 0
21    d = fix(dec);
22
23    if d < 0
24        dm = abs((dec-d)*60);
25        m = floor(dm);
26        ms = (dm-m)*60;
27        s = ms;
28
29    elseif d == 0
30        dm = (dec-d)*60;
31        m = fix(dm);
32
33        if m < 0
34            ms = abs((dm-m)*60);
35            s = ms;
36
37        elseif m == 0
38            ms = (dm-m)*60;
39            s = ms;
40        end
41    end
42 end
43
44 return
```

Script for testing DMS and DD

```
1
2 %Name: test_dec.m
3 %Last-Updated: 15.05.2014
4 %Formulae:
5 %Notes: Test of function for converting
6 %         between decimal and regular coordinates
7 %         Test values from MATLAB DMS function
8
9 clear all
10
11 deg = [30, -82, 0, 0];
12 min = [50, 2, -30, 0];
13 sec = [44.7801200001663, 39.9082499998644,
14        17.1234500000003, 14.8200000000012];
15 dec = [30.8457722555556, -82.0444189583333,
16        -0.504756513888889, 0.004116666666667];
17
18 for i = 1:length(deg)
19     [dec2(i)] = coord2dec(deg(i), min(i), sec(i));
20     [deg2(i), min2(i), sec2(i)] = dec2coord(dec(i));
21     off_deg(i) = deg(i) - deg2(i);
22     off_min(i) = min(i) - min2(i);
23     off_sec(i) = sec(i) - sec2(i);
24     off_dec(i) = dec(i) - dec2(i);
25
26 end
```

Functions for Cartesian and Geographic Coordinates

Cartesian to Geographic

```
1 function [lat , long ,H]=cart2geo (X,Y,Z, a, b)
2
3 %Name:   cart2geo.m
4 %Last-Updated: 10.04.2014
5 %Formulae: OS / Master 1.5-1.9
6 %Notes: The calculation of lat is an iterative process,
7 %       where the number after > dictates the accuracy.
8
9 p = sqrt (X.^2+Y.^2);
10 ecc2 = (a.^2-b.^2)/a.^2;
11 lat0 = atand (Z/(p*(1-ecc2)));
12 N0 = a/(sqrt (1-ecc2*(sind (lat0).^2)));
13 H = (p/cosd (lat0))-N0;
14 lat = atand ((Z+ecc2*N0*(sind (lat0)))/p);
15
16 while abs (lat0-lat)>1e-20
17     lat0 = lat;
18     N0 = a/(sqrt (1-ecc2*(sind (lat0).^2)));
19     H = p/cosd (lat0)-N0;
20     lat = atand ((Z+ecc2*N0*(sind (lat0)))/p);
21 end
22
23 long = atand (Y/X);
24 return
```

Geographic to Cartesian

```
1 function [X,Y,Z]=geo2cart(lat, long, H, a, b)
2
3 %Name: geo2cart.m
4 %Last-Updated: 08.04.2014
5 %Formulae: GNSS / Master 1.3-1.4
6 %Notes: lat/long in decimal degrees
7
8 N = a.^2/(sqrt((a.^2)*(cosd(lat).^2)+(b.^2)*(sind(lat).^2))
9 );
9 X = (N+H)*cosd(lat)*cosd(long);
10 Y = (N+H)*cosd(lat)*sind(long);
11 Z = ((b.^2/a.^2)*N+H)*sind(lat);
12
13 return
```

Script for testing Cartesian to Geographic

```
1 %Name: test_cart2geo.m
2 %Last-Updated: 22.05.2014
3 %Formulae:
4 %Notes: The calculation of lat is an iterative process,
5 %       where the number after > dictates the accuracy.
6
7 clear all
8
9 lat1 = 33.782341679;
10 long1 = 64.56450923;
11 H1 = 203.2343298;
12
13
14 %Get ellipsoid parameters for ED50
15 [a,b,f]=ellipsoid(17);
16
17 [X,Y,Z]=geo2cart(lat1 ,long1 ,H1,a,b);
18
19 acc(1) = 1e-25;
20 ecc2 = (a.^2-b.^2)/a.^2;
21 p = sqrt(X.^2+Y.^2);
22
23 for i = 1:700
24
25     n(i) = 1;
26     acc(i+1) = acc(i)*1.1 ;
27     lat0(i) = atand(Z/(p*(1-ecc2)));
28     N0(i) = a/(sqrt(1-ecc2*(sind(lat0(i)).^2)));
29     H(i) = (p/cosd(lat0(i))-N0(i));
30     lat(i) = atand((Z+ecc2*N0(i)*(sind(lat0(i))))/p);
31     while abs(lat0(i)-lat(i))>acc(i)
32         lat0(i) = lat(i);
33         N0(i) = a/(sqrt(1-ecc2*(sind(lat0(i)).^2)));
34         H(i) = p/cosd(lat0(i))-N0(i);
35         lat(i) = atand((Z+ecc2*N0(i)*(sind(lat0(i))))/p);
36         n(i) = n(i)+1;
37     end
38     long(i) = atand(Y/X);
39     dist(i) = vdist(lat1 ,long1 ,lat(i) ,long(i));
40 end
```

Function for collecting Ellipsoid Parameters

```
1 function [a,b,f]=ellipsoid(ID)
2
3 %Name: ellipsoid.m
4 %Last-Updated: 03.04.2014
5 %Formulae:
6 %Notes: All values from NGA
7 % All measurements are in meters
8 % * Through adoption of a new yard to meter conversion-
9 % factor in the referenced country.
10
11 if ID==1
12
13     %Airy (1930)
14     a=6377563.396;
15     b=6356256.9090;
16     f=1/299.3249646;
17
18 elseif ID==2
19
20     %Australian National
21     a=6378160;
22     b=6356774.7190;
23     f=1/298.25
24
25 elseif ID==3
26
27     %Bessel 1841 – Ethiopia , Indonesia , Japan , Korea
28     a=6377397.155;
29     b=6356078.9630;
30     f=1/299.1528128;
31
32 elseif ID==4
33
34     %Bessel 1841 – Namibia
35     a=6377483.865;
36     b=6356165.383;
37     f=1/299.1528128;
38
39 elseif ID==5
40
41     %Clarke 1866)
42     a=6378206.4;
43     b=6356583.800;
44     f=1/294.9786982;
```

