ESCOLA NAVAL

talant de bien faire

João Pedro Duarte Gonçalves

Desenvolvimento de algoritmo para previsão 2D e 3D de propagação submarina de raios acústicos com reflexões

Dissertação para obtenção do Grau de Mestre em Ciências Militares Navais, na especialidade de Marinha



Alfeite
2022



João Pedro Duarte Gonçalves

Desenvolvimento de algoritmo para previsão 2D e 3D de propagação submarina de raios acústicos com reflexões

Dissertação para obtenção do Grau de Mestre em Ciências Militares Navais, na especialidade de Marinha

Orientação de: Professor Doutor Miguel Bento Moreira
Coorientação de: Professor Doutor Mário Costa Gatta
Coorientação de: CMG Carlos Nélson Lopes da Costa

O Aluno Mestrando,

Assinado por: **João Pedro Duarte Gonçalves**
Num. de Identificação: 15291866
Data: 2022.09.19 00:06:43+01'00'

João Gonçalves

O Orientador,

Miguel Moreira

O Coorientador,

Mário Gatta

O Coorientador,

Carlos Costa

Alfeite
2022

"There are no secrets to success. It is the result of preparation, hard work, and learning from failure."
– Colin Powell

Dedico ao meu pai. Obrigado por tudo.

Agradecimentos

Começo por agradecer ao meu Orientador, Professor Miguel Bento Moreira, não só pelo apoio, disponibilidade, confiança e conhecimentos passados, mas também por todos os desafios que me colocou ao longo deste ano, que contribuiram para o meu desenvolvimento enquanto estudante.

Quero agradecer igualmente aos meus coorientadores, Capitão-de-mar-e-guerra Carlos Nélson Lopes da Costa e Professor Mário Costa Gatta, pelos conhecimentos transmitidos nas áreas da oceanografia e da matemática, e pelo esclarecimento de todas as dúvidas que foram surgindo ao longo do desenvolvimento da dissertação de mestrado.

Agradeço à minha família e amigos, embora por vezes fisicamente distantes, por serem sempre a minha base e motivo de procurar fazer melhor.

Ao meu curso, Capitão-de-mar-e-guerra Fernando Amor Monteiro de Barros, por estes incríveis cinco anos de Escola Naval repleto de bons e maus momentos.

À minha namorada, Ana Queirós, por presenciar e apoiar todos os momentos de desespero e de conquista ao longo destes últimos anos. Por acreditar incondicionalmente nas minhas capacidades, motivando-me para chegar mais longe.

Resumo

O objetivo deste trabalho é construir dois programas de acústica submarina em 2D e 3D, capazes de prever a propagação do som num ambiente oceânico real.

Os modelos de propagação utilizados são modelos acústicos paramétricos de ray tracing baseados no princípio do tempo mínimo de Fermat e são compostos por sistemas de equações algébrico-diferenciais. Estes sistemas de equações que modelam as trajetórias acústicas são estabelecidos através das equações de Euler-Lagrange.

Para estudar, prever e simular a propagação de raios acústicos num ambiente oceânico real, é importante ter dados reais sobre a batimetria e a climatologia. Para isso, utilizou-se três bases de dados diferentes que nos fornecem esse tipo de informação. As bases de dados são a GEBCO, a WOD18 e a TEOS-10.

A base de dados GEBCO permite caracterizar a batimetria oceânica em quase todo o mundo e neste trabalho utilizou-se a versão mais recente desta base de dados relativamente à área envolvente a Portugal Continental.

A WOD18 é uma base de dados que contém vários tipos de dados com interesse oceanográfico. Para calcular a velocidade de propagação do som, usou-se os valores de salinidade e temperatura desta base de dados.

A TEOS-10 é uma biblioteca de software que contém diversas funções que permitem modelar várias variáveis oceanográficas, e, neste caso, utilizou-se para calcular a velocidade de som à medida que o raio acústico se propagava no meio.

Posto isto, foram desenvolvidos dois programas com a capacidade de estudar a propagação de raios acústicos em um ambiente oceânico real. Os nomes dos programas são `REOART 2D` e `REOART 3D` (**R**eal **E**nvironment **O**cean-**A**coustic **R**ay **T**racing). Estes programas usam as bases de dados mencionadas para retirar informação sobre a batimetria e climatologia dos oceanos para calcular as trajetórias dos raios acústicos em um ambiente oceânico real.

Ambos os programas foram submetidos a vários testes e os resultados foram credíveis e fiáveis. Não só as trajetórias são credíveis, mas também as reflexões na superfície e no fundo do oceano satisfazem a lei de Snell. Dada a singularidade de calcular a velocidade do som ao longo da trajetória do raio acústico, estes programas não podiam ser comparados com outros modelos de propagação devido à diferença entre os valores para velocidade do som.

Palavras-chave: Acústica Submarina, Ambiente Oceânico Real, Modelação Acústica, Reflexão

Abstract

The purpose of this work is to build two ocean-acoustic programs in 2D and 3D able to predict the propagation of sound in a real ocean environment.

The propagation models used are ocean-acoustic ray tracing parametric models based on Fermat's least time principle and they are composed by systems of algebraic-differential equations. These systems of equations that model the acoustic trajectories are established through the Euler-Lagrange's equations.

To study, predict and simulate the propagation of acoustic rays in a real ocean environment is important to have real data about the bathymetry and climatology. In order to do that we used three different databases that give us that kind of information. The databases are GEBCO, WOD18 and TEOS-10.

The GEBCO database enables characterize the ocean bathymetry almost worldwide. In this work it as used the latest version of this database relative to the Portugal oceanic coast.

The WOD18 is a database that contains a variety of data types relevant to oceanography. To calculate the speed of sound in underwater we used the salinity and temperature values from this database.

The TEOS-10 is a software library that contains numerous oceanographic functions for computing relevant ocean variables, and we used it to calculate the speed of sound along the ray trajectory.

Post this, two programs were developed with the ability to study sound propagation in a real ocean environment. The names of the programs are `REOART 2D` and `REOART 3D` (**R**eal **E**nvironment **O**cean-**A**coustic **R**ay **T**racing). These programs use the databases mentioned before to withdraw information regarding the ocean topography and climatology to compute the acoustic ray trajectories in a real ocean environment.

Both programs were submitted to several tests and the results were credible and reliable. Not only do the trajectories seem credible, but also the reflections on the surface that at the bottom of the ocean seem to satisfy Snell's law. Given the uniqueness of calculating the speed of sound along the path of the acoustic ray, these programs could not be compared with other propagation models because of the difference between the speed of sound.

Keywords: Underwater Acoustics, Real Ocean Environment, Acoustic Modeling, Reflection

Índice Geral

Epígrafe III

Dedicatória V

Agradecimentos VII

Resumo IX

Abstract XI

Índice Geral XIII

Lista de Figuras XVI

Lista de Símbolos XVIII

1 Introdução 1

2 Estado da Arte 3

3 Fundamentos Teóricos 5
3.1 Traçado de raios acústicos . . . 5
3.2 Princípio do tempo mínimo de Fermat . . . 6
3.3 Cálculo variacional . . . 7
3.4 Resolução numérica de equações diferenciais . . . 8
3.5 Propagação acústica 2D no oceano . . . 11
3.6 Interpolação . . . 13
3.6.1 Interpolação multilinear . . . 13
3.6.2 Interpolação pelo vizinho mais próximo . . . 14

4 Formulação da propagação de raios acústicos com reflexões em ambiente real 17
4.1 Formulação do problema . . . 17
4.2 Fecho do problema . . . 20
4.2.1 Cálculo da propagação do raio acústico . . . 20
4.2.2 Modelação das reflexões . . . 20
4.2.2.1 A reflexão em 3D . . . 20
4.2.2.2 A reflexão em 2D . . . 21
4.2.3 Modelação em ambiente real . . . 22

5 Desenvolvimento dos algoritmos para modelação acústica **23**
5.1 Fluxograma . . . 24
5.2 Base de dados . . . 24
5.2.1 Dados batimétricos GEBCO . . . 24
5.2.1.1 Batimetria em 2D . . . 24
5.2.1.2 Batimetria em 3D . . . 26
5.2.2 Dados climatológicos WOD18 . . . 29
5.2.2.1 Climatologia em 2D . . . 29
5.2.2.2 Climatologia em 3D . . . 30
5.2.3 Funções TEOS-10 . . . 31
5.3 Velocidade do som . . . 32
5.4 Cálculo dos gradientes espaciais . . . 32
5.5 Reflexão do raio acústico nas fronteiras . . . 33

6 Simulações numéricas e discussão de resultados **35**
6.1 `REOART 2D` — simulação numérica com um raio acústico . . . 35
6.2 `REOART 2D` — simulação numérica com múltiplos raios acústicos . . . 37
6.3 `REOART 3D` — simulação numérica com um raio acústico . . . 39

7 Conclusões e trabalho futuro **43**
7.1 Conclusões . . . 43
7.2 Trabalho futuro . . . 44

Bibliografia **45**

Apêndices **49**

A Dedução da matriz R(n) **49**

B Tabela das técnicas de derivação numérica **53**

C Fórmulas de transporte de coordenadas geodésicas **55**

D Trabalho apresentado na conferência Internoise 2022 **57**

E Trabalho apresentado no colóquio 7^{as} Jornadas de Engenharia Hidrográfica/2^{as} Jornadas Luso-Espanholas de Hidrografia **71**

F Trabalho apresentado no colóquio Jornadas do Mar 2022 **77**

G Código do programa `REOART 2D` **89**

H Código do programa `REOART 3D` **109**

Lista de Figuras

3.1 Representação do trajeto γ, entre $[\gamma_a, \gamma_b]$, de um raio acústico e o esboço da respetiva parametrização vetorial $\mathbf{r}(\tau)$ e de um comprimento elementar de arco Δs. 8
3.2 Representação da curvatura de curvas planas com a descrição do plano da curvatura em termos das coordenadas espacias (x, z). 11
3.3 Representação da secção normal (geodésica) numa esfera. 12
3.4 14

4.1 Ilustração de uma reflexão especular em 3D. 20

5.1 Fluxograma dos programas `REOART 2D` e `REOART 3D`. 25
5.2 Mapa batimétrico da área envolvente a Portugal Continental. 26
5.3 Mapa batimétrico da área envolvente a Portugal Continental com a representação da trajetória (geodésica). 27
5.4 Perfil batimétrico da posição geográfica de latitude $36,31°$N e longitude 11° O, ao longo do azimute 300°, até à distância de 80 quilómetros. 27
5.5 Mapa batimétrico da área envolvente a Portugal Continental com a área de simulação 3D (assinalada a cor vermelha). 28
5.6 Perfil batimétrico 3D da área definida na figura (5.5). 28
5.7 Mapa com a distribuição dos pontos com dados climatológicos da área envolvente a Portugal Continental. 29
5.8 Mapa com a distribuição dos pontos com dados climatológicos da área envolvente a Portugal Continental com detalhe da representação da trajetória assumida. 30
5.9 Mapa com a distribuição dos pontos com dados climatológicos da área envolvente a Portugal Continental com a representação da área de simulação 3D. 31
5.10 Cálculo da normal no plano α no ponto de incidência através de três pontos não colineares. 34

6.1 Simulação numérica na posição geográfica de latitude $36,31°$N e longitude 11° O, no azimute 300°, com um raio acústico projetado na imersão de 2000 m, com um ângulo de 0°. 36
6.2 Simulação numérica na posição geográfica de latitude $36,31°$N e longitude 11° O, no azimute 300°, com um raio acústico projetado na imersão de 2000 m, com um ângulo de 0°, com a representação da velocidade de som por esquema de cores. 37

6.3 Mapa batimétrico da área envolvente a Portugal Continental com a representação da trajetória com início na posição geográfica de latitude 39°N e longitude 10, 1° O, ao longo do azimute 270°. 38
6.4 Perfil batimétrico da posição geográfica de latitude 39°N e longitude 10, 1° O, ao longo do azimute 270°, até à distância de 80 quilómetros. 38
6.5 Mapa com a distribuição dos pontos com dados climatológicos da área envolvente a Portugal Continental com detalhe da representação da trajetória com início na posição geográfica de latitude 39°N e longitude 10, 1° O, ao longo do azimute 270°. 39
6.6 Simulação numérica na posição geográfica de latitude 39°N e longitude 10, 1° O, ao longo do azimute 270°, com múltiplos raios acústicos projetados na imersão de 1 m. 40
6.7 Mapa batimétrico da área envolvente a Portugal Continental com a área de simulação 3D (assinalada a cor vermelha). 41
6.8 Mapa com a distribuição dos pontos com dados climatológicos da área envolvente a Portugal Continental com a representação da área de simulação 3D. 41
6.9 Representação de uma simulação com um raio acústico, utilizando o programa `REOART 3D`, projetado na imersão de 70 m. 42
6.10 Representação de uma simulação com um raio acústico, utilizando o programa `REOART 3D`, projetado na imersão de 70 m. 42

Lista de Símbolos

Latim

x, y, z	Coordenadas cartesianas espaciais a três dimensões
a, b, c	Primeiras derivadas de $x(\tau)$, $y(\tau)$ e $z(\tau)$ a três dimensões
x, y	Coordenadas espaciais a duas dimensões
z, w	Primeiras derivadas de $x(\tau)$ e $y(\tau)$ a duas dimensões
y ou $y'(x)$	Primeira derivada de y relativamente a x
y'' ou $y''(x)$	Segunda derivada de y relativamente a x
t	Variável de tempo
p	Função que define a pressão num ponto em função do tempo e do espaço
c	Função que define a velocidade do som no espaço
A	Amplitude de p num determinado ponto
j	Unidade imaginária
e	Número de Neper
L	Longitude
$\hat{s}$	Vetor unitário com a direção de propagação do raio acústico
n	Índice de refração
$\mathbf{i}$	Versor unitário do raio acústico incidente
$\mathbf{r}$	Versor unitário do raio acústico refletido
$\mathbf{n}$	Versor unitário da normal ao plano no ponto de incidência
$\mathbb{R}$	Conjuntos dos números reais

Grego

α	Azimute
ω	Frequência da variação de p
Γ	Função de fase da variação de p no espaço
γ	Trajetória de um raio acústico
∇^2	Operador laplaciano
∇	Operador gradiente
τ	Parâmetro de comprimento ao longo da trajetória
δx	Variação associada à função x
$\epsilon\zeta$	Variação entre a curva C e C'
φ	Latitude
θ_i	Ãngulo de incidência
θ_r	Ângulo de reflexão

Capítulo 1

Introdução

Nas últimas quatro décadas, a acústica submarina, ciência que estuda a propagação do som na água, tem vindo a ser fortemente desenvolvida e aplicada na resolução de importantes problemas como a previsão sonar (Keller & Papadakis, 1977). Uma coluna de água pode ser descrita a partir da salinidade, pressão e temperatura, as quais variam no tempo e no espaço, o que torna o estudo, previsão e modelação da propagação do som num meio não homogéneo, como a água, numa tarefa difícil.

O avanço tecnológico dos computadores permitiu prever com maior exatidão o comportamento do som na água, possibilitando a aplicação desta ciência em vários cenários. Atualmente, a acústica submarina é utilizada não só nos sistemas acústicos de deteção e localização submarina, numa vertente mais operacional, mas também na estimativa de propriedades ambientais, no âmbito civil (Tolstoy, 1996).

Diferentes tipos de modelos de propagação têm sido aplicados no desenvolvimento de aplicações computacionais dedicadas ao estudo deste fenómeno. De referir, por exemplo, a utilização de modelos de traçado de raios sonoros baseados na aproximação eikonal (*ray tracing*), modelos baseados na resolução da equação de onda recorrendo a modos ortogonais, modelos baseados em técnicas de geometria diferencial e modelos baseados no princípio do tempo mínimo de Fermat (Rodriguez et al., 2021).

A adição de modelos alternativos de propagação e inovação na modelação acústica está na base do presente trabalho. Neste intuito, foram desenvolvidos duas aplicações computacionais de propagação acústica submarina 2D e 3D denominadas — `REOART 2D` e `REOART 3D` (***R**eal **E**nvironment **O**cean-**A**coustic **R**ay **T**racing*). Estes dois programas recorrem a dois modelos paramétricos de *ray tracing* 2D e 3D de propagação de raios acústicos submarinos, que têm por base o princípio do tempo mínimo de Fermat, materializados por sistemas de equações algébrico-diferenciais (Raposo, 2019). O `REOART 2D` e `REOART 3D`, para além dos modelos paramétricos referidos, utilizam as bases de dados da GEBCO (*General Bathymetric Charts of the Ocean*), da WOD18 (*World Ocean Database 2018*) e a biblioteca de software TEOS-10 (*Thermodynamic Equation of State 2010*), e têm a capacidade de modelar trajetórias de raios acústicos e respetivas reflexões, na superfície e no fundo, em

ambiente oceânico real, a duas e três dimensões.

Capítulo 2

Estado da Arte

Primeiramente, para se desenvolver um trabalho sobre acústica submarina é necessário compreendê-la, e, neste caso em concreto, como modelá-la, tornando-se importante conhecer trabalhos já realizados nesta área. Este trabalho insere-se em três áreas de conhecimento, sendo estas a acústica submarina, a matemática e a climatologia, originando um vasto leque de bibliografia.

As obras de Brekhovskikh (Brekhovskikh, 1983), Kinsler (Kinsler, 2000) e Urick (Urick, 1967), são o princípio para o estudo da acústica submarina. A partir destas obras compreende-se a parte teórica e matemática da acústica submarina, assim como os possíveis campos de estudo nesta área.

Com a base teórica consolidada, vários modelos de propagação foram criados. Nas obras de Etter (Etter, 1995), Keller e Papadakis (Keller & Papadakis, 1977), e Tolstoy (Tolstoy, 1996), além de serem expostos os modelos de propagação acústica submarina desenvolvidos, como por exemplo, os modelos de *ray tracing*, os modelos normais, os modelos das equações parabólicas, entre outros, são apresentados os desafios na modelação acústica e as vantagens de ferramentas de previsão acústica 3D.

Os modelos utilizados nas ferramentas desenvolvidas são modelos de *ray tracing* desenvolvidos por Pedro Raposo na sua dissertação de mestrado (Raposo, 2019). Para a compreensão mais detalhada destes modelos é essencial perceber a base matemática utilizada, e, para isso, salientam-se alguns autores como Cornelius Lanczos, George Simmons e Steven Krantz, Max Born e Emil Wolf, Linda Petzold e Kent Nagle, Edward B. Saff e Arthur David Snider, sendo as respetivas obras (Lanczos, 1970), (Simmons, 2016), (Born & Wolf, 1964), (Ascher & Petzold, 1998) e (Nagle et al., 2014).

Internamente, na Escola Naval, nos últimos anos tem-se vindo a desenvolver modelos e programas ligados à acústica submarina e previsão sonar, realçando os trabalhos da Ana Torres (Torres, 2019), Pedro Marques (Marques, 2018) e Beatriz Paiva (Paiva, 2018), permitindo também um bom ponto de partida para a introdução de alguns conceitos ligados a esta temática.

Atualmente existem várias ferramentas e modelos para a propagação do som e previsão acústica, no entanto, destacam-se os programas seguidamente apresentados.

Começando pelo TRACEO, este é um programa que utiliza o modelo de *ray tracing* baseado na resolução da equação eikonal. Para mais detalhes sobre este programa, consultar o manual de utilizador do mesmo (Rodrıguez, 2010).

O BELLHOP também é um programa de *ray tracing* baseado na resolução da equação eikonal, contudo, tem algumas diferenças do TRACEO, como se pode verificar no manual de utilizador do BELLHOP (Porter, 2011).

O WaveQ3D, igualmente um modelo de *ray tracing* baseado na resolução da equação eikonal, foi desenvolvido na Universidade de Rhode Island. No manual deste programa (Reilly et al., 2014), é retratado o problema da reflexão dos raios e como foi modelado.

Outra ferramenta é o programa HARPO (Jones et al., 1986). É um modelo de *ray tracing* baseado nas equações de Hamilton elaborado pela NOAA (*National Oceanic and Atmospheric Administration*).

Capítulo 3

Fundamentos Teóricos

3.1 Traçado de raios acústicos

O fenómeno do som é a manifestação da oscilação de um meio elástico onde ocorre e a correspondente propagação dessa perturbação. Este fenómeno é modelado pela equação de onda (linear). Admitindo que o fenómeno oscilatório ocorre a uma frequência suficientemente elevada é possível deduzir a partir da equação de onda a chamada teoria de propagação de raios acústicos (*ray tracing*) (Hovem, 2013). Assume-se então que a propagação do som se dá ao longo de raios que são perpendiculares às frentes de onda, e que se definem como superfícies de igual fase das ondas acústicas e onde a energia segue caminhos bem definidos ao longo do meio onde se propaga (Hovem, 2013; Kinsler, 2000). Contudo, os raios acústicos não substituem na íntegra as ondas acústicas, uma vez que são uma aproximação válida carecendo do cumprimento de algumas condições, que veremos mais adiante. No livro de Kinsler (Kinsler, 2000), a equação de onda é dada por

$$\left(\nabla^2 - \frac{1}{c^2(x,y,z)}\frac{\partial^2}{\partial t^2}\right)\, p(x,y,z,t) = 0, \tag{3.1}$$

onde $c = c(x,y,z)$ é a velocidade do som na água, p é a pressão acústica e t é o tempo. A equação (3.1) exprime as oscilações acústicas em termos da pressão acústica que representa as variações espaciais e temporais da pressão em torno de uma pressão média do meio.

Assumindo uma solução da forma

$$p(x,y,z,t) = A(x,y,z)e^{j\omega[t-\Gamma(x,y,z)/c_0]}, \tag{3.2}$$

em que p representa a pressão no ponto (x,y,z) num determinado instante t, A representa a amplitude da variação de p no espaço, ω a frequência oscilatória da pressão, c_0 representa a velocidade de propagação da perturbação acústica e $j = \sqrt{-1}$. A quantidade $\frac{\Gamma(x,y,z)}{c_0}$ é designada por *eikonal*, em que $\Gamma(x,y,z)$ é a variação de fase em termos das coordenadas espaciais. Os valores de (x,y,z) para cada Γ

constante definem superfícies isofásicas de propagação, e como $\Gamma(x, y, z)$ é um campo escalar, recorrendo à definição básica de gradiente, $\nabla\Gamma(x, y, z)$ é perpendicular a essas superfícies em qualquer instante t.

Substituindo (3.2) em (3.1) obtemos

$$-\frac{\nabla^2 A}{A} + \left(\frac{\omega}{c_0}\right)^2 \nabla\Gamma \cdot \nabla\Gamma = \left(\frac{\omega}{c}\right)^2 \tag{3.3}$$

e

$$2\frac{\nabla A}{A} \cdot \nabla\Gamma + \nabla^2\Gamma = 0 \tag{3.4}$$

que representam as partes real e imaginária, respetivamente. A resolução destas equações é complicada devido à sua interdependência e natureza não linear. Todavia, se exigirmos

$$\left|\frac{\nabla^2 A}{A}\right| \ll \left(\frac{\omega}{c}\right)^2 \tag{3.5}$$

pode escrever-se (3.3) de uma forma aproximada simplificada

$$\nabla\Gamma \cdot \nabla\Gamma = \left(\frac{c_0}{c}\right)^2 = n^2 \tag{3.6}$$

dando origem à equação *eikonal*, em que $n = \frac{c_0}{c}$ é o índice de refração. Com esta igualdade, é possível deduzir que

$$\nabla\Gamma = n\hat{s} \tag{3.7}$$

onde $\hat{s}$ é o vetor unitário que nos fornece a direção da propagação do raio acústico em todos os pontos. É possível calcular a trajetória de um raio a partir de um valor conhecido de $\hat{s}$ num ponto (x,y,z) no campo de velocidades e da variação espacial de $\hat{s}$ (Kinsler, 2000). De notar que a direção de propagação do raio é perpendicular à *eikonal*. Como se referiu, a teoria de propagação de raios acústicos carece do cumprimento de algumas condições, nomeadamente a condição expressa na equação (3.5), em que a amplitude da onda de propagação e a velocidade do som não podem variar consideravelmente com a distância quando comparadas com o comprimento de onda (Kinsler, 2000).

3.2 Princípio do tempo mínimo de Fermat

Segundo o princípio do tempo mínimo de Fermat, os raios acústicos propagam-se em trajetos que minimizam o tempo de percurso. Em meios homogéneos, nos quais a velocidade de propagação do som é uniforme, o trajeto que minimiza o tempo de percurso é rectilíneo. Em meios não homogéneos o raio acústico propagar-se-á refratando-se ao longo do seu percurso (Abrahamsson, 2003).

Tendo por base a parametrização vetorial $(x, y, z) = \mathbf{r}(\tau)$, $a \leq \tau \leq b$, do trajeto γ de um raio acústico, o comprimento elementar ds de comprimento de arco é

$$ds = \left\| \frac{d\mathbf{r}}{d\tau} \right\| d\tau. \tag{3.8}$$

Então, o tempo de trajeto $t_{a\to b}$ do raio acústico será

$$t_{a\to b} = \int_\gamma \frac{ds}{c}, \tag{3.9}$$

em que $c = c(x, y, z)$ representa a velocidade do som no meio de propagação. Notemos que (3.9) pode reescrever-se como

$$t_{a\to b} = \int_a^b F(x', y', z', x, y, z)\, d\tau, \tag{3.10}$$

com

$$F(x', y', z', x, y, z) = \frac{\sqrt{\left(\frac{dx}{d\tau}\right)^2 + \left(\frac{dy}{d\tau}\right)^2 + \left(\frac{dz}{d\tau}\right)^2}}{c(x, y, z)}, \tag{3.11}$$

pois

$$ds = \left\| \frac{d\mathbf{r}}{d\tau} \right\| d\tau = \sqrt{\left(\frac{dx}{d\tau}\right)^2 + \left(\frac{dy}{d\tau}\right)^2 + \left(\frac{dz}{d\tau}\right)^2}\, d\tau. \tag{3.12}$$

Na Figura 3.1 podemos observar a representação do trajeto γ, entre $[\gamma_a, \gamma_b]$, de um raio acústico e o esboço da respetiva parametrização vetorial $\mathbf{r}(\tau)$ e de um comprimento elementar de arco Δs. Como sugere a figura, $\|\Delta\bar{\mathrm{r}}\| = \|\bar{\mathrm{r}}(\tau) - \bar{\mathrm{r}}(\tau + \Delta\tau)\|$ é o comprimento da linha que une os pontos τ e $\tau + \Delta\tau$, e isto permite aproximar o comprimento do arco Δs. Note-se que ao aumentar número de pontos na parametrização, melhor será a aproximação do comprimento do arco.

De acordo com o princípio do tempo mínimo de Fermat, o trajeto γ adotado por um raio acústico deverá minimizar (3.10).

3.3 Cálculo variacional

A determinação do trajeto do raio acústico, isto é, a determinação do trajeto que minimiza (3.10) recorre ao conceito de *variação* (Lanczos, 1970) do cálculo variacional. A partir deste conceito é possível deduzir as Equações de Euler-Lagrange tendo como ponto de partida (3.11). As Equações de Euler-Lagrange são equações diferenciais cuja solução é precisamente o trajeto γ que o raio acústico assume. Existem várias abordagens para a dedução das equações de Euler-Lagrange, como

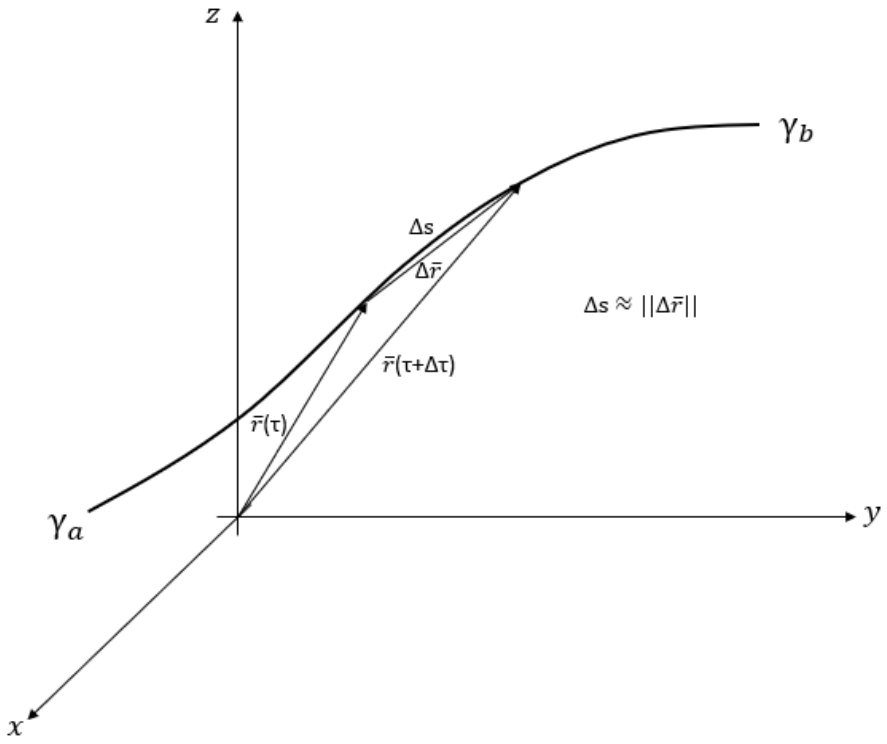


FIGURA 3.1: Representação do trajeto γ, entre $[\gamma_a, \gamma_b]$, de um raio acústico e o esboço da respetiva parametrização vetorial $\mathbf{r}(\tau)$ e de um comprimento elementar de arco Δs.

se pode ver nos trabalhos realizados por Lanczos em (Lanczos, 1970), Simmons em (Simmons, 2016), Lemons em (Lemons, 1997) e por Max Born e Emil Wolf em (Born & Wolf, 1964), realçando a dedução feita com cinco variáveis por estes últimos autores em (Born & Wolf, 1964).

É possível mostrar que as Equações de Euler-Lagrange (Born & Wolf, 1964), obtidas a partir de (3.11) conhecendo o campo $c = c\,(x, y, z)$, têm a forma:

$$\begin{cases} \frac{d\left(\frac{\partial F}{\partial x'}\right)}{d\tau} - \frac{\partial F}{\partial x} = 0 \\ \frac{d\left(\frac{\partial F}{\partial y'}\right)}{d\tau} - \frac{\partial F}{\partial y} = 0 \\ \frac{d\left(\frac{\partial F}{\partial z'}\right)}{d\tau} - \frac{\partial F}{\partial z} = 0 \end{cases} \tag{3.13}$$

Como se referiu, a solução destas equações diferenciais corresponde ao trajeto do raio acústico que minimiza o tempo de percurso do mesmo (Helliwell & Sahakian, 2020).

3.4 Resolução numérica de equações diferenciais

As equações diferenciais ordinárias relacionam funções desconhecidas com as suas derivadas de diferentes ordens. A forma geral de uma equação diferencial

ordinária de ordem um pode ser descrita como

$$F\left(x', x, t\right) = 0, \tag{3.14}$$

em que $x = x\left(t\right)$ representa a função incógnita procurada.

O problema apresentado em (3.13) pode apresentar-se na forma de problema de valores iniciais ou problema com valores de fronteira, formas estas que são determinantes nas técnicas de resolução.

Como a generalidade das equações diferenciais não possui solução analítica torna-se necessário recorrer a técnicas numéricas para aproximar a solução das mesmas.

Se o problema (3.13) assumir a forma do problema de valores iniciais

$$\begin{cases} x'\left(t\right) = f\left(t, x\left(t\right)\right) \\ x\left(t_0\right) = x_0 \end{cases}, \tag{3.15}$$

são bem conhecidas as técnicas clássicas de resolução numérica que se baseiam na argumentação que se segue.

Ao integrar, entre t e $t + h$, ambos os membros de $x'\left(t\right) = f\left(t, x\left(t\right)\right)$ somos conduzidos à expressão

$$x\left(t + h\right) = x\left(t\right) + \int_t^{t+h} f\left(\xi, x\left(\xi\right)\right) d\xi \tag{3.16}$$

que sugere a possibilidade de conhecer $x\left(t + h\right)$ conhecendo $x\left(t\right)$ e estimando

$$\int_t^{t+h} f\left(\xi, x\left(\xi\right)\right) d\xi. \tag{3.17}$$

Seja $t \in [t_0, T]$,

$$t_0 < t_1 < \cdots < t_{n-1} < t_n = T$$

que representa uma malha do intervalo indicado e

$$h_i = t_i - t_{i-1}, i = 1, \ldots, N$$

os correspondentes passos da malha.

Tendo presente a discretização indicada é habitual utilizar uma instância adequada da família de métodos de Runge-Kutta na aproximação numérica da solução. Os métodos de Runge-Kutta foram desenvolvidos por Carl Runge e Martin Kutta. Estes métodos de passo simples no qual o integral (3.16) é aproximado recorrendo à seguinte expressão genérica

$$h \sum_{j=1}^{m} \beta_j f\left(t_{ij}, x\left(t_{ij}\right)\right) \tag{3.18}$$

com $h = t_{i+1} - t_i, t_{ij} = t_i + \rho_j h, 0 \leq \rho_j \leq 1$. Note-se que t_{ij} representam novos nós

no intervalo $[t_i, t_{i+1}]$. Nestes novos nós os valores $x(t_{ij})$ são estimados recorrendo à fórmula

$$x(t_{ij}) = x_i + h\sum_{l=1}^{m} \gamma_{jl} f(t_{il}, x(t_{il})). \tag{3.19}$$

Combinando (3.17) com (3.18) resulta método de Runge-Kutta de m passos,

$$x_{i+1} = x_i + h\sum_{j=1}^{m} \beta_j k_j, \tag{3.20}$$

com

$$k_j = f\left(t_i + \rho_j h, x_i + h\sum_{j=1}^{m} \gamma_{jl} k_l\right). \tag{3.21}$$

Os coeficientes necessários são vulgarmente apresentados nas seguintes matrizes de Butcher:

$$\begin{array}{c|ccc} \rho_1 & \gamma_{11} & \cdots & \gamma_{1m} \\ \vdots & \vdots & & \vdots \\ \rho_m & \gamma_{m1} & \cdots & \gamma_{mm} \\ \hline & \beta_1 & \cdots & \beta_m \end{array}.$$

O método de Runge-Kutta de ordem 4 é dado por

$$x_{i+1} = x_i + h\left[\frac{1}{6}k_1 + \frac{1}{3}k_2 + \frac{1}{3}k_3 + \frac{1}{6}k_4\right], \tag{3.22}$$

com

$$\begin{aligned} k_1 &= f(t_i, x_i), \\ k_2 &= f\left(t_i + \frac{1}{2}h, x_i + \frac{1}{2}hk_1\right), \\ k_3 &= f\left(t_i + \frac{1}{2}h, x_i + \frac{1}{2}hk_2\right), \\ k_4 &= f(t_i + h, x_i + hk_3). \end{aligned}$$

Note-se que esta classe de técnicas é aplicável a problemas de valores iniciais na forma explícita. Infelizmente nem sempre os problemas a resolver têm esta forma, o que exige a utização de outras técnicas (Pina, 2010).

As equações que descrevem os modelos paramétricos de Raposo em (Raposo, 2019), caracterizam-se por serem sistemas de equações algébrico-diferenciais, com equações implícitas não lineares. Neste caso, além da resolução numérica destes sistemas ser bastante mais técnica e exigente, os sistemas não podem ser colocados na forma de Cauchy o que implica que a resolução numérica dos mesmos seja através de algoritmos ímplicitos. Posto isto, o autor utilizou um algoritmo programado por Tony Roberts (Roberts, 1998), em *MATLAB*®, que se destina à resolução de

sistemas de equações algébrico-diferenciais. Nas ferramentas desenvolvidas, esse mesmo programa foi utilizado para a resolução numérica das equações.

3.5 Propagação acústica 2D no oceano

É habitual admitir que no oceano a propagação de um raio acústico ocorre num plano vertical que interseta a superfície numa geodésica. Esta assunção pressupõe a ausência de correntes e a inexistência de variações significativas da velocidade de som em direções transversais relativamente ao plano vertical referido e simplifica o cálculo da trajetória que pode passar a ser descrita a duas dimensões, a cota z e a coordenada horizontal local x (no plano de propagação). Nestas circunstâncias, o azimute do raio acústico ao longo deste plano não será constante ao longo da sua trajetória.

Sendo λ o vetor unitátio tangente a um ponto da curva e μ um vetor unitário perpendicular a λ nesse plano, permitem definir a direção do vetor normal à curva. Uma curva numa superfície pode ter uma curvatura tal que não é plausível introduzi-la num plano, que é o caso das geodésicas. Isto deve-se ao facto de a curva da geodésica conter duas curvaturas ou torsões — Figura 3.2.

Para simplificar o seu cálculo considerou-se apenas curvas planas, ou seja, curvas com uma única curvatura, o que limitará o comprimento do arco da curvatura. Admitindo que $z = z(x)$ descreve o plano da curva em termos das coordenadas espaciais (x, z), em termos do comprimento de arco, s, pode-se formular $x = x(s)$ e $z = z(s)$ — a base matemática por detrás do cálculo da geodésica encontra-se detalhada no livro de Christopher Jekeli (Jekeli, 2006).

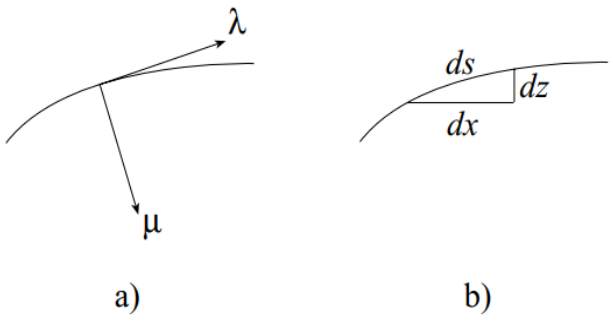


FIGURA 3.2: Representação da curvatura de curvas planas com a descrição do plano da curvatura em termos das coordenadas espacias (x, z).

Fonte: Jekeli, 2006.

As linhas geodésicas são curvas torsas que representam a distância mais curta entre dois pontos numa superfície. Numa esfera a geodésica é um círculo máximo, ao passo que no plano caracterizam-se por segmentos de reta (Jekeli, 2006). Na Figura 3.3 está representado um paralelo, uma secção normal e um vetor unitário que define a direção da normal à superfície (ξ). Neste figura é possível constatar

que a secção normal é um círculo máximo na esfera, e, por conseguinte, trata-se de uma geodésica.

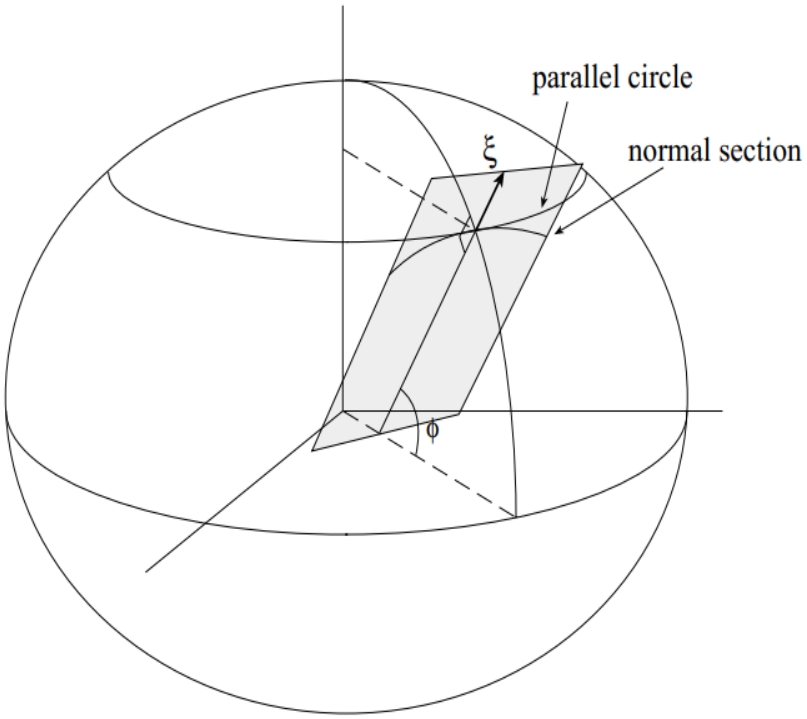


FIGURA 3.3: Representação da secção normal (geodésica) numa esfera.

Fonte: Jekeli, 2006.

A geodesia tem como finalidade calcular as coordenadas geodésicas, distância e direção num determinado elipsoide de referência. Existem várias equações que permitem calcular as linhas geodésicas e as suas coordenadas, no entanto, todas elas têm um nível de complexidade elevado o que dificulta também a sua computação. A escolha das equações dependem do comprimento da linha geodésica (distância) e a exatidão que se pretende obter. Todas as equações estão limitadas pela distância, visto que se ultrapassar um certo valor — varia consoante as fórmulas escolhidas — a exatidão dos cálculos começa a diminuir consideravelmente (Ewing & Mitchell, 1970).

