UNIVERSIDADE BEIRA INTERIOR
Engenharia

Development of a Python Library for Processing Seismic Time Series
Versão final após apresentação

Eduardo Rodrigues de Almeida

Dissertação para a obtenção do grau de Mestre em
Engenharia Informática
(2º ciclo de estudos)

Orientador: Professor Doutor Paul Andrew Crocker
Co-orientador: Doutor Hamzeh Mohammadigheymasi

Covilhã Novembro 2021

Contents

List of Figures

List of Tables

Listings

Acronyms

ARIMA AutoRegressive Integrated Moving Average

ASDF Adaptable Seismic Data Format

ASP Analog Signal Processing

CPU Central Processing Unit

CSS Cascading Style Sheets

CWT Continuous Wavelet Transform

C4G Collaboratory for Geosciences

DSP Digital Signal Processing

EPOS European plate observing system

EGU European Geosciences Union

FT Fourier transform

FFT Fast Fourier Transform

FWI full waveform imaging

GPU Graphics Processing Unit

GUI Graphical User Interface

HDF Hierarchical Data Format

HT Hilbert Transform

HTML HyperText Markup Language

IEEE Institute of Electrical and Electronics Engineers

IFFT Inverse Fourier Transform

IRIS Incorporated Research Institutions for Seismology

KKT Karush-Kuhn-Tucker

MPI Message Passing Interface

ORFEUS Observatories and Research Facilities for European Seismology

P primary

pP pulse P

S secondary

SAC Seismic Analysis Code

SLI Scalable Link Interface

ST S transform

STFT short time Fourier transform

SHAZAM Seismicity and HAzards of the sub-saharian Atlantic Margin

TF	Time-Frequency
UBI	Universidade da Beira Interior
RAM	Random Access Memory
VM	Virtual Machine
VRAM	Video Random Access Memory

Acknowledgments

I want to start by thanking my entire family for everything that they have done for me and mainly for all the support I received from each one of them. In the end, I have no doubt that was one of the main factors for helping me endure and overcome every hardship faced during these last years which have been rough on everyone.
Furthermore, I would like to give a very special thank you to my supervisor, Paul Crocker, and my co-supervisor, Hamzeh Mohammadigheymasi, for always being available, positive and supportive in every step of this dissertation development. It was a pleasure to work with both. I truly believe that I've grown a lot in the last year from their experience in life, and for that, I'll be forever in debt with them.
Also, I want to thank all my friends who have been accompanying me so far. Thank you all for always being there not just for the good parts of this journey but also, and most importantly, for helping me through the bad parts as well. Thank you all for being amazing. I really couldn't ask for more.
Finally, I want to express my undivided gratitude to the two persons who are the closest to me and the best thing that university has given me. To Patricia Almeida, and my best friend, Carolina Lopes, I want to apologise for my putting up with me on the bad days when nothing goes right and thank you for all the love and support I've received. You are truly amazing and achieving this wouldn't be possible without you. I hope you know that. In sum, I hope that I've made everyone proud and that I keep making you all feel that way. Thank you for everything.

This work is funded by the SHAZAM, Seismicity and HAzards of the sub-saharian Atlantic Margin project (02/SAICT/2017/31475) and also FCT/MCTES through national funds and when applicable co-funded EU funds under the project UIDB/EEA/50008/2020.

Abstract

Earthquakes occur around the world every day. This natural phenomena can result in enormous destruction and loss of life. However, at the same time, it is the primary source for studying Earth, the active planet. The seismic waves generated by earthquakes propagate deep into the Earth, carrying considerable information about the Earth's structure, from the shallow depths in the crust to the core. The information transferred by seismic waves needs advanced signal processing and inversion tools to be converted into useful information about the Earths inner structures, from local to global scales. The ever-evolving interest for investigating more accurately the terrestrial system led to the development of advanced signal processing algorithms to extract optimal information from the recorded seismic waveforms. These algorithms use advanced numerical modeling to extract optimal information from the different seismic phases generated by earthquakes. The development of algorithms from a mathematical-physical point of view is of great interest; on the other hand, developing a platform for their implementation is also significant.
This research aims to build a bridge between the development of purely theoretical ideas in seismology and their functional implementation. In this dissertation SeisPolPy, a high quality Python-based library for processing seismic waveforms is developed. It consists of the latest polarization analysis and filter algorithms to extract different seismic phases in the recorded seismograms. The algorithms range from the most common algorithms in the literature to a newly developed method, sparsity-promoting time-frequency filtering. In addition, the focus of the work is on the generation of high-quality synthetic seismic data for testing and evaluating the algorithms. SeisPolPy library, aims to provide seismology community a tool for separation of seismic phases by using high-resolution polarization analysis and filtering techniques. The research work is carried out within the framework of the Seismicity and HAzards of the sub-saharian Atlantic Margin (SHAZAM) project that requires high quality algorithms able to process the limited seismic data available in the Gulf of Guinea, the study area of the SHAZAM project.

Keywords

Library, Synthetic Data, Methods, Python, Signal Processing, Time Series, Seismology, Seismic Signal.

Resumo

Terramotos ocorrem todos os dias em todo o mundo. Esta fenomeno natural pode vir a resultar numa enorme destruição e perda de vidas. No entanto, ao mesmo tempo, é a principal fonte para o estudo da Terra, o planeta activo. As ondas sísmicas geradas pelos terramotos propagam-se profundamente na Terra, levando informação considerável sobre a estrutura da Terra, desde as zonas de menor profundidade da crosta até ao núcleo. A informação transferida por ondas sísmicas necessita de processamento avançado de sinais e ferramentas de inversão para ser convertida em informação util sobre a estrutura interna da Terra, desde escalas locais a globais. O interesse sempre crescente em investigar com maior precisão o sistema terrestre levou ao desenvolvimento de algoritmos avançados de processamento de sinais para extrair informação óptima das formas de ondas sísmicas registadas. Estes algoritmos fazem uso de modelos numéricos avançados para extrair informação óptima das diferentes fases sísmicas geradas pelos terramotos. O desenvolvimento de algoritmos de um ponto de vista matemático-físico é de grande interesse; por outro lado, o desenvolvimento de uma plataforma para a sua implementação é também significativo.

Esta investigação visa construir uma ponte entre o desenvolvimento de ideias puramente teóricas em sismologia e a sua implementação funcional. Com o decorrer desta dissertação foi desenvolvido o SeisPolPy, uma biblioteca de alta qualidade baseada em Python para o processamento de formas de ondas sísmicas. Consiste na mais recente análise de polarização e algoritmos de filtragem para extrair diferentes fases sísmicas nos sismogramas registados. Os algoritmos variam desde os algoritmos mais comuns na literatura até um método recentemente desenvolvido, que promove a frequência de filtragem por tempo e frequência. Além disso, o foco do trabalho é a geração de dados sísmicos sintéticos de alta qualidade para testar e avaliar os algoritmos. A biblioteca SeisPolPy, visa fornecer à comunidade sismológica uma ferramenta para a separação das fases sísmicas, utilizando técnicas de análise de polarização e filtragem de alta resolução. O trabalho de investigação é realizado no âmbito do projecto SHAZAM que requer algoritmos de alta qualidade que possuam a capacidade de processar os dados sísmicos, limitados, disponíveis no Golfo da Guiné, a área de estudo do projecto.

Palavras-chave

Biblioteca, Dados Sintéticos, Métodos, Python, Processamento de Sinais, Séries temporais, Sismologia, Sinais Sísmicos.

Resumo Alargado

Ao colocar sensores sísmicos em locais adequados, as ondas sísmicas geradas por deslizamentos de terra, fluxos de magma, ondas do mar e terramotos são registadas como uma função discreta do tempo em amostras de tempo regulares, as chamadas séries de tempo sísmicas. A série temporal é uma combinação de efeitos de diferentes parâmetros, incluindo os parâmetros da fonte sísmica, ambiente de propagação, efeitos do local, etc.
O estudo de séries de tempo sísmicas oferece aos cientistas uma visão profunda desses parâmetros críticos, o que requer o desenvolvimento e design de algoritmos de processamento bem definidos para analisar as séries de tempo e extrair informações importantes das mesmas. Neste sentido, embora o desenvolvimento de algoritmos matemáticos robustos seja um aspecto fundamental, a plataforma computacional e as versáteis linguagens de programação para a implementação dos algoritmos desenvolvidos são de particular importância. Consequentemente, foram publicados vários programas de última geração, entre eles, ObsPy uma das ferramentas de *software* mais prevalentes na comunidade sismológica. ObsPy, é uma *framework* baseada em Python e de código aberto, permitindo que a comunidade sismológica contribua livremente com seu desenvolvimento.
Várias bibliotecas de processamento foram desenvolvidas com base na plataforma de software ObsPy, por exemplo as plataformas MsNoise, Noisi e NoisePy, implementadas por Lecocq, Ermert e Jiang, respectivamente.
ObsPy fornece módulos de pré-processamento para leitura e conversão de vários formatos de dados em servidores locais ou remotos, aplicando filtros *band-pass* para o aumento do sinal e cancelamento de ruído, remoção de tendência, deconvolução da resposta do instrumento. Este apresenta a série temporal como uma matriz compatível com bibliotecas matemáticas padrão como por exemplo NumPy, SciPy, Matplotlib.
Este trabalho de pesquisa tem como objectivo desenvolver uma biblioteca Python de última geração consistente com ObsPy para a implementação de ideias desenvolvidas para processamento de séries temporais sísmicas. Como o trabalho de pesquisa combina ciências da computação e sismologia teórica, uma apresentação abstracta de conceitos básicos em ambos os campos também é apresentada e também um conjunto de referencias serão indicadas para os leitores não familiarizados com todos estes tópicos.
Os objectivos da presente tese são os que se seguem:

1. Implementar e aprimorar os algoritmos introduzidos por Flinn [1965]; Vidale [1986]; Pinnegar [2006] para melhoria do sinal e recuperação de séries temporais sísmicas. Isso envolve estudar e analisar esses algoritmos e o desenvolvimento e implementação de código de computador de alta qualidade capaz de processar grandes conjuntos de dados, possivelmente usando técnicas de computação de alto desempenho;

2. Desenho de um novo algoritmo;

3. Desenvolver um pacote de biblioteca Python para processamento de séries de tempo sísmicas com os algoritmos descritos acima. Esta etapa inclui estudar os desafios técnicos para construir novos módulos de processamento, testá-los e avaliá-los.

As principais contribuições que se originaram desta pesquisa de tese as seguintes:

- Estudo, implementação e optimização de métodos de processamento de sinais sísmicos introduzidos na secção 3;
- Desenvolvimento de uma biblioteca Python de alta qualidade Almeida [2021] para processamento de séries temporais sísmicas;
- Publicação e apresentação do artigo Almeida et al. [2021] na conferência EGU em Abril de 2021;
- Artigo submetido e aceite para publicação por IEEE Transações em Geociência e Sensing Remoto.
  Scimago Journal & Country Rank, Quartiles (2021): Q1 (Engenharia Eletrotécnica e Eletrónica & Ciências
  da Terra e Planetárias). Este está atualmente disponível como pré-impressão Mohammadigheymasi et al. [2021a]
  e no anexo B.

