A GEOGRAPHIC SEGMENTATION APPROACH FOR SATELLITE-DERIVED BATHYMETRY

By

Juliane Jussara Affonso

THESIS

Submitted to the University of New Hampshire
in partial fulfillment of
the requirements for the degree of

Master of Science

In

Earth Sciences

December 2022

This thesis has been examined and approved in partial fulfillment of the requirements for the degree of Master of Science in Earth Sciences by:

Thesis Director, Christos Kastrisios, Research Assistant Professor,
University of New Hampshire.

Christopher Parrish, Associate Professor,
Oregon State University.

Brian Calder, Research Professor,
University of New Hampshire.

On September 2nd, 2022

Original approval signatures are on file with the University of New Hampshire Graduate School.

DEDICATION

To my friend and love, Felipe, who, even far away, was always present. I couldn’t have done this without you.

ACKNOWLEDGEMENTS

First, I would like to thank God for giving me the strength, knowledge, and conditions necessary to conclude this research successfully. Mainly during a difficult period during the pandemic crisis.

Thanks to the Brazilian Navy for trusting me in this mission and investing in this enriching program, where I had the opportunity to expand my knowledge in hydrography. I have no words to express my gratitude, especially due to the barriers during the pandemic. I hope I can reciprocate with my work.

I am grateful to CCOM and UNH faculty and members for the opportunity to provide me with tremendous professional growth. I am very proud to be part of the CCOM/UNH family.

I would like to express my sincere thanks to Christos Kastrisios, Christopher Parrish, and Brian Calder for the guidance, inspiration, and insightful comments they provided during the course of the project study. Brian for receiving me at CCOM and his brilliant comments and notes, further enriching this work. My immense thanks to my co-advisor, Chris, for his guidance and support that enriched this research. In particular, I am grateful to my advisor, Christos, who, since the beginning, when borders were closed to coming to the US, he believed in me and gave me the opportunity to take this important step in my life. He has always been friendly and patient with me, and it was a great pleasure to have been his first graduate student at CCOM. It has been a privilege to be under the supervision of such brilliant minds.

My sincere thanks to people from the Directorate of Hydrography and Navigation (DHN) and Navy Hydrography Center (CHM) for their help and guidance during the selection process. In particular, to CF (EN) Ricardo Freire for his advice throughout the ideas of this project, for sharing his valuable knowledge, and for helping with some important decisions.

I would also like to give special thanks to CC Leonardo and CC Ivan on my arrival, sharing the best tips and advice I could have - looking forward to working together.

Thanks to all UNH and Brazilian (online) colleagues who, even during a difficult time in the first year, made my life lighter with their friendship and cordiality, helping me to lighten this challenge experienced. A special thanks to Daniel and Miguel, who offered their expertise unconditionally besides their warm Brazilian/Portuguese friendship and camaraderie.

I am extremely grateful to my family for their endless love, prayers (my mom has been busy God), and sacrifices for educating and trusting in me. Their words and support have helped me with the challenges of living abroad.

Felipe, who never hesitated to my desires and madness, always being by my side despite all adversities. Thank you for your unconditional patience and love. I am blessed to share my life and dreams with you.

“Obrigada”

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACRONYMS

ACOLITE – Atmospheric Correction of Operational Land Imager

ALB – Airborne Lidar bathymetry

AOP - Apparent Optical Properties

APR - Aggregated Path Reef

AUV - Autonomous Underwater Vehicles

BOA - Bottom Of Atmosphere

CATZOC - CATegory of ZOne of Confidence

CHC- Canadian Hydrographic Service

DADR – Depth Areas by Depth Range

DSF - Dark Spectrum Fitting

ENC - Electronic Navigational Chart

ESA - European Space Agency

EXP - Exponential extrapolation

GLDR – Global-Local by Depth Range

IHO - International Hydrographic Organization

IOP - Inherent Optical Properties

L1C - Level-1C

L2A - Level-2A

LPF - Low Pass Filter

LRSG - Low Relief Spur and Groove

LSM- Least Squares Method

MBES – Multibeam Echo Sounde

MLLW - Mean Lower Low Water

MPA- Marine Protected Area

MSI - Multispectral Instrument

NDWI - Normalized Difference Water Index

NGS - National Geodetic Survey

NOAA - National Oceanic and Atmospheric Administration

NPS - National Park Service

RBINS - Royal Belgian Institute of Natural Sciences

RM - Remnant

RMSE - Root Mean Square Error

ROV - Remotely Operated Vehicle

RT - Reef Terrace

RTE - Radiative Transfer Equation

SBES - Single Beam Echo Sounder

SDB - Satellite-Derived Bathymetry

SDC - Satellite-Derived Chart

SHOM - Service Hydrographique et Océanographique de la Marine

THU - Total Horizontal Uncertainties

TOA - Top Of Atmosphere

TVU - Total Vertical Uncertainties

UKHO - United Kingdom Hydrographic Office

US - Surrounded by Sand

USGS- United States Geological Survey

VDatum - Vertical Datum

ABSTRACT

Safety of navigation depends on our knowledge of seabed and its features, and, as such, any improvements in deriving bathymetry for nautical chart updating are of major importance. Satellite-Derived Bathymetry (SDB) is an alternative to traditional surveys using ship and airborne sensors, particularly for mapping remote and shallow areas, due to its reduced cost and the absence of navigational risks in very shallow and unsurveyed areas. However, the accuracy of SDB can be judged as relatively low for nautical charting purposes and, therefore, is mostly used for reconnaissance or/and for filling gaps in remote or very shallow areas. One of the reasons may be that the conventional approaches assume that bottom type and water clarity are constant and negligible within the entire image, and consequently, a single (global) and linear model is performed to retrieve bathymetric information. To address the spatial heterogeneity within a scene and with the aim to increase the accuracy and coverage of estimated depths, this work investigates the segmentation of the scene, both horizontally and vertically, into smaller spatial units, and accounts for water column parameters in the SDB equation. In practice, the main idea of the segmentation is to divide the image scene into small spatial units and then calibrate the model within each segment. The individual models use the same algorithm but varying model parameters from place to place. Also, to account for water column and sea bottom variations, an extended Dierssen model is applied. The performance of the methods is evaluated in two study areas in the Dry Tortugas, Florida, and St. Thomas East and Reserve, U.S. Virgin Islands. Overall, the results indicate that the accuracy of bathymetry may be improved when the area is divided into smaller spatial units, particularly with a vertical (by depth) segmentation of the scene. In detail, compared to the conventional global and linear approach, the accuracy in both study areas is increased by over 40% with segmenting the area and calibrating the water parameters within each spatial unit. Furthermore, as it is demonstrated with the two study areas, besides the improvements in the depth accuracy, the SDB coverage

is increased with the extraction of bathymetry beyond the depth considered as the effective optical depth of the conventional global and linear approach. However, further work is recommended to investigate and verify the accuracy improvement demonstrated by the vertical segmentation and particularly that of the smallest utilized depth range of 1m. since questions are raised about the discontinuity of the models and their quantized depth predictions, and more precisely whether this is due to overfitting rather than an actual improvement in accuracy. Lastly, the results demonstrate that considering the water column and sea bottom heterogeneity for solving the global SDB model increases the accuracy of bathymetry estimates. Nonetheless, when the area is segmented into small spatial units, adding the water column contribution as a parameter to the equation did not produce a significant contribution.

CHAPTER 1 : INTRODUCTION

The nautical chart is one of the most important tools used by mariners for safe navigation, providing detailed information about the coastal and marine environment. Among the variety of elements portrayed on nautical charts, the soundings, shorelines, and depth curves are of utmost importance. In the past, these elements were traditionally drawn by hand by skilled nautical cartographers. Advances in technology and increasing users' needs led to the transformation of the nautical chart from its analog form to a digital vector product, Electronic Navigational Chart (ENCs). The ENC contains not only the information available in paper charts but also additional information that facilitates the use of nautical charts (Kastrisios *et al*., 2022), providing significant advantages over the paper chart (Mavraeidopoulos, *et al*., 2017). The technical design of ENCs in circulation is defined by the IHO standards S-57 (2000) and S-52 (2014). Since 2005, the IHO has been working on developing a new, more versatile standard, the S-100, which, besides incorporating the requirements of S-57 for ENCs, supports items that go beyond the scope of traditional hydrography (Ward, *et al*., 2009). With the increase in cartographic production in recent years, and the expansion of the chart's portfolio with S-100, there has been an increasing need for improving chart production workflow by automating chart compilation routines. The efforts of the nautical charting community have allowed for the automation of various ENC compilation tasks and the transformation of chart compilation from a manual to a semi-automated process (Kastrisios & Calder, 2018).

Besides the needs for automation, there has been an even more pressing need for keeping the charts up-to-date with resurveying shallow coastal waters and surveying areas that have not been previously surveyed, due to factors such as the increasing sizes of ships that operate in tighter spaces (Kastrisios & Ware, 2022) and the global effort to map the world's ocean by 2030 (Seabed 2030 Project) (GEBCO, 2020). The most traditional techniques used to achieve reliable bathymetric data to use and keep up-to-date nautical charts are those collected by echo-sounding systems, such as Single Beam Echo Sounders (SBESs) that transmit and receives a single beam of sound waves, and Multibeam Echo Sounders (MBESs) that hundreds of beams are received simultaneously in a single swath providing complete insonification of the area and better coverage than SBESs. Also, due to some drawbacks and challenges in surveying in very shallow waters presented by MBES and SBES, such as saturation or/and inaccessibility of survey vessels (Su *et al*., 2008), Airborne Lidar Bathymetry (ALB), Remotely Operated Vehicle (ROV's), and Autonomous Underwater Vehicles (AUV's) have emerged for acquiring bottom-depth measurements. Although these technologies can collect high resolution and accurate bathymetry, they also come with limitations. For example, some remote and complex areas with difficult access pose challenges in undertaking hydrographic surveys, due to risk for the crew and loss of materials (Forfinski-Sarkozi *et al*., 2019). Furthermore, in shallow areas, the swath becomes narrow, which limits the seabed coverage and increases the time and effort to survey these areas. Consequently, surveying very shallow waters with boat-based multibeam sensors is costly, often challenging, and, possibly, unsafe. Therefore, conventional surveying and mapping technologies, although are very accurate information, they represent an operational and logistical challenge, due to time-limitations, the risk of surveying in nearshore environments, and mainly the high cost of operations.

The above limitations and difficulties have led researchers to search for alternative methods to collect bathymetric information, including solutions provided by space technology. In this context, Satellite Derived Bathymetry (SDB), a technique that utilizes optical satellite image for bathymetric purposes, provides a cost-effective reconnaissance tool for assessing remote areas and mapping shallow waters (Freire, 2017). SDB also presents some limitations that prevent accurate depth information. Depending on the method applied, for instance, SDB is highly dependent on the available depths to perform surface modeling (Freire, 2017), the approach is limited to optically shallow waters (Westley, 2021), and, usually, depth information cannot be retrieved beyond 30 m, even in clear water. In addition, the accuracy is considerably low when compared to conventional surveying and mapping technologies. Although IHO S-4 (2018) recognizes the use of satellite images for representing shallow areas when reliable hydrographic survey data is limited or non-existent, the use of SDB data on charts is still in its infancy. While the IHO decision represents an excellent beginning for the use of SDB in nautical cartography products, further investigation related to the accuracy, datasets, and models of this technique is required.

Despite the limitations, previous studies have demonstrated the potential of estimating bathymetry using satellite data. Assuming homogeneity of bottom type and water clarity within the image, the usual practice is to apply a single global bathymetric calibration model for the entire area. This assumption also considers the water column parameters to be negligible, while a linear relationship between reference depths and the SDB model (e.g., that developed by Stumpf, *et al*., 2003) is applied to retrieve bathymetry. However, recognizing that homogeneity does not generally exist in the entire scene and that spatial heterogeneity of sea bottom type and water column may affect the model calibration (and, thus, depth estimation), this work investigates:

- an extended Dierssen model including the water column parameters into the SDB model, and
- vertically and horizontally segmenting the image scene into smaller spatial units for model training

with the aim to minimize the bottom and water clarity heterogeneity reflected in the model calibration and enhance the overall accuracy and coverage of the bathymetry estimation model. The remainder of this work is organized in the following manner:

Chapter 2 discusses the problem, questions, and objectives of this research work.

Chapter 3 provides a review of the important concepts for the development of this study, especially with the focus on SDB concepts and the development of the extended Dierssen model.

Chapter 4 presents the methodology used to develop the geographic models and the steps necessary for its application.

Chapter 5 details the areas of study and the results of each segmentation approach and a comparison to those of the conventional approach.

Chapter 6 is the discussions and suggestions for future works, and, lastly,

Chapter 7 is the conclusion.

CHAPTER 2 : RESEARCH QUESTIONS AND OBJECTIVES

Due to the limited resources and the limitations of traditional survey methods discussed in the Introduction, the current availability of accurate bathymetric data remains incomplete for nautical chart purposes, regardless of the many efforts to keep charts up-to-date and fill coastal data gaps. Given the advantages of deriving bathymetry from satellite images over large areas at low cost, SDB is considered to have a potential to supplement traditional surveys in shallow areas. The common practice in previous studies has been to consider the bottom type and water column contribution constant within the scene. In reality, and especially in complex environments, the assumed homogeneity is usually violated, something that degrades the performance of the conventional method and the accuracy of estimated depths. To overcome this limitation, this study attempts to incorporate the spatial heterogeneity of seabed and water column into the solution with the aim to improve the accuracy and coverage of the derived bathymetry from imageries. Thus, accuracy improvement of retrieved depths from satellite images is of major interest in this research, in an attempt to demonstrate the ability of SDB as an alternative solution for nautical chart purposes.

2.1 - Research Questions

Focusing on the statement previously described, the following questions summarize the present study:

a. Can the calculation of water column parameters improve the accuracy of depth estimation?

b. Can the horizontal segmentation of the scene improve the accuracy of depth estimation?

c. Can the vertical segmentation of the scene improve the accuracy of depth estimation?

2.2 - Research Objectives

To answer the research questions, the following objectives are defined in this study:

1. Enhance the SDB with a extended Dierssen model by adding information from water column contribution as a parameter in calibrating the model.
2. Apply the extended Dierssen model to estimate bathymetry using geographic models, i.e., subdividing the scene into vertical and horizontal segments.
3. Evaluate the performance of the various geographic model against the conventional global and linear method.

CHAPTER 3 : BACKGROUND INFORMATION

3.1 SDB concept

The ability to retrieve bathymetric information from satellite image is a promising technique that emphasizes the potential use of optical satellite remote sensing sensors to support various applications, such as coastal zone management, reconnaissance surveys, or safety of navigation. It began to be considered a tool for bathymetric purposes in the early 1970's with the mapping of shallow areas in the Bahamas and off the coast of Florida using Landsat 1 (Polcyn and Lyzengam, 1973).

SDB relies on the observed radiance as a function of wavelength and depth. Irradiance decays exponentially with depth, due to properties that depend on the water and its constituents (Ashphaq, 2021). Several factors, such as the nature of seabed, atmospheric conditions, and water column parameters, attenuate the signal and the radiance captured by the sensor.

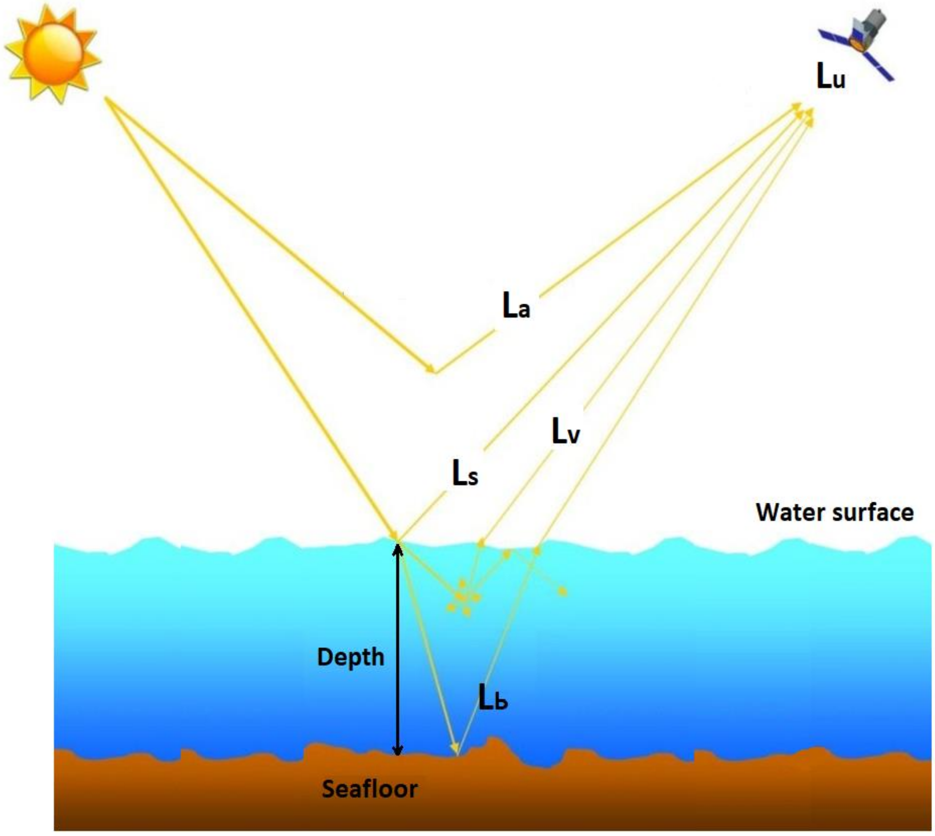


Figure 1 - Physical principle of the SDB (adapted from Vojinovic, 2013).

The light (upwelling radiance) from the sun (L_u),captured and recorded by the remote sensor, comprises four components: the bottom reflectance (L_b), which represents the energy reflected from the seafloor and carries the information about the water depth; the atmospheric path radiance (L_a), which is the light that never reaches the water and is scattered one or more times by the atmospheric gases and aerosols; the subsurface volumetric radiance (L_v), which is the result of energy reflecting in the water by inorganic and organic constituents; and the surface radiance (L_s), which represents the radiance reflected toward the sensor by the water surface (Mobley, 2021). Based on the above and as illustrated in the Figure 1, the radiance registered (L_u) in the sensor is provided by the following equation:

$$L_u = L_b + L_a + L_v + L_s \qquad (1)$$

3.1.1 Optical properties

The sunlight that is scattered upward within the water is captured by the sensor. Its radiant power per unit area incident on the surface is defined as solar irradiance (Ed), while the electromagnetic radiation collected by the sensor within a given solid angle in a specified direction is the radiance (Lu) (Mobley, 1994). The energy flow within the sun results in a surface temperature of around 5800 K, so the solar radiation spectrum is like that of a 5800 K blackbody. However, the spectral range of solar radiation flux reaching the Earth's surface is slightly different after the absorption and dispersion of the solar irradiance in the atmosphere due to various gas components. Visible rays (distributed from 400 to 700nm), for example, represent 38.2% of the irradiance at sea level of the total solar energy (Mobley, 1994). Figure 2 shows the spectral solar radiance for different wavelengths at the top of the atmosphere and after interaction with gases and aerosols on the sea surface.

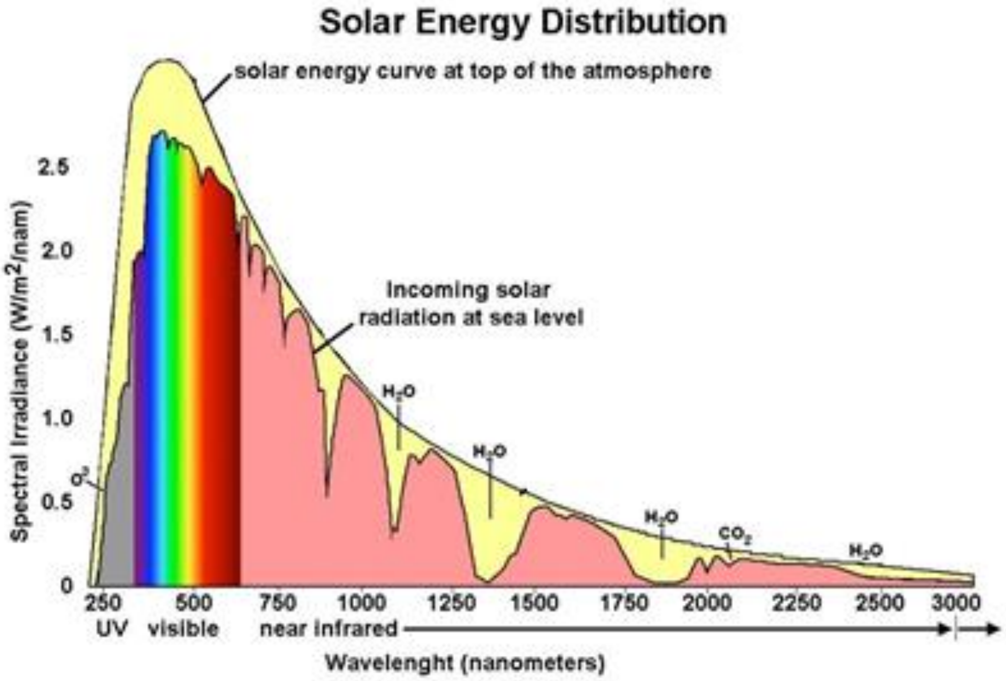


Figure 2 - The spectral energy distribution of solar radiation at the top of the atmosphere compared with that at sea level (Northern Arizona Wind & Sun, 2022).

The light’s behavior is highly affected by the nature of the medium it is crossing. Properties that depend only on the water and its constituents are defined as inherent optical properties (IOPs). Absorption and scattering are the two most important IOPs, represented by the absorption coefficient and the volume scattering (Mobley, 1994). In contrast, the apparent optical properties (AOPs) depend on the water's physical proprieties and its constituents and the geometric structure of the ambient light field. Since it is relatively easy to measure AOPs, such as upwelling radiance (L_u) and downwelling radiance (E_d), they are usually computed to retrieve unknown parameters from the water body.

Diffuse attenuation coefficient (K) is the main AOP used for describing optical properties in natural water bodies. This parameter represents the sunlight decay and depends on water and atmospheric conditions and wavelength (λ) (Herlevi, 2002). Under typical water conditions, i.e., ignoring water surface and bottom boundary conditions, the solar irradiance decreases almost exponentially as going with depth. Assuming no losses of energy in the atmosphere, the radiance on the water surface is $L(0, \lambda_i)$ and the radiance at some optical depth $L(z, \lambda_i)$, can be calculated by applying Beer's Law (Beer, 1852):

$$L(z,\lambda_i) = L(0,\lambda_i) \times e^{-2K(\lambda_i).z} \quad (2)$$

Following Equation 2, observed radiance $L_{obs}(\lambda_i)$ captured by the sensor can be related to the radiance interacting with the water column and bottom throughout a simplified radiative transfer equation (RTE) (Maritorena *et al.* 1994; Philpot 1989).

$$L_{obs}(\lambda_i) = \left(L_b(\lambda_i) - L_w(\lambda_i)\right) \times e^{-2K(\lambda_i).z} + L_w(\lambda_i) \quad (3)$$

Where $L_b(\lambda_i)$ is the contribution from the bottom/target radiance, $L_w(\lambda_i)$ represents the radiance from *optically deep water* (i.e., the depth beyond which contribution are only from water color and suspended particles (Pe'eri *et al.*, 2014)), and $K(\lambda_i)$ is the diffuse attenuation coefficient, to water depth (z). It is important to mention that most of the light is absorbed or scattered in the first meters of the ocean, with most (78%) of the visible spectrum being absorbed within 10 meters (Godwin, 2021). In clear water conditions, longer wavelengths of visible spectrum, such as red are absorbed at a shallower depth than shorter wavelengths such as blue (around 0.4 - 0.5 µm) and green (around 0.5 - 0.6 µm), that contain more energy and penetrate deeper into water (Godwin, 2021). In fact, from 100 to 200 meters deep, practically the total solar radiation has been absorbed, and at some point, depending on the wavelength, the light will not penetrate the water (Figure 3). This maximum depth that the light can penetrate water and the sensor capture the reflected radiance (L_{obs} → L_w) is defined as the *extinction depth* in SDB approaches (Pe'eri *et al.*, 2014).

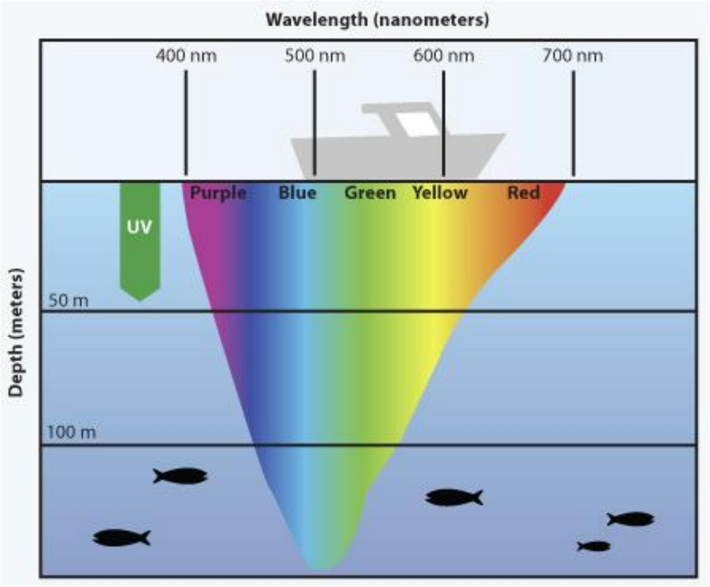


Figure 3 – Different wavelengths of visible light penetrate differently into the ocean depths, (University of Minnesota Sea Grant Program).

3.1.2 Approaches

Researchers have focused on developing and improving cheaper methods of deriving bathymetry since the late 1970s. SDB from multispectral remote sensing has shown to be a promising approach to providing a solution for shallow areas at a low cost. The simplified RTE, defined by Equation 3, can describe light propagation through the water column, including reflection from the bottom. However, an inverse model is required to extract depth information from remotely sensed image, driving different approaches. According to Philpot *et al*. (2004), the techniques are classified into analytical (Lyzenga, 1978; Philpot, 1989; Frener *et al*., 2012) and optimization methods (Dierssen *et al.*, 2003; Stumpf *et al.*, 2003; Vanderstraete *et al.*, 2006; Su *et al*., 2008):

- The **analytical methods** are purely based on the manner of light penetration in water. The optical properties of the water body, such as the absorption coefficient, bottom reflectance, and sediment concentration, are related to the light within the water through the RTE (Zandbergen, 2020). However, even for a very simple situation, many variables are required to achieve an analytical solution.

Consequently, these methods result in highly accurate estimations of water depth; however, they are very complex to execute as several in-situ parameters are required, including a precise atmospheric correction. The feasibility of deriving bathymetry from remote sensing image was first demonstrated by Lyzenga in 1978 using the analytical method to formulate a relationship between radiance observed in aerial imageries over clear shallow water and bottom reflectance. Afterwards, numerous approaches have been developed relating the optical properties of the water body and light using RTE. Philpot (1989) expanded Lyzenga's work by discussing an extension of the inversion algorithm to more complex scenes where both bottom types and water clarity vary spatially within the scene. Lyzenga *et al*. (2006) presented a correction for sun-glint effects applied prior to the application of the algorithm without compromising the correlation between radiance and depths.

- In the **optimization methods**, the relationship between the observed radiance of a water body and the depth is established empirically without regard to how light propagates in water. The statistical relationship between the water depth and the radiance of spectral bands is used for SDB estimation (Holman, 2020). Optimization methods have also been widely used due to their relative simplicity. The most common approach was proposed by Stumpf *et al.* (2003) where a band-ratio algorithm was suggested assuming the ratio of logarithms of band reflectance varies linearly with depth. Dierssen *et al.* (2003), following the same principle, applied a log-difference concept to derive bathymetry in turbid waters. Su *et al.* (2008) enhanced Stumpf's algorithm by using Levenberg-Marquardt optimization to deal with the nonlinear inversion by applying an automated method for model calibration. Kanno *et al*. (2011) presented a semi-parametric regression model by combining Lyzenga *et al.* (2006) method and spatial interpolation for more accurate water-depth mapping.

Besides analytical and optimization methods, other methodologies have been investigated, such as quasi-analytical or semi-analytical methods (McKinna *et al*., (2015), Werdell *et al.*, (2018), and Lee, *et al. (*2002). Thenceforth several algorithms have been proposed and tested to extract bathymetry for a variety of environmental conditions and satellite imageries by establishing the relationship between image pixel values and known water depth values.

In the context of optimization approaches, typically, constant water column parameters within a scene are assumed, resulting in poor bathymetry estimates in areas with non-uniform bottom and water column conditions (Freire, 2017). Freire (2017) developed a solution into an extended SDB model based on Dierssen's band ratio algorithm to reduce bottom heterogeneity and address these constraints.

Furthermore, common practice is to apply a single model for the entire area. Based on that, Su *et al*. (2013) presented a log-linear Lyzenga's model partitioning the study area into geographic regions based on bottom depths, distance from the shoreline, and distribution of the prediction residuals of the model, and then calibrated the parameters within each subregion. Poursanidis *et al*. (2019) investigated the effectiveness of deriving depth information from high temporal and spatial resolution data, using PlanetScope CubeSats imagery. For a best-fitted model, two regressions were applied training polynomial models in 0-10m and 10-25m depth ranges. Vargas, *et al*. (2021) tested different depth ranges with a simple regression, replacing green by red band to estimate depth in shallow water.

Wei & Theuerkauf, 2020 proposed a multitemporal SDB workflow based on three different regression procedures partitioning the dataset and pointing out areas suitable for each case.

3.1.3 Band-ratio method

A subcategory of the optimization approaches, is the *band-ratio algorithm.* This technique uses the band-ratio transform to linearize the relationship between bands and depth (Wei & Theuerkauf, 2020). It is based on the concept that solar irradiance for each wavelength is absorbed differently due to the IOP of the water. The changes in depth compared to the change in spectral reflectance due to bottom variation will have more influence (Figure 4). Therefore, at different depths, the difference in ratio between two bands will be more evident than the change caused by bottom type heterogeneity (Stumpf *et al*., 2003).

Typically, the procedure for deriving bathymetry using the band-ratio technique can be summarized in the following steps (Pe'eri *et al*., 2014):

1. Preprocessing
2. Radiometric enhancement
3. Land/water separation
4. Spatial filtering
5. Applying the bathymetry algorithm
6. Identifying the extinction depth
7. Vertical referencing.

Preprocessing refers to a set of data preparation tasks. It begins with the study area selection and discovery of satellite images with suitable environmental conditions for bathymetric extraction (Pe'eri *et al*., 2014). Besides finding the appropriate location, applying a *radiometric enhancement* of atmospheric effects to the image is also beneficial. Radiometric enhancement can improve the accuracy of the products, which is particularly important for marine and inland water analysis. In this context, *spatial filters* guarantee a radiometric enhancement and provide more quality to the image.

Next, l**and areas** are separated from water bodies based on a remote sensing derived index that relates their reflectance in different bands**.** The Normalized Difference Water Index (NDWI) is commonly used among several approaches. As water is close to opaque (lower reflectance) in the NIR portion of the electromagnetic spectrum, the water appears dark (low digital values), while land areas appear brighter (high digital values) (Gao, 1996). On the other hand, the water body is characterized by higher digital values than NIR-band pixel values in visible bands (Figure 4). By relating the green and NIR bands, NDWI positive values represent water areas, while negative values represent land areas (McFeeters, 1996).

$$NDWI = \frac{L_G - L_{NIR}}{L_G + L_{NIR}} \tag{4}$$

where, L_G represents the reflectance of the green band and L_{NIR}, the reflectance of the near-infrared wavelengths.

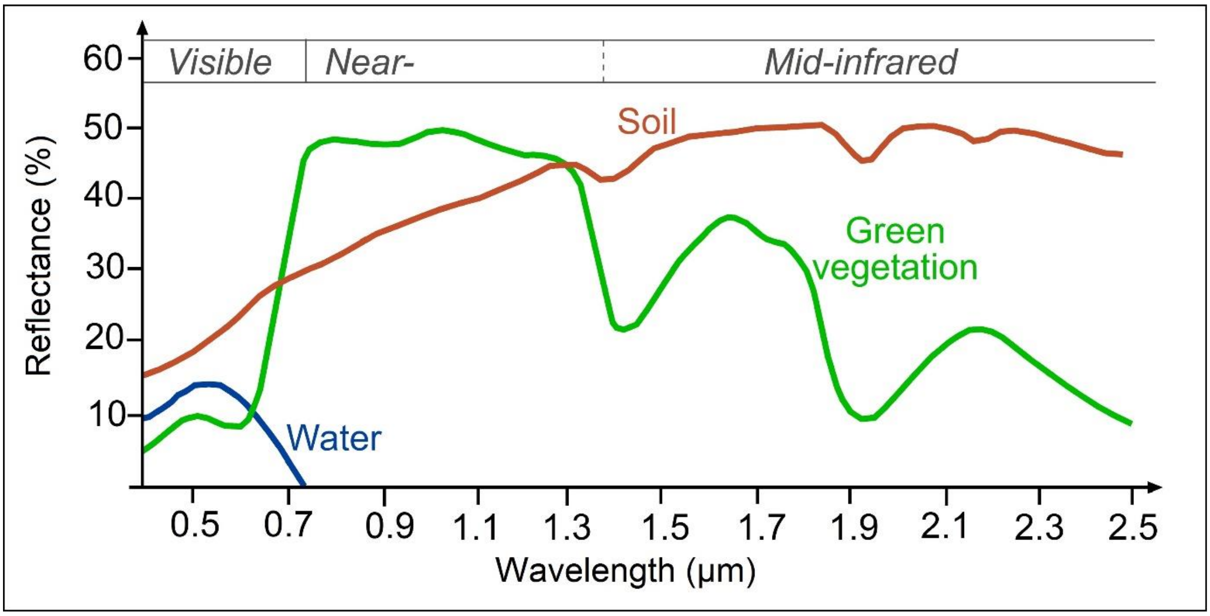


Figure 4 - Reflectance of soil and water at different wavelength (SEOS, 2022).

The next step refers to the *application of the algorithm to extract bathymetry***.** The band-ratio algorithm utilizes *in-situ* water depth data to establish a statistical relationship between water depth and radiance (pixel value) to estimate bathymetry (Lyzenga, 1978; Lyzenga, 1985; Stumpf *et al.*, 2003). This step includes the selection of control points used for training and testing of the model. Ideally, soundings should come from a temporally near to the image, survey (ALB or acoustic); however, chart soundings may also be used to aid the process.

While many of these algorithms are well-established, their performance is subject to environmental conditions, availability of auxiliary data, quality, and preprocessing of images (Holman, 2020). Thus, most research has focused on developing algorithms to improve the accuracy of bathymetry through the above. The development of detailed calculations is described to understand the origin of this algorithm. The band ratio is derived by rearranging Equation 3 and isolating the radiance parameters, considering bands 1(λ_1) and 2 (λ_2).

$$\frac{\left(L_{obs}(\lambda_1) - L_w(\lambda_1)\right)}{\left(L_b(\lambda_1) - L_w(\lambda_1)\right)} = e^{-2K(\lambda_1).z}$$

$$\frac{\left(L_{obs}(\lambda_2) - L_w(\lambda_2)\right)}{\left(L_b(\lambda_2) - L_w(\lambda_2)\right)} = e^{-2K(\lambda_2).z} \tag{5}$$

By dividing both wavelengths, the equation becomes:

$$\frac{\left(L_{obs}(\lambda_1) - L_w(\lambda_1)\right)}{\left(L_{obs}(\lambda_2) - L_w(\lambda_2)\right)} . \frac{\left(L_b(\lambda_2) - L_w(\lambda_2)\right)}{\left(L_b(\lambda_1) - L_w(\lambda_1)\right)} = \frac{e^{-2K(\lambda_1).z}}{e^{-2K(\lambda_2).z}} \tag{6}$$

This allows to reduce the computation and isolate depth z:

$$\ln\left[\frac{\left(L_{obs}(\lambda_1) - L_w(\lambda_1)\right)}{\left(L_{obs}(\lambda_2) - L_w(\lambda_2)\right)} \times \frac{\left(L_b(\lambda_2) - L_w(\lambda_2)\right)}{\left(L_b(\lambda_1) - L_w(\lambda_1)\right)}\right] = -2K(\lambda_1)\mathrm{z} + 2K(\lambda_2)\mathrm{z} \tag{7}$$

Subsequently, the solution for depth z is given by:

$$z = \frac{1}{-2(K(\lambda_1) - K(\lambda_2))}\left[\ln\left(\frac{L_{obs}(\lambda_1) - L_w(\lambda_1)}{L_{obs}(\lambda_2) - L_w(\lambda_2)}\right) + \ln\left(\frac{L_b(\lambda_2) - L_w(\lambda_2)}{L_b(\lambda_1) - L_w(\lambda_1)}\right)\right] \tag{8}$$

For simplifications, m_0 and m_1 are applied, and Equation 9 can be written:

$$m_1 = \frac{1}{-2(K(\lambda_1) - K(\lambda_2))} \qquad m_0 = m_1 \ln\left[\frac{L_b(\lambda_2) - L_w(\lambda_2)}{L_b(\lambda_1) - L_w(\lambda_1)}\right] \tag{9}$$

$$z = m_1 \times \ln\left[\frac{L_{obs}(\lambda_1) - L_w(\lambda_1)}{L_{obs}(\lambda_2) - L_w(\lambda_2)}\right] + m_0 \tag{10}$$

The most common band-ratio optimization methods were developed by Dierssen *et al.* (2003) and Stumpf *et al.* (2003). Under the assumption of the bottom type and water column homogeneity within the entire scene, both algorithms consider the water column (L_w) as approximately zero. Thus, the procedure to extract bathymetric information is relatively simple and requires solving for only two parameters, m_0 and m_1. Dierssen *et al.* (2003) used a log-difference concept to derive bathymetry according to the following Equation:

$$z = m_1 \times \ln\left(\frac{L_{obs}(\lambda_i)}{L_{obs}(\lambda_j)}\right) + m_0 \tag{11}$$

Similarly, Stumpf *et al.* (2003) presented a model that applies the division between the observed radiance log values of two bands:

$$z = m_1 \times \left(\frac{\ln(L_{obs}(\lambda_i))}{\ln(L_{obs}(\lambda_j))}\right) - m_0 \tag{12}$$

Where $L_{obs}(\lambda)$ represents the observed radiance for each band; m_0, the translation coefficient, based on bottom return and a diffuse attenuation coefficient; and m_1, the scaling coefficient based on the diffuse attenuation. The main advantage of these algorithms is that they do not require any optical properties to determine the depth and consider very few parameters.