Caso as distâncias de propagação sejam diminutas (inferiores a 80 quilómetros) o cálculo de variação de latitude $\Delta\varphi$, da variação de longitude ΔL e da convergência de meridianos $\Delta\alpha$ podem ser feitos através das fórmulas de transporte de coordenadas geodésicas adaptadas do Manual de Hidrografia (Fernandes, 1967) — ver apêndice C. Dado esta restrição, o estudo da propagação do som será feito até distâncias da ordem dos 80 quilómetros. Durante a utilização destas fórmulas, as latitudes Norte (N) e as longitudes Oeste (O) são introduzidas com sinal positivo, enquanto as latitudes Sul (S) e as longitudes Este (E) com o sinal negativo, por uma questão de convenção.

3.6 Interpolação

A interpolação, na matemática, é um método que, ao aproximar valores de conjuntos discreto de dados locais previamente conhecidos, permite construir um novo conjunto de dados.

Com a interpolação, é possível deduzir uma função que englobe estes dados locais, conferindo-lhes o conceito de continuidade. Por outras palavras, a interpolação permite aproximar uma função a um grupo de dados locais, podendo posteriormente prever novos dados. No entanto, visto ser uma aproximação, há sempre erros associados às projeções (Bankman, 2009).

Existem vários tipos de interpolação, contudo, neste trabalho, foram apenas utilizadas interpolações multilineares e interpolações através do vizinho mais próximo.

3.6.1 Interpolação multilinear

Na interpolação linear, através do ajuste de uma curva utilizando polinómios, é possível construir um novo conjunto de dados dentro do intervalo de um conjunto de pontos de dados discretos conhecidos. Porém, caso a matriz de dados discretos conhecidos tenha uma dimensão superior a duas dimensões, o novo conjunto de dados não estará sobre uma reta ou um polinómio, mas sim dentro de um polígono (Wagner, 2008).

A interpolação bilinear e trilinear, são uma extensão da interpolação linear que operam em espaços bidimensionais e tridimensionais, respetivamente. Nos programas desenvolvidos é necessário recorrer a estas interpolações, bilinear e trilinear, para associar a velocidade do som à posição geográfica do raio acústico — será detalhado no subcapítulo 5.4.

Seguidamente vai ser discriminado o processo de interpolação trilinear, e que facilmente se justifica para os casos de interpolação linear ou superior.

Considere-se $v = f(x, y, z)$ uma função quase ou rigorosamente linear e A, B, C, D, E, F, G e H, pontos que definem um paralelepípedo.

Na Figura 3.4, o paralelepípedo $ABCDEFGH$ é decomposto em oito volumes pelos planos $x = x_2$, $y = y_2$ e $z = z_2$, a fim de interpolar o valor de v_8 na função f no ponto I — note-se que as coordenadas do ponto I são (x_2, y_2, z_2) — e, para isso, calcula-se os volumes normalizados do retângulo decomposto (Wagner, 2008).

Os volumes normalizados N_a, N_b, N_c, N_d,N_e, N_f, N_g e N_h, são dados por

$$N_a = \frac{(x_1 - x_2) \cdot (y_1 - y_2) \cdot (z_2 - z_0)}{(x_1 - x_0) \cdot (y_1 - y_0) \cdot (z_1 - z_0)},$$

$$N_b = \frac{(x_1 - x_2) \cdot (y_2 - y_0) \cdot (z_2 - z_0)}{(x_1 - x_0) \cdot (y_1 - y_0) \cdot (z_1 - z_0)},$$

$$N_c = \frac{(x_2 - x_0) \cdot (y_1 - y_2) \cdot (z_2 - z_0)}{(x_1 - x_0) \cdot (y_1 - y_0) \cdot (z_1 - z_0)},$$

$$N_d = \frac{(x_2 - x_0) \cdot (y_2 - y_0) \cdot (z_2 - z_0)}{(x_1 - x_0) \cdot (y_1 - y_0) \cdot (z_1 - z_0)},$$

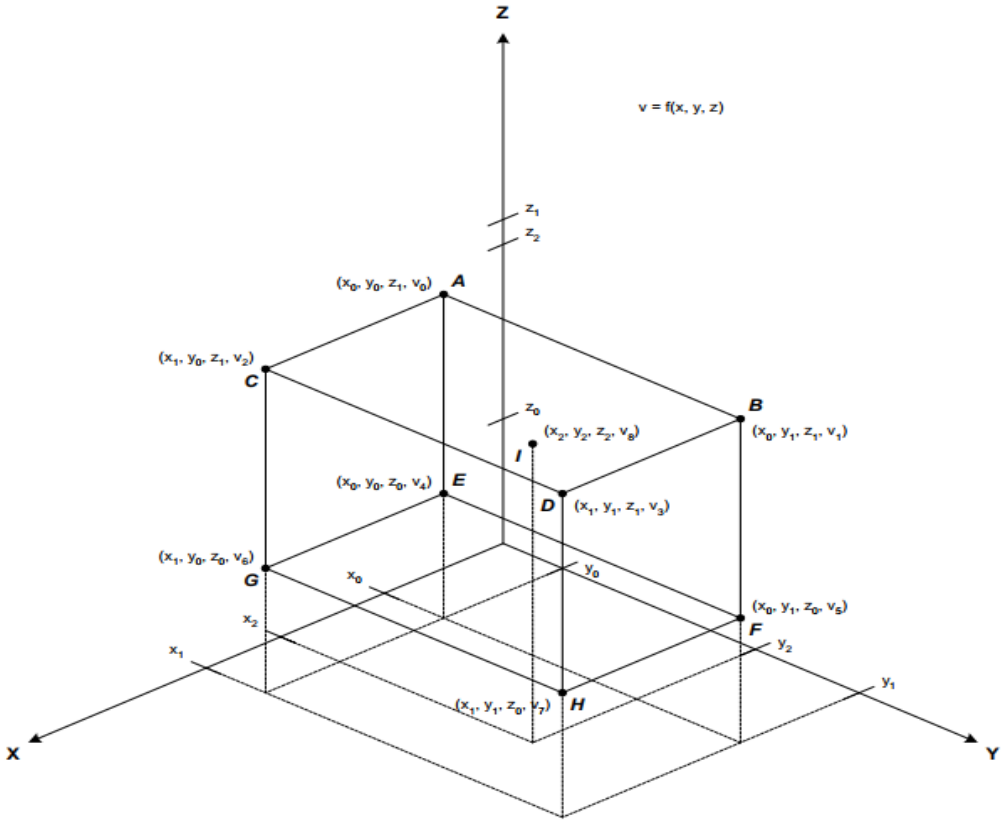


FIGURA 3.4

Fonte: Wagner, 2008.

$$N_e = \frac{(x_1 - x_2) \cdot (y_1 - y_2) \cdot (z_1 - z_2)}{(x_1 - x_0) \cdot (y_1 - y_0) \cdot (z_1 - z_0)},$$
$$N_f = \frac{(x_1 - x_2) \cdot (y_2 - y_0) \cdot (z_1 - z_2)}{(x_1 - x_0) \cdot (y_1 - y_0) \cdot (z_1 - z_0)},$$
$$N_g = \frac{(x_2 - x_0) \cdot (y_1 - y_2) \cdot (z_1 - z_2)}{(x_1 - x_0) \cdot (y_1 - y_0) \cdot (z_1 - z_0)},$$
$$N_h = \frac{(x_2 - x_0) \cdot (y_2 - y_0) \cdot (z_1 - z_2)}{(x_1 - x_0) \cdot (y_1 - y_0) \cdot (z_1 - z_0)}.$$

Desta forma, é então possível calcular o valor de v_8 na função f no ponto I através da equação

$$v_8 = v_0 \cdot N_a + v_1 \cdot N_b + v_2 \cdot N_c + v_3 \cdot N_d + v_4 \cdot N_e + v_5 \cdot N_f + v_6 \cdot N_g + v_7 \cdot N_h. \quad (3.23)$$

3.6.2 Interpolação pelo vizinho mais próximo

O método de interpolação pelo vizinho mais próximo atribui o valor de um dado conhecido a um dado desconhecido, simplesmente pela sua proximidade ao mesmo.

Considere-se uma base de dados composta por n objetos O_1, O_2, ..., O_n, e um objeto O_q, contido ou não na base de dados. O método pela pesquisa do vizinho mais próximo (método NN - *Nearest Neighbor*) procura o objeto que $O_{NN} \neq O_q$ e que tenha a menor distância a O_q do que qualquer outro objeto na base de dados (Papadopoulos & Manolopoulos, 2006).

Visto que o método procura a proximidade entre objetos, é necessário calcular a distância aritmética. Comummente o cálculo das distâncias é feito através da distância Euclidiana e da distância de Manhattan. Nos programas desenvolvidos, a aplicação do método do vizinho mais próximo é feito atendendo à distância Euclidiana.

Dados dois vetores x e y em $\mathbb{R}^n$, a distância Euclidiana (L_2) é determinada através da equação

$$L_2(x, y) = \left(\sum_{j=1}^{n} \mid x_j - y_j \mid^2 \right)^{\frac{1}{2}}, \tag{3.24}$$

em que x_j e y_j representam as coordenadas dos vetores x e y, de modo respetivo.

Capítulo 4

Formulação da propagação de raios acústicos com reflexões em ambiente real

4.1 Formulação do problema

O objetivo deste trabalho é modelar a propagação de raios acústicos com reflexões, utilizando o modelo de Pedro Raposo (Raposo, 2019), em ambiente real. Para a construção de ferramentas com estas capacidades levantam-se alguns problemas como a resolução das equações dos modelos de Pedro Raposo, a reflexão dos raios acústicos nas fronteiras, a modelação da batimetria real e o cálculo da velocidade de som e respetivos gradientes espaciais reais.

Comecemos pela explicação da formulação dos modelos de propagação de Pedro Raposo (Raposo, 2019). Como referido no capítulo 3, a abordagem utilizada para a modelação da propagação acústica consiste em considerar que a propagação do som se dá ao longo de raios que são perpendiculares à frente de onda, e as equações que modelam as trajetórias desses raios são deduzidas a partir do princípio do tempo mínimo de Fermat. Considerando γ uma determinada trajetória entre os pontos a e b, o tempo de trajeto t é dado por

$$t_{a\to b} = \int_\gamma \frac{ds}{c\,(x,y,z)} = \int_\gamma \frac{\sqrt{\left(\frac{dx}{dt}\right)^2 + \left(\frac{dy}{dt}\right)^2 + \left(\frac{dz}{dt}\right)^2}}{c\,(x,y,z)}\, dt \tag{4.1}$$

para o caso tridimensional, e

$$t_{a\to b} = \int_\gamma \frac{ds}{c\,(x,y)} = \int_\gamma \frac{\sqrt{\left(\frac{dx}{dt}\right)^2 + \left(\frac{dy}{dt}\right)^2}}{c\,(x,y)}\, dt \tag{4.2}$$

para o caso bidimensional. A aplicação do princípio de Fermat ao caso bidimensional,

permite deduzir a equação diferencial

$$y''(x) = \frac{1 + y'(x)^2}{c\,(x,y)} \left(y'(x) \frac{\partial c\,(x,y)}{\partial x} - \frac{\partial c\,(x,y)}{\partial y} \right), \tag{4.3}$$

cuja solução representa a trajetória dos raios acústicos a duas dimensões, como se pode consultar em (Lu et al., 2000), (Cai et al., 2012) e (Paiva, 2018). No entanto, a utilização desta equação diferencial apresenta limitações na simulação de trajetórias cujas direções sejam normais ao eixo coordenado associado à variável x. A resolução desta limitação passou pela formulação das equações de Euler-Lagrange em termos das coordenadas espaciais devidamente parametrizadas em termos do comprimento do arco γ — consulte-se (Raposo, 2019). A partir das parametrizações $\mathbf{r}(\tau) = (x(\tau), y(\tau)), \tau \in [a,b]$ e $\mathbf{r}(\tau) = (x(\tau), y(\tau), z(\tau)), \tau \in [a,b]$, é possível reescrever (4.1) e (4.2), como

$$t_{a\to b} = \int_\gamma \frac{\sqrt{\left(\frac{dx}{d\tau}\right)^2 + \left(\frac{dy}{d\tau}\right)^2 + \left(\frac{dz}{d\tau}\right)^2}}{c\,(x(\tau), y(\tau), z(\tau))}\, d\tau \tag{4.4}$$

e

$$t_{a\to b} = \int_\gamma \frac{ds}{c\,(x,y)} \;=\; \int_\gamma \frac{\sqrt{\left(\frac{dx}{d\tau}\right)^2 + \left(\frac{dy}{d\tau}\right)^2}}{c\,(x(\tau), y(\tau))}\, d\tau, \tag{4.5}$$

e obter as seguintes equações de Euler-Lagrange:

$$\begin{cases} \left((y')^2 + (z')^2\right) x''c \;-\; \left(y'y'' + z'z''\right) x'c \\ -\; \left((x')^2 + (y')^2 + (z')^2\right) x' \frac{dc}{d\tau} + \frac{dc}{dx} \left((x')^2 + (y')^2 + (z')^2\right)^2 = 0 \\ \left((x')^2 + (z')^2\right) y''c \;-\; \left(x'x'' + z'z''\right) y'c \\ -\; \left((x')^2 + (y')^2 + (z')^2\right) x' \frac{dc}{d\tau} + \frac{dc}{dy} \left((x')^2 + (y')^2 + (z')^2\right)^2 = 0 \\ \left((y')^2 + (x')^2\right) z''c \;-\; \left(y'y'' + x'x''\right) z'c \\ -\; \left((x')^2 + (y')^2 + (z')^2\right) z' \frac{dc}{d\tau} + \frac{dc}{dz} \left((x')^2 + (y')^2 + (z')^2\right)^2 = 0 \end{cases} \tag{4.6}$$

para as trajetórias de raios acústicos a três dimensões, e

$$\begin{cases} x''y' - y''x' + \frac{1}{c(x,y)} \frac{\partial c}{\partial x} \left((y')^3 + (x')^2 y'\right) - \frac{1}{c(x,y)} \frac{\partial c}{\partial y} \left((x')^3 + (y')^2 x'\right) = 0 \\ y''x' - x''y' + \frac{1}{c(x,y)} \frac{\partial c}{\partial y} \left((x')^3 + (y')^2 x'\right) - \frac{1}{c(x,y)} \frac{\partial c}{\partial x} \left((y')^3 + (x')^2 y'\right) = 0 \end{cases} \tag{4.7}$$

para as trajetórias de raios acústicos a duas dimensões.

Infelizmente as equações de Euler-Lagrange obtidas não são linearmente independentes, pelo que a determinação da solução em termos das variáveis espaciais escolhidas, exige a consideração de relações adicionais que permitam determinar os modelos 2D e 3D. Na dissertação de mestrado de Pedro Raposo (Raposo, 2019), pode-se encontrar a argumentação que conduz à resolução do problema acabado de

referir e que permitiu obter os sistemas de equações diferenciais (4.10) e (4.11), que representam os modelos 2D e 3D de propagação de raios acústicos.

No problema 2D, uma das equações foi simplesmente substituída por uma expressão equivalente à relação

$$\left(\frac{dx}{d\tau}\right)^2 + \left(\frac{dy}{d\tau}\right)^2 = 1. \tag{4.8}$$

Analogamente, no problema 3D uma das equações foi substituída por uma expressão equivalente à relação

$$\left(\frac{dx}{d\tau}\right)^2 + \left(\frac{dy}{d\tau}\right)^2 + \left(\frac{dz}{d\tau}\right)^2 = 1. \tag{4.9}$$

Apresenta-se seguidamente as equações obtidas desta forma, devidamente reduzidas à primeira ordem:

$$\begin{cases} x' - a = 0 \\ y' - b = 0 \\ a'b - b'a + \frac{b}{c(x,y)}\frac{\partial c}{\partial x} - \frac{a}{c(x,y)}\frac{\partial c}{\partial y} = 0 \\ aa' + bb' = 0 \end{cases} \tag{4.10}$$

e

$$\begin{cases} x' - a = 0 \\ y' - b = 0 \\ z' - e = 0 \\ (1-a')a' + (1-e^2)e' - ae(a'+e') - b(ab'+eb') \\ \quad + \frac{1}{c}\left(\frac{\partial c}{\partial x} + \frac{\partial c}{\partial z}\left(\frac{dc}{dx}a + \frac{dc}{dy}b + \frac{dc}{dz}e\right)(a+e)\right) = 0 \\ (1-b')b' + (1-e^2)e' - be(b'+e') - a(ba'+ea') \\ \quad + \frac{1}{c}\left(\frac{\partial c}{\partial y} + \frac{\partial c}{\partial z}\left(\frac{dc}{dx}a + \frac{dc}{dy}b + \frac{dc}{dz}e\right)(b+e)\right) = 0 \\ aa' + bb' + ee' = 0 \end{cases}, \tag{4.11}$$

onde x, y, z representam, respetivamante, a posição do raio acústico em coordenadas cartesianas e c a velocidade do som. As primeiras derivadas de x, y, z, simbolizadas por a, b, e, representam a variação do tempo, já as segundas derivadas de x, y, z, simbolizadas por a', b', e', representam a acelaração.

De referir que estes sistemas de equações constituem equações algébrico-diferenciais (Ascher & Petzold, 1998), o que leva a que a resolução destes sistemas possa colocar algumas dificuldades numéricas.

Para além das dificuldades numéricas acabadas de referir, a utilização destes modelos exige o conhecimento da velocidade do som e respetivos gradientes ao longo da propagação do raio acústico

Outro grande desafio encontrado é a modelação da reflexão, considerada especular, dos raios acústicos nas fronteiras ditas reais, ou seja, modelar a matemática associada à reflexão e fazer a interação com a batimetria real.

4.2 Fecho do problema

4.2.1 Cálculo da propagação do raio acústico

Como foi exposto, os modelos de propagação acústica utilizados neste trabalho caracterizam-se por serem equações algébrico-diferenciais, com equações implícitas não lineares, com uma grande complexidade numérica, o que dificulta a sua resolução. Na resolução numérica dos modelos de propagação utilizou-se o algoritmo desenvolvido por Tony Roberts (Roberts, 1998), programado em *MATLAB*®, que se destina justamente à resolução deste tipo de sistemas de equações.

4.2.2 Modelação das reflexões

Para a construção de um modelo de propagação de raios acústicos é fundamental considerar as fronteiras existentes numa coluna de água, sendo estas a superfície e o fundo do mar. Estes limites físicos podem influenciar o modo de propagação da energia da onda acústica através da ação de reflexão, absorção ou refração (Born & Wolf, 1964). Uma onda sonora ao encontrar uma superfície plana cria duas ondas, uma que é refletida novamente para o meio original e outra que é transmitida para o meio onde incidiu, sendo este tipo de reflexão uma reflexão especular. Quando a superfície não é linear, os raios refletem-se para todas as direções, sendo este tipo de reflexão uma reflexão difusa. Para as duas ferramentas, 2D e 3D, irá apenas ser considerado o fenómeno da reflexão especular, ou seja, a capacidade de calcular reflexões especulares do raio acústico nas fronteiras.

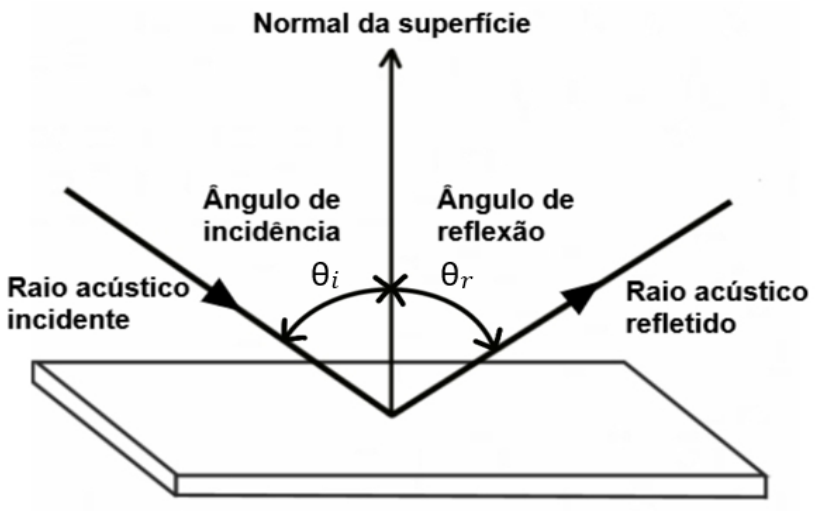


FIGURA 4.1: Ilustração de uma reflexão especular em 3D.

4.2.2.1 A reflexão em 3D

Para o cálculo do raio acústico refletido, $\mathbf{r} = (r_1, r_2, r_3)$, necessitamos previamente de conhecer o raio acústico incidente, $\mathbf{i} = (i_1, i_2, i_3)$, e a normal ao plano

no ponto de incidência, $\mathbf{n} = (n_1, n_2, n_3)$. Como se trata de uma reflexão especular, segundo a lei da reflexão, obrigatoriamente $\theta_i = \theta_r$. Esta equação impõe que o ângulo de incidência, θ_i, seja igual ao ângulo de reflexão, θ_r. Para viabilizar o cálculo, admitiremos que os vetores acima referidos são unitários, satisfazendo

$$\|\mathbf{i}\|^2 = i_1^2 + i_2^2 + i_3^2 = 1, \tag{4.12}$$

$$\|\mathbf{r}\|^2 = r_1^2 + r_2^2 + r_3^2 = 1, \tag{4.13}$$

$$\|\mathbf{n}\|^2 = n_1^2 + n_2^2 + n_3^2 = 1. \tag{4.14}$$

A reflexão, de acordo com as condições acima referidas, é uma aplicação linear ortogonal em $\mathbb{R}^3$. Para materializarmos o cálculo do raio acústico refletido, iremos recorrer à matriz de aplicação linear $\mathbf{R}(\mathbf{n})$, descrita usando a base canónica $\{\mathbf{e_1}, \mathbf{e_2}, \mathbf{e_3}\}$ de $\mathbb{R}^3$ (ver em apêndice A a dedução da matriz $\mathbf{R}(\mathbf{n})$) . Para determinarmos o versor $\mathbf{r}$ que caracteriza o raio acústico refletido basta recorrer ao produto matricial

$$\mathbf{R}(\mathbf{n})\,\mathbf{i} = \mathbf{r}. \tag{4.15}$$

Substituíndo os valores na expressão (4.15), obtemos

$$\underbrace{\begin{bmatrix} 1-2n_1^2 & -2n_1n_2 & -2n_1n_3 \\ -2n_1n_2 & 1-2n_2^2 & -2n_2n_3 \\ -2n_1n_3 & -2n_2n_3 & 1-3n_1^2 \end{bmatrix}}_{\mathbf{R}(\mathbf{n})} \underbrace{\begin{bmatrix} i_1 \\ i_2 \\ i_3 \end{bmatrix}}_{\mathbf{i}} = \underbrace{\begin{bmatrix} r_1 \\ r_2 \\ r_3 \end{bmatrix}}_{\mathbf{r}} \tag{4.16}$$

4.2.2.2 A reflexão em 2D

O cálculo do raio acústico refletido a duas dimensões é semelhante ao caso tridimensional, basta retirar a variável espacial z aos três versores, ficando

$$\mathbf{r} = (r_1, r_2),$$

$$\mathbf{i} = (i_1, i_2),$$

$$\mathbf{n} = (n_1, n_2),$$

passando assim a ser uma aplicação linear ortogonal em $\mathbb{R}^2$. Para o cálculo do raio acústico iremos novamente recorrer à matriz $\mathbf{R}(\mathbf{n})$, mas descrita usando a base canónica $\{\mathbf{e_1}, \mathbf{e_2}\}$ de $\mathbb{R}^2$. A partir da equação (4.15) temos

$$\underbrace{\begin{bmatrix} 1-2n_1^2 & -2n_1n_2 \\ -2n_1n_2 & 1-2n_2^2 \end{bmatrix}}_{\mathbf{R}(\mathbf{n})} \underbrace{\begin{bmatrix} i_1 \\ i_2 \end{bmatrix}}_{\mathbf{i}} = \underbrace{\begin{bmatrix} r_1 \\ r_2 \end{bmatrix}}_{\mathbf{r}}, \tag{4.17}$$

que permite calcular o versor $\mathbf{r}$ no caso bidimensional.

4.2.3 Modelação em ambiente real

A simulação da propagação de raios acústicos em ambiente real permite estudar a propagação sonora, por exemplo, em laboratório em ambiente controlado. Contudo, a criação de um cenário que se assemelhe à realidade é, em todos os sentidos, um desafio. Neste caso em concreto, no desenvolvimento das ferramentas deste trabalho, a única variável física que é necessária saber é a velocidade de propagação do som, dado que se está a estudar somente a propagação no meio, não tendo em conta a absorção e a dissipação acústica. Além das condições climatológicas, para a modelação em ambiente real é também necessário conhecer-se as caracteristicas do fundo, nomeadamente a batimetria.

Tendo por objetivo colmatar os desafios mencionados em cima, foi através do uso de bases de dados credíveis e reconhecidas pela comunidade científica, que obtivemos dados alusivos da batimetria e velocidade de som. Como a velocidade de som é calculada a partir da salinidade, pressão hidrostática e temperatura, optou-se por usar bases de dados com esses elementos e posteriormente calcular a velocidade do som.

Neste sentido, foram utilizadas as base de dados da GEBCO, da WOD18 e a biblioteca de software TEOS-10, que permitiram retirar informação da batimetria e da climatologia, respetivamente, e ainda possibilitou o acesso a funções que permitiram calcular diferentes variáveis oceanográficas, como o caso da velocidade do som.

No próximo capítulo é explicado com mais detalhe os algoritmos das ferramentas desenvolvidas, assim como uma explicação mais detalhada do uso das bases de dados mencionadas, e dos seus pressupostos, e a forma como é modelada a reflexão do raio acústico nas fronteiras.

Capítulo 5

Desenvolvimento dos algoritmos para modelação acústica

Neste capítulo irá apresentar-se os algoritmos destinados a modelar a propagação de raios acústicos em ambiente real, que deram origem aos programas `REOART 2D` e `REOART 3D` (***R**eal **E**nvironment **O**cean-**A**coustic **R**ay **T**racing*), desenvolvidos no presente trabalho.

O modelo de propagação utilizado foi desenvolvido por Pedro Raposo em 2019 (Raposo, 2019), tendo por base a aplicação do princípio do tempo mínimo de Fermat — ver o capítulo 4. O modelo de propagação consiste num sistema de equações algébrico-diferenciais não lineares cuja resolução numérica dotada das condições iniciais apropriadas, permite determinar o trajeto do raio acústico em ambientes espaciais com velocidade de propagação do som conhecidas.

Neste trabalho adicionaremos ao modelo, não só a capacidade para simular a propagação em ambientes oceânicos nos quais a velocidade do som pode ser conhecida ou calculada, como também a modelação de reflexões quer na superfície quer no fundo conhecido do oceano. Para tal, utilizaremos as bases de dados da GEBCO, da WOD18 e a biblioteca de software TEOS-10.

Nos programas desenvolvidos, `REOART 2D` e `REOART 3D`, ao introduzir-se as coordenadas geográficas, entre outros elementos, de uma determinada região é possível retirar-se informação alusiva à topografia e à climatologia permitindo assim estudar a propagação do som na região com dados reais retirados das bases de dados.

No subcapítulo 5.2, serão abordadas as bases de dados utilizadas e como os programas `REOART 2D` e `REOART 3D` utilizam essa informação para determinar a topografia e a climatologia na área de simulação.

Nos subcapítulos 5.3 e 5.4, está referido como os programas calculam a velocidade do som e os gradientes espaciais mediante os dados climatológicos da área de simulação — retirados da base de dados WOD18.

E por fim, no subcapítulo 5.5, está pormenorizado como é calculada a reflexão do raio acústico nas fronteiras — no fundo oceânico e na superfície da água.

Os códigos dos programas foram programados na linguagem MATLAB. Nos apêndices G e H estão expostos os algoritmos dos mesmo. A pasta dos ficheiros está disponível para download no GitHub (para aceder clique aqui).

5.1 Fluxograma

Na Figura 5.1 está ilustrado o fluxograma do funcionamento lógico dos algoritmos dos programas `REOART 2D` e `REOART 3D`. Esta figura permite compreender a ordem das etapas dos programas.

A primeira etapa será a introdução da posição geográfica, do azimute, da distância de propagação e da distância entre posições, determinando assim a zona de simulação, podendo ela ser um plano (2D) ou uma área (3D), consoante o programa selecionado.

Com a área de simulação estipulada, com recurso às bases de dados da GEBCO e da WOD18, a segunda etapa é a caracterização da topografia e da climatologia da área de simulação.

A terceira etapa passa pelo cálculo da velocidade de som e respetivos gradientes espaciais da área através da biblioteca TEOS-10 (GSW), determinando assim os campos de velocidade de som e os gradientes.

Na quarta etapa, é realizada a resolução numérica do modelos de propagação de Raposo (cálculo da trajétoria do raio acústico), através do algoritmo de Tony Roberts, e a modelação das reflexões.

5.2 Base de dados

5.2.1 Dados batimétricos GEBCO

Para a discretização batimétrica do fundo utilizou-se a base de dados disponibilizada pela GEBCO, na resolução espacial de 0,5 milhas, que na versão de 2021 fornece dados batimétricos da generalidade dos mares e oceanos da terra. Nas simulações realizadas, o espaço de simulação foi restringido ao espaço marítimo envolvente a Portugal Continental, e na Figura 5.2 está a demonstração desses dados carregados nos programas.

5.2.1.1 Batimetria em 2D

Os algoritmos de propagação acústica em 2D, em ambiente real, foram desenvolvidos admitindo que em certas circunstâncias o raio acústico pode propagar-se num plano que define um circulo máximo. Assim será se os gradientes transversais da velocidade do som não forem significativos. Nestas circunstâncias, o plano de propagação do raio acústico intersetará a superfície do oceano numa linha geodésica (ver subcapítulo 3.5). A determinação rigorosa das posições geográficas nestas linhas a partir de um ponto inicial e de um azimute inicial recorre a expressões válidas para distâncias inferiores a 80 quilómetros (Fernandes, 1967). Note-se que em geral o azimute definido por estas linhas não é constante ao longo do seu comprimento.

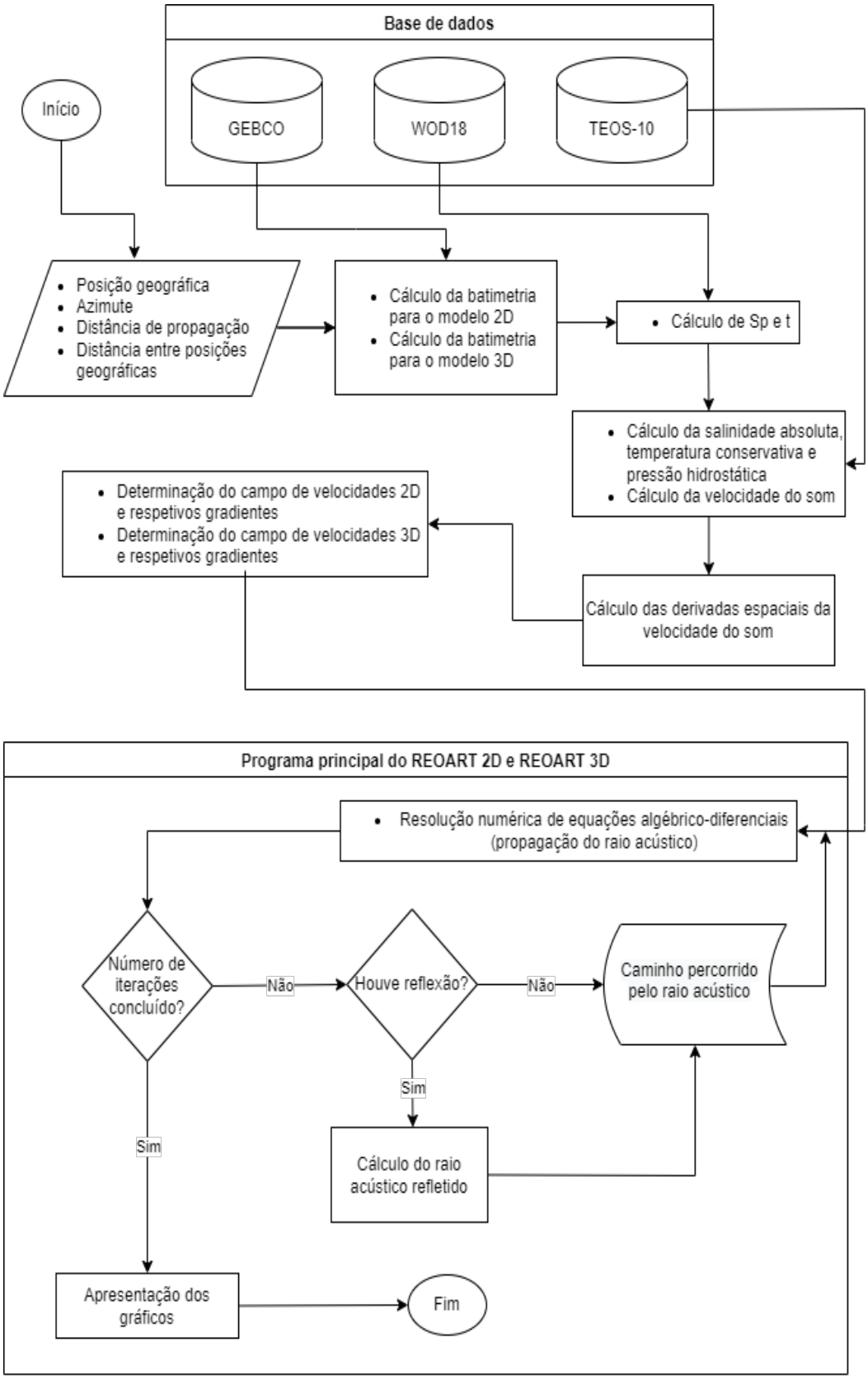


FIGURA 5.1: Fluxograma dos programas `REOART 2D` e `REOART 3D`.

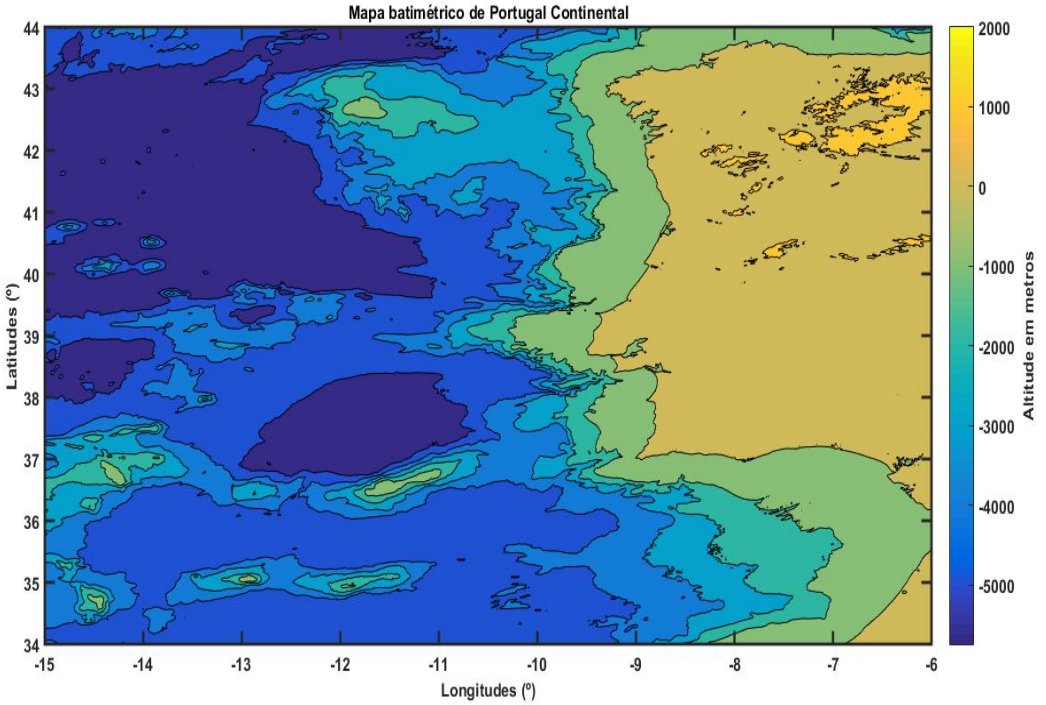


FIGURA 5.2: Mapa batimétrico da área envolvente a Portugal Continental.

A linha vermelha na Figura 5.3 representa uma trajetória que começa na posição geográfica de latitude $36,31°$N e longitude $11°$ O, com o azimute inicial $300°$, até aos 80 quilómetros. Com a trajetória calculada e bem definida no mapa batimétrico, é possível retirar o perfil batimétrico 2D da trajetória, como se pode ver na Figura 5.4. Nessa figura, no eixo do x está discretizada a distância, em quilómetros, à fonte de emissão, e no eixo do y está representada a profundidade, em metros. O círculo vermelho, comum em ambas as figuras, representa a posição geográfica inicial introduzida.

5.2.1.2 Batimetria em 3D

No caso de se verificarem gradientes transversais de velocidade de som, a propagação do raio acústico não se cingirá a um plano de propagação. Nesta situação, a simulação da propagação terá de ser realizada a 3 dimensões.

No programa 3D, analogamente ao programa 2D, para se obter a batimetria tem de se estabelecer a princípio uma área de simulação. Na Figura 5.5, está representado um quadrado vermelho que simboliza essa mesma área, onde o eixo do x representa as longitudes, em graus, e o eixo do y representa as latitudes, em graus. Note-se que um dos vértices do quadrado coincide com o ponto geográfico inicial utilizado para a obtenção do perfil batimétrico a 2D. A Figura 5.6 corresponde ao perfil batimétrico 3D da área de simulação (o quadrado vermelho). Nos eixos x e y manter-se-ão as longitudes e latitudes, respetivamente, e o eixo do z representa as profundidades da área.

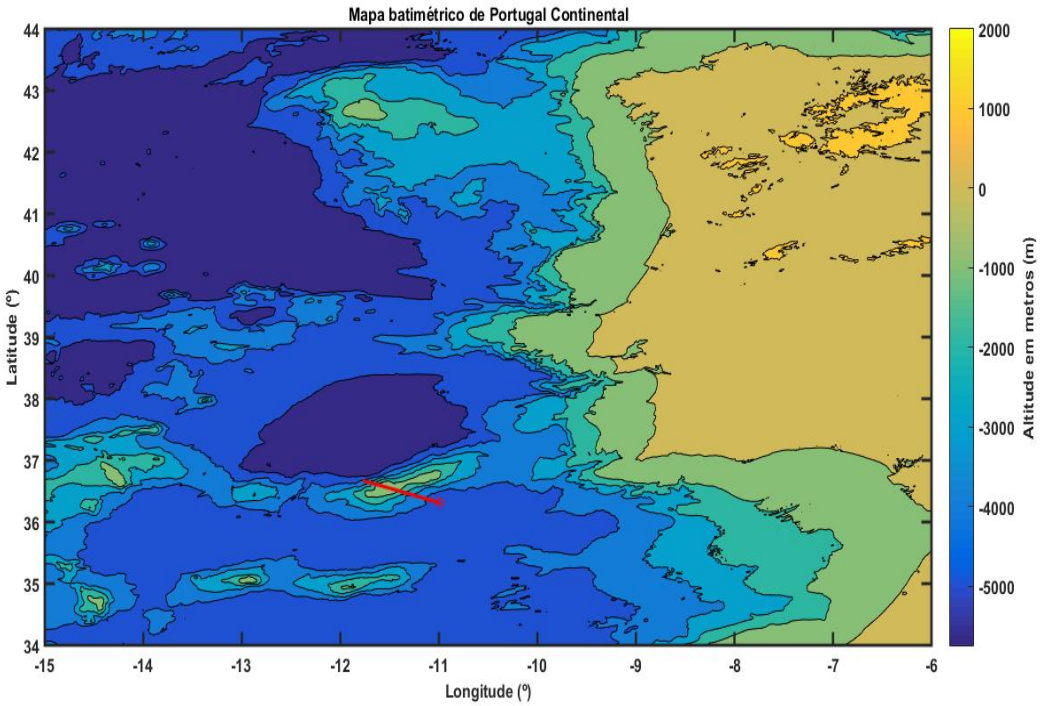


FIGURA 5.3: Mapa batimétrico da área envolvente a Portugal Continental com a representação da trajetória (geodésica).