Esta tese apresenta uma visão geral sobre conceitos e técnicas consideradas fundamentais, assim como bibliotecas e *frameworks* utilizadas para o desenvolvimento da biblioteca proposta.
Como tal, é feita uma descrição de conceitos básicos de ondas sísmicas, provedores de informação apresentando as plataformas mais conhecidas, IRIS e Observatories and Research Facilities for European Seismology (ORFEUS). A plataforma ORFEUS é uma das mais antigas e importantes para a divulgação de *wave forms*. Actualmente muitas das plataformas como o ORFEUS encontram-se integradas na plataforma European plate observing system (EPOS) que agrega o acesso a muitos fornecedores de dados e serviços Europeus.
De seguida, é feita uma descrição de dois conceitos e três técnicas fundamentais no contexto deste trabalho, nomeadamente, processamento de sinais sísmicos, séries temporais sísmicas, séries de Fourier, transformada discreta de Fourier e transformada *wavelet*, respectivamente. Por fim, na última secção são apresentadas três bibliotecas que se assemelham ao trabalho que se pretende desenvolver, e que por sua vez, serviram como base para o desenvolvimento do mesmo, ObsPy, Noisi, NoisePy. Nesta secção, são também descritas as bibliotecas utilizadas na fase de implementação dos diferentes métodos, NumPy, Matplotlib, SciPy, Cython, Pytest.
É, também, apresentada uma revisão de literatura descrevendo os métodos e algoritmos que cada artigo contém. Os artigos escolhidos como mais relevantes para o trabalho desenvolvido foram organizados de forma cronológica e abordam diferentes tópicos relativos ao processamento de series temporais sísmicas: análise de sinais, análise de polarização, filtros de polarização e processamento de sinais sísmicos. Relativamente ao tópico de análise de sinais é apresentada a publicação feita por Flinn, em 1965, no que respeita o tópico de análise de sinal usando rectilinearidade e direcção do movimento das partículas. No tópico de análise da polarização destacam-se os artigos publicados por Vidale, em

1986, e Jurkevics, em 1988, onde é explorada a análise da polarização complexa do movimento das partículas e análise da polarização de dados de matriz com três componentes, respectivamente. De seguida no tópico de filtros de polarização foram escolhidas as publicações de Montalbetti e Kanasewich, em 1970, e Reading, em 2001, nestas é explorado o aumento da percepção das fases *body* tele-sísmicas com um filtro de polarização e filtragem da polarização para recolha automática de dados sísmicos e melhoria na detecção da fase convertida, respectivamente. Quanto aos tópicos de análise e filtros de polarização foi também escolhido o artigo publicado por Pinnegar, em 2006, que, por sua vez, se insere nos dois. Este explora a análise da polarização e filtro da polarização de um sinal com três componentes implementando a transformada S de frequência de tempo. Por fim, relativamente ao processamento de sinais sísmicos são de realçar as publicações de Bensen, em 2007, em que é explorado o processamento de dados de ruído ambiente sísmico para obter medições confiáveis de dispersão de ondas de superfície *broadband* e Wang, em 2019, que por sua vez apresenta discussões sobre o processamento do campo vectorial sísmico de multi-componentes.

No capítulo 4 é apresentada uma visão geral dos métodos implementados e sua incorporação na biblioteca Python desenvolvida. A primeira secção deste capítulo apresenta um resumo matemático do processo de implementação dos métodos publicados por Flinn, Vidale, Pinnegar e ainda o novo método desenhado no contexto do projecto SHAZAM. A segunda secção descreve a biblioteca desenvolvida nomeando as bibliotecas utilizadas e a sua finalidade, descrevendo as funções presentes em cada módulo presente na biblioteca desenvolvida, SeisPolPy Almeida [2021], e descrevendo o processo de distribuição e instalação da mesma.

O capitulo 5, por sua vez, apresenta uma discussão sobre a geração dos dados sintéticos usados, os resultados obtidos e o modulo de testes implementado na biblioteca desenvolvida. A primeira secção, apresenta o processo de geração e pre-processamento dos dados sintéticos para cada ferramenta utilizada, SPECFEM3D Globe e IRIS syngine service, descrevendo em detalhe cada passo tomado. De seguida, são explicados os resultados obtidos com a execução de cada modulo, usando os dados sintéticos gerados e pre-processados. Para cada resultado é também indicado o tempo de execução de cada modulo implementado para cada conjunto de dados. Por fim, na ultima secção, é apresentado o módulo de testes implementado. Neste, são descritos os testes efectuados à biblioteca que, por sua vez, vêm validar o seu bom funcionamento.

Em conclusão, o objectivo deste documento foi fornecer uma perspectiva detalhada sobre o desenvolvimento de *software* para análise de séries temporais sísmicas e uma descrição detalhada do trabalho desenvolvido. Com os resultados obtidos e os testes realizados, é possível verificar que todos os objectivos propostos foram alcançados com sucesso.

Chapter 1

Introduction

1.1 Problem Statement

By deploying seismic sensors at suitable sites the seismic waves generated by sources of seismic waves are recorded as a discrete function of time on regular time samples, the so-called seismic time series. The time series is a combination of effects of different parameters including the seismic source parameters, propagation environment, site effects, etc. Richards and Aki [1980]; Hutchings and Viegas [2012].
The study of seismic time series gives scientists a deep insight into these critical parameters but requires the development and design of well-defined processing algorithms to analyse the time series and extract the key information Bensen et al. [2007]; Wüstefeld and Bokelmann [2007]. In this regard, although the development of robust mathematical algorithms is a fundamental aspect, the computer platform and versatile programming languages for implementing the developed algorithms are of particular importance. Several state-of-art computer codes have been published, among them, ObsPy, a Python based and open-source framework, allowing the seismological community to contribute to its further development freely.
Several processing libraries have been developed based on the ObsPy software platform Lecocq et al. [2014]; Ermert et al. [2020]; Jiang and Denolle [2020]. ObsPy provides preprocessing modules for reading and converting various data formats on local or remote servers, applying band-pass filters for signal enhancement and noise cancellation, trend removal, deconvolution of the instrument response. It presents the time-series as an array compatible with the standard mathematical libraries like NumPy, SciPy, Matplotlib. Moreover, ObsPy compatibility for wrapping external shared C or FORTRAN libraries makes it an ideal toolbox to develop signal processing tools.
This research work aims to develop a state-of-art Python library consistent with ObsPy for implementing developed ideas to process seismic time series and, in this way, capacitate seismologists with a basis for further developing their ideas by building a bridge between pure ideas and their implementation. As the research work combines computer sciences and theoretical seismology, a presentation of the basic concepts in both fields is also presented in this thesis. The reader can refer to the book Ingle and Proakis [2016] and glossary USGS [b] which provides a more extensive and specific background knowledge relating to the subjects of signal processing and seismic events.

1.2 Objectives

The objectives of this thesis are as follows:

1. To implement and enhance the algorithms introduced byFlinn [1965]; Vidale [1986]; Pinnegar [2006] for signal enhancement and recovery of seismic time-series. This involves studying and analysing these algorithms and the development and implementation of high-quality computer code capable of processing large data sets, possibly using high-performance computing techniques;

2. Design of a new algorithm;

3. To develop a Python library package for processing seismic time-series with the algorithms described above. This step includes studying the technical challenges for building new processing modules, testing and evaluating them.

1.3 Contributions

This thesis project was undertaken within the context of the SHAZAM project which aims to solve problems arising from the limited seismic data available in the study region, namely the Gulf of Guinea, and the need to develop sophisticated data processing algorithms for extracting optimal information from the available seismic data. The present research work includes the study of existing methods and their techniques, to improve and optimise their accuracy and efficiency and the development of a new algorithm.
The main contributions from this thesis are:

- Study, implementation and optimisation of seismic signal processing methods introduced in section 3;

- Development of a high-quality Python library Almeida [2021] for processing seismic time-series available for the community on GitHub;

- An article, Almeida et al. [2021], published and presented at the EGU conference in April 2021. The Abstract, Poster and other digital materials are available at https://doi.org/10.5194/egusp egu21-15267. The poster is also available in annex C;

- An article submitted and accepted for publication by IEEE Transactions on Geoscience and Remote Sensing. Scimago Journal & Country Rank, Quartiles (2021): Q1 (Electrical and Electronic Engineering & Earth and Planetary Sciences). This is currently available as a preprint Mohammadigheymasi et al. [2021a] and in annex B.

1.4 Document Organisation

The present document is comprised of six chapters.

1. **First chapter** (Introduction) Presents the problem statement, the objectives to be achieved, expected contributions and the organisation of this document.

2. **Second chapter** (Seismic Time-Series: Background and Tools) presents a discussion of seismic time-series, in particular the scientific background and tools available.

3. **Third chapter** (Literature Review) Presents an overview of concepts and existing tools related to Seismic Time-Series. In this chapter, the focus is on presenting the techniques already applied, tools to be used in their implementation, and a review of selected papers related to this research work. The concepts presented, while crucial to the subject and its techniques understanding, are presented in more contextualising forms.

4. **Fourth chapter** (Implementation) introduces the implemented algorithms, describes the implemented functions and the building and installation process for the developed library.

5. **Fifth chapter** (Results) describes the steps undertaken to generate synthetic data, the results obtained with each implemented module and the implemented tests.

6. **Sixth chapter** (Main Conclusions and Future Work) Contains the main conclusions obtained during this thesis.

Chapter 2

Seismic Time-Series: Background and Tools

2.1 Introduction

This chapter presents an overview of seismic time-series. Section 2.2 describes the basic concepts of seismic waves, seismic signal processing, and the time-series techniques required to analyse the seismic time-series. Section 2.3 reviews open-source libraries and frameworks related to the subject giving some insight over their individual applications and usefulness in developing the proposed Python library.

2.2 Fundamental Techniques and Concepts

2.2.1 Seismic Waves

The shifting rock due to faulting in an earthquake, or the expanding medium due to an explosion, results in vibrations called seismic waves that propagate inside the earth or along its surface. Scientists use seismic sensors to record seismic waves. The recorded data can help scientists study the source of the seismic wave and the structure of the earth itself. In earthquake terminology, the place where the energy is released is called the Hypo center. It's normal projection to the earth surface is called Epicenter. There are three more crucial words of terminology namely, phase, rays, and wavefronts. Phase refers to a segment in a full-wave cycle in which a change occurs, for example, a reflection. A ray corresponds to the wave's direction. Wavefronts are the points in a wave with the same phase. Both of these, rays and wavefronts, are used separately to describe the propagation of a wave as described in Fig. 2.1.

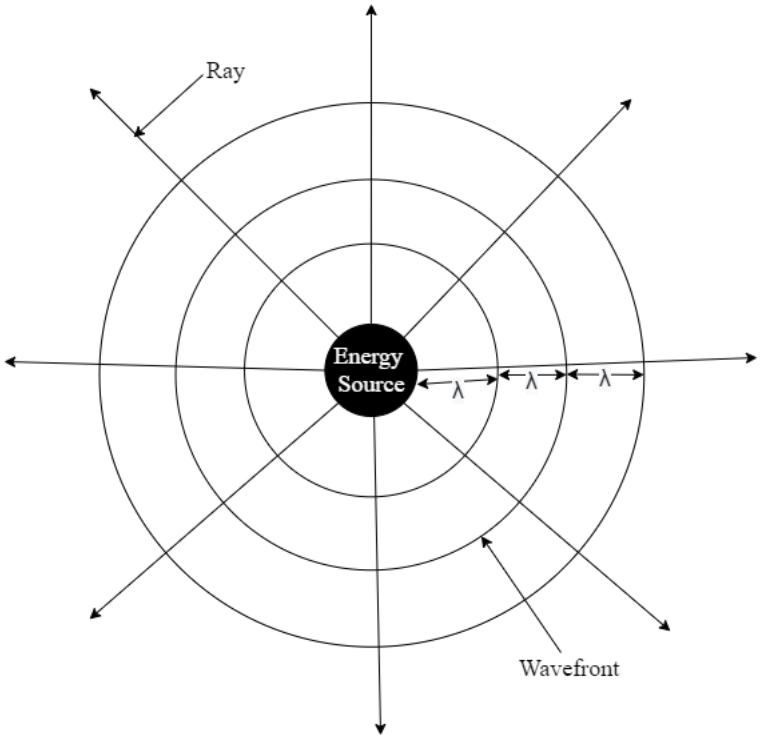


Figure 2.1: Wavefronts are the points in a wave with the same phase and rays the direction of propagation corresponding to a wave, as represented in this image the rays are always perpendicular to the wavefronts. The wavefronts are all equally distanced, being that same distance represented in the image by the λ symbol.

Depending on the type of particle motion due to the seismic wave, the so-called polarization, and the wave propagation environment, seismic waves are categorised into two main types, body waves, and surface waves.