When the linear model is applied, depth estimation is limited by the best linearity between the algorithm and depth (6m in Figure 5). Areas deeper than this point will show an almost constant value and are assumed optically deep water, where the seafloor is too deep for a bottom reflectance signal to be returned. This depth is commonly referred to as “extinction depth”, however, in the context of this work the term “effective optical depth” is used, due to that it is still possible to extract information from the seafloor beyond this depth, represented by quasi-optically deep (areas with weak correlation but still detectable) instead of optically deep water.

When a single and linear model is applied, the effective optical depth is identified by plotting the SDB algorithm (pseudo bathymetry (pSDB)) against the reference bathymetry. Based on a visual inspection of the graph, the area with the best linearity (highest coefficient of determination R^2) between reference bathymetry and the SDB algorithm can be selected (Pe’eri *et al.*, 2014). Over this area, the calculated parameters in the regression analysis, gain (m_1), and offset (m_0) can be estimated and applied to the ratio transform output for each pixel value, performing *vertical referencing***.** This procedure excludes the need for tide correctors and tide-coordinated images because this is automatically accounted for using control points selected from a nautical chart to determine the transformation parameters (gain and offset) (Pe’eri *et al.* 2014).

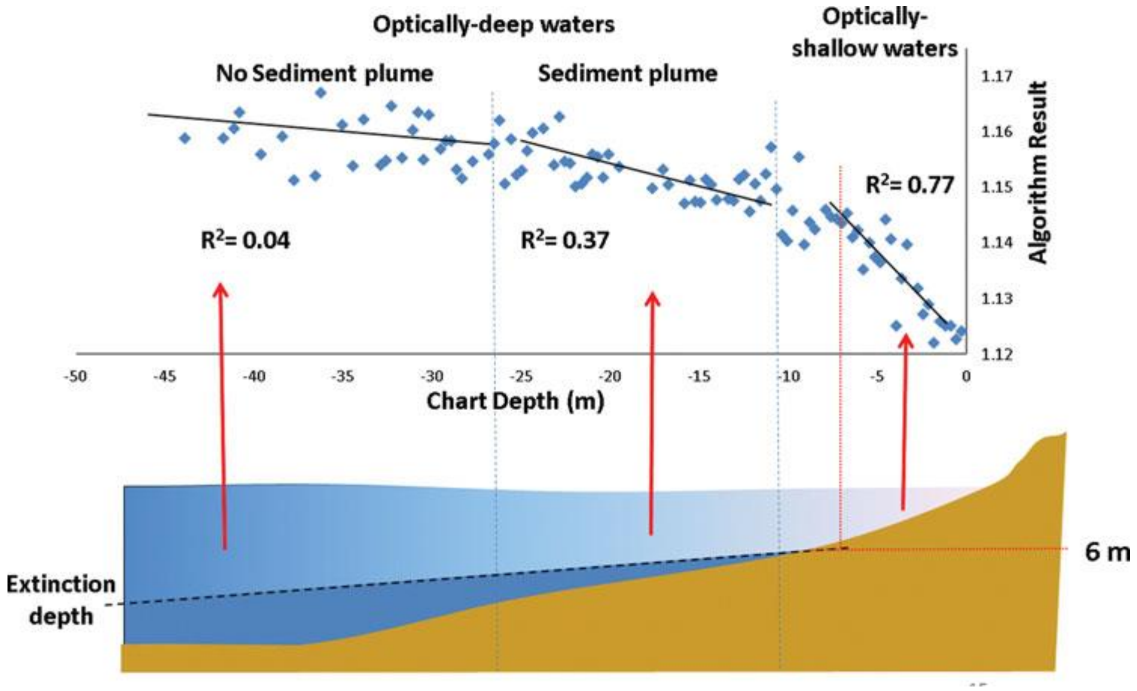


Figure 5 - *Extinction depth* determination based on linear band ratio algorithm applied over a study area in Cape Ann, MA (Pe'eri et. al, 2014).

Although the linear model, represented by equation (11) is simple and, in my opinion reasonably adequate, areas with variable seafloor types and/or water optical proprieties will affect depth estimates using SDB (Holman, 2020). As such, including a water contribution in the model can produce improved results over areas with a rough bottom.

To address these constraints and recognizing that water column contribution is considerable, Freire (2017) extended Dierssen's algorithm assuming the water column as a parameter, solving four unknowns $(m_1, L_w(\lambda_1), L_w(\lambda_2), m_0)$, of Equation 10. To reach a solution, Freire (2017) used the Least Squares Method (LSM) by applying parametric adjustment. Following the LSM steps presented by Wells and Krakiwsky (1971), a nonlinear solution can be obtained by the following steps:

1. Linearization of the function using Taylor's series generating an approximated solution.
2. Form the Jacobian matrix, the derivative of the equation.

3. Apply the Least squares estimation to minimize the quadratic sum of the residuals.

The process is repeated until it achieves minimum Root Mean Square Error (RMSE).

The first consideration is that if the mathematical model expresses the sum of the total observation vector (L_a) as a function of the sum of the total solution vector (X_a), it can be solved by applying the parametric method.

$$F(X_a) = L_a \tag{13}$$

Considering the full Dierssen's model, i.e., accounting for the water column parameter, the function can be written as:

$$\underbrace{\begin{pmatrix} z_1 \\ \cdot \\ \cdot \\ z_n \end{pmatrix}}_{L} + \underbrace{\begin{pmatrix} v_1 \\ \cdot \\ \cdot \\ v_n \end{pmatrix}}_{V} = \underbrace{\begin{pmatrix} m_{1.}\ln\left[\dfrac{L(\lambda_1)_1 - L_w(\lambda_1)}{L(\lambda_2)_1 - L_w(\lambda_2)}\right] + m_0 \\ \cdot \\ \cdot \\ m_1.\ln\left[\dfrac{L(\lambda_1)_n - L_w(\lambda_1)}{L(\lambda_2)_n - L_w(\lambda_2)}\right] + m_0 \end{pmatrix}}_{F(X)} \tag{14}$$

Where, L is the observation vector, V, the residual vector and F(X), the nonlinear function. As mentioned above, the first step is given by the linearization of the model (Equation 15) using Taylor's series, and an initial estimate of the solution vector X$_0$ needs to be defined to perform the solution.

$$L_a = F(X_0) + \left.\frac{\partial F}{\partial X_a}\right|_{X_0} (X - X_0) \tag{15}$$

Since $L_a = L_b + V$, where L_a refers to the sum of the observation vector, L_b represents the observation vector and V the residual vector, the linearization can be expressed as:

$$\underbrace{F(X_0)}_{L_0} + \underbrace{\left.\frac{\partial F}{\partial X_a}\right|_{X_0}}_{A} X - (L_b + V) = 0 \rightarrow L_0 + AX - L_b = V \tag{16}$$

The minimum value of $V^T PV$ is the metric for adjustment goodness under LSM (Camargo, 2000). Assigning the minimum $V^T PV$ to $L_0 + AX - L_b = V$, the solution is given by:

$$X = -(A^T PA)(A^T PL) \tag{17}$$

Where:

$$L = L_0 - L_b$$

The second step is expressed by the Jacobian or design matrix (A). In the case of the extended Dierssen model, A is represented by:

$$A = \begin{pmatrix} \ln\left[\frac{L(\lambda_1)_1 - L_w(\lambda_1)}{L(\lambda_2)_1 - L_w(\lambda_2)}\right] & -\frac{m_1}{L(\lambda_1)_1 - L_w(\lambda_1)} & \frac{m_1}{L(\lambda_2)_1 - L_w(\lambda_2)} & 1 \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \ln\left[\frac{L(\lambda_1)_n - L_w(\lambda_1)}{L(\lambda_2)_n - L_w(\lambda_2)}\right] & -\frac{m_1}{L(\lambda_1)_n - L_w(\lambda_1)} & \frac{m_1}{L(\lambda_2)_n - L_w(\lambda_2)} & 1 \end{pmatrix} \tag{18}$$

Since adopting the initial approximation vector introduces errors in the adjustment, an iteration process is required (Camargo, 2020). The convergence criterion adopted in this work is the minimum RMSE between the observation vector (L_b) and the model ($F(X)$). Figure 6 shows the flowchart of steps required in the nonlinear model.

The biggest challenge in solving a system of nonlinear equations is the definition of the initial approximation to the solution, the vector X_0. If it is poorly defined, it will induce non-convergence of the model or converging to an incorrect solution.

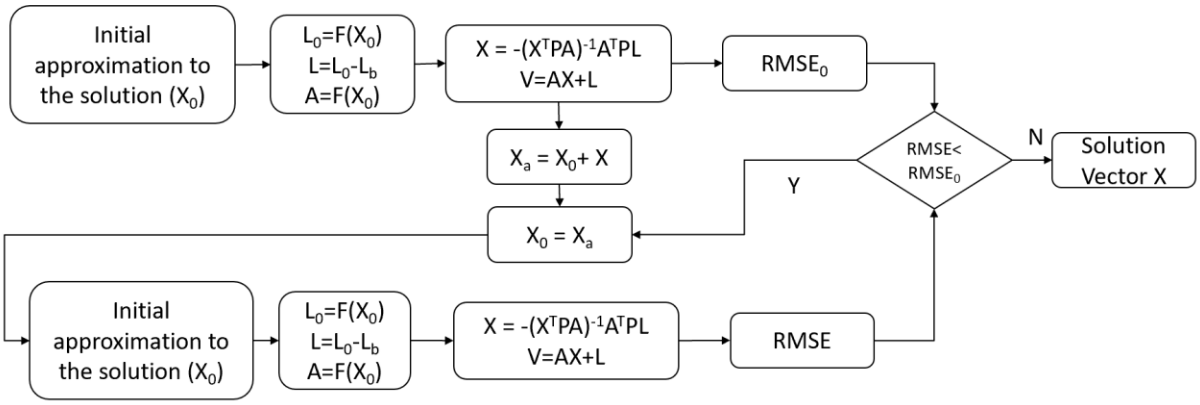


Figure 6 - Flowchart of nonlinear adjustment solution.

3.2 SDB and Nautical Charts

Nautical charts provide important information that enables the mariner to make decisions necessary for safe navigation. Collecting and keeping the information up to date in a chart is a continuous process. However, despite the best efforts of the hydrographic community, the chart will always depict elements compiled from data of varying quality, collected using different technologies, and at different times (Kastrisios & Ware, 2022). To support mariners information on data reliability, one of the elements compiled in the nautical chart is data quality, more recently portrayed by the Zone of Confidence (ZOC), classified in six categories: A1, A2, B, C, D, and U (where data is yet to be assessed) (IHO, 2002). ZOC or categorical ZOC (CATZOC) classification is a result of the total horizontal (THU) and vertical (TVU) uncertainties of the hydrographic data, seafloor coverage requirements, and typical survey characteristics. Table 1 summarizes the CATZOC categories and the associated TVU required of charted soundings for depths of 10 and 20 m.

Table 1 - A summary of the ZOC classification with the associated THU, TVU, seabed coverage and feature detection.

ZOC	THU (m)	TVU (m)			Full Seabed Coverage Achieved	Significant Features Detected
		Accuracy	10m	20m		
A1	5 + 5%d	0.5 + 1%d	0.6	0.7	Yes	Yes
A2	20	1 + 2%d	1.2	1.4	Yes	Yes
B	50	1 + 2%d	1.2	1.4	No	Undetected features may exist
C	500	2 + 5%d	2.5	3	No	Anomalies may be expected
D	>500	> 2 + 5%d	-	-	No	Large anomalies may be expected
U	The Quality of data has yet to be assessed					

For SDB to be incorporated into cartographic products, it is necessary to assign a CATZOC, and meet all data quality parameters. However, this practice can lead to underrepresentation of horizontal and vertical uncertainty when they are more accurate than the other requirements. As an alternative, SOUACC and POSACC are attributes, defined by IHO standards S-57 (2000), capable of individually indicating the reliability of the value of a sounding, or position, respectively. Nonetheless, a major challenge in using SDB for nautical charting purposes is the achievable vertical accuracy, especially when compared to traditional surveys.

Although meeting a high level of depth accuracy is not a simple task in SDB, as it strongly depends on the environmental conditions and the image quality, in recent years, extracting bathymetry from imagery has been recognized as a powerful tool to be used in cartographic production. Several studies have been conducted to demonstrate the ability to produce and update nautical charts. For instance, Service Hydrographique et Océanographique de la Marine (SHOM), the French Hydrographic Office, produced the first satellite-derived chart (SDC) in 1983, using Spot SDB simulation with a CASI radiometer in New Caledonia, and is investigating different sources and approaches to this day. Dekker *et al.* (2012), using ALOS and Quickbird data in Queensland, Australia, demonstrated the adequacy of SDB for use in nautical charts, providing a TVU value within (IHO,2008) 1a and 1b Order survey standards. The United Kingdom Hydrographic Office (UKHO) published in 2015 its first nautical chart (chart number 2066 – Figure 7), with depths derived from satellite image. In 2018, the Canadian Hydrographic Service (CHS) also used information derived from SDB on a chart for Havre-aux-Maisons (chart number 4955), classified as CATZOC C level.

Despite the advances using SDB to estimate depths for cartographic purposes, there is still the need to improve the models and testing different datasets in different geographic configurations and environmental conditions in order to achieve better accuracy.

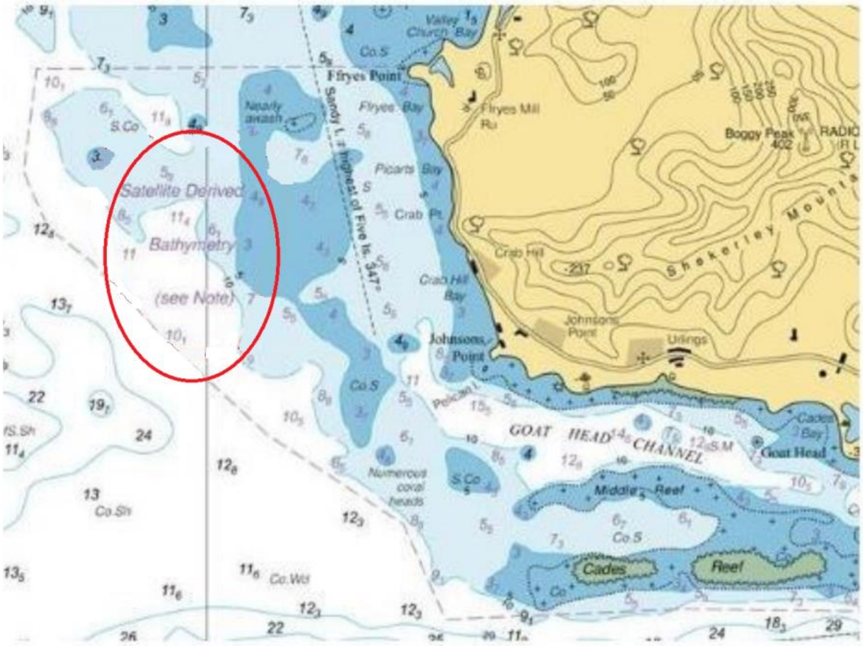
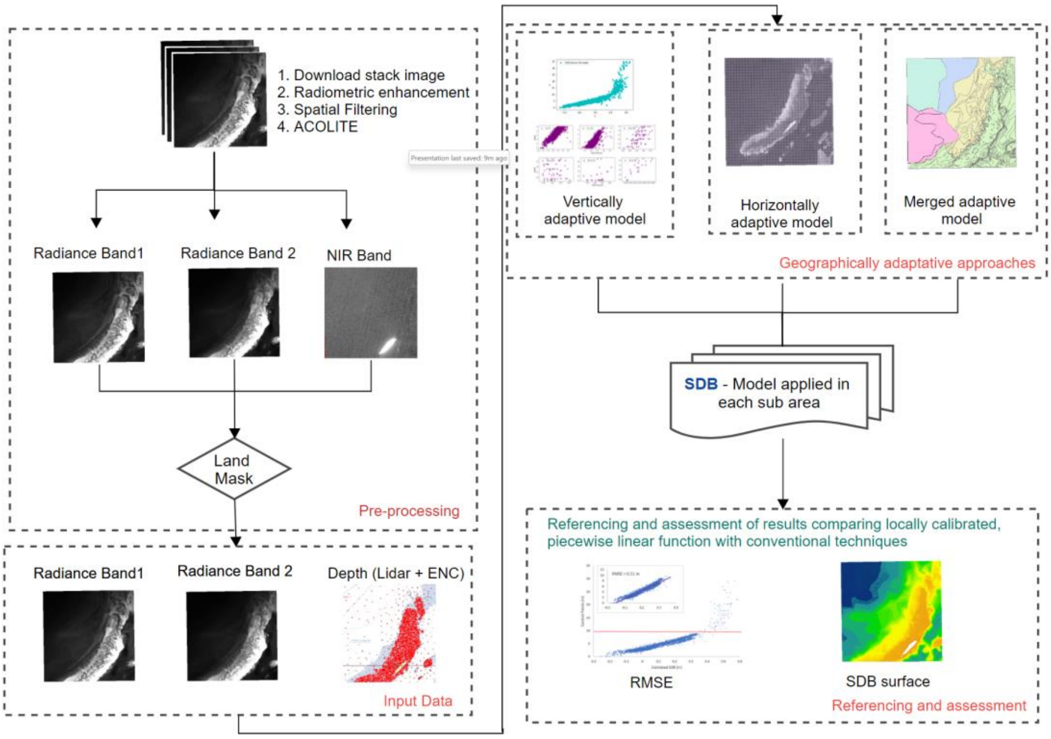


Figure 7 - Nautical Chart BA 2066, using SDB for mapping shalow water bathymetry (Mavraeidopoulos, *et al.*, 2017)

CHAPTER 4 : METHODOLOGY

4.1 Overview

This research investigates how segmenting the area can achieve better bathymetric estimates using satellite image for nautical charting purposes, aiming to assist in safe and efficient maritime navigation. The proposed methodology, presented in this Section and summarized in the workflow (Figure 8), uses the extended Dierssen model (Freire, 2007) that incorporates water column parameters. By splitting the data spatially into smaller units, this work aims to account for non-uniform sea-bottom conditions and improve the accuracy of estimated depths. The rationale behind the segmentation is that in smaller spatial units the environmental conditions should be more stable than within the entire scene, an assumption that finds its basis on Tobler's (1970) first law of geography.



1. Download stack image
2. Radiometric enhancement
3. Spatial Filtering
4. ACOLITE
Presentation last saved: 9m ago
Radiance Band1
Radiance Band 2
NIR Band
Land Mask
Pre-processing
Radiance Band1
Radiance Band 2
Depth (Lidar + ENC)
Input Data
Vertically adaptive model
Horizontally adaptive model
Merged adaptive model
Geographically adaptative approaches
SDB - Model applied in each sub area
Referencing and assessment of results comparing locally calibrated, piecewise linear function with conventional techniques
RMSE
SDB surface
Referencing and assessment

Figure 8 - Workflow for deriving bathymetry using the geographic model.

4.1.1 Radiometric enhancement

Obtaining a satellite image free of clouds, waves, and turbidity across an entire area is a challenging task. Depending on the environmental conditions and image quality, the estimated depth for a target area obtained from a given satellite image may be different from another image. Therefore, SDB is typically used in regions with clear water conditions since the light penetration will be higher (Pe'eri *et al*., 2014).

Also, ideally, the survey data should had been collected shortly before or after the acquisition of satellite image, which will enable the solution provide more accurate depth estimates. Therefore, besides the image quality, images should be selected as near in time as possible to the available survey dataset.

A few works do not consider rigorous atmospheric correction with the rationale that band ratio mitigates these effects. However, this work incorporates atmospheric correction to improve depth estimation accuracy. The ACOLITE algorithm, a free processor developed by Vanhellemont (2019) and Vanhellemont & Ruddick (2016) of the Royal Belgian Institute of Natural Sciences (RBINS), has been used for image quality enhancement, as demonstrated by Cabalerro *et al*., 2019. ACOLITE allows removal of the scattering effects of aerosol components over clear and turbid water (Martins *et al*., 2017) and supports atmospheric corrections of both Landsat-8 and Sentinel-2, and includes two algorithms, *Dark Spectrum Fitting* (DSF) and *Exponential extrapolation* (EXP) (Caballero *et al*., 2020).

The EXP assumes that the signal reflected from the water is zero in the SWIR bands, which means that any signal in the SWIR is assumed to be entirely caused by aerosol and Rayleigh scattering (Holman, 2020). After Rayleigh correction, the signal that remains is assumed to come from aerosol scattering, and it can be extrapolated from the SWIR to visible and near-infrared bands using an exponential function. However, this algorithm has generally poor performance in the blue bands. To address this common issue pointed out in the EXP algorithm, the "Dark Spectrum Fitting" (DSF) algorithm includes a robust automated band selection process and an optional image-based glint correction (Vanhellemont, 2019).

The DSF computes the best fitting aerosol model to estimate the atmospheric path reflectance by using multiple dark targets in the scene or sub-scene to construct a “dark spectrum” with no prior defined band (Vanhellemont, 2019).

ACOLITE products correspond to Remote sensing reflectance (Rrs, 1/sr) in all visible and Near-infrared (NIR) bands resampled to 10m pixel size (Caballero *et al.*, 2020), which can be directly utilized to retrieve depth information.

The atmospheric correction algorithm applied in this study is the recommended DSF. Both radiometric enhancements with ACOLITE and uncorrected atmospheric imageries were tested for further analysis.

4.1.2 Spatial filtering

To avoid speckle noise and enhance the image quality, a smoothing Low Pass Filter (LPF) is applied to bands blue coastal (B01), blue (B02), green (B03), and red (B04) utilizing the ArcGIS filter tool. A simple 2D convolution with 2 smoothing Kernel is applied removing noise and local variation. By default, it calculates the average value for each 3x3 neighborhood.

4.1.3 Land/ water separation

NDWI is computed to separate water bodies from land areas, using the green (B03) and the near-infrared (B08) bands in Sentinel-2, as proposed by McFeeters (1996) (Equation 4). This step is performed using the raster calculator tool in ArcGIS to create the NDWI and the set null tool to identify cell locations where NDWI presented negative values. These values represented by dry areas are removed from coastal aerosol, blue, green, and red bands. Figure 9 illustrates an example, where the white color represents land and the shades of grey the water body.

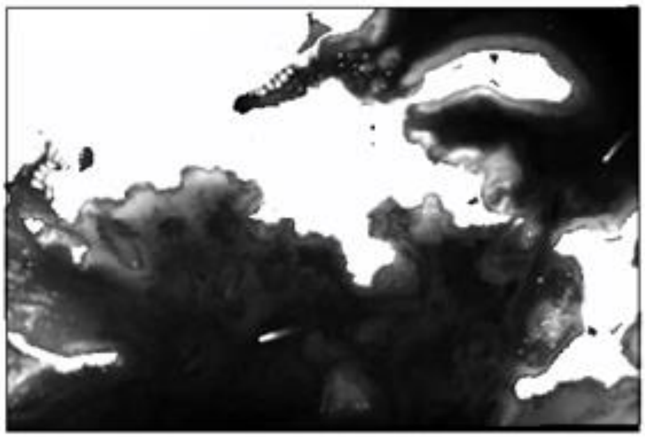


Figure 9 - NDWI for Dry Tortugas on the left and for STEER on the right.

4.1.4 SDB algorithm

The procedure to derive bathymetry from satellite scenes applied in this thesis is based upon the concept of Dierssen *et al.* (2003), which uses a linear log-difference concept to extract depth, resulting in the following Equation:

$$z = m_0 + m_1 \times \ln\left(\frac{L_{obs}(\lambda_1)}{L_{obs}(\lambda_2)}\right) \tag{19}$$

The pixel value of the ratio of two bands ($L_{obs}(\lambda_1)/L_{obs}(\lambda_2)$) was captured for each training point to create a list containing the log-difference model (pSDB) and depths. The data were reviewed for possible null values, and the depths were organized in ascending order. Averages of the band ratio values were compiled for each unique depth and plotted against the control points. By analyzing the scatter plot it is possible to infer the effective optical depth (Figure 10).

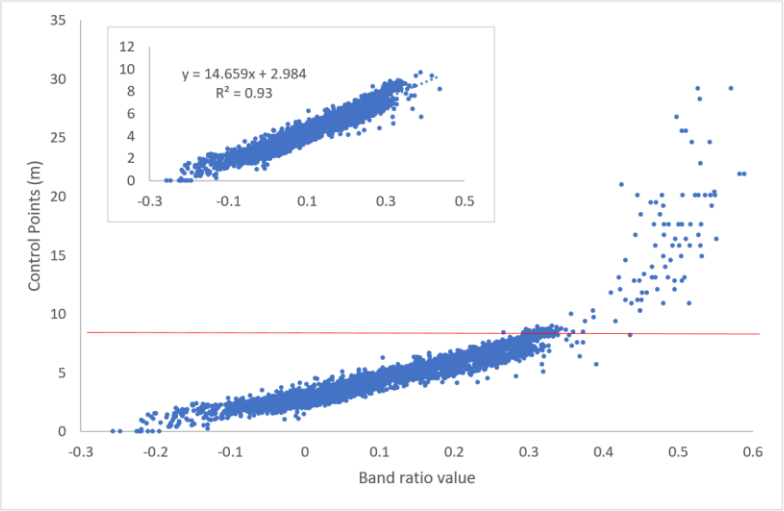


Figure 10 -Scatter plot showing the correlation between the band ratio (linear) values and the training points (control points) to estimate effective optical depth in the SDB (red line). Plot above shows the linear regression through the selected data up to extinction depth to extract m1=14.659 and m0=2.984.

Usually, the depth where the plotted points no longer portray a linear trend (represented by a red line in Figure 10) is considered the effective optical depth up to which depths can be estimated, and the single linear regression is applied to derive bathymetry. The parameters, m_0, and m_1, from the linear regression up to the limit depth are computed and applied to the image. These steps generate depth estimates referenced to the reference bathymetry according to a linear function.

The Dierssen model was extended by Freire (2017), who incorporated the water column parameters to address the heterogeneity of the bottom and water quality within the scene, as shown in Equation 20:

$$z = m_0 + m_1 \times \ln\left[\frac{L_{obs}(\lambda_1) - L_w(\lambda_1)}{L_{obs}(\lambda_2) - L_w(\lambda_2)}\right] \qquad (20)$$

As in the linear case (Dierssen model), for the extended Dierssen model a solution is reached through linear regression, using depth measurements as control points. However, due to the model not being linear, the final estimate represents an approximated solution to the observations. In addition, instead of two control points, at least four points are required for the extended Dierssen model solution. The process, implemented in Python, consists of the development of the following LSM steps:

1. Estimation of the initial vector,
2. Development of the Jacobian matrix, and
3. Minimization of the quadratic sum of the residuals.

The four parameters calculated for the entire image are applied to each image pixel. These steps generate the estimated SDB according to a extended Dierssen function.

4.1.4.1 Estimation of the initial vector

The unknown variables $(m_1, L_w(\lambda_1), L_w(\lambda_2), m_0)$ are split into linear ($X0_1$ and $X0_4$) and nonlinear ($X0_2$ and $X0_3$) parameters to estimate the initial approximation vector. Initially, considering the correlation between the log-difference bands and control points, the parametric method of the LSM is used to calculate the scaling coefficient ($X0_1$) and the translation coefficient ($X0_4$).

The steps are the same as the flowchart described in Figure 6. The difference refers to the Jacobian matrix, as the function to be solved is provided by Equation 19 instead of Equation 20. Also, the iteration process is not performed as it relates to just an initial solution. The following Jacobian matrix is applied to find $X0_1(m_1^0)$ and $X0_4$ (m_0^0).

$$A = \begin{pmatrix} \ln\left[\frac{L(\lambda_1)_1}{L(\lambda_2)_1}\right] & 1 \\ \cdot & \cdot \\ \cdot & \cdot \\ \ln\left[\frac{L(\lambda_1)_n}{L(\lambda_2)_n}\right] & 1 \end{pmatrix} \quad (21)$$

Where $L(\lambda_1)$ represents the radiance in the band 1, $L(\lambda_2)$ the radiance in the band 2, and n, the number of control points used to calibrate the model. Once $X0_1(m_1^0)$ and $X0_4$ (m_0^0) are calculated, the nonlinear model is used to estimate the initial water column components, considering the first guess as zero ($L_{w1}^0 = L_{w2}^0$=0). The parametric method of the LSM is conducted again to calculate the remaining parameters, which are composed of the initial solution vector X0₂ (L_{w1}^{0}) and X0₃ (L_{w2}^{0}), based on the following Jacobian matrix.

$$A = \begin{pmatrix} -\dfrac{m_1^0}{L(\lambda_1)_1 - L_{w1}^0} & \dfrac{m_1^0}{L(\lambda_2)_1 - L_{w2}^0} \\ \cdot & \cdot \\ \cdot & \cdot \\ -\dfrac{m_1^0}{L(\lambda_1)_n - L_{w1}^0} & \dfrac{m_1^0}{L(\lambda_2)_n - L_{w2}^0} \end{pmatrix} \qquad (22)$$

The initial approximation vector of the solution $\mathrm{X0} = [m_1^0,\ L_{w1}^0, L_{w2}^0, m_0^0]$ is then used to start an interactive adjustment for the nonlinear case.

4.1.4.2 Solution of vector

To calculate the solution vector $\mathrm{X} = \ [m_1, L_{w1}, L_{w1}, m_0]$, X0 is implemented in the Jacobian matrix:

$$A = \begin{pmatrix} \ln\left[\dfrac{L(\lambda_1)_1\ - L_{w1}^0}{L(\lambda_2)_1\ - L_{w2}^0}\right] & -\dfrac{m_1^0}{L(\lambda_1)_1 - L_{w1}^0} & \dfrac{m_1^0}{L(\lambda_2)_1 - L_{w2}^0} & 1 \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \ln\left[\dfrac{L(\lambda_1)_n\ - L_{w1}^0}{L(\lambda_2)_n - L_{w2}^0}\right] & -\dfrac{m_1^0}{L(\lambda_1)_n - L_{w1}^0} & \dfrac{m_1^0}{L(\lambda_2)_n - L_{w2}^0} & 1 \end{pmatrix} \qquad (23)$$

To calculate the solution $\mathrm{X} = -(A^T A)(A^T L)$, it is necessary to compute the vector L , based on the depths of the control points and expressed by:

$$L = \begin{pmatrix} m_1^0 \ln\left[\dfrac{L(\lambda_1)_1\ - L_{w1}^0}{L(\lambda_2)_1\ - L_{w2}^0}\right] + m_1^0 \\ \cdot \\ \cdot \\ m_1^0 \ln\left[\dfrac{L(\lambda_1)_n\ - L_{w1}^0}{L(\lambda_2)_n - L_{w2}^0}\right] + m_1^0 \end{pmatrix} \qquad (24)$$

4.1.4.3 Minimize the quadratic sum of the residuals

Among different convergence criteria, this study uses the quadratic sum of the residuals (RMSE) as a threshold. The final solution is achieved after n-iterations, (n), when no further convergence occurs, i.e., the RMSE between the observation vector, $L,$ and the nonlinear model, $F(X)$, converges, $RMSE_{n-1} < RMSE_n$.

$$RMSE_{\mathrm{n-1}} = \sqrt{\frac{V_{n-1}^T V_{n-1}}{N-U}} \quad RMSE_n = \sqrt{\frac{V_n^T V_n}{N-U}} \tag{25}$$

Where, N, represents the number of equations, i.e., the number of control points, and U, the number of parameters, i.e., 4 $(m_1, L_{w1}, L_{w1}, m_0)$.

4.1.5 Vertical Referencing

4.1.5.1 Reference system

Existing bathymetry is used as control points for the vertical referencing. Since different data sources can be applied to compile bathymetry, a common reference system between the dataset is essential to avoid irregularities. VDatum, a vertical datum transformation tool developed by the National Oceanic and Atmospheric Administration (NOAA), provides easy horizontal and vertical transformation into the same reference system for all coastal regions of the continental U.S., Puerto Rico, and the U.S. Virgin Islands.

4.1.5.2 ALB gridding

Control points from a reference dataset are used to establish a vertical transformation between the reference bathymetry and algorithm values. In most scenarios, where different source data constitute the study area, the control points' density and spatial distribution varies within the area under investigation. When sounding density is greater, the number of soundings must be reduced to a single point value per image pixel, usually using the average of the reference depths within each image pixel.

4.1.5.3 Selection of control points

From the reference bathymetry, depths are selected for training and testing of the model. If a small number of control points is selected, the solution may not converge; on the other hand, with a large number of control points the processing time required to calculate the vertical transformation increases and sometimes this crashes the system. After some testing, it was found that 5% of the available data for the control points performs satisfactorily, whereas with twice as many points, the system crashed. Of the 5%, randomly selected using the *create random points* in ArcGIS, two independent subsets are created in order to split training and test data sets - 70% are randomly selected again for training the model, and the remainder are used for testing (30%).

4.2 Geographic approaches

Assuming spatial homogeneity of bottom and water quality within a scene, the parameters defined by a single global model are ideal for the entire area, and the conventional model (hereinafter “conventional global”), is applicable to the entire image scene. However, conventional models present limited results due to the spatial heterogeneity of bottom type and water quality within the scene (Su *et. al*, 2013; Freire, 2017). Assuming that the magnitude of spatial heterogeneity is smaller (and possibly non-existent) in small regions, segmenting the scene into smaller spatial units should enable the algorithm estimate optimal parameters in a sufficient manner.

In addition, defining a threshold for a single model (considering the highest correlation between model and depths as the limit to derive depths) suggests that the bathymetric data are unexplored and losing potential information since the seafloor is still detectable depending on the water conditions. Vertical and horizontal segmentation of the scene might identify depths in quasi-optically deep water, previously defined as "optically-deep water" limiting bathymetric extraction.

In this work, six different segmentation approaches (highlighted in gray in Figure 11) are investigated. The are divided into two broad categories, *horizontal* and *vertical* segmentation methods.

- Horizontal segmentation
  - **Regular segmentation**
  - Irregular segmentation:
    - **Global-Local**
    - **Depth areas**
- **Vertical segmentation** and
- Merged segmentation
  - **Global-Local by Depth Range (GLDR)**
  - **Depth Areas by Depth Range (DADR)**

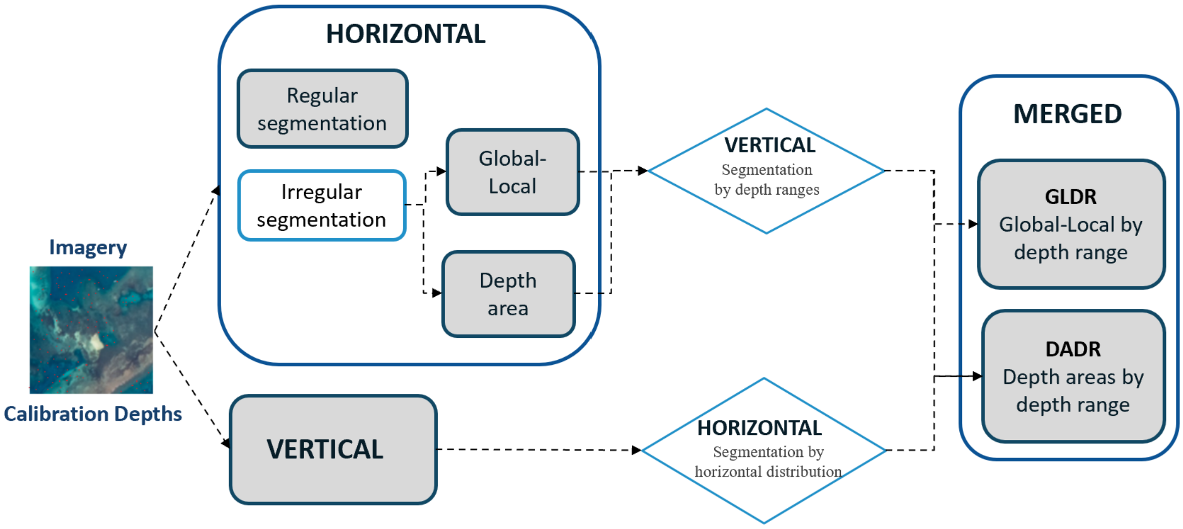


Figure 11 - Flowchart to represent the geographic approaches. The rounded square represents the geographic approaches applied, and the diamond represents the procedure involved for the merged approach.