```
45
46 elseif ID==6
47
48     %Clarke 1880
49     a=6378249.145;
50     b=6356514.870;
51     f=1/293.465;
52
53 elseif ID==7
54
55     %Everest – Brunei & E. Malasia (Sabah & Sarawak)
56     a=6377298.556;
57     b=6356097.550;
58     f=1/300.8017;
59
60 elseif ID==8
61
62     %Everest – India 1830
63     a=6377276.345;
64     b=6356075.413;
65     f=1/300.8017;
66
67 elseif ID==9
68
69     %Everest – India 1956*
70     a=6377301.243;
71     b=6356100.228;
72     f=1/300.8017;
73
74 elseif ID==10
75
76     %Everest – Pakistan*
77     a=6377309.613;
78     b=6356109.571;
79     f=1/300.8017;
80
81 elseif ID==11
82
83     %Everest – W. Malasia and Singapore 1948
84     a=6377304.063;
85     b=6356103.039;
86     f=1/300.8017;
87
88 elseif ID==12
89
90     %Everest – W. Malasia 1969*
```



```
91         a=6377295.664;
92         b=6356094.668;
93         f=1/300.8017;
94
95     elseif ID==13
96
97         %Geodetic Reference System 1980 (GRS 80)
98         a=6378137;
99         b=6356752.3141;
100        f=298.257222101;
101
102     elseif ID==14
103
104        %Helmert 1906
105        a=6378200;
106        b=6356818.170;
107        f=1/298.3;
108
109     elseif ID==15
110
111        %Hough 1960
112        a=6378270;
113        b=6356794.343;
114        f=1/297;
115
116     elseif ID==16
117
118        %Indonesian 1974
119        a=6378160;
120        b=6356774.504;
121        f=1/298.247;
122
123     elseif ID==17
124
125        %International 1924 (ED50)
126        a=6378388;
127        b=6356911.946;
128        f=1/297;
129
130     elseif ID==18
131
132        %Krassovsky 1940
133        a=6378245;
134        b=6356863.019;
135        f=1/298.3;
136
```

```
137 elseif ID==19
138
139     %Modified Airy
140     a=6377340.189;
141     b=6356034.4480;
142     f=1/299.3249646;
143
144 elseif ID==20
145
146     %Modified Fischer 1960 (South Asia)
147     a=6378155;
148     b=6356773.320;
149     f=1/298.3;
150
151 elseif ID==21
152
153     %South American 1969
154     a=6378160;
155     b=6356774.719;
156     f=1/298.25;
157
158 elseif ID==22
159
160     %World Geodetic System 1972 (WGS 72)
161     a=6378135;
162     b=6356750.520;
163     f=1/298.26;
164
165 elseif ID==23
166
167     %World Geodetic System 1984 (WGS 84)
168     a=6378137;
169     b=6356752.3142;
170     f=1/298.257223563;
171
172 else
173     error('The ID code is invalid, see reference table')
174 end
175
176 return
```

Reversible Polynomial in North Sea Formulae

Function for Reversibel Polynomial in the North Sea Formulae

```

1 function [D_lat ,D_long]=northsea1(lat ,long ,A)
2
3 %Name: northsea1.m
4 %Last-Updated: 17.04.2014
5 %Formulae: Kartverk 90 / 2.54 Master's Thesis
6 %Notes: Output is change in latitude and longitude
7 %           All parameters are taken from Kartverk 90.
8 %           Transformation of ED50 to ED87 when A = 1
9 %           Transformation from ED87 to ED50 when A =
           -1
10
11
12 %Latitude polynomials
13 A_lat=A.*[-0.556098*10.^-5 -0.155391*10.^-5
           -0.402620*10.^-6 -0.509693*10.^-6 ...
14 -0.819775*10.^-6 -0.247592*10.^-6 0.136682*10.^-6
           0.186198*10.^-6 0.123350*10.^-6 ...
15 0.568797*10.^-7 -0.232217*10.^-8 -0.769931*10.^-8
           -0.786953*10.^-8 -0.612216*10.^-8 ...
16 -0.401382*10.^-8];
17
18 %Longitude polynomials
19 A_long=A.*[0.148944*10.^-4 0.268191*10.^-5 0.245290*10.^-5
           0.294400*10.^-6 ...
20 0.152260*10.^-5 0.910592*10.^-6 -0.368241*10.^-6
           -0.851732*10.^-6 ...
21 -0.566713*10.^-6 -0.185188*10.^-6 0.284312*10.^-7
           0.684853*10.^-7 ...
22 0.500828*10.^-7 0.415937*10.^-7 0.762236*10.^-8];
23
24 %Origin: 55N, 0E (Greenwich)
25 U=lat -55;
26 V=long;
27
28 %Change in latitude in degrees
29 D_lat=A_lat(1)+A_lat(2)*U+A_lat(3)*V+A_lat(4)*U.^2 ...
30 +A_lat(5)*U*V+A_lat(6)*V.^2+A_lat(7)*U.^3 ...
31 +A_lat(8)*U.^2*V+A_lat(9)*U*V.^2+A_lat(10)*V.^3 ...
32 +A_lat(11)*U.^4+A_lat(12)*U.^3*V+A_lat(13)*U.^2*V.^2 ...
33 +A_lat(14)*U*V.^3+A_lat(15)*V.^4;
34
35 %Change in longitude in degrees

```

```
36 D_long=A_long(1)+A_long(2)*U+A_long(3)*V+A_long(4)*U.^2 ...
37 +A_long(5)*U*V+A_long(6)*V.^2+A_long(7)*U.^3 ...
38 +A_long(8)*U.^2*V+A_long(9)*U*V.^2+A_long(10)*V.^3 ...
39 +A_long(11)*U.^4+A_long(12)*U.^3*V+A_long(13)*U.^2*V.^2 ...
40 +A_long(14)*U*V.^3+A_long(15)*V.^4;
41
42 return
```

Script for testing the Reversibel Polynomial