2.2.1.1 Body Waves

In contrast to surface waves 2.2.1.2, which travel near the planet's surface, a body wave travels through the interior of the earth. This type of wave is further classified into two different types, P waves, and S waves USGS [b].

primary (P) waves, as described in Bent [2013a], are seismic body waves that propagate through the interior of the Earth. Primary waves get their name from the fact that they're usually the first waves a seismograph records. This type of wave particle motion is similar to a sound wave particle motion in that it consists of a series of compressions and dilatations parallel to the wavefront's propagation direction as shown in Fig. 2.2. P waves are recorded by the vertical (Z) and radial (R) components of the seismograph. They can move through liquids and gases, but at a far slower rate than they can through solids. In comparison to S waves and surface waves, P waves usually have smaller amplitude.

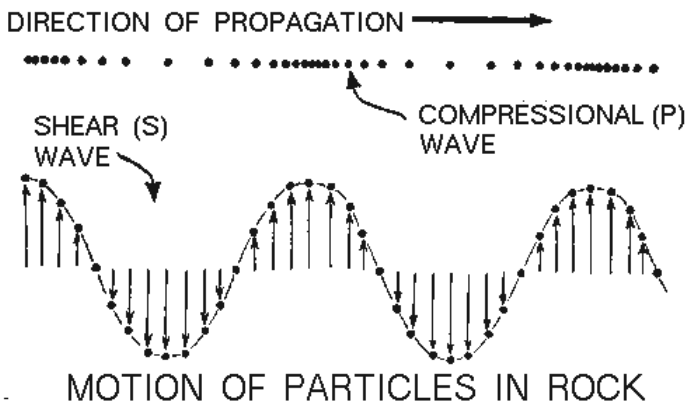


Figure 2.2: The motion of particles in rock during P and S waves is seen in this figure. The vibration of the rock, known as P or compressional waves, travels in the direction of propagation. Furthermore, with S or shear waves, the rock oscillates perpendicular to the wave propagation direction. This image was taken from USGS [a].

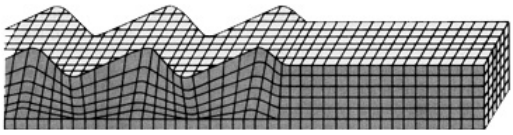
secondary (S) , like the P waves, are seismic body waves meaning they travel through the interior of the Earth. Their velocity is lower than of P waves, and they are usually the second major phase seen on a seismogram, earning them the designation secondary waves. S waves have a greater amplitude than P waves, and they can cause significant shaking and/or damage. As described in Fig. 2.2, these waves have particle motion that is perpendicular to the propagation direction. Furthermore, S waves are classified into two categories: SV waves, which occur on the vertical (Z) and radial (R) components of seismographs, and SH waves, which are registered on the transverse (T) component. The fact that S waves cannot travel through liquids or gases aided in the discovering that the outer core was liquid, Bent [2013b].

2.2.1.2 Surface Waves

Surface waves are classified as directed and dispersed waves. Their propagation is horizontal to the Earth's surface. The phase velocities of these waves is faster and these act like stationary waves in the vertical coordinate. Over a specified time period surface waves have the lowest velocity as written by V. Babuska in 1991 Babuska and Cara [1991].

The surface waves can be divided in two types, Love waves and Rayleigh waves.

Love wave these type of waves particle motion is rectilinear and purely horizontal and perpendicular to the direction of propagation which means the Love wave is only recorded in the Tranverse (T) and Radial (R) components, as shown in Fig. 2.3. SH waves are what make up Love waves.



(Image courtesy of European Center for Geodynamics and Seismology)

Figure 2.3: A Love wave is a surface wave possessing a horizontal motion which is transverse to the direction of the wave. This image was taken from USGS [b].

Rayleigh wave this type of wave is depicted in Fig. 2.4 below. These waves originate from interfering P and Sv waves. Their particle motion is elliptical which makes the Rayleigh wave occur only in the Vertical (Z) component of a seismograph that contains the wave propagation direction.

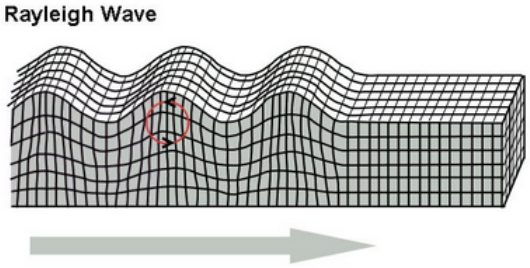


Figure 2.4: A Rayleigh wave is a seismic surface wave that causes the ground to move in an elliptical motion without a perpendicular or transverse motion. This image was taken from USGS [b].

2.2.2 Data Providers

Currently, there are several ways to access data related to seismic events due to the establishment of several platforms that make such information available.

The most commonly used platform is IRIS [b] which is a consortium of more than 100 US universities devoted to operating science facilities to acquire, manage and distribute seismic data. This platform offers many many services, for instance a seismic monitor IRIS [a] which allows the daily tracking of seismic events as shown in Fig. 2.5.

Another well known project and platform is the EPOS, which at the start of its development integrated the much older ORFEUS, shares data from many different earth science communities. EPOS much like IRIS makes seismology data and products available through web services integrated within its data portal. An example of these services is shown in Fig. 2.6. EPOS also has a data portal EPOS [2019] which allows its users to access a workspace, after logging into their account, as well as data search and download interactive tools. Integrated tools to process data within the EPOS workspace are currently in development.

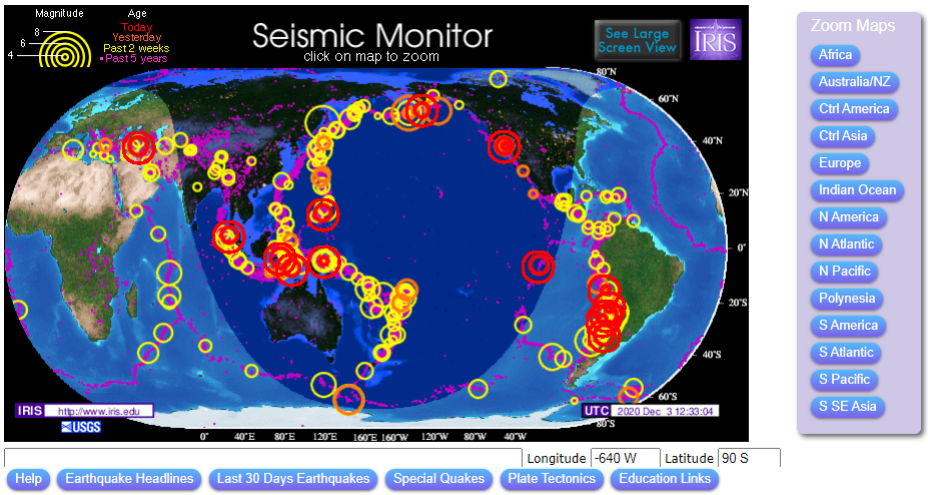


Figure 2.5: Seismic monitor web service provided by IRIS available at IRIS [a].

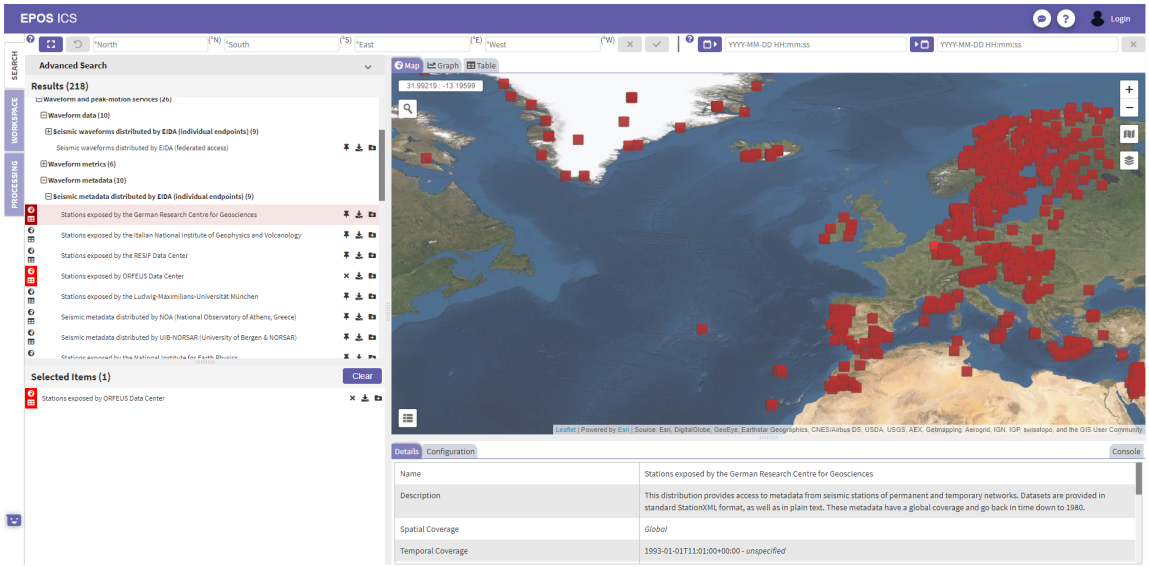


Figure 2.6: EPOS web service for data search and download EPOS [2019].

The data stored and made available by these platforms can be extracted for research purposes via requests to some API or by using high level frameworks (such as ObsPy Beyreuther et al. [2009] described in the 2.3.2.1 section). Data requests are made by giving input information like the seismic event name, time period, seismic station, and other optional details.

For this thesis, most of the data used has already been published and made accessible through platforms like the previously mentioned IRIS. In particular, information was collected from countries bordering the Gulf of Guinea and specifically Cameroon. More seismic data is being acquired from seismic stations located in São Tomé and Principe under the SHAZAM project, and which is expected to be stored on Collaboratory for Geosciences (C4G) (Universidade da Beira Interior (UBI)) servers.

2.2.3 Seismic time-series

A time-series is no more than a series of data points ordered in time where time is usually the independent variable (independent in the sense that its variation does not depend on any another variable) Peixeiro [2019].

When dealing with time-series, there are a few aspects to consider, such as seasonality, auto-correlation, and stationary. Seasonality refers to the periodic fluctuation in the data (and can be derived from auto-correlation) and if the time-series plot has a sinusoidal shape. To identify seasonality it is enough to look at the period of some given effect, which gives us the season's length.

Auto-correlation indicates the similarity between the data points as a function of the time separating them.

The last aspect to consider is the term "stationary", which is attributed to a time-series if the statistical properties do not change over time, which is the same as saying that mean and variance are constant and co-variance is independent of time.

Many modelling techniques are employed to predict "trends" in a time-series, such as moving average, exponential smoothing, and AutoRegressive Integrated Moving Average (ARIMA). These are some of the modelling techniques presented and explained in the article Kehtarnavaz [2008].

These will not be further explained since they are not the focus of this thesis and are here only to provide an example of what is available in this field of study.

2.2.4 Digital Signal Processing

Nowadays, we are surrounded by an incredible spectrum of signals in various forms, with a natural or human-made source. Most often, the desired signals are contaminated by unwanted or unnecessary signals called noise. The goal in science and engineering is to extract the information from a mix of useful and unwanted signals. Signal processing is a tool to obtain this goal. In the other words, signal processing is an operation designed to extract, enhance, store, and transmit useful information. The signals we encounter in practice are often an analog and continuous function of time (or space). Two basic methods of signal processing have been introduced. The traditional Analog Signal Processing (ASP) by using electrical active and passive circuits, and the more advance method, Digital Signal Processing (DSP). DSP is applied on a digital signal, which is a binary valued form of analog signals at specific instances in time Ingle and Proakis [2016]. DSP operations are based numerical methods methods.

DSP is done through the implementation of mathematical techniques like the Fourier transform (FT) and the wavelet transform. Since the information in every signal is different the usefulness of each employed technique is subjective to the specific implementation goals Ingle and Proakis [2016].

2.2.5 Fourier Series

The Fourier series was introduced originally by Jean-Baptiste Joseph Fourier as a tool to expand a time- or space-domain signal in terms of pure sine and cosine functions. The orthogonal property of the base function in the Fourier series is used to make the transformation invertible and lossless.