4.2.1 Horizontal Segmentation

Dividing the scene horizontally is an effort to account for a certain level of heterogeneity due to the horizontal distribution of phenomena. Regular and irregular segmentation methods are investigated.

4.2.1.1 Regular Segmentation

This method divides the image scene into regular cells and provides the SDB solution within each cell. Two arbitrarily selected grid cell sizes are tested, one of 30 × 30 pixels and one 150 × 150 pixels. Initially, the image is divided into the smaller square grids of 30 × 30 pixels and if a solution of the model cannot be derived due to insufficient control points within small grid cells, they are merged into larger grid cells of 150 × 150 pixels.

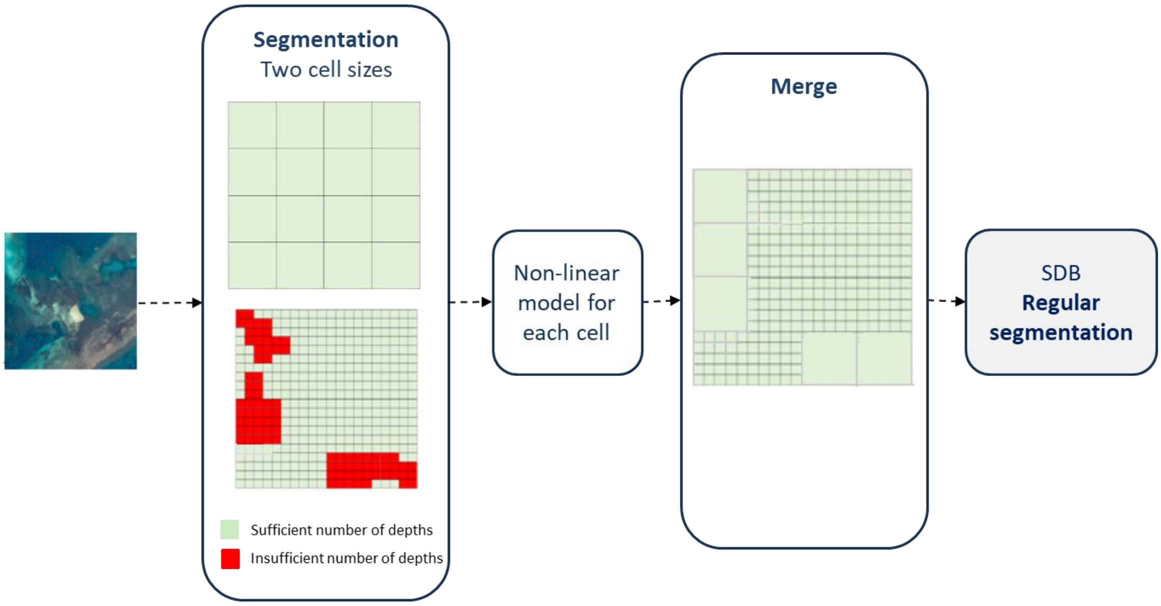

Segmentation
Two cell sizes
🟩 Sufficient number of depths
🟥 Insufficient number of depths
Non-linear model for each cell
Merge
SDB
Regular segmentation

Figure 12 - Workflow of geographic approach based on the regular segmentation.

4.2.1.2 Irregular Segmentation

4.2.1.2.1 Global-Local

The method begins with calibrating a single extended Dierssen model ("Global"), similarly to conventional approaches. Accordingly, the prediction residuals are calculated for the Global model (i.e., the difference between the Global model (z_{global}) and the referenced depths (z_{ref}) as shown in Equation 26) and the area is subdivided into classes/regions for the residual and depths ("Local").

$$Residual = z_{ref} - z_{global} \qquad (26)$$

A model is calibrated for each class using the training data points within each area. Figure 13 shows a workflow of the procedure implemented in the irregular horizontal approach by Global-Local, which is summarized as follows:

1. *Global* model: calibration of a single extended Dierssen model for the entire scene.
2. Regions are created based on two variables: depths and the prediction of residuals from the global model.
3. *Local* model: calibration of the extended Dierssen model for each segment.

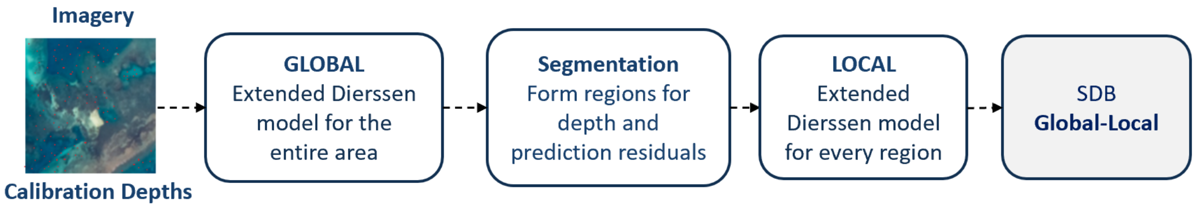

Imagery
Calibration Depths
GLOBAL
Extended Dierssen model for the entire area
Segmentation
Form regions for depth and prediction residuals
LOCAL
Extended Dierssen model for every region
SDB
Global-Local

Figure 13 - Workflow of geographic approach based on the irregular segmentation – Global-Local segmentation.

4.2.1.2.2 Depth Areas

For the *Depth Areas* method, the image scene is segmented into depth areas of defined ranges and a model is calibrated within each depth area. When a small depth area does not contain the minimum necessary number of control points, it is merged with the same depth in a different location. For this work, depth areas are formed every 5m, i.e., 0-5m, 5-10m (Figure 14). Figure 15 summarizes the workflow.

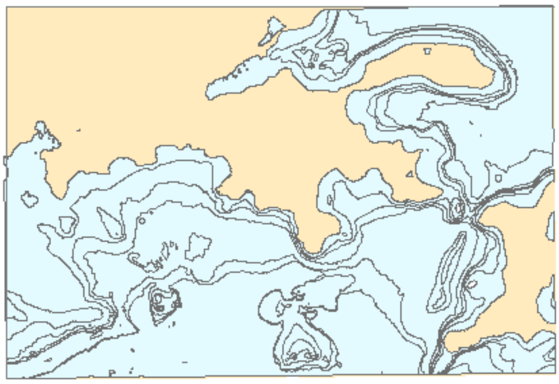


Figure 14 - Area segmented according to depth areas of 5 meters to calibrate the depth area approach.

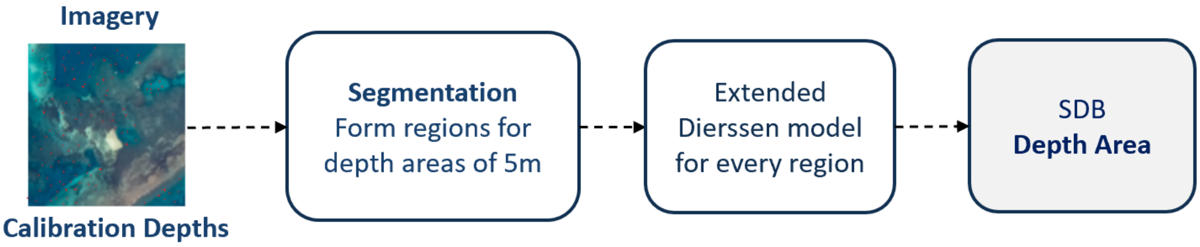


Figure 15 - Workflow of geographic approach based on the irregular segmentation – Depth Area.

4.2.2 Vertical Segmentation

Conventional approaches search for the best solution using a linear regression for the band-ratio/depth (Figure 16). However, this results in increased depth estimation uncertainty beyond the effective optical depth which is represented by the plot break (represented by a dashed pink line in Figure 16). Such approaches prevent deriving depth information beyond this point, i.e., from quasi-optically deep water. To address this limitation of a linear regression, the vertical segmentation approach applies a piecewise regression to the band-ratio/depth to improve the accuracy in depths below the effective optical depth and to attempt the extraction of bathymetry beyond this point. To accomplish the vertical segmentation, the dataset is divided by depth, beginning with depth ranges of 5m which progressively decrease to 1m, and, given that enough control points exist, a model is calibrated for each subset of training data points (Figure 17).

Based on the resulting RMSE, the best range selection provides the final bathymetry estimation. Figure 18 shows the workflow developed in this approach.

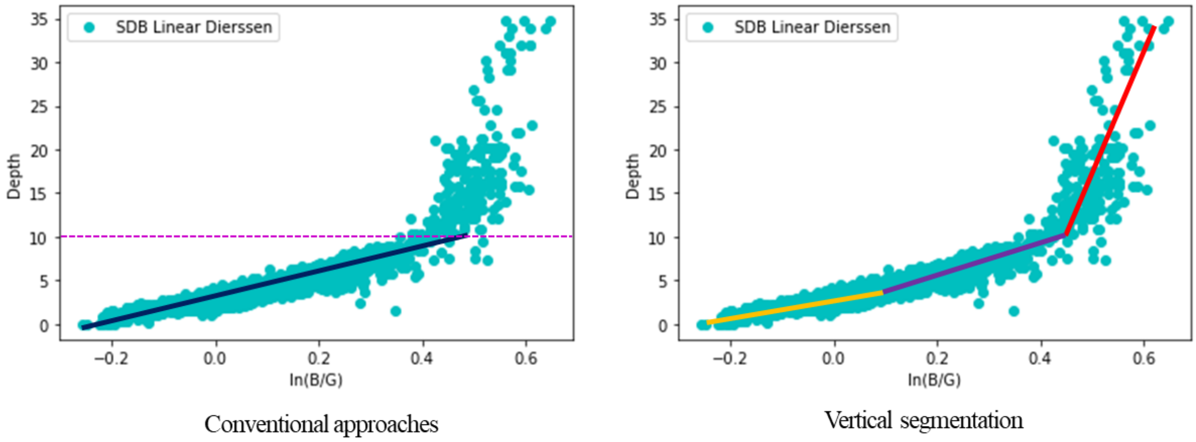


Figure 16 - Concept of conventional approach rather than the vertical segmentation. On the left, the linear regression for the entire area up to the effective optical depth, and on the right the solution is approximated with a piecewise function by depth ranges, even beyond the conventional method's "effective optical depth"

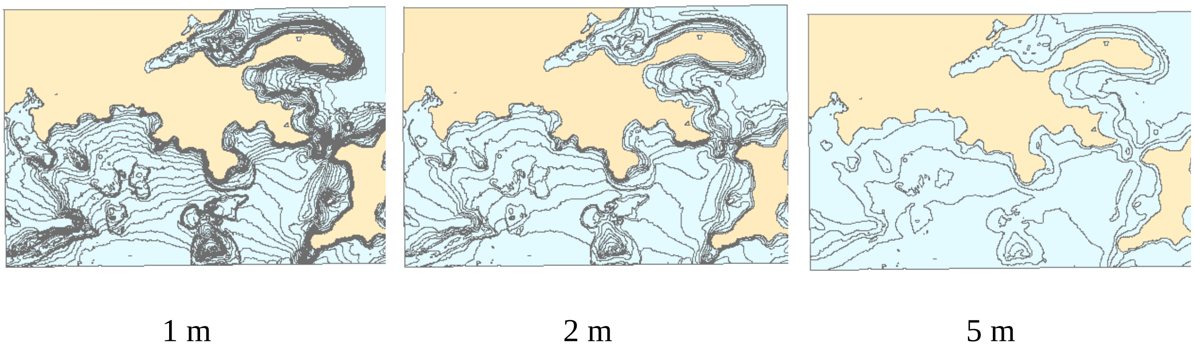


Figure 17 - Area segmented into three different depth ranges, every 1, 2 and 5 meters to calibrate the vertical approach.

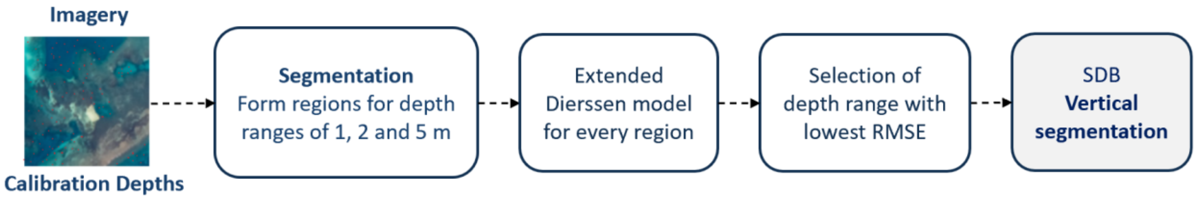

Imagery
Calibration Depths
Segmentation
Form regions for depth ranges of 1, 2 and 5 m
Extended Dierssen model for every region
Selection of depth range with lowest RMSE
SDB
Vertical
segmentation

Figure 18 - Workflow of geographic approach based on the irregular segmentation.

It should be highlighted that in a few cases, especially in deeper areas charted with fewer soundings, adjoined depth ranges are merged to complement the reference data required for the model performance.

4.2.3 Merged Segmentation

The merged segmentation is the combination of the horizontal and vertical segmentation methods. In detail, each spatial unit created with the above two horizontal segmentation approaches is divided vertically into depth ranges of 1 m. Thus, two merged approaches are performed:

- Global-Local by Depth Range (GLDR): spatial segments created in the *Global-Local* approach are vertically divided every 1 m.
- Depth Areas by Depth Range (DADR): spatial segments created in the *Depth Area* approach are divided into depth ranges every 1 m.

4.2.3.1 Depth Areas by Depth Range (DADR)

This method merges the *Depth Areas* and *Vertical Segmentation* approaches. In detail, after dividing the image scene into the depth areas, each depth area polygon is vertically segmented by depth.

It is noted that dividing depth areas into depth ranges of 1 m is the same as dividing the depth ranges of 1 m according to their horizontal distribution, which explains the arrows from two methods in the flowchart to generate the DADR approach (Figure 11).

Figure 19 illustrates the difference between *Depth Area*, *Vertical Segmentation* and DADR approaches. In the first one, depth areas of 5 m are formed, i.e., same depth range (0 – 5 m) creates more than one segments (three segments in Figure 19 - a). Figure 19 - b, the same depth range belongs to a unique segment, computing a single model for this region. In contrast, in the DADR, the same depth range is divided into subareas (depth areas), as for the Depth Area approach, however divided every 1 m. In Figure 19 – c, for example, there are three models instead of only one for a depth range of 1 m. Figure 20 shows the workflow of the geographic approach based on the merged DADR segmentation.

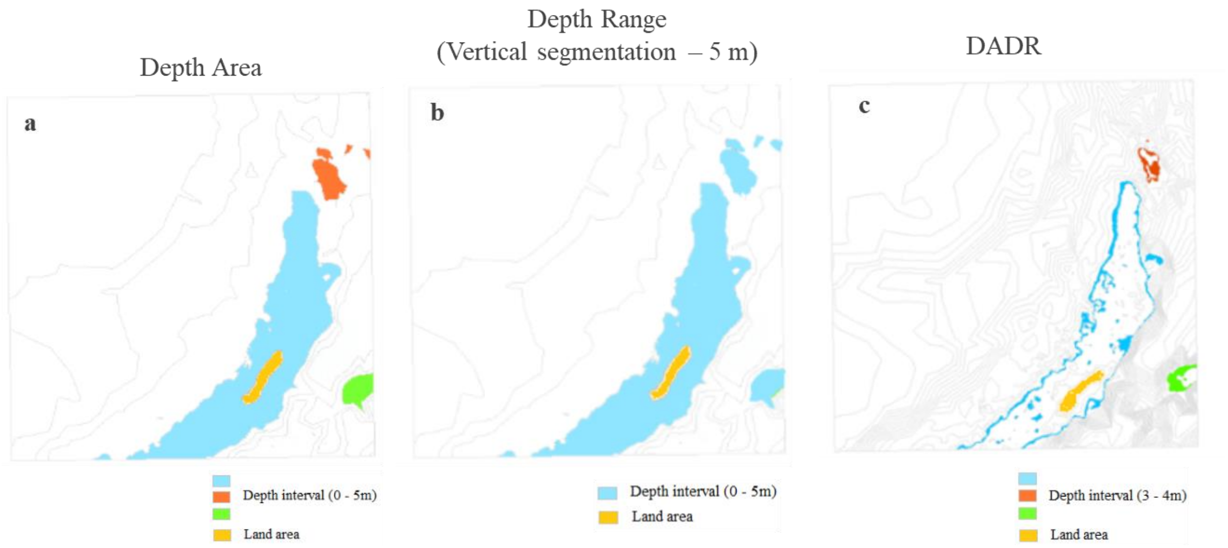


Figure 19 - Difference between Depth Area, Vertical segmentation and DADR approaches. a – three depth area of 5 m are formed, b - the area in blue represents a depth range of 5 m, and c - the same depth range is divided according to its horizontal distribution, forming three depth areas of 1m, in figure depth range of 3 – 4m.

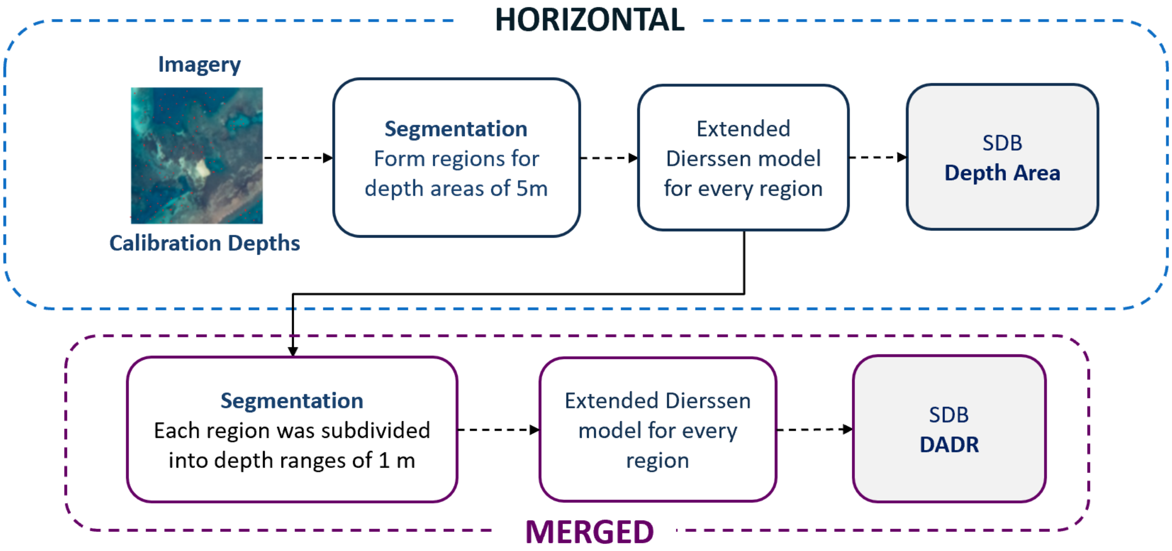
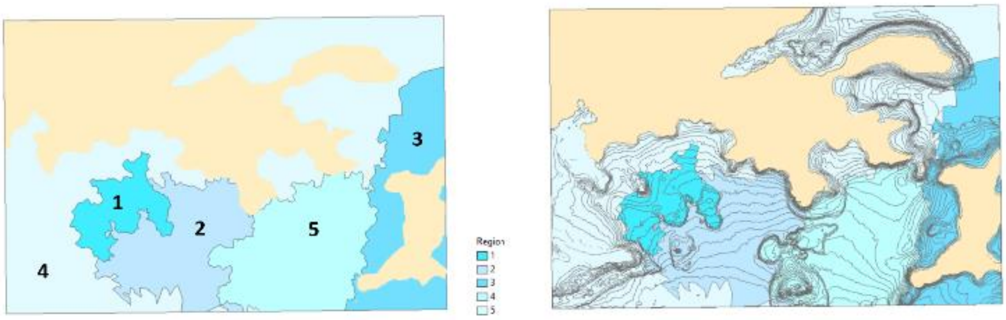

HORIZONTAL
Imagery
Calibration Depths
Segmentation
Form regions for depth areas of 5m
Extended Dierssen model for every region
SDB
Depth Area
Segmentation
Each region was subdivided into depth ranges of 1 m
Extended Dierssen model for every region
SDB
DADR
MERGED

Figure 20 - Workflow of geographic approach based on the merged DADR segmentation.

When the number of control points is insufficient (fewer than four) within any of the subareas, it is merged with the adjoining depth area to reach the required number of training depths.

4.2.3.2 Global-Local by Depth Range (GLDR)

Each class provided by the *Global-Local* approach is divided, vertically, in smaller regions, every 1m depth range. Instead of applying and calculating the model for classes created in the *Global-Local* horizontal segmentation, the solution in the GLDR is computed for a larger number of classes subdivided by depth intervals of 1 m (Figure 21). Figure 22 shows the workflow of the geographic approach based on the merged GLDR segmentation.



Global-Local

GLDR

Figure 21 – Example of segments created in the global-local approach (horizontal segmentation) (left) and in the GLDR (right).

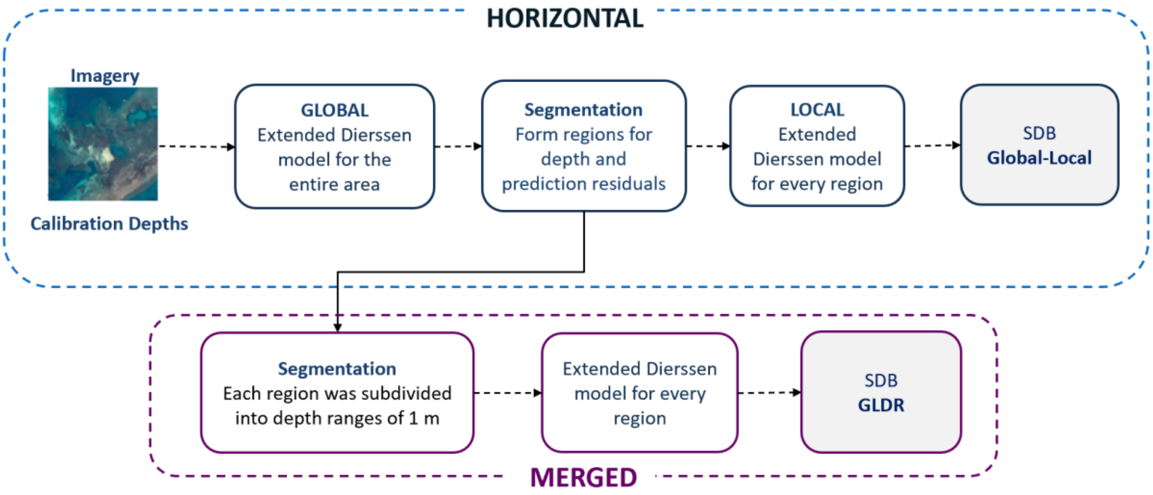

HORIZONTAL
Imagery
Calibration Depths
GLOBAL
Extended Dierssen model for the entire area
Segmentation
Form regions for depth and prediction residuals
LOCAL
Extended Dierssen model for every region
SDB
Global-Local
Segmentation
Each region was subdivided into depth ranges of 1 m
Extended Dierssen model for every region
SDB
GLDR
MERGED

Figure 22 - Workflow of geographic approach based on the merged GLDR segmentation.

4.3 Accuracy assessment

The accuracy of the geographic models for SDB is assessed by comparing estimated and reference bathymetry, applying the following metrics:

- Root Mean Square Error (RMSE) statistic: it is widely used among SDB to assess the quality of predictions. This metric indicates how the data is dispersed around the best-fit line and describes a standard deviation of residuals. Independent testing points are used for the RMSE calculation.
- CATZOC: the metric defined by IHO and used to classify electronic navigational chart accuracy, indicating whether data meets a set of criteria for the position, depth accuracy, and seafloor coverage. Five categories may represent CATZOC on a chart: A1 (best accurate data), A2, B, C, D (least accurate data), or U (quality data unassessed). In this study, CATZOC is assigned only with respect to vertical accuracy, computed at the 95% confidence level. With the assumption that the vertical error is normally distributed, a factor of 1.96 is applied to compute the vertical (depth) accuracy (FGDS, 1998):

  $$\text{Accuracy}_z = 1.96 * \text{RMSE}$$

- Comparison of the geographic models with conventional Stumpf and Dierssen models, both calibrated as a single linear model for the entire image scene.

CHAPTER 5 : STUDY AREAS, TESTS AND RESULTS

5.1 Study area

For the evaluation of the proposed methods, two study areas were selected: Dry Tortugas (study site 1), west of Key West between the Gulf of Mexico and the Atlantic Ocean, and St. Thomas East End Reserve (STEER), U.S. Virgin Islands (study site 2). The locations were chosen based on their clear water conditions, medium to low turbidity levels, and complex aquatic environments, while, for both regions, LiDAR and chart data were freely available from a NOAA website.

5.1.1 Study Site 1 - Dry Tortugas

The Dry Tortugas is a small archipelago of coral islands in South Florida, consisting of seven islands (NPS, 2014). The specific area under investigation is Loggerhead Key (Figure 23), the largest island of the complex (NPS, 2014). Loggerhead Key is approximately 1,430 m (4,690 ft) long by 200 m (650 ft) wide, and its highest point is 3 m (10 ft) above sea level on the northeastern tip of the island (Ginsburg, 1953). Loggerhead Key is considered a stable island that suffers from only minimal erosion due to the protection afforded by the extensive beach rock deposits, unlike the other islands in Dry Tortugas that compose mainly of carbonate sand and smaller amounts of coral rubble (NPS, 2014).

Figure 24 illustrates a benthic map of the area published in 2014 by the National Park Service (NPS), Natural Resource Stewardship and Science. This map shows a large concentration of aggregated path reef (APR) southwest of Loggerhead Key, surrounded by sand (US). In the northwest, a large variety of benthic habitats is notable, settling the largest area of reef terrace (RT) in the park. Between Loggerhead Key and the reef terrace is a large remnant (RM) area and low relief spur and groove (LRSG) (NPS, 2014).

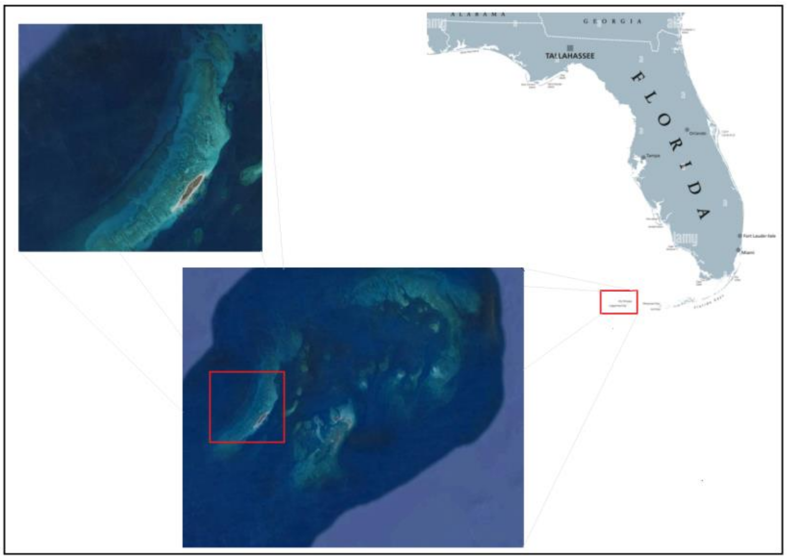


Figure 23 - The geographic location of study site 1: Loggerhead Key Island, Dry Tortugas.

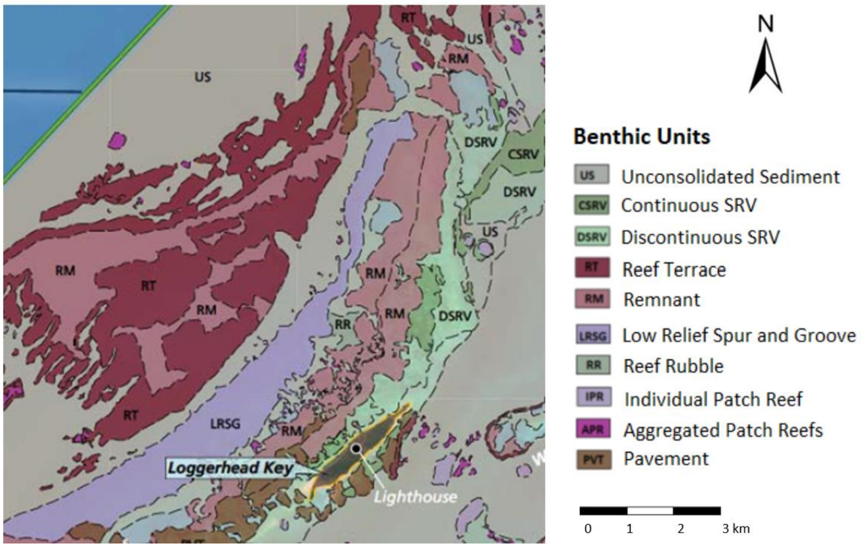


Figure 24 - Benthic Map of Loggerhead Key Island (NPS, 2014).

5.1.2 Study Site 2 – St. Thomas East End Reserve

The STEER, located on the southeastern end of the island of St. Thomas in the U. S. Virgin Islands, is a marine protected area (MPA) with a total area of approximately 9.6 km^2 (Figure 25). This area has been the focus of studies for providing a management plan to the community to guide conservation and sustainable use activities (Costa *et al.*, 2013). The benthic habitat map illustrated in Figure 26 was developed by NOAA in 2013 and represents some of the products provided to support the management plan in the STEER.

The STEER represents a complex environment covered by unconsolidated sediments (74.3%), mainly categorized by rhodoliths and sand. Coral reef and hardbottom represent the rest of the seafloor, mainly covered by pavement structure type (Costa *et al*., 2013). Concerning biological cover (Figure 26), sand habitats are dominated by continuous seagrass beds, comprising more than 20% of the mapped area inside the STEER's boundary. Hardbottom habitats are primarily colonized by algae (Costa *et al*., 2013).

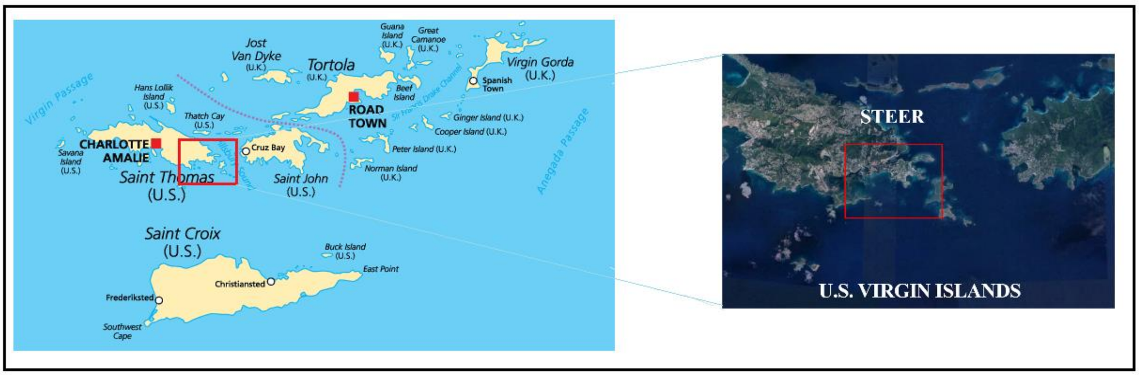


Figure 25 - The geographic location of study site St. Thomas East End Reserve (STEER), U.S. Virgin Islands.

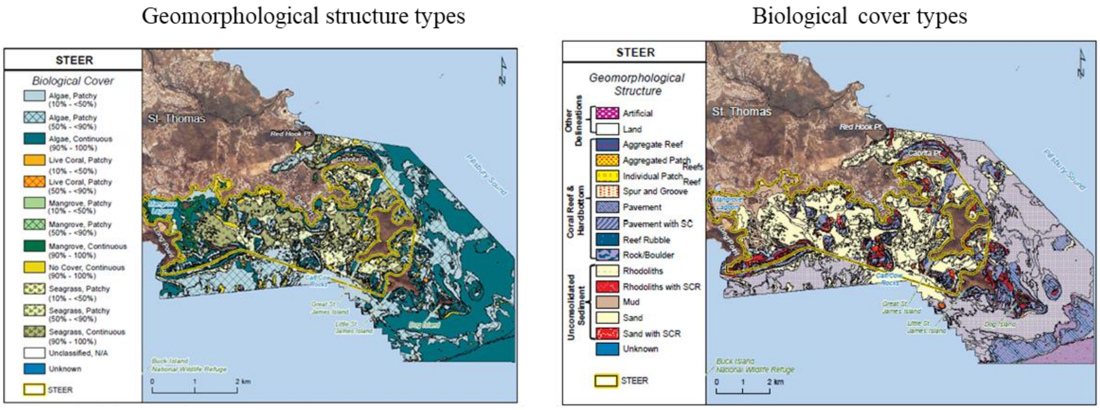

Geomorphological structure types
Biological cover types
STEER
Biological Cover
Algae, Patchy (10% - <50%)
Algae, Patchy (50% - <90%)
Algae, Continuous (90% - 100%)
Live Coral, Patchy (10% - <50%)
Live Coral, Patchy (50% - <90%)
Mangrove, Patchy (10% - <50%)
Mangrove, Patchy (50% - <90%)
Mangrove, Continuous (90% - 100%)
No Cover, Continuous (90% - 100%)
Seagrass, Patchy (10% - <50%)
Seagrass, Patchy (50% - <90%)
Seagrass, Continuous (90% - 100%)
Unclassified, N/A
Unknown
STEER
St. Thomas
Red Hook Pt
Pillsbury Sound
0 1 2 km
Geomorphological Structure
Other Delineations
Artificial
Land
Coral Reef & Hardbottom
Aggregate Reef
Aggregated Patch Reefs
Individual Patch Reef
Spur and Groove
Pavement
Pavement with SC
Reef Rubble
Rock/Boulder
Unconsolidated Sediment
Rhodoliths
Rhodoliths with SCR
Mud
Sand
Sand with SCR
Unknown
STEER

Figure 26 - Geomorphological structure types (left) and biological cover types (right) in the STEER (Costa *et al*., 2013).

5.2 Datasets

5.2.1 Satellite images

After the selection of the study areas, the criteria adopted to choose the image was low cloud coverage (<10%) and the time span between the bathymetry data and the image. In the STEER, the efforts also focused on finding images taken before Hurricane Dorian (September 2019) since intense resuspension and currents may have modified shallow seafloor morphology.

This study used images captured by Sentinel-2 Multispectral Instrument (MSI), acquired from the United States Geological Survey (USGS) Earth Explorer, which currently provides a freely available multispectral dataset. The Sentinel-2 mission is managed by the European Space Agency (ESA) and provides a high spatial resolution (four bands at 10 m, six bands at 20 m, and three bands at 60 m) multispectral imagery. Each Sentinel mission comprises a constellation of two satellites, Sentinel-2A, launched in June 2015, and Sentinel-2B, launched in March 2017. Sharing the same orbital plane, but separated by 180° degrees, Sentinel-2A and Sentinel-2B cross in the equator at 10:30 AM local time and have a collective repeat cycle of 5 days (10 days individually) (Forfinski-Sarkozi, 2019).

The Sentinel-2 MSI acquires data in 13 spectral bands ranging from the visible and near infra-red to the shortwave infrared parts of the electromagnetic spectrum. For this work, blue (490nm) and green (560nm) bands were utilized; however, in searching for higher correlation, other combinations were also investigated, e.g., coastal-blue and red bands.

Sentinel-2 products are available in Level-1C (L1C) and Level-2A (L2A), compiled in a minimum indivisible product partition denominated granule. The granules, also called tiles, are 100 x 100 km^2 ortho-images in UTM/WGS84 projection. The acquired packages used in this study were Sentinel-2 L1C - top of atmosphere (TOA) products, i.e., data were radiometrically and geometrically corrected. To derive bathymetry utilizing the information from the image, TOA must be processed to the bottom of atmosphere (BOA), which represents Level-2A. This step was performed using ACOLITE, described in Section 4.1.1. Table 2 provides the description of the dataset used for the two study sites.

5.2.2 Depth datasets

For the Dry Tortugas lidar and chart soundings are utilized. The topographic and bathymetric airborne lidar bathymetry (ALB) was collected by the National Geodetic Survey (NGS) in 2015, covering shallow waters from 0 up to 8.8 m. A Riegl VQ-820-G sensor, providing high-resolution surveying of the bottom of shallow waters, was used. The products include a 5 m resolution BAG/CSAR surface, referenced to Mean Lower Low Water (MLLW) (NOAA, 2017). The high-resolution lidar observations are used as the reference bathymetry to train and test the model. Also, charted depths from ENC US5FL91M are used as complementary data for the areas unserved by ALB, dated up to 2015. In Dry Tortugas, 6338 points were randomly selected over the entire area (6203 from ALB data and the total number of available chart depth beyond 8.8 m – 135 depth points), while in the STEER, 7644 ALB data were selected. The independence between training and test data was provided by the random selection of 70% of the points for training while the remaining 30% for testing the model performance. The selection was performed using the using the *create random points* in ArcGIS.

In the STEER area, NOAA NGS collected ALB data from January 20, 2019, to June 2, 2019 using a Riegl VQ-880-G II sensor. The horizontal and vertical datum is NAD83 (2011) and, coordinate system UTM Zone 20 N. As the ALB fully covers the area under investigation, no additional chart depth is required.