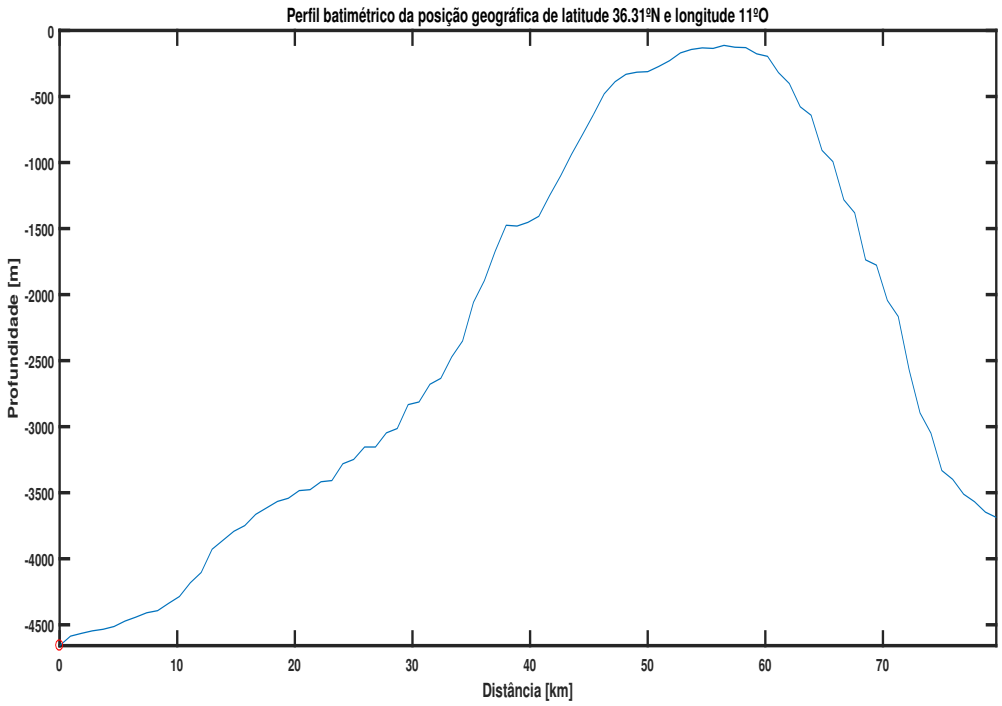


FIGURA 5.4: Perfil batimétrico da posição geográfica de latitude $36,31°$N e longitude $11°$ O, ao longo do azimute $300°$, até à distância de 80 quilómetros.

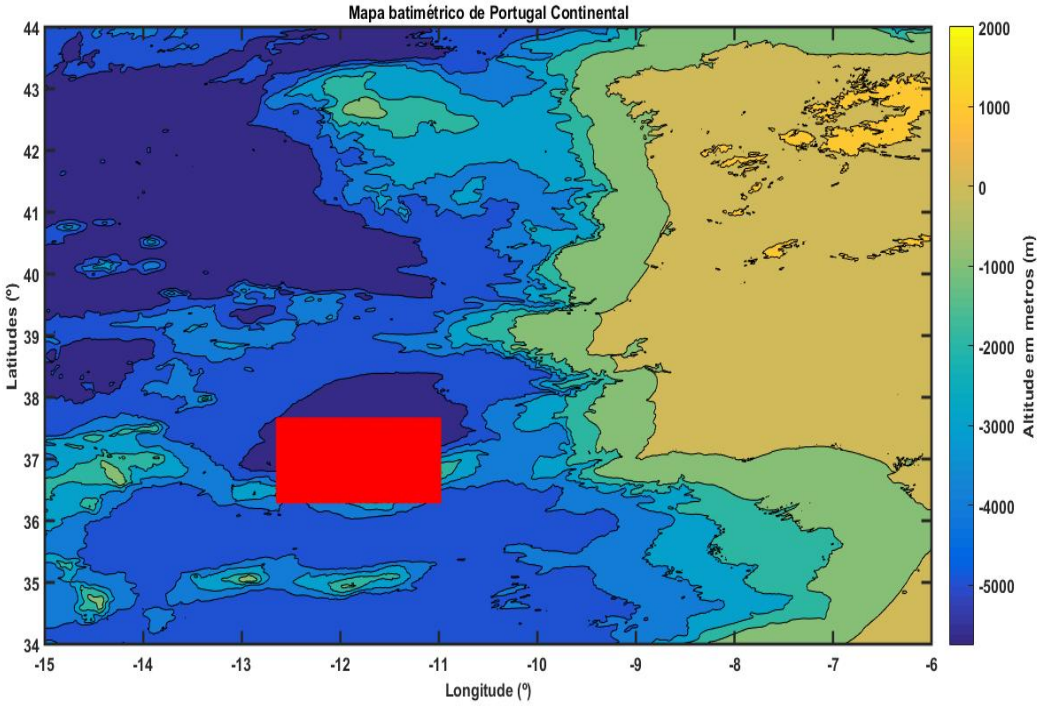


FIGURA 5.5: Mapa batimétrico da área envolvente a Portugal Continental com a área de simulação 3D (assinalada a cor vermelha).

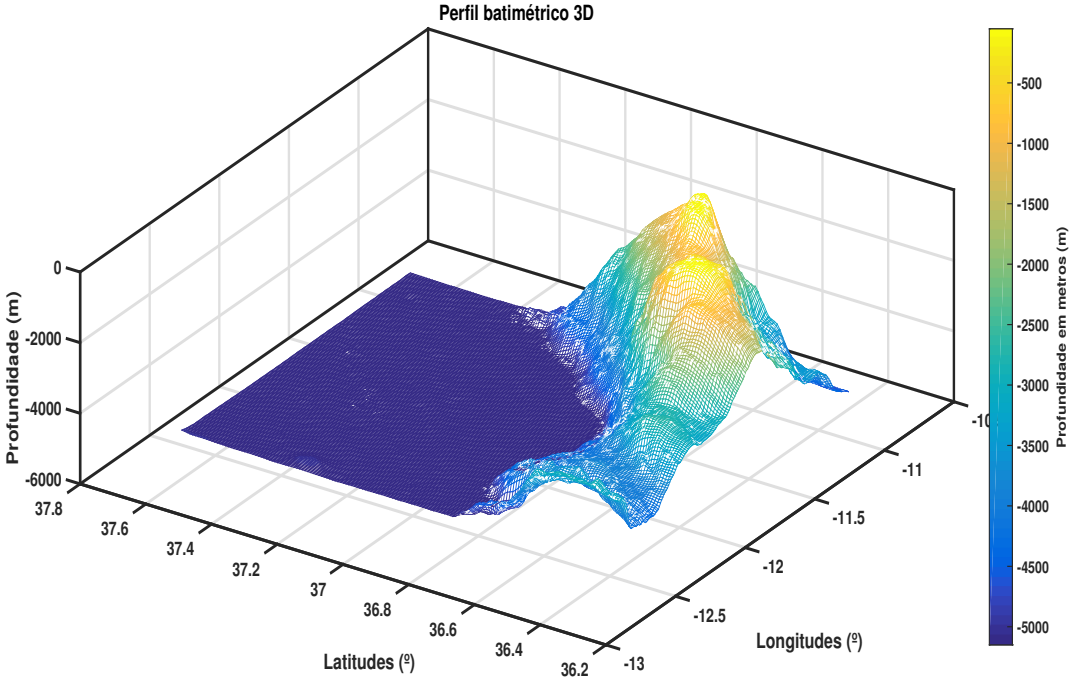


FIGURA 5.6: Perfil batimétrico 3D da área definida na figura (5.5).

5.2.2 Dados climatológicos WOD18

A WOD18 é uma base de dados disponibilizada pela NOAA (*National Oceanic and Atmospheric Administration*) que contém vários tipos de dados com interesse oceanográfico. Esta base de dados pode disponibilizar dados de acordo com os meses e as estações do ano com resoluções espaciais de cinco graus, um grau e um quarto de grau de latitude e longitude. Em termos de discretização climatológica da coluna de água, há dados até aos 5.000 metros de profundidade — de 5 em 5 metros até aos 100 metros, e de 25 em 25 metros até aos 5.000 metros. Nos programas `REOART 2D` e `REOART 3D` foram utilizados os dados alusivos à salinidade prática e à temperatura *in situ*[1]. Na Figura 5.7 pode-se observar os dados climatológicos correspondentes à distribuição espacial da salinidade prática e da temperatura *in situ* da coluna de água, na resolução de um quarto de grau, da média anual, representados por circunferências azuis, na posição geográfica respetiva.

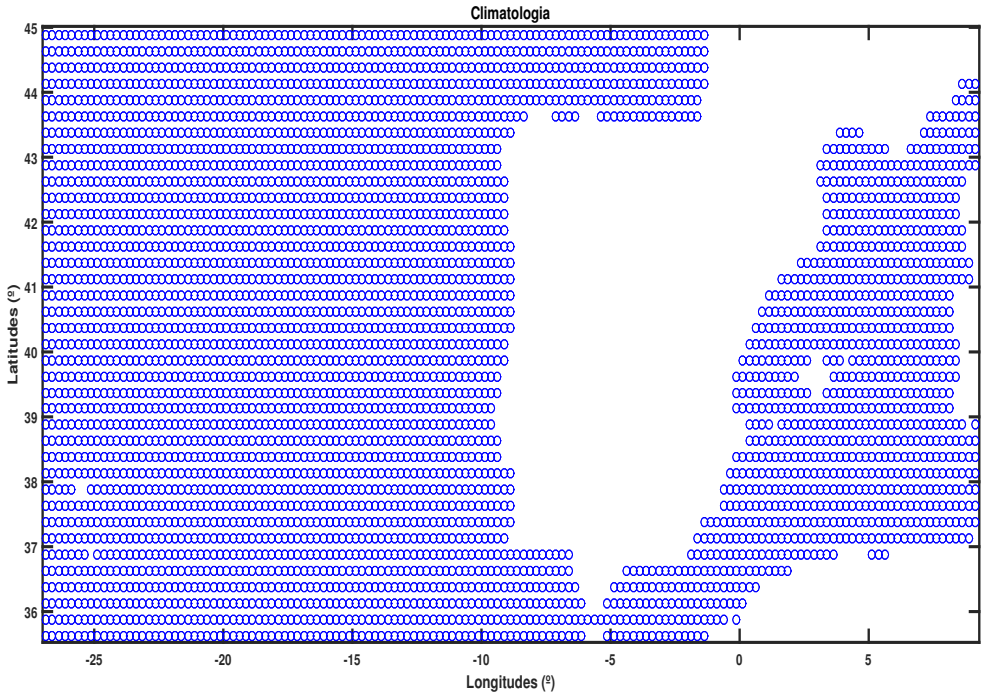


FIGURA 5.7: Mapa com a distribuição dos pontos com dados climatológicos da área envolvente a Portugal Continental.

5.2.2.1 Climatologia em 2D

Na determinação do campo de velocidades do som verificado no plano de propagação 2D, foram utilizados os dados climatológicos conhecidos com a resolução descrita anteriormente (resolução de um quarto de grau). Naturalmente, para a coerência do programa, para se estabelecer a climatologia a 2D ter-se-á de utilizar

[1]Termo utilizado para descrever a temperatura de uma amostra de água numa posição e profundidade específicas.

a linha geodésica estipulada para determinar a batimetria a 2D. Na Figura 5.8 está representado o mapa com as posições geográficas dos dados climatológicos da área envolvente a Portugal Continental e a trajetória assumida. Nesta figura é bem visível a necessidade de interpolação dos dados climatológicos ao longo da trajetória e, para tal, recorreu-se à interpolação pelo vizinho mais próximo (ver subcapítulo 3.6, ponto 3.6.2). Para a determinação dos vizinhos mais próximos entre os pontos com dados climatológicos e a trajetória utilizou-se um algoritmo do *MATLAB*® (denomidado por `knnsearch`) utilizando a distância Euclidiana. Representados a verde estão os pontos com dados climatológicos mais próximos dos pontos da trajetória. Deste modo, associou-se os dados climatológicos ao vizinho mais próximo da trajetória permitindo assim determinar a climatologia, em profundidade, ao longo da trajetória.

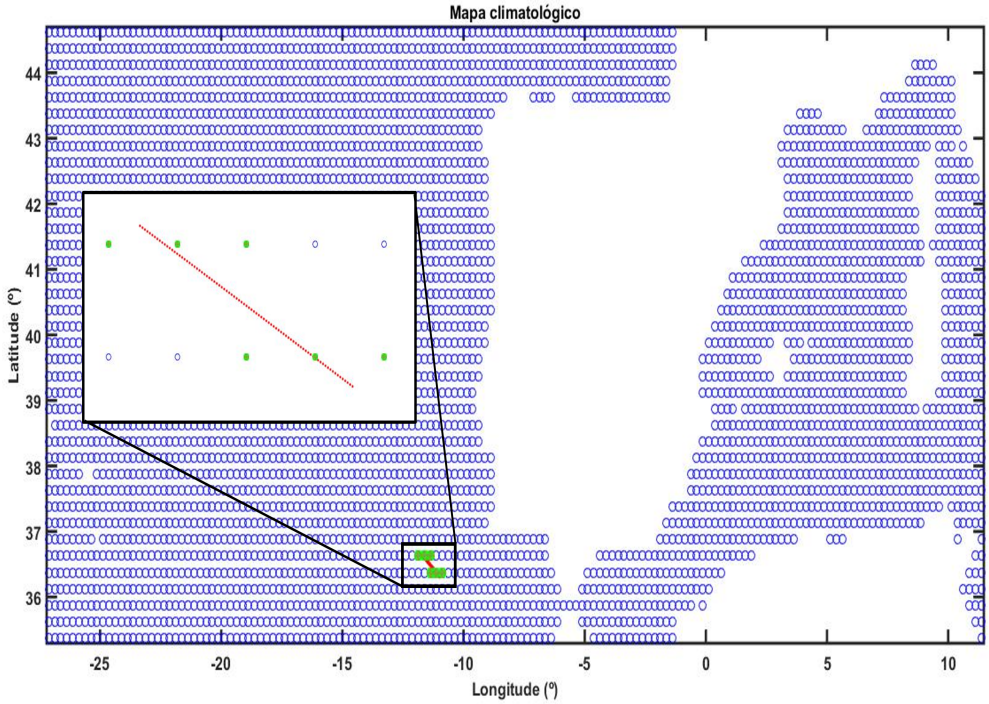


FIGURA 5.8: Mapa com a distribuição dos pontos com dados climatológicos da área envolvente a Portugal Continental com detalhe da representação da trajetória assumida.

5.2.2.2 Climatologia em 3D

Na determinação do campo de velocidade do som, necessário à realização de simulações de propagação de raios acústicos em 3D, foram utilizados os dados climatológicos disponibilizados na resolução indicada anteriormente. De forma idêntica ao `REOART 2D`, para a coerência do programa `REOART 3D` utilizou-se a mesma área de simulação do cálculo da batimetria para o cálculo da climatologia a 3D. Como é possível verificar-se na Figura 5.9, a área de simulação não coincide na totalidade com os pontos com dados climatológicos, tendo assim de se utilizar a abordagem

feita no programa `REOART 2D` — atribuir os dados climatológicos dos pontos mais próximos à área de simulação. Nessa mesma figura, os vizinhos mais próximos estão representados a verde.

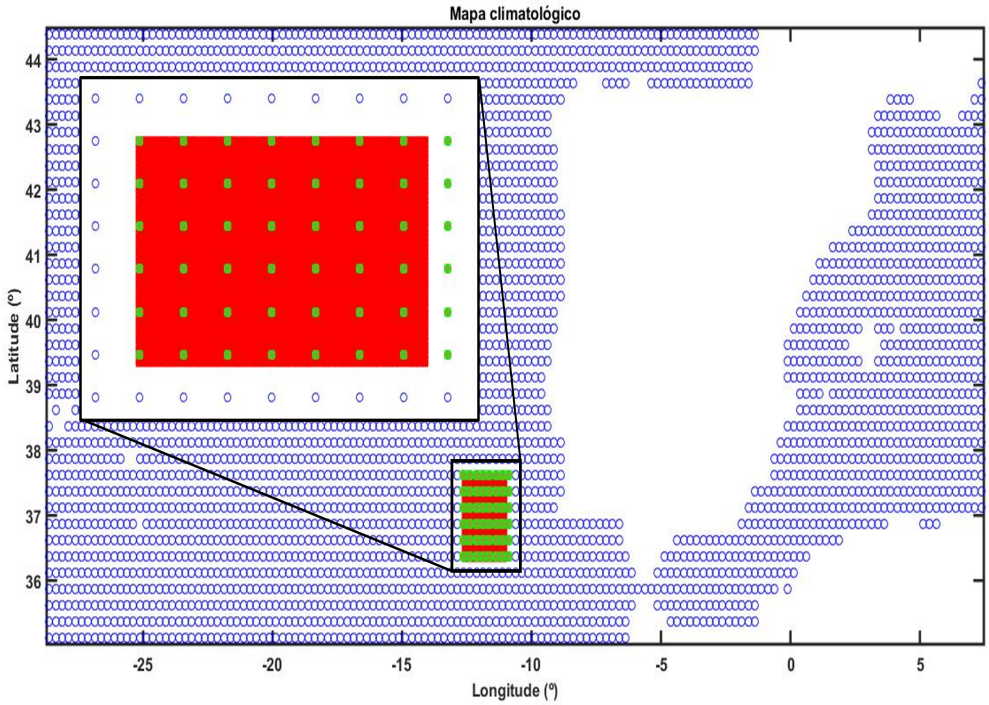


FIGURA 5.9: Mapa com a distribuição dos pontos com dados climatológicos da área envolvente a Portugal Continental com a representação da área de simulação 3D.

5.2.3 Funções TEOS-10

A TEOS-10 foi internacionalmente adotada com o intuito de substituir e colmatar, entre outros fatores, inconsistências nos polinómios da EOS-80 (*Equation of State of Seawater*). Anteriormente, com recurso à EOS-80, a densidade da água era calculada a partir da salinidade prática (S_P) e da temperatura *in situ* (t), e com a TEOS-10 foram introduzidas novas variáveis, como a salinidade absoluta[2] (S_A) e a temperatura conservativa[3] (Θ), que permitiu calcular a densidade da água, e outras propriedades, com maior exatidão.

A *Gibbs-SeaWater (GSW) Oceanographic Toolbox* (abreviadamente GSW *Toolbox*) contém as sub-rotinas da TEOS-10 para avaliar as propriedades termodinâmicas da água do mar. São várias as funções disponibilizadas na biblioteca deste

[2]A sanilidade absoluta é definida como a fração de massa de material dissolvido na água do mar, em g/kg. Distingue com precisão o que constitui a água e o que constitui material dissolvido. Os valores de salinidade absoluta diferem ligeiramente dos de salinidade prática (Feistel et al., 2009).

[3]A temperatura conservativa é a entalpia potencial dividida pelo calor específico da água na superfície do mar a pressão constante $C_p^0 \equiv 3991,86795711963$ J kg^{-1} K^{-1}. A diferença entre a temperatura potencial e a temperatura conservativa pode chegar a $-1,4°C$ (Feistel et al., 2009).

software — consulte-se o apêndice N em (Feistel et al., 2009). Para os programas `REOART 2D` e `REOART 3D` foram apenas utilizadas quatro funções desta biblioteca, mais especificamente as funções `gsw_SA_from_SP`, `gsw_CT_from_t`, `gsw_p_from_z` e `gsw_sound_speed`, que serão apronfundadas mais adiante.

5.3 Velocidade do som

A velocidade do som na água é a variável oceanográfica mais importante para a propagação do som no oceano, dado que influencia todos os outros fenómenos acústicos. A temperatura da água, a salinidade e a pressão hidrostática, são três variáveis que influenciam diretamente a velocidade de som, ou seja, caso haja alterações nos mesmos, por exemplo o aumento de um deles, a velocidade do som aumentará (Etter, 1995).

No cálculo da velocidade do som foram utilizadas três variáveis: a salinidade absoluta (S_A), a temperatura conservativa (Θ) e a pressão (p). Como foi visto anteriormente, os dados climatológicos fornecidos pela base de dados da WOD18 são a salinidade prática e temperatura *in situ*, portanto é preciso efetuar a conversão dos dados para salinidade absoluta e temperatura conversativa, respetivamente, para se poder utilizar as funções TEOS-10. Para isto, utilizaram-se as funções `gsw_SA_from_SP` e `gsw_CT_from_t`, que convertem as salinidades práticas em absolutas e as temperaturas *in situ* em conservativas. Para o cálculo da pressão hidrostática (p) utilizou-se a função `gsw_p_from_z`, onde os valores de entrada são as imersões utlizadas pela base de dados WOD18 (de 5 em 5 metros até aos 100 metros, e de 25 em 25 metros até aos 5.000 metros). Com as três variáveis calculadas, foi então possível utilizar a função `gsw_sound_speed` para calcular a velocidade do som nos pontos duma malha com a resolução espacial (vertical e horizontal) referida anteriormente.

Nos modelos de propagação acústica 2D e 3D por nós utilizados, para além do campo de velocidade do som é necessário conhecer os seus gradientes espaciais para se poder calcular a propagação do raio acústico. Assim sendo, para o cálculo dos gradientes referidos utilizaram-se técnicas de derivação numérica que apresentaremos seguidamente.

5.4 Cálculo dos gradientes espaciais

No cálculo numérico dos gradientes espaciais da velocidade de som utilizou-se as expressões deduzidas em Chapra (Chapra & Canale, 2011) de diferenças divididas centrais, regressivas e progressivas, de primeira e segunda ordem, que seguidamente, respetivamente, se apresentam:

$$f'(x_i) = \frac{-f(x_{i+2}) + 8f(x_{i+1}) - 8f(x_{i-1}) + f(x_{i-2})}{12h}, \tag{5.1}$$

$$f''(x_i) = \frac{-f(x_{i+2}) + 16f(x_{i+1}) - 30f(x_i) + 16f(x_{i-1}) - f(x_{i-2})}{12h^2}, \tag{5.2}$$

$$f'(x_i) = \frac{3f(x_i) - 4f(x_{i-1}) + f(x_{i-2})}{2h}, \tag{5.3}$$

$$f''(x_i) = \frac{2f(x_i) - 5f(x_{i-1}) + 4f(x_{i-2}) - f(x_{i-3})}{h^2}, \tag{5.4}$$

$$f'(x_i) = \frac{-f(x_{i+2}) + 4f(x_{i+1}) - 3f(x_i)}{2h}, \tag{5.5}$$

$$f''(x_i) = \frac{-f(x_{i+3}) + 4f(x_{i+2}) - 5f(x_{i+1}) + 2f(x_i)}{h^2}. \tag{5.6}$$

A utilização das expressões referidas permitiu calcular os valores aproximados dos diferentes gradientes de velocidade de som nos pontos da malha em que a velocidade do som foi determinada. Começando com o `REOART 2D`, os gradientes espaciais, cujo cálculo numérico foi efetuado da forma indicada, são os seguintes: $\frac{\partial c}{\partial x}$, $\frac{\partial c}{\partial y}$, $\frac{\partial^2 c}{\partial x^2}$, $\frac{\partial^2 c}{\partial y^2}$, $\frac{\partial c}{\partial y}(\frac{\partial c}{\partial x})$ e $\frac{\partial c}{\partial x}(\frac{\partial c}{\partial y})$, em que c representa a velocidade do som como função das coordenadas espaciais. Naturalmente, o `REOART 3D` calcula os gradientes espaciais de forma idêntica à mencionada, salientando o facto de ter mais número de gradientes espaciais devido às diferentes dimensões, sendo estes: $\frac{\partial c}{\partial x}$, $\frac{\partial c}{\partial y}$, $\frac{\partial c}{\partial z}$, $\frac{\partial^2 c}{\partial x^2}$, $\frac{\partial c}{\partial y^2}$, $\frac{\partial^2 c}{\partial z^2}$, $\frac{\partial c}{\partial y}(\frac{\partial c}{\partial x})$, $\frac{\partial c}{\partial z}(\frac{\partial c}{\partial x})$, $\frac{\partial c}{\partial x}(\frac{\partial c}{\partial y})$, $\frac{\partial c}{\partial z}(\frac{\partial c}{\partial y})$, $\frac{\partial c}{\partial x}(\frac{\partial c}{\partial z})$ e $\frac{\partial c}{\partial y}(\frac{\partial c}{\partial z})$, em que c, nesta situação, representa a velocidade do som como função de três coordenadas espaciais.

Na periferia da malha foram utilizadas as fórmulas de diferenças regressivas ou progressivas. Já nos pontos interiores da malha, foram utilizadas as fórmulas de diferenças centrais, que são mais exatas. De destacar que o passo de derivação utilizado foi o correspondente à resolução espacial da malha nas direções correspondentes.

Desta forma, foi possível estabelecer malhas espaciais com as informações da velocidade do som e respetivos gradientes; parâmetros estes conhecidos com a resolução espacial vertical de 5 em 5 metros até aos 100 metros, e de 25 em 25 metros até aos 5.000 metros, e resolução espacial horizontal de 0,5 milhas.

O conhecimento das informações referidas, velocidade do som e gradientes espaciais, permite obter, por interpolação multilinear, o conhecimento destes parâmetros em coordenadas espaciais arbitrárias da região descrita pela malha espacial em causa.

Nos programas `REOART 2D` e `REOART 3D`, utilizam-se as funções `Interp2` e `Interp3` do *MATLAB*®, respetivamente, para interpolar a malha espacial com os parâmetros em cima referidos à posição geográfica (latitude, longitude e profundidade) do raio acústico em propagação.

5.5 Reflexão do raio acústico nas fronteiras

Recapitulando do subcapítulo 4.2.2, a determinação do versor do raio acústico refletido **r** efetua-se tendo por base o versor do raio acústico incidente **i** no ponto de incidência e o versor do vetor normal $\overrightarrow{n}$ ao plano no ponto de incidência.

O cálculo do versor normal ao plano tangente à superfície no ponto de incidência do raio acústico está ilustrado na Figura 5.10.

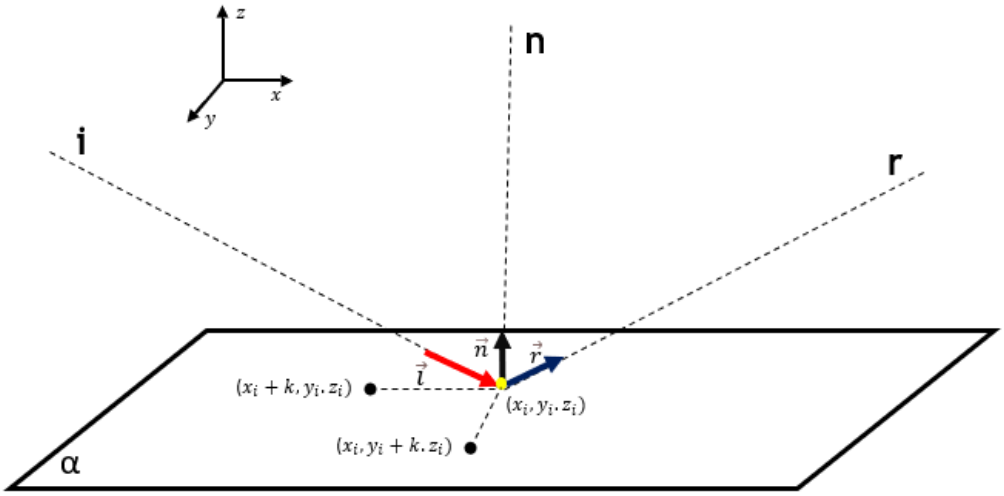


FIGURA 5.10: Cálculo da normal no plano α no ponto de incidência através de três pontos não colineares.

Representado por (x_i, y_i, z_i) estão as coordenadas do ponto de incidência e por $(x_i + k, y_i, z_i)$ e $(x_i, y_i + k, z_i)$ as coordenadas batimétricas da superfície nas posições incrementadas da distância k nas direções horizontais. O incremento da distância k garante a criação de vetores perpendiculares, com pontos não colineares. O produto externo entre os dois segmentos orientados definidos pelos pontos atrás referidos permite gerar um versor normal ao plano tangente à superfície no ponto de incidência.

A obtenção do versor do raio refletido r resulta do estabelecimento da matriz $\mathbf{R}(\mathbf{n})$ e do cálculo $\mathbf{R}(\mathbf{n})\,\mathbf{i} = \mathbf{r}$.

Capítulo 6

Simulações numéricas e discussão de resultados

Por forma a testar os programas `REOART 2D` e `REOART 3D`, neste capítulo serão apresentadas três de simulações numéricas, em diferentes posições geográficas, utilizando ambos os programas. Na primeira simulação, usou-se o programa `REOART 2D` e foi realizada na posição geográfica de latitude $36,31°$N e longitude $11°$ O com apenas um raio acústico — note-se que é a mesma posição geográfica utilizada no Capítulo 5 para gerar o plano de simulação 2D. Na segunda simulação, utilizou-se também o programa `REOART 2D` e foi realizada na posição geográfica de latitude $39°$N e longitude $10,1°$ O. Neste caso, a simulação numérica foi efetuada com vários raios acústicos. Na última simulação, foi utilizado o programa `REOART 3D`, na posição geográfica de latitude $41°$N e longitude $8,8°$ O, com um raio acústico.

6.1 `REOART 2D` — simulação numérica com um raio acústico

Como visto no capítulo anterior, a introdução da posição geográfica, do azimute, da distância de propagação e da distância entre posições geográficas, permite determinar as características do plano de propagação (a batimetria e a velocidade do som). Esta simulação numérica foi realizada na posição geográfica de latitude $36,31°$N e longitude $11°$ O, no azimute $300°$, com um raio acústico, com um ângulo de $0°$.

Esta posição geográfica foi utilizada no capítulo 5 para exemplificar como o programa `REOART 2D` determina a área de simulação em ambiente real. Desta forma, na Figura 5.3 está representada a batimetria da área envolvente a Portugal Continental e o plano de simulação 2D (a trajetória assumida), e na Figura 5.4 está representado o perfil batimétrico da trajetória. Por fim, na Figura 5.8, está representada a trajetória assumida com a distribuição dos pontos com dados climatológicos da área envolvente a Portugal Continental.

Com a área de simulação bem determinada e calculada, foi efetuada uma simulação numérica com um único raio acústico projetado na imersão de 2000 m, com um ângulo de 0°, como é possível ver na Figura 6.1. Nesta figura, é possível ver o raio acústico a refratar-se e a refletir-se na superfície e no fundo (representado pela linha a azul).

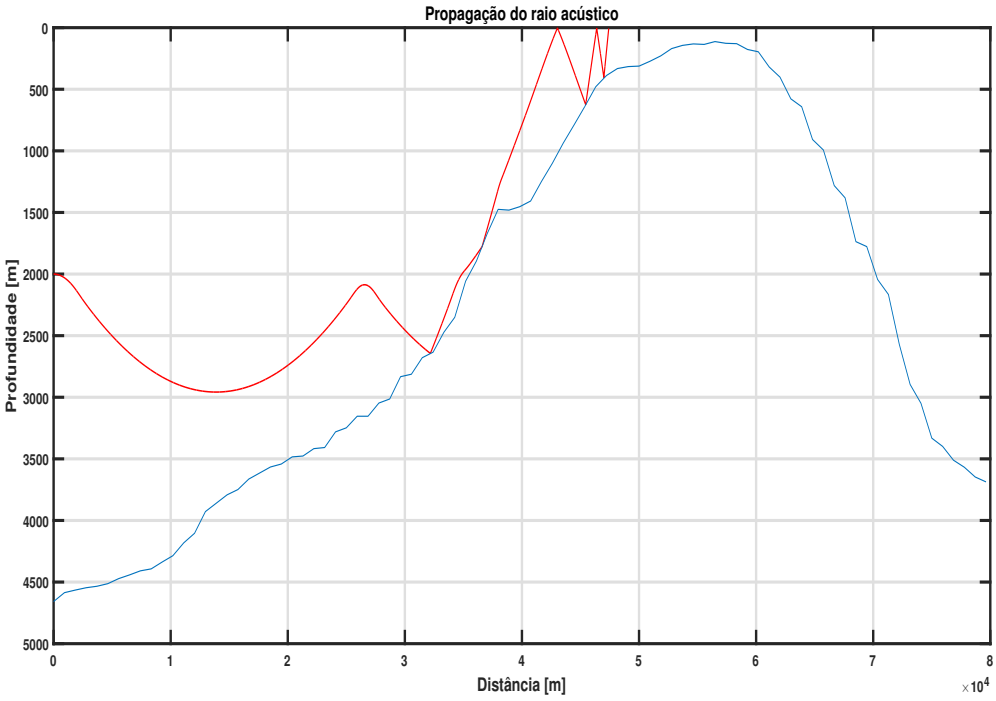


FIGURA 6.1: Simulação numérica na posição geográfica de latitude 36, 31°N e longitude 11° O, no azimute 300°, com um raio acústico projetado na imersão de 2000 m, com um ângulo de 0°.

Como é sabido, de acordo com a Lei de Snell, o som refrata-se para regiões com menor velocidade de propagação. Na Figura 6.2, além de estar representada a trajetória do raio e a batimetria, está também caracterizada a velocidade de som, num esquema de cores, ao longo do plano de propagação 2D. Nesse esquema, as cores azuis simbolizam as regiões de menor velocidade e a verde/amarelo as regiões de maior velocidade de propagação. Nesta figura, é bem visível o raio a refratar-se para as regiões de menor velocidade de propagação — deflete para as regiões de cor azul — satisfazendo qualitativamente a lei de Snell.

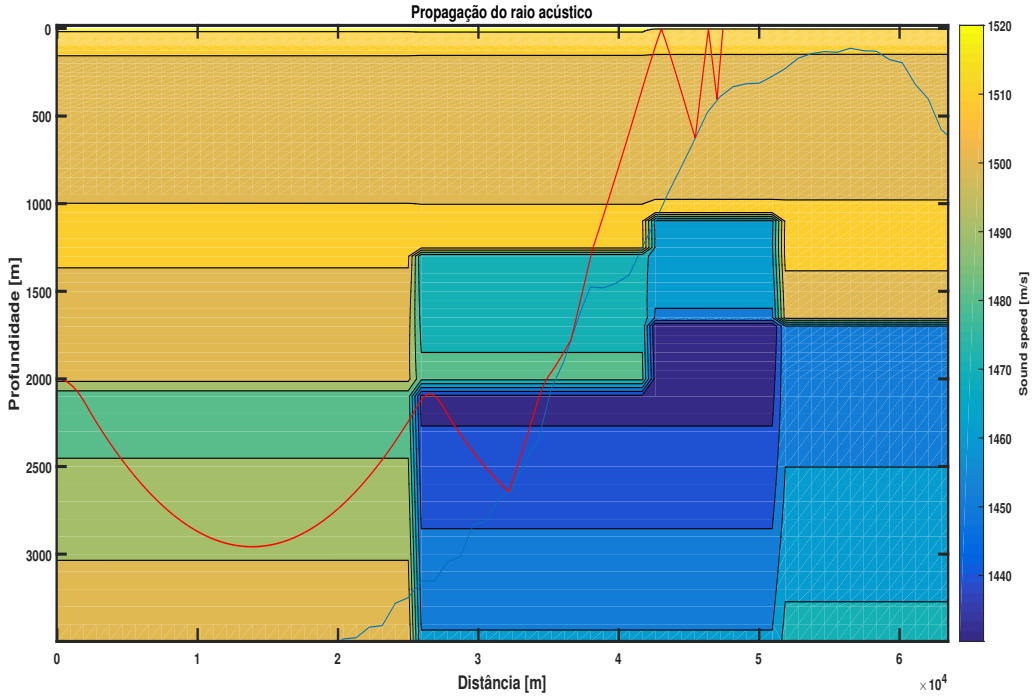


FIGURA 6.2: Simulação numérica na posição geográfica de latitude 36, 31°N e longitude 11° O, no azimute 300°, com um raio acústico projetado na imersão de 2000 m, com um ângulo de 0°, com a representação da velocidade de som por esquema de cores.

6.2 REOART 2D — simulação numérica com múltiplos raios acústicos

A simulação numérica efetuada com múltiplos raios acústicos, através do programa REOART 2D, foi realizada na posição geográfica de latitude 39°N e longitude 10, 1° O, no azimute 270°, projetados na imersão de 1 m. Os ângulos dos raios variam entre os 0° e os 12°.

Nas Figuras 6.3 e 6.4, está representado o mapa batimétrico da área envolvente a Portugal Continental, a linha geodésica criada a partir das condições iniciais introduzidas e o perfil do fundo do plano de simulação.

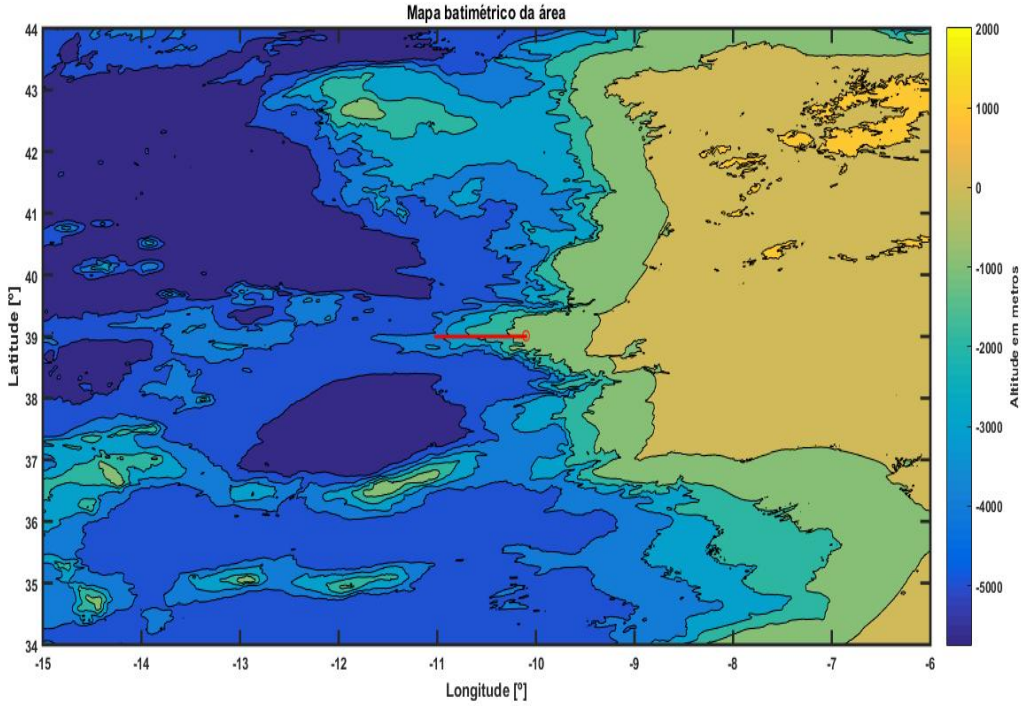


FIGURA 6.3: Mapa batimétrico da área envolvente a Portugal Continental com a representação da trajetória com início na posição geográfica de latitude 39°N e longitude $10,1°$ O, ao longo do azimute 270°.

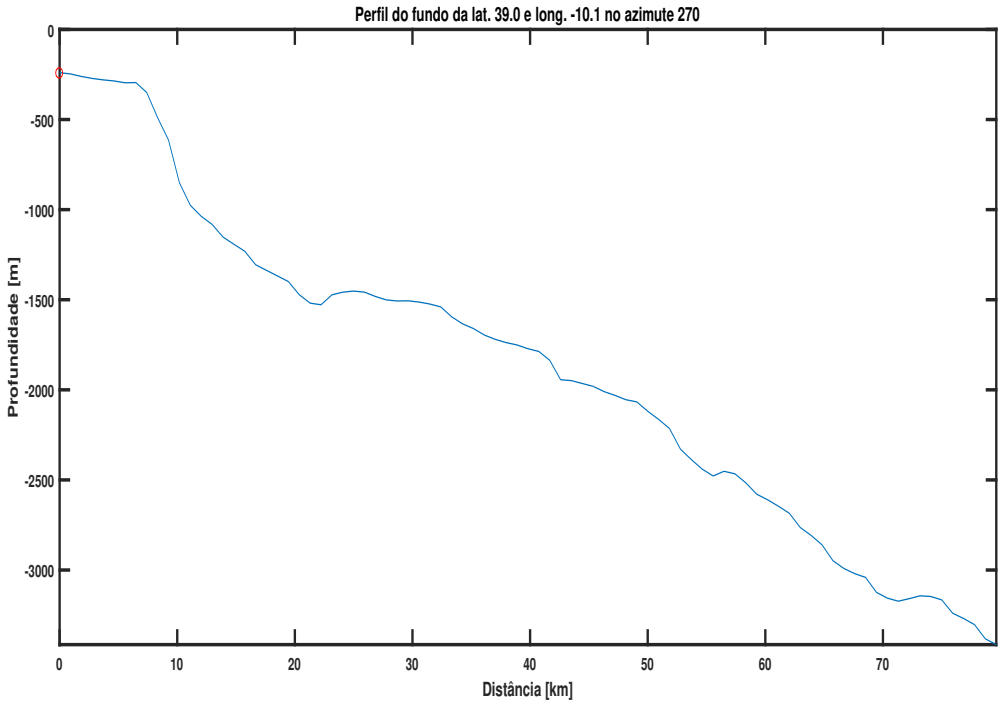


FIGURA 6.4: Perfil batimétrico da posição geográfica de latitude 39°N e longitude $10,1°$ O, ao longo do azimute 270°, até à distância de 80 quilómetros.