```

1 %Name: test_NorthSea1.m
2 %Last-Updated: 16.05.2014
3 %Formulae: 14 polynomial SKV
4 %Notes: All values from Statens Kartverk
5 %           Purpose is to test the 14-polynomial
6 %           function from ED87 to ED50
7
8 clear all
9
10 ED87_lat = [52 53 54 55 56 57 58 59 60 61 62];
11 ED87_long = [2 4 0 6 -2 8 8 4 2 0 2];
12
13 ED50_lat_deg = [52 53 54 55 56 57 58 59 60 61 61];
14 ED50_lat_min = [0 0 0 0 0 0 0 0 0 0 59];
15 ED50_lat_sec = [0.0133 0.0089 0.0168 0.0353 0.023 0.0582
16                 0.0572 0.049 0.0404 0.0242 59.9944];
17
18 ED50_long_deg = [1 3 0 5 -2 7 7 3 2 0 2];
19 ED50_long_min = [59 59 0 59 0 59 59 59 0 0 0];
20 ED50_long_sec = [59.9699 59.9402 -0.0464 59.8838 0.0504
21                 59.9366 59.9777 59.9888 0.011 0.004 0.078];
22
23 %Find the amount of elements
24 N=length(ED87_long);
25
26 for i = 1:N;
27     %Step 1: Calculate change in lat/long to ED87
28     [D_lat(i),D_long(i)]=northsea1(ED87_lat(i),
29     ED87_long(i),-1);
30
31     %Add changes to initial values
32     ED50_lat2(i)=D_lat(i)+ED87_lat(i);
33     ED50_long2(i)=D_long(i)+ED87_long(i);
34
35     %Convert ED50 solution to DD
36     [ED50_lat(i)]=coord2dec(ED50_lat_deg(i),
37     ED50_lat_min(i),ED50_lat_sec(i));
38     [ED50_long(i)]=coord2dec(ED50_long_deg(i),
39     ED50_long_min(i),ED50_long_sec(i));
40
41     %Calculate difference
42     off_lat(i) = ED50_lat(i) - ED50_lat2(i);
43     off_long(i) = ED50_long(i) - ED50_long2(i);

```

```
40     [off_lat_deg(i), off_lat_min(i), off_lat_sec(i)] =  
        dec2coord(off_lat(i));  
41     [off_long_deg(i), off_long_min(i), off_long_sec(i)] =  
        dec2coord(off_long(i));  
42     dist(i) = vdist(off_lat(i), off_long(i), 0, 0);  
43  
44     end
```

Bursa-Wolf Transformation

Function for the Bursa-Wolf Transformation

```
1 function [Xt, Yt, Zt]=helmert7(Xs, Ys, Zs, tX, tY, tZ, rX, rY, rZ, dS)
2
3 %Name: helmert7.m
4 %Last-Updated: 11.04.2014
5 %Formulae: OGP-7.2 / 1.37+1.38 Master's Thesis
6 %Notes: Input and output in Cartesian coordinates
7
8 M=(1+dS.*10.^-6);
9 t=[tX; tY; tZ];
10 S=[Xs; Ys; Zs];
11 H=[1 -rZ rY; rZ 1 -rX; -rY rX 1];
12 T=M*H*S+t;
13 Xt=T(1,1);
14 Yt=T(2,1);
15 Zt=T(3,1);
16
17 return
```

Script for testing the Bursa-Wolf Transformation

```
1 %Name: test_NorthSea2.m
2 %Last-Updated: 16.05.2014
3 %Formulae: Helmert-7
4 %Notes: All values from Statens Kartverk
5 %           Purpose is to test the 7-parameter
6 %           function from ED87 to WGS84SEA
7
8 clear all
9
10 ED87_lat = [52 53 54 55 56 57 58 59 60 61 62];
11 ED87_long = [2 4 0 6 -2 8 8 4 2 0 2];
12
13 WGS84SEA_lat_deg = [51 52 53 54 55 56 57 58 59 60 61];
14 WGS84SEA_lat_min = [59 59 59 59 59 59 59 59 59 59 59];
15 WGS84SEA_lat_sec = [57.0927 57.2812 57.2224 57.5876 57.3584
16                     57.9079 58.0334 57.9938 58.0339 58.0722 58.2998];
17
18 WGS84SEA_long_deg = [1 3 0 5 -2 7 7 3 1 0 1];
19 WGS84SEA_long_min = [59 59 0 59 0 59 59 59 59 0 59];
20 WGS84SEA_long_sec = [55.14 55.1916 -5.2508 55.1325 5.6811
21                     55.0669 54.9297 54.3814 54.0159 -6.3666 53.6259];
22
23 %Find the amount of elements
24 N=length(ED87_long);
25
26 %Parameters for ED87 to WGS84SEA
27 tX=-82.981;
28 tY=-99.719;
29 tZ=-110.709;
30 rX=-0.5076E-6;
31 rY=+0.1503E-6;
32 rZ=+0.3898E-6;
33 D=-0.3143;
34
35 %Assume MSL = 0, H = 0
36 H = 0;
37
38 %Get ellipsoid parameters for ED87
39 [a_ED87,b_ED87,f_ED87]=ellipsoid(17);
40
41 %Get ellipsoid parameters for WGS84 (23)
42 [a_WGS84,b_WGS84,f_WGS84]=ellipsoid(23);
43
44 for i = 1:N;
```