Table 2 summarizes the datasets used for the two study sites.

Table 2 – Study sites dataset specifications.

Data	**Description**	**Dry Tortugas, FL**	**STEER, VI**
Sentinel-2	ID	L1C_T17RLH_A008531_20170208T160849	L1C_T20QKF_A018626_20190115T145721
	Acquisition date	08/02/2017	15/01/2019
	Cloud Cover (%)	0.0	0.0

	Coordinate system	UTM 17N	UTM 20N
	Horizontal datum	WGS84 (G1762)	WGS84 (G1762)
	Grid resolution	10m	10m
ALB	Collecting agency	NOAA	NOAA
	Date of survey	03/14/2015-03/23/2015	01/20/2019-06/02/2019
	Equipment	Riegl VQ-820-G	Riegl VQ-880-G II
	Horizontal Datum	NAD83 (2011)	NAD83 (2011)
	Vertical Datum	MLLW	NAD83 (2011)
	Depth range	0 – 8.8m	0 – 20m
	Number of control points	6203	7644
Chart depth	Surveys year	1850 - 2015	-
	Horizontal Datum	WGS84 (2011)	-
	Vertical Datum	MLLW	-
	Depth range	0 – 30m	-
	Number of control points	135	-

5.3 Band and SDB model selection

To select the best band configuration for retrieving depths from satellite images, different scenarios were considered. The analysis was carried out in both study sites using the conventional global approach by correlating four band combinations in the log-difference model. In the visible spectrum, blue and green wavelengths are typically used as they reach the deepest depths (Su *et al*., 2008). Nonetheless, the depth of light penetration is dependent on the environmental conditions, such as turbidity, which reduces light penetration. Traditional optimization methods usually disregard the red band in the pair combination, however depending on the water quality, red light may provide additional information in shallower regions of 6 – 8 m (Caballero *et al*., 2019), while blue coastal is the least absorbed (in clearest waters, Jerlov, 1976) and it may provide a closer inspection of the coastal and inland waters depending on the water quality, useful for imaging shallow water (Staff, 2013). Blue coastal and red bands ratios were tested through the log difference to extract bathymetry, but, blue and green performed best.

In Dry Tortugas, all band ratio combinations presented the same effective optical depth of 10 m (illustrated by the red line in Figure 27). However, coastal blue-green and blue-green showed a likely depth extraction beyond 10 m (represented by quasi-optically deep waters), while blue-red and green-red showed a negative correlation between model and depth, preventing bathymetric extraction. Unlike Dry Tortugas, in STEER the effective optical depth depends on the band-ratio. Blue-red and green-red bands could recover depths only up to 5 m, while for the blue-green and blue coastal-green bands, the effective optical depth was 20 m. Among the band combinations, in both study sites, the best configuration was blue and green bands, with RMSE of 1.30 (Dry Tortugas) and 1.29 m (STEER) (Table 3).

Table 3 – RMSE and effective optical depths from band-ratio combinations investigated in Dry Tortugas and STEER.

		Dry Tortugas		**STEER**	
Band-ratio combination	**Application of ACOLITE**	**Effective optical depth (m)**	**RMSE (m)**	**Effective optical depth (m)**	**RMSE (m)**
Blue Coastal/ Green	✓	10	1.76	20	2.16
Blue/Red	✓	10	2.17	5	4.68
Green/Red	✓	10	2.47	5	5.46
Blue/Green	✗	10	1.49	20	1.73
Blue/Green	✓	10	1.30	20	1.29

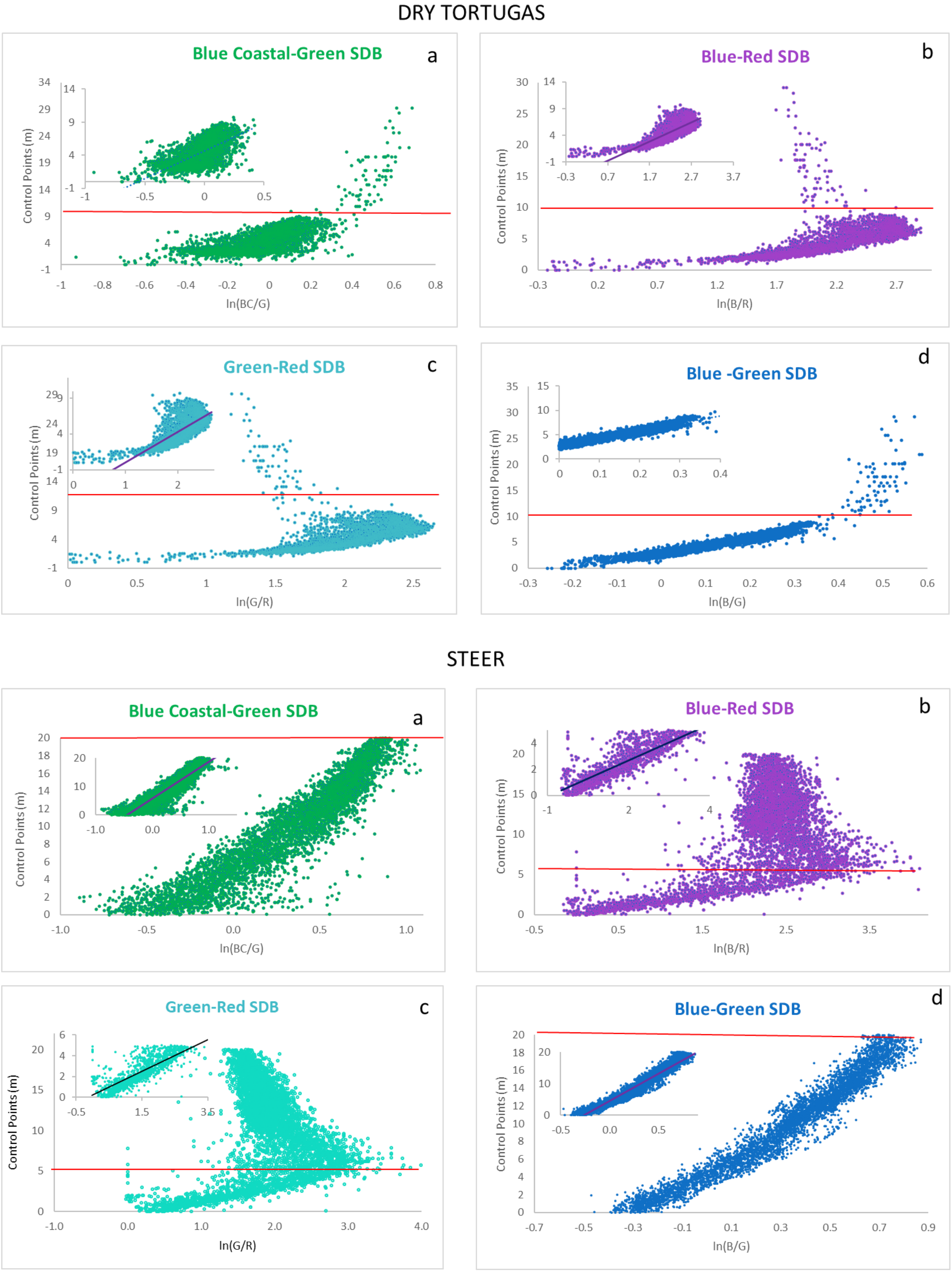


Figure 27 - Scatter plots of estimated SDB using blue coastal-green (a), blue-red (b), green-red (c), and blue-green (d) band ratios against reference bathymetry in Dry Tortugas and STEER. The horizontal red line represents the effective optical depth.

In addition, this study confirmed the findings of other works (e.g., Cabellero *et al*., 1019; Cabellero *et al*., 2020) that ACOLITE is a robust and consistent atmospheric correction method for Sentinel-2, as it increased the accuracy in both locations. In Dry Tortugas, an improvement of 12.7% was observed (RMSE = 1.30 m) compared to the same band ratio without atmospheric correction (RMSE of 1.49 m). Similarly, in the STEER, ACOLITE improved the RMSE to 1.29 m instead of RMSE 1.73 m without the correction.

Accordingly, an extended Dierssen model was applied. In Dry Tortugas, the extended Dierssen model improved the accuracy by 31.5%, while in the STEER, by 10.8% relative to the simplified Dierssen model (Table 4). Figure 28 represents the scatter plots of estimated SDB using the blue-green Dierssen and extended Dierssen models versus reference depth points in both study sites.

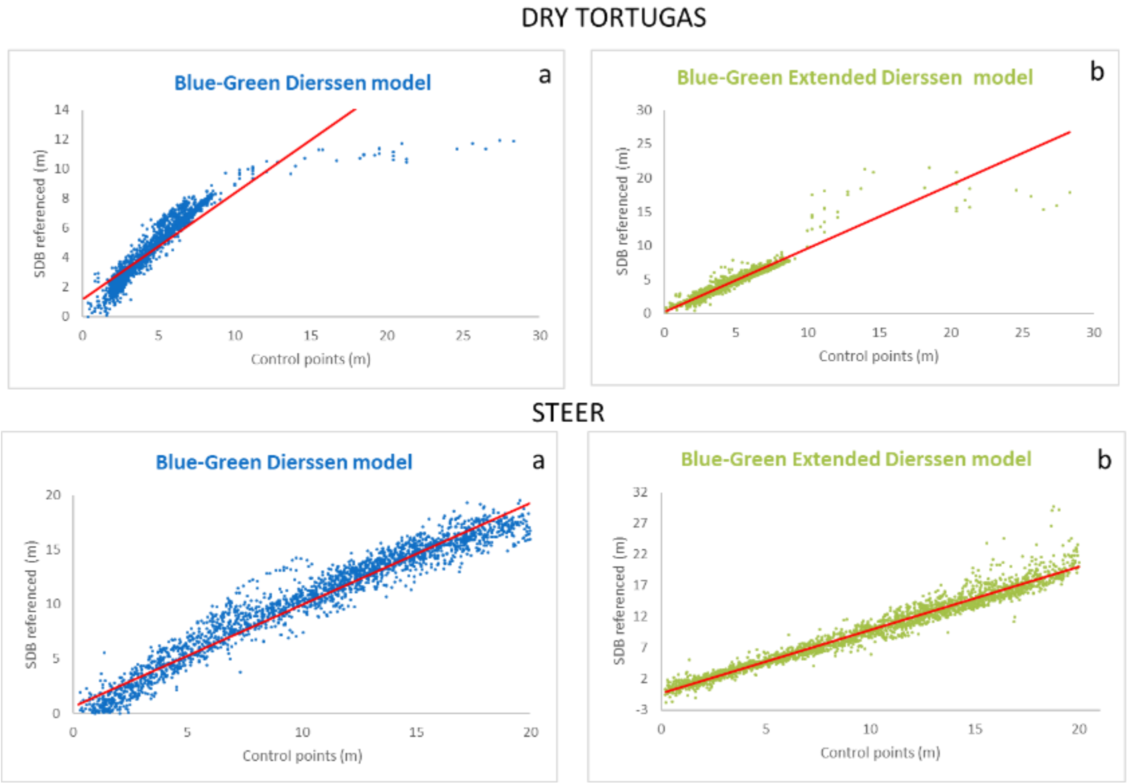


Figure 28 - Scatter plots of SDB referenced blue-green bands using Dierssen model (a) and blue-green bands using extended Dierssen model (b) against reference depth measurements in Dry Tortugas and STEER.

Table 4 - RMSE from Dierssen and extended Dierssen models investigated in Dry Tortugas and STEER.

Model		**RMSE (m)**	
Band-ratio	**Model**	**Dry Tortugas**	**STEER**
Blue/green	Dierssen	1.30	1.29
	Extended Dierssen	0.89	1.15

Based on the results in Table 3 and 4, the blue-green band-ratio with the extended Dierssen model with ACOLITE performed best, thus is used hereafter for the evaluation of the geographic models.

5.4 Results

By subdividing the image scene into multiple small segments and calibrating the extended Dierssen model within each subarea, new depth estimations (SDB solution) were calculated. In this section, the performance of geographic models is compared with the conventional global model for the entire scene, based on the aggregated RMSE, using the independent points from reference depth datasets (see Section 5.2.2), and classified according to the CATZOC. Furthermore, the accuracy of the derived bathymetry every 5 meters of depth is also investigated, as this may reveal a pattern and direction of errors in the estimation. Subsequently, each geographic model is analyzed individually, discussing their limitations and benefits. Finally, the geographic model that yields the best estimation results is evaluated for its performance using both the Dierssen and extended Dierssen equations for the model calibration, and compared with the most traditional Stumpf *et al.* (2003) model.

5.4.1 Summary of Geographic vs Conventional Models

The RMSE presented in Table 5 represents the error between the estimated SDB and the reference bathymetry dataset, calculated for depth ranges of 5 m and the overall result in both study areas. The green numbers represent results better than the conventional model, red numbers when was worst than conventional approach, and in blue the best RMSE. Based on the achieved accuracy, the respective CATZOC for TVU at 10m and 20m has been defined, at 95% CI, and the coefficient of determination (R^2). The number of segments created in each approach is treated as Unit.

Table 5 - Comparison between geographic approaches according to RMSE and CATZOC in the STEER and Dry Tortugas

		Approach	0-5	5-10	10-15	15-20	20-25	25-30	RMSE (10 m)	RMSE (Overall)	Unit	ZOC 10m	ZOC 20m	R^2
Dry Tortugas	Horizontal	Regular	0.56	0.62	2.48	5.07	4.94	5.05	0.58	0.83	400	A2/B	C	0.91
		DA	0.38	0.49	1.38	1.41	1.55	3.75	0.42	0.47	10	A2/B	A2/B	0.95
		Global-Local	0.41	0.46	1.14	3.28	7.5	2.87	0.43	0.72	5	A2/B	A2/B	0.93
	Vertical	1m	0.29	0.28	0.49	0.65	0.88	1.42	0.29	0.30	20	A1	A1	0.99
	Merged	DADR	0.29	0.29	0.49	0.65	0.88	1.42	0.29	0.30	32	A1	A1	0.99
		GLDR	0.29	0.31	0.78	1.10	3.13	2.87	0.30	0.40	28	A1	A1	0.98
	Conventional	Entire scene	0.49	0.55	5.04	7.18	10.2	16	0.51	1.30	1	A2/B	A2/B	0.88
STEER	Horizontal	Regular	0.81	0.63	0.93	1.16	-	-	-	0.90	122	C	C	0.97
		DA	0.57	0.54	0.68	1.04	-	-	-	0.72	16	C	C	0.98
		Global-Local	0.65	0.88	0.99	1.44	-	-	-	1.02	5	C	C	0.96
	Vertical	1m	0.41	0.33	0.32	0.36	-	-	-	0.35	20	A2/B	A1	0.99
	Merged	DADR	0.34	0.33	0.32	0.37	-	-	-	0.34	55	A2/B	A1	0.99
		GLDR	0.37	0.52	0.34	1.01	-	-	-	0.60	78	A2/B	A2/B	0.98
	Conventional	Entire scene	1.13	1.54	0.95	1.54	-	-	-	1.29	1	C	C	0.94

Following the procedures applied in the conventional approach, the optical depth limit for inferring bathymetry in Dry Tortugas is 10 m, while the geographic models were able to estimate bathymetry beyond this depth. For a fair comparison between methods, the comparison between the geographic model and the conventional methods is limited to optically shallow waters, i.e., 10 m.

In the geographic models, the calculated RMSE ranges from ‑0.29 to 0.58 m in Dry Tortugas and from 0.34 m to 1.02 m in the STEER. As would generally be expected, the R^2 coefficient of determination is strongly related to the magnitude of the RMSE, i.e., a lower correlation coefficient is associated with greater error. Among all approaches, for both study sites, the *vertical segmentation* (depth range of 1 m) and DADR performed the best in terms of RMSE (0.29 m in Dry Tortugas and 0.34 in the STEER) and R^2, improving the overall accuracy by 43.1% in Dry Tortugas and 73.64% in the STEER compared to the conventional global method up to the effective optical depth. On the other hand, the *regular segmentation* performed worst among the geographic approaches in Dry Tortugas (RMSE = 0.58 m), whereas the *Global-Local* approach performed worst in the STEER (RMSE = 1.02 m). The *Depth Areas* approach, presented the best performance among horizontal segmentation approaches, reinforcing the RMSE reduction performed by vertical data division. The accuracy of bathymetric estimates with the geographic models is improved in all depth ranges compared to the conventional global approach, with the exception of the 0 – 10m regular segmentation in Dry Tortugas, and between 10 – 15 Global-Local in the STEER. In addition to the overall accuracy, a significantly reduced error in depth estimates for depths beyond the conventional method’s effective optical depth is noticed with the geographic methods.

Besides wide data availability to calibrate the model and better estimate depths, in terms of benthic habitats, the STEER area presents greater bottom heterogeneity than that found in Dry Tortugas (illustrated in Figure 24 and Figure 26 of Section 5.1). Comparing the depth estimation up to 20 m for both locations, the best geographic model provided an accuracy improvement of 73.6% in the STEER (0.34m vs 1.29m of the conventional global) and 52.4% (0.29m vs 0.61m for the conventional approach) in Dry Tortugas. This suggests that areas of greater bottom heterogeneity are better estimated when the dataset is divided into small spatial units, however this is subject to further research.

Figure 29 shows an accuracy analysis from each method in Dry Tortugas computed every 2 m up to 20 m and every 5 m beyond the 20m, (due to the fact that fewer reference points being available, splitting every 2m was not possible). The increase of error in greater depths for the conventional and horizontal segmentations indicates that the model generally performs well in the optically shallow waters (between 0 to 10 m), but more deviations tend to occur in the quasi-optically deep waters (beyond 10 m). Nonetheless, it is evident that for the vertical and merged approaches, the model can estimate depths beyond 10 m with accuracy comparable to that in shallow waters. Thus, the piecewise linear regression implemented with the vertical and merged approaches improves the coverage of SDB as it allows estimation of depths beyond the depth that is considered the limit for the conventional global approaches.

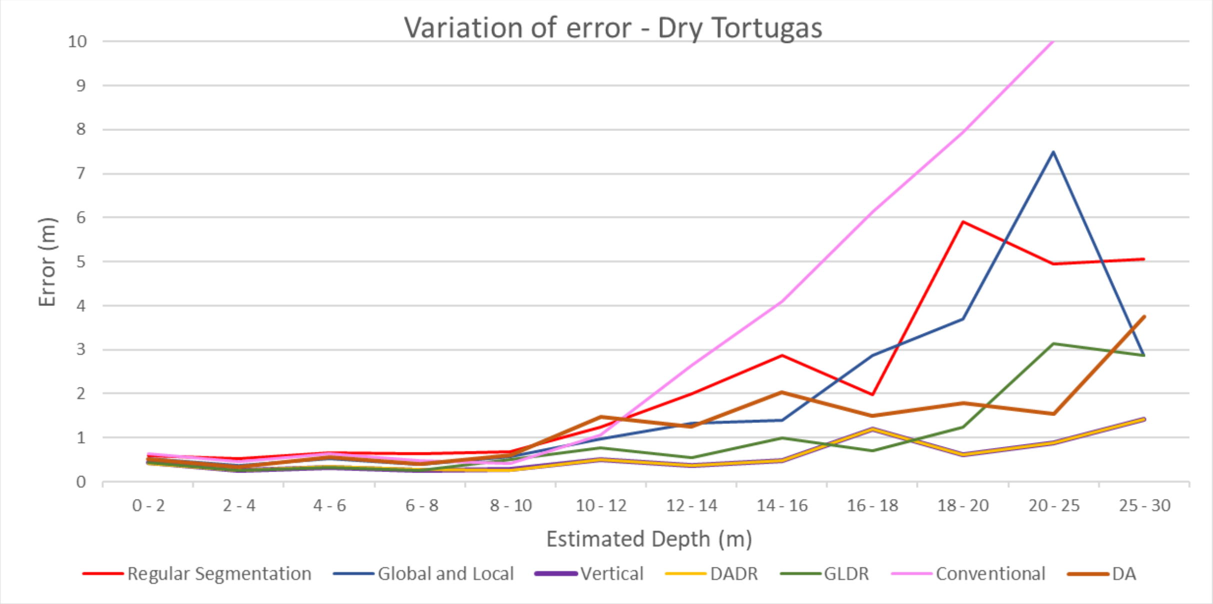


Figure 29 - Performance of geographic models and conventional SDB approach in Dry Tortugas. The error variation is represented in depth ranges of 2 m

In the STEER (Figure 30) the conventional and *horizontal segmentation* methods displayed an overall accuracy increasing trend, whereas the error of the GLDR approach fluctuated around 0.5 m (with an increase trend for depths deeper than 16 m)). The DADR and *vertical segmentation* displayed similar performance, with RMSEs constant around 0.35 m for the entire area (thus, demonstrating the ability to estimate depths).

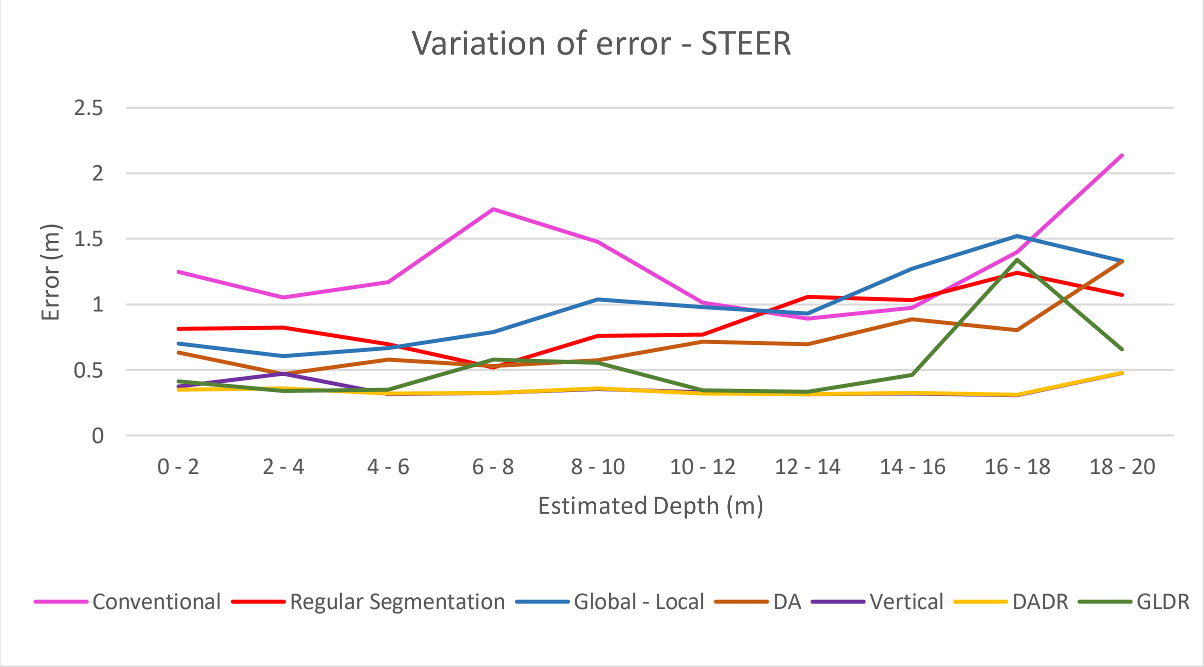


Figure 30 - Variation of error of geographic models and conventional SDB approaches in the STEER.

In the following sub-sections, a detailed analysis of the geographic methods is performed.

5.4.2 Horizontal Geographic

5.4.2.1 Regular Segmentation

Analyzing the RMSE in the *regular segmentation* (Table 6), the error in Dry Tortugas significantly increases beyond 10 meters, which represents the effective optical depth of the conventional approach. On the other hand, in the STEER area, the error is more consistent with traditional SDB results, i.e., reducing accuracy with deeper bottom and very shallow water. The reason is that *regular segmentation* is highly dependent on the number of points available within each grid. When a small or insufficient number of depth reference points are within the small cell (30x30 pixels), the larger size grid cell is used (150x150 pixels) (as explained in Section 4.2.1.1), a model less representative of sea-bottom variances is calibrated for the bigger grid cells. Characteristically, in Dry Tortugas, the studied area included sparse nautical charts depths in deeper areas; thus a large number of big grid cells were used for the model calibration, while only 26% of the 400 small grid cells were used. On the other hand, for the STEER area, covered by dense ALB reference depth points, a large proportion (77%) of the smaller grids were used for the model calibration.

The final bathymetric map by regular segmentation after combining two SDB cell sizes is shown in Figure 31 (Dry Tortugas) and Figure 32 (STEER).

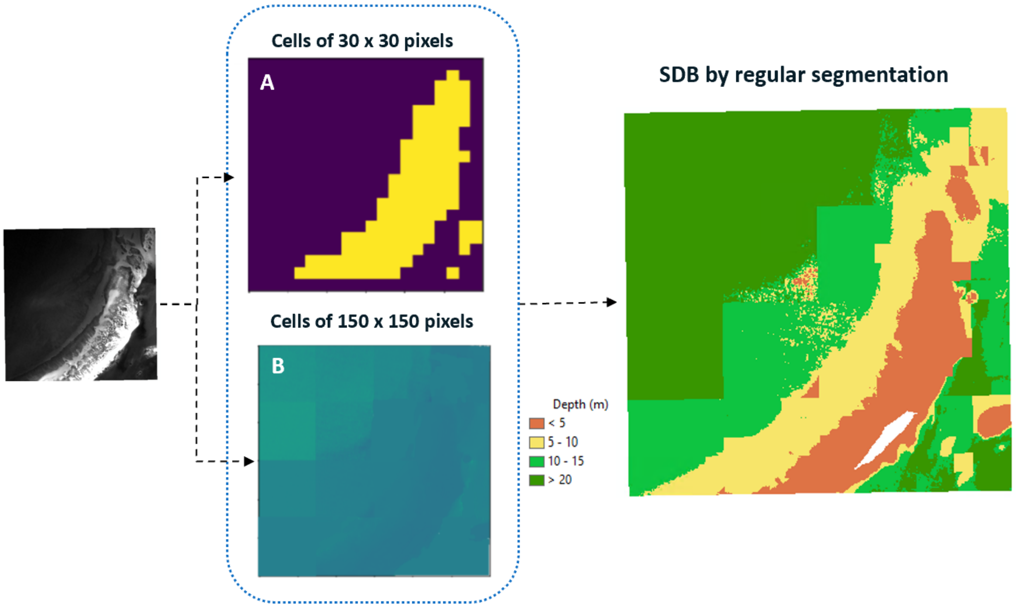


Figure 31 - SDB surface from the regular segmentation of 150 x 150 pixels cells (A), SDB surface of 30 x 30 pixels cells (B) and the final bathymetry produced by regular segmentation in Dry Tortugas.

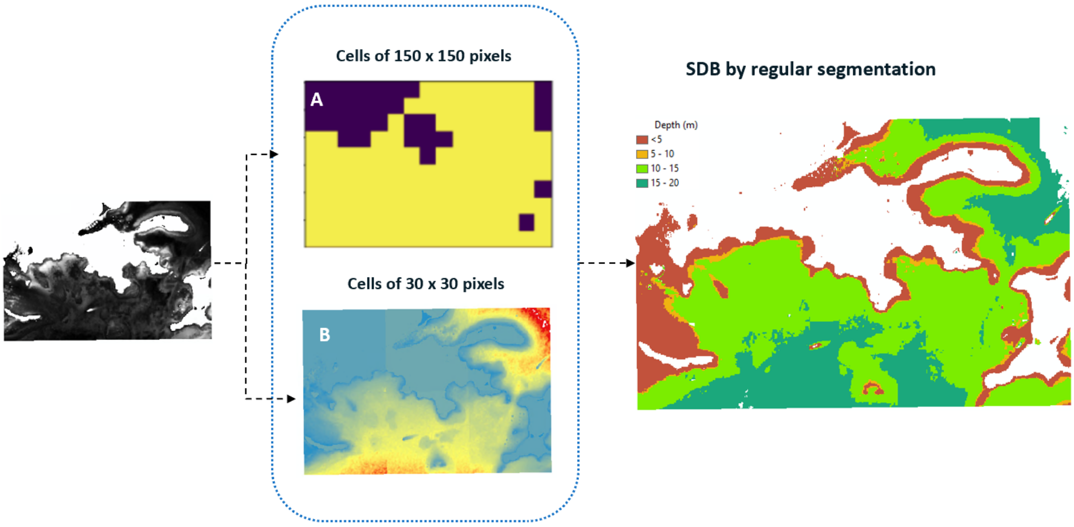


Figure 32 - SDB surface from the regular segmentation of 150 x 150 pixels cells (A), SDB surface of 30 x 30 pixels cells (B) and the final bathymetry produced by regular segmentation in the STEER.

Table 6 - RMSE of the estimated depths from regular segmentation approach in STEER and Dry Tortugas.

RMSE (m)	**Approach**	**0–5**	**5–10**	**10–15**	**15–20**	**20-25**	**25-30**	**Overall**
Dry Tortugas	Regular Segmentation	0.56	0.62	2.48	5.07	4.94	5.05	0.83
STEER		0.81	0.63	0.93	1.16	-	-	0.90

5.4.2.2 Depth areas

The segmentation by *Depth Areas* aims to account for the variations of the sea bottom, considering five meters of depth range. In Dry Tortugas, 10 regions were created, resulting in an overall RMSE of 0.47 m. In the STEER, despite the smaller depth range (0 - 20m), a greater number of segments were created (16) resulting in an overall performance of RMSE = 0.72 m (Table 7).

Table 7 – RMSE of the estimated depths from irregular segmentation approach – DA in STEER and Dry Tortugas.

RMSE (m)	**Approach**	**0–5**	**5–10**	**10–15**	**15–20**	**20-25**	**25-30**	**Overall**
Dry Tortugas	Irregular segmentation DA	0.38	0.49	1.38	1.41	1.55	3.75	0.47
STEER		0.57	0.54	0.68	1.04	-	-	0.72

5.4.2.3 Global-Local

For the *Global-Local*, first, a standard conventional global extended Dierssen model was calibrated for the entire scene using blue-green bands and the corresponding depth points in the calibration dataset. Figure 33 shows the parameters used to predict bottom depths using four parameters in both study sites. In Dry Tortugas, $m_1 = 9.1332$, $L_{w1} = 0.0269$, $L_{w2} = 0.0271$ and $m_0 = 2.8871$, while in the STEER $m_1 = 9.8015$, $L_{w1} = 0.008$, $L_{w2} = 0.0114$ and $m_0 = 3.1398$. Applying these parameters for all image pixels provides the global SDB approach.

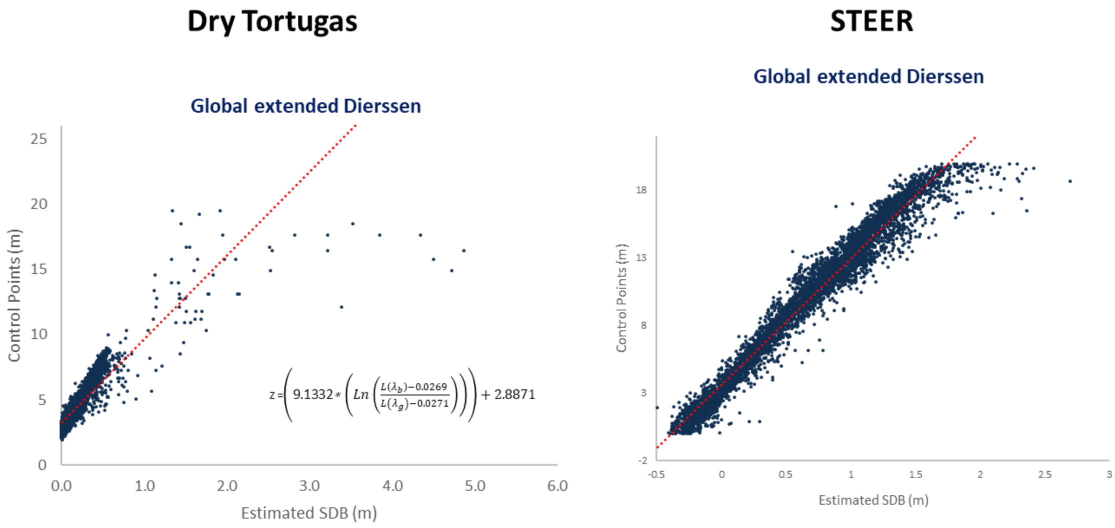


Figure 33 - Scatter plots of estimated SDB from global model against reference bathymetry in Dry Tortugas (left) and STEER (right), and the equation of the extended Dierssen model showing the parameters applied in the image.

Considering the SDB results provided by the global model, five groups were created based on two variables: the global model prediction residuals (i.e., the difference between the global estimated depths and the observed depths) and depths. A model was then calibrated for each of the five regions using the training points within each of the formed regions, denominated as the Local model. Table 8 presents the accuracy of *Global-Local* model in both study sites. The followed steps with the Global-Local approach are illustrated in Figures 34

(Dry Tortugas) and Figure 35 (STEER). These figures show the SDB applied to the entire area, the classes created, and the final bathymetry produced by *Global-Local* irregular segmentation. The Global model resulted in an overall RMSE of 0.89 m in Dry Tortugas while by dividing into five regions for the Local, the overall RMSE was decreased to 0.72 m.

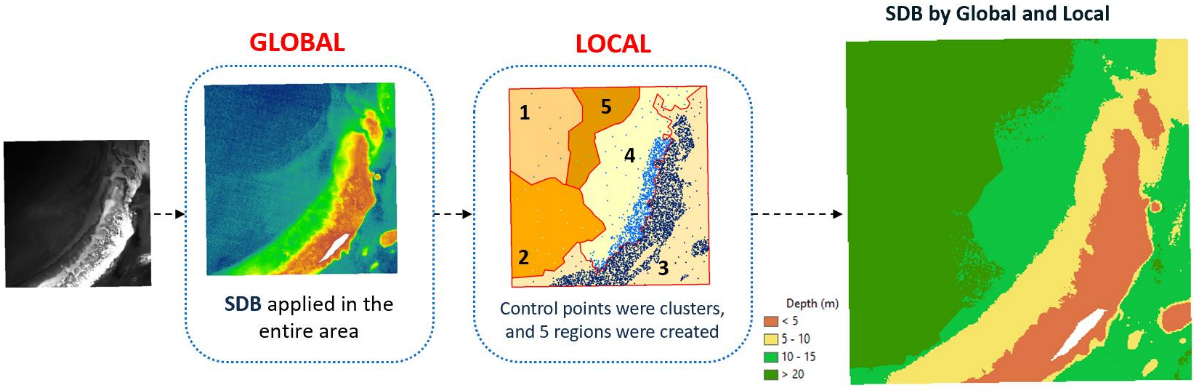

GLOBAL
LOCAL
SDB by Global and Local
1
5
4
2
3
SDB applied in the entire area
Control points were clusters, and 5 regions were created
Depth (m)
< 5
5 - 10
10 - 15
> 20

Figure 34 – Sequence of results from global model SDB; five regions created according to control points clustered based on two variables and the final bathymetry produced by irregular segmentation – Global-Local – in Dry Tortugas.

In contrast, in the STEER, smaller segments applied from the local approach did not produce any difference for depths up to 10 m in terms of accuracy. However, deeper bottom depths presented a slight improvement, resulting in a better overall result for the local approach than that applied for the entire area. The Global model produced an overall RMSE of 1.15 m, while with Local method the overall RMSE decreased to 1.02 m.

Table 8 – RMSE of the estimated depths from irregular segmentation approach in STEER and Dry Tortugas.

RMSE (m)	Approach	0–5	5–10	10–15	15–20	20-25	25-30	Overall (up to 30)
Dry Tortugas	Global	0.46	0.56	4.13	8.19	8.07	9.42	0.89
	Local	0.41	0.46	1.14	3.28	7.5	2.87	0.72
STEER	Global	0.65	0.88	1.15	1.72	-	-	1.15
	Local	0.65	0.88	0.99	1.44	-	-	1.02

The comparison between *Global-Local* models suggests that dividing the area into small spatial units produces better depth estimation, which was expected since geographically smaller areas imply a higher level of homogeneity in terms of water clarity and bottom type.

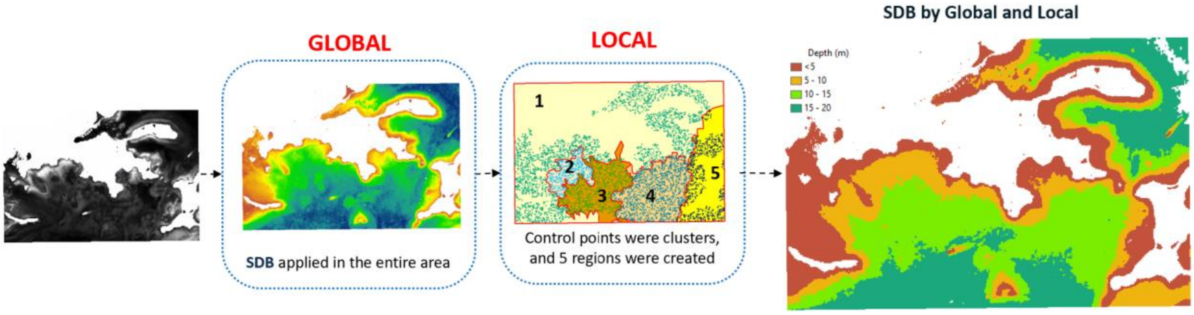

GLOBAL
LOCAL
SDB by Global and Local
SDB applied in the entire area
1
2
3
4
5
Control points were clusters, and 5 regions were created
Depth (m)
<5
5 - 10
10 - 15
15 - 20

Figure 35 - SDB surface created from global model; five regions created according to control points clustered based on two variables and the final bathymetry produced by irregular segmentation – Global-Local – in STEER.