Na Figura 6.5 está retratada a interpolação necessária a realizar para atribuir os dados climatológicos ao plano de simulação.

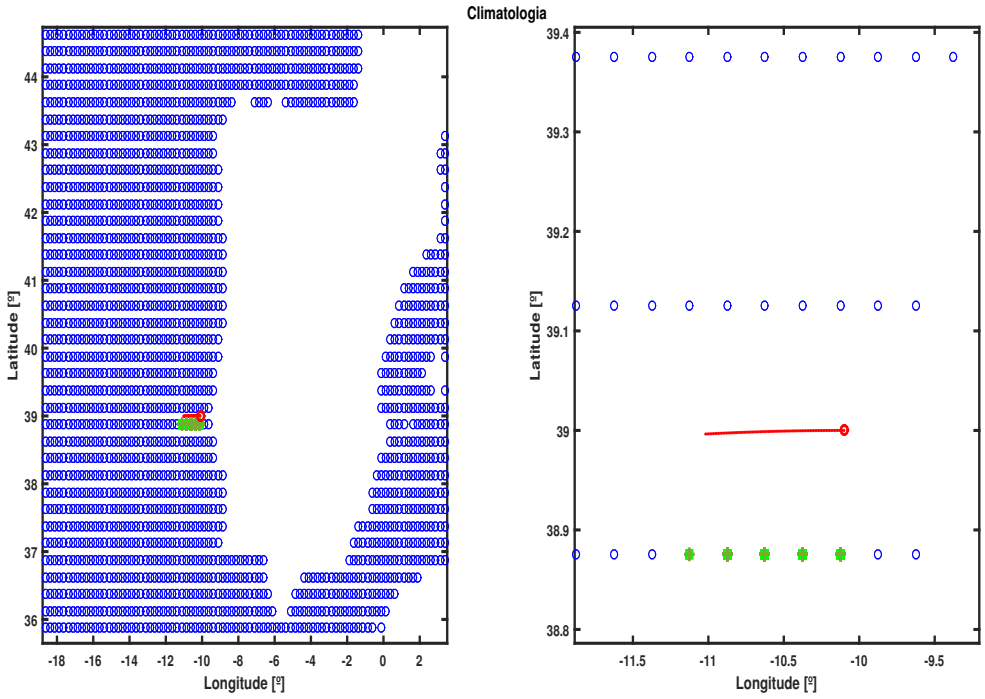


FIGURA 6.5: Mapa com a distribuição dos pontos com dados climatológicos da área envolvente a Portugal Continental com detalhe da representação da trajetória com início na posição geográfica de latitude 39°N e longitude 10, 1° O, ao longo do azimute 270°.

Posteriormente, com o uso das funções TEOS-10, é estabelecida a velocidade do som para a malha do plano de simulação, possibilitando o cálculo da propagação dos raios acústicos, como demonstrado na Figura 6.6. Nesta figura, é possível observar os diferentes raios acústicos a refletirem-se nas fronteiras e a refratarem-se ao longo da trajetória assumida. É de realçar o facto dos raios acústicos não alcançarem a mesma distância — isto acontence porque ambos os raios têm um tempo limitado de propagação e, consoante o trajeto que assumem, propagam-se para diferentes distâncias.

6.3 REOART 3D — simulação numérica com um raio acústico

A última simulação, realizada em 3D, utilizando o programa REOART 3D, foi efetuada na posição geográfica de latitude 41°N e longitude 8, 8° O, com um raio acústico projetado na imersão de 70 m. O raio acústico foi projetado com um ângulo de 60°.

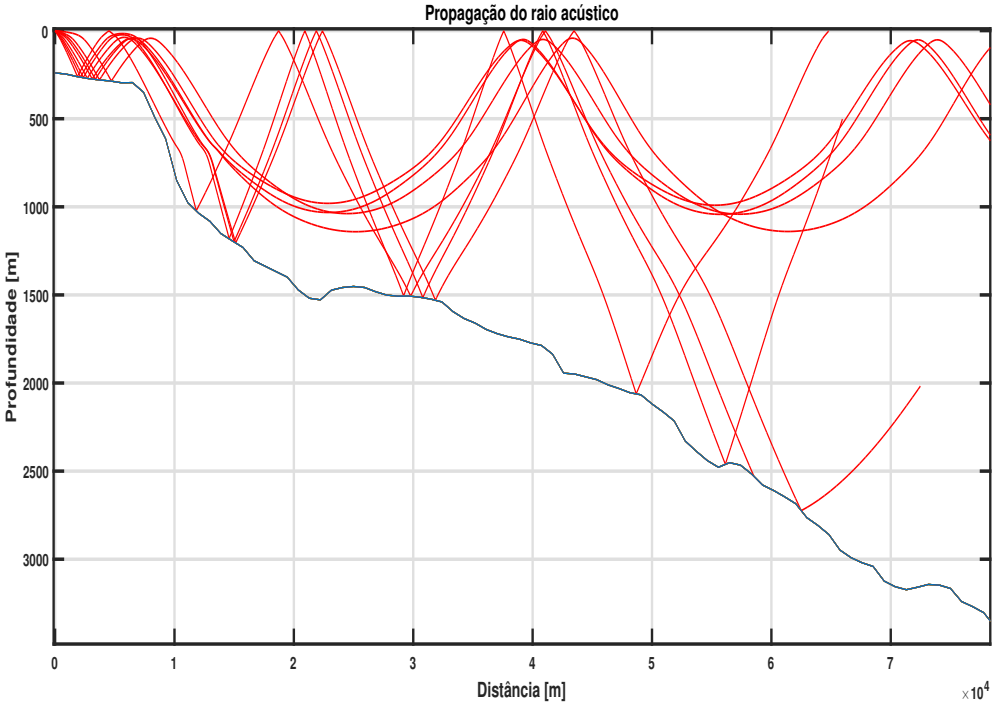


FIGURA 6.6: Simulação numérica na posição geográfica de latitude 39°N e longitude 10, 1° O, ao longo do azimute 270°, com múltiplos raios acústicos projetados na imersão de 1 m.

Na Figura 6.7, está retratado o mapa batimétrico da área envolvente a Portugal Continental e a área de simulação 3D (o quadrado vermelho). Note-se que o vértice da área de simulação 3D a sudeste corresponde à posição geográfica introduzida.

De forma semelhante ao demonstrado anteriormente, na Figura 6.8 está retratada a interpolação necessária a realizar para atribuir os dados climatológicos à área de simulação 3D.

Seguidamente, com a velocidade de som e gradientes espaciais calculados através das funções TEOS-10, para a malha da área de simulação 3D, é então realizável o cálculo da propagação do raio acústico, como ilustrado nas Figuras 6.9 e 6.10.

Nas simulações efetuadas com o programa `REOART 3D`, o tempo de cálculo para gerar a simulação 3D demonstrou-se substancialmente moroso, por esse motivo optou-se por ilustrar apenas um trajeto compatível com o tempo de cálculo disponível.

A presente simulação, nesta fase de desenvolvimento do programa `REOART 3D`, descreve apenas o seu conceito e sugere a necessidade de se optimizar os algoritmos do programa.

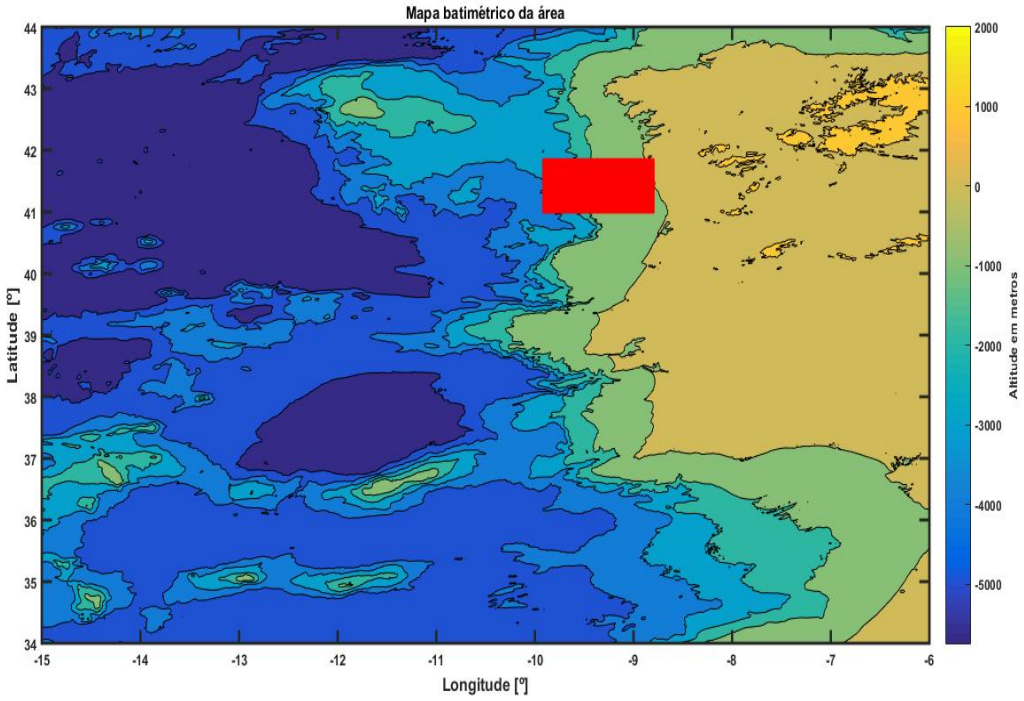


FIGURA 6.7: Mapa batimétrico da área envolvente a Portugal Continental com a área de simulação 3D (assinalada a cor vermelha).

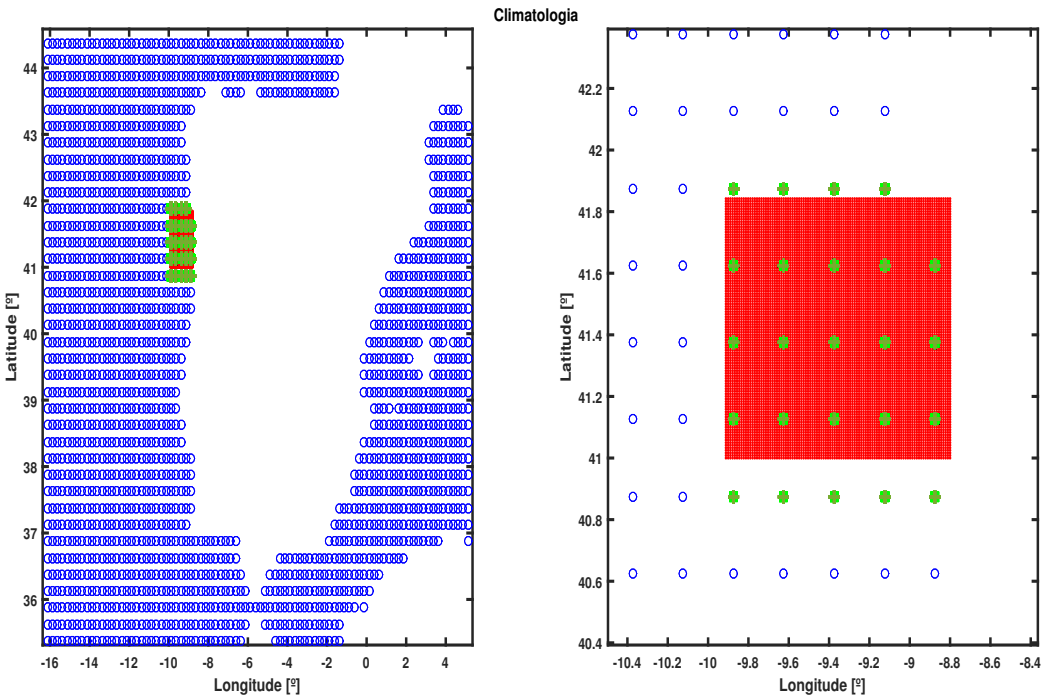


FIGURA 6.8: Mapa com a distribuição dos pontos com dados climatológicos da área envolvente a Portugal Continental com a representação da área de simulação 3D.

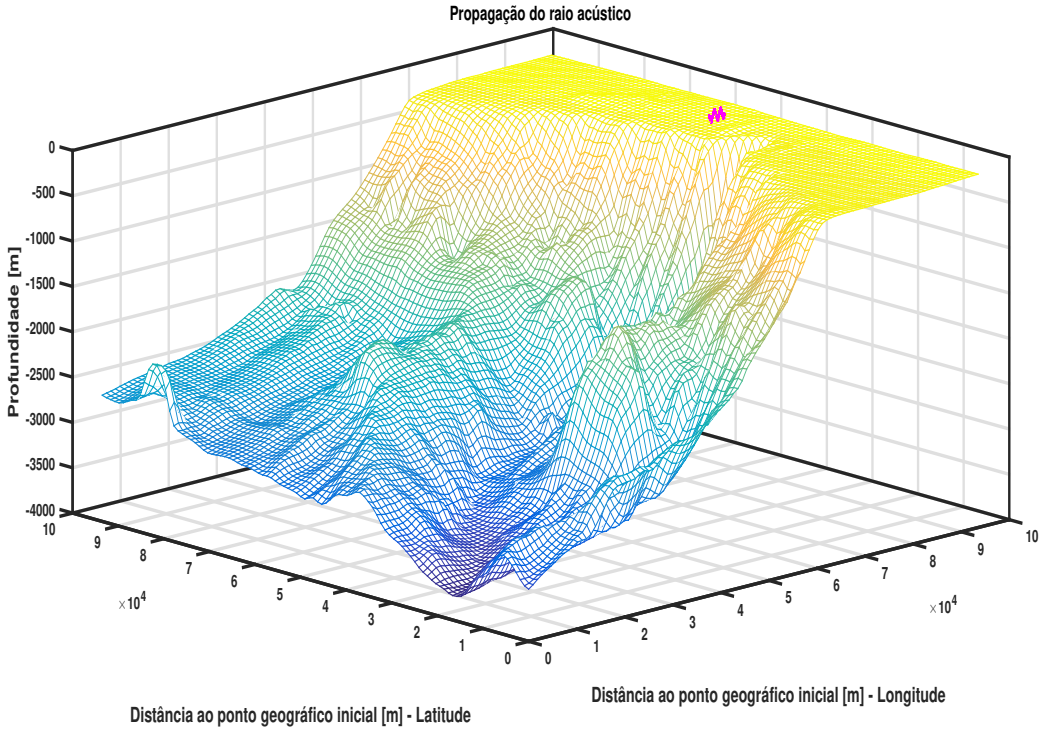


FIGURA 6.9: Representação de uma simulação com um raio acústico, utilizando o programa `REOART 3D`, projetado na imersão de 70 m.

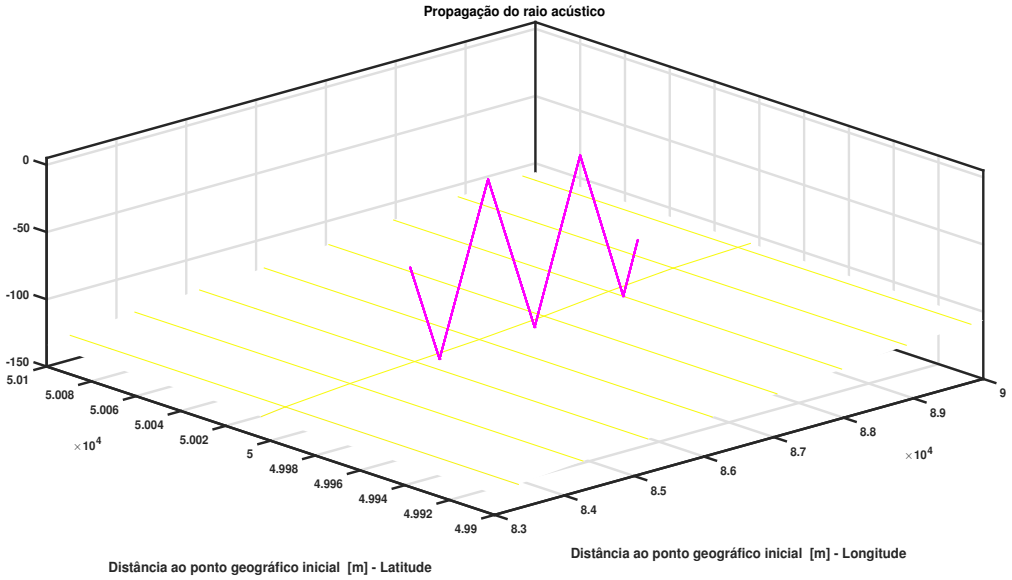


FIGURA 6.10: Representação de uma simulação com um raio acústico, utilizando o programa `REOART 3D`, projetado na imersão de 70 m.

Capítulo 7

Conclusões e trabalho futuro

7.1 Conclusões

O objetivo princinpal deste trabalho foi criar duas ferramentas de propagação acústica em ambiente real com a capacidade de modelar especularmente as reflexões dos raios acústicos. Após o desenvolvimento e simulações dos programas `REOART 2D` e `REOART 3D` conseguiu-se constatar o seguinte:

- Debruçando primeiramente sobre o programa `REOART 2D`, após as simulações numéricas o programa mostrou-se bastante credível e robusto. Os modelos de propagação utilizados foram anteriormente validados e as bases de dados utilizadas são fiáveis, dado que foram elaboradas por instituições reconhecidas pela comunidade científica. Este programa revelou-se adequado para estudar e prever a propagação do som na água com maior exatidão, visto que modela os raios em ambiente real. Contudo, existem três melhorias a fazer ao mesmo para que se torne uma ferramenta pronta para o uso operacional. A primeira melhoria baseia-se no melhoramento dos algoritmos de cálculo da propagação acústica; embora o tempo de cálculo seja aceitável, é possível reduzir o mesmo para que faça a previsão de propagação dos raios de uma forma célere. A segunda melhoria será a criação de uma interface gráfica mais amigável para o uso do programa. A terceira melhoria, e talvez a mais relevante, será dotar o programa com a capacidade de calcular perdas de transmissão e absorção acústica, que são fenómonos muito importantes para o alcance sonar.

- Relativamente ao programa `REOART 3D`, embora o modelo de propagação dos raios acústicos não tenha sido validado, mostrou-se fiável no trabalho de Pedro Raposo (Raposo, 2019). Este programa é adequado para o estudo da previsão acústica em 3D, no entanto o tempo de cálculo para gerar a simulação é muito demorado e limita o programa para calcular apenas trajetos que sejam compatíveis com o tempo de cálculo disponível.

- A modelação das reflexões especulares, através do cálculo matricial de versores, desenvolvidas e utilizadas nos programas, mostraram-se bastante robustas, válidas e adequadas para a modelação do fenómeno de reflexão dos raios acústicos nas fronteiras.

O `REOART 2D` e o `REOART 3D` têm a particularidade de calcular a velocidade do som à medida que o raio acústico se propaga no meio, tornando estes programas inovadores neste aspeto. Por esta razão, não foi possível comparar o `REOART 2D` e o `REOART 3D` com outros programas existentes de previsão acústica, visto que os resultados não seriam fiáveis porque a velocidade de propagação do raio não seria igual.

7.2 Trabalho futuro

Como trabalho futuro, preve-se para os programas `REOART 2D` e `REOART 3D` o seguinte:

- Otimização dos algoritmos dos programas por forma a tornar o tempo de cálculo mais expedito;
- Criação de interfaces gráficas para facilitar a utilização dos programas;
- Dotar ambos os programas com a capacidade de calcular perdas de transmissão e absorção acústica;
- Efetuar a validação dos programas com outros eventuais programas de previsão acústica em ambiente real.

Bibliografia

Abrahamsson, L. (2003). *RAYLAB: A Ray Tracing Program in Underwater Acoustics ; Scientific Report*. Systems Technology, Swedish Defence Research Agency (Systemteknik, Totalförsvarets forskningsinstitut (FOI)).

Apostol, T. M. (1969). Calculus, volume II: multi-variable calculus and linear algebra, with applications to differential equations and probability.

Ascher, U. M. & Petzold, L. R. (1998). *Computer methods for ordinary differential equations and differential-algebraic equations* (Vol. 61). Siam.

Bankman, I. N. (2009). *Handbook of Medical Image Processing and Analysis* (Second Edition). Academic Press.

Born, M. & Wolf, E. (1964). *Principles of optics: electromagnetic theory of propagation, interference and diffraction of light; 2nd ed.* Pergamon.

Brekhovskikh, L. (1983). Fundamentals of Ocean Acoustics.

Cai, Y., Fan, W., Zhou, Y., qiao Fu, X., Fang, H. & Jin, T. (2012). Study on the high precision acoustic measurement techniques for determining temperature field around sea‡oor hydrothermal vent. *China Ocean Engineering*, 26(4):723–732.

Chapra, S. C. & Canale, R. P. (2011). *Numerical methods for engineers* (Vol. 1221). Mcgraw-hill New York.

Edwards, C. H., Penney, D. E. & Calvis, D. T. (2016). *Differential equations and boundary value problems*. Pearson Education Limited.

Etter, P. C. (1995). *Underwater acoustic modeling: principles, techniques and applications*. CRC Press.

Ewing, C. E. & Mitchell, M. M. (1970). Introduction to geodesy. *New York*.

Feistel, TJ, R., McDougall, Millero, F., Jackett, D., Wright, D., King, B., Marion, G., Chen, C., Spitzer, P. & Seitz, S. (2009). The international thermodynamic equation of seawater 2010 (teos-10): Calculation and use of thermodynamic properties. *Global Ship-based Repeat Hydrography Manual, IOCCP Report No*, *14*.

Fernandes, J. B. (1967). *Manual de Hidrografia*. Instituto Hidrográfico.

Gatta, M. C. (s.d.). *Notas antigas de deduções acústicas*.

Helliwell, T. M. & Sahakian, V. (2020). *Modern Classical Mechanics*. Cambridge University Press.

Hovem, J. M. (2013). Ray trace modeling of underwater sound propagation. *Modeling and measurement methods for acoustic waves and for acoustic microdevices*. IntechOpen.

Jekeli, C. (2006). Geometric reference systems in geodesy. *Division of Geodesy and Geospatial Science, School of Earth Sciences, Ohio State University*, *25*.

Jones, R. M., Riley, J. P. & Georges, T. M. (1986). HARPO: A versatile three-dimensional Hamiltonian ray-tracing program for acoustic waves in an ocean with irregular bottom. *Unknown*.

Keller, J. B. & Papadakis, J. S. (1977). *Wave propagation and underwater acoustics* (Vol. 1). Springer Berlin, Heidelberg.

Kinsler, L. E. (2000). *Fundamentals of acoustics*. Wiley.

Lanczos, C. (1970). *The variational principles of mechanics*. Dover Pubns.

Lemons, D. S. (1997). *Perfect Form : Variational Principles, Methods, and Applications in Elementary Physics*. Princeton University Press.

Lu, J., Wakai, K., Takahashi, S. & Shimizu, S. (2000). Acoustic computer tomographic pyrometry for two-dimensional measurement of gases taking into account the effect of refraction of sound wave paths. *Measurement Science and Technology*, 11(6):692–697.

Marques, P. A. A. V. (2018). *Acústica submarina: Desenvolvimento de modelo de previsão de propagação acústica global.* (tese de mestrado). Escola Naval.

Nagle, R. K., Saff, E. B. & Snider, A. D. (2014). *Fundamentals of differential equations*. Pearson Education.

Paiva, B. M. E. (2018). *Caracterização de fontes hidrotermais oceânicas com recurso a técnicas acústicas* (tese de mestrado). Escola Naval.

Papadopoulos, A. N. & Manolopoulos, Y. (2006). *Nearest Neighbor Search:: A Database Perspective*. Springer Science & Business Media.

Pina, H. L. G. (2010). Análise Númerica.

Porter, M. B. (2011). The bellhop manual and user's guide: Preliminary draft. *Heat, Light, and Sound Research, Inc., La Jolla, CA, USA, Tech. Rep*, *260*.

Raposo, P. A. M. A. (2019). *Estudo da propagação acústica 3D em meios não homogéneos – Raytracing baseado no príncipio de Fermat* (tese de mestrado). Escola Naval.

Reilly, S. M., Potty, G., Thibaudeau, D. & Goodrich, M. (2014). WaveQ3D: Fast and accurate acoustic transmission loss (TL) eigenrays, in littoral environments, for active sonar simulation/stimulation systems.

Ricardo, H. J. (2020). *A Modern Introduction to Differential Equations*. Elsevier Science Publishing Co Inc.

Roberts, A. J. (1998). *Differential algebraic equations solvers* (rel. téc.). The University of Adelaide.

Rodriguez, O., Jesus, S. & Calazan, R. M. (2021). *Fundamentos de acústica submarina*. Universidade do Algarve.

Rodrıguez, O. C. (2010). The TRACEO ray tracing program. *Universidade do Algarve-Signal Processing Laboratory*.

Rogers, D. W. (2011). *Concise physical chemistry*. John Wiley & Sons.

Schmidt, V. (2009). Raytrace. MATLAB.

Simmons, G. F. (2016). *Differential equations with applications and historical notes*. CRC Press.

Simmons, G. F. & Krantz, S. G. (2007). *Differential equations: theory, technique and practice*. McGraw-Hill.

Tolstoy, A. (1996). 3-D propagation issues and models. *Journal of Computational Acoustics*, *4*(03), 243–271.

Torres, A. C. L. (2019). *Modelação da Acústica Submarina e Previsão Sonar - Desenvolvimento de uma Ferramenta de Modelação* (tese de mestrado). Escola Naval.

Urick, R. J. (1967). *Principle of underwater sound* (Vol. 3rd edition). Peninsula.

Wagner, R. (2008). Multi-linear interpolation. *Beach Cities Robotics*.

Apêndice A

Dedução da matriz R(n)

Na base da dedução das entradas da matriz $\mathbf{R}(\mathbf{n})$ estão as expressões (4.12), (4.13) e (4.14), que tornam os vetores unitários, e a propriedade

$$(\mathbf{r}+\mathbf{i})\,.\,\mathbf{n}=0. \tag{A.1}$$

As colunas da matriz $\mathbf{R}(\mathbf{n})$ serão as componentes, na base canónica, das componentes das imagens ($\mathbf{R}(\mathbf{n})\mathbf{e}_i$, $i=1,2$ e 3) de cada um dos vetores da base canónica. Consideremos $\mathbf{i}=\mathbf{e}_1$ para calcularmos $\mathbf{r}$. Primeiramente, vamos começar por fazer $\mathbf{r}=\alpha\mathbf{i}+\beta\mathbf{n}$, admitindo que $\mathbf{i}$ e $\mathbf{n}$ são linearmente independentes (caso $\mathbf{r}=-\mathbf{i}$ não se verifique). Como $\mathbf{i}=\mathbf{e}_1$, ter-se-á

$$\mathbf{r}=\alpha\mathbf{e}_1+\beta\mathbf{n}. \tag{A.2}$$

Atendendo à propriedade (A.1), deduzimos

$$(\alpha\mathbf{e}_1+\beta\mathbf{n}+\mathbf{e}_1)\,.\,\mathbf{n}=0 \implies \beta=-(1+\alpha)n_1. \tag{A.3}$$

De outro modo, $\|\mathbf{r}\|^2=1$ e $\|\mathbf{n}\|^2=1$, logo

$$\begin{aligned}
1 &= \|\mathbf{r}\|^2=\|\alpha\mathbf{e}_1+\beta\mathbf{n}\|^2\\
&= \|(\alpha+\beta n_1,\beta n_2,\beta n_3)\|^2\\
&= (\alpha+\beta n_1)^2+(\beta n_2)^2+(\beta n_3)^2\\
&= \alpha^2+2\alpha\beta n_1+(\beta n_1)^2+(\beta n_2)^2+(\beta n_3)^2\\
&= \alpha^2+2\alpha\beta n_1+\beta^2.
\end{aligned} \tag{A.4}$$

Ao substituir (A.3) em (A.4) obtemos

$$\begin{aligned}
1 &= \alpha^2+2\alpha\beta n_1+\beta^2\\
&= \alpha^2+2\alpha(-(1+\alpha)\,n_1)\,n_1+(-1(1+\alpha)\,n_1)^2\\
&= \alpha^2+n_1^2-\alpha_1^{22}\Rightarrow\\
(1-n_1^2)\,\alpha^2 &= (1-n_1^2)\Rightarrow\alpha=\pm1.
\end{aligned}$$

Como a solução $\alpha=-1$ não é satisfatória, tendo em conta (A.3), façamos então $\alpha=1$:

$$\alpha=1\Longrightarrow\beta=-2n_1. \tag{A.5}$$

Por conseguinte, concluímos que

$$\begin{aligned}
\mathbf{r} &= \alpha\mathbf{e}_1+\beta\mathbf{n}\\
&= \mathbf{e}_1-2\mathbf{n}_1\mathbf{n}\\
&= (1-2n_1^2,-2n_1n_2,-2n_1n_3).
\end{aligned}$$

Ou seja,

$$\mathbf{R}(\mathbf{n})\mathbf{e}_1=\begin{bmatrix}1-2n_1^2\\-2n_1n_2\\-2n_1n_3\end{bmatrix}.$$

O cálculo de $\mathbf{R}(\mathrm{n})\mathbf{e}_2$ e de $\mathbf{R}(\mathrm{n})\mathbf{e}_3$ é de forma semelhante. Sendo assim, obtemos

$$\mathbf{R}(\mathbf{n})\mathbf{e}_2 = \begin{bmatrix} -2n_1n_2 \\ 1-2n_2^2 \\ -2n_2n_3 \end{bmatrix},$$

$$\mathbf{R}(\mathbf{n})\mathbf{e}_3 = \begin{bmatrix} -2n_1n_3 \\ -2n_2n_3 \\ 1-2n_3^2 \end{bmatrix}.$$

Donde se deduz, por fim,

$$\mathbf{R}(\mathbf{n}) = \begin{bmatrix} 1-2n_1^2 & -2n_1n_2 & -2n_1n_3 \\ -2n_1n_2 & 1-2n_2^2 & -2n_2n_3 \\ -2n_1n_3 & -2n_2n_3 & 1-3n_1^2 \end{bmatrix} \tag{A.6}$$

Apêndice B

Tabela das técnicas de derivação numérica

Método numérico	Primeira derivada	Segunda derivada
Diferenças aproximadas centrais	$f'(x_i) = \frac{-f(x_{i+2})+8f(x_{i+1})-8f(x_{i-1})+f(x_{i-2})}{12h}$	$f''(x_i) = \frac{-f(x_{i+2})+16f(x_{i+1})-30f(x_i)+16f(x_{i-1})-f(x_{i-2})}{12h^2}$
Diferenças aproximadas para trás	$f'(x_i) = \frac{3f(x_i)-4f(x_{i-1})+f(x_{i-2})}{2h}$	$f''(x_i) = \frac{2f(x_i)-5f(x_{i-1})+4f(x_{i-2})-f(x_{i-3})}{h^2}$
Diferenças aproximadas para a frente	$f'(x_i) = \frac{-f(x_{i+2})+4f(x_{i+1})-3f(x_i)}{2h}$	$f''(x_i) = \frac{-f(x_{i+3})+4f(x_{i+2})-5f(x_{i+1})+2f(x_i)}{h^2}$

Apêndice C

Fórmulas de transporte de coordenadas geodésicas

De seguida serão listadas as fórmulas de transporte de coordenadas geodésicas, retiradas do Manual de Hidrografia (Fernandes, 1967), utilizadas no presente trabalho:

$$\Delta\varphi'' = -s \,.\, cos\ \alpha \,.\, B \ -\ s^2 \,.\, sen^2\alpha \,.\, C \ -\ (\delta\varphi'')^2 \,.\, D \ +\ h \,.\, s^2 \,.\, sen^2\alpha \,.\, E$$

$$log\ \Delta\ \lambda'' \ =\ log\ s \ +\ log\ sen\ \alpha \ +\ log\ A' \ +\ log\ sec\ \varphi' \ +\ correção\ (arco - seno)$$

$$\Delta\alpha'' \ =\ -\ \Delta\ \lambda'' \,.\, sen\left(\frac{\varphi\ +\ \varphi'}{2}\right) \,.\, sec\left(\frac{\Delta\,\varphi}{2}\right) \ -\ (\Delta\lambda'') \,.\, F$$

$$\delta\varphi \ =\ -\ s \,.\, cos\ \alpha \,.\, B \ -\ s^2 \,.\, sen^2\alpha \,.\, C$$

$$h \ =\ s \,.\, cos\ \alpha \,.\, B$$

$$A \ =\ \frac{1}{N\ sen\ (1'')}$$

$$B \ =\ \frac{1}{R\ sen\ (1'')}$$

$$C \ =\ \frac{tg\,\varphi}{2\ RN\ sen\ (1'')}$$

$$D \ =\ \frac{3\ e^2\ sen\ \varphi\ cos\ \varphi\ sen\ (1'')}{2\ (1\ -\ e^2\ sen^2\varphi)}$$

$$E \ =\ \frac{1\ +\ 3\ tg^2\varphi}{6\ N^2}$$

$$F \ =\ \frac{1}{12}\ sen\left(\frac{\varphi\ +\ \varphi'}{2}\right)\ cos^2\left(\frac{\varphi\ +\ \varphi'}{2}\right)\ sen^2\ (1'')$$

$$R \ =\ \frac{a\ (1\ -\ e^2}{(1\ -\ e^2\ sen^2\varphi_1)^{3/2}}$$

$$N \ =\ \frac{a}{(1\ -\ e^2\ sen^2\varphi_1)^{1/2}}$$

$$N' \ =\ \frac{a}{(1\ -\ e^2\ sen^2\varphi_2)^{1/2}}$$

$$A' \ =\ \frac{1}{N'\ sen\ (1'')}.$$

Apêndice D

Trabalho apresentado na conferência Internoise 2022



An ocean acoustical ray tracing tool based on Fermat's least time principle in a real environment

Gonçalves, João[1]
CINAV (Navy Research Center)/Escola Naval (Navy Academy)
2810-001 Almada, Portugal

ABSTRACT

Using an ocean-acoustic parametric ray tracing propagation models in 2D and 3D, based on Fermat's least time principle, two programs were developed with the ability to study sound propagation in a real ocean environment. `REOART 2D` *and* `REOART 3D` *use three different databases to withdraw information regarding the ocean topography and climatology to compute the acoustic ray trajectories in a real environment.*

KEYWORDS: Ray-tracing, Ocean-acoustic propagation, Simulation in a real environment

1. INTRODUCTION

Over the last four decades, underwater acoustics, and more precisely, ocean-acoustic propagation models, have been intensely researched in response to practical requirements [1]. As a result, numerous types of models have been developed, including ray tracing, parabolic equation (PE), and normal mode.

The development and application of techniques for forecasting sound propagation trajectories in the ocean is critical for military and societal purposes. It enables, for example, the military to pinpoint places with a low probability of detection. It enables, for instance, the development and application of acoustical technologies for imaging and measuring underwater structures and oceanic parameters in the civil field.

The development of underwater acoustic models permit the simulation of sonar performance under laboratory conditions. Therefore, these models are now used to predict sonar performance at sea and designing optimized sonar systems [2].

In this work, two ocean-acoustic propagation programs were developed — `REOART 2D` and `REOART 3D` (**R**eal **E**nvironment **O**cean-**A**coustic **R**ay **T**racing). These programs use an ocean-acoustic parametric ray-tracing propagation models in 2D and 3D, based on Fermat's least time principle. These models, constituted by differential algebraic system of equations (DAE's), are very robust and reliable — see [3] and [4]. Through these ray tracing propagation models and utilizing the GEBCO (*General Bathymetric Charts of the Ocean*), WOD18 (*World Ocean Database 2018*) and TEOS-10 (*Thermodynamic Equation of State 2010*) databases, the `REOART 2D` and `REOART 3D` are capable of studying and forecasting sound propagation in a real ocean environment.

[1]joaopdgoncalves@gmail.com

2. RAY TRACING PROPAGATION MODELS

We present below the acoustic propagation models used in this work deduced by Raposo [4] [3]. Here is the 2D model

$$\begin{cases} x' - a = 0 \\ y' - b = 0 \\ a'b - b'a + \frac{b}{v(x,y)}\frac{\partial v}{\partial x} - \frac{a}{v(x,y)}\frac{\partial v}{\partial y} = 0 \\ aa' + bb' = 0 \end{cases}, \tag{1}$$

and here is the 3D model,

$$\begin{cases} x' - a = 0 \\ y' - b = 0 \\ z' - c = 0 \\ (1-a')a' + (1-c^2)c' - ac(a'+c') - b(ab' + cb') \\ \quad + \frac{1}{v}\left(\frac{\partial v}{\partial x} + \frac{\partial v}{\partial z}\left(\frac{dv}{dx}a + \frac{dv}{dy}b + \frac{dv}{dz}c\right)(a+c)\right) = 0 \\ (1-b')b' + (1-c^2)c' - bc(b'+c') - a(ba' + ca') \\ \quad + \frac{1}{v}\left(\frac{\partial v}{\partial y} + \frac{\partial v}{\partial z}\left(\frac{dv}{dx}a + \frac{dv}{dy}b + \frac{dv}{dz}c\right)(b+c)\right) = 0 \\ aa' + bb' + cc' = 0 \end{cases}, \tag{2}$$

where x, y, z and v stand, respectively, for the ray positioning and the speed of sound.

The numerical resolution of the DAE's (1) and (2) presents difficulties that require the use of appropriate solvers [5], such as the algorithm developed in MATLAB by Tony Roberts [6], which it was used in this work.

3. 2D AND 3D FORMULATION IN REAL ENVIRONMENT

The majority of propagation models assume that sound propagates in a two-dimensional plane. That is not always true. This is not the case when sound speed presents spatial gradients perpendicular to the propagation plane. In this scenario, the propagation takes place in a three-dimensional environment. Additionally, when a ray reflects in the bottom of a 2D plane, it may exit the initial 2D plane [7]. For these reasons, a three-dimensional propagation model must be employed to adequately understand sound propagation.

Under the 2D acoustic propagation, hypothesis it is supposed that an acoustic ray propagates in the ocean in a vertical plane that intersects the surface geodesically. Therefore, the acoustic ray's azimuth along the plane will not be constant along its path. The study of sound propagation will be conducted in this work up to distances of approximately 80 kilometers. To do so, we can utilize the transport formulas for geodesic coordinates adopted from Barahona Fernandes' Hydrography Manual [8] to determine the variation of latitude ($\Delta\phi$), the variation of longitude (ΔL), and the meridian convergence ($\Delta\alpha$).

Both programs — `REOART 2D` and `REOART 3D` — use three different databases to withdraw information regarding the topography and climatology to compute the acoustic ray trajectories in a real environment. These databases are the GEBCO, the WOD18 and the TEOS-10 software library.

3.1. GEBCO bathymetric data

The data provided by GEBCO enables to describe the bathymetry of the ocean almost worldwide. In this work it as used the latest version (2021) of the bathymetric data relative to the North Atlantic

Ocean. In Figure 1 it is possible to see a part of the database through the representation of the area surrounding the United Kingdom.

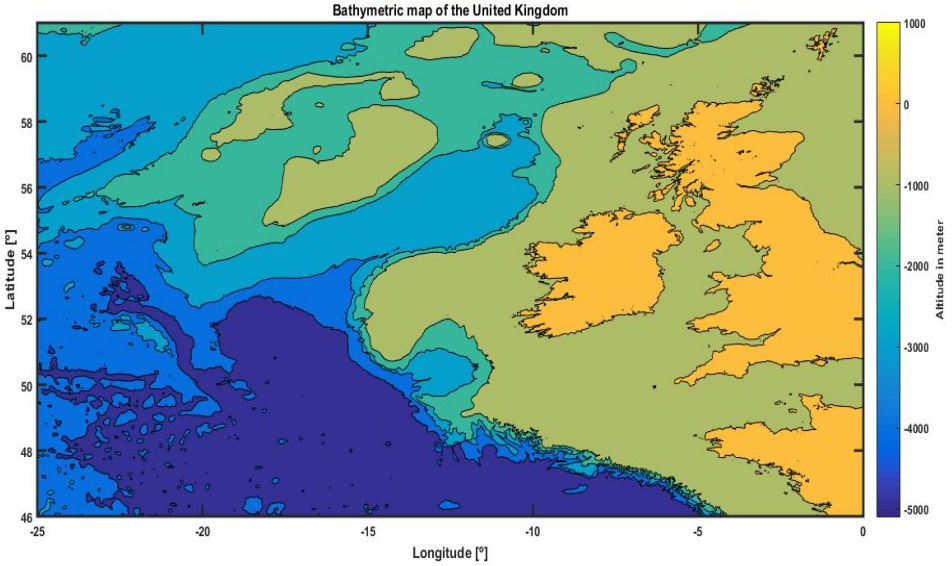


Figure 1: Bathymetric map of the area surrounding the United Kingdom.

3.2. WOD18 climatology data

The WOD18 is a NOAA (*National Oceanic and Atmospheric Administration*) database that contains a variety of data types relevant to oceanography. Nonetheless, we calculated the speed of sound using only the practical salinity and *in situ* temperature. The points with climatological data (practical salinity and *in situ* temperature) in the area surrounding the United Kingdom are depicted in Figure 2. These points are separated by a quarter of a degree.