```

43
44 %Convert to Cartesian
45 [X_ED87(i),Y_ED87(i),Z_ED87(i)]=geo2cart(ED87_lat(i)
    ),ED87_long(i),H,a_ED87,b_ED87);
46
47 %Transform ED87 to WGS84SEA
48 [X_WGS84SEA(i),Y_WGS84SEA(i),Z_WGS84SEA(i)]=
    helmert7(X_ED87(i),Y_ED87(i),Z_ED87(i),tX,tY,tZ,
    rX,rY,rZ,D);
49
50 %Convert back to Geographic with WGS84 ellipsoid
51 [WGS84SEA_lat2(i),WGS84SEA_long2(i),WGSSEA_H2(i)]=
    cart2geo(X_WGS84SEA(i),Y_WGS84SEA(i),Z_WGS84SEA(
    i),a_WGS84,b_WGS84);
52
53 %Convert WGS84SEA solution to DD
54 [WGS84SEA_lat(i)]=coord2dec(WGS84SEA_lat_deg(i),
    WGS84SEA_lat_min(i),WGS84SEA_lat_sec(i));
55 [WGS84SEA_long(i)]=coord2dec(WGS84SEA_long_deg(i),
    WGS84SEA_long_min(i),WGS84SEA_long_sec(i));
56
57 %Calculate difference
58 off_lat(i) = abs(WGS84SEA_lat(i) - WGS84SEA_lat2(i)
    );
59 off_long(i) = abs(WGS84SEA_long(i) - WGS84SEA_long2
    (i));
60 [off_lat_deg(i),off_lat_min(i),off_lat_sec(i)] =
    dec2coord(off_lat(i));
61 [off_long_deg(i),off_long_min(i),off_long_sec(i)] =
    dec2coord(off_long(i));
62 dist(i) = vdist(off_lat(i),off_long(i),0,0);
63
64 end

```

Scripts for testing complete transformations

Script for transformations North of 62 degrees

```
1 %Name: test_Northof62.m
2 %Last-Updated: 13.04.2014
3 %Formulae: North
4 %Notes: All values from Statens Kartverk
5 %           Script made to test functions/formulae
6 %           vs. test values given by EPSG10 Annex 1
7 %           Tests tfm ED50 to WGS84 along 62 deg N
8 %           Name "test_Northof62"
9
10 clear all
11
12 %Initial latitude values in ED50
13 lat_deg_ED50=[62 62 62 62 62 62 62 62 62 62 62 62];
14 lat_min_ED50=[0 0 0 0 0 0 0 0 0 0 0 0];
15 lat_sec_ED50=[0 0 0 0 0 0 0 0 0 0 0 0];
16
17 %Initial longitude values in ED50
18 long_deg_ED50=[1 1 2 2 2 3 3 3 4 4 4 4];
19 long_min_ED50=[22 40 0 20 40 0 20 40 0 20 40 52];
20 long_sec_ED50=[22.769 0 0 0 0 0 0 0 0 0 0 45.24];
21
22 %End latitude values in WGS84
23 lat_deg_WGS84_SKV=[61 61 61 61 61 61 61 61 61 61 61 61 ];
24 lat_min_WGS84_SKV=[59 59 59 59 59 59 59 59 59 59 59 59 ];
25 lat_sec_WGS84_SKV=[58.343 58.355 58.369 58.383 58.397
26     58.411 58.425 58.439 58.453 58.466 58.48 58.488 ];
27
28 %End longitude values in WGS84
29 long_deg_WGS84_SKV=[1 1 1 2 2 2 3 3 3 4 4 4];
30 long_min_WGS84_SKV=[22 39 59 19 39 59 19 39 59 19 39 52 ];
31 long_sec_WGS84_SKV=[16.425 53.689 53.726 53.764 53.802
32     53.84 53.879 53.917 53.956 53.995 54.034 39.299 ];
33
34 %Parameters for ED50 to WGS84 North of 62deg
35 tX=-116.641;
36 tY=-56.931;
37 tZ=-110.559;
38 rX=+4.327*10.^-6;
39 rY=+4.464*10.^-6;
40 rZ=-4.444*10.^-6;
41 dS=-3.520;
```

```

41 %Find the amount of elements
42 N=length(lat_deg_ED50);
43
44 %Assume MSL = 0, H = 0
45 H = 0;
46
47 %Get ellipsoid parameters for ED50
48 [a_ED50,b_ED50,f_ED50]=ellipsoid(17);
49
50 %Get ellipsoid parameters for WGS84 (23)
51 [a_WGS84,b_WGS84,f_WGS84]=ellipsoid(23);
52
53 %Run the transformation for every element
54 for i=1:N;
55
56     %Convert to decimal
57     [lat_ED50(i)]=coord2dec(lat_deg_ED50(i),
58         lat_min_ED50(i),lat_sec_ED50(i));
59     [long_ED50(i)]=coord2dec(long_deg_ED50(i),
60         long_min_ED50(i),long_sec_ED50(i));
61
62     %Convert to Cartesian with ED50 ellipsoid
63     [X_ED50(i),Y_ED50(i),Z_ED50(i)]=geo2cart(lat_ED50(i)
64         ),long_ED50(i),H,a_ED50,b_ED50);
65
66     %Use Helmert 7 North of 62N
67     [X_WGS84(i),Y_WGS84(i),Z_WGS84(i)]=helmert7(X_ED50(
68         i),Y_ED50(i),Z_ED50(i),tX,tY,tZ,rX,rY,rZ,dS);
69
70     %Convert back to Geographic with WGS84 ellipsoid
71     [lat_WGS84(i),long_WGS84(i),H_WGS84(i)]=cart2geo(
72         X_WGS84(i),Y_WGS84(i),Z_WGS84(i),a_WGS84,b_WGS84
73         );
74
75 end
76
77 %Calculate the error/off-set
78 for i=1:N;
79
80     %Convert WGS84_SKV to decimal
81     [lat_WGS84_SKV(i)]=coord2dec(lat_deg_WGS84_SKV(i),
82         lat_min_WGS84_SKV(i),lat_sec_WGS84_SKV(i));
83     [long_WGS84_SKV(i)]=coord2dec(long_deg_WGS84_SKV(i)
84         ),long_min_WGS84_SKV(i),long_sec_WGS84_SKV(i));
85
86     %Calculate off-set

```