5.4.3 Vertical Geographic

With *vertical segmentation*, the division of the area into smaller depth intervals substantially increased the accuracy, particularly in relatively deep waters. In Dry Tortugas, the overall RMSE of vertical segmentation every 1 m was reduced by 33.3% relative to every 2 m and 54.5% compared to 5 m segmentation (Table 9). The scatter plot represents the final SDB against the reference bottom depths, presenting a coefficient of determination of 0.99 and RMSE of 0.30 m. The residuals plot shows the coefficient of determination of 0.99 in the vertical segmentation (Figure 36). The final bathymetric surface based on the best performing segmentation interval (1 m) is displayed in Figure 37.

Table 9 - RMSE of the estimated depths from vertical segmentation approach in STEER and Dry Tortugas by depth ranges of 1, 2, 5 and 10 meters.

RMSE (m)	**Approach**	**Depth range**	**0–5**	**5–10**	**10–15**	**15–20**	**20-25**	**25-30**	**Overall**
Dry Tortugas	Vertical Segmentation	1m	0.29	0.28	0.49	0.65	0.88	1.42	0.30
		2m	0.34	0.40	0.77	1.49	1.73	5.11	0.45
		5m	0.39	0.50	1.35	1.75	4.29	9.69	0.66

STEER	1m	0.41	0.33	0.32	0.36	-	-	0.35
	2m	0.45	0.55	0.51	0.54	-	-	0.51
	5m	0.66	0.96	0.83	1.06	-	-	0.88

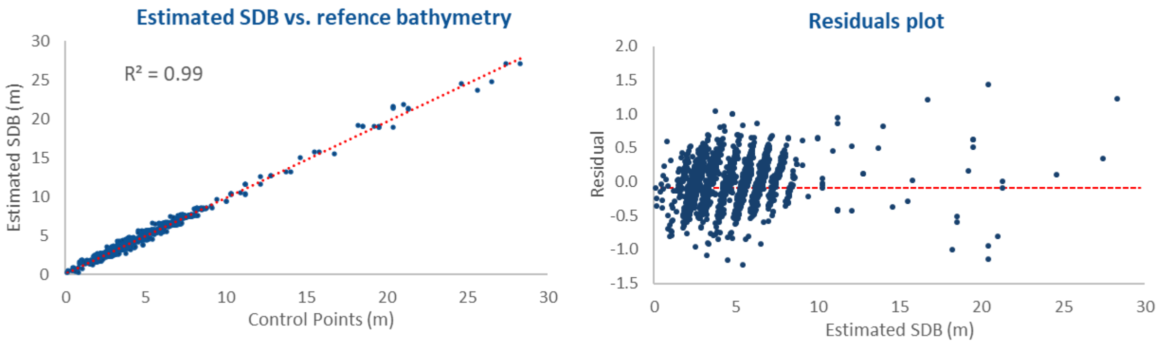


Figure 36 - The scatter plot (left) of the estimated SDB with the vertical segmentation (1 m interval) against reference bathymetry, and the respective residuals plot (right) for Dry Tortugas.

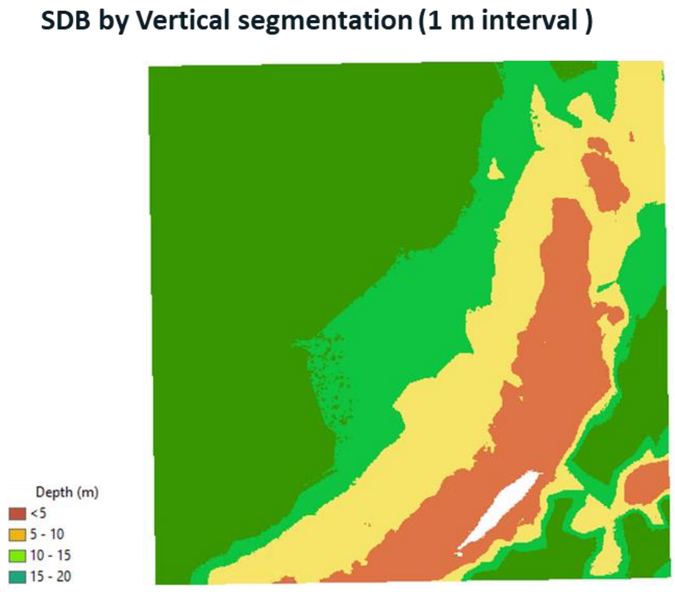


Figure 37 - SDB surface created from vertical segmentation (1 m interval) in Dry Tortugas.

The results in the STEER are even more impressive for the 1 m segmentation compared to the depth intervals of 2m and 5m. The overall RMSE using a 1 m interval was 0.35 m, improved by 31.37% relative to the 2m segmentation m and 60.22% compared to the 5 m segmentation. The scatter plot of the final SDB against the reference bottom depths along with residuals plot demonstrated a high coefficient of determination of 0.99 (RMSE = 0.35 m)

(Figure 38). Figure 39 illustrates the bathymetric surface from the vertical segmentation approach based on the best performing depth interval (1 m).

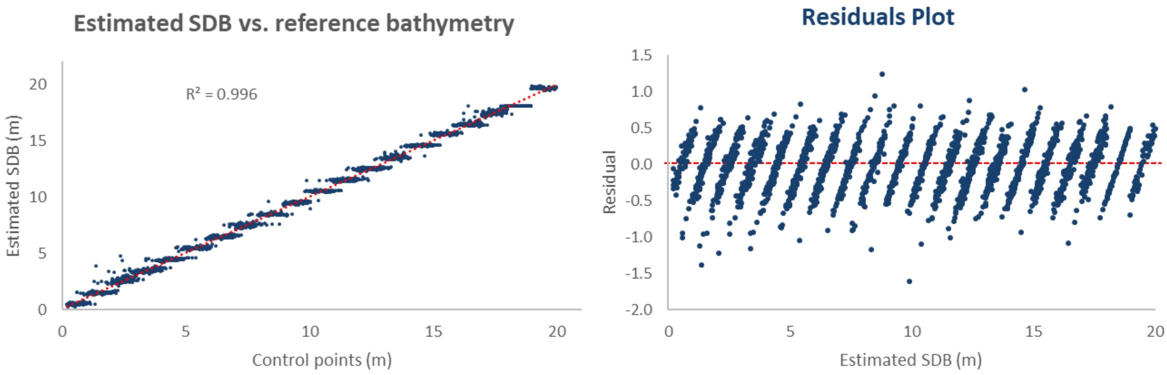


Figure 38 - The scatter plot (left) of the estimated SDB with the vertical segmentation (1 m interval) against reference bathymetry, and the respective residuals plot (right) for STEER.

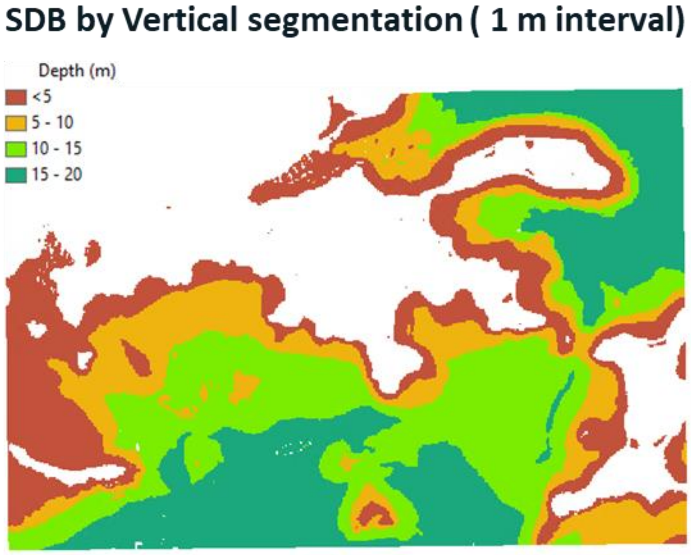


Figure 39 - SDB surface created from vertical segmentation (1 m interval) in the STEER.

5.4.4 Merged Geographic

5.4.4.1 Depth Areas by Depth Range (DADR)

For the DADR, after dividing the entire scene according to the depth areas of 5 m (resulting in 10 spatial units in Dry Tortugas and 16 in the STEER), each region was vertically subdivided into intervals of 1 m (that resulted in 32 spatial units in Dry Tortugas and 55 units in the STEER). Conceptually, this procedure is the same as subdividing the 1m vertical segments of Section 5.4.3 in terms of depth areas. Therefore, the accuracy of depth estimates by the DADR approach is compared with that of horizontal *Depth Areas* segmentation and the vertical segmentation to assess the benefits of the merged method (Table 10).

Table 10 - RMSE of the estimated depths from vertical and DA segmentation in comparison to the merged DADR approach in both sites.

RMSE (m)	**Approach**		**Units**	**0-5**	**5–10**	**10–15**	**15–20**	**20-25**	**25-30**	**Overall**
Dry Tortugas	Horizontal	DA	10	0.38	0.49	1.38	1.41	1.55	3.75	0.47
	Vertical	1 m	20	0.29	0.28	0.49	0.65	0.88	1.42	0.30
	Merged	**DADR**	32	0.29	0.29	0.49	0.65	0.88	1.42	0.30
STEER	Horizontal	DA	16	0.57	0.54	0.68	1.04	-	-	0.72
	Vertical	1 m	20	0.41	0.33	0.32	0.36	-	-	0.35
	Merged	**DADR**	55	0.34	0.33	0.32	0.37	-	-	0.34

There is no statistically significant difference between the vertical and merged segmentations (Table 11). The overall estimated depth values were equivalent, with RMSE of 0.30 m in Dry Tortugas and 0.34 m in the STEER. In the STEER, the improvement in the DADR approach compared to the vertical segmentation was between 0 to 5 m and 10 to 15 m, while for 5 to 10 and 15 to 20, the error decreases, which supports the conclusion that besides having more procedures, subdividing the vertical segments did not improve the estimated final bathymetry.

Table 11 - Statistics details from ANOVA test

Area	**Approach**	**Mean**	**Std. dev.**	***f***	***p***
Dry Tortugas	Vertical	-0.0221	0.2949	0.00687	0.93399
	DADR	-0.0241	0.2965		
STEER	Vertical	-0.0287	0.3847	0.26275	0.60842
	DADR	-0.0137	0.3287		

On the contrary, comparing the *Depth Areas* horizontal irregular segmentation approach (DA) with the DADR, the RMSE is 36.1% greater in Dry Tortugas and 52.77 % in STEER. Thus, the merged DADR approach is better than the DA method.

5.4.4.2 Global-Local by Depth Range (GLDR)

The performance of the extended Dierssen model from GLDR is compared to the *Global-Local* model. Five models (regions) were calculated with the *Global-Local*, whereas 28 models in Dry Tortugas and 78 in the STEER were created with the GLDR. GLDR offered more accurate depth estimates in all computed depths ranges, reducing the overall RMSE by 44.4% in Dry Tortugas and 41.1% in the STEER relative to the *Global-Local* approach (Table 12).

Table 12 – RMSE of horizontal irregular segmentation- Global-Local approach compared with the merged GLDR in both sites.

RMSE (m)	**Approach**		**Units**	**0–5**	**5–10**	**10-15**	**15–20**	**20-25**	**25-30**	**Overall**
Dry Tortugas	Horizontal	Global-Local	5	0.41	0.46	1.14	3.28	7.5	2.87	0.72
	Merged	**GLDR**	28	0.29	0.31	0.78	1.10	3.13	2.87	0.40
STEER	Horizontal	Global-Local	5	0.65	0.88	0.99	1.44	-	-	1.02
	Merged	**GLDR**	78	0.37	0.52	0.34	1.01	-	-	0.60

According to the above analysis, the vertical segmentation (1 m depth intervals) and DADR performed the best. The final SDB surface using the vertical segmentation approach at both study sites is shown in Figure 40 and Figure 41.

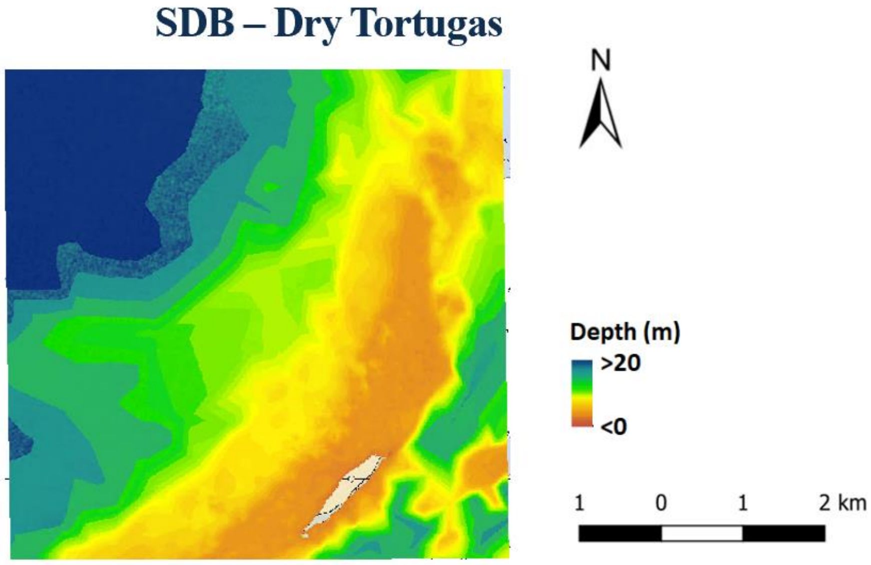


Figure 40 - Final Bathymetry derived from the vertical segmentation approach using the extended Dierssen model in Dry Tortugas.

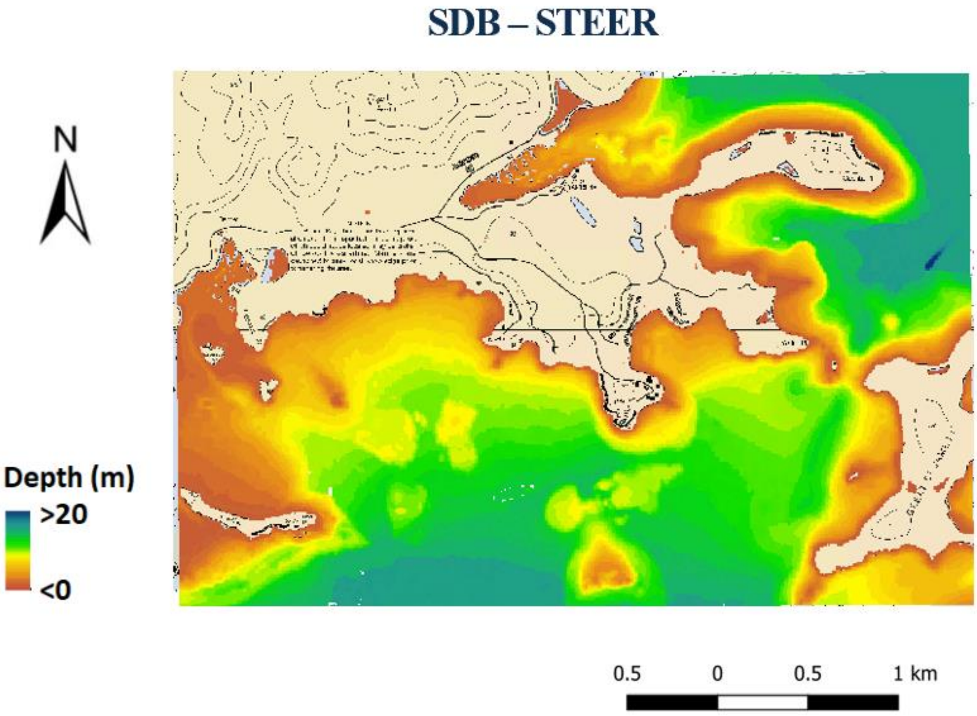


Figure 41 - Final Bathymetry derived from the vertical segmentation approach using the extended Dierssen model in the STEER area.

5.4.5 Dierssen vs. extended Dierssen Geographic

The geographic approaches discussed in the previous sections were carried out using the extended Dierssen model as extended by Freire (2017). The extended Dierssen model was proven more accurate than the conventional global models (see Table 4 in Section 5.3); however it is more complex in its implementation and poses the risk of the model not converring (or converging to the wrong solution) if the initial vector is not appropriately selected. Therefore, in this section we evaluate the best performing extended Dierssen geographic approach against the respective linear model (Dierssen).

Section 5.4.1 demonstrated that the vertical segmentation and the merged DADR methods perform the best. However, due to simplicity of the *vertical segmentation* method, compared to the DADR that requires additional steps and computations, for the evaluation of the Dierssen vs. extended Dierssen algorithm, the *vertical segmentation* is utilized.

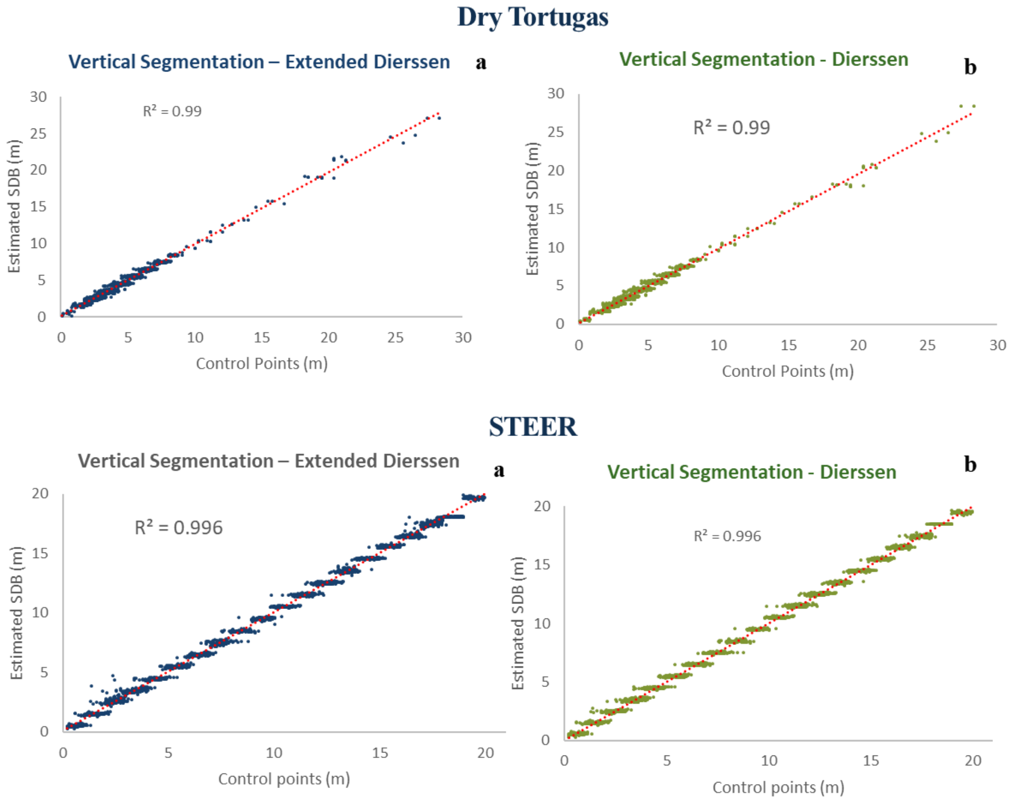


Figure 42 - Scatter plot of estimated SDB against reference bathymetry from the vertical segmentation using extended Dierssen model (a) and using Dierssen (b) in Dry Tortugas and STEER.

The scatter plots of Figure 42 present the correlation between control points and the estimated depths from: (a) the *vertical segmentation* approach using the extended Dierssen model; (b) the *vertical segmentation* approach using the Dierssen model (simplified). Table 13 summarizes the results in the form of the computed RMSE for every 5 m depth ranges.

Table 13 - Comparison of the vertical segmentation using Dierssen and extended Dierssen model according RMSE in Dry Tortugas and STEER.

		Approach	**0 – 5**	**5–10**	**10–15**	**15–20**	**20-25**	**25-30**	**Overall**
Dry Tortugas	Vertical	extended Dierssen	0.29	0.28	0.49	0.65	0.88	1.42	0.30
		Dierssen	0.29	0.29	0.43	0.88	0.94	1.36	0.31
STEER		extended Dierssen	0.41	0.33	0.32	0.36			0.35
		Dierssen	0.33	0.34	0.31	0.31			0.32

Figure 42 illustrates similar performance between the *vertical segmentation* using Dierssen and extended Dierssen models in both areas, with no statistically significant difference at 95% (*f*-1=0.66306; *p*=0.415805 in Dry Tortugas, and *f*-1=0.49994; *p*=0.479801 in STEER) and the same coefficient of determination. The overall accuracy in Dry Tortugas was 0.30 m in the extended Dierssen model and 0.31 in the Dierssen (simplified), while in the STEER, 0.35 m in the extended Dierssen, compared to 0.32 m in the Dierssen (Table 13).

5.4.6 Geographic (vertical) vs. conventional algorithms

In this section the best performing geographic approach (*vertical segmentation*) was compared against linear Dierssen and the most used conventional band-ratio Stumpf (2003) model (Pe’eri et al., 2014, Wei & Theuerkauf, 2020) applied for the entire area. Again, the comparison in Dry Tortugas was limited to optically shallow waters, i.e., 10 m. However, it is important to reiterate the ability of geographic models, especially the *vertical segmentation*, to estimate depths beyond the effective optical depth from the conventional approach, allowing depth extraction in quasi-optically deep waters. Figure 43 shows the final SDB surface using the conventional approach and the surface obtained in the *vertical segmentation* in Dry Tortugas, demonstrating the ability to increase the mapped coverage.

Furthermore, as the analysis was reduced to depths up to 10 m, the accuracy of the derived bathymetry every 2 m of depth is also investigated, as this may reveal a pattern and direction of errors in the estimation.

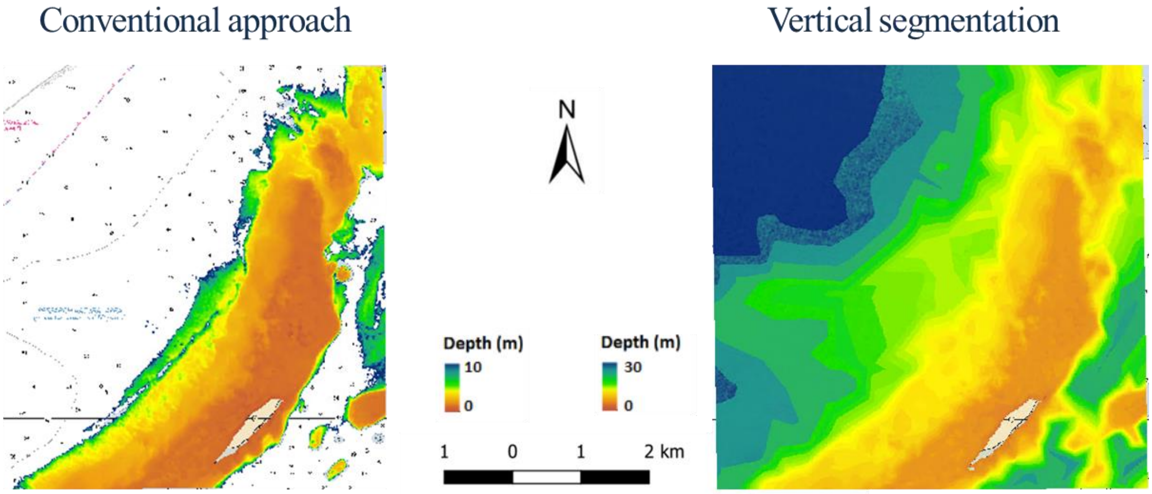
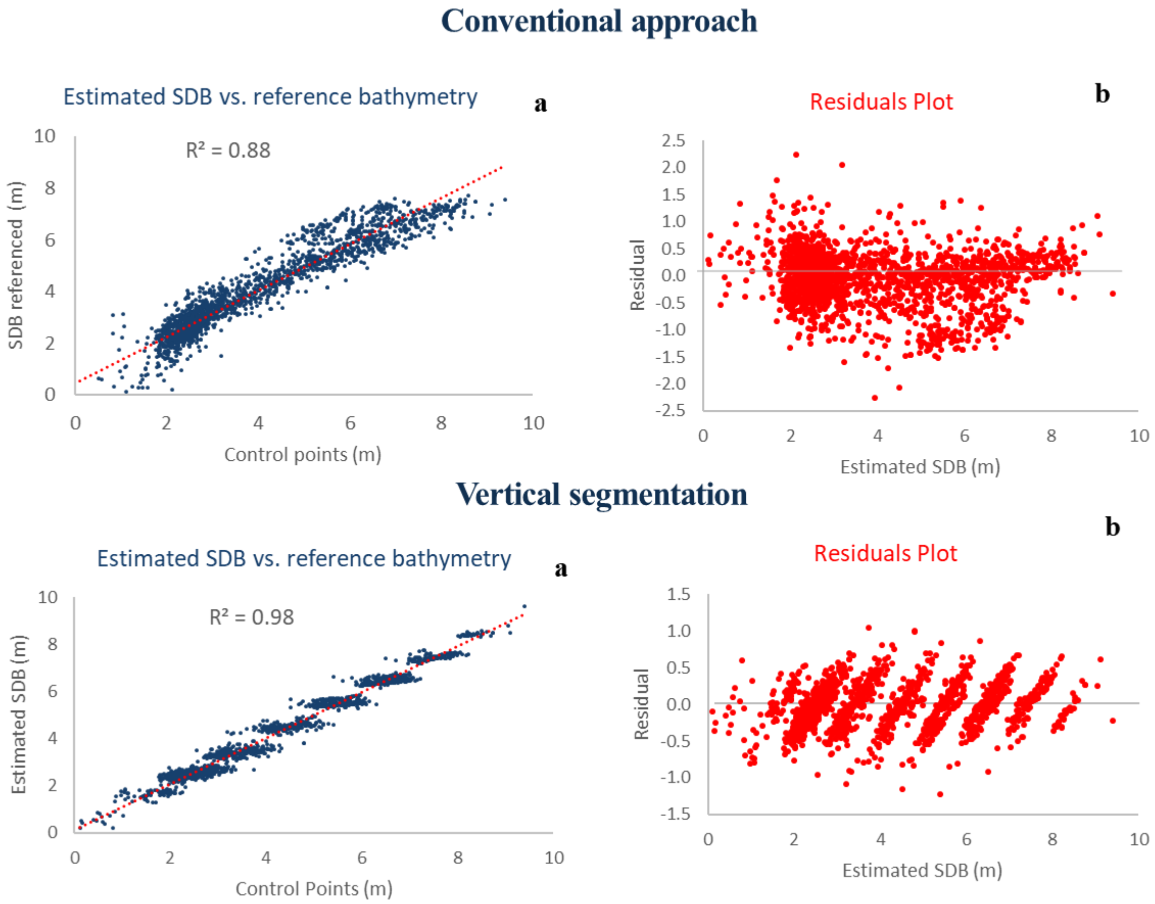


Figure 43 - Final Bathymetry derived from the conventional approach for the entire area and from vertical segmentation in Dry Tortugas.

When comparing the *vertical segmentation* with the conventional approaches in Dry Tortugas (Table 14), the accuracy has been substantially improved at all 2 m depth ranges, particularly in the relatively shallow-water areas, between 0 to 2 m. The overall accuracy of depth estimation with the *vertical segmentation* was increased by 43.1% compared to the conventional Dierssen model and 53.9% compared to the traditional Stumpf model. In addition, depth accuracy per CATZOC classification has been improved from A2/B and C to A1.

Figures 44 (Dry Tortugas) and Figure 45 (STEER) show the correlation of estimated bottom depts from SDB versus the reference bathymetry (a) and the residuals (b) for the conventional and *vertical segmentation* approaches. The residual plot of the conventional approach in both areas indicates a curve pattern which suggests that the independent variables have not captured the entire deterministic component. This trend coincides with the spatial distribution of prediction residuals in Figure 46. On the other hand, in the vertical segmentation, there is no clear tendency since the residuals are randomly distributed for the depth ranges.



Conventional approach

Vertical segmentation

Figure 44 - Scatter plot of estimated SDB against reference bathymetry (a) and residuals plot of the model (b) from the conventional algorithm and vertical segmentation in Dry Tortugas .

Table 14 - Comparison of the vertical segmentation and the conventional algorithms according RMSE and CATZOC in Dry Tortugas.

Area	**Approach**	**0 – 2**	**2 – 4**	**4 – 6**	**6 – 8**	**8 – 10**	**Overall (0– 10m)**	**ZOC**
Dry Tortugas	Vertical	0.37	0.26	0.32	0.27	0.28	0.29	A1
	Dierssen	0.64	0.45	0.63	0.48	0.42	0.51	A2/B
	Stumpf	1.11	0.47	0.59	0.68	1.20	0.63	C

In the STEER, the overall RMSE of the vertical segmentation approach was reduced by 72.8% relative to the RMSE of the conventional Dierssen model and 81.8% relative to the Stumpf model. The CATZOC is improved from C and D to A1 (Table 15).

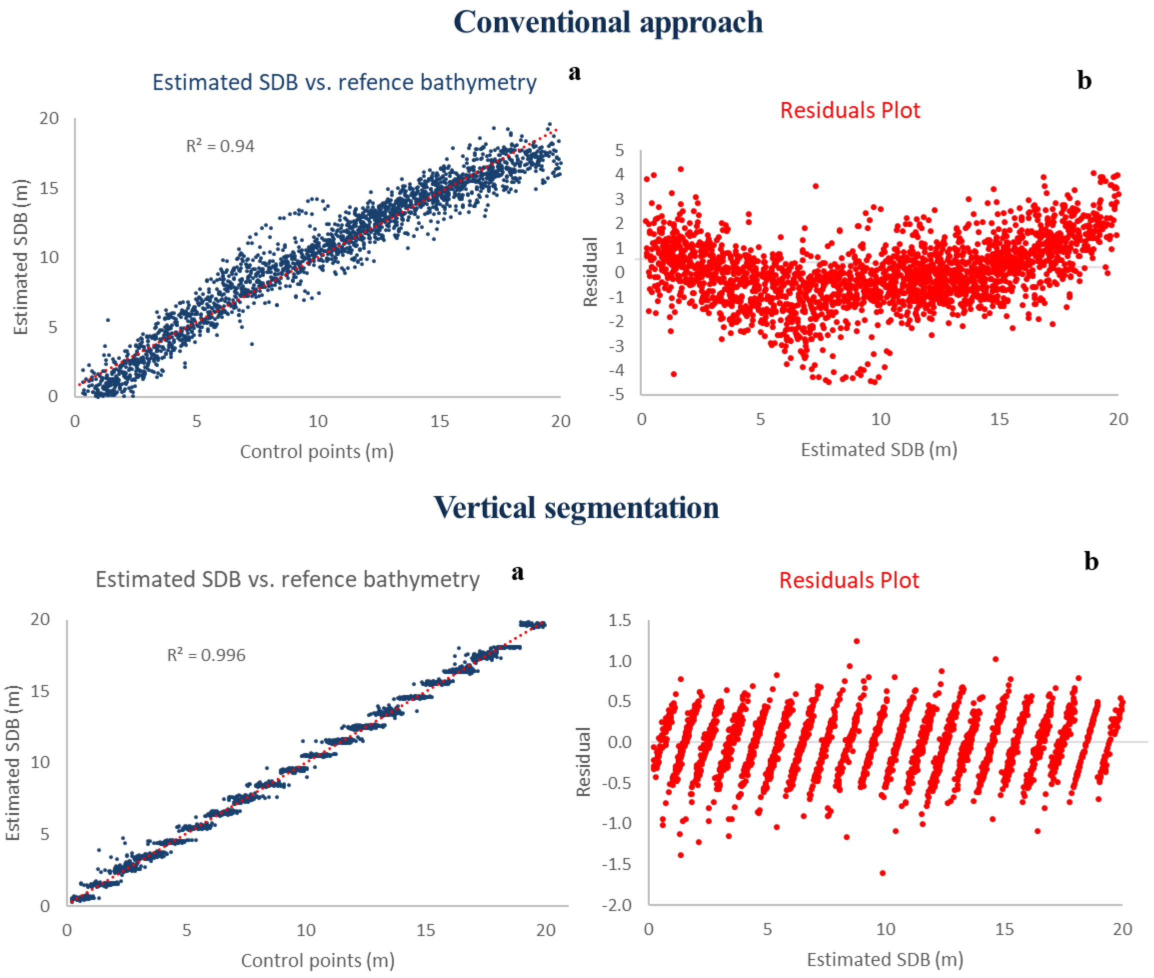


Figure 45 - Scatter plot of estimated SDB against reference bathymetry (a) and residuals plot of the model (b) from the conventional algorithm and vertical segmentation in the STEER.

Table 15 - Comparison of the vertical segmentation and the conventional algorithms according RMSE and CATZOC in STEER.

Area	**Approach**	**0 – 5**	**5 – 10**	**10 – 15**	**15 – 20**	**Overall (0– 20m)**	**ZOC**
STEER	Vertical	0.41	0.33	0.32	0.36	0.35	A1
	Dierssen	1.13	1.54	0.95	1.54	1.29	C
	Stumpf	1.58	2.47	1.11	2.43	1.93	D

Compared to the conventional model, dividing the entire area according to the vertical segmentation 1 m depth ranges has evidently improved the estimated depths. Nonetheless, the prediction residuals in the vertical segmentation and traditional approaches were mapped in order to investigate patterns related to their distribution. If the model is robust to bottom type variation, the prediction of residuals will be randomly distributed over the scene. On the other hand, if any cluster is observed, it will suggest an overestimation or underestimation of water depth depending on the bottom variations (Su *et. al.*, 2008). Figure 46 shows the spatial distribution of prediction residuals for both sites. The two maps on the left represent the spatial distribution of error, i.e., the difference between SDB estimated depths and the reference points, using the conventional approach, and the maps on the right show the distribution of error in the vertical segmentation. As discussed in Section 3.1.3, the band-ratio is based on the concept that the change in ratio between two bands due to bottom variation is less affected than the change in depth. However, the results of this work demonstrate that a degree of bottom heterogeneity is not captured by applying a single model in the entire scene (Figure 46). Clearly, the conventional global method - a single model - did not compensate for variations in the bottom type, and clusters of positive and negative residuals suggest that bottom variation is significant to the model's performance in depth estimation. On the other hand, the vertical segmentation - multiple models - approach presented smaller residuals and random location mapped distributed within the scene, which demonstrated a process with reduced heterogeneity of bottom type and water quality.

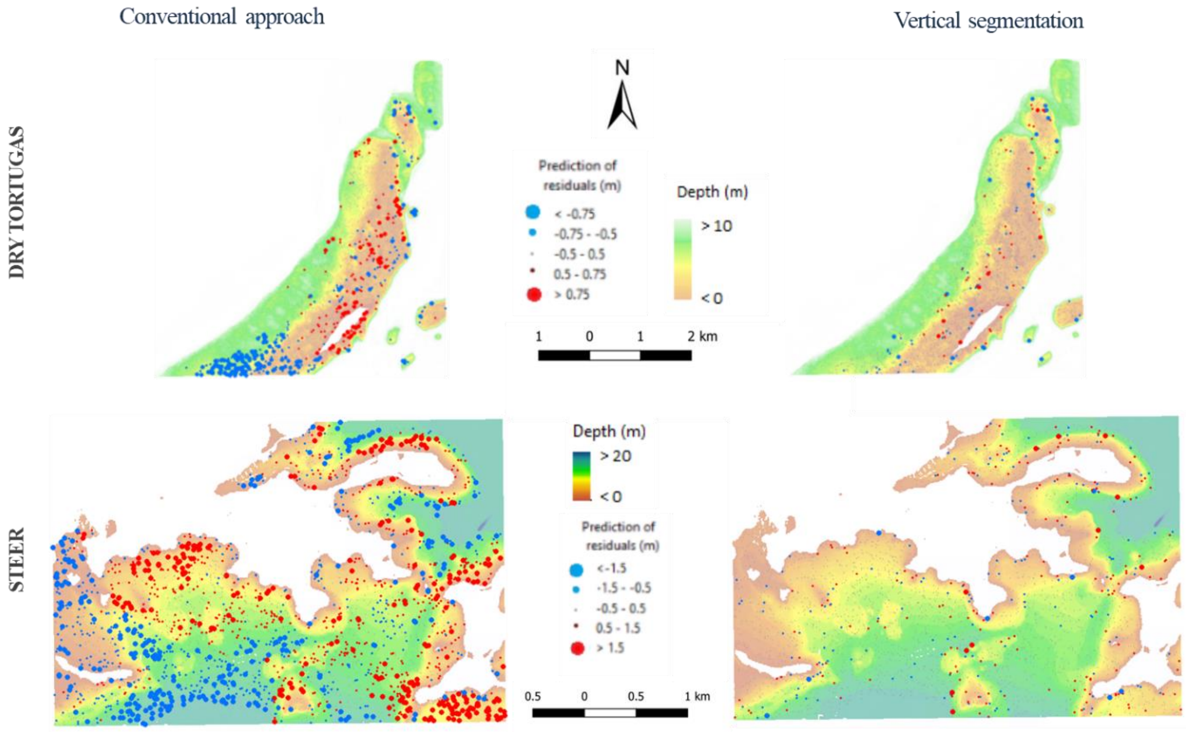


Figure 46 - Spatial distribution of prediction residuals of the models: (a) from the conventional approach using the Dierssen model and (b) from the vertical segmentation approach.