2.2.6 Discrete Fourier Transform

Here, we shortly review the mathematical formula of the Fourier transform. Suppose that

$$\boldsymbol{x} \in \mathbb{R}^{L\times 1}, \quad (2.1)$$

is a seismic time-series which samples the wavefield arriving at a sensor along the time axis on $t_k = k\delta t$, δt is the sampling interval and $k = 0, 1, ..., 2n$ is the time index assuming an odd length $L = 2n + 1$ for the seismogram. Then, the discrete Fourier domain, $\boldsymbol{x}^f = \boldsymbol{\mathcal{FT}}\{\boldsymbol{x}\} = \boldsymbol{f} \in \mathbb{R}^{L\times 1}$, is obtained by modulating $\boldsymbol{x}$, with pure sinusoids having discrete frequencies as follows,

$$f(l) = \frac{1}{2n+1}\sum_{k=0}^{2n} x(k) exp(\frac{-2\pi \jmath kl}{2n+1}). \quad (2.2)$$

Here, $l = -n, ..., n$, is the frequency index giving frequency content of the signal on discrete frequencies $\omega_f = \frac{f}{(2n+1)\delta t}$, and $\jmath$ is the imaginary symbol. The original signal is obtained by applying the inverse Fourier transform, $\boldsymbol{x} = \boldsymbol{\mathcal{IFT}}\{\boldsymbol{f}\}$,

$$x(k) = \sum_{l=0}^{2n} f(l) exp(\frac{2\pi \jmath kl}{2n+1}). \quad (2.3)$$

The FT therefore consists of a mathematical technique to expand the signal in order to facilitate its analysis on system performance and representation. Using this technique, we obtain an alternative representation of the signal Madisetti and Vijay [2018]; Sevgi [2014] and this can then be used to decompose a given signal into different components so that the signal processing can run more effectively.

2.2.7 Wavelet Transform

The ordinary FT presents a frequency spectrum for the stationary signals, however, it can not resolve the non-stationary characteristics of real-world signals. To circumvent this, several time-frequency representation methods have been introduced in the literature Stockwell [2007,?]. The Wavelet Transform is a tool to extract the local time-frequency information of the signal. It brings a new perspective over signal processing not just for the improvements achieved but also for being simple to implement Chun-Lin [2010].

$$Wf(s,u) = \int_{-\infty}^{\infty} f(t)\frac{1}{\sqrt{s}}\psi^*(\frac{t-u}{s})\,\mathrm{d}t. \quad (2.4)$$

The wavelet function is designed to obtain an equilibrium between time and frequency domain. Due to this functions ability of multi-resolution it outperforms Fourier-based transforms. This is achieved by locating time and frequency with more precision as stated by Chun-Lin [2010].

2.3 Python Libraries and Seismic Frameworks

Quite a few libraries and frameworks are currently available in Python that are related to implementing and using the methods presented so far.
There are libraries which are used to facilitate the implementation of mathematical functions, such as the Fast Fourier Transform, or facilitate the implementation of testing and third party library integration. These are in particular NumPy,SciPy, Pytest and Cython. Others serve as tools for presenting the results from processing the raw data, Matplotlib is one widely used such library.
There are many Python frameworks specific to seismology. In particular ObsPy which contains many important tools for fetching and processing data and is in fact almost a standard for Python software development in the seismology field. Other frameworks studied were Noisi, NoisePy, and MSNoise, these served as a basis for the development of SeisPolPy, in that they served as a way of learning and studying best practices in the field of library development and documentation.
The libraries and frameworks presented in the following subsections were essential for the development of the seismic time-series processing software and will now be described.

2.3.1 Python Libraries

2.3.1.1 NumPy

Numpy was created by Oliphant [2010] and currently serves as the basic open-source package for scientific computing using the Python programming language. It offers its users a powerful N-dimensional array object, sophisticated (broadcasting) functions, tools for integrating C/C++ and Fortran code, as well as numerical algorithms for linear algebra, Fourier transforms, and random number generation.

2.3.1.2 Matplotlib

Matplotlib is a library for creating static, animated, and interactive visualisations in Python, as shown by the Fig. 2.7 plots taken from the website referenced at the beginning of this subsection, Hunter et al. [2012].

Figure 2.7: Examples of some of the possible plot visualisations using Matplotlib library. This image was taken from Hunter et al. [2012].

2.3.1.3 SciPy

SciPy is an important open-source package for signal processing, SciPy [2012a]. It makes use of other packages, such as NumPy, Matplotlib, SymPy and the Pandas packages. From each of these it makes use of some features, like the NumPy N-dimensional array, the Matplotlib comprehensive 2D plotting, and the SymPy symbolic mathematics pandas data structures and analysis algorithms. SciPy can be summed up as a collection of mathematical algorithms and functions built on NumPy , SciPy [2012b]. Through these, it adds power to Python by entrusting to the user high-level commands and classes for visualisation and manipulation of data.

2.3.1.4 Cython

Cython is a Python extension that allows explicit type declarations and the direct compilation of Python to C code. Using this library it's possible to address Pythons large overhead for numerical loops, as well as the difficulty in making efficient and simplified use of existing C or Fortran code Behnel et al. [2010].

2.3.1.5 Pytest

Pytest is a Python testing tool which can be used for all types and levels of software testing. It contains powerful features such as "assert" rewriting, a third-party plugin module and a simple but powerful fixture model Okken [2017].
This tool, being a software test framework, is a command-line tool that automatically finds any tests written by its users, runs them, and reports the results. It also has a library containing various functions to make testing more effective. Furthermore, Pytest can be extended by writing new plugins or installing third-party plugins.

2.3.2 Python Seismic Frameworks

2.3.2.1 ObsPy: A Python toolbox for Seismology

ObsPy, as a toolbox for seismology, has the objective of filling the existing gap between analysis and automatic data acquisition systems. This tool is open-source and extends Python by supplying the users basic seismological routines and by storing data in NumPy n-dimensional arrays. This allows the use of powerful modules for numerical array programming such as, Numpy (2.3.1.1) and SciPy (2.3.1.3).

ObsPy provides a unified access to read/write seismograms with MiniSEED, GSE2, SAC, SEISAN, Seismic Handler formats Q and ASCII format. All the data regarding this aspect is stored as a stream object, which means that multiple seismic data records can be stored in one file. Each record can be accessed separately. Furthermore, the data can then be processed after being stored or, via Python Ctypes library, the ObsPy routines can be shared to any C or FORTRAN library.
Among many modules provided by this tool, one of the most important is XSEED. This module provides users with the capability of converting data from Dataless SEED, Full SEED volumes, and XML-SEED and back, thus supporting the IRIS data supplier formats. ObsPy supports many different formats and this gives it the possibility of connecting to different kinds of servers used by data-centers (IRIS, ORFEUS, etc.) and to read data from local files or to import data directly from data centers by only specifying the desired combination of labels.
Beyreuther et al. [2010] state that ObsPy contains only basic signal processing routines. These do however allow data centers to process some of the most common processing steps and visualise the results obtained via the Matplotlib (2.3.1.2) package.

2.3.2.2 Noisi

Noisi is an open-source Python tool which allows (forward and inverse) modelling of ambient seismic cross-correlations that contain a spatially varying source spectra Ermert et al. [2020]. This becomes possible by making use of pre-computed databases for the representation of seismic wave propagation between the ambient seismic source and receivers. This tool also depends on ObsPy module for accessing the seismological waveform data.
The main purpose of Noisi is the study of ambient seismic sources while taking into account the effects of realistic wave propagation. Moreover, it can be used for guidance in terms of interpreting ambient seismic auto and cross-correlations.

2.3.2.3 MSNoise: A Python Package for Monitoring Seismic Velocity Changes Using Ambient Seismic Noise

MSNoise is also an open-source, well commented and documented Python integrated solution. It is designed to be fully cross-platform and has a processing workflow separated into steps for any user to replace the code in any step. However, after the replacements, inputs, and outputs must be respected for the tool to still function properly.
The article published by Lecocq et al. [2014] aims to present an overview of all the processing workflow steps from the computer scientist's perspective. It has also shown a validation of the proposed toolbox (MSNoise) using archived seismic data related to a volcano where precursory changes in seismic velocity had been observed before two known eruptions.
As stated in the reviewed document and citing T. Lecocq in 2014, "MSNoise is the first complete software package for computing and monitoring relative velocity variations using ambient seismic noise. MSNoise is a fully integrated solution that automatically scans

data archives and determines which jobs need to be done whenever the scheduled task is executed".
The authors also affirm that the testing results have verified the usefulness of the tool. These results show that MSNoise is extensible and a useful tool for researchers by allowing them to focus only on implementing the codes related to processing and benefit from the framework. Furthermore, they claim that, although the presented validation is in a volcanic environment, this does not mean that its application is restricted to this specific case.

2.3.2.4 NoisePy

NoisePy is defined as a new Python open-source, high-performance tool design to target large-scale ambient noise seismology in the reviewed article. This tool provides most of the current processing techniques developed for ambient field data and literature correlations, parallel download routines, dispersion analysis, and monitoring subroutines.
NoisePy, as stated by Jiang and Denolle [2020], overcomes MSNoise (2.3.2.3) by achieving a 4-time improvement in computational time over the computing of cross-correlations. After undergoing successful testing, this tool has proved capable of handling data with sizes ranging from gigabytes to terabytes. This is achieved by taking advantage of a parallel input/output enabled Hierarchical Data Format (HDF) 5 data format designed specifically for seismology. The use of this data format results in a structured organisation of cross-correlation data. Furthermore, NoisePy obeys the computation of noise correlations over time windows parallelism by using Message Passing Interface (MPI) and thus achieving strong scaling based on the number of available cores. It also takes advantage of some Python libraries' functionalities to achieve its goals. These libraries are ObsPy (2.3.2.1), as well as Numpy (2.3.1.1), SciPy (2.3.1.3), mpi4py, pyASDF and Numba.
In the present document, the authors present the workflow, functionalities, and computation performance of the tool. It is also presented a single-core analysis made on an iMac and other performance tests. These originated good computational times even for tasks with a data size of 10+terabytes. The authors expect to reduce the computation time of the tool when they achieve the use of GPUs for the correlation step. Thus, satisfying the growing influence of architectures combining CPUs and GPUs in large, more recent clusters.

2.4 Conclusions

In this chapter a description of the fundamental seismic concepts required for this dissertation has been presented. A description of the data providers concerning seismic events was given. A brief overview of seismic time-series, and crucial techniques regarding digital signal processing were also discussed. This chapter also contained an overview of libraries and frameworks most commonly used for implementing the methods described throughout the present document. All of these libraries and frameworks will be used throughout this research in order to achieve the proposed goals in 1.2.