3.3. TEOS-10

The TEOS-10 software library contains numerous oceanographic functions for computing relevant ocean variables. To calculate the sound speed, the Gibbs-SeaWater (GSW) Oceanographic Toolbox was used, which is a subset of this library. Nevertheless, there are several irregularities when utilizing the function that calculates the speed of sound. To begin, practical salinity and *in situ* temperature cannot be employed, as TEOS-10 operates on an absolute salinity and conservative temperature basis, as demonstrated in [9]. Second, the function used to determine the sound speed requires three variables: absolute salinity, conservative temperature, and hydrostatic pressure. This necessitated the usage of the `gsw_SA_from_SP` and `gsw_CT_from_t` functions to convert practical salinity to absolute salinity and *in situ* temperature to conservative temperature, respectively. Additionally, we calculated the hydrostatic pressure using the `gsw_p_from_z` function. Finally, with the three variables defined, we can use the `gsw_sound_speed` function to calculate the sound speed and define the sound velocity field.

4. REOART 2D AND REOART 3D

As previously stated, these programs —- `REOART 2D` and `REOART 3D` — can simulate in mode 2D or 3D acoustic propagation in a real ocean environment. As already mentioned, these programs use the

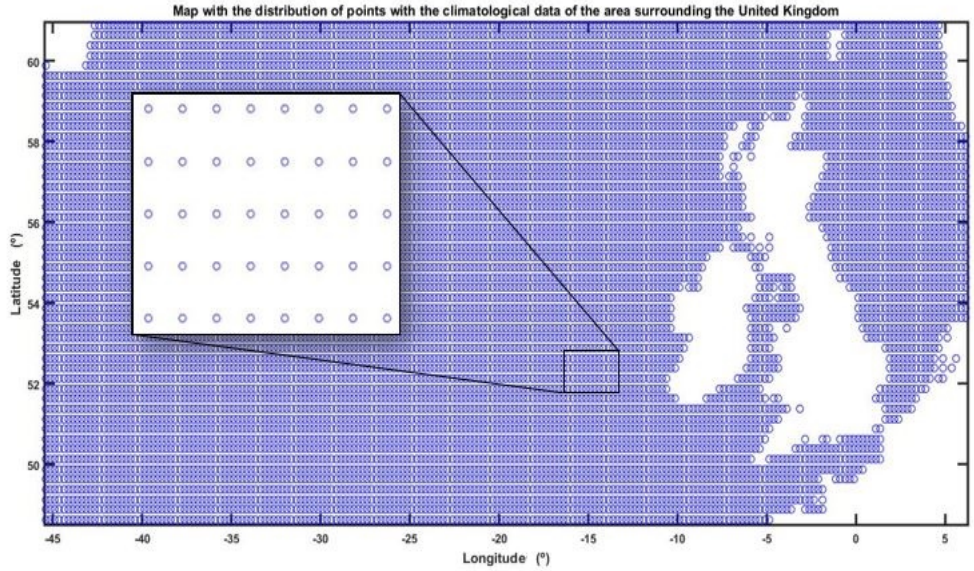


Figure 2: Climatological data of the area surrounding the United Kingdom.

ray tracing propagation models developed by Raposo [4] and the databases mentioned before.

From geographical coordinates, time and simulation area, `REOART 2D` and `REOART 3D`, figure out information about the topography and climatology, and compute the corresponding grid of spatial sound speed and all the necessary gradients used by (1) and (2).

These programs are distinguished from others by their capacity to calculate successive ray reflections in the bottom and sea surface, as well as their use of real bathymetry and sound speed.

5. NUMERICAL SIMULATIONS

In this section we will carry out preliminary simulations in order to test the developed programs. All simulations will be performed in the area surrounding the United Kingdom.

In Section 5.1, we will illustrate the emission of a single acoustic beam using the `REOART 2D` program. In Section 5.2, we will simulate the emission of multiple acoustic rays with different initial directions. In Section 5.3, we will simulate the emission of a single acoustic ray using the `REOART 3D` program.

5.1. REOART 2D — single ray simulation

The first simulation was performed at geographic coordinates 50°N and 15°W on azimuth 270° — note that geographic coordinates to the west have a negative sign.

We can observe the bathymetry of the area surrounding the United Kingdom and the geodesic (represented by the red line) in Figure 3. The geodesic is the 2D propagation plane and, as seen, it cannot extend beyond 80 kilometers. The bathymetry of the 2D propagation plane is depicted in Figure 4.

The following step is to create the sound velocity field. Similarly to the bathymetry determination, we employed the same 2D propagation plane to define the plane's climatology. As a result, this enables the establishment of the sound speed grid and the corresponding gradients.

Two figures are showed in Figure 5. On the left, a similar figure to Figure 2 can be seen, but with the addition of the geodesic representation and some green points. These are the points with

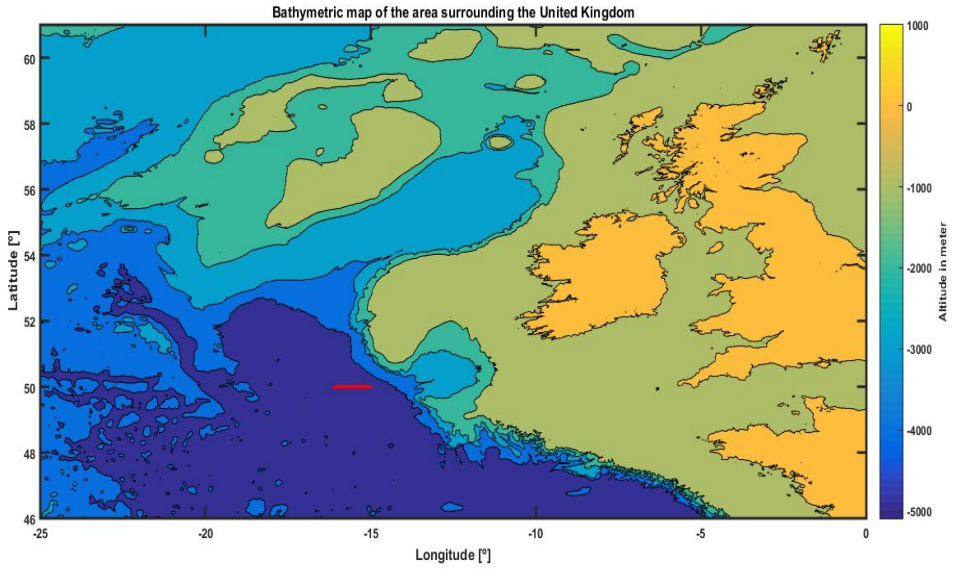


Figure 3: Bathymetric map of the area surrounding the United Kingdom and the representation of the geodesic (2D propagation plane).

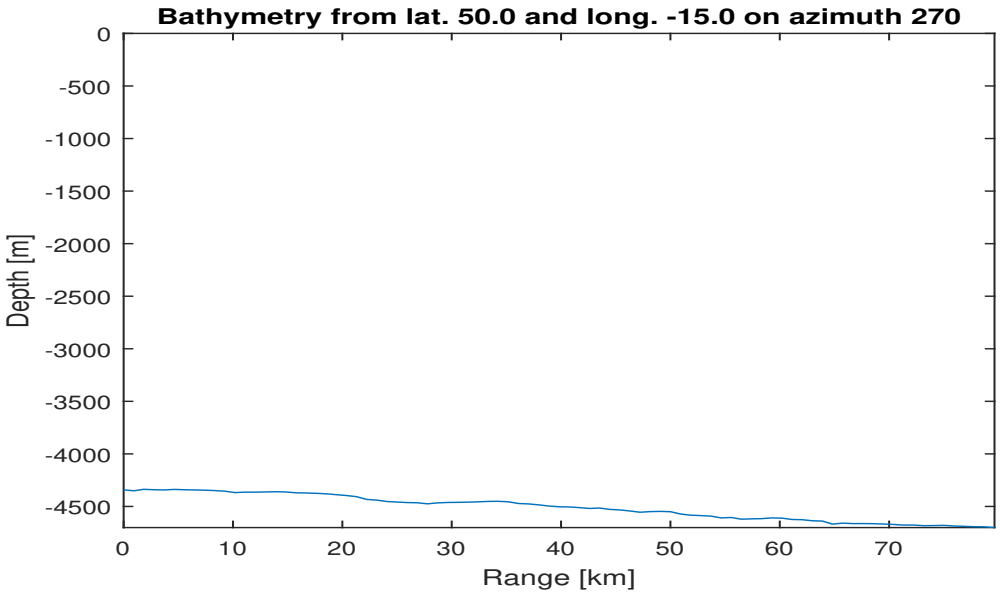


Figure 4: Representation of the bathymetry of the geographic coordinates 50°N and 15°W on azimuth 270°.

climatological data closest to the trajectory. The second one, on the right, is an approximation of the left figure. The requirement for interpolation of climatological data throughout the trajectory is clearly visible. To do this, the concept of the nearest neighbor was applied. Thus, the climatological data were associated with the trajectory's nearest neighbor. As a result, it was possible to determine the climatology in depth along the trajectory. After defining the climatology in the 2D propagation plane, we can finally compute the sound velocity field using the previously described TEOS-10 functions.

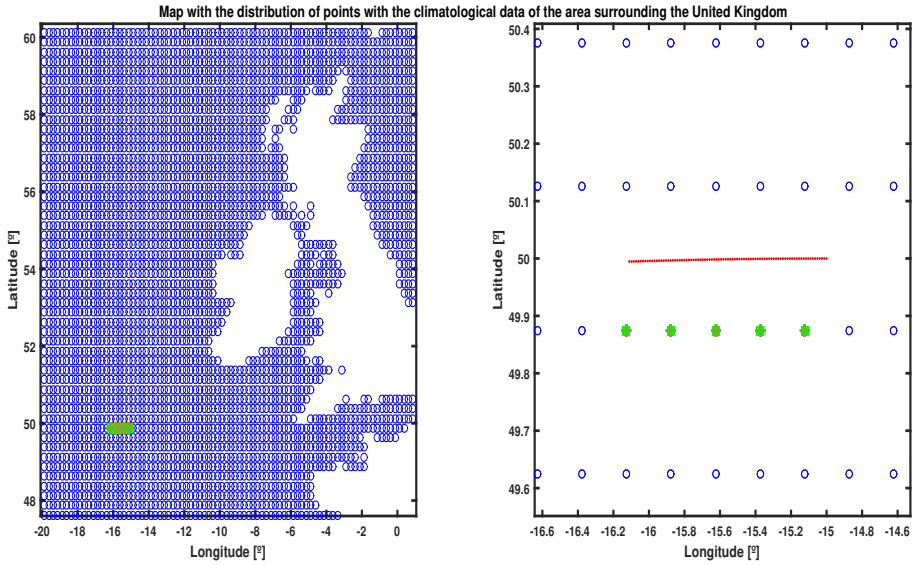


Figure 5: Representation of the map with the distribution of points with the climatological data of the area surrounding the United Kingdom, the geodesic and the nearest neighbors (the green points).

Figure 6 shows a simulation of the acoustic ray propagation at the geographic coordinate 50°N and 15°W on azimuth 270° at a depth of zero meters. We can observe the ray refracting along its path of propagation and reflecting in the sea surface in this figure. As is well known, the sound is described as "lazy". This means that the sound bends towards the regions of slower sound speed. As illustrated in Figure 7, this occurred during the simulation.

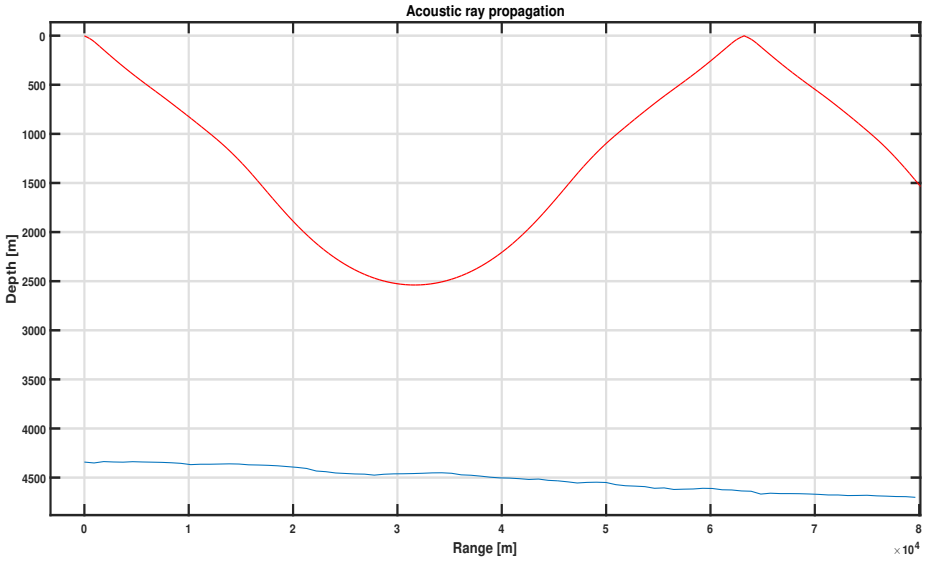


Figure 6: Representation of the simulation of a single ray in the geographic coordinates 50°N and 15°W on azimuth 270° at a depth of zero meters.

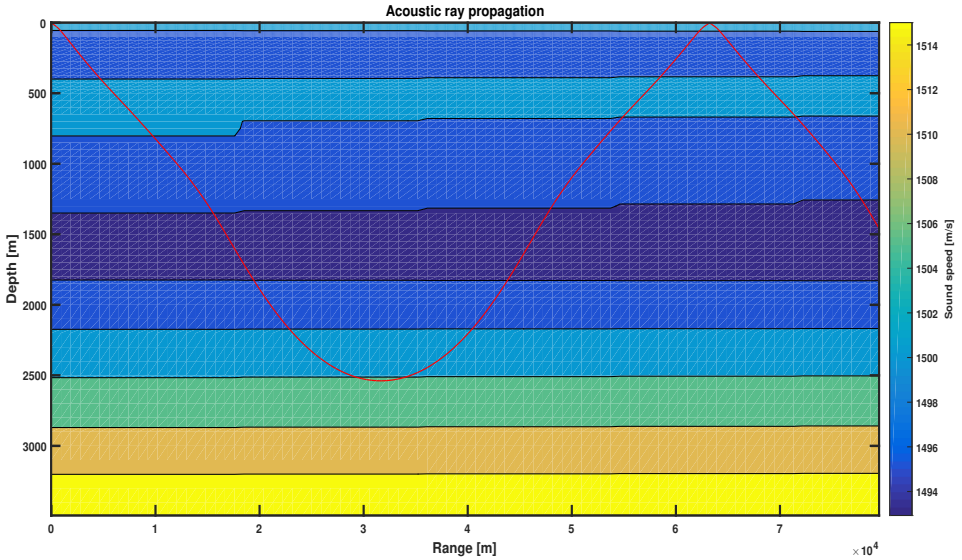


Figure 7: Representation of the simulation of a single ray in the geographic coordinates 50°N and 15°W on azimuth 270° at a depth of zero meters and the sound velocity field.

5.2. REOART 2D — multiple rays simulation

The second simulation was performed at the geographic coordinates 48°N and 9°W on azimuth 200° using multiple rays at zero meters deep. Obtaining the bathymetry and climatology of the 2D propagation plane is exactly the same as in the previous simulation. We will determine the bathymetry and then the climatology through the closest climatological points to the propagation plane.

In Figure 8 it is possible to see the bathymetry of the area surrounding the United Kingdom as the 2D propagation plane (the red line). In Figure 9 we can see the real bathymetry of the 2D propagation plane.

Figure 10 illustrates how the nearest neighbor concept is used to establish the climatology of the 2D propagation plane (the green points are the neighbors). The multiple ray simulation was performed at geographic coordinates 48°N and 9°W on azimuth 200° (Figure 11). Five rays were used in this simulation, each five degrees apart vertically, commencing at zero (0° − 5° − 10° − 15° − 20°), at a depth of one meter.

5.3. REOART 3D — single ray simulation

For the final simulation, we used `REOART 3D` to do a 3D simulation. This simulation was done at the geographic coordinates 53°N and 13°W with a single ray. `REOART 3D` does not permit degrees when calculating acoustic propagation. It is only effective in meters. As a consequence, the axes will be measured in meters rather than degrees from our starting geographic position, along longitude and latitude. In Figure 12 it is possible to see the 3D simulation area (the red square). Note that geographic position 53°N and 13°W is the westernmost lower vertex. This position is our starting position. The bathymetry of the 3D simulation area is shown in Figure 13.

As with `REOART 2D`, the 3D area's climatology is determined using the nearest neighbor concept (Figure 14). The green points represent the neighbors.

The simulation was performed at a depth of 350 m, $8x10^4$ m from the initial geographic coordinates (53°N and 13°W) measured from longitudes and latitudes. In Figure 15 it is possible to see the acoustic ray refracting and reflecting in the bottom.

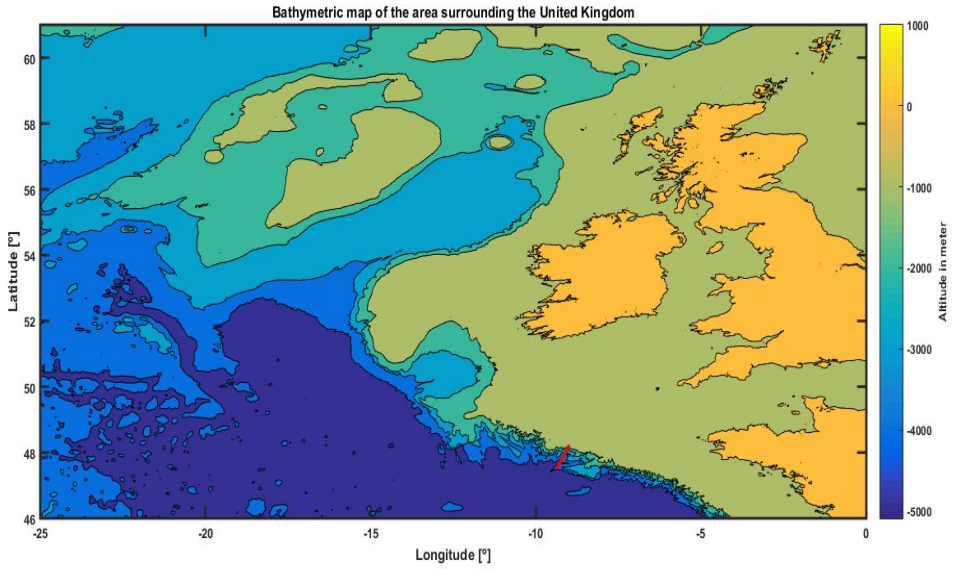


Figure 8: Bathymetric map of the area surrounding the United Kingdom and the representation of the geodesic (2D propagation plane — the red line).

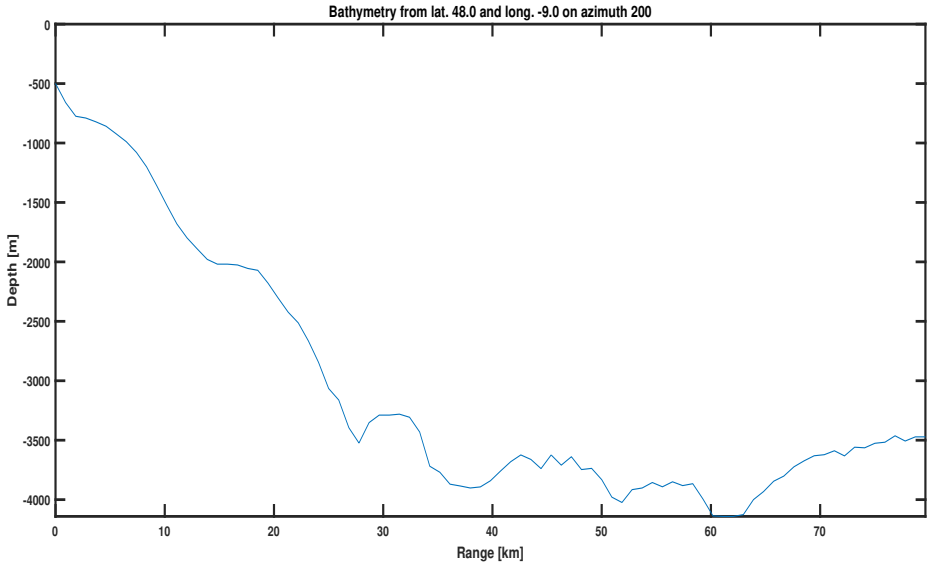


Figure 9: Representation of the bathymetry of the geographic coordinates 48°N and 9°W on azimuth 200°.

6. CONCLUSIONS

The aim of this work was to test preliminary `REOART 2D` and `REOART 3D` programs and assess the credibility of the simulations carried out. Not only do the trajectories seem credible, but also the reflections on the surface that at the bottom of the ocean seem to satisfy Snell's law.

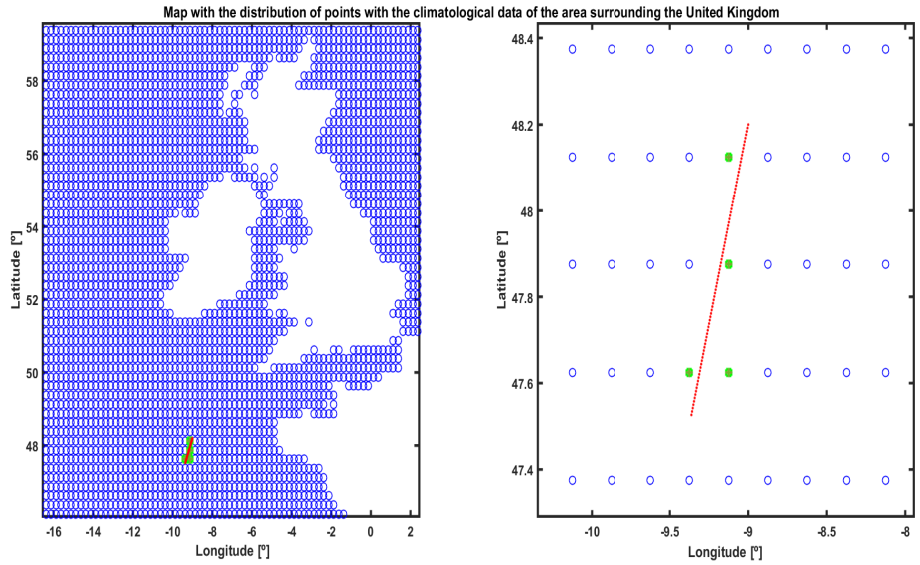


Figure 10: Representation of the map with the distribution of points with the climatological data of the area surrounding the United Kingdom, the geodesic and the nearest neighbors.

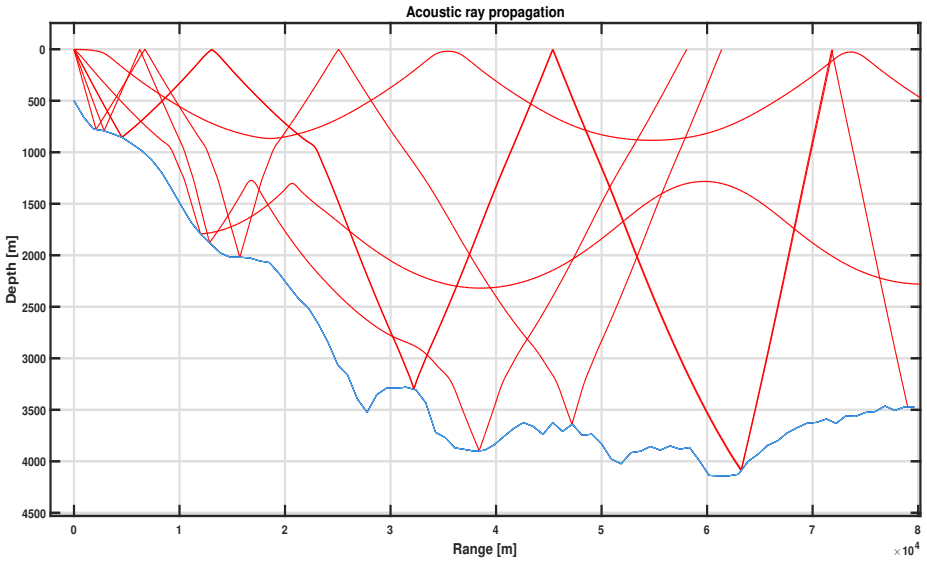


Figure 11: Representation of the simulation of the multiple rays, with five degrees apart vertically, in the geographic coordinates 48°N and 9°W on azimuth 200° at a depth of one meter.

7. FUTURE WORK

The preliminary satisfactory results suggest the following developments:

– Validate the ray trajectories predicted by `REOART 2D` using classic ray tracing software under the same real sound speed grid;

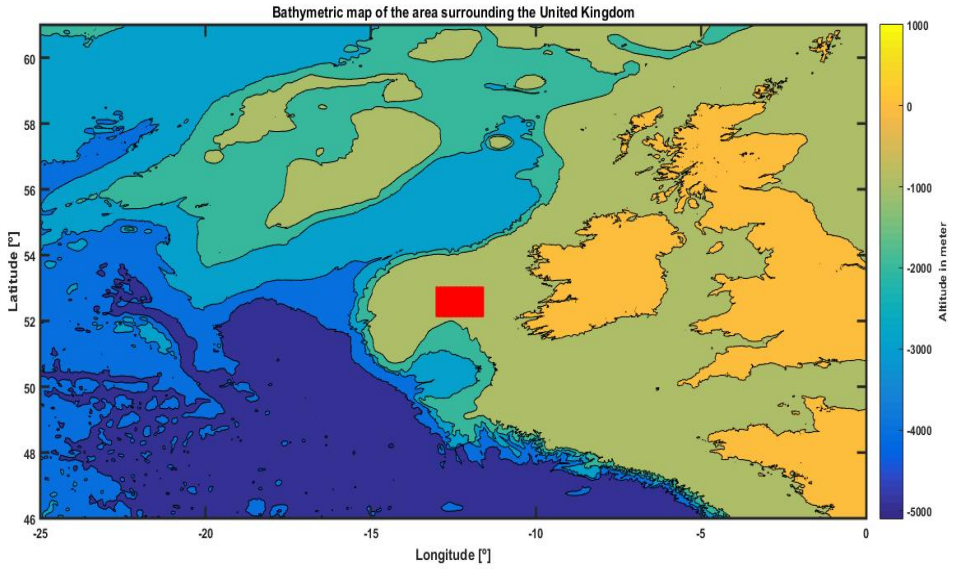


Figure 12: Bathymetric map of the area surrounding the United Kingdom and the representation of the 3D simulation area (the red square).

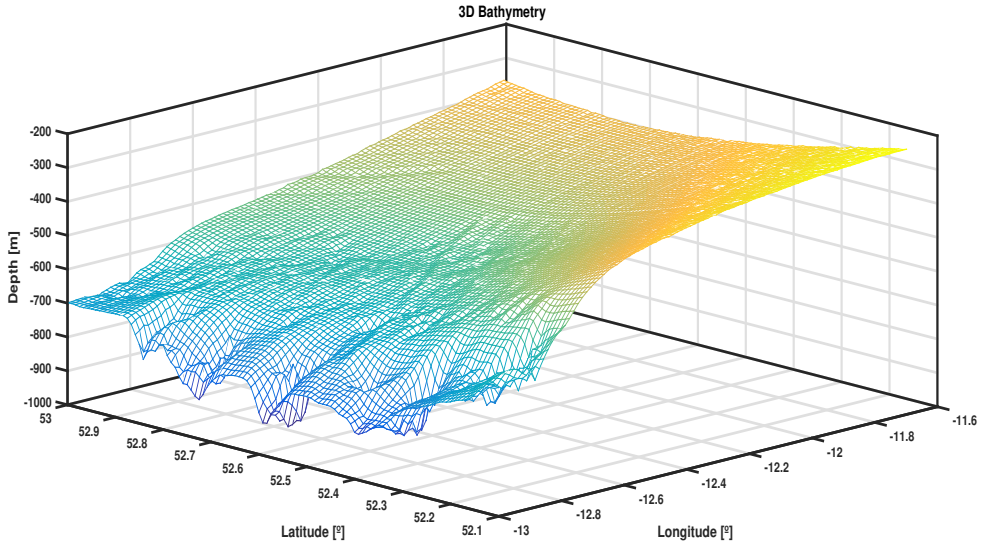


Figure 13: Bathymetry of the 3D simulation area.

– Improve the efficiency of the programs developed, namely the `REOART 3D` program.

ACKNOWLEDGEMENTS

Firstly, I would like to express my very great appreciation to my thesis advisor Miguel Moreira, for his valuable and constructive suggestions during the planning and development of this research work.

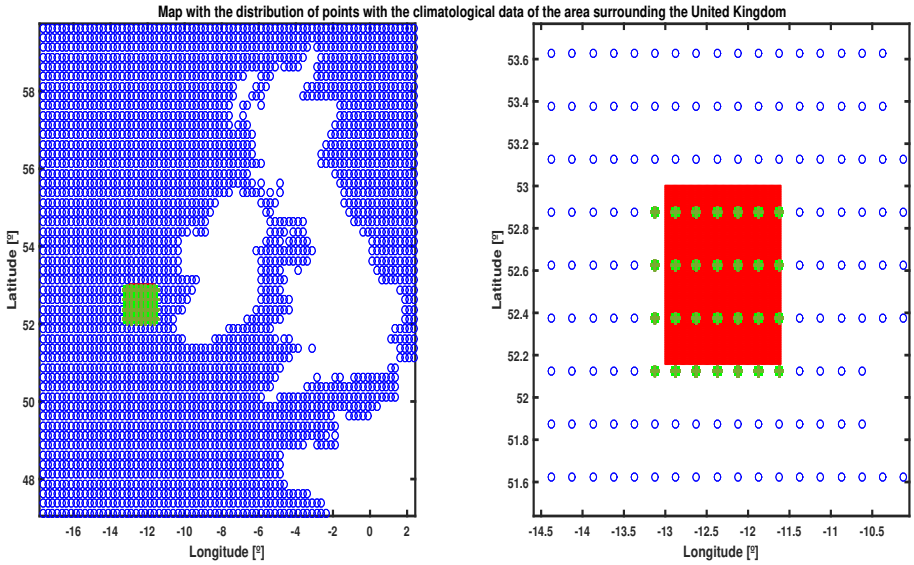


Figure 14: Representation of the map with the distribution of points with the climatological data of the area surrounding the United Kingdom, the 3D simulation area and the nearest neighbors.

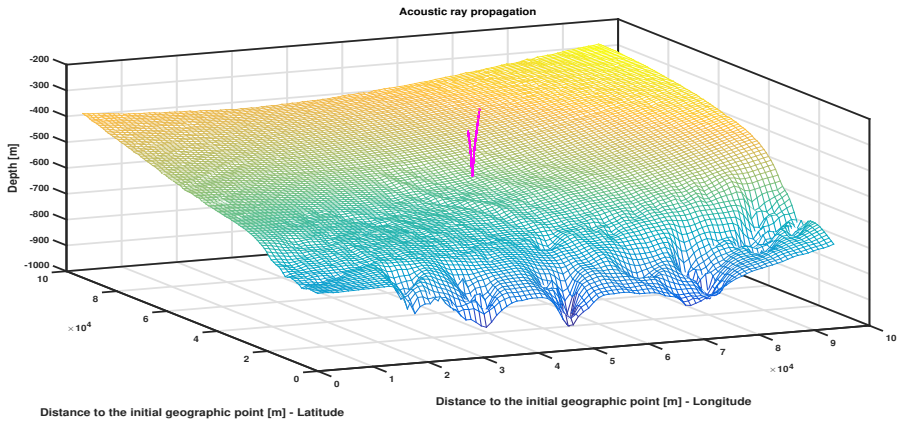


Figure 15: Perspective 1 — Representation of the simulation of a single ray at 8x10^4 m from the initial geographic coordinates (53°N and 13°W) measured from longitudes and latitudes at a depth of 350 m.

To my co-advisors, Carlos Costa and Mário Gatta, I would also like to extend my thanks for all the knowledge passed in the area of oceanography and their valuable technical support on this work.

Last but not least, I acknowledge CINAV the Portuguese Navy's Research Center for the support in carrying out this work.

REFERENCES

[1] Daljit S Ahluwalia and Joseph B Keller. *Wave propagation and underwater acoustics*. Springer, 1977.

[2] Paul C Etter. *Underwater acoustic modeling: principles, techniques and applications*. CRC Press, 1995.

[3] P. Raposo, M. Gatta, and M. Moreira. Ocean-acoustic raytracing propagation models based on fermat's least time principle. *Inter-Noise*, 2019.

[4] P. Raposo. *Estudo da propagação acústica 3D em meios não homogéneos – Raytracing baseado no príncipio de Fermat*. Escola Naval, 2019.

[5] Uri M Ascher and Linda R Petzold. *Computer methods for ordinary differential equations and differential-algebraic equations*, volume 61. Siam, 1998.

[6] A. J. Roberts. Differential algebraic equations solvers. Technical report, The University of Adelaide, 1998.

[7] Tiago CA Oliveira, Ying-Tsong Lin, and Michael B Porter. Underwater sound propagation modeling in a complex shallow water environment. *Frontiers in Marine Science*, page 1464, 2021.

[8] J.A. Barahona Fernandes. *Manual de Hidrografia*. Instituto Hidrográfico, 1967.

[9] Feistel, R TJ, McDougall, FJ Millero, DR Jackett, DG Wright, BA King, GM Marion, C Chen, P Spitzer, and S Seitz. The international thermodynamic equation of seawater 2010 (teos-10): Calculation and use of thermodynamic properties. *Global Ship-based Repeat Hydrography Manual, IOCCP Report No*, 14, 2009.

Apêndice E

Trabalho apresentado no colóquio 7^{as} Jornadas de Engenharia Hidrográfica/2^{as} Jornadas Luso-Espanholas de Hidrografia

7as Jornadas de Engenharia Hidrográfica / 2as Jornadas Luso-Espanholas de Hidrografia

Desenvolvimento de algoritmo para previsão 2D e 3D de propagação submarina de raios acústicos com reflexões – ferramentas REOART 2D e REOART 3D

Autores: J. Gonçalves (1), M. Gatta (2), C. Costa (3),

(1) CINAV, Escola Naval, IUM, Base Naval de Lisboa, joao.duarte.goncalves@escolanaval.pt.

(2) CINAV, Escola Naval, IUM, Base Naval de Lisboa, mario.gatta@escolanaval.pt.

(3) CINAV, Base Naval de Lisboa, cnlcosta@gmail.com.

Resumo: Utilizando modelos paramétricos 2D e 3D de propagação de raios acústicos deduzidos no passado recente foram desenvolvidos dois programas de propagação de raios acústicos com a capacidade de simular a propagação do som em ambiente oceânico real (REOART 2D e REOART 3D). Estes programas utilizam as bases de dados da GEBCO, da WOD18 e da biblioteca de software TEOS-10, para recolher dados reais referentes à topografia, climatologia da região de simulação e velocidade de propagação do som. As ferramentas desenvolvidas, para além de simularem a propagação de raios acústicos de forma mais realista, incluem também a ocorrência do fenómeno de reflexão do raio acústico (suposta especular) na superfície e no fundo marinho entrando em consideração com a batimetria real. Neste trabalho introduzimos as ferramentas referidas e apresentaremos algumas simulações ilustrativas.

Palavras-chave: propagação acústica 2D, propagação acústica 3D, propagação acústica em ambiente real

1. INTRODUÇÃO

Nos últimos anos, a modelação da propagação acústica submarina têm vindo a ser particularmente desenvolvida e aplicada na resolução de importantes problemas tais como, a previsão sonar (Daljit e Joseph, 1997), entre outros. Diferentes tipos de modelos de propagação têm sido aplicados no desenvolvimento de aplicações computacionais dedicadas ao estudo deste fenómeno. De referir, por exemplo, a utilização de modelos de traçado de raios sonoros baseados na aproximação *eikonal*, modelos baseados na resolução da equação de onda recorrendo a modos ortogonais, modelos baseados em técnicas de geometria diferencial e modelos baseados no princípio do tempo mínimo de Fermat (Rodriguez *et al.*, 2021).

O desenvolvimento deste tipo de ferramentas destinadas a simular e a prever o fenómeno de propagação acústica tem uma grande importância nos domínios militar e civil. No campo militar, a previsão sonar permite, por exemplo, identificar sectores nos quais a detecção de alvos não é viável. No contexto civil, este tipo de ferramentas permite, por exemplo, simular e avaliar a exposição de ecossistemas a diferentes tipos de ruído (Etter, 1995).

Foi recentemente desenvolvida, no contexto das atividades do CINAV/Escola Naval, uma aplicação de apoio à previsão sonar (FAPSO - Ferramenta de Apoio à Previsão SOnar) (Marques, 2018). A adição de modelos alternativos de propagação e funcionalidades adicionais às aplicações referidas está na base do presente trabalho. Neste intuito, foram desenvolvidas as aplicações computacionais de propagação acústica submarina denominadas – REOART 2D e REOART 3D (**R**eal **E**nvironment **O**cean-**A**coustic **R**ay **T**racing). Estas recorrem a dois modelos paramétricos 2D e 3D de propagação de raios acústicos submarinos, que têm por base o princípio do tempo mínimo de Fermat, materializados por sistemas de equações algébrico-diferenciais (Raposo, 2019). As aplicações desenvolvidas - REOART 2D e REOART 3D utilizam os atrás referidos modelos paramétricos, as bases de dados da GEBCO (*General Bathymetric Charts of the Ocean*), da WOD18 (World Ocean Database 2018) e a biblioteca de *software* TEOS-10 (*Thermodynamic Equation of State 2010*), e têm a capacidade de simular trajetórias de raios acústicos e respectivas reflexões na superfície e fundo, em ambiente oceânico real (2D e 3D).

2. FUNCIONAMENTO DOS ALGORITMOS DESENVOLVIDOS

Nesta secção irá explicar-se o funcionamento lógico dos algoritmos desenvolvidos através da linguagem MATLAB. Na Figura 1 está representado o fluxograma com as etapas executadas pelos programas REOART 2D e REOART 3D.

A primeira etapa para o cálculo da propagação do raio acústico em ambiente real é a introdução da posição geográfica, do azimute, da distância de propagação e da distância entre posições. Isto permite definir a zona de simulação, podendo ela ser um plano (2D) ou uma área (3D), consoante o programa utilizado. Com a

zona de simulação determinada, com recurso às bases de dados da GEBCO e da WOD18, a segunda etapa é a caracterização da topografia e da climatologia da zona de simulação. A terceira etapa passa pelo cálculo da velocidade de som e respetivos gradientes espaciais da zona de simulação, através da biblioteca TEOS-10 (GSW), determinando assim os campos de velocidade de som e os gradientes. Na quarta etapa é onde é feita a resolução numérica dos modelos de propagação de Raposo (que serão abordados mais adiante) – cálculo da trajetória do raio acústico – e a modelação das reflexões.

De referir que a referenciação relativamente às coordenadas espaciais no meio de propagação de todos os parâmetros físicos e batimétricos necessários ao desenvolvimento dos cálculos e resolução do modelo de propagação, recorreu ao procedimento de interpolação multilinear dos dados discretizados disponíveis nas bibliotecas da GEBCO e da WOD18.

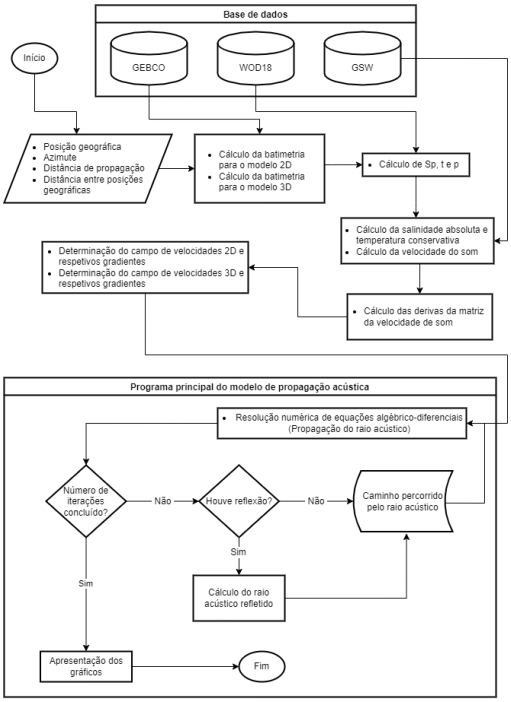


Figura 1 - Fluxograma do funcionamento lógico dos programas REOART 2D e REOART 3D.