CHAPTER 6 : DISCUSSIONS, AND SUGGESTIONS FOR FUTURE WORKS

6.1 Discussion

This study sought to explore different segmentation procedures to address the inadequacy of conventional SDB models when the bottom type and water clarity vary spatially within the scene. By adding the water column contribution (L_w) to the model, six approaches were investigated based on horizontal (Regular, Global-Local, and Depth-Areas), vertical (Depth-Range), and merged (Global-Local by Depth Range and (GLDR) and Depth-Areas by Depth Range (DADR)) segmentation for inferring bathymetry.

By incorporating the water column contributions to the model through the Dierssen model (as extended by Freire (2007)), the L_w enhanced the overall SDB results, particularly when the model was calibrated globally, i.e., one model for the entire scene. The results from the two study areas demonstrated that, by including L_w to the SDB solution, it is possible to account for a certain level of the spatial heterogeneity of bottom type and water quality and better estimate depths. Nonetheless, when the area is segmented into smaller spatial units using our geographic models, no significant difference could be noted compared to the Dierssen model. This can be explained by that, with segmenting the scene into smaller spatial units, the water column contribution and bottom returns are approximately constant and/or insignificant within the small spatial units. Consequently, as Section 5.4.5. demonstrated, the Dierssen model (simplified) performs equally well to the extended Dierssen model when the geographic models are utilized.

The extended Dierssen model requires more effort and presents more limitations: the required number of control points within each spatial unit is greater than that for solving the Dierssen model, while calibrating the extended Dierssen model is also more computationally demanding. More importantly, the extended Dierssen function solution can fail if the pseudo-SDB (pSDB) computes a negative value or if the approximation vector ($X0$) is insufficiently defined and, thereby, the solution does not converge. In this study, the initial vector (*X*0) was derived performing two steps: first by considering water column contributions as negligible for computing the translation (m_0) and scaling (m_1) coefficients from a linear correlation between pixel values log-difference and depth measurements; second, to retrieve water column parameters, by the linearization of the RTE using the initial m_0 and m_1 thus computing an initial solution for the four parameters. As the extended Dierssen parameters are restricted to linear calculations, a meaningful SDB solution could be estimated.

Regardless of the Dierssen or extended Dierssen model, the application of the geographic approaches demonstrated promising results for the inclusion of SDB in hydrography in terms of depth accuracy, and especially for use on nautical charts. Categorizing the SDB results according to CATZOC showed that the area segmentation resulted in higher CATZOCs, i.e., higher level of depth accuracy category. In detail, in Dry Tortugas the CATZOC increased from A2/B to A1 (depth up to 10 m) and from C to A1 in the STEER study area (depth up to 20 m) when compared to the conventional Dierssen algorithm. Compared to the conventional Stumpf algorithm the improvement was even more impressive: from C to A1 in Dry Tortugas and from D to A1 in the STEER. However, it is pointed out that the above results and CATZOCs represent only the vertical accuracy. For a survey to be assigned a CATZOC, all data quality parameters must meet the requirements, i.e., feature detection, coverage, and horizontal accuracy besides the depth accuracy. Considering that full seabed coverage and feature detection requirements are not met with SDB, as well as the horizontal accuracy requirements for A1 and A2, it makes sense that the improved depth accuracy of the proposed methods may be reflected on the charts not with a higher CATZOC but rather with populating the QUASOU attribute for the derived soundings. The QUASOU attribute provides the reliability of a sounding which is meant to be populated when the depth uncertainty of charted soundings is better than that of the aggregated CATZOC for the area.

The findings of this study showed that all the investigated geographic approaches (i.e., horizontal, vertical, and merged) could better estimate depth and improve SDB coverage compared to the widely used conventional band-ratio algorithms. By segmenting the area and recognizing water column contribution to the model, the effects of the variation of bottom type and water quality were reduced, improving the accuracy and reliability of depth estimates from the image. One of the limitations of the method is the required number of training depths; as the segments become smaller, the total number of points to train the model increases. When the Dierssen model is applied, two points for each segment are required to provide a solution, whereas, for the extended Dierssen model, four training points are required for each spatial unit. While these numbers of points are capable of providing a solution, to achieve a good model performance and, consequently, better depth estimation, in practice a greater number of calibration points are necessary. The optimal number of calibration points may depend on factors, such as the geomorphology of the study area, the number and types of substrates and benthic cover types, and the size of the study area. This and the determination of the minimum / optimal number of points, from a model performance perspective, is part of future work.

Furthermore, the geographic models consist of multiple steps compared to the conventional method, thus, increased processing time is required. Since the individual steps and analysis were performed as a semi-automated process, Table 16 represents only an estimation of the total processing time for each geographic model (based on the experience derived from this work), where T is the processing time for the conventional Dierssen approach. Besides the estimated processing time, Table 16 summarizes the minimum number of training points required to derive a solution for each approach in the STEER, the number of models that were calibrated (in parenthesis), and the factors that may influence the number of points needed for an optimal solution.

Table 16 – Factors to be considered by applying geographic models.

Approach	Number of training points		Estimated Processing Time	Determinant Factors
	Dierssen	Extended Dierssen		
Horizontal Regular (#122)	188	376	7 x	Required accuracy The size of the investigated area Presence of estuary, bays, and freshwater outflow Geomorphological factors Number and types of substracts and benthic cover types
Depth Area (#16)	32	64	7 x	
Global-Local (#5)	10	20	5 x	
Vertical – 1m (#20)	40	80	5 x	
DADR (#55)	110	220	10 x	
GLDR (#78)	156	312	10 x	
Conventional (#1)	2	4	1 x	

Among the different geographic approaches, some considerations can be pointed out regarding the horizontal, vertical, and merged segmentation methods:

- **Vertical:** as the Results section demonstrated, dividing the image scene into smaller depth ranges improved the accuracy of estimated depths. In detail, for the depth ranges of 5m, 2m, and 1m used in this study, the accuracy of the depth estimates was improved by more than 15%, 29% and 43%, respectively, compared to the conventional approach in the optically shallow waters, and more than 49%, 65%, and 76%, respectively, in the quasi-optically deep area, in Dry Tortugas. Furthermore, the method demonstrated the capacity to estimate depths beyond the effective optical depth, considered as a fundamental limitation of the conventional approach, due to the piecewise nonlinear approximation of the depth. Nevertheless, it is important to point out the quantized prediction appearance (discontinuous surface) related to the depth range size. The discontinuity of depth estimates is an artifact caused by the segmentation since the models are calibrated separately for each depth range.

As the intervals become smaller, the depths are adjusted within the segment, resulting in the quantized appearance of Figure 47. The discretized appearance caused by piecewise functions is also mentioned by Wei & Theuerkauf (2020) and Rubert et al. (2003), wherein the latter presented similar residual plot pattern as those illustrated in Figures. 35 and 36, indicating that the homoscedasticity assumption is reasonable. This inherent attribute / limitation of the piecewise linear regression leads to a fundamental questions that requires further investigation: What is the optimal range that result in acceptable discontinuities and quantization of predictions vs. the improvement in accuracy? In this regard, and by looking at Figure 48, another research question is which approach provides the most reliable and useful estimated bathymetry for the application. In addition, the vertical segmentation presented cell size dependencies. Different spatial resolutions between satellite images and reference bathymetry surface can cause estimated out-of-range depths. Due to the image clipping having a lower spatial resolution than contour lines, some depth points may fall within the adjoined model. In practice, the estimated out-of-range depths are located on the wrong spatial unit/segment for which the adjacent depth range model is applied, resulting in these outliers and reducing accuracy.

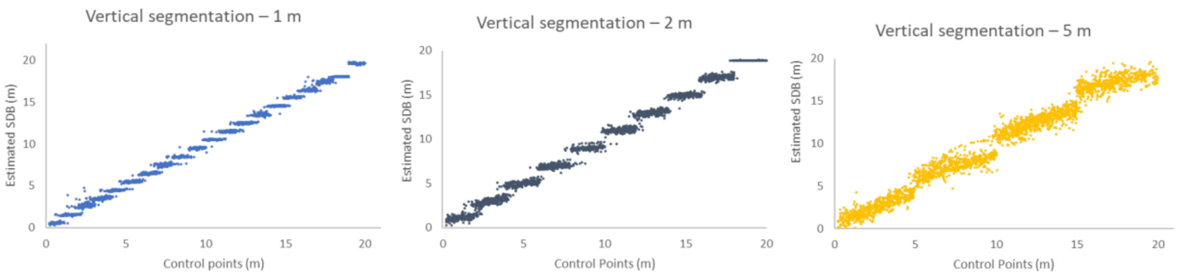


Figure 47 - The quantized appearance in different depth ranges – 1m depth range (left), 2m (center), and 5m (right).

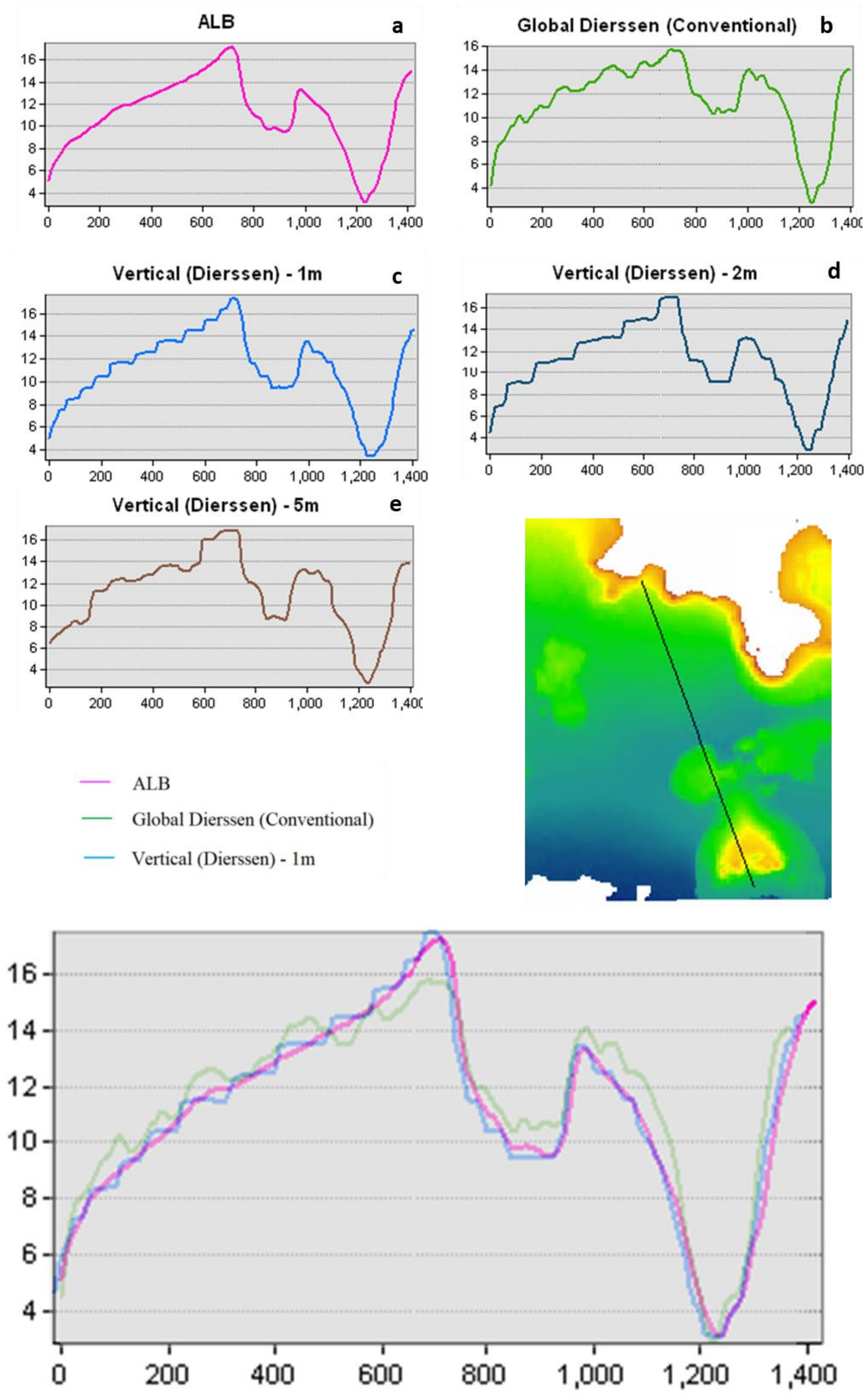


Figure 48 - Vertical profiles of the estimated bathymetry with the different models: (a) Lidar surface, (b) Global Dierssen – Conventional, (c) Vertical segmentation – 1m, (d) Vertical segmentation – 2m, and (e) Vertical segmentation – 5m in the STEER.

- **Horizontal:** despite observing that segmenting the area horizontally enhanced depth estimation compared to the conventional model applied in the entire scene, some limitations are evident. Defining the cell size to calculate the solution can be challenging with different bottom types and water quality, and can cause potential problems in estimating depths: in the Global-Local approach, 5 classes were created in both study sites, but in increasing the number of classes no significant improvement was noticed; for Depth-Areas horizontal segmentation, 10 classes were created in Dry Tortugas and 16 in the STEER; and contrary to the DA and Global-Local, the regular segmentation resulted in a considerably bigger number of small regions (400 segments in Dry Tortugas and 122 in the STEER) that, consequently, often contained a low number of control points for training the model. Fewer than four depths within a grid cell results in cells merging as a solution cannot be achieved. The fact that Dry Tortugas presents lower accuracy in the horizontal segmentation compared to the STEER is the number of small cells required to be merged with the larger ones: 74% compared to 23% in the STEER. Also, as the number of cells created is significantly greater in the regular segmentation than from the other horizontal segmentation approaches, the number of the required control points to calibrate the model and represent the bottom depth is also greater. And the computational cost to fit all the required models.
- **Merged:** the results of the merged method presented no statistical difference when compared to the vertical approach. In addition to its complexity, as it requires a greater number of steps, number of control points is required to calibrate the model. Likewise to those discussed for the vertical segmentation,

the quantized prediction and model discontinuity issue remains (with the aforementioned concerns).

While the vertical segmentation approach tested in this study was found to have substantially better accuracy than the conventional approach using a single (global) linear model, it is important to note that questions remain about the interpretation of this result, and, in particular, whether it might be related to overfitting, rather than an actual improvement in accuracy, particularly for the 1m segmentation. This concern arises from the analysis of the plots in Figures. 44, 45, and 47, and the quantized appearance of the vertically-segmented SDB vs. reference bathymetry scatterplots. Further analysis indicated that the reason for this quantization is that, in the vertical segmentation approach, many of the individual models (where each model corresponds to a particular depth range, such as 1-2 m) have low linear correlation between relative SDB and the lidar-derived depths used for training, with low slopes of the regression lines (Figure 49). This means that within a particular segment, the model output is within a narrow depth range (approximately 0.4m in the 1m segments) around the mean depth.which results in clustering of outputs and the quantized appearance of the scatterplots. Based on this analysis, it is strongly recommended that future work include an additional test to investigate whether the vertical segmentation approach is actually leading to a substantial increase in accuracy. Specifically, it is recommended to train the vertical-segmentation model using lidar data for one area, and then test it (without retraining) using a different data set (e.g., MBES) in a nearby but spatially-separate site.

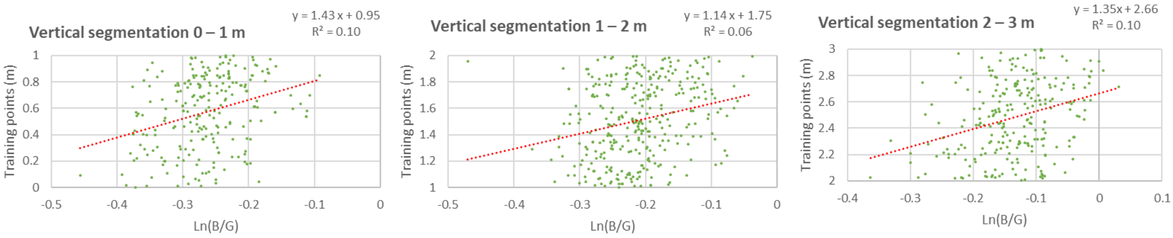


Figure 49 - Regression line in individual models in the vertical segmentation– depth range of 0 – 1m (left), 1 – 2m (center), and 2 – 3m (right).

Although previous works have investigated dataset segmentation, a comparison would not be possible due to differences between datasets, conditions of the area under investigation, and the model applied. As example, while this work implemented the Dierssen algorithm by vertically and horizontally segmenting the image scene into smaller spatial units, Poursanidis *et al*. (2019) divided the dataset in two groups of depth using the empirical method by Lyzenga (1985), and Vargas *et al*. (2021) tested three groups of depths using Stumpf algorithm.

6.2 Future work

Regarding the SDB model, the estimation of optical parameters was conducted from the Diersssen model using LSM adjustment to derive a solution. Further research could be conducted exploring IOP's and AOP's of water, such as diffuse attenuation, bottom sediments, and turbidity, in order to provide more accurate meanings of parameters.

Segmentation approaches, may be further investigated:

- the regular segmentation comes with some limitations, i.e., the size of the grids, that influences the bathymetric estimation. When the grid is too large, it may not contain the proper homogeneity of bottom type, while if it is too small, it may not include the number of control points required to model the surface sufficiently or it may result in overfitted models. In this context, the grid size was arbitrarily defined, but more complex approaches could be investigated. Quadtree data structures, for example, could be applied to partition the image scene by recursive subdivisions based on maintaining a sufficient number of control points within each cell. This approach could ensure an appropriate grid size to cover bottom type and water quality heterogeneity.
- vertical segmentation also is characterized by a limitation. The quantized prediction toward the center leads to a discontinuous surface with break points between the segments. As demonstrated in this study, the piecewise function

produces this discretized appearance; however, it is highly dependent on the depth range size. Instead of fixed depth intervals, such as 1, 2, and 5 m preset in the vertical segmentation, future research could include tests to find a good agreement between relative SDB and referenced bathymetry and, simultaneously, provide a smoother surface. Instead of only using RMSE as a criterion to find the best solution, the analysis could, as well, include a minimum coefficient of determination (R^2) as a threshold.

Although geographic models demonstrated an ability to estimate depth very well in all depth intervals, up to the effective optical depth and in quasi-optically water, they may be overfitting. Identifying whether the vertical segmentation approach is actually leading to a substantial increase in accuracy is strongly recommended. To evaluate model performance more accurately, further studies could train the vertical model using lidar data for one area and then test it for a nearby site (geographically separate) with a different dataset (e.g., existing, ALB, MBES, or other reference data).

This study showed an improvement in estimate depth when vertical segmentation is applied over the area. Although, clearly this approach was able to account for variations in the water column and seabottom, when the prediction of residuals was plotted over the benthic habitat map developed by NOAA (Figure 50), there is a possible correlation between residuals and specific bottom types. Further potential research could investigate the influence of bottom type on SDB final results.

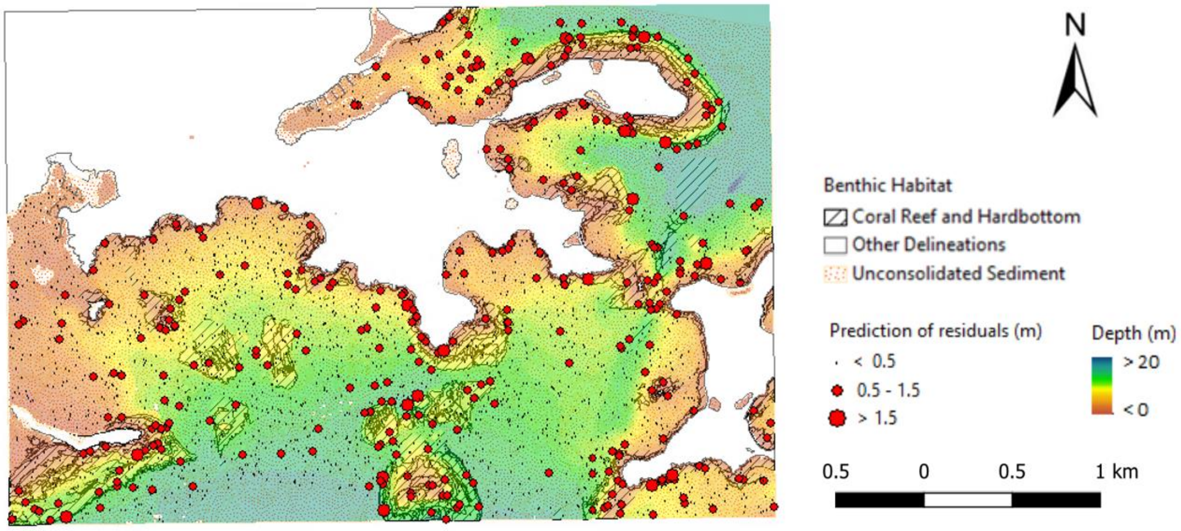


Figure 50 - Spatial distribution of prediction residuals from verical approach againt depth and benthic habitat in STEER.

CHAPTER 7 : CONCLUSION

This study investigated the segmentation of geographic space and the incorporation of the extended Dierssen model to address SDB limitations regarding bottom type and water quality heterogeneity in order to extract more accurate depth information from satellite images. The key assumption was that in small areas the water column and sea bottom demonstrate sufficient homogeneity to provide a better bottom depth discrimination. In practice, contrary to the conventional approach, where a single model for the entire scene is calculated, the area was partitioned vertically and/or horizontally. Each subarea was modeled over the extended Dierssen function varying from place to place.

Although the conventional approach can account for a certain level of heterogeneity in bottom types returns, this study demonstrated the capability of reducing the variation by segmenting Sentinel-2 satellite image acquired over Dry Tortugas and the St. Thomas East End Reserve (STEER). The estimated depths in both study areas presented a significantly improved accuracy, more than 40% in Dry Tortugas and more than 60% in the STEER, compared to conventional approaches in optically shallow water. Vertical segmentation provided greater performance in detecting and improving depth accuracy due to the piecewise functions and the reduced heterogeneity of bottom type and water quality by depth ranges. Furthermore, when the data was vertically divided in Dry Tortugas, the depths presented the same accuracy estimation for the entire dataset (shallow and quasi-optically deep water), demonstrating an ability to infer bathymetry beyond the effective optical depth (10 m in the particular study area) which is the limit for depth estimation in the conventional approaches. However, the quantized predictions of the individual linear regression models demonstrated by the vertical segmentation raises the question of overfitting rather than an actual improvement in accuracy, something that is part of future work.

On the other hand, although there was an improvement in estimating depth with the horizontal segmentation methods compared to the traditional method, this work’s results indicated the need for additional control points and cell size dependencies. At the same time, in spite of providing the same accuracy as the vertical segmentation, the merged approach suggested a limit to splitting the area. If it is too small, it can create segments where the number of points will not ensure a better depth estimation, besides being more computationally demanding and requiring more control points. This makes the vertical segmentation the preferable method.

The two case studies also suggested that applying the extended Dierssen model and fully considering the RTE improves the bottom discrimination when the entire area is considered to calibrate the model. However, by adding the water column parameter into the model, the extended Dierssen model did not add significant contribution by sub-dividing the scene into small pieces. Furthermore, the extended Dierssen model adds complexity and limitations, which suggests that the simpler function is preferable for the geographic models.

Although geographic models provides significant improvements, the need for an increased number of control points should be noted, due to the greater number of segments formed for model training. Therefore, data with dense coverage of points is necessary, and studies with a reduced number of control points should be investigated.

Overall, the accuracy of bathymetric estimates was improved when the scene was segmented into smaller spatial units. The findings presented in this study are an exciting opportunity to lead to many potential new and improved applications for HOs. Since the accuracy has increased substantially, the bathymetry data generated using the geographic model can be used to provide valuable information to help ensure safe navigation. As such, understanding the potential of remote sensing for contributing to improved nautical products is critical for HOs to ensure another step of being included for regular application as a mapping tool within the IHO standards.

REFERENCES

Ashphaq, M., Srivastava, P., Mitra, D., 2021. Review of near-shore satellite derived bathymetry: Classification and account of five decades of coastal bathymetry research. *Journal of Ocean Engineering and Science*, 6: 10-101

Baban, S.M.J., 1993. The evaluation of different algorithms for bathymetric charting of lakes using Landsat image. *International Journal of Remote Sensing*, 14, pp.2263-73. Retrieved from https://www.bbc.com/news/world-latin-america-50223106. 2019.

Bäckström, A., 2018. *A new bathymetric model of Lake Vättern, Southern Sweden.* (Master's thesis, Stockholm University). Stockholm. Retrieved from https://www.su.se/polopoly_fs/1.555019.1620715618!/menu/standard/file/2018_Alexander%20Backstrom_msc_Marine-Geology_60%20hp.pdf

Beer, 1852. Determination of the absorption of red light in colored liquids. Annalen der Physik und Chemie, German. 162 (5): 78-88. Doi:10.1002/andp.18521620505.

Bramante, J.F., Raju, D.K., Sin, T.M., 2013. Multispectral derivation of bathymetry in Singapore's shallow, turbid waters. *International Journal of Remote Sensing*, 34:6, 2070-2088.

Caballero I., Stumpf, R. P., 2019 Retrieval of nearshore bathymetry from Sentinel-2A and 2B satellites in South Florida coastal waters. Estuarine, Coastal and Shelf Science, Vol.226, 2019, 106277.

Caballero I., Stumpf, R. P., 2020. Atmospheric correction for satellite-derived bathymetry in the Caribbean waters: from a single image to multi-temporal approaches using Sentinel-2A/B. *Optics Express*, Vol.28, No. 8/13.

Ceyhun, Ö., Yalçın, A., 2010. Remote sensing of water depths in shallow waters via artificial neural networks. *Estuarine, Coastal and Shelf Science,* 89, 89–96.

Costa, B.M., Kendall, M. S., Edwards, K., Kagesten, G., Battista, T. A., 2013. Benthic Habitats of Fish Bay, Coral Bay, and the St. Thomas East end Reserve. NOAA Technical Memorandum NOS NCCOS 175 (Online).

Custom Script Tutorial, 2019. *Create Useful and Beautiful Satellite Visualizations in EO Browser*. Sinergise laboratory for geographical information systems, Ltd., Ljubljana.

Dierssen, H. M., Zimmerman, R. C., Leathers R. A., Downes, T. V. and Davis, C. O., 2003. Ocean color remote sensing of seagrass and bathymetry in the Bahamas Banks by high-resolution airborne image. *Limnology and Oceanography,* 48: 444–455.

Dekker, A.G., Sagar, S., Brando, V.E. and Hudson, D., 2012. "Bathymetry From Satellites for Hydrographic Purposes", Shallow Survey Conference, Wellington, New Zealand.

Forfibski-Sarkozi, N.A., 2019. Mapping Nearshore Bathymetry with Spaceborne Data Fusion and State Space Modeling (Doctoral dissertation, Oregon State University). Oregon. Retrieved from https://ir.library.oregonstate.edu/concern/ hh63t309t

Forfinski-Sarkozi, N. A., Parrish, C. E., 2019. Active-Passive Spaceborne Data Fusion for Mapping Nearshore Bathymetry. *Photogrammetric Engineering and Remote Sensing*, Vol. 85, No. 4, pp. 281-295.

Evagorou, E., Mettas, C., Agapiou A., Themistocleous, K., Hadjimitsis, D., 2019. Bathymetry maps from multi-temporal analysis of Sentinel-2 data: the case study of Limassol, Cyprus. *Advances in Geosciences*, 45, 397–407, 2019.

Federal Geographic Data Committee (FGDC), 1998. Geospatial Positioning Accuracy Standards. Part 3: National Standard for Spatial Data Accuracy.

Freire, R.R., 2017. *Evaluating Satellite Derived Bathymetry in Regard to Total Propagated Uncertainty, Multi-Temporal Change Detection, and Multiple Non-Linear Estimation* (Doctoral dissertation, University of New Hampshire). Durham. Retrieved from https://scholars.unh.edu/dissertation/2281/

Gao, B.C., 1996. NDWI - A normalized difference water index for remote sensing of vegetation liquid water from space. *Remote Sensing of Environment,* 58: 257-266.

GEBCO, 2020. Seabed2030 Project. https://www.gebco.net/about_us/seabed2030_project/

Ginsburg, R. N., 1953. Beachrock in South Florida*. Journal of Sedimentary Petrology,* 23:89–92.

Godwin B., 2021. Penetration of Visible Radiation from Sunlight through Water. University of Minnesota Sea Grant Program.

Herlevi, A., 2002. *Inherent and apparent optical properties in relation to water quality in Nordic waters*. (Doctoral dissertation, University of Helsinki). Finland. Retrieved from http://hdl.handle.net/10138/23105.

Holman, K., 2020. *Testing approaches and sensors for satellite-derived bathymetry in Nunavut*. (Master's thesis, University of Ottawa). Ottawa. Retrieved from https://ruor.uottawa.ca/handle/10393/41402

International Hydrographic Organization (IHO), 2008. S-44. IHO Standards for Hydrographic Surveys, 5th ed.; International Hydrographic Bureau (IHO): Monaco, the Principality of Monaco.

International Hydrographic Organization (IHO), 2018. S-4 - Regulations of the iho for international (INT) charts and chart specifications of the IHO: Monaco, the Principality of Monaco.

International Hydrographic Organization (IHO), 2000. S-57 – IHO Transfer Standard for Digital Hydrographic Data: Monaco, the Principality of Monaco.

Kanno, A., Koibuchi, Y., Isobe, M., 2011b. Shallow Water Bathymetry from Multispectral Satellite Images: Extensions of Lyzenga’s Method for Improving Accuracy. *Coastal Engineering Journal,* 53, 431– 450.

Kastrisios, C., & Calder, B., 2018. Algorithmic implementation of the triangle test for the validation of charted soundings. *In Proceedings of the 7th International Conference on Cartography and GIS*, Sozopol, Bulgaria, June, BulgarianCartographic Association. (pp. 18–23). https://doi.org/10.13140/RG.2.2.12745.39528

Kastrisios, C., Sullivan, B., Powell, J., Baek, Y., 2022. Hydrographic Geospatial Data Standards. The Geographic Information Science & Technology Body of Knowledge (2nd Quarter 2022 Edition), John P. Wilson (Ed.) https://doi.org/10.22224/gistbok/2022.2.5.

Kastrisios, C., Ware, C., 2022. Textures for coding bathymetric data quality sectors on electronic navigational chart displays: design and evaluation, *Cartography and Geographic Information Science*. https://doi.org/10.1080/15230406.2022.2059572.

Lee, Z., Carder, K. L., & Arnone, R. A., 2002. Deriving inherent optical properties from water color: a multiband quasi-analytical algorithm for optically deep waters. Applied Optics, 41(27), 23–30.

Lyzenga, D. R., 1978. Passive remote sensing techniques for mapping water depth and bottom features. *Applied Optics*, 17: 379–383.

Lyzenga, D. R., 1985. Shallow-water bathymetry using combined lidar and passive multispectral scanner data. *International Journal of Remote Sensing*, 6: 115–125.

Lyzenga, D. R., Malinas, N. P. and Tanis, F. J., 2006. Multispectral bathymetry using a simple physically based algorithm. *IEEE Transactions on Geoscience and Remote Sensing*, 44: 2251–2259.

Martins, V. S., Barbosa, C. C. F., Carvalho, L. A. S., Jorge, D. S. F., Lobo, F. L., Novo, E. M., L. M., 2017. Assessment of Atmospheric Correction Methods for Sentinel-2 MSI Images Applied to Amazon Floodplain Lakes. *Remote Sensing*, 9, 322; doi:10.3390/rs9040322

Mavraeidopoulos, A. K., Pallikaris, A., & Oikonomou, E. 2017.. Satellite derived bathymetry (SDB) and safety of navigation. *International Hydrographic Review*.

McFeeters, S.K., 1996. The use of the Normalized Difference Water Index (NDWI) in the delineation of open water features. *International Journal of Remote Sensing*, 1425–1432.

McKinna, L. I. W., Fearns, P. R. C., Weeks, S. J., Werdell, P. J., Reichstetter, M., Franz, B. A., Feldman, G. C. (2015). A semianalytical ocean color inversion algorithm with explicit water column depth and substrate reflectance parameterization. *Journal of Geophysical Research:Oceans*, 120, 1741–1770. https://doi.org/10.1002/2014JC010224.

National Park Service, 2014. Natural Resource Stewardship and Science. Dry Tortugas National Park – Geologic Resources Inventory Report.

National Oceanic and Atmospheric Administration- NOAA, 2017. Topobathy Lidar Survey Report. Project Geographic Location: Dry Tortugas, Florida Descriptive report W00368.

National Oceanic and Atmospheric Administration- NOAA, 2021.Hydrographic Survey Specifications and Deliverables. Office of Coast Survey Hydrographic Surveys Division. NOAA. Retrieved from https://nauticalcharts.noaa.gov/publications/docs/standardsrequirements/specs/HSSD_2021.pdf

Pe'eri, S., Parrish, C. E., Azuike, C., Alexander, L. and Armstrong, A., 2014. Satellite Remote Sensing as a Reconnaissance Tool for Assessing Nautical Chart Adequacy and Completeness. *Marine Geodesy*, 37(3): 293-314.

Philpot, W. D., 1989. Bathymetric mapping with passive, multispectral image. *Applied Optics*, 28:1569–1578.

Philpot, W.D., Davis, C.O., Bissett, W.P., Mobley, C., Kohler, D.D.R., Lee, Z.P., Snyder, W., Steward, R.G., Gould, R. and Arnone, R., 2004. Bottom characterization from hyperspectral image data. Oceanography, 17(2): 76-85

Poursanidis, D., Traganos, D., Chrysoulakis, N., Reinartz, P., 2019. Cubesats Allow High Spatiotemporal Estimates of Satellite-Derived Bathymetry. *Remote Sensing*. 11, 1299. https://doi.org/10.3390/rs11111299.

Preisendorfer, R. W., 1976. Hydrologic Optics, vol. 1, Introduction, Natl. Tech. Inf. Serv., Springfield, VA.

Ruppert D., Wand, M.P., Carroll, R.J. 2003. Semiparametric Regression. Cambridge, New York: Cambridge University Press; pp. 386.

Sagawa, T.; Yamashita, Y.; Okumura, T.; Yamanokuchi, T. Satellite Derived Bathymetry Using Machine Learning and Multi-Temporal Satellite Images. *Remote Sensing.* 2019, *11*, 1155. https://doi.org/10.3390/rs11101155

Science Education through Earth Observation for High Schools – SEOS, 2022. Introduction to Categorisation of Objects from their Data.

Salah, M., 2016. Determination of Shallow Water Depths using Inverse Probability Weighted Interpolation: A Hybrid System-Based Method. *International Journal of Geoinformatics*.

Staff, D., 2013. The Benefits of the 8 Spectral Bands of WorldView-2: Applications Whitepaper. Digital Globe, London, Technical Report WP-8SPEC**.**

Stumpf, R. P., Holderied, K. and Sinclair, M., 2003. Determination of water depth with high-resolution satellite image over variable bottom types. *Limnology and Oceanography*, 48: 547–556.

Su, H, Liu, H., Heyman, W. D., 2008. Automated Derivation of Bathymetric Information from Multi-Spectral Satellite Image Using a Non-Linear Inversion Model. *Taylor & Francis*, 31: 281–298.

Su, H, Liu H., Wang L., Filippi A. M., Heyman W. D., Beck R. A., 2013.Geographically Adaptive Inversion Model for Improving Bathymetric Retrieval From Satellite Multispectral Image. *IEEE Transactions on Geoscience and Remote Sensing*, vol. 52, no. 1, pp. 465-476, doi: 10.1109/TGRS.2013.2241772.

Tobler, W. R., 1970. A Computer Movie Simulating Urban Growth in the Detroit Region. Economic Geography, 46, 234-240. https://doi.org/10.2307/143141

Traganos, D., Poursanidis, D., Aggarwal, B., Chrysoulakis, N., Reinartz, P., 2018. Estimating Satellite-Derived Bathymetry (SDB) with the Google Earth Engine and Sentinel-2. *Remote Sensing*, 10, 859.

Vanhellemont, Q., 2019. Adaptation of the dark spectrum fitting atmospheric correction for aquatic applications of the Landsat and Sentinel-2 archives. *Remote Sensing of Environment*, 225, 175–192.

Vinayaraj, P., 2019. *Development of Algorithms for Near-shore Satellite Derived Bathymetry Using Multispectral Remote Sensing Images*. (Master's thesis). Tokyo. Retrieved from http://orcid.org/0000-0001-9872-1225.

Vojinovic, Z; Abebe, Y A; Ranasinghe, R; Vacher, A; Martens, P; et al. A machine learning approach for estimation of shallow water depths from optical satellite images and solar measurements. *Journal of Hydroinformatics;* London Vol. 15, Iss. 4, (Oct 2013): 1408-1424. DOI:10.2166/hydro.2013.234

Wang, 2019. *Analyzing the possibility of using satellite-derived bathymetry in Greenland.* (Master's thesis, Aalborg University). Aalborg. Retrieved from https://projekter.aau.dk/projekter/files/306330716/master_thesis.pdf.