```
79     offset_lat(i)=lat_WGS84(i)-lat_WGS84_SKV(i);
80     offset_long(i)=long_WGS84(i)-long_WGS84_SKV(i);
81     [off_deg_lat(i),off_min_lat(i),off_sec_lat(i)] =
        dec2coord(offset_lat(i));
82     [off_deg_long(i),off_min_long(i),off_sec_long(i)] =
        dec2coord(offset_long(i));
83
84     dist(i) = vdist(abs(offset_lat(i)),abs(offset_long(
85     i)),0,0);
85 end
```

Script for computing the deviation between the North Sea Formula and the Bursa-Wolf Transformation

```

1 %Name: test_62degext.m
2 %Last-Updated: 14.05.2014
3 %Formulae:
4 %Notes: A script made to check the difference between
5 %       the North Sea Formulae and Helmert 7 at
6 %       the 62 deg N latitude band when
7 %       transforming
8 %       from ED50 to WGS84
9
10
11 clear all
12
13
14 %Amount of steps
15 n = 100;
16
17 %Start - UK/Norway boundary (roughly)
18 start = 1.1; %degrees East
19
20 %Stop - Mainland Norway (roughly)
21 stop = 4.9; %degrees East
22
23 %Length of step
24 dn = (stop-start)/n;
25
26
27 %Setting lat and long values for each step
28 long = zeros(n:1);
29 H = zeros(n:1);
30
31 lat = [59, 60, 61, 62, 63, 64, 65, 66, 67];
32 s = length(lat);
33 long(1) = start;
34
35 for i=2:n;
36     long(i) = long(i-1)+dn;
37 end
38
39 %Get ellipsoid parameters for ED50
40 [a_ED50,b_ED50,f_ED50]=ellipsoid(17);
41
42 %Get ellipsoid parameters for WGS84 (23)
43 [a_WGS84,b_WGS84,f_WGS84]=ellipsoid(23);
44
45 %Parameters for ED87 to WGS84SEA
46 tX = -82.981;

```

```
43 tY = -99.719;
44 tZ = -110.709;
45 rX = -0.5076E-6;
46 rY = +0.1503E-6;
47 rZ = +0.3898E-6;
48 D = -0.3143;
49
50 %Parameters for ED50 to WGS84 North of 62deg
51 tX2 = -116.641;
52 tY2 = -56.931;
53 tZ2 = -110.559;
54 rX2 = +4.327*10.^-6;
55 rY2 = +4.464*10.^-6;
56 rZ2 = -4.444*10.^-6;
57 dS2 = -3.520;
58
59
60 for r = 1:s;
61
62     for i = 1:n;
63
64         %Assume MSL = 0, H = 0
65         H(i) = 0;
66
67         %Start North Sea Formulae
68         %Step 1: Calculate change in lat/long to ED87
69         [D_lat(r,i),D_long(r,i)]=northsea1(lat(r),long(i)
70             ,1);
71
72         %Add changes to initial values
73         lat_ED87(r,i)=D_lat(r,i)+lat(r);
74         long_ED87(r,i)=D_long(r,i)+long(i);
75
76         %Convert to Cartesian with ED50 ellipsoid (same as
77         ED87)
78         [X_ED87(r,i),Y_ED87(r,i),Z_ED87(r,i)]=geo2cart(
79             lat_ED87(r,i),long_ED87(r,i),H(i),a_ED50,b_ED50)
80             ;
81
82         %Step 2: Transform ED87 to WGS84SEA
83         [X_WGS84SEA(r,i),Y_WGS84SEA(r,i),Z_WGS84SEA(r,i)]=
84             helmert7(X_ED87(r,i),Y_ED87(r,i),Z_ED87(r,i),tX
85                 (1),tY(1),tZ(1),rX(1),rY(1),rZ(1),D(1));
86
87         %Convert back to Geographic with WGS84 ellipsoid
88         [lat_WGS84SEA(r,i),long_WGS84SEA(r,i),H.WGS84SEA(r,
```

```

    i)] = cart2geo(X_WGS84SEA(r, i), Y_WGS84SEA(r, i),
    Z_WGS84SEA(r, i), a_WGS84, b_WGS84);
83 %End of North Sea Formulae
84
85 %Start Helmert 7-parameter
86 %Convert to Cartesian with ED50 ellipsoid
87 [X_ED50(r, i), Y_ED50(r, i), Z_ED50(r, i)] = geo2cart(lat(
    r), long(i), H(i), a_ED50, b_ED50);
88
89 %Use Helmert 7 North of 62N
90 [X_WGS84(r, i), Y_WGS84(r, i), Z_WGS84(r, i)] = helmert7(
    X_ED50(r, i), Y_ED50(r, i), Z_ED50(r, i), tX2(1), tY2
    (1), tZ2(1), rX2(1), rY2(1), rZ2(1), dS2(1));
91
92 %Convert back to Geographic with WGS84 ellipsoid
93 [lat_WGS84(r, i), long_WGS84(r, i), H_WGS84(r, i)] =
    cart2geo(X_WGS84(r, i), Y_WGS84(r, i), Z_WGS84(r, i),
    a_WGS84, b_WGS84);
94 %End Helmert 7-parameter
95
96 %Calculate the difference of each step in meters
97 d_lat(r, i) = lat_WGS84(r, i) - lat_WGS84SEA(r, i);
98 d_long(r, i) = long_WGS84(r, i) - long_WGS84SEA(r, i);
99 d_H(r, i) = H_WGS84(r, i) - H_WGS84SEA(r, i);
100 [deg_lat(r, i), min_lat(r, i), sec_lat(r, i)] =
    dec2coord(d_lat(r, i));
101 [deg_long(r, i), min_long(r, i), sec_long(r, i)] =
    dec2coord(d_long(r, i));
102 dist(r, i) = vdist(lat_WGS84(r, i), long_WGS84(r, i),
    lat_WGS84SEA(r, i), long_WGS84SEA(r, i));
103
104     end
105 end
106
107
108 Y = long;
109 X = lat;
110 Z = dist;
111
112 figure;
113 h = bar3(Z);
114 set(gca(gcf), 'xticklabel', {'1.1', '1.52', '1.94', '2.36',
    '2.78', '3.21', '3.63', '4.05', '4.47', '4.9'}, '
    yticklabel', {'59', '60', '61', '62', '63', '64', '65', '
    66', '67'})
115 colorbar

```