3. MODELOS PARAMÉTRICOS DE PROPAGAÇÃO SONORA ACÚSTICA

Os modelos de propagação utilizados nos programas REOART 2D e REOART 3D foram desenvolvidos por Raposo em 2019 (Raposo, 2019). Estes modelos paramétricos de traçado de raios acústicos, são constituídos pelos sistemas de equações algébrico-diferenciais que seguidamente se apresentam e configuram, respectivamente o modelo 2D,

$$\begin{cases} x' - a = 0 \\ y' - b = 0 \\ a'b - b'a + \frac{b}{v(x,y)}\frac{\partial v}{\partial x} - \frac{a}{v(x,y)}\frac{\partial v}{\partial y} = 0 \\ aa' + bb' = 0 \end{cases}$$

e o modelo 3D,

$$\begin{cases} x' - a = 0 \\ y' - b = 0 \\ z' - c = 0 \\ (1-a')a' + (1-c^2)c' - ac(a'+c') - b(ab'+cb') \\ \quad + \frac{1}{v}\left(\frac{\partial v}{\partial x} + \frac{\partial v}{\partial z}\left(\frac{dv}{dx}a + \frac{dv}{dy}b + \frac{dv}{dz}c\right)(a+c)\right) = 0 \\ (1-b')b' + (1-c^2)c' - bc(b'+c') - a(ba'+ca') \\ \quad + \frac{1}{v}\left(\frac{\partial v}{\partial y} + \frac{\partial v}{\partial z}\left(\frac{dv}{dx}a + \frac{dv}{dy}b + \frac{dv}{dz}c\right)(b+c)\right) = 0 \\ aa' + bb' + cc' = 0 \end{cases}$$

em que *x*, *y*, *z*, representam, respectivamente, a posição do raio acústico em coordenadas cartesianas e *v* a velocidade do som.

4. FORMULAÇÃO 2D E 3D EM AMBIENTE REAL

Quando a velocidade de som apresenta gradientes normais ao plano de propagação, o raio acústico não se mantém no plano em causa. Neste caso, a modelação do fenómeno de propagação acústica em 3D será mais realista. Por outro lado, mesmo na ausência de gradientes normais ao plano de propagação, a ocorrência de reflexões no fundo marinho irregular determina em geral o abandono do plano de propagação por parte do raio refletido (Oliveira *et al.*, 2021). Estes factos justificam a importância de se utilizar, nas circunstâncias referidas, um modelo/aplicação que permita simular a propagação de raios acústicos em 3D.

5. REOART 2D E REOART 3D

Como foi referido, os programas REOART 2D e REOART 3D utilizam as bases de dados GEBCO e WOD18 para aceder à informação batimétrica e climatológica. A utilização da biblioteca de *software* TEOS-10 permite efetuar a determinação do campo espacial de velocidade do som na região oceânica considerada.
Refira-se que a base de dados da GEBCO descreve a batimetria na generalidade das regiões terrestres com resolução de 0.5 milhas.

.

6. SIMULAÇÕES NUMÉRICAS

Tendo em vista ilustrar/demonstrar as aplicações desenvolvidas apresentaremos nesta secção simulações de propagação de raios em regiões envolventes a Portugal Continental.

Começaremos por efetuar simulações de propagação e de reflexões de múltiplos raios acústicos com a aplicação REOART 2D. Seguidamente

apresentaremos a simulação da trajectória com algumas reflexões dum único raio acústico, utilizando o programa REOART 3D.

5.1 REOART 2D – simulação numérica com múltiplos raios acústicos

Esta simulação numérica foi realizada na posição geográfica de latitude 39º N e longitude 10.1º W, no azimute 270º, com múltiplos raios acústicos.

Na Figura 2 está representada a área, com os valores batimétricos, envolvente a Portugal Continental e o plano de simulação vertical 2D (a linha vermelha). A linha referida é uma geodésica, calculada através das fórmulas de transporte de coordenadas geodésicas adaptadas do Manual de Hidrografia (Fernandes, 1967).

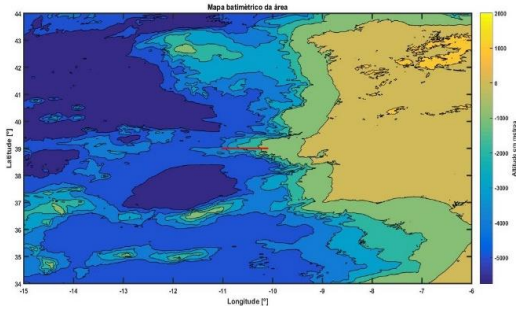


Figura 2 - Mapa batimétrico da área envolvente a Portugal Continental com a representação da trajetória (geodésica).

Na Figura 3 está representado o plano de simulação 2D (a linha vermelha), a rede de dados climatológicos (círculos azuis) e a rede dos dados climatológicos mais próximos do plano de simulação 2D (pontos verdes).

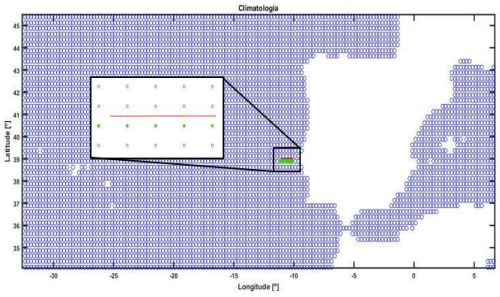


Figura 3 - Mapa com a distribuição dos pontos com dados climatológicos da área envolvente a Portugal Continental com o detalhe da representação da trajetória assumida.

Na Figura 4 está ilustrada a simulação realizada recorrendo à emissão de múltiplos raios acústicos, a um metro de profundidade. Nesta figura é possível observar a reflexão especular nas fronteiras e a refração ao longo da trajetória, satisfazendo qualitativamente a lei de Snell. Esta simulação sugere a robustez e credibilidade do programa REOART 2D.

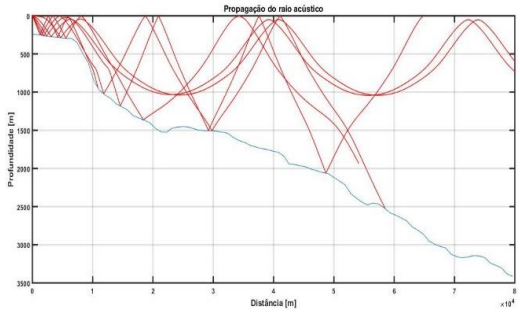


Figura 4 - Representação de uma simulação com múltiplos raios acústicos. utilizando o programa REOART 2D.

5.1 REOART 3D – simulação numérica com um raio acústico

A simulação numérica com o programa REOART 3D foi realizada na posição geográfica de latitude 36.9º N e longitude 9º W, com um único raio acústico, a uma profundidade de cem metros. Na Figura 5 está representada a área, com os valores batimétricos, envolvente a Portugal Continental e a área de simulação 3D (quadrado vermelho).

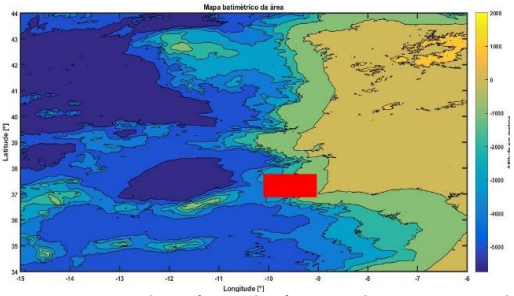


Figura 5- Mapa batimétrico da área envolvente a Portugal Continental com a área de simulação 3D (assinalada a cor vermelha)

Na Figura 6 representamos a rede dos dados climatológicos consultados pela aplicação.

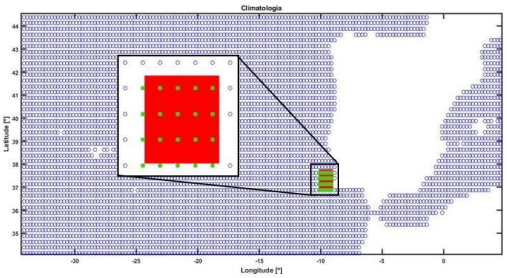


Figura 6 - Mapa com a distribuição dos pontos com dados climatológicos da área envolvente a Portugal Continental com o detalhe da representação da área de simulação 3D.

Nas Figuras 7 e Figura 8, ilustramos a simulação efetuada onde podemos observar algumas reflexões do raio acústico ao longo do seu trajecto, na superfície e no fundo. Uma vez que o tempo de cálculo para gerar a simulação 3D revelou-se substancialmente

longo optámos por ilustrar apenas um trajecto compatível com o tempo de cálculo disponível. Naturalmente, a presente simulação ilustra/demostra nesta fase de desenvolvimento da aplicação REOART 3D apenas o seu "conceito" e sugere a necessidade de se optimizar os algoritmos utilizados.

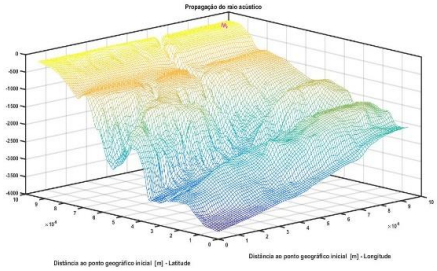


Figura 7 - Representação de uma simulação com um raio acústico, utilizando o programa REOART 3D.

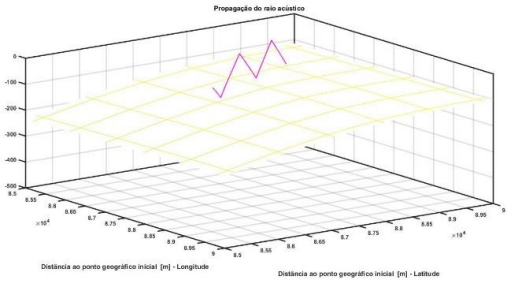


Figura 8 - Representação de uma simulação com um raio acústico, utilizando o programa REOART 3D.

7. CONCLUSÕES E TRABALHO FUTURO

Os resultados das simulações apresentadas são credíveis.

A consideração de reflexões nos trajectos dos raios acústicos simulados pelas aplicações desenvolvidas acrescenta realismo no fenómeno de propagação do som num ambiente real.

A disponibilidade duma aplicação 3D funcional para simulação do trajeto de raios acústicos em ambiente real pode ser importante e vantajoso em aplicações especializadas, nomeadamente em zonas com gradientes transversais de velocidade do som importantes tais como na vizinhança de fontes hidrotermais.

Os resultados preliminares das simulações numéricas sugerem o desenvolvimento futuro das seguintes ações:

1. Validação das aplicações desenvolvidas recorrendo a aplicações disponíveis comercialmente;
2. Optimização das aplicações e desenvolvimento dum interface amigável para a sua utilização;
3. Integração das aplicações REOART 2D e REOART 3D na ferramenta FAPSO.
4. Adicionar a consideração do fenómeno de absorção acústica nos modelos de propagação das aplicações REOART 2D e REOART 3D.

REFERÊNCIAS

Daljit S. Ahluwalia and Joseph B. Keller (1997). *Wave propagation and underwater acoustics*. Springer.

Etter, P. C. (1995). *Underwater acoustic modeling: principles, techniques and applications*. CRC Press.

Fernandes, J. B. (1967). *Manual de Hidrografia*. Instituto Hidrográfico.

Marques, P. A. V. (2018). *Acústica submarina: Desenvolvimento de modelo de previsão de propagação acústica global*. Escola Naval.

Orlando Rodriguez, Sérgio de Jesus e Rogério M. Calazan (2021). *Fundamentos de acústica submarina*. Universidade do Algarve.

Raposo, P. A. M. A (2019). *Estudo da propagação acústica em 3D em meios não homogéneos – Raytracing baseado no princípio de Fermat*. Escola Naval.

Tiago C. Oliveira, Ying-Tsong Lin and Michael B. Porter (2021). *Underwater sound propagation modeling in a complex shallow water environment*. Frontiers in Marine Science, 1460-1464.

Apêndice F

Trabalho apresentado no colóquio Jornadas do Mar 2022

Desenvolvimento de algoritmo para previsão 2D de propagação submarina de raios acústicos com reflexões – ferramenta REOART 2D

João Gonçalves
Escola Naval
Joao.duarte.goncalves@marinha.pt

Resumo: Utilizando modelos paramétricos 2D de propagação de raios acústicos deduzidos no passado recente, foi desenvolvido um programa de propagação de raios acústicos com a capacidade de simular a propagação do som em ambiente oceânico real (REOART 2D). Este programa utiliza as bases de dados da GEBCO, da WOD18 e da biblioteca de software TEOS-10, para recolher dados reais referentes à topografia, climatologia da região de simulação e velocidade de propagação do som. A ferramenta desenvolvida, para além de simular a propagação de raios acústicos de forma mais realista, inclui também a ocorrência do fenómeno de reflexão do raio acústico (suposta especular) na superfície e no fundo marinho, entrando em consideração com a batimetria real. Neste trabalho introduzimos a ferramenta referida e apresentaremos algumas simulações ilustrativas.

Palavras-chave: propagação acústica 2D, propagação acústica em ambiente real

INTRODUÇÃO

Nos últimos anos, as modelações da propagação acústica submarina têm vindo a ser particularmente desenvolvidas e aplicadas na resolução de importantes problemas tais como, a previsão sonar (Daljit e Joseph, 1997), entre outros. Diferentes tipos de modelos de propagação têm sido aplicados no desenvolvimento de aplicações computacionais dedicadas ao estudo deste fenómeno. De referir, por exemplo, a utilização de modelos de traçado de raios sonoros baseados na aproximação *eikonal*, modelos baseados na resolução da equação de onda recorrendo a modos ortogonais, modelos baseados em técnicas de geometria diferencial e modelos baseados no princípio do tempo mínimo de Fermat (Rodriguez et al., 2021).

O desenvolvimento deste tipo de ferramentas destinadas a simular e a prever o fenómeno de propagação acústica tem uma grande importância nos domínios militar e civil. No campo militar, a previsão sonar permite, por exemplo, identificar setores nos quais a deteção de alvos não é viável. No contexto civil, este tipo de ferramentas permite, por exemplo, simular e avaliar a exposição de ecossistemas a diferentes tipos de ruído (Etter, 1995).

Foi recentemente desenvolvida, no contexto das atividades do CINAV/Escola Naval, uma aplicação de apoio à previsão sonar (FAPSO - Ferramenta de Apoio à Previsão SOnar) (Marques, 2018). A adição de modelos alternativos de propagação e funcionalidades adicionais às aplicações referidas está na base do presente trabalho. Neste intuito, foi desenvolvida a aplicação computacional de propagação acústica submarina denominada – REOART 2D (**R**eal **E**nvironment **O**cean-**A**coustic **R**ay **T**racing). Este recorre ao modelo paramétrico 2D de propagação de raios acústicos submarinos, que tem por base o princípio do tempo mínimo de

Fermat, materializados por sistemas de equações algébrico-diferenciais (Raposo, 2019). A aplicação desenvolvida – REOART 2D – utiliza o referido modelo paramétrico, as bases de dados da GEBCO (General Bathymetric Charts of the Ocean), da WOD18 (World Ocean Database 2018) e a biblioteca de *software* TEOS-10 (Thermodynamic Equation of State 2010), e tem a capacidade de simular trajetórias de raios acústicos e respetivas reflexões na superfície e fundo, em ambiente oceânico real em 2D.

1. MODELO PARAMÉTRICO DE PROPAGAÇÃO SONORA ACÚSTICA

O modelo de propagação utilizado no programa REOART 2D foi desenvolvido por Raposo em 2019 (Raposo, 2019). Este modelo paramétrico de traçado de raios acústicos, é constituído por um sistema de equações algébrico-diferenciais. No modelo 2D,

$$\begin{cases} x' - a = 0 \\ y' - b = 0 \\ a'b - b'a + \frac{b}{v(x,y)}\frac{\partial v}{\partial x} - \frac{a}{v(x,y)}\frac{\partial v}{\partial y} = 0 \\ aa' + bb' = 0 \end{cases} \tag{1}$$

x, *y* e *z*, representam, respetivamente, a posição do raio acústico em coordenadas cartesianas e *v* a velocidade do som.

2. MODELAÇÃO DA REFLEXÃO

Na modelação da reflexão suposta especular, utilizámos a aplicação linear materializada pelas matrizes

$$\mathbf{R}(\mathbf{n})\,\mathbf{r} = \mathbf{i}. \tag{2}$$

Na expressão anterior **i** representa o raio incidente e **r** o raio refletido (ambos normalizados). Os termos da matriz de reflexão **R** são calculados em cada ponto de incidência conhecendo o vetor normal ao plano de incidência no ponto de incidência. Apresentamos seguidamente as expressões utilizadas para calcular os termos da matriz de reflexão **R**:

$$\underbrace{\begin{bmatrix} 1-2n_1^2 & -2n_1n_2 \\ -2n_1n_2 & 1-2n_2^2 \end{bmatrix}}_{\mathbf{R}(\mathbf{n})} \underbrace{\begin{bmatrix} i_1 \\ i_2 \end{bmatrix}}_{\mathbf{i}} = \underbrace{\begin{bmatrix} r_1 \\ r_2 \end{bmatrix}}_{\mathbf{r}} \tag{3}$$

Neste caso, o plano considerado será a superfície da água, ou o plano tangente à batimetria no ponto de incidência do raio acústico considerado. Na dissertação de mestrado (Gonçalves, 2022) (em preparação) podem consultar-se os detalhes da dedução da formulação do fenómeno de reflexão referido.

3. FORMULAÇÃO 2D EM AMBIENTE OCÊANICO REAL

Na propagação acústica em 2D, é habitual admitir que no oceano a propagação de um raio acústico ocorre num plano vertical que interseta a superfície numa geodésica. Esta assunção pressupõe a ausência de correntes e a inexistência de variações significativas na velocidade de som em direções transversais relativamente ao plano vertical referido e simplifica o cálculo da trajetória, que pode passar a ser descrita a duas dimensões, a cota *z* e a coordenada local *x* (no plano de propagação). Nestas circunstâncias, o azimute do raio acústico ao longo deste plano não será constante ao longo da sua trajetória.

As linhas geodésicas são curvas que representam a distância mais curta entre dois pontos num espaço. Na superfície esférica a geodésica é um círculo máximo, enquanto numa superfície plana caracteriza-se por segmentos de reta.

Caso as distâncias de propagação forem diminutas (inferiores a 80 quilómetros) o cálculo de variação de latitude $\Delta\phi$, da variação de longitude ΔL e da convergência de meridianos $\Delta\alpha$, devem ser feitos através das fórmulas de transporte de coordenadas geodésicas adaptadas do Manual de Hidrografia (Fernandes, 1967). De referir que a utilização das expressões anteriormente referidas só é válida para distâncias inferiores a cerca de 80 quilómetros.

Resumidamente, o programa REOART 2D, além de usar o modelo paramétrico 2D de Raposo (Raposo, 2019), utiliza três diferentes bases de dados para retirar informação alusiva à batimetria, climatologia e velocidade do som, para modelar os raios acústicos em ambiente oceânico real. Essas bases de dados são a GEBCO, WOD18 e a biblioteca de *software* TEOS-10.

3.1. DADOS BATIMÉTRICOS GEBCO

Para a descrição batimétrica do fundo utilizou-se a base de dados disponibilizada pela GEBCO, que na versão de 2021 fornece dados batimétricos da generalidade dos mares e oceanos da terra. Contudo, na simulação realizada, o espaço de simulação foi

restringido ao espaço marítimo envolvente ao Arquipélago da Região Autónoma dos Açores – ver Figura 1.

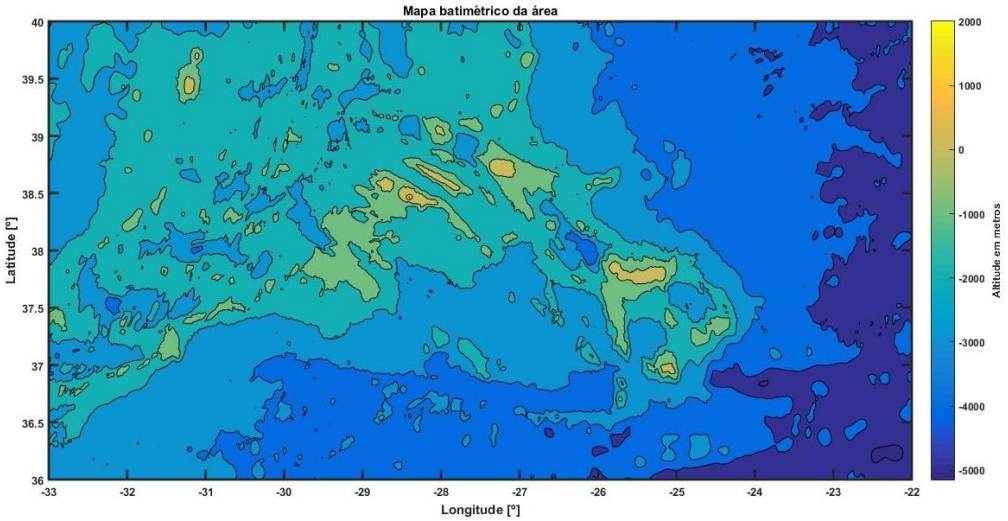


Figura 1- Mapa batimétrico da área envolvente ao Arquipélago da Região Autónoma dos Açores.

3.2. DADOS CLIMATOLÓGICOS WOD18

A WOD18 é uma base de dados disponibilizada pela NOAA (*National Oceanic and Atmospheric Administration*) que contém vários tipos de dados com interesse oceanográfico. Esta base de dados pode disponibilizar dados de acordo com os meses e as estações do ano com resoluções espaciais de cinco graus, um grau e um quarto de grau de latitude e longitude. No programa REOART 2D foi somente utilizado os dados alusivos à salinidade prática e temperatura *in situ*. Na Figura 2 estão representados dados climatológicos correspondentes à distribuição espacial da salinidade prática e da temperatura *in situ* da coluna de água, na resolução de um quarto de grau, da média anual, representados por circunferências azuis, na posição geográfica respetiva.

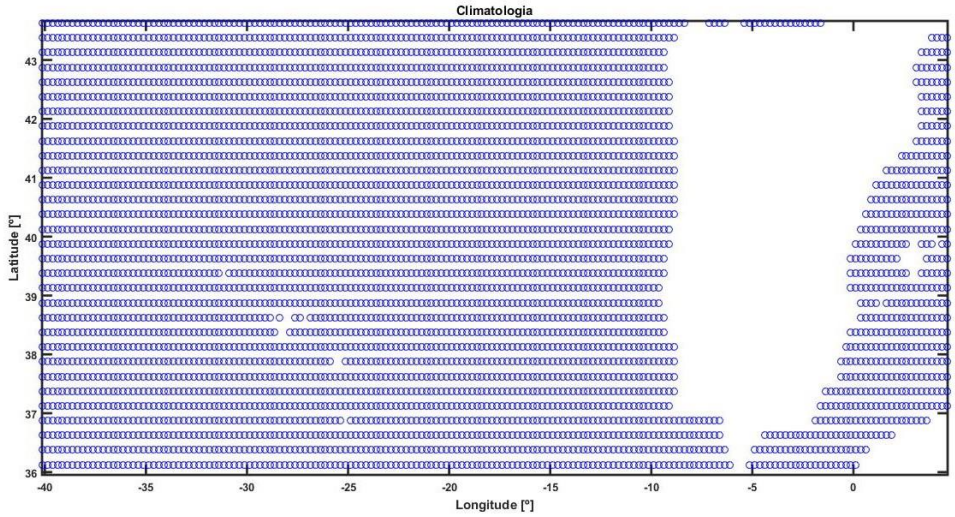


Figura 2 - Mapa com a distribuição dos pontos com dados climatológicos da área envolvente a Portugal Continental e ao Arquipélago da Região Autónoma dos Açores.

3.3. FUNÇÕES TEOS-10

A biblioteca de software da TEOS-10 contém várias funções que permitem modelar várias variáveis oceanográficas, como o caso da velocidade de som no oceano. Para calcular a velocidade do som, foi usada uma secção desta biblioteca chamada Gibbs-SeaWater (GSW) *Oceanographic Toolbox*. Porém, existem alguns inconvenientes ao usar a função para calcular a velocidade do som (`gsw_sound_speed`). Para começar, os dados fornecidos pela WOD18 (salinidade prática e temperatura *in situ*) não podem ser utilizados na biblioteca TEOS-10, pois como se pode consultar no documento oficial desta biblioteca (Feistel et al., 2009), apenas opera com salinidade absoluta e temperatura conservativa. Em segundo, a função utilizada para calcular a velocidade do som requer o uso de três variáveis, a salinidade absoluta, a temperatura conservativa e a pressão hidrostática. Para colmatar o exposto, a biblioteca de software TEOS-10 dispõe de outras três funções que permitem transformar a salinidade prática em absoluta, a temperatura *in situ* em conservativa e calcular a pressão hidrostática, sendo estas, respetivamente, `gsw_SA_from_SP`, `gsw_CT_from_t` e `gsw_p_from_z`. Por último, com as três variáveis determinadas, basta utilizar a função `gsw_sound_speed` para calcular a velocidade do som.

4. REOART 2D

Como foi referido, o programa REOART 2D tem a funcionalidade de simular a propagação de raios acústicos em ambiente oceânico real. Este programa utiliza o modelo paramétrico 2D de Raposo e as bases de dados acabadas de mencionar.

O fornecimento das coordenadas geográficas, o tempo de propagação e o espaço de simulação, ao programa REOART 2D conduz à determinação dos valores da batimetria e da climatologia, e ao cálculo da velocidade do som e de todos os gradientes necessários, numa malha apropriada, para o uso do modelo paramétrico 2D.

A aplicação desenvolvida tem a funcionalidade de calcular sucessivas reflexões de raios acústicos no fundo e na superfície do oceano, para além de utilizar a batimetria e velocidade do som reais.

Na Figura 3 está representado o fluxograma do funcionamento lógico do programa REOART 2D e está dividido em quatro etapas. A primeira etapa consiste na introdução da posição geográfica, do azimute, da distância de propagação e da distância entre posições, sendo que isto permite definir o plano de simulação. Com a zona de simulação determinada (o plano 2D), com recurso às bases de dados da GEBCO e da WOD 18, a segunda etapa é a caracterização da topografia e da climatologia da zona de simulação. A terceira etapa, através da biblioteca de programas TEOS-10 (GSW), é o cálculo da velocidade de som e respetivos gradientes espaciais da zona de simulação. Por fim, a quarta etapa é onde é efetuada a resolução numérica do modelo de propagação de Raposo, que traduzido simboliza a trajetória do raio acústico, e a modelação das reflexões.

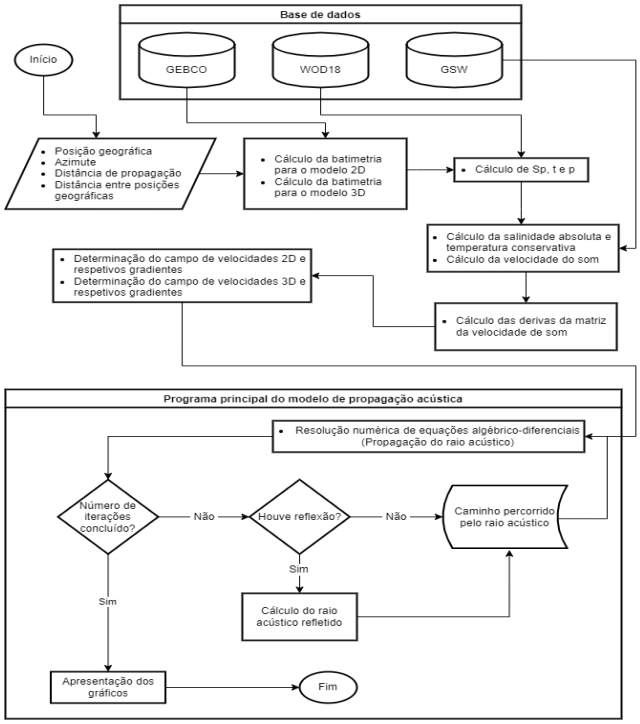


Figura 3 - Fluxograma do funcionamento lógico do programa REOART 2D.

5. SIMULAÇÃO NUMÉRICA

Por forma a testar o programa realizou-se uma simulação numérica com múltiplos raios acústicos na área envolvente ao Arquipélago da Região Autónoma dos Açores. A simulação foi realizada na posição geográfica de latitude 37º N e longitude 27º W ao longo do azimute 90º.

Na Figura 4, é possível observar a batimetria da área envolvente ao Arquipélago da Região Autónoma dos Açores, assim como a linha geodésica (a linha vermelha) que representa o plano de propagação.

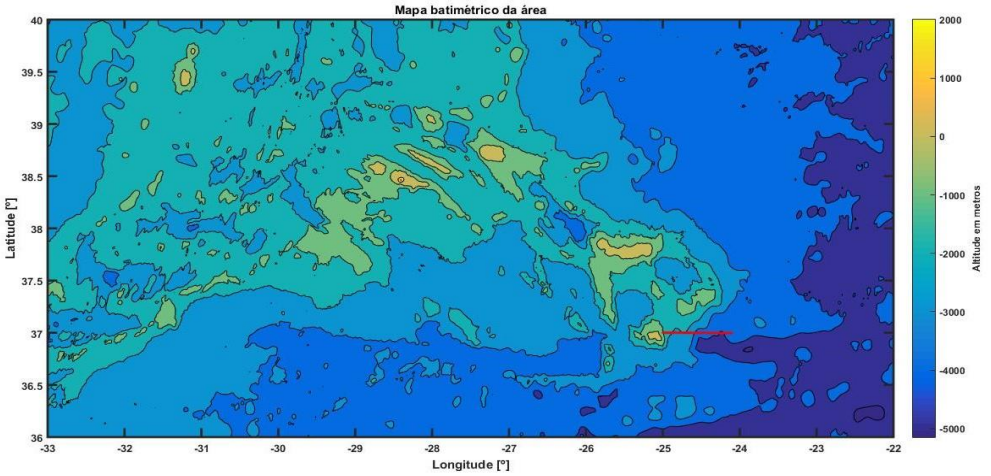


Figura 4 - Mapa batimétrico da área envolvente ao Arquipélago da Região Autónoma dos Açores com detalhe da representação do plano de propagação (a linha vermelha).

Tal como foi assinalado, o plano de propagação não se poderá estender para além dos oitenta quilómetros. Na Figura 5 está retratado o perfil batimétrico do plano de propagação.

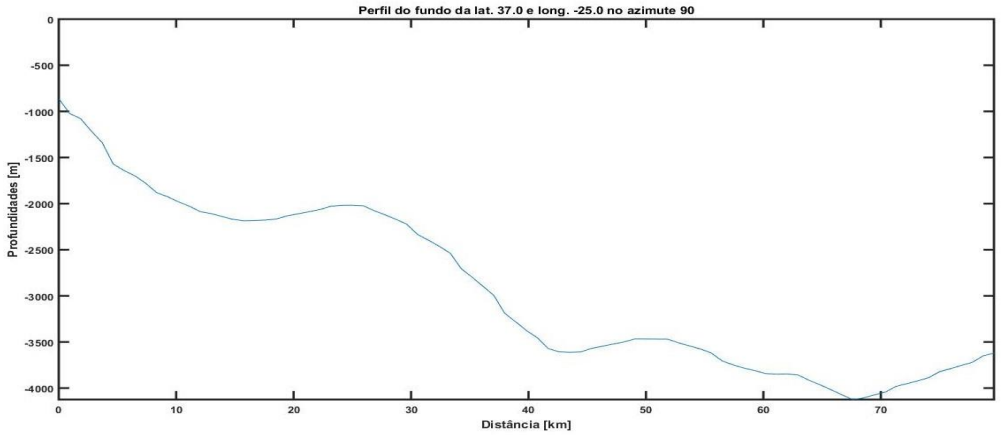


Figura 5 - Perfil batimétrico da posição geográfica de latitude 37º N e longitude 25º W, ao longo do azimute 90º, até à distância de oitenta quilómetros.

Tal como se procedeu relativamente à batimetria, associou-se ao plano de propagação os dados climatológicos, como se pode constatar na Figura 6. Nesta figura pode-se observar a necessidade de interpolar os pontos com dados climatológicos com o plano de propagação.

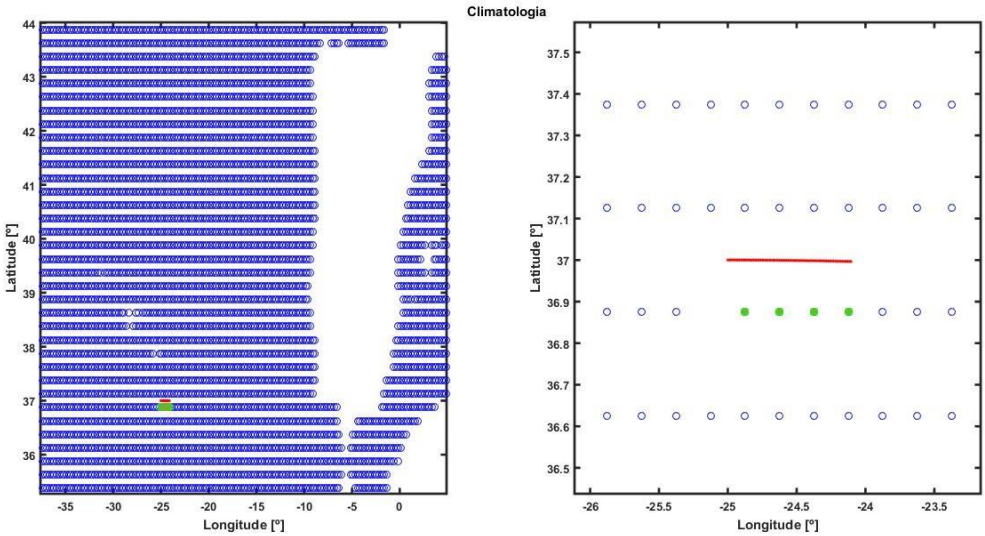


Figura 6 - Mapa com a distribuição dos pontos com dados climatológicos da área envolvente a Portugal Continental e ao Arquipélago da Região Autónoma dos Açores, o plano de propagação e os pontos mais próximos.

Após o plano de propagação estar definido, com a batimetria e a climatologia determinada, é então possível estabelecer o campo de velocidades de som recorrendo às funções TEOS-10.

Como foi mencionado, a simulação numérica realizou-se recorrendo a múltiplos raios acústicos na área envolvente ao Arquipélago da Região Autónoma dos Açores, na posição geográfica de latitude 37º N e longitude 27º W, ao longo do azimute 90º - ver Figura 7. Nesta figura pode-se observar a reflexão especular nas fronteiras e a refração ao longo da trajetória satisfazendo qualitativamente a lei de Snell. As trajetórias dos raios acústicos apresentam-se credíveis porque são compatíveis com o comportamento conhecido em ambientes oceânicos com estas características.

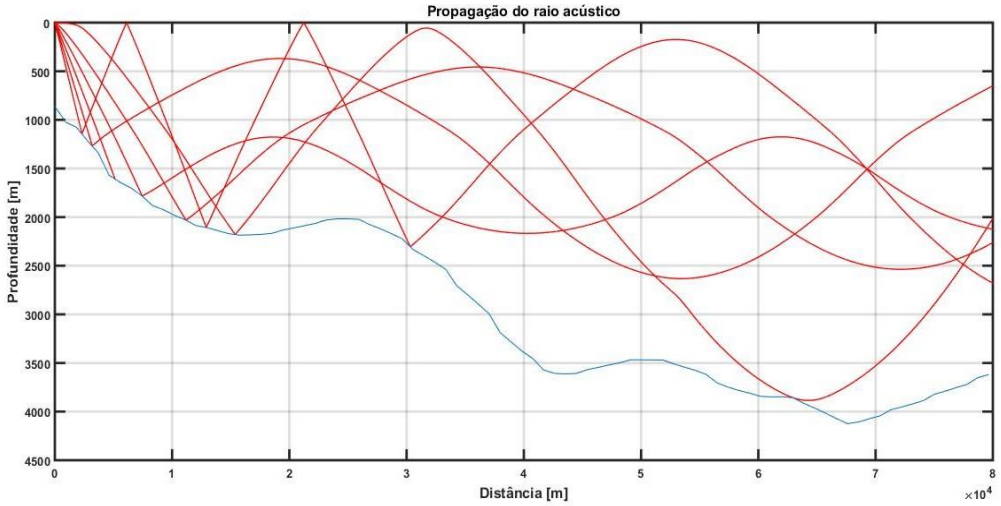


Figura 7 - Representação da simulação numérica com múltiplos raios, efetuada na posição geográfica de latitude 37º N e longitude 25º W, ao longo do azimute 90º.

6. CONCLUSÕES E TRABALHO FUTURO

O resultado da simulação apresentada é credível. A consideração de reflexões nos trajetos dos raios acústicos simulados pela ferramenta desenvolvida acrescenta realismo no fenómeno de propagação do som num ambiente real.

A disponibilidade duma aplicação 3D funcional para simulação do trajeto de raios acústicos em ambiente real pode ser importante e vantajoso em aplicações especializadas.

O resultado preliminar da simulação numérica sugere o desenvolvimento futuro das seguintes ações:

I. Validação do programa REOART 2D com aplicações disponíveis comercialmente e utilizadas na comunidade científica;

II. Otimização do programa REOART 2D e desenvolvimento de uma interface gráfica amigável para a sua utilização;
III. Fusão do REOART 2D com a ferramenta FAPSO;
IV. Adicionar a consideração do fenómeno de absorção acústica no modelo de propagação do programa REOART 2D.

AGRADECIMENTOS

Em primeiro lugar, gostaria de expressar o meu enorme apreço ao meu orientador de mestrado, Professor Miguel Moreira, pelas suas valiosas e construtivas sugestões durante o planeamento e desenvolvimento deste trabalho de investigação.

Aos meus coorientadores, Carlos Costa e Mário Gatta, gostaria também de estender o meu agradecimento por todo o conhecimento passado na área da oceanografia e pelo seu valioso apoio técnico neste trabalho.

REFERÊNCIAS BIBLIOGRÁFICAS

Daljit S. Ahluwalia and Joseph B. Keller (1997). *Wave propagation and underwater acoustics*. Springer.

Etter, P. C. (1995). *Underwater acoustic modeling: principles, techniques, and applications*. CRC Press.

Fernandes, J. B. (1967). *Manual de Hidrografia*. Instituto Hidrográfico.

Gonçalves, J. P. D. (2022). *Desenvolvimento de algoritmo para previsão 2D e 3D de propagação submarina de raios acústicos com reflexões – ferramentas REOART 2D e REOART 3D*. Escola Naval.

Marques, P. A. V. (2018). *Acústica submarina: Desenvolvimento de modelo de previsão de propagação acústica global*. Escola Naval.

Orlando Rodriguez, Sérgio de Jesus e Rogério M. Calazan (2021). *Fundamentos de acústica submarina*. Universidade do Algarve.

Raposo, P. A. M. A (2019). *Estudo da propagação acústica em 3D em meios não homogéneos –Raytracing baseado no princípio de Fermat*. Escola Naval.

Tiago C. Oliveira, Ying-Tsong Lin and Michael B. Porter (2021). *Underwater sound propagation modeling in a complex shallow water environment*. Frontiers in Marine Science, 1460-1464.