Ward, R., Alexander, L., Greenslade, B., 2009. *IHO S-100: The New IHO Hydrographic Geospatial Standard for Marine Data and Information*. International Hydrographic.

Wei, C. & Theuerkauf, S. J., 2020. A Novel Multitemporal Approach for Satellite-Derived Bathymetry for Coastal Waters of Palau. *Journal of Coastal Research*, 37: 336–348.

Wells, D. E. and Krakiwsky, E. J., 1971. The Method of Least Squares. Department of Survey Engineering, Lecture Notes nº 18, University of New Brunswick, Fredericton.

Werdell, P. J., McKinna, L. I. W., Boss, E., Ackleson, S. G., Craig, S. E., Gregg, W. W., Zhang, X., 2018. An overview of approaches and challenges for retrieving marine inherent optical properties from ocean color remote sensing. *Progress in Oceanography*, *160*(January), 186–212.https://doi.org/10.1016/j.pocean.2018.01.001

Westley, K., 2021. Satellite-derived bathymetry for maritime archaeology: Testing its effectiveness at two ancient harbours in the Eastern Mediterranean. *Journal of Archaeological Science*: Reports 38 https://doi.org/10.1016/j.jasrep.2021.10303

White, S. A., Parrish, C. E., Calder, B. R., Pe'eri, S. and Rzhanov, Y., 2011. LIDAR-Derived National Shoreline: Empirical and Stochastic Uncertainty Analyses. *Journal of Coastal Research*: Special Issue 62: 62 – 74.

Yunus, A.P.; Dou, J.; Song, X.; Avtar, R., 2019. Improved Bathymetric Mapping of Coastal and Lake Environments Using Sentinel-2 and Landsat-8 Images. *Sensors 19*, 2788. https://doi.org/10.3390/s19122788