```
116
117 title('Difference between North Sea Formulae and Bursa-Wolf
        ');
118 xlabel('Degrees East');
119 ylabel('Degrees North');
120 zlabel('Meters');
121
122 for k = 1:length(h)
123     zdata = get(h(k), 'ZData');
124     set(h(k), 'CData', zdata, ...
125           'FaceColor', 'interp')
126 end
```

Function for calculating the distance between two Geographical coordinates

```

1 function s = vdist(lat1,lon1,lat2,lon2)
2 %       - NOT MINE - by Michael Kleder, 30 Jun 2004 (
   Updated 01 Sep 2004)
3
4
5
6 %VDIST - compute distance between points on the WGS-84
   ellipsoidal Earth
7 %       to within a few millimeters of accuracy using
   Vincenty's algorithm
8 %
9 % s = vdist(lat1,lon1,lat2,lon2)
10 %
11 % s = distance in meters
12 % lat1 = GEODETIC latitude of first point (degrees)
13 % lon1 = longitude of first point (degrees)
14 % lat2, lon2 = second point (degrees)
15 %
16 % Original algorithm source:
17 % T. Vincenty, "Direct and Inverse Solutions of Geodesics
   on the Ellipsoid
18 % with Application of Nested Equations", Survey Review,
   vol. 23, no. 176,
19 % April 1975, pp 88-93
20 %
21 % Notes: (1) Error correcting code, convergence failure
   traps, antipodal corrections,
22 %         polar error corrections, WGS84 ellipsoid
   parameters, testing, and comments
23 %         written by Michael Kleder, 2004.
24 %         (2) Vincenty describes his original algorithm as
   precise to within
25 %         0.01 millimeters, subject to the ellipsoidal
   model.
26 %         (3) Essentially antipodal points are treated as
   exactly antipodal,
27 %         potentially reducing accuracy by a small
   amount.
28 %         (4) Failures for points exactly at the poles are
   eliminated by
29 %         moving the points by 0.6 millimeters.
30 %         (5) Vincenty's azimuth formulas are not
   implemented in this

```

```
31 %           version , but are available as comments in the
    code .
32 %           (6) The Vincenty procedure was transcribed
    verbatim by Peter Cederholm ,
33 %           August 12, 2003. It was modified and
    translated to English by Michael Kleder .
34 %           Mr. Cederholm's website is http://www.plan.aau
    .dk/~pce/
35 %           (7) Code to test the disagreement between this
    algorithm and the
36 %           Mapping Toolbox spherical earth distance
    function is provided
37 %           as comments in the code. The maximum
    differences are:
38 %           Max absolute difference: 38 kilometers
39 %           Max fractional difference: 0.56 percent
40
41 % Input check:
42 if abs(lat1)>90 | abs(lat2)>90
43     error('Input latitudes must be between -90 and 90
    degrees , inclusive .')
44 end
45 % Supply WGS84 earth ellipsoid axis lengths in meters:
46 a = 6378137; % definitionally
47 b = 6356752.31424518; % computed from WGS84 earth
    flattening coefficient definition
48 % convert inputs in degrees to radians:
49 lat1 = lat1 * 0.0174532925199433;
50 lon1 = lon1 * 0.0174532925199433;
51 lat2 = lat2 * 0.0174532925199433;
52 lon2 = lon2 * 0.0174532925199433;
53 % correct for errors at exact poles by adjusting 0.6
    millimeters:
54 if abs(pi/2-abs(lat1)) < 1e-10;
55     lat1 = sign(lat1)*(pi/2-(1e-10));
56 end
57 if abs(pi/2-abs(lat2)) < 1e-10;
58     lat2 = sign(lat2)*(pi/2-(1e-10));
59 end
60 f = (a-b)/a;
61 U1 = atan((1-f)*tan(lat1));
62 U2 = atan((1-f)*tan(lat2));
63 lon1 = mod(lon1,2*pi);
64 lon2 = mod(lon2,2*pi);
65 L = abs(lon2-lon1);
66 if L > pi
```

```

67     L = 2*pi - L;
68 end
69 lambda = L;
70 lambdaold = 0;
71 itercount = 0;
72 while ~itercount | abs(lambda-lambdaold) > 1e-12 % force
    at least one execution
73     itercount = itercount+1;
74     if itercount > 50
75         warning('Points are essentially antipodal.
76                 Precision may be reduced slightly.');
```

$$\lambda = L + (1-C) * f * \sin(\alpha) * (\sigma + C * \sin(\sigma) * \dots$$

$$(\cos 2\sigma + C * \cos(\sigma) * (-1 + 2 * \cos 2\sigma^2)));$$

```

76         lambda = pi;
77         break
78     end
79     lambdaold = lambda;
80     sinsigma = sqrt((cos(U2)*sin(lambda))^2+(cos(U1)*...
81                 sin(U2)-sin(U1)*cos(U2)*cos(lambda))^2);
82     cossigma = sin(U1)*sin(U2)+cos(U1)*cos(U2)*cos(lambda);
83     sigma = atan2(sinsigma , cossigma);
84     alpha = asin(cos(U1)*cos(U2)*sin(lambda)/sin(sigma));
85     cos2sigmam = cos(sigma)-2*sin(U1)*sin(U2)/cos(alpha)^2;
86     C = f/16*cos(alpha)^2*(4+f*(4-3*cos(alpha)^2));
87     lambda = L+(1-C)*f*sin(alpha)*(sigma+C*sin(sigma)*...
88         (cos2sigmam+C*cos(sigma)*(-1+2*cos2sigmam^2)));
89 % correct for convergence failure in the case of
    essentially antipodal points
90     if lambda > pi
91         warning('Points are essentially antipodal.
92                 Precision may be reduced slightly.');
```

$$u2 = \cos(\alpha)^2 * (a^2 - b^2) / b^2;$$

$$A = 1 + u2 / 16384 * (4096 + u2 * (-768 + u2 * (320 - 175 * u2)));$$

$$B = u2 / 1024 * (256 + u2 * (-128 + u2 * (74 - 47 * u2)));$$

$$\text{deltastigma} = B * \sin(\sigma) * (\cos 2\sigma + B/4 * (\cos(\sigma) * (-1 + 2 * \cos 2\sigma^2) \dots$$

$$- B/6 * \cos 2\sigma * (-3 + 4 * \sin(\sigma)^2) * (-3 + 4 * \cos 2\sigma^2)$$