Apêndice G

Código do programa REOART 2D

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%-------------------------------------------------------------------------%
%                              REOART 2D                                  %
%-------------------------------------------------------------------------%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% REOART 2D (Real Environment Ocean-Acoustic Ray Tracing) is an
% ocean-acoustic ray tracing model with the ability to study sound
% propagation in a real ocean environment.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% In this script is where the sound speed and bathymetry are
 calculated
%-------------------------------------------------------------------------%
clearvars -global; close all; clc;
global Mvxy Mdx_vxy Mdy_vxy Mdx_dx_vxy Mdy_dx_vxy Mdy_dy_vxy...
    Mdx_dy_vxy dis_m depth

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%-------------------Initial geographic position----------------------
lat_pos=39;
long_pos=-10.1;
azimute = 270;
distance = 43.1965443;
inc_dis = 0.5;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%-------------------------------------------------------------------------%
%                            GEBCO DATABASE                               %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Selecting the area of the world. In this case is just the
% North Atlantic area (gebco=1).
gebco = 1;
if gebco==1
    longitude=ncread('GEBCO_AtlanticoNorte.nc','lon');
    latitude=ncread('GEBCO_AtlanticoNorte.nc','lat');
    elevation=ncread('GEBCO_AtlanticoNorte.nc','elevation');
end

area=5; % Change the number to change the simulation area
if area==1
    lat_S=34.5; lat_N=44.5; long_W=-15.5; long_E=-7.5; % North
elseif area==2                                          % Atlantic
    lat_S=30; lat_N=45; long_W=-40; long_E=-5; % Northeast Atlantic
elseif area==3
    lat_S=28.5; lat_N=36.5; long_W=-20.5; long_E=-13.5; % Madeira
elseif area==4
    lat_S=36; lat_N=40; long_W=-33; long_E=-22; % Açores
elseif area==5
    lat_S=34; lat_N=44; long_W=-15; long_E=-6; % Continental Portugal
elseif area==6
     lat_S=30; lat_N=43; long_W=-32; long_E=-6; % Portugal ZEE
```

```
elseif area==7
    lat_S=46; lat_N=61; long_W=-25; long_E=0; % Scotland
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%-------------------------------------------------------------------------%
%                              GEODESIC LINE                              %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[y_lat,x_long,dis_m,prof_area,long_area,lat_area,inc_long,...
    inc_lat]=
 Geodesic_Line(latitude,longitude,elevation,lat_S,lat_N,...
    long_W,long_E,lat_pos,long_pos,azimute, distance,inc_dis);

ind_x_long=round((x_long-min(longitude))./inc_long+1);
ind_y_lat=round((y_lat-min(latitude))./inc_lat+1);
perfil_fundo(1:length(ind_x_long))=0;
for i=1:length(ind_x_long)
perfil_fundo(i)=elevation(ind_x_long(i),ind_y_lat(i));
end
perfil_fundo;
dis_km=dis_m./1000;

% Posição geográfica com incrementos
posicao_geo = [y_lat;x_long];
posicao_geo=posicao_geo';

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%-------------------------------------------------------------------------%
%                            TEOS-10 FUNCTIONS                            %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%----- WOD18 Files
%loading the salinity and temperature values from WOD18 database
WOD18_T=csvread('woa18_95A4_t00an04.csv',2,0);
WOD18_S=csvread('woa18_95A4_s00an04.csv',2,0);

linhas_WOD18_T=size(WOD18_T,1);
colunas_WOD18_T=size(WOD18_T,2);
linhas_WOD18_S=size(WOD18_S,1);
colunas_WOD18_S=size(WOD18_S,2);

lat_T=WOD18_T(1:linhas_WOD18_T,1);
long_T=WOD18_T(1:linhas_WOD18_T,2);
T(1:linhas_WOD18_T,1:colunas_WOD18_T-2)=WOD18_T(1:linhas_WOD18_T,3:...
    colunas_WOD18_T);
lat_S=WOD18_S(1:linhas_WOD18_S,1);
long_S=WOD18_S(1:linhas_WOD18_S,2);
S(1:linhas_WOD18_S,1:colunas_WOD18_S-2)=WOD18_S(1:linhas_WOD18_S,3:...
    colunas_WOD18_S);

%Salinity and temperature matrices
if area==7 % For Scotland simulation area
    % Salinity
    posicao_geoS_1=[lat_S long_S];
    posicao_geoS=posicao_geoS_1(478830:585857,1:2);
```

```
    S=S(478830:585857,:);
    % Temperature
    posicao_geoT_1=[lat_T long_T];
    posicao_geoT=posicao_geoS;
    T=T(478830:585857,:);
else % All other simulation area
    % Salinity
    posicao_geoS_1=[lat_S long_S];
    posicao_geoS=posicao_geoS_1(478830:533094,1:2);
    S=S(478830:533094,:);
    % Temperature
    posicao_geoT_1=[lat_T long_T];
    posicao_geoT=posicao_geoS;
    T=T(478830:533094,:);
end

% Finding the nearest neighbors between data and geographic positions
Idx_S = knnsearch(posicao_geoS(:,1:2),posicao_geo(:,1:2));
Idx_T = knnsearch(posicao_geoT(:,1:2),posicao_geo(:,1:2));

% Construction of Salinity and Temperature matrices for the geographic
% position
for i=1:length(posicao_geo)
    Practical_Salinity(i,:)=S(Idx_S(i),:);
    Insitu_Temperature(i,:)=T(Idx_T(i),:);
end

% Depth values in meters
depth=[0,5,10,15,20,25,30,35,40,45,50,55,60,65,70,75,80,85,90,...
    95,100,125,150,175,200,225,250,275,300,325,350,375,400,425,...
    450,475,500,550,600,650,700,750,800,850,900,950,1000,1050,...
    1100,1150,1200,1250,1300,1350,1400,1450,1500,1550,1600,1650,...
    1700,1750,1800,1850,1900,1950,2000,2100,2200,2300,2400,2500,...
    2600,2700,2800,2900,3000,3100,3200,3300,3400,3500,3600,3700,...
    3800,3900,4000,4100,4200,4300,4400,4500,4600,4700,4800,4900,...
    5000,5100,5200,5300,5400,5500];

% Pressure calculation
for i=1:length(posicao_geo)
     p(i,1:length(depth)) = gsw_p_from_z(-depth,posicao_geo(i,1));
end

% Absolute salinity calculation
SA = gsw_SA_from_SP(Practical_Salinity,p,posicao_geo(:,2),...
    posicao_geo(:,1));

% Conservative temperature calculation
CT = gsw_CT_from_t(SA,Insitu_Temperature,p);

% Sound speed calculation
sound_speed = gsw_sound_speed(SA,CT,p);

% Sound speed matrix
Mvxy =sound_speed';
```

```
%-------------------------------------------------------------------%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%         Calculating spatial gradients of the sound speed          %
%-------------------------------------------------------------------%
% The derivatives: dx_vxy dy_vxy dx_dx_vxy dy_dx_vxy dx_dy_vxy
%                  dy_dy_vxy
% x - distance
% y - depth

%-----Centered finite-difference formulas----
% First Derivative: f'(xi)= -f(xi+2)+8f(xi+1)-8f(xi-1)+f(xi-2)/12h
% Second Derivative: f''(xi)= -f(xi+2)+16f(xi+1)-30f(xi)+16f(xi-1)-...
%                             f(xi-2)/12h^2
% h is the increment used between geographic positions

%-----Backward finite-difference formulas----
% First Derivative: f'(xi)= 3f(xi)-4f(xi-1)+f(xi-2)/2h
% Second Derivative: f''(xi)= 2f(xi)-5f(xi-1)+4f(xi-2)-f(xi-3)/h^2

%-----Forward finite-difference formulas-----
% First Derivative: f'(xi)= -f(xi+2)+4f(xi+1)-3f(xi)/2h
% Second Derivative: f''(xi)= -f(xi+3)+4f(xi+2)-5f(xi+1)+2f(xi)/h^2

%-------------------------------------------------------------------%
%%%%%%%%%%$%%%%%%%%%%%%%% First Derivatives %%%%%%%%%%%%%%%%%%%%%%%%%
% dx - variation along the lines
% dy - variation along the columns

depth_2=[0,5,10,15,20,25,30,35,40,45,50,55,60,65,70,75,80,85,90,...
    95,100,125,150,175,200,225,250,275,300,325,350,375,400,425,...
    450,475,500,550,600,650,700,750,800,850,900,950,1000,1050,...
    1100,1150,1200,1250,1300,1350,1400,1450,1500,1550,1600,1650,...
    1700,1750,1800,1850,1900,1950,2000,2100,2200,2300,2400,2500,...
    2600,2700,2800,2900,3000,3100,3200,3300,3400,3500,3600,3700,...
    3800,3900,4000,4100,4200,4300,4400,4500,4600,4700,4800,4900,...
    5000,5100,5200,5300,5400,5500,5600]; %acrecentei 5600 metros
depth_2=depth_2';

%--------- dx_vxy calculation (horizontal variation)----------
%h=diff(dis_m);
hx=inc_dis*1852;
for i = 1:length(Mvxy);
    for ii = 1:length(posicao_geo);
        if ii==1 % Forward finite-difference method
            Mdx_vxy(i,ii)= (-Mvxy(i,ii+2)+4.*Mvxy(i,ii+1)-3.*...
                Mvxy(i,ii))/(2*hx);
        elseif ii==2
            Mdx_vxy(i,ii)= (-Mvxy(i,ii+2)+4.*Mvxy(i,ii+1)-3.*...
                Mvxy(i,ii))/(2*hx);
        elseif ii==length(posicao_geo)% Backward finite-difference
 method
            Mdx_vxy(i,ii)= (3.*Mvxy(i,ii)-4.*Mvxy(i,ii-1)+...
                Mvxy(i,ii-2))/(2*hx);
```

```
        elseif ii==(length(posicao_geo)-1)
            Mdx_vxy(i,ii)= (3.*Mvxy(i,ii)-4.*Mvxy(i,ii-1)+...
                Mvxy(i,ii-2))/(2*hx);
        else % Centered finite-difference method
            Mdx_vxy(i,ii)= (-Mvxy(i,ii+2)+8.*Mvxy(i,ii+1)-8.*...
                Mvxy(i,ii-1)+Mvxy(i,ii-2))/(12*hx);
        end
    end

end


%--------- dy_vxy calculation (vertical variation)------------
hy=diff(depth_2);
for i = 1:length(Mvxy);
    for ii = 1:length(posicao_geo);
        if i==1 % Forward finite-difference method
            Mdy_vxy(i,ii)= (-Mvxy(i+2,ii)+4.*Mvxy(i+1,ii)-3.*...
                Mvxy(i,ii))/(2*hy(i,1));
        elseif i==2
            Mdy_vxy(i,ii)= (-Mvxy(i+2,ii)+4.*Mvxy(i+1,ii)-3.*...
                Mvxy(i,ii))/(2*hy(i,1));
        elseif i==length(Mvxy)% Backward finite-difference method
            Mdy_vxy(i,ii)= (3.*Mvxy(i,ii)-4.*Mvxy(i-1,ii)+...
                Mvxy(i-2,ii))/(2*hy(i,1));
        elseif i==(length(Mvxy)-1)
            Mdy_vxy(i,ii)= (3.*Mvxy(i,ii)-4.*Mvxy(i-1,ii)+...
            Mvxy(i-2,ii))/(2*hy(i,1));
        else % Centered finite-difference method
            Mdy_vxy(i,ii)= (-Mvxy(i+2,ii)+8.*Mvxy(i+1,ii)-8.*...
                Mvxy(i-1,ii)+Mvxy(i-2,ii))/(12*hy(i,1));
        end
    end

end


%-------------------------------------------------------------------%
%%%%%%%%%%%%%%%%%%%%%%%% Second Derivatives %%%%%%%%%%%%%%%%%%%%%%%%
% The derivatives: dx_dx_vxy dy_dx_vxy dx_dy_vxy dy_dy_vxy
%-------------------------------------------------------------------%

%--------- dx_dx_vxy calculation ---------
for i = 1:length(Mdx_vxy);
    for ii = 1:length(posicao_geo);
        if ii==1 % Forward finite-difference method
            Mdx_dx_vxy(i,ii)= (-Mdx_vxy(i,ii+2)+4.*Mdx_vxy(i,ii+1)...
                -3.*Mdx_vxy(i,ii))/(2*hx);
        elseif ii==2
            Mdx_dx_vxy(i,ii)= (-Mdx_vxy(i,ii+2)+4.*Mdx_vxy(i,ii+1)...
                -3.*Mdx_vxy(i,ii))/(2*hx);
        elseif ii==length(posicao_geo)% Backward finite-difference
 method
            Mdx_dx_vxy(i,ii)= (3.*Mdx_vxy(i,ii)-4.*Mdx_vxy(i,ii-1)+...
```

```
                Mdx_vxy(i,ii-2))/(2*hx);
        elseif ii==(length(posicao_geo)-1)
            Mdx_dx_vxy(i,ii)= (3.*Mdx_vxy(i,ii)-4.*Mdx_vxy(i,ii-1)+...
                Mdx_vxy(i,ii-2))/(2*hx);
        else % Centered finite-difference method
            Mdx_dx_vxy(i,ii)= (-Mdx_vxy(i,ii+2)+8.*Mdx_vxy(i,ii+1)...
                -8.*Mdx_vxy(i,ii-1)+Mdx_vxy(i,ii-2))/(12*hx);
        end
    end

end

%--------- dy_dx_vxy calculation ---------
for i = 1:length(Mdx_vxy);
    for ii = 1:length(posicao_geo);
        if i==1 % Forward finite-difference method
            Mdy_dx_vxy(i,ii)= (-Mdx_vxy(i+2,ii)+4.*Mdx_vxy(i+1,ii)...
                -3.*Mdx_vxy(i,ii))/(2*hy(i,1));
        elseif i==2
            Mdy_dx_vxy(i,ii)= (-Mdx_vxy(i+2,ii)+4.*Mdx_vxy(i+1,ii)...
                -3.*Mdx_vxy(i,ii))/(2*hy(i,1));
        elseif i==length(Mdx_vxy)% Backward finite-difference method
            Mdy_dx_vxy(i,ii)= (3.*Mdx_vxy(i,ii)-4.*Mdx_vxy(i-1,ii)+...
                Mdx_vxy(i-2,ii))/(2*hy(i,1));
        elseif i==(length(Mdx_vxy)-1)
            Mdy_dx_vxy(i,ii)= (3.*Mdx_vxy(i,ii)-4.*Mdx_vxy(i-1,ii)+...
                Mdx_vxy(i-2,ii))/(2*hy(i,1));
        else % Centered finite-difference method
            Mdy_dx_vxy(i,ii)= (-Mdx_vxy(i+2,ii)+8.*Mdx_vxy(i+1,ii)...
                -8.*Mdx_vxy(i-1,ii)+Mdx_vxy(i-2,ii))/(12*hy(i,1));
        end
    end

end

%--------- dx_dy_vxy calculation---------
for i = 1:length(Mdy_vxy);
    for ii = 1:length(posicao_geo);
        if ii==1 % Forward finite-difference method
            Mdx_dy_vxy(i,ii)= (-Mdy_vxy(i,ii+2)+4.*Mdy_vxy(i,ii+1)...
                -3.*Mdy_vxy(i,ii))/(2*hx);
        elseif ii==2
            Mdx_dy_vxy(i,ii)= (-Mdy_vxy(i,ii+2)+4.*Mdy_vxy(i,ii+1)...
                -3.*Mdy_vxy(i,ii))/(2*hx);
        elseif ii==length(posicao_geo)% Backward finite-difference
 method
            Mdx_dy_vxy(i,ii)= (3.*Mdy_vxy(i,ii)-4.*Mdy_vxy(i,ii-1)+...
                Mdy_vxy(i,ii-2))/(2*hx);
        elseif ii==(length(posicao_geo)-1)
            Mdx_dy_vxy(i,ii)= (3.*Mdy_vxy(i,ii)-4.*Mdy_vxy(i,ii-1)+...
                Mdy_vxy(i,ii-2))/(2*hx);
        else % Centered finite-difference method
            Mdx_dy_vxy(i,ii)= (-Mdy_vxy(i,ii+2)+8.*Mdy_vxy(i,ii+1)...
                -8.*Mdy_vxy(i,ii-1)+Mdy_vxy(i,ii-2))/(12*hx);
```

```
        end
    end

end

%--------- dy_dy_vxy calculation ---------
for i = 1:length(Mdy_vxy);
    for ii = 1:length(posicao_geo);
        if i==1 % Forward finite-difference method
            Mdy_dy_vxy(i,ii)= (-Mdy_vxy(i+2,ii)+4.*Mdy_vxy(i+1,ii)...
                -3.*Mdy_vxy(i,ii))/(2*hy(i,1));
        elseif i==2
            Mdy_dy_vxy(i,ii)= (-Mdy_vxy(i+2,ii)+4.*Mdy_vxy(i+1,ii)...
                -3.*Mdy_vxy(i,ii))/(2*hy(i,1));
        elseif i==length(Mdy_vxy)% Backward finite-difference method
            Mdy_dy_vxy(i,ii)= (3.*Mdy_vxy(i,ii)-4.*Mdy_vxy(i-1,ii)+...
                Mdy_vxy(i-2,ii))/(2*hy(i,1));
        elseif i==(length(Mdy_vxy)-1)
            Mdy_dy_vxy(i,ii)= (3.*Mdy_vxy(i,ii)-4.*Mdy_vxy(i-1,ii)+...
                Mdy_vxy(i-2,ii))/(2*hy(i,1));
        else % Centered finite-difference method
            Mdy_dy_vxy(i,ii)= (-Mdy_vxy(i+2,ii)+8.*Mdy_vxy(i+1,ii)...
                -8.*Mdy_vxy(i-1,ii)+Mdy_vxy(i-2,ii))/(12*hy(i,1));
        end
    end

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Priting the graphics
disp('1 to print')
disp('0 to continue')
graphics=input('Do you wish to print the graphics: ');

if graphics ==1
    clear colorbar
    figure % Bathymetry Map
    contourf(long_area,lat_area,prof_area')
    title ('Bathymetric map of the simulation area')
    xlabel ('Longitude [º]')
    ylabel ('Latitude [º]')
    hold on
    plot(posicao_geo(:,2),posicao_geo(:,1),'r.-')
    hold on
    plot(posicao_geo(1,2),posicao_geo(1,1),'o-r')
    colorbar;
    colorbar=colorbar;
    colorbar.Label.String = 'Altitude in meters';

 set(gca,'Fontsize',10,'FontWeight','bold','LineWidth',2.0,'box','on')

    figure % 2D Bathymetry
    plot(dis_km,perfil_fundo)
    hold on
```

```
    plot(dis_km(1),perfil_fundo(1),'o-r')
    axis([0 max(dis_km) min(perfil_fundo) 0])
    xlabel('Range [km]')
    ylabel('Depth [m]')
    title(sprintf ...
     ('Bathymetry from lat. %0.1f and long. %0.1f on azimuth
%0.0f', ...
     lat_pos,long_pos,azimute))

set(gca,'Fontsize',10,'FontWeight','bold','LineWidth',2.0,'box','on');

    figure % 3D Bathymetry
    mesh(long_area,lat_area,prof_area') % Gráfico 3D
    title ('3D bathymetry map of the simulation area')
    xlabel ('Longitude [º]')
    ylabel ('Latitude [º]')

set(gca,'Fontsize',10,'FontWeight','bold','LineWidth',2.0,'box','on')

    figure % Climatology - nearest neighbors
    scatter(posicao_geoS(:,2),posicao_geoS(:,1),'b');
    hold on
    scatter(posicao_geoT(:,2),posicao_geoT(:,1),'b');
    hold on
    scatter(posicao_geo(:,2),posicao_geo(:,1),'r','.','LineWidth',2);
    hold on
    scatter(posicao_geoS(Idx_S,2),posicao_geoS(Idx_S,1),'g','s',...
        'LineWidth',2)
    hold on

scatter(posicao_geoT(Idx_T,2),posicao_geoT(Idx_T,1),'+','LineWidth',2)
    title('Map with the distribution of points with the climatological
data')
    xlabel ('Longitude [º]')
    ylabel ('Latitude [º]')

set(gca,'Fontsize',10,'FontWeight','bold','LineWidth',2.0,'box','on')
    disp(' ')
    disp(' ')
    disp(' ')
    disp('Graphics printed')
    disp('End')
elseif graphics ==0
    disp('End')
elseif isempty(graphics)==1
    disp('1 to print')
    disp('0 to continue')
    graphics=input('Do you wish to print the graphics: ');
else
    disp('1 to print')
    disp('0 to continue')
    graphics=input('Do you wish to print the graphics: ');
end
```

Published with MATLAB® R2015a

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%-------------------------------------------------------------------------%
%                     REOART 2D - Ray Propagation                         %
%-------------------------------------------------------------------------%
% This is the REOART 2D main script where the propagation of the
% acoutic ray is calculated
%-------------------------------------------------------------------------%
%
clc
Vmatrix=[]; % ray trajectory matrix
Tsmatrix=[]; % propagation time matrix

% Initial conditions (change as you intend)
delta=10;
ts=0:delta:1000*delta;
theta0=0; %Angle in degrees 0<theta0<90
theta0=pi*theta0/180;
x0=3500; % Horizontal position
y0=2500; % Vertical position
z0=cos(theta0);
w0=sin(theta0);
v0=[x0;y0;z0;w0] % Ray initial conditions

for il=1:5
    v=dae4('pa2dr',ts,v0,10);
    Yq=interp1(dis_m,-perfil_fundo,v(1,:));
    ctemp1=min(find(v(2,2:end)<=1));
    ctemp2=min(find(v(2,2:end)>Yq(1,2:end)));

    if isempty(ctemp1)==1 && isempty(ctemp2)==1 % Without reflection
        disp('No reflection')
        ts=ts+ts(end);
        Vmatrix=[Vmatrix v(:,1:end-1)];
        Tsmatrix=[Tsmatrix ts(1,1:end-1)];
        v0=v(:,end);

    elseif isempty(ctemp1)~=1 && isempty(ctemp2)==1 % Surface
 reflection
        disp('Surface reflection')
        Vmatrix=[Vmatrix v(:,1:ctemp1)];
        Tsmatrix=[Tsmatrix ts(1,1:ctemp1)];
        v0=Vmatrix(:,end); % New initial conditions
        [Rn,i,r] = sup_reflection2D(v,ctemp1)
        v0(4)=r(2,1);
        ts=ts+Tsmatrix(end);
    elseif isempty(ctemp1)==1 && isempty(ctemp2)~=1 % Bottom
 reflection
        disp('Bottom reflection')
        Vmatrix=[Vmatrix v(:,1:ctemp2)];
        Tsmatrix=[Tsmatrix ts(1,1:ctemp2)];
        N=points';
        newpoint=[v(1,ctemp2),v(2,ctemp2)];
```

```
            [n,d] = knnsearch(N,newpoint,'k',3);
            n1=unique(n);
            N_closest = N(n1,:);
            [normal_R] = normal_ponto_reflexao_2D(normals',n1)
            [Rn,i,r] = reflection2D(normal_R,v,ctemp2)
            v0=Vmatrix(1:2,end); % New initial conditions
            v0(3,1)=r(1,1);
            v0(4,1)=r(2,1);
            ts=ts+Tsmatrix(end);
        end
    end

    figure
    plot(Vmatrix(1,:),Vmatrix(2,:),'r','LineWidth',1)
    hold on
    plot(dis_m,-perfil_fundo)
    axis ij
    grid on
    set(gca,'Fontsize',10,'FontWeight','bold','LineWidth',2.0,'box','on');
    a1=xlabel('Range [m]');
    a2=ylabel('Depth [m]');
    a3=title('Acoustic ray propagation');
    set(a1,'Fontsize',12,'FontWeight','bold');
    set(a2,'Fontsize',12,'FontWeight','bold')
    set(a3,'Fontsize',12,'FontWeight','bold')
```

Published with MATLAB® R2015a

```
function [y_lat,x_long,dis_m,prof_area,long_area,lat_area,...
    inc_long,inc_lat]= Geodesic_Line(latitude,longitude,...
    elevation,lat_S,lat_N,long_W,long_E,lat_pos,long_pos,...
    azimute, distancia,inc_dis)
%=========================================================================%
% This function calculates the geodesic line.
% The output are the geographical positions that compose the geodesic
% line.
%=========================================================================%


inc_long=30/3600;
inc_lat=inc_long;
prof_max=min(min(elevation));
prof_min=max(max(elevation));
ind_long=round((long_pos-min(longitude))/inc_long+1);
ind_lat=round((lat_pos-min(latitude))/inc_lat+1);
prof_pos=elevation(ind_long,ind_lat);
ind_long_W=round((long_W-min(longitude))/inc_long+1);
ind_long_E=round((long_E-min(longitude))/inc_long+1);
ind_lat_S=round((lat_S-min(latitude))/inc_lat+1);
ind_lat_N=round((lat_N-min(latitude))/inc_lat+1);
prof_area=elevation(ind_long_W:ind_long_E,ind_lat_S:ind_lat_N);
size(prof_area);
max_prof_area=min(min(prof_area));
min_prof_area=max(max(prof_area));
long_area=long_W:inc_long:long_E;
size(long_area);
lat_area=lat_S:inc_lat:lat_N;
size(lat_area);
%-------------------------------------------------------------------------%
distancia_m=distancia*1852;
inc_dis_m=inc_dis*1852;
dis_m=0:inc_dis_m:distancia_m;
long_pos=-long_pos;
s=dis_m;
exc=0.0819919;
a_eli=6378388;
if azimute<=180
    azimute_S=180+azimute;
else
    azimute_S=azimute-180;
end
R_mer=(a_eli*(1-exc^2))/((1-exc^2*(sind(lat_pos))^2)^(3/2));
N=a_eli/((1-exc^2*sind(lat_pos)^2)^(1/2));
B=1/(R_mer*sind(1/3600));
C=tand(lat_pos)/(2*R_mer*N*sind(1/3600));
D=(3*exc^2*sind(lat_pos)*cosd(lat_pos)*sind(1/3600))/(2*(1-exc^2*...
    (sind(lat_pos))^2));
E=(1+3*(tand(lat_pos))^2)/(6*N^2);
del_fi=-s.*cosd(azimute_S).*B-s.^2.*(sind(azimute_S)).^2.*C;
h=s.*cosd(azimute_S).*B;
```

```
delta_fi_2linha=-s.*cosd(azimute_S).*B-s.^2.*(sind(azimute_S)).^2.*...
    C-del_fi.^2.*D+h.*s.^2.*(sind(azimute_S)).^2.*E;
delta_fi=delta_fi_2linha./3600;
lat_2=lat_pos+delta_fi;
N_linha=a_eli./((1-exc.^2.*sind(lat_2).^2).^(1/2));
A_linha=1./(N_linha.*sind(1./3600));
log_delta_lambda_2linha=log10(s)+log10(abs(sind(azimute_S)))+...
    log10(abs(A_linha))+log10(abs(secd(lat_2)));
delta_long=10.^log_delta_lambda_2linha./3600;
if azimute_S <= 180
    long_2=long_pos+delta_long;
else
    long_2=long_pos-delta_long;
end
y_lat=lat_2;
x_long=-long_2;
end
```

Published with MATLAB® R2015a

```
function [f,k,m]=pa2dr(t,v,vd)
%=========================================================================%
% t - time
% v = [x;y;z;w]
% xd - derivative of x
% yd - derivative of y
% zd - derivative of z
% wd - derivative of w
% We suppose: xd^2+yd^2=1
%=========================================================================%
x=v(1); y=v(2); z=v(3); w=v(4);
%
xd=vd(1); yd=vd(2); zd=vd(3); wd=vd(4);

%
[vxy dx_vxy dy_vxy dx_dx_vxy dy_dx_vxy dx_dy_vxy dy_dy_vxy] =
 cv2dr(v);
% vxy- sound velocity field

%
k_31=(w*dx_dx_vxy-z*dx_dy_vxy)/vxy-(w*dx_vxy-z*dy_vxy)*dx_vxy/vxy^2;
k_32=(w*dy_dx_vxy-z*dy_dy_vxy)/vxy-(w*dx_vxy-z*dy_vxy)*dy_vxy/vxy^2;
k_33=-wd-dy_vxy/vxy;
k_34=zd+dx_vxy/vxy;

k_41=0;
k_42=0;
k_43=zd;
k_44=wd;

% System of differential equations without equal sign and all reduced
% to the left hand side
f=[xd-z;
   yd-w;
   zd*w-wd*z+(w*dx_vxy-z*dy_vxy)/vxy;
  % zd*z+wd*w-vxy*(dx_vxy*z+dy_vxy*w)];
   zd*z+wd*w];

% Jacobian with respect to vector v: x,y,z,w
k=[0,0,-1,0;
   0,0,0,-1;
    k_31,k_32,k_33,k_34;
    k_41,k_42,k_43,k_44];

%Jacobian with respect to vector vd: xd, yd, zd, wd
m=[1,0,0,0;
   0,1,0,0;
   0,0,w,-z;
   0,0,z,w];
end
```

Published with MATLAB® R2015a

```
function [vxy dx_vxy dy_vxy dx_dx_vxy dy_dx_vxy dx_dy_vxy...
    dy_dy_vxy] = cv2dr(v);
%=========================================================================%
% This function interpolates the sound speed and the spatial
% gradients with the position of the acoustic ray. This allows to
% determinate the sound speed along the acoustic ray trajectory.
%
% The global variables are the sound speed and spatial gradients
%  matrices, distance and depth of the 2D simulation plan.
%=========================================================================%
global Mvxy Mdx_vxy Mdy_vxy Mdx_dx_vxy Mdy_dx_vxy Mdy_dy_vxy...
    Mdx_dy_vxy dis_m depth

x=v(1); y=v(2); z=v(3); w=v(4);

[X,Y]=meshgrid(dis_m,depth);

vxy =interp2(X,Y,Mvxy,x,y,'linear',1500);

dx_vxy=interp2(X,Y,Mdx_vxy,x,y,'linear',0.005);

dy_vxy=interp2(X,Y,Mdy_vxy,x,y,'linear',0.0005);

dx_dx_vxy=interp2(X,Y,Mdx_dx_vxy,x,y,'linear',0.0005);

dy_dx_vxy=interp2(X,Y,Mdy_dx_vxy,x,y,'linear',0.0005);

dy_dy_vxy=interp2(X,Y,Mdy_dy_vxy,x,y,'linear',0.0005);

dx_dy_vxy=interp2(X,Y,Mdx_dy_vxy,x,y,'linear',0.0005);
end
```

Published with MATLAB® R2015a

```
function [normal_R] = normal_ponto_reflexao_2D(normals,n1)
%=========================================================================%
% calculation of the normal vector at the surface at the
% reflection point
%=========================================================================%

normal_R=[normals(n1(1,1),1)+normals(n1(1,2),1)/length(n1);...
    normals(n1(1,1),2)+normals(n1(1,2),2)/length(n1)];

normal_R=(normal_R/norm(normal_R));
normal_R(2,1)=-normal_R(2,1);
end
```

Published with MATLAB® R2015a

```
function [Rn,i,r] = reflection2D(normal_R,v,ctemp2)
%=========================================================================%
% This function calculates the reflection of the acoustic ray in
% the bottom of the ocean
%  Rn = normal vector to the plane at the point of reflection(Matrix)
%  i = incident acoustic ray
%  r = reflected acoustic ray
%=========================================================================%

Rn=[1-2*(normal_R(1,1)^2),-2*normal_R(1,1)*normal_R(2,1);...
    -2*normal_R(1,1)*normal_R(2,1),1-2*(normal_R(2,1)^2)];
Rn=Rn/norm(Rn);

i=[v(3,ctemp2);v(4,ctemp2)];
i=i/norm(i);

r=Rn*i;
r=r/norm(r);
end
```

Published with MATLAB® R2015a

```
function [Rn,i,r] = sup_reflection2D(v,ctemp1)
%=========================================================================%
% This function calculates the reflection of the acoustic ray in
% the ocean surface
%  Rn = normal vector to the plane at the point of reflection(Matrix)
%  i = incident acoustic ray
%  r = reflected acoustic ray
%=========================================================================%

Rn=[-1,0;...
    0,-1];
Rn=Rn/norm(Rn);

i=[v(3,ctemp1);v(4,ctemp1)];
i=i/norm(i);

r=Rn*i;
r=r/norm(r);
end
```

Published with MATLAB® R2015a

Apêndice H

Código do programa REOART 3D

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%-----------------------------------------------------------------------%
%                              REOART 3D
%-----------------------------------------------------------------------%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% REOART 2D (Real Environment Ocean-Acoustic Ray Tracing) is an
% ocean-acoustic ray tracing model with the ability to study sound
% propagation in a real ocean environment.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% In this script is where the sound speed and bathymetry are
% calculated
%-----------------------------------------------------------------------%
clearvars -global; close all; clear; clc;
global Mvxyz Mdx_vxyz Mdy_vxyz Mdz_vxyz Mdx_dx_vxyz Mdx_dy_vxyz...
   Mdx_dz_vxyz Mdy_dx_vxyz Mdy_dy_vxyz Mdy_dz_vxyz Mdz_dx_vxyz...
   Mdz_dy_vxyz Mdz_dz_vxyz X_2 Y_2 Lat_3D Long_3D X_m Y_m...
   depth Depth_3D

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%-------------------Initial geographic position---------------------%
lat_pos=39;
long_pos=-9;
azimute = 270;
azimute_1 = azimute+90;
distance = 50.8;
inc_dis = 0.5;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%-----------------------------------------------------------------------%
%                           GEBCO DATABASE                              %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Selecting the area of the world.
% In this case is just the North Atlantic
% area (gebco=1).
gebco = 1;
if gebco==1
    longitude=ncread('GEBCO_AtlanticoNorte.nc','lon');
    latitude=ncread('GEBCO_AtlanticoNorte.nc','lat');
    elevation=ncread('GEBCO_AtlanticoNorte.nc','elevation');
end

area=5; % Change the number to change the simulation area
if area==1
    lat_S=34.5; lat_N=44.5; long_W=-15.5; long_E=-7.5; % North
elseif area==2                                         % Atlantic
    lat_S=30; lat_N=45; long_W=-40; long_E=-5; % Northeast
elseif area==3                                 % Atlantic
    lat_S=28.5; lat_N=36.5; long_W=-20.5; long_E=-13.5; % Madeira
elseif area==4
    lat_S=36; lat_N=40; long_W=-33; long_E=-22; % Açores
```

```
elseif area==5
    lat_S=34; lat_N=44; long_W=-15; long_E=-6; % Continental Portugal
elseif area==6
     lat_S=30; lat_N=43; long_W=-32; long_E=-6; % Portugal ZEE
elseif area==7
    lat_S=46; lat_N=61; long_W=-25; long_E=0; % Scotland
end

% Central position geodesic
[y_lat,x_long,dis_m]=Geodesica_GEBCO(lat_pos,long_pos,azimute,...
    distance,inc_dis);

% position 1 geodesic
[y_lat_1,x_long_1,dis_m_1]=Geodesica_GEBCO(lat_pos,long_pos,azimute_1,...
    distance,inc_dis);

% Central position
posicao_geo = [x_long;y_lat];
posicao_geo=posicao_geo';

% Position 1
posicao_geo_1 = [x_long_1;y_lat_1];
posicao_geo_1=posicao_geo_1';

[X]=meshgrid(posicao_geo(:,1),posicao_geo_1(:,1) ); % Longitude
[Y]=meshgrid(posicao_geo_1(:,2),posicao_geo(:,2)); % Latitude
X=X';

% Positions in meters
[X_m]=meshgrid(dis_m(1,:),dis_m(1,:));
[Y_m]=meshgrid(dis_m_1(1,:),dis_m_1(1,:));
X_m=X_m';

X_2=X(:,1); %Mx1 matrix - Longitude
Y_2=Y(1,:); %Mx1 matrix - Latitude

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%-------------------------------------------------------------------------%
%                           GEODESIC LINE                                 %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[y_lat,x_long,dis_m,prof_area,long_area,lat_area,inc_long,...
    inc_lat]=
 Geodesic_Line(latitude,longitude,elevation,lat_S,lat_N,...
    long_W,long_E,lat_pos,long_pos,azimute, distance,inc_dis);

ind_x_long=round((x_long-min(longitude))./inc_long+1);
ind_y_lat=round((y_lat-min(latitude))./inc_lat+1);
perfil_fundo(1:length(ind_x_long))=0;
for i=1:length(ind_x_long)
perfil_fundo(i)=elevation(ind_x_long(i),ind_y_lat(i));
end
perfil_fundo;
dis_km=dis_m./1000;
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%-------------------------------------------------------------------------%
%                            TEOS-10 FUNCTIONS                            %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%----- WOD18 Files
%loading the salinity and temperature values from WOD18 database
WOD18_T=csvread('woa18_95A4_t00an04.csv',2,0);
WOD18_S=csvread('woa18_95A4_s00an04.csv',2,0);

linhas_WOD18_T=size(WOD18_T,1);
colunas_WOD18_T=size(WOD18_T,2);
linhas_WOD18_S=size(WOD18_S,1);
colunas_WOD18_S=size(WOD18_S,2);

lat_T=WOD18_T(1:linhas_WOD18_T,1);
long_T=WOD18_T(1:linhas_WOD18_T,2);
T(1:linhas_WOD18_T,1:colunas_WOD18_T-2)=WOD18_T(1:linhas_WOD18_T,3:...
    colunas_WOD18_T);
lat_S=WOD18_S(1:linhas_WOD18_S,1);
long_S=WOD18_S(1:linhas_WOD18_S,2);
S(1:linhas_WOD18_S,1:colunas_WOD18_S-2)=WOD18_S(1:linhas_WOD18_S,3:...
    colunas_WOD18_S);

%Salinity and temperature matrices
if area==7 % For Scotland simulation area
    % Salinity
    posicao_geoS_1=[lat_S long_S];
    posicao_geoS=posicao_geoS_1(478830:585857,1:2);
    S=S(478830:585857,:);
    % Temperature
    posicao_geoT_1=[lat_T long_T];
    posicao_geoT=posicao_geoS;
    T=T(478830:585857,:);
 else % All other simulation area
    % Salinity
    posicao_geoS_1=[lat_S long_S];
    posicao_geoS=posicao_geoS_1(478830:533094,1:2);
    S=S(478830:533094,:);
    % Temperature
    posicao_geoT_1=[lat_T long_T];
    posicao_geoT=posicao_geoS;
    T=T(478830:533094,:);
end

X_1=reshape(X,1,[]); %transform matrix X into a matrix with 1 column
Y_1=reshape(Y,1,[]); %transform matrix Y into a matrix with 1 column

posicao_geo3D = [Y_1;X_1];
posicao_geo3D = posicao_geo3D';

% Finding the nearest neighbors between data and geographic positions
Idx_S = knnsearch(posicao_geoS(:,1:2),posicao_geo3D(:,1:2));
Idx_T = knnsearch(posicao_geoT(:,1:2),posicao_geo3D(:,1:2));
```

```
Idx_S = reshape(Idx_S,length(X),[]);
Idx_T = reshape(Idx_T,length(X),[]);

% Depth values in meters
depth=[0,5,10,15,20,25,30,35,40,45,50,55,60,65,70,75,80,85,90,...
    95,100,125,150,175,200,225,250,275,300,325,350,375,400,425,...
    450,475,500,550,600,650,700,750,800,850,900,950,1000,1050,...
    1100,1150,1200,1250,1300,1350,1400,1450,1500,1550,1600,1650,...
    1700,1750,1800,1850,1900,1950,2000,2100,2200,2300,2400,2500,...
    2600,2700,2800,2900,3000,3100,3200,3300,3400,3500,3600,3700,...
    3800,3900,4000,4100,4200,4300,4400,4500,4600,4700,4800,4900,...
    5000,5100,5200,5300,5400,5500];

% Construction of Salinity and Temperature matrices
for i=1:length(X)
    for ii=1:length(depth)
        for j=1:length(Y)
            Practical_Salinity(i,j,ii)=S(Idx_S(i),ii);
            Insitu_Temperature(i,j,ii)=T(Idx_T(i),ii);
        end
    end
end

% Pressure calculation
for i=1:length(Y)
    for j=1:length(Y)
        p(i,j,1:length(depth)) = gsw_p_from_z(-depth,Y(i,j));
    end
end

% 3D Longitude
Long_3D=zeros(length(X),length(Y),length(depth));
for k=1:length(depth)
    Long_3D(:,:,k)=depth(1,k);
end
for j=1:length(Y)
    Long_3D(:,j,:)=Y(1,j);
end
for i=1:length(X)
    Long_3D(i,:,:)=X(i,1);
end

% 3D Latitude
Lat_3D = zeros(length(X),length(Y),length(depth));
for k=1:length(depth)
    Lat_3D(:,:,k)=depth(1,k);
end
for i=1:length(X)
    Lat_3D(i,:,:)=X(i,1);
end
for j=1:length(Y)
    Lat_3D(:,j,:)=Y(1,j);
end
```

```
%  3D Depth
Depth_3D = zeros(length(X),length(Y),length(depth));
for i=1:length(X)
    Depth_3D(i,:,:)=X(i,1);
end
for j=1:length(Y)
    Depth_3D(:,j,:)=Y(1,j);
end
for k=1:length(depth)
    Depth_3D(:,:,k)=depth(1,k);
end

% 3D absolute salinity calculation
SA = gsw_SA_from_SP(Practical_Salinity,p,Long_3D,Lat_3D);

% 3D conservative temperature calculation
CT = gsw_CT_from_t(SA,Insitu_Temperature,p);

%  3D sound speed calculation
sound_speed = gsw_sound_speed(SA,CT,p);
Mvxyz=sound_speed;

%-------------------------------------------------------------------%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%        Calculating spatial gradients of the sound speed           %
%-------------------------------------------------------------------%
% dx_vxyz dy_vxyz dz_vxyz - First derivatives
% dx_dx_vxyz dx_dy_vxyz dx_dz_vxyz -    -|
% dy_dx_vxyz dy_dy_vxyz dy_dz_vxyz -     |   Second derivatives
% dz_dx_vxyz dz_dy_vxyz dz_dz_vxyz -    -|
%-------------------------------------------------------------------%
% x - Longitude
% y - Latitude
% z - Depth

%-----Centered finite-difference formulas----
% First derivatives: f'(xi)= -f(xi+2)+8f(xi+1)-8f(xi-1)+f(xi-2)/12h
% Second derivatives: f''(xi)= -f(xi+2)+16f(xi+1)-30f(xi)+...
%                               16f(xi-1)-f(xi-2)/12h^2
% h is the increment used between geographic positions

%-----Backward finite-difference formulas----
% First derivatives: f'(xi)= 3f(xi)-4f(xi-1)+f(xi-2)/2h
% Second derivatives: f''(xi)= 2f(xi)-5f(xi-1)+4f(xi-2)-f(xi-3)/h^2

%-----Forward finite-difference formulas-----
% First derivatives: f'(xi)= -f(xi+2)+4f(xi+1)-3f(xi)/2h
% Second derivatives: f''(xi)= -f(xi+3)+4f(xi+2)-5f(xi+1)+2f(xi)/h^2