Chapter 3

Literature Review

3.1 Introduction

A seismic wave is a mechanical disturbance or energy packet generated by a sudden release of energy such as that from an earthquake, volcanic eruption, or explosion. The waves propagate through the layers inside the Earth Aki and Richards [2002]. Due to the Earth's non-homogeneous and anisotropic characteristics, the generated seismic waves undergo different physical phenomenons, including reflection, refraction, dispersion, and diffraction Udias and Buforn [2017]. As a result, different seismic phases are generated and propagated throughout the Earth's physical layers. The propagated wave-field is recorded by seismic sensors installed in specific seismic stations. These sensors usually consist of three components, each component records particle motion in a particular orientation (usually East, North, and Upwards).
The recorded wave-field provides input data for studying the Earth in different scales, from shallow depth explorations to large-scale studies that aim to image the core, crust or mantle mechanisms. Extracting information from these seismic time series demands sophisticated processing and inversion methods to convert the raw data to a physical model that seismologists and geologists can interpret Bensen et al. [2007]; Jurkevics [1988]; Yilmaz [2001]. These numerical methods often have high computational costs and need up-to-date hardware and software facilities. Developing new numerical methods on one hand and using new programming languages and software platforms on the other profoundly facilitates the optimal extraction of information from the raw data.
This research aims to address both needs by developing a Python based library in the ObsPy platform to implement and optimize time series processing algorithms. Taking advantage of all the available pre-processing modules of ObsPy and its compatibility with the numerical libraries like SciPy, Numpy, and Matplotlib, the aim is to develop high-quality algorithms that can be used on real and synthetic seismic data for optimal extraction of desired information. This thesis is part of the SHAZAM project, which studies the Gulf of Guinea as the target area.
Each section of the present chapter will review the literature on methods and algorithms. The literature chosen as the most relevant for this research work are, in chronological order, as follows:

- E. Flinn, "Signal analysis using rectilinearity and direction of particle motion", Proceedings of the IEEE, 53(12):1874–1876, 1965.18,19; Flinn [1965]
- J. F. Montalbetti and Ernest R Kanasewich, "Enhancement of teleseismic body phases with a polarization filter", Geophysical Journal International,21(2):119–129,1970.17,18,19; Montalbetti and Kanasewich [1970]

- J. E. Vidale, “Complex polarization analysis of particle motion”, Bulletin of the Seismological society of America, 76(5):1393–1405, 1986. 22. Vidale [1986]
- A. Jurkevics, “Polarization analysis of three component array data”, Bulletin of the seismological society of America, 78(5):1725–1743, 1988; Jurkevics [1988]
- A. Reading, et al. “Polarization filtering for automatic picking of seismic data and improved converted phase detection”, Geophysical Journal International 147.1 (2001): 227–234; Reading et al. [2001]
- R. Pinnegar, “Polarization analysis and polarization filtering of three component signals with the time—frequency S transform”, Geophysical Journal International,165(2):596–606, 2006; Pinnegar [2006]
- R. H. Herrera, et al. “Body wave separation in the time-frequency domain”, IEEE Geoscience and Remote Sensing Letters,vol. 12, no. 2, pp. 364–368, 2014; Herrera et al. [2014]
- C. Wang, et al. “Discussions on the Processing of the Multi-Component Seismic Vector Field.” Applied Sciences 9.9 (2019): 1770; Wang et al. [2019]

3.2 Signal Analysis Using Rectilinearity and Direction of Particle Motion

In this paper a processor for three-component seismic records designed to enhance particle motion both in rectilinear and in any particular direction in a three-dimensional space is discussed, Flinn [1965].
This processor has the advantage of separating the body waves motion from the surface waves motion when both distance and azimuth to the seismic source is specified. The processor is able to isolate and identify the pulse P (pP) that propagates practically the same path as the first arrival detected by the three-component sensors and arrives simultaneously as a compressional pulse. The pP referred previously is the one that propagated vertically, directly from the seismic source, and was reflected from the surface. Identifying this pulse is essential due to the time interval’s dependence between pP and P with the source’s burial.
This scheme’s application has some details that need to be considered, such as the slight frequency-dependent phase shift. This problem, while it should be considered since the records have been linearly processed to achieve a high ratio of signal to ambient noise, it’s to be expected that the mutations of the data on the original seismogram into particle motion polarization results will be useful and more importantly meaningful to the scientist analysing them.
Even though the work related to this document has been implemented, as will be seen in the following sections 3.3 3.5, what E. Flinn in fact describes in this paper is a general idea and not an actual algorithm.

3.3 Enhancement of Teleseismic Body Phases with a Polarisation Filter

The article proposes a modification to the previously described polarization filter described by Flinn [1965]. The authors modified the filter to increase the signal to noise ratio of seismic body phases, Montalbetti and Kanasewich [1970].

The authors obtain rectilinearity and direction of particle motion by applying the covariance matrix to the three components data of ground movement during a small interval of time. By diagonalizing the matrix and creating a function that contains the ratio of the intermediate and most extensive principle axes J. Montalbetti and E. Kanasewich find the estimation of rectilinearity. On the other hand, the polarization direction is obtained from the eigenvector largest principle axis. With these, a set of time-varying operators is obtained through which control is achieved to modify the digital seismic records.

With these filters, it is possible to enhance most of the compressional or shear phases.

According to the authors of this article, and the resulting data, applying the filter to an array of three-component stations makes it possible to identify multiple events on the source of some earthquakes. It is then possible to conclude that this polarization filter enhances phases with well-defined properties, making it very useful in processing teleseismic data by making these events easier to correlate and interpret due to the signal character distortion being reduced over time.

3.4 Complex Polarization Analysis of Particle Motion

In this article the author, J. E. Vidale, proposes a method that identifies the problem regarding multiple arrivals by presenting a low degree of polarization and by being able to handle elliptical polarization. With the reviewed article the author essentially aims to extend the scheme presented by Montalbetti and Kanasewich [1970], reviewed in 3.3, to analytic three-component seismograms. This is achieved by having the imaginary part of the signal has the Hilbert transform of the real part. The author affirms that obtaining elliptical polarization was only made possible by making use of the analytic signal.

Furthermore, the complex polarization filter shown throughout this paper allows three-component seismogram routine analysis to yield the wave type of the arrivals. The data present within the obtained analysis will aid in the interpretation of strong motions recorded in a given seismic event Vidale [1986].

With the results, obtained and presented by the author, he concludes that *S* body waves, which are converted into surface waves upon striking basin structures, supply energy that makes the basins shake for a longer duration of time when compared to surrounding mountains.

3.5 Polarization Analysis of Three-component Array Data

A technique for polarization analysis which can be applied to data recorded from a three-component sensors array or a single sensor Montalbetti and Kanasewich [1970]. This technique is also based on the original algorithm proposed by Flinn [1965].
The importance of three-component recordings is the importance of monitoring regional seismic activity as the regional phases can exhibit large horizontal motions. Moreover, while simple three-component seismograms help interpret these events, a more complete and accurate analysis requires processing the signals and extracting/enhancing the polarization content.
The author describes two different approaches to analysing three-component data. The first is the application of a non-linear filter based on the data polarization content and output the modified seismograms. The second comprises in estimating parameters of some model already fitted to the data in a time-varying manner.
The extensions to E. Flinn presented in the paper were the application to an array of sensors (three-component) and frequency decomposition. This technique consists in filtering the signals into a series of small frequency bands applying short time windows and finally computing, for each window, the polarization ellipse from the covariance matrix. Through this process are produced a series of attributes representing particle motion as time and frequency. The covariance matrices are averaged for all arrays before finding the eigenproblem solution to apply this technique to an array of sensors.
The objective when using this technique was to examine the polarization properties of regional seismic phases by exploiting signal to noise advantage gained by using an array of sensors.
The results, obtained by the author, showed that crucial factors were the chosen time and frequency, as well as the purest polarised motions, in a given phase. According to A. Jurkevics in 1988, *"The use of an array of three-component sensors was found to reduce the estimation variance of polarization attributes by 1/M, where M is the number of sensors when the noise and scattering distortions are uncorrelated between sensors."* Jurkevics [1988].

3.6 Polarization Filtering for Automatic Picking of Seismic Data and Improved Converted Phase Detection

With the published document, it is proposed the improvement of picking speed, of the already developed methods, and converted phase yield, Reading et al. [2001]. The authors also indicate that this work is most beneficial when dealing with data sets used for multi-phase tomography.
This improvement came since, after phase conversion, the data analysts had to handpick the information related to the converted arrivals from extensive data sets, which was very time-consuming due to the interference of P-coda waves, which lead to the fact that many converted arrivals could not be picked. To be able to improve the picking process as well

as pick correct converted phase arrival times, the authors applied automatic picking as well as polarization filtering.
Polarization filtering is applied to three-component data to improve phase identification and facilitate automatic detection. For different types of problems, the are different methods to apply in order to separate the waves arriving, methods like the ones applied in Jurkevics [1988], and Montalbetti and Kanasewich [1970] and reviewed in the previous sections. In this paper, the application of a polarization filter is made to improve the automatic picking of direct waves and mode converted PS waves and SP waves arrivals. With this goal, A. Reading in 2001 applied the data-adaptive filters, which may be built under the premise that the noise is less polarised than that signal of interest. A time-domain picking-method follows these filters to improve the automatic detection of direct and converted phase arrivals.
According to the authors, "98 percent of the P arrivals and 95 percent of S arrivals picked manually from high-pass-filtered records were successfully picked automatically after polarization filtering. Many more intermediate arrivals can be seen on the polarization filtered data compared to those on the manually picked traces. Without polarization filtering, they number 455 SP and 83 PS phases; with filtering, this count improves to 528 SP and 482 PS phases.". The converted waves' energy is more linearly polarised than the scattered energy, making them easier to identify and pick.

3.7 Polarization Analysis and Polarization Filtering of Three-Component Signals with the Time–Frequency S Transform

With the article, here reviewed, the author presents an application of windowed Fourier transforms, such as the S transform, to design a signal-adaptive polarization filter. The S transform contributes to the filter to be designed by obtaining the time-frequency spectra of the polarization characteristics from three-component seismic signals. The obtained spectra can then be used to separate the entire signal in "circular" and "linear" parts. Using this Fourier transform and the obtained parts it is possible to design a filter which targets specific polarization properties, Pinnegar [2006].
In the paper the author presents, in detail, the mathematics regarding developing the signal-adaptive polarization filter. Furthermore, it is also included three similar examples in which the designed filter aims to remove the Rayleigh wave from a three-component seismic signal, which, has shown by the results, is achieved with success.

3.8 Body Wave Separation in the Time-Frequency Domain

Dividing a seismogram into its constituent phases has long been a challenge. In a way to overcome this challenge, the authors employ a high-resolution time-frequency transform and recreate their individual waveforms in the time domain. Due to the employed trans-

form, named synchrosqueezing transform which consists in a derivation of the Continuous Wavelet Transform (CWT), they obtained high-resolution decomposition's map in the time-frequency domain which enabled the identification and separation of P and S waves. The transform algorithm used can be inverted like any other Fourier transform which means that after the separation the signal can be reconstructed in the time domain. Furthermore, Herrera et al. [2014], state that due to blurring in the frequency domain the previously mentioned contents would not be recoverable using STFT. After obtaining the results, and analysing them, was concluded that the employed method brings out additional useful information that until now the traditional methods could not obtain. Furthermore, the authors highlight the fact that their proposed method can also be utilized successfully when the different phases present in a seismic waves overlap in time. Although, these need to be separated in terms of their frequency content.

3.9 Discussions on the Processing of the Multi-Component Seismic Vector Field

Multi-component seismic data contain considerable information about the medium the seismic waves propagate through. The methods developed for processing the vector field information have been developed but still lack effectively retaining and using it. One of the reasons this is yet to be achieved relates to the fact that the processing techniques currently treat the individual agents independently as a scalar field. While correct, this approach creates the problem of restricting potential utilities of the information by not excavating the vector characteristics of the wavefield.
The article's focus, written by Wang et al. [2019], is to present a summarised discussion of the already developed pre-processing techniques, such as polarization filtering, denoising using vector order statistics, group sparse representation, and vector separation of compressional waves and shear waves. With this, the authors hope to help researchers develop more field processing methods and also incite the development of vector processing techniques for multi-component seismic data.

3.10 Conclusions

In this chapter, the techniques and methods for processing and pre-processing data from seismic events were described. Through each reviewed paper has become possible to perceive the basis of the currently employed techniques and the improvements made to overcome the difficulties faced by researchers, such as efficiency, accuracy, or even the application of these techniques.

Chapter 4

Implementation

4.1 Introduction

This chapter is an overview of the methods implemented and their incorporation into our Python library. The first section 4.2 presents a mathematical summary of the implementation process for the different algorithms. The second section 4.3, describes the developed library by naming the libraries utilised, describing the functions present in each module and describing the library building and installation process.

4.2 Implemented Algorithms

The methods introduced by Flinn [1965], Pinnegar [2006], and Vidale [1986] were chosen as the main modules of SeisPolPy library. Furthermore, a newly developed seismic polarization analysis method, a regularised sparsity-based approach (RS-TFR) Mohammadigheymasi et al. [2021a], was implemented in SeisPolPy library. This new method was developed as part of the SHAZAM project with the purpose of improving on separation of the different seismic phases. In the following, we give an in-detailed explanation of the algorithms implemented, accompanied by some numerical examples.