```

92         lambda = pi;
93         break
94     end
95 end
96 u2 = cos(alpha)^2*(a^2-b^2)/b^2;
97 A = 1+u2/16384*(4096+u2*(-768+u2*(320-175*u2)));
98 B = u2/1024*(256+u2*(-128+u2*(74-47*u2)));
99 deltastigma = B*sin(sigma)*(cos2sigmam+B/4*(cos(sigma)
100     *(-1+2*cos2sigmam^2)...
    -B/6*cos2sigmam*(-3+4*sin(sigma)^2)*(-3+4*cos2sigmam^2)
    ));
101 s = b*A*(sigma-deltastigma);
```

Function for Converting from Geographical to UTM Coordinates using USGS

```
1 function [N,E,Z,hemi]=coord2grid_USGS(latd,longd,a,b,f)
2
3 %Name: coord2grid_USGS.m
4 %Last-Updated: 01.05.2014
5 %Formulae: OGP 7-2, 1.3.5.1 USGS
6 %Notes: Modified TM for UTM
7 %           Hemi 1 means North, Hemi -1 means South
8
9 %Error codes for limits
10 if latd < -180 | latd > 180
11     error('Latitude must be between 180 and -180.')
12 elseif longd < -90 | longd > 90
13     error('Longitude must be between 90 and -90.')
14 elseif longd < -89 | longd > 89
15     error('Longitude values close to 90 will not yield
16         accurate results.')
17
18 %Zone width, degrees
19 W = 6;
20
21 %False Easting, metre
22 FE = 500000;
23
24 %False Northing, metre
25 if latd >= 0;
26     FN = 0;
27     hemi = 1;
28 else
29     FN = 10000000;
30     hemi = -1;
31 end
32
33 %Central Meridian scale factor
34 k0 = 0.9996;
35
36 %Calculate UTM Zone number
37 Z = 1+floor((longd+180)/W);
38
39 %Longitude of natural origin
40 longd0 = (3+W*(Z-1)-180);
41
42 %Latitude of natural origin
```

```

43 latd0 = 0;
44
45 %eccentricity
46 e = sqrt(2*f-f.^2);
47
48 %second eccentricity
49 ecc = sqrt(e.^2/(1-e.^2));
50
51 %Degrees to radians
52 lat = (latd/360)*2*pi;
53 lat0 = (latd0/360)*2*pi;
54 long = (longd/360)*2*pi;
55 long0 = (longd0/360)*2*pi;
56
57 T = (tan(lat).^2);
58 C = (e.^2*(cos(lat).^2))/(1-e.^2);
59 A = (long-long0)*cos(lat);
60 nu = a/(1-e.^2*(sin(lat).^2)).^0.5;
61
62 M = a*((1-(e.^2)/4-(3*e.^4)/64-(5*e.^6)/256)*lat ...
63 -((3*e.^2)/8+(3*e.^4)/32+(45*e.^6)/1024)*sin(2*lat) ...
64 +((15*e.^4)/256+(45*e.^6)/1024)*sin(4*lat) ...
65 -((35*e.^6)/3072)*sin(6*lat));
66
67 M0 = a*((1-(e.^2)/4-(3*e.^4)/64-(5*e.^6)/256)*lat0 ...
68 -((3*e.^2)/8+(3*e.^4)/32+(45*e.^6)/1024)*sin(2*lat0) ...
69 +((15*e.^4)/256+(45*e.^6)/1024)*sin(4*lat0) ...
70 -((35*e.^6)/3072)*sin(6*lat0));
71
72 E = FE+k0*nu*(A+(1-T+C)*((A.^3)/6)+(5-18*T+T.^2+72*C-58*ecc
    .^2)*((A.^5)/120));
73 N = FN+k0*(M-M0+nu*tan(lat)*((A.^2)/2+(5-T+9*C+4*C.^2)*((A
    .^4)/24)+(61-58*T+T.^2+600*C-330*ecc.^2)*((A.^6)/720));
74
75 return

```

Function for converting from UTM to geographical coordinates using USGS

```
1 function [latd , longd]=grid2coord_USGS(N,E,Z, a , b , f , hemi)
2
3 %Name: grid2coord_USGS.m
4 %Last-Updated: 01.05.2014
5 %Formulae: OGP 7-2, 1.3.5.1 USGS
6 %Notes:
7
8 %Error codes for limits
9 if E < 160000 | E > 840000
10     error('Easting may not exceed 840000 or be below
11           160000.')
12 elseif N < 0
13     error('Negative Northing is not allow.')
14 elseif N > 10000000
15     error('Northing may not exceed 10000000.')
16 end
17 %Central Meridian scale factor
18 k0 = 0.9996;
19
20 %eccentricity
21 e = sqrt(2*f-f.^2);
22
23 %Zone width, degrees
24 W = 6;
25
26 %False Northing
27 if hemi == 1
28     FN = 0;
29 elseif hemi == -1
30     FN = 10000000;
31 else
32     error('Hemisphere must be 1 for North, or -1 for South.
33           ')
34 end
35 %second eccentricity
36 ecc = sqrt(e.^2/(1-e.^2));
37
38 %eccentricity squared
39 e2 = e.^2;
40
41 %second eccentricity squared
```