%-------------------------------------------------------------------%
%%%%%%%%%%$%%%%%%%%%%%%% First Derivatives %%%%%%%%%%%%%%%%%%%%%%%%%
% dx - variation along the lines (Depth)
% dy - variation along the columns (Latitude)
% dz - variation between matrices (Longitude)
```

```
imersoes_2=[0,5,10,15,20,25,30,35,40,45,50,55,60,65,70,75,80,85,90,...
    95,100,125,150,175,200,225,250,275,300,325,350,375,400,425,...
    450,475,500,550,600,650,700,750,800,850,900,950,1000,1050,...
    1100,1150,1200,1250,1300,1350,1400,1450,1500,1550,1600,1650,...
    1700,1750,1800,1850,1900,1950,2000,2100,2200,2300,2400,2500,...
    2600,2700,2800,2900,3000,3100,3200,3300,3400,3500,3600,3700,...
    3800,3900,4000,4100,4200,4300,4400,4500,4600,4700,4800,4900,...
    5000,5100,5200,5300,5400,5500,5600];
imersoes_2=imersoes_2';

%--------- dx_vxyz calculation -----------
hx=inc_dis*1852;
for i = 1:length(depth);
    for k = 1:length(Y);
        for ii = 1:length(X);
            if i==1 % Forward finite-difference method
                Mdx_vxyz(i,ii,k)= (-Mvxyz(i+2,ii,k)+4.*...
                    Mvxyz(i+1,ii,k)-3.*Mvxyz(i,ii,k))/(2*hx);
            elseif i==2
                Mdx_vxyz(i,ii,k)= (-Mvxyz(i+2,ii,k)+4.*...
                    Mvxyz(i+1,ii,k)-3.*Mvxyz(i,ii,k))/(2*hx);
            elseif i==length(depth)% Backward finite-difference method
                Mdx_vxyz(i,ii,k)= (3.*Mvxyz(i,ii,k)-4.*...
                    Mvxyz(i-1,ii,k)+Mvxyz(i-2,ii,k))/(2*hx);
            elseif i==(length(depth)-1)
                Mdx_vxyz(i,ii,k)= (3.*Mvxyz(i,ii,k)-4.*...
                    Mvxyz(i-1,ii,k)+Mvxyz(i-2,ii,k))/(2*hx);
            else % Centered finite-difference method
                Mdx_vxyz(i,ii,k)= (-Mvxyz(i+2,ii,k)+8.*...
                    Mvxyz(i+1,ii,k)-8.*Mvxyz(i-1,ii,k)+...
                    Mvxyz(i-2,ii,k))/(12*hx);
            end
        end
    end

end

%--------- dy_vxyz calculation ---------------
hy=inc_dis*1852;
for i = 1:length(depth);
    for k = 1:length(Y);
        for ii = 1:length(X);
            if ii==1 %  Forward finite-difference method
                Mdy_vxyz(i,ii,k)= (-Mvxyz(i,ii+2,k)+4.*...
                    Mvxyz(i,ii+1,k)-3.*Mvxyz(i,ii,k))/(2*hy);
            elseif ii==2
                Mdy_vxyz(i,ii,k)= (-Mvxyz(i,ii+2,k)+4.*...
                    Mvxyz(i,ii+1,k)-3.*Mvxyz(i,ii,k))/(2*hy);
            elseif ii==length(X)% Backward finite-difference method
                Mdy_vxyz(i,ii,k)= (3.*Mvxyz(i,ii,k)-4.*...
                    Mvxyz(i,ii-1,k)+Mvxyz(i,ii-2,k))/(2*hy);
            elseif ii==(length(X)-1)
                Mdy_vxyz(i,ii,k)= (3.*Mvxyz(i,ii,k)-4.*...
```

```
                    Mvxyz(i,ii-1,k)+Mvxyz(i,ii-2,k))/(2*hy);
            else % Centered finite-difference method
                Mdy_vxyz(i,ii,k)= (-Mvxyz(i,ii+2,k)+8.*...
                    Mvxyz(i,ii+1,k)-8.*Mvxyz(i,ii-1,k)+...
                    Mvxyz(i,ii-2,k))/(12*hy);
            end
        end
    end
end

%--------- dz_vxyz calculation --------------
hz=diff(imersoes_2);
for k =1:length(depth)
   for i = 1:length(X);
       for ii = 1:length(Y);
           if ii==1 %  Forward finite-difference method
               Mdz_vxyz(k,i,ii)= (-Mvxyz(k,i,ii+2)+4.*...
                   Mvxyz(k,i,ii+1)-3.*Mvxyz(k,i,ii))/(2*hz(i,1));
           elseif ii==2
               Mdz_vxyz(k,i,ii)= (-Mvxyz(k,i,ii+2)+4.*...
                   Mvxyz(k,i,ii+1)-3.*Mvxyz(k,i,ii))/(2*hz(i,1));
           elseif ii==length(X)% Backward finite-difference method
               Mdz_vxyz(k,i,ii)= (3.*Mvxyz(k,i,ii)-4.*...
                   Mvxyz(k,i,ii-1)+Mvxyz(k,i,ii-2))/(2*hz(i,1));
           elseif ii==(length(X)-1)
               Mdz_vxyz(k,i,ii)= (3.*Mvxyz(k,i,ii)-4.*...
                   Mvxyz(k,i,ii-1)+Mvxyz(k,i,ii-2))/(2*hz(i,1));
           else % Centered finite-difference method
               Mdz_vxyz(k,i,ii)= (-Mvxyz(k,i,ii+2)+8.*...
                   Mvxyz(k,i,ii+1)-8.*Mvxyz(k,i,ii-1)+...
                   Mvxyz(k,i,ii-2))/(12*hz(i,1));
           end
       end
   end
end

%-------------------------------------------------------------------%
%%%%%%%%%%%%%%%%%%%%%% Second Derivatives %%%%%%%%%%%%%%%%%%%%%%%%%%
% The derivatives: dx_dx_vxyz dx_dy_vxyz dx_dz_vxyz
%                  dy_dx_vxyz dy_dy_vxyz dy_dz_vxyz
%                  dz_dx_vxyz dz_dy_vxyz dz_dz_vxyz
%-------------------------------------------------------------------%

%--------- dy_dx_vxyz calculation ---------
hy=inc_dis*1852;
for i = 1:length(depth);
    for k = 1:length(Y);
        for ii = 1:length(X);
            if ii==1 %  Forward finite-difference method
                Mdy_dx_vxyz(i,ii,k)= (-Mdx_vxyz(i,ii+2,k)+4.*...
                    Mdx_vxyz(i,ii+1,k)-3.*Mdx_vxyz(i,ii,k))/(2*hy);
            elseif ii==2
                Mdy_dx_vxyz(i,ii,k)= (-Mdx_vxyz(i,ii+2,k)+4.*...
                    Mdx_vxyz(i,ii+1,k)-3.*Mdx_vxyz(i,ii,k))/(2*hy);
```

```
            elseif ii==length(X)% Backward finite-difference method
                Mdy_dx_vxyz(i,ii,k)= (3.*Mdx_vxyz(i,ii,k)-4.*...
                    Mdx_vxyz(i,ii-1,k)+Mdx_vxyz(i,ii-2,k))/(2*hy);
            elseif ii==(length(X)-1)
                Mdy_dx_vxyz(i,ii,k)= (3.*Mdx_vxyz(i,ii,k)-4.*...
                    Mdx_vxyz(i,ii-1,k)+Mdx_vxyz(i,ii-2,k))/(2*hy);
            else % Centered finite-difference method
                Mdy_dx_vxyz(i,ii,k)= (-Mdx_vxyz(i,ii+2,k)+8.*...
                    Mdx_vxyz(i,ii+1,k)-8.*Mdx_vxyz(i,ii-1,k)+...
                    Mdx_vxyz(i,ii-2,k))/(12*hy);
            end
        end
    end
end

%--------- dx_dx_vxyz calculation ---------
hx=inc_dis*1852;
for i = 1:length(depth);
    for k = 1:length(Y);
        for ii = 1:length(X);
            if i==1 %  Forward finite-difference method
                Mdx_dx_vxyz(i,ii,k)= (-Mdx_vxyz(i+2,ii,k)+4.*...
                    Mdx_vxyz(i+1,ii,k)-3.*Mdx_vxyz(i,ii,k))/(2*hx);
            elseif i==2
                Mdx_dx_vxyz(i,ii,k)= (-Mdx_vxyz(i+2,ii,k)+4.*...
                    Mdx_vxyz(i+1,ii,k)-3.*Mdx_vxyz(i,ii,k))/(2*hx);
            elseif i==length(depth)% Backward finite-difference method
                Mdx_dx_vxyz(i,ii,k)= (3.*Mdx_vxyz(i,ii,k)-4.*...
                    Mdx_vxyz(i-1,ii,k)+Mdx_vxyz(i-2,ii,k))/(2*hx);
            elseif i==(length(depth)-1)
                Mdx_dx_vxyz(i,ii,k)= (3.*Mdx_vxyz(i,ii,k)-4.*...
                    Mdx_vxyz(i-1,ii,k)+Mdx_vxyz(i-2,ii,k))/(2*hx);
            else % Centered finite-difference method
                Mdx_dx_vxyz(i,ii,k)= (-Mdx_vxyz(i+2,ii,k)+8.*...
                    Mdx_vxyz(i+1,ii,k)-8.*Mdx_vxyz(i-1,ii,k)+...
                    Mdx_vxyz(i-2,ii,k))/(12*hx);
            end
        end
    end
end

%--------- dz_dx_vxyz calculation ---------
hz=diff(imersoes_2);
for k = 1:length(depth)
    for i = 1:length(X);
        for ii = 1:length(Y);
            if ii==1 % Forward finite-difference method
                Mdz_dx_vxyz(k,i,ii)= (-Mdx_vxyz(k,i,ii+2)+4.*...
                    Mdx_vxyz(k,i,ii+1)-3.*Mdx_vxyz(k,i,ii))/...
                    (2*hz(i,1));
            elseif ii==2
                Mdz_dx_vxyz(k,i,ii)= (-Mdx_vxyz(k,i,ii+2)+4.*...
                    Mdx_vxyz(k,i,ii+1)-3.*Mdx_vxyz(k,i,ii))/...
                    (2*hz(i,1));
```

```
            elseif ii==length(X)% Backward finite-difference method
                Mdz_dx_vxyz(k,i,ii)= (3.*Mdx_vxyz(k,i,ii)-4.*...
                    Mdx_vxyz(k,i,ii-1)+Mdx_vxyz(k,i,ii-2))/...
                    (2*hz(i,1));
            elseif ii==(length(X)-1)
                Mdz_dx_vxyz(k,i,ii)= (3.*Mdx_vxyz(k,i,ii)-4.*...
                    Mdx_vxyz(k,i,ii-1)+Mdx_vxyz(k,i,ii-2))/...
                    (2*hz(i,1));
            else % Centered finite-difference method
                Mdz_dx_vxyz(k,i,ii)= (-Mdx_vxyz(k,i,ii+2)+8.*...
                    Mdx_vxyz(k,i,ii+1)-8.*Mdx_vxyz(k,i,ii-1)+...
                    Mdx_vxyz(k,i,ii-2))/(12*hz(i,1));
            end
        end
    end
end

%--------- dy_dy_vxy calculation ---------
hy=inc_dis*1852;
for i = 1:length(depth);
    for k = 1:length(Y);
        for ii = 1:length(X);
            if ii==1 %  Forward finite-difference method
                Mdy_dy_vxyz(i,ii,k)= (-Mdy_vxyz(i,ii+2,k)+4.*...
                    Mdy_vxyz(i,ii+1,k)-3.*Mdy_vxyz(i,ii,k))/(2*hy);
            elseif ii==2
                Mdy_dy_vxyz(i,ii,k)= (-Mdy_vxyz(i,ii+2,k)+4.*...
                    Mdy_vxyz(i,ii+1,k)-3.*Mdy_vxyz(i,ii,k))/(2*hy);
            elseif ii==length(X)% Backward finite-difference method
                Mdy_dy_vxyz(i,ii,k)= (3.*Mdy_vxyz(i,ii,k)-4.*...
                    Mdy_vxyz(i,ii-1,k)+Mdy_vxyz(i,ii-2,k))/(2*hy);
            elseif ii==(length(X)-1)
                Mdy_dy_vxyz(i,ii,k)= (3.*Mdy_vxyz(i,ii,k)-4.*...
                    Mdy_vxyz(i,ii-1,k)+Mdy_vxyz(i,ii-2,k))/(2*hy);
            else % Centered finite-difference method
                Mdy_dy_vxyz(i,ii,k)= (-Mdy_vxyz(i,ii+2,k)+8.*...
                    Mdy_vxyz(i,ii+1,k)-8.*Mdy_vxyz(i,ii-1,k)+...
                    Mdy_vxyz(i,ii-2,k))/(12*hy);
            end
        end
    end
end

%--------- dx_dy_vxy calculation ---------
hx=inc_dis*1852;
for i = 1:length(depth);
    for k = 1:length(Y);
        for ii = 1:length(X);
            if i==1 %  Forward finite-difference method
                Mdx_dy_vxyz(i,ii,k)= (-Mdy_vxyz(i+2,ii,k)+4.*...
                    Mdy_vxyz(i+1,ii,k)-3.*Mdy_vxyz(i,ii,k))/(2*hx);
            elseif i==2
                Mdx_dy_vxyz(i,ii,k)= (-Mdy_vxyz(i+2,ii,k)+4.*...
                    Mdy_vxyz(i+1,ii,k)-3.*Mdy_vxyz(i,ii,k))/(2*hx);
```

```
            elseif i==length(depth)% Backward finite-difference method
                Mdx_dy_vxyz(i,ii,k)= (3.*Mdy_vxyz(i,ii,k)-4.*...
                    Mdy_vxyz(i-1,ii,k)+Mdy_vxyz(i-2,ii,k))/(2*hx);
            elseif i==(length(depth)-1)
                Mdx_dy_vxyz(i,ii,k)= (3.*Mdy_vxyz(i,ii,k)-4.*...
                    Mdy_vxyz(i-1,ii,k)+Mdy_vxyz(i-2,ii,k))/(2*hx);
            else % Centered finite-difference method
                Mdx_dy_vxyz(i,ii,k)= (-Mdy_vxyz(i+2,ii,k)+8.*...
                    Mdy_vxyz(i+1,ii,k)-8.*Mdy_vxyz(i-1,ii,k)+...
                    Mdy_vxyz(i-2,ii,k))/(12*hx);
            end
        end
    end
end

%--------- dz_dy_vxy calculation ---------
hz=diff(imersoes_2);
for i = 1:length(depth);
    for k = 1:length(X);
        for ii = 1:length(Y);
            if ii==1 %  Forward finite-difference method
                Mdz_dy_vxyz(i,k,ii)= (-Mdy_vxyz(i,k,ii+2)+4.*...
                    Mdy_vxyz(i,k,ii+1)-3.*Mdy_vxyz(i,k,ii))/...
                    (2*hz(i,1));
            elseif ii==2
                Mdz_dy_vxyz(i,k,ii)= (-Mdy_vxyz(i,k,ii+2)+4.*...
                    Mdy_vxyz(i,k,ii+1)-3.*Mdy_vxyz(i,k,ii))/...
                    (2*hz(i,1));
            elseif ii==length(Y)% Backward finite-difference method
                Mdz_dy_vxyz(i,k,ii)= (3.*Mdy_vxyz(i,k,ii)-4.*...
                    Mdy_vxyz(i,k,ii-1)+Mdy_vxyz(i,k,ii-2))/...
                    (2*hz(i,1));
            elseif ii==(length(Y)-1)
                Mdz_dy_vxyz(i,k,ii)= (3.*Mdy_vxyz(i,k,ii)-4.*...
                    Mdy_vxyz(i,k,ii-1)+Mdy_vxyz(i,k,ii-2))/...
                    (2*hz(i,1));
            else % Centered finite-difference method
                Mdz_dy_vxyz(i,k,ii)= (-Mdy_vxyz(i,k,ii+2)+8.*...
                    Mdy_vxyz(i,k,ii+1)-8.*Mdy_vxyz(i,k,ii-1)+...
                    Mdy_vxyz(i,k,ii-2))/(12*hz(i,1));
            end
        end
    end
end

%--------- dx_dz_vxy calculation ---------
hx=inc_dis*1852;
for k = 1:length(depth)
    for i = 1:length(X);
        for ii = 1:length(Y);
            if k==1 %  Forward finite-difference method
                Mdx_dz_vxyz(k,i,ii)= (-Mdz_vxyz(k+2,i,ii)+4.*...
                    Mdz_vxyz(k+1,i,ii)-3.*Mdz_vxyz(k,i,ii))/(2*hx);
            elseif k==2
```

```
                    Mdx_dz_vxyz(k,i,ii)= (-Mdz_vxyz(k+2,i,ii)+4.*...
                        Mdz_vxyz(k+1,i,ii)-3.*Mdz_vxyz(k,i,ii))/(2*hx);
                elseif k==length(depth)% Backward finite-difference method
                    Mdx_dz_vxyz(k,i,ii)= (3.*Mdz_vxyz(k,i,ii)-4.*...
                        Mdz_vxyz(k-1,i,ii)+Mdz_vxyz(k-2,i,ii))/(2*hx);
                elseif k==(length(depth)-1)
                    Mdx_dz_vxyz(k,i,ii)= (3.*Mdz_vxyz(k,i,ii)-4.*...
                        Mdz_vxyz(k-1,i,ii)+Mdz_vxyz(k-2,i,ii))/(2*hx);
                else % Centered finite-difference method
                    Mdx_dz_vxyz(k,i,ii)= (-Mdz_vxyz(k+2,i,ii)+8.*...
                        Mdz_vxyz(k+1,i,ii)-8.*Mdz_vxyz(k-1,i,ii)+...
                        Mdz_vxyz(k-2,i,ii))/(12*hx);
                end
            end
        end
    end

    %--------- dy_dz_vxy calculation ---------
    hy=inc_dis*1852;
    for i = 1:length(depth);
        for k = 1:length(Y);
            for ii = 1:length(X);
                if ii==1 %  Forward finite-difference method
                    Mdy_dz_vxyz(i,ii,k)= (-Mdz_vxyz(i,ii+2,k)+4.*...
                        Mdz_vxyz(i,ii+1,k)-3.*Mdz_vxyz(i,ii,k))/(2*hy);
                elseif ii==2
                    Mdy_dz_vxyz(i,ii,k)= (-Mdz_vxyz(i,ii+2,k)+4.*...
                        Mdz_vxyz(i,ii+1,k)-3.*Mdz_vxyz(i,ii,k))/(2*hy);
                elseif ii==length(X)% Backward finite-difference method
                    Mdy_dz_vxyz(i,ii,k)= (3.*Mdz_vxyz(i,ii,k)-4.*...
                        Mdz_vxyz(i,ii-1,k)+Mdz_vxyz(i,ii-2,k))/(2*hy);
                elseif ii==(length(X)-1)
                    Mdy_dz_vxyz(i,ii,k)= (3.*Mdz_vxyz(i,ii,k)-4.*...
                        Mdz_vxyz(i,ii-1,k)+Mdz_vxyz(i,ii-2,k))/(2*hy);
                else % Centered finite-difference method
                    Mdy_dz_vxyz(i,ii,k)= (-Mdz_vxyz(i,ii+2,k)+8.*...
                        Mdz_vxyz(i,ii+1,k)-8.*Mdz_vxyz(i,ii-1,k)+...
                        Mdz_vxyz(i,ii-2,k))/(12*hy);
                end
            end
        end
    end

    %--------- dz_dz_vxy calculation ---------
    hz=diff(imersoes_2);
    for i = 1:length(depth);
        for k = 1:length(X);
            for ii = 1:length(Y);
                if ii==1 %  Forward finite-difference method
                    Mdz_dz_vxyz(i,k,ii)= (-Mdz_vxyz(i,k,ii+2)+4.*...
                        Mdz_vxyz(i,k,ii+1)-3.*Mdz_vxyz(i,k,ii))/...
                        (2*hz(i,1));
                elseif ii==2
                    Mdz_dz_vxyz(i,k,ii)= (-Mdz_vxyz(i,k,ii+2)+4.*...
```

```
                    Mdz_vxyz(i,k,ii+1)-3.*Mdz_vxyz(i,k,ii))/...
                    (2*hz(i,1));
            elseif ii==length(Y)% Backward finite-difference method
                Mdz_dz_vxyz(i,k,ii)= (3.*Mdz_vxyz(i,k,ii)-4.*...
                    Mdz_vxyz(i,k,ii-1)+Mdz_vxyz(i,k,ii-2))/...
                    (2*hz(i,1));
            elseif ii==(length(Y)-1)
                Mdz_dz_vxyz(i,k,ii)= (3.*Mdz_vxyz(i,k,ii)-4.*...
                    Mdz_vxyz(i,k,ii-1)+Mdz_vxyz(i,k,ii-2))/...
                    (2*hz(i,1));
            else % Centered finite-difference method
                Mdz_dz_vxyz(i,k,ii)= (-Mdz_vxyz(i,k,ii+2)+8.*...
                    Mdz_vxyz(i,k,ii+1)-8.*Mdz_vxyz(i,k,ii-1)+...
                    Mdz_vxyz(i,k,ii-2))/(12*hz(i,1));
            end
        end
    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Priting the graphics
disp('1 to print')
disp('0 to continue')
graphics=input('Do you wish to print the graphics: ');

if graphics ==1
    clear colorbar
    figure % % Bathymetry Map
    contourf(long_area,lat_area,prof_area(1:size(long_area(:)),...
        1:size(lat_area(:)))')
    hold on

 scatter(posicao_geo3D(:,2),posicao_geo3D(:,1),'r','.','LineWidth',2);
    title ('Bathymetric map of the simulation area')
    xlabel ('Longitude [°]')
    ylabel ('Latitude [°]')
    colorbar=colorbar;
    colorbar.Label.String = 'Altitude in meters';

 set(gca,'Fontsize',10,'FontWeight','bold','LineWidth',2.0,'box','on');

    figure% 3D Graphic
    mesh(X_m(:,1),Y_m(1,:),prof_area(1:length(X_m),1:length(Y_m))')
    title('3D Bathymetry')
    xlabel('Distance to the initial geographic point [m] -
 Longitude');
    ylabel('Distance to the initial geographic point [m] - Latitude');
    zlabel('Depth [m]');

 set(gca,'Fontsize',10,'FontWeight','bold','LineWidth',2.0,'box','on');

    figure
    scatter(posicao_geoS(:,2),posicao_geoS(:,1),'b');
    hold on
```

```
    scatter(posicao_geoT(:,2),posicao_geoT(:,1),'b');
    hold on

 scatter(posicao_geo3D(:,2),posicao_geo3D(:,1),'r','.','LineWidth',2);
    hold on
    scatter(posicao_geoS(Idx_S,2),posicao_geoS(Idx_S,1),'g',...
        's','LineWidth',2)
    hold on
    scatter(posicao_geoT(Idx_T,2),posicao_geoT(Idx_T,1),'+',...
        'LineWidth',2)
    title('Map with the distribution of points with the climatological
 data')
    xlabel ('Longitude [º]')
    ylabel ('Latitude [º]')

 set(gca,'Fontsize',10,'FontWeight','bold','LineWidth',2.0,'box','on');
elseif graphics ==0
    disp('End')
elseif isempty(graphics)==1
    disp('1 to print')
    disp('0 to continue')
    graphics=input('Do you wish to print the graphics: ');
else
    disp('1 to print')
    disp('0 to continue')
    graphics=input('Do you wish to print the graphics: ');
end
```

Published with MATLAB® R2015a

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%-------------------------------------------------------------------------%
%                     REOART 3D - Ray Propagation                         %
%-------------------------------------------------------------------------%
% This is the REOART 3D main script where the propagation of the
% acoutic ray is calculated
%-------------------------------------------------------------------------%
%
clc
Vmatrix=[]; % ray trajectory matrix
Tsmatrix=[]; % propagation time matrix

% Initial conditions (change as you intend)
delta=0.5;
ts=0:delta:200;
theta0=0; %Angle in degrees 0<theta0<90
theta0=pi*theta0/180;
phi0=3; %Angle in degrees -90<phi0<90
phi0=pi*phi0/180;
%
a0= 50000;
b0= 80000;
c0=500;
%
x0=cos(phi0)*cos(theta0);
y0=cos(phi0)*sin(theta0);
z0=sin(phi0);
%
v0 =[a0;b0;c0;x0;y0;z0];

Zz=-prof_area(1:length(X_m),1:length(Y_m))';

for il=1:20
    v=dae4('pa3dr',ts,v0,10);
    Zq=interp2(Y_m,X_m,Zz,v(1,:),v(2,:));
    ctemp1=min(find(v(3,2:end)>= Zq(1,2:end)));
    ctemp2=min(find(v(3,2:end)<=0));
    if isempty(ctemp1)==1 && isempty(ctemp2)==1
        disp('No Reflection')
        ts=ts+ts(end);
        Vmatrix=[Vmatrix v(:,1:end-1)];
        Tsmatrix=[Tsmatrix ts(1,1:end-1)];
        v0=v(:,end);

    elseif isempty(ctemp1)~=1 && isempty(ctemp2)==1
        disp('Bottom Reflection')
        %
        Zq_lat=interp2(Y_m,X_m,Zz,v(1,ctemp1+1)+100,v(2,ctemp1));
        Zq_long=interp2(Y_m,X_m,Zz,v(1,ctemp1),v(2,ctemp1+1)+100);
        %
        p1=[v(1,ctemp1),v(2,ctemp1),v(3,ctemp1)];
        p2=[v(1,ctemp1+1)+100,v(2,ctemp1),Zq_lat];
```

```
            p3=[v(1,ctemp1),v(2,ctemp1+1)+100,Zq_long];
            %
            A=p1-p2;
            B=p3-p2;
            C = cross(A,B);
            C = C./norm(C);
            dot(C,A);
            dot(C,B);
            normal_R=C';
            %
            [Rn,i,r]=reflection3D(normal_R,v,ctemp1)
            %
            a0=v(1,ctemp1);
            b0=v(2,ctemp1);
            c0=v(3,ctemp1);
            x0=r(1,1);
            y0=r(2,1);
            z0=r(3,1);
            %
            v0 =[a0;b0;c0;x0;y0;z0]; % New initial conditions
            Vmatrix=[Vmatrix v(:,1:ctemp1)];
            Tsmatrix=[Tsmatrix ts(1,1:ctemp1)];

        elseif isempty(ctemp1)==1 && isempty(ctemp2)~=1
            disp('Surface reflection')
            %
            p1=[v(1,ctemp2),v(2,ctemp2),v(3,ctemp2)];
            p2=[v(1,ctemp2+1)+100,v(2,ctemp2),v(3,ctemp2)];
            p3=[v(1,ctemp2),v(2,ctemp2+1)+100,v(3,ctemp2)];
            %
            A=p1-p2;
            B=p3-p2;
            C = cross(A,B);
            C = C./norm(C);
            dot(C,A)
            dot(C,B)
            normal_R_sup=C'
            %
            [Rn,i,r]=sup_reflection3D(normal_R_sup,v,ctemp2)
            %
            a0=v(1,ctemp2);
            b0=v(2,ctemp2);
            c0=v(3,ctemp2);
            x0=r(1,1);
            y0=r(2,1);
            z0=r(3,1);
            %
            v0 =[a0;b0;c0;x0;y0;z0]; % New initial conditions
            Vmatrix=[Vmatrix v(:,1:ctemp2-1)];
            Tsmatrix=[Tsmatrix ts(1,1:ctemp2-1)];
        end
        v0
    end
```

```
figure
mesh(X_m(:,1),Y_m(1,:),prof_area(1:length(X_m),1:length(Y_m))')
% xlim([{number} {number}])        -| To see closely uncomment
% ylim([{number} {number}])        -| the xlim and ylim.
hold on;
plot3(Vmatrix(1,:),Vmatrix(2,:),-Vmatrix(3,:),'r','LineWidth',1.5)
set(gca,'Fontsize',10,'FontWeight','bold','LineWidth',2.0,'box','on');
a1=xlabel('Distance to the initial geographic point [m] - Longitude');
a2=ylabel('Distance to the initial geographic point [m] - Latitude');
a3=zlabel('Depth [m]');
a4=title('Acoustic ray propagation');
set(a1,'Fontsize',12,'FontWeight','bold');
set(a2,'Fontsize',12,'FontWeight','bold');
set(a3,'Fontsize',12,'FontWeight','bold');
set(a4,'Fontsize',12,'FontWeight','bold');
```

Published with MATLAB® R2015a

```
function [y_lat,x_long,dis_m,prof_area,long_area,lat_area,inc_long,...
    inc_lat]=
 Geodesic_Line(latitude,longitude,elevation,lat_S,lat_N,...
    long_W,long_E,lat_pos,long_pos,azimute, distancia,inc_dis)

% This function calculates the geodesic line.
% The output are the geographical positions that compose the geodesic
% line.

inc_long=30/3600;
inc_lat=inc_long;
prof_max=min(min(elevation));
prof_min=max(max(elevation));
ind_long=round((long_pos-min(longitude))/inc_long+1);
ind_lat=round((lat_pos-min(latitude))/inc_lat+1);
prof_pos=elevation(ind_long,ind_lat);
ind_long_W=round((long_W-min(longitude))/inc_long+1);
ind_long_E=round((long_E-min(longitude))/inc_long+1);
ind_lat_S=round((lat_S-min(latitude))/inc_lat+1);
ind_lat_N=round((lat_N-min(latitude))/inc_lat+1);
prof_area=elevation(ind_long_W:ind_long_E,ind_lat_S:ind_lat_N);
size(prof_area);
max_prof_area=min(min(prof_area));
min_prof_area=max(max(prof_area));
long_area=long_W:inc_long:long_E;
size(long_area);
lat_area=lat_S:inc_lat:lat_N;
size(lat_area);
%-------------------------------------------------------------------%
distancia_m=distancia*1852;
inc_dis_m=inc_dis*1852;
dis_m=0:inc_dis_m:distancia_m;
long_pos=-long_pos;
s=dis_m;
exc=0.0819919;
a_eli=6378388;
if azimute<=180
    azimute_S=180+azimute;
else
    azimute_S=azimute-180;
end
R_mer=(a_eli*(1-exc^2))/((1-exc^2*(sind(lat_pos))^2)^(3/2));
N=a_eli/((1-exc^2*sind(lat_pos)^2)^(1/2));
B=1/(R_mer*sind(1/3600));
C=tand(lat_pos)/(2*R_mer*N*sind(1/3600));
D=(3*exc^2*sind(lat_pos)*cosd(lat_pos)*sind(1/3600))/(2*(1-exc^2*...
    (sind(lat_pos))^2));
E=(1+3*(tand(lat_pos))^2)/(6*N^2);
del_fi=-s.*cosd(azimute_S).*B-s.^2.*(sind(azimute_S)).^2.*C;
h=s.*cosd(azimute_S).*B;
delta_fi_2linha=-s.*cosd(azimute_S).*B-s.^2.*(sind(azimute_S)).^2.*...
    C-del_fi.^2.*D+h.*s.^2.*(sind(azimute_S)).^2.*E;
```

1

```
delta_fi=delta_fi_2linha./3600;
lat_2=lat_pos+delta_fi;
N_linha=a_eli./((1-exc.^2.*sind(lat_2).^2).^(1/2));
A_linha=1./(N_linha.*sind(1./3600));
log_delta_lambda_2linha=log10(s)+log10(abs(sind(azimute_S)))+...
    log10(abs(A_linha))+log10(abs(secd(lat_2)));
delta_long=10.^log_delta_lambda_2linha./3600;
if azimute_S <= 180
    long_2=long_pos+delta_long;
else
    long_2=long_pos-delta_long;
end
y_lat=lat_2;
x_long=-long_2;
end
```

Published with MATLAB® R2015a

```
function [f,k,m]=pa3dr(t,v,vd)
%=================================================================%
% t - time
% v = [a;b;c;x;y;z]
% ad - derivative of a
% bd - derivative of b
% cd - derivative of c
% xd - derivative of x
% yd - derivative of y
% zd - derivative of z
%=================================================================%
a=v(1); b=v(2); c=v(3); x=v(4); y=v(5); z=v(6);
%
ad=vd(1); bd=vd(2); cd=vd(3); xd=vd(4); yd=vd(5); zd=vd(6);

%
[vabc da_vabc db_vabc dc_vabc da_da_vabc db_da_vabc dc_da_vabc
 da_db_vabc...
    db_db_vabc dc_db_vabc da_dc_vabc db_dc_vabc dc_dc_vabc ...
    dt_vabc da_dt_vabc db_dt_vabc dc_dt_vabc]=cv3dr(v);
% vabc- sound velocity field

% System of differential equations without equal sign and all reduced
% to the left hand side
f=[ad-x;
   bd-y;
   cd-z;
   (x*z-(y^2+z^2))*xd+(x*y+y*z)*yd+(x*z-(x^2+y^2))*zd+(dt_vabc*(x
+z)-...
   da_vabc-dc_vabc)/vabc;
   (x*y)*xd-(x^2+z^2)*yd+(y*z)*zd+(dt_vabc*y-db_vabc)/vabc;
   x*xd+y*yd+z*zd];

k_41=(da_dt_vabc*(x+z)-da_da_vabc-da_dc_vabc)/vabc-da_vabc*...
    (dt_vabc*(x+z)-da_vabc-dc_vabc)/vabc^2;
%
k_42=(db_dt_vabc*(x+z)-db_da_vabc-db_dc_vabc)/vabc-db_vabc*...
    (dt_vabc*(x+z)-da_vabc-dc_vabc)/vabc^2;
%
k_43=(dc_dt_vabc*(x+z)-dc_da_vabc-dc_dc_vabc)/vabc-dc_vabc*...
    (dt_vabc*(x+z)-da_vabc-dc_vabc)/vabc^2;
%
k_44=z*xd+y*yd+z*zd+dt_vabc/vabc;
%
k_45=-2*y*xd+x*yd-2*y*zd;
%
k_46=x-2*z*xd+y*yd+x*zd+dt_vabc*x/vabc;
%
%
k_51=(da_dt_vabc*y-da_db_vabc)/vabc-da_vabc*(dt_vabc*y-db_vabc)/
vabc^2;
```

```
k_52=(db_dt_vabc*y-db_db_vabc)/vabc-db_vabc*(dt_vabc*y-db_vabc)/
vabc^2;
k_53=(dc_dt_vabc*y-dc_db_vabc)/vabc-dc_vabc*(dt_vabc*y-db_vabc)/
vabc^2;
%
k_54=y*xd-2*x*yd;
k_55=x*xd+z*zd+dt_vabc/vabc;
k_56=-2*z*yd+y*zd;
%
k_61=0;
k_62=0;
k_63=0;
%
k_64=xd;
k_65=yd;
k_66=zd;

% Jacobian with respect to vector v: a,b,c,x,y,z
k=[0,0,0,-1,0,0;
   0,0,0,0,-1,0;
   0,0,0,0,0,-1;
    k_41,k_42,k_43,k_44,k_45,k_46;
    k_51,k_52,k_53,k_54,k_55,k_56;
    k_61,k_62,k_63,k_64,k_65,k_66];

%Jacobian with respect to vector vd: ad, bd, cd, xd, yd, zd
m=[1,0,0,0,0,0;
   0,1,0,0,0,0;
   0,0,1,0,0,0;
   0,0,0,(x*z-(y^2+z^2)),(x*y+y*z),(x*z-(x^2+y^2));
   0,0,0,x*y,-(x^2+z^2),y*z;
   0,0,0,x,y,z];
end
```

Published with MATLAB® R2015a

```
function [vabc da_vabc db_vabc dc_vabc da_da_vabc db_da_vabc...
    dc_da_vabc da_db_vabc db_db_vabc dc_db_vabc da_dc_vabc...
    db_dc_vabc dc_dc_vabc dt_vabc da_dt_vabc db_dt_vabc...
    dc_dt_vabc] = cv3dr(v);
%==========================================================================%
% This function interpolates the sound speed and the spatial
% gradients with the position of the acoustic ray. This allows to
% determinate the sound speed along the acoustic ray trajectory.
%
% The global variables are the sound speed and spatial
% gradients matrices, distance and depth of the 3D simulation plan.
%==========================================================================%
global Mvxyz Mdx_vxyz Mdy_vxyz Mdz_vxyz Mdx_dx_vxyz Mdx_dy_vxyz...
    Mdx_dz_vxyz Mdy_dx_vxyz Mdy_dy_vxyz Mdy_dz_vxyz Mdz_dx_vxyz...
    Mdz_dy_vxyz Mdz_dz_vxyz X_2 Y_2 Lat_3D Long_3D X_m Y_m depth...
    Depth_3D

a=v(1); b=v(2); c=v(3); x=v(4); y=v(5); z=v(6);

[X,Y,Z] = meshgrid(X_m(:,1)',Y_m(1,:),depth);

vabc=interp3(X,Y,Z,Mvxyz,a,b,c,'linear',1495);

da_vabc=interp3(X,Y,Z,Mdx_vxyz,a,b,c,'linear',0.0005);

db_vabc=interp3(X,Y,Z,Mdy_vxyz,a,b,c,'linear',0.0005);

dc_vabc=interp3(X,Y,Z,Mdz_vxyz,a,b,c,'linear',0.0005);

da_da_vabc=interp3(X,Y,Z,Mdx_dx_vxyz,a,b,c,'linear',0.0005);

db_da_vabc=interp3(X,Y,Z,Mdy_dx_vxyz,a,b,c,'linear',0.0005);

dc_da_vabc=interp3(X,Y,Z,Mdz_dx_vxyz,a,b,c,'linear',0.0005);

da_db_vabc=interp3(X,Y,Z,Mdx_dy_vxyz,a,b,c,'linear',0.0005);

db_db_vabc=interp3(X,Y,Z,Mdy_dy_vxyz,a,b,c,'linear',0.0005);

dc_db_vabc=interp3(X,Y,Z,Mdz_dy_vxyz,a,b,c,'linear',0.0005);

da_dc_vabc=interp3(X,Y,Z,Mdx_dz_vxyz,a,b,c,'linear',0.0005);

db_dc_vabc=interp3(X,Y,Z,Mdy_dz_vxyz,a,b,c,'linear',0.0005);

dc_dc_vabc=interp3(X,Y,Z,Mdz_dz_vxyz,a,b,c,'linear',0.0005);


dt_vabc=da_vabc*x+db_vabc*y+dc_vabc*z;
%
da_dt_vabc=da_da_vabc*x+da_db_vabc*y+da_dc_vabc*z;
db_dt_vabc=db_da_vabc*x+db_db_vabc*y+db_dc_vabc*z;
```

```
dc_dt_vabc=dc_da_vabc*x+dc_db_vabc*y+dc_dc_vabc*z;
end
```

Published with MATLAB® R2015a

```
function [Rn,i,r] = reflection3D(normal_R,v,ctemp1)
%=========================================================================%
% This function calculates the reflection of the acoustic ray
% in the bottom of the ocean.
%  Rn = normal vector to the plane at the point of reflection (Matrix)
%  i = incident acoustic ray
%  r = reflected acoustic ray
%=========================================================================%
Rn=[1-2*(normal_R(1,1)^2),-2*normal_R(1,1)*normal_R(2,1),-2*...
    normal_R(1,1)*normal_R(3,1);...
    -2*normal_R(1,1)*normal_R(2,1),1-2*(normal_R(2,1)^2),-2*...
    normal_R(2,1)*normal_R(3,1);...
    -2*normal_R(1,1)*normal_R(3,1),-2*normal_R(2,1)*...
    normal_R(3,1),1-2*(normal_R(3,1)^2)];
Rn=Rn/norm(Rn);

i=[v(4,ctemp1);v(5,ctemp1);v(6,ctemp1)];
i=i/norm(i);

r=Rn*i;
r=r/norm(r);
end
```

Published with MATLAB® R2015a

```
function [Rn,i,r] = sup_reflection3D(normal_R_sup,v,ctemp2)
%=========================================================================%
% This function calculates the reflection of the acoustic ray
% in the ocean surface.
%  Rn = normal vector to the plane at the point of reflection (Matrix)
%  i = incident acoustic ray
%  r = reflected acoustic ray
%=========================================================================%
Rn=[1-2*(normal_R_sup(1,1)^2),-2*normal_R_sup(1,1)*...
    normal_R_sup(2,1),-2*normal_R_sup(1,1)*normal_R_sup(3,1);...
    -2*normal_R_sup(1,1)*normal_R_sup(2,1),1-2*...
    (normal_R_sup(2,1)^2),-2*normal_R_sup(2,1)*normal_R_sup(3,1);...
    -2*normal_R_sup(1,1)*normal_R_sup(3,1),-2*normal_R_sup(2,1)*...
    normal_R_sup(3,1),1-2*(normal_R_sup(3,1)^2)];
Rn=Rn/norm(Rn);

i=[v(4,ctemp2);v(5,ctemp2);v(6,ctemp2)];
i=i/norm(i);

r=Rn*i;
r=r/norm(r);

end
```

Published with MATLAB® R2015a