4.2.1 Flinn Method

The present method was designed by Flinn [1965] to perform signal analysis by making use of rectilinearity and direction of particle motion, as described previously in 3.2. This method, unlike any other here mentioned, obtains the information necessary without the need to transform the signal from the time domain to the time/frequency domain. For this method, the first step is to separate the three components of the signal. Next, a rolling window is applied to each component of the data to obtain samples. For this second step, a Gaussian window is used, which is defined as

$$w(n) = e^{-\frac{1}{2}(\frac{n}{\sigma})^2}. \quad (4.1)$$

The third step for the implementation of the Flinn method consists in a part procedure. Firstly we obtain the covariance matrix, C, by examining the N-dimensional sample, $X = [x_1, x_2, x_3]^T$ (in which each x corresponds to one of three components). Thus we can obtain the covariance matrix elements C_{ij} (which are the covariance of x_i and x_j) as follows,

$$C = \frac{\mathbf{X}^T\mathbf{X}}{N}. \quad (4.2)$$

The second step consists in finding the eigenvalues ($\lambda_1\lambda_2, \lambda_3$) and eigenvectors ($\mathbf{u}_1\mathbf{u}_2\mathbf{u}_3$), which are nontrivial solutions to,

$$(\mathbf{C} - \lambda^2\mathbf{I})\mathbf{u} = 0, \quad (4.3)$$

where $(.)^T$ denotes the transpose operator, $\mathbf{I}$ a three-by-three identity matrix, and 0 a column vector containing zeros only.
The fourth, and final step, consists of creating rectilinearity and direction of particle motion arrays. For rectilinearity, this is done for each sample and after obtaining the respective eigenvectors and eigenvalues, by calculating

$$Rec[i] = (1 - (\frac{\lambda_2}{\lambda_1})), \quad (4.4)$$

with i = 0, 1, ..., N (being N the total number of samples), and adding the result to the rectilinearity vector, Rec. As for the direction of polarization motion, the process is similar, but instead, we have a vector for each component and a coordinate transformation operator for each component,

$$Cto_{x_1} = [\cos(inc), \sin(inc) \times \sin(\pi), \sin(inc) \times \cos(\pi)], \quad (4.5)$$
$$Cto_{x_2} = [\sin(inc), \cos(inc) \times \sin(\pi), \cos(inc) \times \cos(\pi)], \quad (4.6)$$
$$Cto_{x_3} = [0, \cos(\pi), \sin(\pi)], \quad (4.7)$$

with inc as $(\frac{0}{180})\pi$. Then at each sample the following operations are performed,

$$Dir_{x_1}[i] = u_1^T \times (Cto_{x_1})^T, \quad (4.8)$$
$$Dir_{x_2}[i] = u_1^T \times (Cto_{x_2})^T, \quad (4.9)$$
$$Dir_{x_3}[i] = u_1^T \times (Cto_{x_3})^T, \quad (4.10)$$

with i = 0, 1, ..., N. With the previous operations finished, we then obtain the direction of particle motion for each component and the rectilinearity.

4.2.2 Vidale Method

An analytic-signal-based polarization analysis was introduced by Vidale [1986]. The method has the advantage of having only one free parameter, the time-window length, in which the polarization parameter is estimated. The azimuth and the dip of the direction of parti-

cle polarization are calculated with a high degree of precision. This method distinguishes itself from every other method here presented by making use of the Hilbert transform. Both this method as the previously described method designed by Flinn [1965], although very old, still posses an immense value not just historically but also in terms of the validity of the information we can can extract from the signals.

The first step for implementing this method is very similar to the process described in 4.2.1, in which we start by separating the three-components of the seismic signal. However, in the present method, the separated components undergo the Hilbert transformation.

The Hilbert Transform (HT) is defined as,

$$\hat{x}t = H[x(t)] = \frac{\int_{-\infty}^{\infty} \frac{x(t)}{t-\tau} d\tau}{\pi} = x(t) * \frac{1}{\pi t}, \quad (4.11)$$

for any signal x(t), where "*" corresponds to the convolution operation.

The second step for the Vidale method implementation consists of windowing the signal components data to obtain samples, which is achieved by making use of the Gaussian window (4.1), in the same way as described in the previous section 4.2.1.

We compute the covariance matrix for the third step of implementation to obtain the three eigenvalues λ_i and the eigenvectors (x_i, y_i, z_i). The computed eigenvector (x_0, y_0, z_0) corresponding to the largest eigenvalue λ_0 points in the direction of the most considerable amount of polarization. Nonetheless, the phase in the complex plane of an eigenvector is arbitrary.

The fourth step is to obtain the length of the eigenvectors real component, where

$$X = \sqrt{(Re(x_0 cis\alpha))^2 + (Re(y_0 cis\alpha))^2 + (Re(z_0 cis\alpha))^2} \quad (4.12)$$

and,

$$cis\alpha = cos\alpha + isin\alpha \quad (4.13)$$

and $Re(x)$ the real part of x. To maximise the value of X, before computing the equation in (4.12), the eigenvector is normalised to have length 1, and then the eigenvector (x_0, y_0, z_0) must be rotated by 0º to 180º. Through this rotation in the complex plane, the largest real component can be found. The rotation in question can be found by searching the values α = 0º to 180º.

The fifth and final step is implementing the equations that will generate the final polarization results, which will enable the analysis of the seismic signal. For the first equation the eigenvector (x_0, y_0, z_0), after being rotated by the angle α, is utilised to compute the elliptical component of polarization defined as,

$$P_E = \frac{\sqrt{1 - X^2}}{X}. \quad (4.14)$$

If P_E is 1, then we have a circularly polarised motion. If this value is 0 then, we have a linearly polarised motion. In cases of ambiguity, these resulting values are usually used for defining the direction of propagation corresponding to the Rayleigh waves. The second equation implemented corresponds to the strike of the direction of maximum polarization, which is defined by Vidale as,

$$\phi = tan^{-1}\left(\frac{Re(y_0)}{Re(x_0)}\right). \quad (4.15)$$

The third equation corresponds to the dip of the direction of maximum polarization and has been defined as,

$$\delta = tan^{-1}\left(\frac{Re(z_0)}{\sqrt{Re(x_0)^2 + Re(y_0)^2}}\right). \quad (4.16)$$

The author states that the dip and strike equations range from -90º to 90º, and 0º representing a vector that points horizontally in the direction of the epicenter. Comparably, the intermediate and smallest eigenvectors λ_2 and λ_1 both point in the direction of the intermediate and minimal amount of polarization, correspondingly.
The last equation measures the strength of polarization in the given seismic signal. This equation was defined by the author as follows,

$$P_s = 1 - \frac{\lambda_1 + \lambda_2}{\lambda_0}. \quad (4.17)$$

If the result is close to 1, it is possible to conclude that the signal is entirely polarised, which means that only one component of polarization exists. On the other hand, if the result from P_s equals then the most significant component of polarization size equals the other two components size.

4.2.3 Pinnegar Method

The third method implemented, designed by Pinnegar [2006], aims to analyze and filter the polarization of three-component signals by making use of the time-frequency S transform. This method presents a very intuitive methodology to extract polarization information from monochromatic waves. This method although extremmely important is blind in terms of extracting linear polarization motion.
The first step, similarly to the previous method 4.2.1, consists of separating the three-component signal data for it to be prepared for the next step. Having the data separated

its now time to apply the S transform algorithm defined as,

$$X(\tau, f) = \int_{-\infty}^{\infty} x(t) \left\{ \frac{|f|}{\sqrt{2\pi}} exp \left[\frac{-f^2(\tau - t)^2}{2} \right] \right\} \times exp(-2\pi i f t)\, dt, \quad (4.18)$$

to change the data domain for the time to the time-frequency domain.

In the second step, we begin with separating the real ($X_R Y_R Z_R$) and imaginary ($X_I Y_I Z_I$) parts for each component. Which is followed by obtaining a (semi-major), b (semi-minor), I (inclination), Ω, and ω values. For this, we proceed as follows,

$$A = X_R^2 + X_I^2 + Y_R^2 + Y_I^2 + Z_R^2 + Z_I^2, \quad (4.19)$$
$$B = X_R^2 - X_I^2 + Y_R^2 - Y_I^2 + Z_R^2 - Z_I^2, \quad (4.20)$$
$$C = -2(X_R X_I + Y_R Y_I + Z_R Z_I), \quad (4.21)$$

then

$$a = \frac{1}{\sqrt{2}} \sqrt{A + \sqrt{B^2 + C^2}}, \quad (4.22)$$
$$b = \frac{1}{\sqrt{2}} \sqrt{A - \sqrt{B^2 + C^2}}, \quad (4.23)$$
$$I = \arctan \left\{ \frac{[(Z_R Y_I - Z_I Y_R)^2 + (Z_R X_I - Z_I X_R)^2]^{\frac{1}{2}}}{(Y_R X_I - Y_I X_R)} \right\}, \quad (4.24)$$
$$\Omega = \arctan \left\{ \frac{(Z_R Y_I - Z_I Y_R)}{(Z_R X_I - Z_I X_R)} \right\}, \quad (4.25)$$
$$\omega = \omega_0 - \pi \left(\frac{sign(\omega_0) - 1}{2} \right), \quad (4.26)$$

where

$$\omega_0 = \arctan \left\{ \frac{b[Z_R \cos\phi_0 - Z_I \sin\phi_0]}{-a[Z_R \sin\phi_0 - Z_I \cos\phi_0]} \right\}, \quad (4.27)$$
$$\phi = \phi_0 + \pi \left(\frac{sign(\omega_0) - 1}{2} \right) sign(\phi_0), \quad (4.28)$$

where

$$\phi_0 = \frac{1}{2} \arctan \left(\frac{C}{B} \right). \quad (4.29)$$

Having obtained all the necessary values, we can now analyse the polarization of the three-component signal.

4.2.4 RS-TFR Method

The fourth method implemented consists of a sparsity-promoting approach to eigenvalue decomposition polarization analysis in the time-frequency domain. This method has been developed in the context of this dissertation as part of the SHAZAM project. It improves the deficiency of the Pinnegar method in analysing the linear particle motion and reformulates the eigenvalue decomposition polarization analysis in the frequency and time frequency domain.
For the present method, the implementation begins the same way as the methods previously described in 4.2.1 4.2.3. It starts by separating the three-component seismic data. After the first step is completed, there are then two different approaches that can be taken, and the first is the implementation of the STFT algorithm defined as,

$$TF_{STFT}(k,l) = \sum_{k=0}^{2n} x(\hat{k}) e^{\frac{-\pi(\hat{k}-k)^2}{\sigma^2}} \exp(\frac{-2\pi j \hat{k} l}{2n+1}), \quad (4.30)$$

where $e^{\frac{-\pi(\hat{k}-k)^2}{\sigma^2}}$ consists in a Gaussian window centered around the k index with a standard deviation of σ, $l = -n, ..., -1, 0, 1, ..., 2n$ and $k = 0, 1, ..., 2n$.
The second approach is as an extension of the STFT to the S transform (ST) as,

$$TF_{ST}(k,l) = \sum_{k=0}^{2n} x(\hat{k}) |l| e^{-\pi l^2 (\hat{k}-k)^2} \exp\left(\frac{-2\pi j \hat{k} l}{2n+1}\right), \quad (4.31)$$

with k and l defined the same as in equation (4.30).
With both the approaches implemented, it is now possible to obtain the time-frequency of the three components of the signal by using one of the algorithms. The third step is consists in implementing auto- and cross-correlation, which terms are obtained by implementing

$$C_{ij}(k,l) = \begin{cases} g_{ij}(k,-l) + g_{ij}(k,l) & l \neq 0, \\ g_{ij}(k,l) & l = 0, \end{cases} \quad (4.32)$$

with $k = 0, 1, ..., 2n+1$ as a time index and $l = 0, 1, ..., n$ as a frequency index.
The forth step in implementing this method consists in obtaining the time-frequency dependent eigenvectors ($u_1(k,l)u_2(k,l)u_3(k,l)$), and their corresponding eigenvalues ($\lambda_1(k,l)\lambda_2(k,l)\lambda_3(k,l)$). These values are calculated by solving,

$$(C(k,l) - \lambda_i(k,l)I)u(k,l) = 0, \quad (4.33)$$

Achieving a higher resolution has always been a big concern in the scientific community Mohammadi Gheimasi et al. [2010]; Gholami and Gheymasi [2016]; Gheymasi [2009]

Here, we use a sparsity based regularized approach to obtain the Time-Frequency (TF) of the signal. The desired TF map can be obtained by using some form of a priori information under the frame of regularization techniques [Gheymasi and Gholami, 2013; Gheymasi et al., 2016]. A sparsity-promoting regularization enables selecting a TF model with a minimum number of non-zero coefficient by solving a constrained optimization problem

$$I = argmin \left\{ \frac{1}{2} ||GI - x||_2^2 + \lambda ||I||^1 \right\}, \quad (4.34)$$

with,

$$x = GI, G \in \mathbb{R}^{L \times L^2}, I \in \mathbb{R}^{L^2 \times 1}, \quad (4.35)$$

with λ being a suitable defined lagrangian multiplier, having as a reference the Karush-Kuhn-Tucker (KKT) condition Kyparisis [1985].