```

42 ecc2 = (e.^2/(1-e.^2));
43
44 %False Easting, metre
45 FE = 500000;
46
47 %Longitude of natural origin
48 longd0 = (3+6*(Z-1)-180);
49
50 %Latitude of natural origin
51 latd0 = 0;
52
53 %Degrees to radians
54 lat0 = (latd0/360)*2*pi;
55 long0 = (longd0/360)*2*pi;
56
57
58 M0 = a*((1-(e.^2)/4-(3*e.^4)/64-(5*e.^6)/256)*lat0 ...
59 -((3*e.^2)/8+(3*e.^4)/32+(45*e.^6)/1024)*sin(2*lat0) ...
60 +((15*e.^4)/256+(45*e.^6)/1024)*sin(4*lat0) ...
61 -((35*e.^6)/3072)*sin(6*lat0));
62
63 M1 = M0+(N-FN)/k0;
64 mu1 = M1/(a*(1-e2/4-e.^4*(3/64)-e.^6*(5/256)));
65 e1 = (1-(1-e2).^0.5)/(1+(1-e2).^0.5);
66
67 lat1 = mu1+(e1*(3/2)+e1.^3*(27/32))*sin(2*mu1) ...
68 +(e1.^2*(21/16)-e1.^4*(55/32))*sin(4*mu1) ...
69 +(e1.^3*(151/96))*sin(6*mu1) ...
70 +(e1.^4*(1097/512))*sin(8*mu1);
71
72 T1 = (tan(lat1).^2);
73 C1 = ecc2*(cos(lat1).^2);
74 nu1 = a/(((1-e2*(sin(lat1).^2)).^0.5);
75 rho1 = (a*(1-e2))/((1-e2*(sin(lat1).^2)).^1.5);
76
77 D = (E-FE)/(nu1*k0);
78
79 lat = lat1 -((nu1*tan(lat1))/rho1)*((D.^2)/2-(5+3*T1+10*C1
80 -4*C1.^2-9*ecc2)*((D.^4)/24) ...
81 +(61+90*T1+298*C1+45*T1.^2-252*ecc2-3*C1.^2)*((D.^6)/720));
82
83 long = long0+(D-(1+2*T1+C1))*((D.^3)/6)+(5-2*C1+28*T1-3*C1
84 .^2+8*ecc2+24*T1.^2)*((D.^5)/120))/cos(lat1);
85
86 return

```


Appendix C

Script

The computations were made using the following script:

```

1  % Worked Example
2
3  clear all
4
5  %Define coordinates
6  latD_rig = 61;
7  latM_rig = 59;
8  latS_rig = 30.45;
9  longD_rig = 2;
10 longM_rig = 30;
11 longS_rig = 20.10;
12 H_rig = 0;
13
14 latD_target = 62;
15 latM_target = 01;
16 latS_target = 04.30;
17 longD_target = 2;
18 longM_target = 30;
19 longS_target = 40;
20 H_target = 2300;
21
22 %Convert to DD
23 [latDD_rig] = coord2dec(latD_rig , latM_rig , latS_rig);
24 [longDD_rig] = coord2dec(longD_rig , longM_rig , longS_rig);
25
26 [latDD_target] = coord2dec(latD_target , latM_target ,
27   latS_target);
28 [longDD_target] = coord2dec(longD_target , longM_target ,
29   longS_target);
30
31 %Calculate geographic offset
32 [lat_off , long_off] = northsea1(latDD_rig , longDD_rig , 1);
33
34 %Transform to ED87
35 latDD_rig_ED87 = latDD_rig + lat_off;
36 longDD_rig_ED87 = longDD_rig + long_off;
37
38 %Get ellipsoid parameters
39 [a_ED87, b_ED87, f_ED87] = ellipsoid(17);
40 [a_WGS84, b_WGS84, f_WGS84] = ellipsoid(23);
41

```

```
40 %Convert to geocentric coordinates
41 [X_rig_ED87, Y_rig_ED87, Z_rig_ED87] = geo2cart(
    latDD_rig_ED87, longDD_rig_ED87, H_rig, a_ED87, b_ED87);
42 [X_target, Y_target, Z_target] = geo2cart(latDD_target,
    longDD_target, H_target, a_ED87, b_ED87);
43
44 %Transform to WGS84
45 [X_rig_WGS84SEA, Y_rig_WGS84SEA, Z_rig_WGS84SEA] = helmert7
    (X_rig_ED87, Y_rig_ED87, Z_rig_ED87, -82.981, -99.719,
    -110.709, -0.5076E-6, 0.1503E-6, 0.3898E-6, -0.3143);
46 [X_target_WGS84, Y_target_WGS84, Z_target_WGS84] = helmert7
    (X_target, Y_target, Z_target, -116.641, -56.931,
    -110.559, 4.327E-6, 4.464E-6, -4.444E-6, -3.520);
47
48 %Convert to geographical coordinates
49 [latDD_rig_WGS84SEA, longDD_rig_WGS84SEA, H_rig_WGS84SEA] =
    cart2geo(X_rig_WGS84SEA, Y_rig_WGS84SEA, Z_rig_WGS84SEA
    , a_WGS84, b_WGS84);
50 [latDD_target_WGS84, longDD_target_WGS84, H_target_WGS84] =
    cart2geo(X_target_WGS84, Y_target_WGS84, Z_target_WGS84
    , a_WGS84, b_WGS84);
51
52 %Convert to UTM coordinates
53 [N_rig, E_rig, Z_rig, hemi_rig] = coord2grid_USGS(
    latDD_rig_WGS84SEA, longDD_rig_WGS84SEA, a_WGS84,
    b_WGS84, f_WGS84);
54 [N_target, E_target, Z_target, hemi_target] =
    coord2grid_USGS(latDD_target_WGS84, longDD_target_WGS84,
    a_WGS84, b_WGS84, f_WGS84);
55
56 %Calculate TVD (H + above MSL, water depth + below MSL)
57 TVD_wellhead = 180 - H_rig_WGS84SEA;
58 TVD_target = H_target_WGS84;
59
60 %Calculate difference
61 DN = N_target - N_rig;
62 DE = E_target - E_rig;
63 DZ = Z_target - Z_rig;
64 DTV = TVD_target - TVD_wellhead;
```