APPENDIX

Python Codes

```
#Reading the radiance of the blue and green bands and depth.

import os
from datetime import datetime, timezone
import matplotlib.pyplot as plt
from numpy import pi, cos, sin, log, exp
import numpy as np

class Depth:
    """A Class for Data"""

    def __init__(self):
        self.data_path = str()
        self.metadata = dict()
        self.w = list()   #index
        self.x = list()   #radiance blue band
        self.y = list()   #radiance green band
        self.z = list()   #depth

    def read_jhc_file(self, fullpath):

        if os.path.exists(fullpath):
            self.metadata["Source File"] = fullpath
            print('Opening water level data file:' + fullpath)
        else:
            raise RuntimeError('Unable to locate the input file'
+ fullpath)
        d_file = open(fullpath)
        d_content = d_file.read()
        d_file.close
        d_lines = d_content.splitlines()
        count = 0
        for d_line in d_lines:
            observations = d_line.split()
            self.w.append(float(observations[0]))
            self.x.append(float(observations[1]))
            self.y.append(float(observations[2]))
            self.z.append(float(observations[3]))
            count += 1
```

```
     #Importing modules

     from PIL import Image
     from IPython.display import display

     %reload_ext autoreload
     %autoreload 3
     %matplotlib inline

     import math
     import sys
     import os
     import matplotlib
     import matplotlib.pyplot as plt
     import matplotlib.ticker as ticker
     from scipy.optimize import curve_fit
     import numpy as np
     from numpy import log as ln
     from collections import Counter
     import pandas as pd
     from osgeo import gdal
     from osgeo import ogr
     from osgeo import osr

     #Importing Sentinel-2 images

     sys.path.append(os.getcwd())  # add the current folder to the
list of paths where Python looks for modules
     np.set_printoptions(precision=2, floatmode='fixed')
     print("GDAL version: %s" % (gdal.__version__, ))

     bag_path_B_AC = os.path.join(os.getcwd(), "Sentinel",
"B_AC.tif")
     bag_path_G_AC = os.path.join(os.getcwd(), "Sentinel",
"G_AC.tif") # the 'sh2007.bag' file is located under the `data` folder

     dataset_B_AC = gdal.Open(bag_path_B_AC, gdal.GA_ReadOnly)
     if not dataset_B_AC:
         raise RuntimeError("Issue in opening the BAG file: %s" %
bag_path_B_AC)
     print("BAG file was successfully opened: %s" % (bag_path_B_AC,))

     dataset_G_AC = gdal.Open(bag_path_G_AC, gdal.GA_ReadOnly)
     if not dataset_G_AC:
         raise RuntimeError("Issue in opening the BAG file: %s" %
bag_path_G_AC)
     print("BAG file was successfully opened: %s" % (bag_path_G_AC,))


     #Reading control points

     #Reading calibration points
     from mycode.SDB_Grid import Depth
     abs_path=os.path.abspath(os.path.curdir)+"/Data/HORIZONTAL/"

     Data = Depth()
```

```
Data.read_jhc_file(abs_path+"DRY_RANGE_HOR_9.txt")

#Transforming list in vector
gr = np.c_[Data.w]
Lb = np.c_[Data.x]
Lg = np.c_[Data.y]
z = np.c_[Data.z]

#Reading validation points
from mycode.SDB_Grid import Depth
abs_path=os.path.abspath(os.path.curdir)+"/Data/HORIZONTAL/"

Data = Depth()
Data.read_jhc_file(abs_path+"DRY_VAL_HOR_9.txt")

#Transforming list in vector
grV = np.c_[Data.w]
LbV = np.c_[Data.x]
LgV = np.c_[Data.y]
zV= np.c_[Data.z]
```

```
#Image settings

print("Projection:")
projection_B_AC = dataset_B_AC.GetProjection()
print("- WKT:\n%s" % projection_B_AC)

srs_B_AC = osr.SpatialReference(wkt=projection_B_AC)
projection_name_B_AC = srs_B_AC.GetAttrValue('projcs')
print("- name: %s" % projection_name_B_AC)

B_AC= dataset_B_AC.GetRasterBand(1)
B_AC_P2_nodata = B_AC.GetNoDataValue()
B = B_AC.ReadAsArray()

G_AC = dataset_G_AC.GetRasterBand(1)
G_AC_P2_nodata = G_AC.GetNoDataValue()
G = G_AC.ReadAsArray()
```

```
    # Per-pixel vector solution application

            # Columns first
             B_splice = B[i*n:(i+1)*n, j*m:(j+1)*m]
             G_splice = G[i*n:(i+1)*n, j*m:(j+1)*m]

            # Rows first
             B_splice = B[j*n:(j+1)*n, i*m:(i+1)*m]
             G_splice = G[j*n:(j+1)*n, i*m:(i+1)*m]

            # Columns first with origin in the lower left corner
            B_splice =  B[(((i)*n)):(((i+1)*n)), j*m:(j+1)*m]
            G_splice =  G[(((i)*n)):(((i+1)*n)), j*m:(j+1)*m]
            Dep = np.zeros((n,m))

            for l in range(n):
                for c in range (m):
                    num=(m0*(ln((B_splice[l,c]-
Lw1)/(G_splice[l,c]-Lw2)))) + m1
                    Dep[l,c] = num
                    DepM[(((i)*n)):(((i+1)*n)),   j*m:(j+1)*m]   =
Dep[:,:]
            countV = np.count_nonzero(grV==gr_value)
            Z_SDB_V=(m0*ln((LbV[grV==gr_value]-
Lw1)/(LgV[grV==gr_value]-Lw2)))+m1
            Diff = Z_SDB_V - zV[grV==gr_value]
            quadrado = Diff*Diff
            soma = sum(quadrado)

            if countV >0:
                RMSE_V = (soma/countV)**0.5
            print (XA)
            print (RMSE20)
            print (RMSE_V)
            plt.figure()

    plt.figure()
    plt.imshow(DepM)

    image_Full_Poly_1_1_NL  =  os.path.join(os.getcwd(),  "Output",
"SDB_VI_2019_9_NL.tif")
    driver = gdal.GetDriverByName('GTiff')
    dataset  =  driver.Create(image_Full_Poly_1_1_NL,  450,  450,  1,
gdal.GDT_Float32)
    dataset.GetRasterBand(1).WriteArray(DepM)

    dataset.SetGeoTransform(dataset_B_AC.GetGeoTransform())
##sets same geotransform as input
    dataset.SetProjection(dataset_B_AC.GetProjection()) ##sets same
projection as input

    dataset.FlushCache()
    dataset = None
    band = None
    dataset_B_AC = None
```

```
#Application of geographic model - HORIZONTAL AND EXTENDED DIERSSEN
#Initial vector (X0)

#Definition

import numpy as np
u = 2
u1 = 1
u4 = 4

gr_nr= int(np.amax(gr)) #number of groups

A_list = list()
U_list = list()
X_list = list()
V_list = list()
z_list = list()
N_list = list()
RMSE_list = list()

A1_list = list()
U1_list = list()
X1_list = list()
V1_list = list()
N1_list = list()
RMSE1_list = list()

L0_list = list()
L_list = list()

n=150
m=150

x_dim = dataset_B_AC.RasterXSize
y_dim = dataset_B_AC.RasterYSize

n_grids_x = x_dim//n
n_grids_y = y_dim//m

DepM = np.zeros((x_dim,y_dim))

for j in range(0,n_grids_y):
    for i in range(0,n_grids_x):
        gr_value =  i + (n_grids_y)*j
        count  =  np.count_nonzero(gr==gr_value)  #  number  of control points in group
        A = np.zeros((count,u))
        z_grp = z[gr==gr_value]

        A[:, 0] = ln(Lb[gr==gr_value] / Lg[gr==gr_value])
        A[:, 1] = 1
        N=A.T@A
        U=A.T@-z_grp
        inv_N = np.linalg.inv(N)
        X= -inv_N@U
        V= (A@X) - z_grp
        RMSE = ((V.T@V)/(count-u))**(1/2)

        U_list.append(U)
        N_list.append(N)
        X_list.append(X)
        V_list.append(V)
        A_list.append(A)
        z_list.append(z_grp)
        RMSE_list.append(RMSE)

        # X01 and X04 → m0 and m1

        Lw1=0
        Lw2=0
        L0=np.zeros((count,1))
        L0=(((X[0]*(ln((Lb[gr==gr_value]-Lw1)/(Lg[gr==gr_value]-Lw2))) + X[1])))
        L = L0 - z_grp
        A1=np.zeros((count,u))
        A1[:, 0]= (-X[0] / (Lb[gr==gr_value] - Lw1))
        A1[:, 1]= (X[0] / (Lg[gr==gr_value] - Lw2))
        N1=A1.T@A1
        U1=A1.T@ L
        inv_N1 = np.linalg.inv(N1)
        X1= -inv_N1@U1
        V1= (A1@X1) + L
        RMSE1 = ((V1.T@V1)/(count-u))**(1/2)
        U1_list.append(U1)
        N1_list.append(N1)
        X1_list.append(X1)
        V1_list.append(V1)
        A1_list.append(A1)
        RMSE1_list.append(RMSE1)

        # X02 and X03 → Lw1 and Lw2

        X0=np.array([X[0],X1[0],X1[1],X[1]])   #[m0 , Lw1, Lw2, m1]

        # First iteration

        RMSE30 = -10
        iteracao = 1
        RMSE20 = 10

        while RMSE30 < RMSE20:
            m0 = X0[0]
            Lw1 = X0[1]
            Lw2 = X0[2]
            m1 = X0[3]
            L0 = np.zeros((count,1))
            L0 = ((X0[0]*(n((Lb[gr==gr_value]-X0[1])/(Lg[gr==gr_value]-X0[2])))+X0[3]))
            L = L0-z_grp
```

```
                 A2=np.zeros((count,u4))
                 A2[:, 0] = ln( (Lb[gr==gr_value] - Lw1 )/
((Lg[gr==gr_value]) - Lw2) )
                 A2[:, 1]= -m0/(Lb[gr==gr_value]- Lw1)
                 A2[:, 2]= m0/(Lg[gr==gr_value] - Lw2)
                 A2[:, 3]= 1

                 N2=A2.T@A2
                 U2=A2.T@L
                 inv_N2 = np.linalg.inv(N2)
                 X2= -inv_N2@U2
                 XA= X0 + X2
                 V2= (A2@X2) + L
                 RMSE20 = ((V2.T@V2)/(count-u4))**(1/2)

                 # Second iteration

                 L01 =np.zeros((count,1))
                 L01= ((XA[0] * ( ln(( Lb[gr==gr_value] - XA[1]) /
(Lg[gr==gr_value] - XA[2]))) + XA[3] ))

                 L1 = L01 - z_grp
                 m0 = XA[0]
                 Lw1 = XA[1]
                 Lw2 = XA[2]
                 m1 = XA[3]

                 A3=np.zeros((count,u4))
                 A3[:, 0] = ln( (Lb[gr==gr_value] - Lw1 )/
((Lg[gr==gr_value]) - Lw2) )
                 A3[:, 1]= -m0/(Lb[gr==gr_value]- Lw1)
                 A3[:, 2]= m0/(Lg[gr==gr_value] - Lw2)
                 A3[:, 3]= 1
                 N3=A3.T@A3
                 U3=A3.T@L1
                 inv_N3 = np.linalg.inv(N3)
                 X3= -inv_N3@U3
                 XA1= XA + X3
                 V3= (A3@X3) + L1
                 RMSE30 = ((V3.T@V3)/(count-u4))**(1/2)

           #Solution Vector (X)

                 iteracao = iteracao +1
                 X0=XA

                 sigma3= (V3.T@V3) / (count-u)
                 MVC3 = sigma3*inv_N3
```

```
#Application of geographic model – HORIZONTAL AND DIERSSEN

#Solution Vector (X)

for j in range(0,n_grids_y):
    for i in range(0,n_grids_x):
        gr_value =  i + (n_grids_y)*j
        count  =  np.count_nonzero(gr==gr_value)  #  number  of control points in group
        A = np.zeros((count,u))

        z_grp = z[gr==gr_value]

        A[:, 0] = ln(Lb[gr==gr_value] / Lg[gr==gr_value])
        A[:, 1] = 1
        N=A.T@A
        U=A.T@-z_grp
        inv_N = np.linalg.inv(N)
        X= -inv_N@U
        V= (A@X) - z_grp
        RMSE = ((V.T@V)/(count-u))**(1/2)

        # X01 e X04

        m0 = X[0]
        m1 = X[1]
```

```
#Application of geographic model – VERTICAL AND EXTENDED DIERSSEN

#Definition

import numpy as np
u = 2
u1 = 1
u4 = 4

gr_nr= int(np.amax(gr))     # number of groups

A_list = list()
U_list = list()
X_list = list()
V_list = list()
z_list = list()
N_list = list()
RMSE_list = list()

A1_list = list()
U1_list = list()
X1_list = list()
V1_list = list()
N1_list = list()
RMSE1_list = list()

AR1_list = list()
UR1_list = list()
XR1_list = list()
VR1_list = list()
NR1_list = list()
zR1_list = list()
RMSER1_list = list()

AR22_list = list()
UR22_list = list()
XR22_list = list()
VR22_list = list()
NR22_list = list()
zR22_list = list()
RMSER22_list = list()

L0_list = list()
L_list = list()
L0R1_list = list()
LR1_list = list()

ERROR= list()
ERROR1= list()
ERROR2= list()
ERROR3= list()
ERROR4= list()
ERROR5= list()
ERROR6= list()
ERROR7= list()
```

```
ERROR8= list()

Diff_list = list()

# RANGE OF 0 - 5

# Initial vector (X0)

for gr_value in range(0, gr_nr+1):

    n =np.count_nonzero(gr==gr_value) #n. of control points in group

    A = np.zeros((n,u))

    z_grp = z[gr==gr_value]

    A[:, 0] = ln(Lb[gr==gr_value] / Lg[gr==gr_value])
    A[:, 1] = 1
    N=A.T@A
    U=A.T@-z_grp
    inv_N = np.linalg.inv(N)
    X= -inv_N@U
    V= (A@X) - z_grp
    RMSE = ((V.T@V)/(n-u))**(1/2)

    U_list.append(U)
    N_list.append(N)
    X_list.append(X)
    V_list.append(V)
    A_list.append(A)
    z_list.append(z_grp)
    RMSE_list.append(RMSE)

    # X01 and X04

    Lw1=0
    Lw2=0

    L0 =np.zeros((n,1))
    L0 = (((X[0] * ( ln(( Lb[gr==gr_value] - Lw1) / (Lg[gr==gr_value] - Lw2))) + X[1] )))
    L = L0 - z_grp

    A1=np.zeros((n,u))
    A1[:, 0] = (-X[0] / (Lb[gr==gr_value] - Lw1))
    A1[:, 1]= (X[0] / (Lg[gr==gr_value] - Lw2))
    N1=A1.T@A1
    U1=A1.T@ L
    inv_N1 = np.linalg.inv(N1)
    X1= -inv_N1@U1
    V1= (A1@X1) + L
    RMSE1 = ((V1.T@V1)/(n-u))**(1/2)

    U1_list.append(U1)
    N1_list.append(N1)
    X1_list.append(X1)
```

```
        V1_list.append(V1)
        A1_list.append(A1)
        RMSE1_list.append(RMSE1)

        # X02 and X03

        X0 = np.array([X[0], X1[0], X1[1], X[1]])   #[m0 , Lw1, Lw2,
m1]

        # First iteration

        RMSE30 = -10
        iteracao = 1
        RMSE20 = 10

        while RMSE30 < RMSE20:

            m0 = X0[0]
            Lw1 = X0[1]
            Lw2 = X0[2]
            m1 = X0[3]

            L0 =np.zeros((n,1))
            L0  =  ((X0[0]  *  (  ln((  Lb[gr==gr_value]  -  X0[1])  /
(Lg[gr==gr_value] - X0[2]))) + X0[3] ))
            L = L0 -  z_grp

            A2=np.zeros((n,u4))
            A2[:,   0]   =   ln(   (Lb[gr==gr_value]   -   Lw1   )/
((Lg[gr==gr_value]) - Lw2) )
            A2[:, 1]= -m0/(Lb[gr==gr_value]- Lw1)
            A2[:, 2]= m0/(Lg[gr==gr_value] - Lw2)
            A2[:, 3]= 1

            N2=A2.T@A2
            U2=A2.T@L
            inv_N2 = np.linalg.inv(N2)
            X2= -inv_N2@U2

            XA= X0 + X2
            V2= (A2@X2) + L
            RMSE20 = ((V2.T@V2)/(n-u4))**(1/2)

             # Second iteration

            L01 =np.zeros((n,1))
            L01=  ((XA[0]  *  (  ln((  Lb[gr==gr_value]  -  XA[1])  /
(Lg[gr==gr_value] - XA[2]))) + XA[3] ))
            L1 = L01 - z_grp

            m0 = XA[0]
            Lw1 = XA[1]
            Lw2 = XA[2]
            m1 = XA[3]

            A3=np.zeros((n,u4))
            A3[:, 0]=ln((Lb[gr==gr_value]-Lw1)/((Lg[gr==gr_value])-Lw2))
            A3[:, 1]= -m0/(Lb[gr==gr_value]- Lw1)
            A3[:, 2]= m0/(Lg[gr==gr_value] - Lw2)
            A3[:, 3]= 1

            N3=A3.T@A3
            U3=A3.T@L1
            inv_N3 = np.linalg.inv(N3)
            X3= -inv_N3@U3

            XA1= XA + X3
            V3= (A3@X3) + L1
            RMSE30 = ((V3.T@V3)/(n-u4))**(1/2)

            # Solution Vector (X)
            iteracao = iteracao +1
            X0=XA

            sigma3= (V3.T@V3) / (n-u)
            MVC3 = sigma3*inv_N3

            count = np.count_nonzero(gr5==gr_value)
            Z_SDB_5 = (XA[0]*ln ((Lb5[gr5==gr_value] -XA[1])/(Lg5[gr5==gr_value] - XA[2])))+XA[3]
            Diff = Z_SDB_5 - z5[gr5==gr_value]
            quadrado = Diff*Diff
            soma = sum(quadrado)

            if count >0:
                RMSE_5 = (soma/count)**0.5

            Diff_list.append(Diff)

        ERROR.append(RMSE_5)


        print( "\033[0;35m" +  "\033[1m" + "Solution of Group: " + str((gr_value * 5))  +  " A " + str(((gr_value *5) +5)))
        print ("\033[0m" + "RMSE: " + str(RMSE20))
        print ("\033[0m" + "RMSE_VAL: " + str(RMSE_5))
        print ("X: " + str(XA) + "\n ")

    range0_5_i5 = ERROR[0]
    range5_10_i5 = ERROR[1]
    range10_15_i5 = ERROR[2]
    range15_20_i5 = ERROR[3]

    # RANGE OF 0 - 10

    # Initial vector (X0)

    for gr_value in range(0, gr_nr):

        n11 = np.count_nonzero(gr==gr_value) #n. of control points in group
        n22 = np.count_nonzero(gr==gr_value+1)
        n = n11+n22

        z_grp1 =np.zeros((n11,1))
        z_grp2 =np.zeros((n22,1))
        z_grpR1 =np.zeros((n,1))

        z_grp1[:,0] = z[gr==gr_value]
        z_grp2[:,0] = z[gr==gr_value+1]
        z_grpR1[:,0] = np.concatenate([z_grp1[:, 0],z_grp2[:, 0]])

        A11 = np.zeros((n11,u))
        A22 = np.zeros((n22,u))
        AR1 = np.zeros((n,u))

        A11[:, 0] = ln(Lb[gr==gr_value] / Lg[gr==gr_value])
        A22[:, 0] = ln(Lb[gr==gr_value+1] / Lg[gr==gr_value+1])

        AR1[:, 0]= np.concatenate([A11[:, 0],A22[:, 0]])
        AR1[:, 1] = 1

        NR1=AR1.T@AR1
        UR1=AR1.T@-z_grpR1
        inv_NR1 = np.linalg.inv(NR1)
        XR1= -inv_NR1@UR1
        VR1= (AR1@XR1) - z_grpR1
        RMSER1 = ((VR1.T@VR1)/(n-u))**(1/2)

        UR1_list.append(UR1)
        NR1_list.append(NR1)
        XR1_list.append(XR1)
        VR1_list.append(VR1)
        AR1_list.append(AR1)
        zR1_list.append(z_grpR1)

        # X01 and X04

        Lw1=0
        Lw2=0

        L01 =np.zeros((n11,1))
        L02 =np.zeros((n22,1))
        L0R1=np.zeros((n,1))

        L01[:,0] =  (((XR1[0] * ( ln(( Lb[gr==gr_value] - Lw1) /
(Lg[gr==gr_value] - Lw2))) + XR1[1] )))
        L02[:,0] =  (((XR1[0] * ( ln(( Lb[gr==gr_value+1] - Lw1) /
(Lg[gr==gr_value+1] - Lw2))) + XR1[1] )))
        L0R1[:,0] = np.concatenate([L01[:, 0],L02[:, 0]])

        LR1 = L0R1 - z_grpR1

        A33 = np.zeros((n11,u))
        A44 = np.zeros((n22,u))
        A1R1 = np.zeros((n,u))
        A33[:, 0] = (-XR1[0] / (Lb[gr==gr_value] - Lw1))
        A44[:, 0] = (-XR1[0] / (Lb[gr==gr_value+1] - Lw1))
        A33[:, 1]= (XR1[0] / (Lg[gr==gr_value] - Lw2))
        A44[:, 1]= (XR1[0] / (Lg[gr==gr_value+1] - Lw2))
        A1R1[:, :]= np.concatenate([A33[:,:],A44[:, :]])

        N1R1=A1R1.T@A1R1
        U1R1=A1R1.T@ LR1
        inv_N1R1 = np.linalg.inv(N1R1)
        X1R1= -inv_N1R1@U1R1
        V1R1= (A1R1@X1R1) + LR1
        RMSE1R1 = ((V1R1.T@V1R1)/(n-u))**(1/2)

        U1_list.append(U1)
        N1_list.append(N1)
        X1_list.append(X1)
        V1_list.append(V1)
        A1_list.append(A1)
        RMSE1_list.append(RMSE1R1)

        # X02 and X03

        X0R1=np.array([XR1[0],X1R1[0],X1R1[1],XR1[1]])  #[m0  ,Lw1,
Lw2, m1]

        # First iteration

        RMSE30R1 = -10
        iteracao = 1
        RMSE20R1 = 10

        while RMSE30R1 < RMSE20R1:

            m0R1 = X0R1[0]
            Lw1R1 = X0R1[1]
            Lw2R1 = X0R1[2]
            m1R1 = X0R1[3]

            L01 =np.zeros((n11,1))
            L02 =np.zeros((n22,1))
            L0R1=np.zeros((n,1))

            L01[:,0]  =  ((X0R1[0]  *  (  ln((  Lb[gr==gr_value]  -
X0R1[1]) / (Lg[gr==gr_value] - X0R1[2]))) + X0R1[3] ))
            L02[:,0]  =  ((X0R1[0]  *  (  ln((  Lb[gr==gr_value+1]  -
X0R1[1]) / (Lg[gr==gr_value+1] - X0R1[2]))) + X0R1[3] ))
            L0R1[:,0] = np.concatenate([L01[:, 0],L02[:, 0]])
            LR1 = L0R1 -  z_grpR1

            A22R1 = np.zeros((n22,u4))
            A11R1 = np.zeros((n11,u4))
            A2R1 = np.zeros((n,u4))

            A11R1[:,  0]  =  ln(  (Lb[gr==gr_value]  -  Lw1R1  )/
((Lg[gr==gr_value]) - Lw2R1) )
            A11R1[:, 1]= -m0R1/(Lb[gr==gr_value]- Lw1R1)
            A11R1[:, 2]= m0R1/(Lg[gr==gr_value] - Lw2R1)
            A11R1[:, 3]= 1

            A22R1[:,  0]  =  ln(  (Lb[gr==gr_value+1]  -  Lw1R1  )/
((Lg[gr==gr_value+1]) - Lw2R1) )
            A22R1[:, 1]= -m0R1/(Lb[gr==gr_value+1]- Lw1R1)
            A22R1[:, 2]= m0R1/(Lg[gr==gr_value+1] - Lw2R1)
            A22R1[:, 3]= 1

            A2R1[:, :]= np.concatenate([A11R1[:,:],A22R1[:, :]])

            N2R1=A2R1.T@A2R1
            U2R1=A2R1.T@LR1
            inv_N2R1 = np.linalg.inv(N2R1)
            X2R1= -inv_N2R1@U2R1

            XAR1= X0R1 + X2R1
            V2R1= (A2R1@X2R1) + LR1


            # Second iteration

            L01R1 =np.zeros((n11,1))
            L02R1 =np.zeros((n22,1))
            L00R1=np.zeros((n,1))

            L01R1[:,0]=  ((XAR1[0]  *  (  ln((  Lb[gr==gr_value]  -
XAR1[1]) / (Lg[gr==gr_value] - XAR1[2]))) + XAR1[3] ))
            L02R1[:,0]=  ((XAR1[0]  *  (  ln((  Lb[gr==gr_value+1]  -
XAR1[1]) / (Lg[gr==gr_value+1] - XAR1[2]))) + XAR1[3] ))
            L00R1[:,0] = np.concatenate([L01R1[:, 0],L02R1[:, 0]])
            L1R1 = L00R1 - z_grpR1

            m0 = XAR1[0]
            Lw1 = XAR1[1]
            Lw2 = XAR1[2]
            m1 = XAR1[3]

            A44R1 = np.zeros((n22,u4))
            A33R1 = np.zeros((n11,u4))
            A3R1 = np.zeros((n,u4))

            A33R1[:,  0]  =  ln(  (Lb[gr==gr_value]  -  Lw1  )/
((Lg[gr==gr_value]) - Lw2) )
            A33R1[:, 1]= -m0/(Lb[gr==gr_value]- Lw1)
            A33R1[:, 2]= m0/(Lg[gr==gr_value] - Lw2)
            A33R1[:, 3]= 1

            A44R1[:,  0]  =  ln(  (Lb[gr==gr_value+1]  -  Lw1  )/
((Lg[gr==gr_value+1]) - Lw2) )
            A44R1[:, 1]= -m0/(Lb[gr==gr_value+1]- Lw1)
            A44R1[:, 2]= m0/(Lg[gr==gr_value+1] - Lw2)
            A44R1[:, 3]= 1

            A3R1[:, :]= np.concatenate([A33R1[:,:],A44R1[:, :]])
            N3R1=A3R1.T@A3R1
            U3R1=A3R1.T@L1R1
            inv_N3R1 = np.linalg.inv(N3R1)
            X3R1= -inv_N3R1@U3R1

            XA1R1= XAR1 + X3R1
            V3R1= (A3R1@X3R1) + L1R1
            RMSE30R1 = ((V3R1.T@V3R1)/(n-u4))**(1/2)

            iteracao = iteracao +1
            X0R1=XAR1

            c1 = np.count_nonzero(gr5==gr_value)
            c2 = np.count_nonzero(gr5==gr_value +1 )
            count = c1 + c2

            Z_SDB_10_1 =np.zeros((c1,1))
            Z_SDB_10_2 =np.zeros((c2,1))
            Z_SDB_10=np.zeros((count,1))

            Z_SDB_10_1[:,0]  =  (XAR1[0]*ln  ((Lb5[gr5==gr_value]  -
XAR1[1])/(Lg5[gr5==gr_value] - XAR1[2])))+XAR1[3]
            Z_SDB_10_2[:,0]  =  (XAR1[0]*ln  ((Lb5[gr5==gr_value+1]  -
XAR1[1])/(Lg5[gr5==gr_value+1] - XAR1[2])))+XAR1[3]
            Z_SDB_10[:,0]     =     np.concatenate([Z_SDB_10_1[:,
0],Z_SDB_10_2[:, 0]])

            z_10_1 =np.zeros((c1,1))
            z_10_2 =np.zeros((c2,1))
            z_10 =np.zeros((count,1))

            z_10_1[:,0] = z5[gr5==gr_value]
            z_10_2[:,0] = z5[gr5==gr_value+1]
            z_10[:,0] = np.concatenate([z_10_1[:, 0],z_10_2[:, 0]])

            Diff_10 = Z_SDB_10 - z_10
            quadrado = Diff_10*Diff_10
            soma = sum(quadrado)

            if count >0:
                RMSE_10 = (soma/count)**0.5

            Diff_list.append(Diff)

        ERROR1.append(RMSE_10)

        print( "\033[0;35m" +  "\033[1m"  + "Solution of Group: " +
str((gr_value * 5))  +  " A " + str(((gr_value *5) +10)))
        print ("\033[0m" + "RMSE: " + str(RMSE20R1))
        print ("\033[0m" + "RMSE_VAL: " + str(RMSE_10))
        print ("X: " + str(XAR1) + "\n ")

    range0_10_i10 = ERROR1[0]
    range5_15_i10 = ERROR1[1]
    range10_20_i10 = ERROR1[2]
```

```
      # RANGE OF 0 - 15

      # Initial vector (X0)

      for gr_value in range(0, gr_nr-1):

          n33= np.count_nonzero(gr==gr_value)# n. of control points in
group
          n44 = np.count_nonzero(gr==gr_value+1)
          n55 = np.count_nonzero(gr==gr_value+2)
          n2 = n33+n44+n55

          z_grp12 =np.zeros((n33,1))
          z_grp22 =np.zeros((n44,1))
          z_grp32 =np.zeros((n55,1))
          z_grpR22 =np.zeros((n2,1))

          z_grp12[:,0] = z[gr==gr_value]
          z_grp22[:,0] = z[gr==gr_value+1]
          z_grp32[:,0] = z[gr==gr_value+2]
          z_grpR22[:,0]  =  np.concatenate([z_grp12[:,  0],z_grp22[:,
0],z_grp32[:, 0] ])

          A112 = np.zeros((n33,u))
          A222 = np.zeros((n44,u))
          A332 = np.zeros((n55,u))
          AR22 = np.zeros((n2,u))

          A112[:, 0] = ln(Lb[gr==gr_value] / Lg[gr==gr_value])
          A222[:, 0] = ln(Lb[gr==gr_value+1] / Lg[gr==gr_value+1])
          A332[:, 0] = ln(Lb[gr==gr_value+2] / Lg[gr==gr_value+2])

          AR22[:, 0]= np.concatenate([A112[:, 0],A222[:, 0], A332[:,
0]])
          AR22[:, 1] = 1

          NR22=AR22.T@AR22
          UR22=AR22.T@-z_grpR22
          inv_NR22= np.linalg.inv(NR22)
          XR22= -inv_NR22@UR22
          VR22= (AR22@XR22) - z_grpR22
          RMSER22 = ((VR22.T@VR22)/(n2-u))**(1/2)

          UR22_list.append(UR22)
          NR22_list.append(NR22)
          XR22_list.append(XR22)
          VR22_list.append(VR22)
          AR22_list.append(AR22)
          zR22_list.append(z_grpR22)

          # X01 and X04

          Lw1=0
          Lw2=0

          L012 =np.zeros((n33,1))
```

```
        L022 =np.zeros((n44,1))
        L032 =np.zeros((n55,1))
        L0R22=np.zeros((n2,1))

        L012[:,0] =  (((XR22[0] * ( ln(( Lb[gr==gr_value] - Lw1) /
(Lg[gr==gr_value] - Lw2))) + XR22[1] )))
        L022[:,0] =  (((XR22[0] * ( ln(( Lb[gr==gr_value+1] - Lw1)
/ (Lg[gr==gr_value+1] - Lw2))) + XR22[1] )))
        L032[:,0] =  (((XR22[0] * ( ln(( Lb[gr==gr_value+2] - Lw1)
/ (Lg[gr==gr_value+2] - Lw2))) + XR22[1] )))
        L0R22[:,0] = np.concatenate([L012[:, 0],L022[:, 0],L032[:,
0] ])

        LR22 = L0R22 - z_grpR22

        A33 = np.zeros((n33,u))
        A44 = np.zeros((n44,u))
        A55 = np.zeros((n55,u))
        A1R22 = np.zeros((n2,u))

        A33[:, 0] = (-XR22[0] / (Lb[gr==gr_value] - Lw1))
        A44[:, 0] = (-XR22[0] / (Lb[gr==gr_value+1] - Lw1))
        A55[:, 0] = (-XR22[0] / (Lb[gr==gr_value+2] - Lw1))

        A33[:, 1]= (XR22[0] / (Lg[gr==gr_value] - Lw2))
        A44[:, 1]= (XR22[0] / (Lg[gr==gr_value+1] - Lw2))
        A55[:, 1]= (XR22[0] / (Lg[gr==gr_value+2] - Lw2))

        A1R22[:, :]= np.concatenate([A33[:,:],A44[:, :], A55[:, :]])

        N1R22=A1R22.T@A1R22
        U1R22=A1R22.T@ LR22
        inv_N1R22 = np.linalg.inv(N1R22)
        X1R22= -inv_N1R22@U1R22
        V1R22= (A1R22@X1R22) + LR22
        RMSE1R22 = ((V1R22.T@V1R22)/(n2-u))**(1/2)

        # X02 and X03


X0R22=np.array([XR22[0],X1R22[0],X1R22[1],XR22[1]])#[m0,Lw1,Lw2, m1]

        # First iteration

        RMSE30R2 = -10
        iteracao = 1
        RMSE20R2 = 10

        while RMSE30R2 < RMSE20R2:
            m0R22 = X0R22[0]
            Lw1R22 = X0R22[1]
            Lw2R22 = X0R22[2]
            m1R22 = X0R22[3]

            L012 =np.zeros((n33,1))
            L022 =np.zeros((n44,1))
            L032 =np.zeros((n55,1))
            L0R22=np.zeros((n2,1))

            L012[:,0] = ((X0R22[0] * ( ln(( Lb[gr==gr_value] -
X0R22[1]) / (Lg[gr==gr_value] - X0R22[2]))) + X0R22[3] ))
            L022[:,0] = ((X0R22[0] * ( ln(( Lb[gr==gr_value+1] -
X0R22[1]) / (Lg[gr==gr_value+1] - X0R22[2]))) + X0R22[3] ))
            L032[:,0] = ((X0R22[0] * ( ln(( Lb[gr==gr_value+2] -
X0R22[1]) / (Lg[gr==gr_value+2] - X0R22[2]))) + X0R22[3] ))

            L0R22[:,0]                                            =
np.concatenate([L012[:,0],L022[:,0],L032[:,0]])
            LR22 = L0R22 -  z_grpR22

            A33R22 = np.zeros((n33,u4))
            A44R22 = np.zeros((n44,u4))
            A55R22 = np.zeros((n55,u4))

            A2R22 = np.zeros((n2,u4))

            A33R22[:,  0]  =  ln(  (Lb[gr==gr_value]  -  Lw1R22  )/
((Lg[gr==gr_value]) - Lw2R22) )
            A33R22[:, 1]= -m0R22/(Lb[gr==gr_value]- Lw1R22)
            A33R22[:, 2]= m0R22/(Lg[gr==gr_value] - Lw2R22)
            A33R22[:, 3]= 1

            A44R22[:,  0]  =  ln(  (Lb[gr==gr_value+1]  -  Lw1R22  )/
((Lg[gr==gr_value+1]) - Lw2R22) )
            A44R22[:, 1]= -m0R22/(Lb[gr==gr_value+1]- Lw1R22)
            A44R22[:, 2]= m0R22/(Lg[gr==gr_value+1] - Lw2R22)
            A44R22[:, 3]= 1

            A55R22[:,  0]  =  ln(  (Lb[gr==gr_value+2]  -  Lw1R22  )/
((Lg[gr==gr_value+2]) - Lw2R22) )
            A55R22[:, 1]= -m0R22/(Lb[gr==gr_value+2]- Lw1R22)
            A55R22[:, 2]= m0R22/(Lg[gr==gr_value+2] - Lw2R22)
            A55R22[:, 3]= 1

            A2R22[:,  :]= np.concatenate([A33R22[:,:],A44R22[:,  :],
A55R22[:, :] ])

            N2R22=A2R22.T@A2R22
            U2R22=A2R22.T@LR22
            inv_N2R22 = np.linalg.inv(N2R22)
            X2R22= -inv_N2R22@U2R22

            XAR22= X0R22 + X2R22
            V2R22= (A2R22@X2R22) + LR22
            RMSE20R2 = ((V2R22.T@V2R22)/(n2-u4))**(1/2)

            # Second iteration

            L01R22 =np.zeros((n33,1))
            L02R22 =np.zeros((n44,1))
            L03R22 =np.zeros((n55,1))
            L00R22=np.zeros((n2,1))
            L01R22[:,0]= ((XAR22[0] * ( ln(( Lb[gr==gr_value] -
XAR22[1]) / (Lg[gr==gr_value] - XAR22[2]))) + XAR22[3] ))
            L02R22[:,0]= ((XAR22[0] * ( ln(( Lb[gr==gr_value+1] -
XAR22[1]) / (Lg[gr==gr_value+1] - XAR22[2]))) + XAR22[3] ))
            L03R22[:,0]= ((XAR22[0] * ( ln(( Lb[gr==gr_value+2] -
XAR22[1]) / (Lg[gr==gr_value+2] - XAR22[2]))) + XAR22[3] ))
            L00R22[:,0]  = np.concatenate([L01R22[:,  0],L02R22[:,
0],L03R22[:, 0] ])
            L1R22 = L00R22 - z_grpR22

            m0 = XAR22[0]
            Lw1 = XAR22[1]
            Lw2 = XAR22[2]
            m1 = XAR22[3]

            A33R222 = np.zeros((n33,u4))
            A44R222 = np.zeros((n44,u4))
            A55R222 = np.zeros((n55,u4))
            A3R222 = np.zeros((n2,u4))

            A33R222[:,  0]  =  ln(  (Lb[gr==gr_value]  -  Lw1  )/
((Lg[gr==gr_value]) - Lw2) )
            A33R222[:, 1]= -m0/(Lb[gr==gr_value]- Lw1)
            A33R222[:, 2]= m0/(Lg[gr==gr_value] - Lw2)
            A33R222[:, 3]= 1

            A44R222[:,  0]  =  ln(  (Lb[gr==gr_value+1]  -  Lw1  )/
((Lg[gr==gr_value+1]) - Lw2) )
            A44R222[:, 1]= -m0/(Lb[gr==gr_value+1]- Lw1)
            A44R222[:, 2]= m0/(Lg[gr==gr_value+1] - Lw2)
            A44R222[:, 3]= 1

            A55R222[:,  0]  =  ln(  (Lb[gr==gr_value+2]  -  Lw1  )/
((Lg[gr==gr_value+2]) - Lw2) )
            A55R222[:, 1]= -m0/(Lb[gr==gr_value+2]- Lw1)
            A55R222[:, 2]= m0/(Lg[gr==gr_value+2] - Lw2)
            A55R222[:, 3]= 1

            A3R222[:,  :]=  np.concatenate([A33R222[:,:],A44R222[:,
:],A55R222[:, :] ])

            N3R22=A3R222.T@A3R222
            U3R22=A3R222.T@L1R22
            inv_N3R22 = np.linalg.inv(N3R22)
            X3R22= -inv_N3R22@U3R22

            XA1R22= XAR22 + X3R22
            V3R22= (A3R222@X3R22) + L1R22
            RMSE30R2 = ((V3R22.T@V3R22)/(n2-u4))**(1/2)

            iteracao = iteracao +1
            X0R22=XAR22

            c1 = np.count_nonzero(gr5==gr_value)
            c2 = np.count_nonzero(gr5==gr_value +1 )
            c3 = np.count_nonzero(gr5==gr_value +2 )
            count = c1 + c2 +c3

            Z_SDB_15_1 =np.zeros((c1,1))
            Z_SDB_15_2 =np.zeros((c2,1))
            Z_SDB_15_3 =np.zeros((c3,1))
            Z_SDB_15=np.zeros((count,1))

            Z_SDB_15_1[:,0] = (XAR22[0]*ln ((Lb5[gr5==gr_value] -
XAR22[1])/(Lg5[gr5==gr_value] - XAR22[2])))+XAR22[3]
            Z_SDB_15_2[:,0] = (XAR22[0]*ln ((Lb5[gr5==gr_value+1] -
XAR22[1])/(Lg5[gr5==gr_value+1] - XAR22[2])))+XAR22[3]
            Z_SDB_15_3[:,0] = (XAR22[0]*ln ((Lb5[gr5==gr_value+2] -
XAR22[1])/(Lg5[gr5==gr_value+2] - XAR22[2])))+XAR22[3]
            Z_SDB_15[:,0] = np.concatenate([Z_SDB_15_1[:,
0],Z_SDB_15_2[:, 0],Z_SDB_15_3[:, 0]])

            z_15_1 =np.zeros((c1,1))
            z_15_2 =np.zeros((c2,1))
            z_15_3 =np.zeros((c3,1))
            z_15 =np.zeros((count,1))

            z_15_1[:,0] = z5[gr5==gr_value]
            z_15_2[:,0] = z5[gr5==gr_value+1]
            z_15_3[:,0] = z5[gr5==gr_value+2]
            z_15[:,0] = np.concatenate([z_15_1[:, 0],z_15_2[:,
0],z_15_3[:, 0] ])

            Diff_15 = Z_SDB_15 - z_15
            quadrado = Diff_15*Diff_15
            soma = sum(quadrado)

            if count >0:
                RMSE_15 = (soma/count)**0.5

            Diff_list.append(Diff)

        ERROR2.append(RMSE_15)

    #    print( "\033[0;35m" + "\033[1m" + "Solution of Group: "
+ str((gr_value * 5)) + " A " + str(((gr_value *5) +15)))
    #    print ("\033[0m" + "RMSE: " + str(RMSE20R2))
    #    print ("\033[0m" + "RMSE_VAL: " + str(RMSE_15))
    #    print ("X: " + str(XAR22) + "\n ")

    range0_15_i15 = ERROR2[0]
    range5_20_i15 = ERROR2[1]


    # RANGE OF 0 - 20

    # Initial vector (X0)

    for gr_value in range(0, gr_nr-2):
        n33= np.count_nonzero(gr==gr_value)# n. of control points in
group
        n44 = np.count_nonzero(gr==gr_value+1)
        n55 = np.count_nonzero(gr==gr_value+2)
        n66 = np.count_nonzero(gr==gr_value+3)
        n20 = n33+n44+n55+n66

        z_grp12 =np.zeros((n33,1))
        z_grp22 =np.zeros((n44,1))
        z_grp32 =np.zeros((n55,1))
        z_grp42 =np.zeros((n66,1))
        z_grpR20 =np.zeros((n20,1))

        z_grp12[:,0] = z[gr==gr_value]
        z_grp22[:,0] = z[gr==gr_value+1]
        z_grp32[:,0] = z[gr==gr_value+2]
        z_grp42[:,0] = z[gr==gr_value+3]
        z_grpR20[:,0]  =  np.concatenate([z_grp12[:,  0],z_grp22[:,
0],z_grp32[:, 0], z_grp42[:, 0] ])

        A112 = np.zeros((n33,u))
        A222 = np.zeros((n44,u))
        A332 = np.zeros((n55,u))
        A442 = np.zeros((n66,u))
        AR22 = np.zeros((n20,u))

        A112[:, 0] = ln(Lb[gr==gr_value] / Lg[gr==gr_value])
        A222[:, 0] = ln(Lb[gr==gr_value+1] / Lg[gr==gr_value+1])
        A332[:, 0] = ln(Lb[gr==gr_value+2] / Lg[gr==gr_value+2])
        A442[:, 0] = ln(Lb[gr==gr_value+3] / Lg[gr==gr_value+3])

        AR22[:, 0]= np.concatenate([A112[:, 0],A222[:, 0], A332[:,
0], A442[:, 0]])
        AR22[:, 1] = 1

        NR22=AR22.T@AR22
        UR22=AR22.T@-z_grpR20
        inv_NR22= np.linalg.inv(NR22)
        XR22= -inv_NR22@UR22
        VR22= (AR22@XR22) - z_grpR20
        RMSER22 = ((VR22.T@VR22)/(n20-u))**(1/2)

        UR22_list.append(UR22)
        NR22_list.append(NR22)
        XR22_list.append(XR22)
        VR22_list.append(VR22)
        AR22_list.append(AR22)
        zR22_list.append(z_grpR20)

        # X01 and X04

        Lw1=0
        Lw2=0

        L012 =np.zeros((n33,1))
        L022 =np.zeros((n44,1))
        L032 =np.zeros((n55,1))
        L042 =np.zeros((n66,1))
        L0R22=np.zeros((n20,1))

        L012[:,0] =  (((XR22[0] * ( ln(( Lb[gr==gr_value] - Lw1) /
(Lg[gr==gr_value] - Lw2))) + XR22[1] )))
        L022[:,0] =  (((XR22[0] * ( ln(( Lb[gr==gr_value+1] - Lw1)
/ (Lg[gr==gr_value+1] - Lw2))) + XR22[1] )))
        L032[:,0] =  (((XR22[0] * ( ln(( Lb[gr==gr_value+2] - Lw1)
/ (Lg[gr==gr_value+2] - Lw2))) + XR22[1] )))
        L042[:,0] =  (((XR22[0] * ( ln(( Lb[gr==gr_value+3] - Lw1)
/ (Lg[gr==gr_value+3] - Lw2))) + XR22[1] )))
        L0R22[:,0] = np.concatenate([L012[:, 0],L022[:, 0],L032[:,
0], L042[:, 0]  ])

        LR22 = L0R22 - z_grpR20

        A33 = np.zeros((n33,u))
        A44 = np.zeros((n44,u))
        A55 = np.zeros((n55,u))
        A66 = np.zeros((n66,u))
        A1R22 = np.zeros((n20,u))

        A33[:, 0] = (-XR22[0] / (Lb[gr==gr_value] - Lw1))
        A44[:, 0] = (-XR22[0] / (Lb[gr==gr_value+1] - Lw1))
        A55[:, 0] = (-XR22[0] / (Lb[gr==gr_value+2] - Lw1))
        A66[:, 0] = (-XR22[0] / (Lb[gr==gr_value+3] - Lw1))

        A33[:, 1]= (XR22[0] / (Lg[gr==gr_value] - Lw2))
        A44[:, 1]= (XR22[0] / (Lg[gr==gr_value+1] - Lw2))
        A55[:, 1]= (XR22[0] / (Lg[gr==gr_value+2] - Lw2))
        A66[:, 1]= (XR22[0] / (Lg[gr==gr_value+3] - Lw2))

        A1R22[:, :]= np.concatenate([A33[:,:],A44[:, :], A55[:, :],
A66[:, :] ])

        N1R22=A1R22.T@A1R22
        U1R22=A1R22.T@ LR22
        inv_N1R22 = np.linalg.inv(N1R22)
        X1R22= -inv_N1R22@U1R22
        V1R22= (A1R22@X1R22) + LR22
        RMSE1R22 = ((V1R22.T@V1R22)/(n20-u))**(1/2)

        # X02 and X03

        X0R22  =  np.array([XR22[0],  X1R22[0],  X1R22[1],  XR22[1]])
#[m0 , Lw1, Lw2, m1]

        # First iteration

        RMSE30R2 = -10
        iteracao = 1
        RMSE20R2 = 10

        while RMSE30R2 < RMSE20R2:
            m0R22 = X0R22[0]
            Lw1R22 = X0R22[1]
            Lw2R22 = X0R22[2]
            m1R22 = X0R22[3]

            L012 =np.zeros((n33,1))
            L022 =np.zeros((n44,1))
            L032 =np.zeros((n55,1))
            L042 =np.zeros((n66,1))
            L0R22=np.zeros((n20,1))

            L012[:,0] = ((X0R22[0] * ( ln(( Lb[gr==gr_value] - X0R22[1]) / (Lg[gr==gr_value] - X0R22[2]))) + X0R22[3] ))
            L022[:,0] = ((X0R22[0] * ( ln(( Lb[gr==gr_value+1] - X0R22[1]) / (Lg[gr==gr_value+1] - X0R22[2]))) + X0R22[3] ))
            L032[:,0] = ((X0R22[0] * ( ln(( Lb[gr==gr_value+2] - X0R22[1]) / (Lg[gr==gr_value+2] - X0R22[2]))) + X0R22[3] ))
            L042[:,0] = ((X0R22[0] * ( ln(( Lb[gr==gr_value+3] - X0R22[1]) / (Lg[gr==gr_value+3] - X0R22[2]))) + X0R22[3] ))

            L0R22[:,0] = np.concatenate([L012[:, 0],L022[:, 0],L032[:, 0], L042[:, 0] ])
            LR22 = L0R22 -  z_grpR20

            A33R22 = np.zeros((n33,u4))
            A44R22 = np.zeros((n44,u4))
            A55R22 = np.zeros((n55,u4))
            A66R22 = np.zeros((n66,u4))

            A2R22 = np.zeros((n20,u4))

            A33R22[:, 0] = ln( (Lb[gr==gr_value] - Lw1R22 )/ ((Lg[gr==gr_value]) - Lw2R22) )
            A33R22[:, 1]= -m0R22/(Lb[gr==gr_value]- Lw1R22)
            A33R22[:, 2]= m0R22/(Lg[gr==gr_value] - Lw2R22)
            A33R22[:, 3]= 1

            A44R22[:, 0] = ln( (Lb[gr==gr_value+1] - Lw1R22 )/ ((Lg[gr==gr_value+1]) - Lw2R22) )
            A44R22[:, 1]= -m0R22/(Lb[gr==gr_value+1]- Lw1R22)
            A44R22[:, 2]= m0R22/(Lg[gr==gr_value+1] - Lw2R22)
            A44R22[:, 3]= 1

            A55R22[:, 0] = ln( (Lb[gr==gr_value+2] - Lw1R22 )/ ((Lg[gr==gr_value+2]) - Lw2R22) )
            A55R22[:, 1]= -m0R22/(Lb[gr==gr_value+2]- Lw1R22)
            A55R22[:, 2]= m0R22/(Lg[gr==gr_value+2] - Lw2R22)
            A55R22[:, 3]= 1

            A66R22[:, 0] = ln( (Lb[gr==gr_value+3] - Lw1R22 )/ ((Lg[gr==gr_value+3]) - Lw2R22) )
            A66R22[:, 1]= -m0R22/(Lb[gr==gr_value+3]- Lw1R22)
            A66R22[:, 2]= m0R22/(Lg[gr==gr_value+3] - Lw2R22)
            A66R22[:, 3]= 1
```

```
             A2R22[:, :]= np.concatenate([A33R22[:,:],A44R22[:, :],
A55R22[:, :] , A66R22[:, :]])

             N2R22=A2R22.T@A2R22
             N2R22=A2R22.T@A2R22
             U2R22=A2R22.T@LR22
             inv_N2R22 = np.linalg.inv(N2R22)
             X2R22= -inv_N2R22@U2R22

             XAR22= X0R22 + X2R22
             V2R22= (A2R22@X2R22) + LR22
             RMSE20R2 = ((V2R22.T@V2R22)/(n20-u4))**(1/2)

             # Second iteration

             L01R22 =np.zeros((n33,1))
             L02R22 =np.zeros((n44,1))
             L03R22 =np.zeros((n55,1))
             L04R22 =np.zeros((n66,1))
             L00R22=np.zeros((n20,1))

             L01R22[:,0]= ((XAR22[0] * ( ln(( Lb[gr==gr_value] -
XAR22[1]) / (Lg[gr==gr_value] - XAR22[2]))) + XAR22[3] ))
             L02R22[:,0]= ((XAR22[0] * ( ln(( Lb[gr==gr_value+1] -
XAR22[1]) / (Lg[gr==gr_value+1] - XAR22[2]))) + XAR22[3] ))
             L03R22[:,0]= ((XAR22[0] * ( ln(( Lb[gr==gr_value+2] -
XAR22[1]) / (Lg[gr==gr_value+2] - XAR22[2]))) + XAR22[3] ))
             L04R22[:,0]= ((XAR22[0] * ( ln(( Lb[gr==gr_value+3] -
XAR22[1]) / (Lg[gr==gr_value+3] - XAR22[2]))) + XAR22[3] ))
             L00R22[:,0] = np.concatenate([L01R22[:, 0],L02R22[:,
0],L03R22[:, 0], L04R22[:, 0] ])
             L1R22 = L00R22 - z_grpR20

             m0 = XAR22[0]
             Lw1 = XAR22[1]
             Lw2 = XAR22[2]
             m1 = XAR22[3]

             A33R222 = np.zeros((n33,u4))
             A44R222 = np.zeros((n44,u4))
             A55R222 = np.zeros((n55,u4))
             A66R222 = np.zeros((n66,u4))
             A3R222 = np.zeros((n20,u4))

             A33R222[:, 0] = ln( (Lb[gr==gr_value] - Lw1 )/
((Lg[gr==gr_value]) - Lw2) )
             A33R222[:, 1]= -m0/(Lb[gr==gr_value]- Lw1)
             A33R222[:, 2]= m0/(Lg[gr==gr_value] - Lw2)
             A33R222[:, 3]= 1

             A44R222[:, 0] = ln( (Lb[gr==gr_value+1] - Lw1 )/
((Lg[gr==gr_value+1]) - Lw2) )
             A44R222[:, 1]= -m0/(Lb[gr==gr_value+1]- Lw1)
             A44R222[:, 2]= m0/(Lg[gr==gr_value+1] - Lw2)
             A44R222[:, 3]= 1
             A55R222[:,  0]  =  ln(  (Lb[gr==gr_value+2]  -  Lw1  )/
((Lg[gr==gr_value+2]) - Lw2) )
             A55R222[:, 1]= -m0/(Lb[gr==gr_value+2]- Lw1)
             A55R222[:, 2]= m0/(Lg[gr==gr_value+2] - Lw2)
             A55R222[:, 3]= 1

             A66R222[:,  0]  =  ln(  (Lb[gr==gr_value+3]  -  Lw1  )/
((Lg[gr==gr_value+3]) - Lw2) )
             A66R222[:, 1]= -m0/(Lb[gr==gr_value+3]- Lw1)
             A66R222[:, 2]= m0/(Lg[gr==gr_value+3] - Lw2)
             A66R222[:, 3]= 1

             A3R222[:,  :]=  np.concatenate([A33R222[:,:],A44R222[:,
:],A55R222[:, :],A66R222[:, :] ])

             N3R22=A3R222.T@A3R222
             U3R22=A3R222.T@L1R22
             inv_N3R22 = np.linalg.inv(N3R22)
             X3R22= -inv_N3R22@U3R22

             XA1R22= XAR22 + X3R22
             V3R22= (A3R222@X3R22) + L1R22
             RMSE30R2 = ((V3R22.T@V3R22)/(n20-u4))**(1/2)

             iteracao = iteracao +1
             X0R22=XAR22

             c1 = np.count_nonzero(gr5==gr_value)
             c2 = np.count_nonzero(gr5==gr_value +1 )
             c3 = np.count_nonzero(gr5==gr_value +2 )
             c4 = np.count_nonzero(gr5==gr_value +3 )
             count = c1 + c2 +c3 + c4

             Z_SDB_20_1 =np.zeros((c1,1))
             Z_SDB_20_2 =np.zeros((c2,1))
             Z_SDB_20_3 =np.zeros((c3,1))
             Z_SDB_20_4 =np.zeros((c4,1))
             Z_SDB_20=np.zeros((count,1))

             Z_SDB_20_1[:,0]  =  (XAR22[0]*ln  ((Lb5[gr5==gr_value]  -
XAR22[1])/(Lg5[gr5==gr_value] - XAR22[2])))+XAR22[3]
             Z_SDB_20_2[:,0] = (XAR22[0]*ln ((Lb5[gr5==gr_value+1] -
XAR22[1])/(Lg5[gr5==gr_value+1] - XAR22[2])))+XAR22[3]
             Z_SDB_20_3[:,0] = (XAR22[0]*ln ((Lb5[gr5==gr_value+2] -
XAR22[1])/(Lg5[gr5==gr_value+2] - XAR22[2])))+XAR22[3]
             Z_SDB_20_4[:,0] = (XAR22[0]*ln ((Lb5[gr5==gr_value+3] -
XAR22[1])/(Lg5[gr5==gr_value+3] - XAR22[2])))+XAR22[3]
             Z_SDB_20[:,0]      =      np.concatenate([Z_SDB_20_1[:,
0],Z_SDB_20_2[:, 0],Z_SDB_20_3[:, 0],Z_SDB_20_4[:, 0]])

             z_20_1 =np.zeros((c1,1))
             z_20_2 =np.zeros((c2,1))
             z_20_3 =np.zeros((c3,1))
             z_20_4 =np.zeros((c4,1))
             z_20 =np.zeros((count,1))
```

```
            z_20_1[:,0] = z5[gr5==gr_value]
            z_20_2[:,0] = z5[gr5==gr_value+1]
            z_20_3[:,0] = z5[gr5==gr_value+2]
            z_20_4[:,0] = z5[gr5==gr_value+3]
            z_20[:,0]   =   np.concatenate([z_20_1[:,   0],z_20_2[:,
0],z_20_3[:, 0],z_20_4[:, 0 ]])

            Diff_20 = Z_SDB_20 - z_20
            quadrado = Diff_20*Diff_20
            soma = sum(quadrado)

            if count >0:
                RMSE_20 = (soma/count)**0.5

            Diff_list.append(Diff)

        ERROR3.append(RMSE_20)

    #    print(   "\033[0;35m" +   "\033[1m"   + "Solution of Group:
" + str((gr_value * 5))  +   " A "   + str(((gr_value *5) +20)))
    #    print ("\033[0m" + "RMSE: " + str(RMSE30R2))
    #    print ("\033[0m" + "RMSE_VAL: " + str(RMSE_20))
    #    print ("X: " + str(XAR22) + "\n ")

    range0_20_i20 = ERROR3[0]
```

```
#Application of geographic model - VERTICAL AND DIERSSEN

# RANGE OF 0 - 5

# Solution Vector (X)

for gr_value in range(0, gr_nr+1):
    nL= np.count_nonzero(gr==gr_value) # n. of control points in group
    AL = np.zeros((nL,u))

    z_grpL = z[gr==gr_value]

    AL[:, 0] = ln(Lb[gr==gr_value] / Lg[gr==gr_value])
    AL[:, 1] = 1
    NL=AL.T@AL
    UL=AL.T@-z_grpL
    inv_NL = np.linalg.inv(NL)
    XL= -inv_NL@UL
    VL= (AL@XL) - z_grpL
    RMSEL = ((VL.T@VL)/(nL-u))**(1/2)

    X0L = np.array([XL[0], XL[1]])   #[m0 , m1]

    count = np.count_nonzero(gr5==gr_value)
    Z_SDB_5L = (X0L[0]*ln ((Lb5[gr5==gr_value])/(Lg5[gr5==gr_value] )))+X0L[1]
    Diff = Z_SDB_5L - z5[gr5==gr_value]
    quadrado = Diff*Diff
    soma = sum(quadrado)

    if count >0:
        RMSE_5L = (soma/count)**0.5

    Diff_list.append(Diff)

#    print( "\033[0;35m" +  "\033[1m"  + "Solution of Group: " + str((gr_value * 5))  +  " A " + str(((gr_value *5) +5)))
#    print ("\033[0m" + "RMSE: " + str(RMSEL))
#    print ("\033[0m" + "RMSE: " + str(RMSE_5L))
#    print ("X: " + str(X0L) + "\n ")

    ERROR4.append(RMSE_5L)

range0_5_i5L = ERROR4[0]
range5_10_i5L = ERROR4[1]
range10_15_i5L = ERROR4[2]
range15_20_i5L = ERROR4[3]

# RANGE OF 0 - 10

# Solution Vector (X)

for gr_value in range(0, gr_nr+1):
    n11L=np.count_nonzero(gr==gr_value) #n. of control points in group
```

```
          n22L = np.count_nonzero(gr==gr_value+1)
          nL = n11L+n22L

          z_grp1L =np.zeros((n11L,1))
          z_grp2L =np.zeros((n22L,1))
          z_grpR1L =np.zeros((nL,1))

          z_grp1L[:,0] = z[gr==gr_value]
          z_grp2L[:,0] = z[gr==gr_value+1]
          z_grpR1L[:,0]  =  np.concatenate([z_grp1L[:,  0],z_grp2L[:,
0]])

          A11L = np.zeros((n11L,u))
          A22L = np.zeros((n22L,u))
          AR1L = np.zeros((nL,u))

          A11L[:, 0] = ln(Lb[gr==gr_value] / Lg[gr==gr_value])
          A22L[:, 0] = ln(Lb[gr==gr_value+1] / Lg[gr==gr_value+1])

          AR1L[:, 0]= np.concatenate([A11L[:, 0],A22L[:, 0]])
          AR1L[:, 1] = 1

          NR1L=AR1L.T@AR1L
          UR1L=AR1L.T@-z_grpR1L
          inv_NR1L = np.linalg.inv(NR1L)
          XR1L= -inv_NR1L@UR1L
          VR1L= (AR1L@XR1L) - z_grpR1L
          RMSER1L = ((VR1L.T@VR1L)/(nL-u))**(1/2)

          X0LL = np.array([XR1L[0], XR1L[1]])   #[m0 , m1]

          c1 = np.count_nonzero(gr5==gr_value)
          c2 = np.count_nonzero(gr5==gr_value +1 )
          count = c1 + c2

          Z_SDB_10_1L =np.zeros((c1,1))
          Z_SDB_10_2L =np.zeros((c2,1))
          Z_SDB_10L=np.zeros((count,1))

          Z_SDB_10_1L[:,0]    =    (X0LL[0]*ln    ((Lb5[gr5==gr_value]
)/(Lg5[gr5==gr_value] )))+X0LL[1]
          Z_SDB_10_2L[:,0]    =    (X0LL[0]*ln    ((Lb5[gr5==gr_value+1]
)/(Lg5[gr5==gr_value+1] )))+X0LL[1]
          Z_SDB_10L[:,0]    =    np.concatenate([Z_SDB_10_1L[:,    0],
Z_SDB_10_2L[:, 0]])

          z_10_1L =np.zeros((c1,1))
          z_10_2L =np.zeros((c2,1))
          z_10L =np.zeros((count,1))

          z_10_1L[:,0] = z5[gr5==gr_value]
          z_10_2L[:,0] = z5[gr5==gr_value+1]
          z_10L[:,0] = np.concatenate([z_10_1L[:, 0],z_10_2L[:, 0]])

          Diff_10L = Z_SDB_10L - z_10L
          quadrado = Diff_10L*Diff_10L
```

```
        soma = sum(quadrado)

        if count >0:
            RMSE_10L = (soma/count)**0.5

        Diff_list.append(Diff)

    #    print( "\033[0;35m" +  "\033[1m"  + "Solution of Group: "
+ str((gr_value * 5))  +  " A " + str(((gr_value *5) +10)))
    #    print ("\033[0m" + "RMSE: " + str(RMSER1L))
    #    print ("\033[0m" + "RMSE: " + str(RMSE_10L))
    #    print ("X: " + str(X0LL) + "\n ")

        ERROR5.append(RMSE_10L)

    range0_10_i10L = ERROR5[0]
    range5_15_i10L = ERROR5[1]
    range10_20_i10L = ERROR5[2]

    # RANGE OF 0 - 15

    # Solution Vector (X)

    for gr_value in range(0, gr_nr-1):
        n33L=np.count_nonzero(gr==gr_value) #n. of control points in
group
        n44L = np.count_nonzero(gr==gr_value+1)
        n55L = np.count_nonzero(gr==gr_value+2)
        n2L = n33L+n44L+n55L

        z_grp12L =np.zeros((n33L,1))
        z_grp22L =np.zeros((n44L,1))
        z_grp32L =np.zeros((n55L,1))
        z_grpR22L =np.zeros((n2L,1))

        z_grp12L[:,0] = z[gr==gr_value]
        z_grp22L[:,0] = z[gr==gr_value+1]
        z_grp32L[:,0] = z[gr==gr_value+2]
        z_grpR22L[:,0] = np.concatenate([z_grp12L[:, 0],z_grp22L[:,
0],z_grp32L[:, 0] ])

        A112L = np.zeros((n33L,u))
        A222L = np.zeros((n44L,u))
        A332L = np.zeros((n55L,u))
        AR22L = np.zeros((n2L,u))

        A112L[:, 0] = ln(Lb[gr==gr_value] / Lg[gr==gr_value])
        A222L[:, 0] = ln(Lb[gr==gr_value+1] / Lg[gr==gr_value+1])
        A332L[:, 0] = ln(Lb[gr==gr_value+2] / Lg[gr==gr_value+2])

        AR22L[:,  0]=  np.concatenate([A112L[:,  0],A222L[:,  0],
A332L[:, 0]])
        AR22L[:, 1] = 1

        NR22L=AR22L.T@AR22L
        UR22L=AR22L.T@-z_grpR22L
        inv_NR22L= np.linalg.inv(NR22L)
        XR22L= -inv_NR22L@UR22L
        VR22L= (AR22L@XR22L) - z_grpR22L
        RMSER22L = ((VR22L.T@VR22L)/(n2L-u))**(1/2)

        X0LL2 = np.array([XR22L[0], XR22L[1]])   #[m0 , m1]

        c1 = np.count_nonzero(gr5==gr_value)
        c2 = np.count_nonzero(gr5==gr_value +1 )
        c3 = np.count_nonzero(gr5==gr_value +2 )
        count = c1 + c2 +c3

        Z_SDB_15_1L =np.zeros((c1,1))
        Z_SDB_15_2L =np.zeros((c2,1))
        Z_SDB_15_3L =np.zeros((c3,1))
        Z_SDB_15L=np.zeros((count,1))

        Z_SDB_15_1L[:,0] = (X0LL2[0]*ln ((Lb5[gr5==gr_value]
)/(Lg5[gr5==gr_value] )))+X0LL2[1]
        Z_SDB_15_2L[:,0] = (X0LL2[0]*ln ((Lb5[gr5==gr_value+1]
)/(Lg5[gr5==gr_value+1] )))+X0LL2[1]
        Z_SDB_15_3L[:,0] = (X0LL2[0]*ln ((Lb5[gr5==gr_value+2]
)/(Lg5[gr5==gr_value+2] )))+X0LL2[1]
        Z_SDB_15L[:,0] = np.concatenate([Z_SDB_15_1L[:, 0],
Z_SDB_15_2L[:, 0], Z_SDB_15_3L[:, 0]])

        z_15_1L =np.zeros((c1,1))
        z_15_2L =np.zeros((c2,1))
        z_15_3L =np.zeros((c3,1))
        z_15L =np.zeros((count,1))

        z_15_1L[:,0] = z5[gr5==gr_value]
        z_15_2L[:,0] = z5[gr5==gr_value+1]
        z_15_3L[:,0] = z5[gr5==gr_value+2]
        z_15L[:,0] = np.concatenate([z_15_1L[:, 0],z_15_2L[:,
0],z_15_3L[:, 0]])

        Diff_15L = Z_SDB_15L - z_15L
        quadrado = Diff_15L*Diff_15L
        soma = sum(quadrado)

        if count >0:
            RMSE_15L = (soma/count)**0.5

        Diff_list.append(Diff)

    #    print( "\033[0;35m" +  "\033[1m"  + "Solution of Group: "
+ str((gr_value * 5))  +  " A " + str(((gr_value *5) +15)))
    #    print ("\033[0m" + "RMSE: " + str(RMSER22L))
    #    print ("\033[0m" + "RMSE: " + str(RMSE_15L))
    #    print ("X: " + str(X0LL2) + "\n ")

        ERROR6.append(RMSE_15L)

    range0_15_i15L = ERROR6[0]
    range5_20_i15L = ERROR6[1]
    # RANGE OF 0 - 20

    # Solution Vector (X)

    for gr_value in range(0, gr_nr-2):
        n33L=np.count_nonzero(gr==gr_value) #n. of control points in
group
        n44L = np.count_nonzero(gr==gr_value+1)
        n55L = np.count_nonzero(gr==gr_value+2)
        n66L = np.count_nonzero(gr==gr_value+3)
        n3L = n33L+n44L+n55L+n66L

        z_grp13L =np.zeros((n33L,1))
        z_grp23L =np.zeros((n44L,1))
        z_grp33L =np.zeros((n55L,1))
        z_grp43L =np.zeros((n66L,1))
        z_grpR23L =np.zeros((n3L,1))

        z_grp13L[:,0] = z[gr==gr_value]
        z_grp23L[:,0] = z[gr==gr_value+1]
        z_grp33L[:,0] = z[gr==gr_value+2]
        z_grp43L[:,0] = z[gr==gr_value+3]
        z_grpR23L[:,0] = np.concatenate([z_grp13L[:, 0],z_grp23L[:,
0],z_grp33L[:, 0],z_grp43L[:, 0] ])

        A113L = np.zeros((n33L,u))
        A223L = np.zeros((n44L,u))
        A333L = np.zeros((n55L,u))
        A433L = np.zeros((n66L,u))
        AR23L = np.zeros((n3L,u))

        A113L[:, 0] = ln(Lb[gr==gr_value] / Lg[gr==gr_value])
        A223L[:, 0] = ln(Lb[gr==gr_value+1] / Lg[gr==gr_value+1])
        A333L[:, 0] = ln(Lb[gr==gr_value+2] / Lg[gr==gr_value+2])
        A433L[:, 0] = ln(Lb[gr==gr_value+3] / Lg[gr==gr_value+3])

        AR23L[:,  0]=  np.concatenate([A113L[:,  0],A223L[:,  0],
A333L[:, 0], A433L[:, 0]])
        AR23L[:, 1] = 1

        NR23L=AR23L.T@AR23L
        UR23L=AR23L.T@-z_grpR23L
        inv_NR23L= np.linalg.inv(NR23L)
        XR23L= -inv_NR23L@UR23L
        VR23L= (AR23L@XR23L) - z_grpR23L
        RMSER23L = ((VR23L.T@VR23L)/(n3L-u))**(1/2)

        X0LL3 = np.array([XR23L[0], XR23L[1]])   #[m0 , m1]

        c1 = np.count_nonzero(gr5==gr_value)
        c2 = np.count_nonzero(gr5==gr_value +1 )
        c3 = np.count_nonzero(gr5==gr_value +2 )
        c4 = np.count_nonzero(gr5==gr_value +3 )
        count = c1 + c2 +c3 +c4
```

```
         Z_SDB_20_1L =np.zeros((c1,1))
         Z_SDB_20_2L =np.zeros((c2,1))
         Z_SDB_20_3L =np.zeros((c3,1))
         Z_SDB_20_4L =np.zeros((c4,1))
         Z_SDB_20L=np.zeros((count,1))

         Z_SDB_20_1L[:,0]   =   (X0LL3[0]*ln   ((Lb5[gr5==gr_value]
)/(Lg5[gr5==gr_value] )))+X0LL3[1]
         Z_SDB_20_2L[:,0]   =   (X0LL3[0]*ln   ((Lb5[gr5==gr_value+1]
)/(Lg5[gr5==gr_value+1] )))+X0LL3[1]
         Z_SDB_20_3L[:,0]   =   (X0LL3[0]*ln   ((Lb5[gr5==gr_value+2]
)/(Lg5[gr5==gr_value+2] )))+X0LL3[1]
         Z_SDB_20_4L[:,0]   =   (X0LL3[0]*ln   ((Lb5[gr5==gr_value+3]
)/(Lg5[gr5==gr_value+3] )))+X0LL3[1]
         Z_SDB_20L[:,0]    =    np.concatenate([Z_SDB_20_1L[:,    0],
Z_SDB_20_2L[:, 0], Z_SDB_20_3L[:, 0], Z_SDB_20_4L[:, 0]])

         z_20_1L =np.zeros((c1,1))
         z_20_2L =np.zeros((c2,1))
         z_20_3L =np.zeros((c3,1))
         z_20_4L =np.zeros((c4,1))
         z_20L =np.zeros((count,1))

         z_20_1L[:,0] = z5[gr5==gr_value]
         z_20_2L[:,0] = z5[gr5==gr_value+1]
         z_20_3L[:,0] = z5[gr5==gr_value+2]
         z_20_4L[:,0] = z5[gr5==gr_value+3]
         z_20L[:,0]   =   np.concatenate([z_20_1L[:,   0],z_20_2L[:,
0],z_20_3L[:, 0],z_20_4L[:, 0]])

         Diff_20L = Z_SDB_20L - z_20L
         quadrado = Diff_20L*Diff_20L
         soma = sum(quadrado)

         if count >0:
             RMSE_20L = (soma/count)**0.5

         Diff_list.append(Diff)

     #    print( "\033[0;35m" +  "\033[1m"  + "Solution of Group: "
+ str((gr_value * 5))  +  " A " + str(((gr_value *5) +20)))
     #    print ("\033[0m" + "RMSE: " + str(RMSER23L))
     #    print ("\033[0m" + "RMSE VAL: " + str(RMSE_20L))
     #    print ("X: " + str(X0LL3) + "\n ")

         ERROR7.append(RMSE_20L)

     range0_20_i20L = ERROR7[0]
```