4.2.4.1 Adaptive filtering in the TF-domain

The last step of the implementation of adaptive filtering to separate the seismic phases Heravi et al. [2012]. It has been extensively applied in seismology Montalbetti and Kanasewich [1970]; Vidale [1986]; Jurkevics [1988]; Mohammadigheymasi et al. [2021b]. The method consists in the design of a TF domain filter based on the rectilinearity, directivity, and amplitude attributes obtained for each time and frequency value in a given signal to either remove or extract different phases of a seismic wave.
Firstly, the rectilinearity attributes are obtained by calculating a degree of rectilinearity,

$$Re(k,l) = 1 - \frac{\lambda_2(k,l) + \lambda_3(k,l)}{\lambda_1(k,l)}, \quad (4.36)$$

defined as a measure to differentiate the rectilinear motion of body and Love waves from the elliptical motion of Rayleigh waves. Next is the application of the obtained degree to the rectilinearity filter designed in the TF domain as

$$\Psi_{Re}(Re(k,l)) = \begin{cases} 1 & -1 < Re(k,l) < \alpha, \\ cos\left(\frac{\pi(Re(k,l)-\alpha}{2(\beta-\alpha)}\right) & \alpha < Re(k,l) < \beta, \\ 0 & \beta < Re(k,l) < 1, \end{cases} \quad (4.37)$$

with α and β being the defined adjusting parameters.
Secondly, the directivity attributes are obtained by calculating a directivity measure,

$$D_i(k,l) = |u_1^T(k,l)e_i|, i \in T, R, Z. \quad (4.38)$$

And then applying the obtained measure to the directivity filter designed in the TF domain as

$$\Psi_D(D_i(k,l)) = \begin{cases} 1 & 0 < D_i(k,l) < \gamma, \\ cos\left(\frac{\pi(D_i(k,l)-\gamma}{2(\lambda-\gamma)}\right) & \gamma < D_i(k,l) < \lambda, \\ 0 & \lambda < D_i(k,l) < 1, \end{cases} \quad (4.39)$$

with γ and λ being the defined adjusting parameters and e the base vectors.
Thirdly, the amplitude attributes are obtained by defining an amplitude measure,

$$A(k,l) = \lambda_3, \quad (4.40)$$

And then applying the defined measure to the amplitude filter, designed in the TF domain, as

$$\Psi_A(A(k,l)) = \begin{cases} 0 & 0 < A(k,l) < \zeta, \\ cos\left(\frac{\pi(A(k,l)-\zeta}{2(\eta-\zeta)}\right) & \zeta < A(k,l) < \eta, \\ 1 & \eta < A(k,l) < 1, \end{cases} \quad (4.41)$$

with ζ and η being the defined adjusting parameters.
After obtaining all the rectilinearity, directivity, and amplitude attribute values, the next step is to combine these attributes properties by implementing the total TF reject filter,

$$\Psi_R = 1 - 1 - \Psi_{Re} \circ 1 - \Psi_D \circ 1 - \Psi_A, \quad (4.42)$$

and the total TF extract filter,

$$\Psi_E = 1 - \Psi_{Re} \circ 1 - \Psi_D \circ 1 - \Psi_A. \quad (4.43)$$

Finally, the last step in implementing the RS-TFR method is to perform an element-wise multiplication of Ψ with the RS-TFR of a given seismic signal three components. After finishing this last step it now becomes possible to reject and extract a particular seismic phase.

4.3 Developed Library

As stated by the title, the SeisPolPy library developed for this thesis aims to assist developers with which goal is to process seismic time series. The developed library is structured

as shown in annex A.

The chosen programming language for the development of the SeisPolPy library was Python. We took advantage of five main Python libraries,

- SciPy 2.3.1.3 because it efficiently implements sparse matrix operations. It also contains a class named "signal" which makes available several functions for development in the field of signal analysis from which we used the,
  - Fast Fourier Transform (FFT) function,
  - HT function,
  - Inverse Fourier Transform (IFFT) function,
  - Gaussian window function;
- Numpy 2.3.1.1. Here we use Numpy functions that implement and deal with array operations, their creation and data type definition and also more importantly for the functions that calculate the covariance matrix and its eigenvalues and eigenvectors;
- Matplotlib 2.3.1.2, for its plotting functions which allowed us to analyze the outputs of each method, and, depending on the quality of the plotted data, ascertain the quality of the written code;
- Cython 2.3.1.4, which allowed us to overcome the memory issues inherent in signal processing by writing "C-like code" which could be compiled into a C shared library then called inside the Python scripts;
- Sphinx, was used to assist in writing the documentation for the developed library and generate an HyperText Markup Language (HTML) and Cascading Style Sheets (CSS) file containing all the information documen+ted previously in each Python script written.

For hosting the previously mentioned HTML, it was used the platform Read the Docs. After linking the GitHub project repository, we chose to build the project documentation to have the platform host the generated web page, as shown in the Fig.s 4.1 and 4.2, respectively.

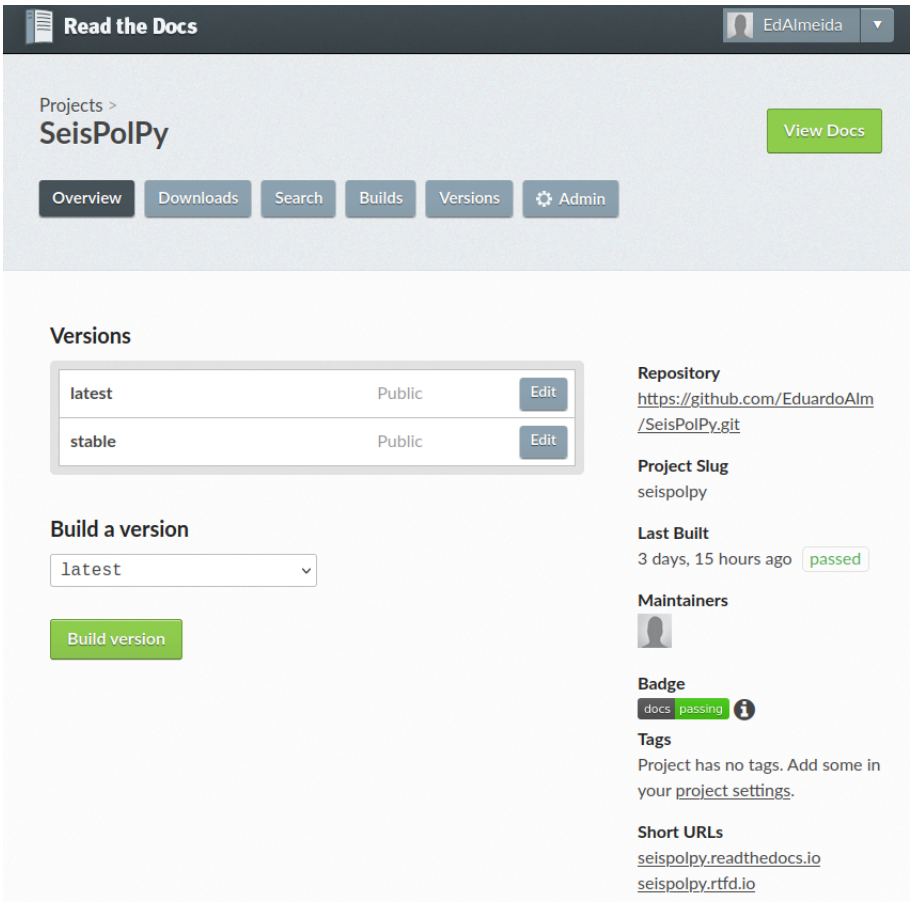


Figure 4.1: Read the Docs platform page for building the documentation of the SeisPolPy library.

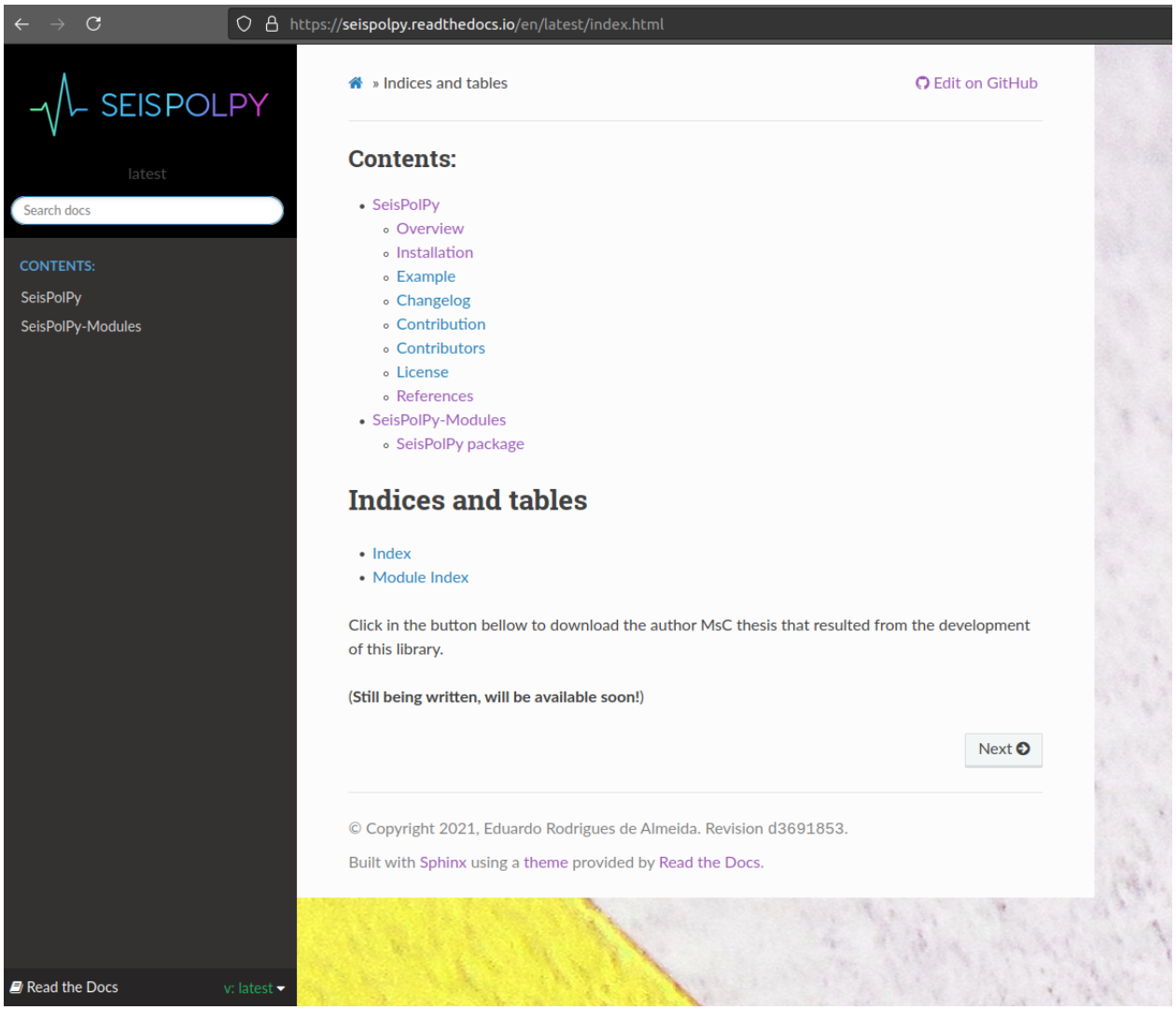


Figure 4.2: Web page containing the SeisPolPy documentation hosted by Read the Docs.

Furthermore, in order to make the change log automatically synchronzie with the GitHub repository, an additional configuration was required to allow the Sphinx library to obtain such information when built. Sphinx also needed the generation of a GitHub OAuth2 token woth only read permissions to the SeisPolPy repository and its addition, as a private environment variable, in the Read the Docs platform, as shown in Fig. 4.3.

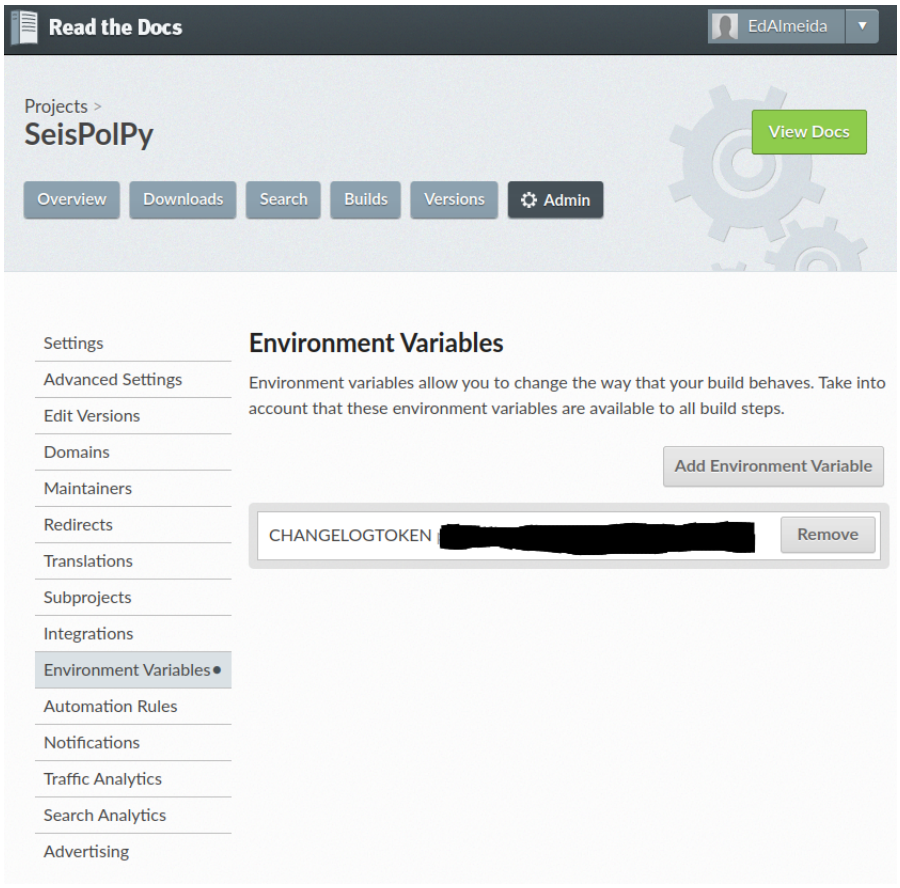


Figure 4.3: Web page, in the Read the Docs platform, to specify environment variables to be accessed during the documentation build.

4.3.1 Implemented Functions

The developed library implements the Flinn, Vidale, Pinnegar and Rstfr methods previously described in 4.2.1, 4.2.2, 4.2.3, 4.2.4 respectively.
The Flinn module holds just a single function, 4.3.1, to obtain the rectilinearity and direction of particle motion in a three-component signal by implementing the method designed by Flinn [1965]. This function takes as input two parameters, *data* (a given signal three-component data) and *window_size* (defines the size of the samples), and outputs two arrays with rectilinearity and direction of particle motion data.

Listing 4.1: Header of the function used in the Flinn script.

```
def flinn(data, window_size):
```

The Vidale module, similarly to the previous module, only contains one function, 4.3.1, to obtain the elliptical component of polarization, strike, inclination (dip) and polarization strength of the signal by implementing the method designed by Vidale [1986]. These function parameters are the same as the previously described function, which implements the Flinn method, and outputs six arrays with the data about elliptical component of polarization, strike, inclination (dip) and polarization strength of the signal.

Listing 4.2: Header of the function used in the Vidale script.

```
def vidale(data, window_size):
```

The Pinnegar module obtains the semi-major, semi-minor, inclination, pitch, phase, and strike by implementing the method designed by C. R. Pinnegar Pinnegar [2006]. This Python script includes three functions and imports two shared C libraries.
The main function, *Pinnegar*, presented in 4.3.1, receives as input *data* which, as in the two previous methods, corresponds to a given signal three-component data and *s* which must be a value between 1 and the length of the data. This function outputs six arrays with semi-major, semi-minor, inclination, pitch, phase and strike data. To obtain these outputs, the Pinnegar function has to perform a variation of the STFT, not present in the SciPy library. This variation is implemented through the functions *stft* and *forward*, shown in 4.3.1, and the C shared libraries *adjoint.so* and *diags.so* created through the use of the Cython package.

Listing 4.3: Header of the functions used in the Pinnegar script.

```
def pinnegar(data, s=100):
def stft(x, s):
def forward(N, s):
```

The *stft* function receives, as the first parameter, the data corresponding to one component of the signal and the s defined by the user and it calls, firstly, the *forward* function with the component length and *s* has its parameters. The called function will return a sparse diagonals matrix created through the *diags* function present in the *diags.so* library. This library consists of a source-code adaptation from a SciPy library function to perform the specific task required by the present method STFT algorithm adaptation and takes as parameters the component data diagonals, the offsets and the resulting matrix desired size, as shown in 4.3.1. Upon receiving the sparse diagonals matrix from the *forward* function the *stft* calls the *adjoint* function present in the shared C library *adjoint.so* and sends the sparse matrix, length of the component and the component has parameters, as presented in 4.3.1. The *adjoint* function then returns the component data in the frequency domain.
The Rstfr module obtains the semi-major and semi-minor by implementing an adaptation of the pinnegar method, which takes advantage of sparsity. This method allows for the choice between the standard STFT and the use of STFT with Sparsity Matrices. This script main function, presented in 4.3.1 takes as input parameters, *data* which, same as the previously implemented method, is the three-component data, *alg* with the options, “stft” and “s_stft” that allows the user to specify which algorithm should be executed (this parameter defaults to “stft”), *s* which must be a value between 1 and the length of the data, and *n_it* that defines the number of iterations to be run when executing the soft thresh holding (this value defaults to 100). The last parameter, *n_it*, is only considered when the chosen algorithm is the sparse STFT. The main function for the RS-TFR method outputs two arrays corresponding to the semi-major and semi-minor.

Listing 4.4: Header of the functions used in the RS-TFR script.

```
def rstfr(data, alg="stft", s=100, n_it=400):
```

```
def soft_threshholding(z, T):
def cross(x1, x2):
def semimm(t, r, z):
def forward(N, s):
def stft_s_ist(x, y, z, s, n_it, mu):
def stft(x, s):
def rectilinearity(alpha, beta, eig_values):
def directivity_rayleigh(gamma, lamb_da, eig_vec3):
def directivity_love(gamma, lamb_da, eig_vec3):
def amplitude(zeta, eta, eig_values):
```

If the chosen algorithm is the STFT, then the *rstfr* function calls the *stft* function and executes the same as described previously in the *pinnegar* function. However, if the the algorithm chosen is the sparse STFT then the *rstfr* functions calls the *stft_s_ist* function that takes has parameters the data corresponding to each of the components of the given signal, *s*, *n_it*, and mu which is a statically defined value of $1e^{-3}$. This function during its executions calls the *diags* and *adjoint* functions from the shared C libraries, previously described, and the *forw_op* function present in the *forw_op.so* shared C library. The latter function takes has parameters a sparse diagonals matrix, length of the component and the component has parameters, as presented in 4.3.1. The *forward* function, after performing a series of operations, then returns the component data in the time domain. The *stft_s_ist* function also calls the function *soft_threshholding* which receives two matrices obtained in the calling function, has parameters and outputs a value corresponding to the threshold of the given data.

After receiving the output from the *stft_s_ist* or the *stft* function, depending on the choice made by the user, the *rstfr* function calls the *semimm* function which takes as parameters, three matrices corresponding to each component data after being processed by one of the previous algorithms, STFT or the sparse STFT, as shown presented in 4.3.1. And outputs two arrays with the semi-major and semi-minor values.

Listing 4.5: Headers of the functions present in the shared C libraries.

```
cpdef forw_o( G, N, x):
cpdef adjoin( G, N, x):
cpdef np.ndarray[double, ndim=2] diagonal(list diagonals,
np.ndarray[int, ndim=1] offsets, int m, int n, dtype=None):
```

Having now obtained the semi-major and semi-minor values and have the RS-TFR/STFT of the three components the functions *rectilinearity*, *directivity_rayleigh* or *directivity_love* (which will depend on choice made by the user), and *amplitude* are then called. The first function, *rectilinearity*, receives as parameters the values alpha and beta, and an array containing eigenvalues and returns the rectilinearity value obtained. The second and third functions, *directivity_rayleigh* and *directivity_love*, both receive as parameters the values gamma and lambda as well as an array with the biggest eigenvector and return their respective directivity value. The last function, *amplitude*, similarly to previ-

ous functions, receives as parameters zeta and eta values, and an array of eigenvalues and returns the amplitude value obtained.
Furthermore, all the modules main functions also return a base64 encoded string of bytes corresponding to the resulting data plots of each of the implemented methods.

4.3.2 Building and Installation

The SeisPolPy library developed as part of this master's thesis is currently running and tested on Linux in Ubuntu version 20.04 and CentOS 8 with the latest Python3 release.
To build the library, the user, in the root folder, has to install the Python3 setuptools package used for Python library packaging and then run

```
Python3 -m build
```

which will generate the .whl and .tar.gz files and place them inside the "dist" folder. Having generated these files to install the library, while still in the root folder, the user has to use the Python package installer (pip) to install the previously generated files during the building phase by executing,

```
pip3 install dist/SeisPolPy-** \
    replacewithcurrentversion**-py3-none-any.whl
```

or

```
pip3 install dist/SeisPolPy-** \
    replacewithcurrentversion**.tar.gz.
```

Furthermore, the user can also install SeisPolPy via the PyPi repository which serves as a software distributor for software developed using Python. PyPi contains the previously generated distribution .whl file for the latest stable version of the developed library and allow for it to be installed through the commands,

```
pip3 install seispolpy
```

or

```
python3 -m pip install seispolpy.
```

After finishing the library installation, the last step is to download the folder "sharedClib" present in the library repository Almeida [2021], and place the shared object (.so) files in the folder where the SeisPolPy modules are imported. Although code efficiency was one of the priorities when writing the different modules, it was still required to create shared C libraries in order to optimise the code further. These were created with the Cython package to improve the efficiency of the scripts, which was necessary due to the high complexity present in the matrix operations performed.

4.4 Conclusions

In the present chapter, the implemented functions and the library developed were described in detail. Each section in this chapter describes the process and choices made,

during the implementation of the SeisPolPy library, in order to deliver an high quality Python library for processing seismic time series. Apart from the Pinnegar method, due to its ongoing patent and the unsuccess in contacting the author, each of the methods presented in this chapter are available publicly in GitHub at the SeisPolPy library repository Almeida [2021].

Chapter 5

Results

5.1 Introduction

The current chapter presents a discussion on the generation and pre-processing of the synthetic data used for testing the methods and implementations and the results obtained. Section 5.2, starts by describing in detail the steps taken to generate and pre-process the synthetic data, the tools and software necessary. Section 5.3, displays and explains the obtained results from each implemented function and finally, in section 5.4, are described the implemented tests module.

5.2 Synthetic Data Generation

Although the final goal of developing a library is to implement it on real data sets, the evaluation of the accuracy of an algorithm is only possible by testing on synthetic data sets, which have been generated by a controlled and known forward modellings. As a result, synthetic data generation is an unavoidable step in every numerical development research work.
There are several ways to generate synthetic data Li et al. [2014]; Herrmann [2013]; Kaser et al. [2010]; Afanasiev et al. [2017]; Reinarz et al. [2020]. In this project, two different methods were used, the IRIS syngine combined with the ObsPy library and the SPECFEM3D Globe software. Both of these methods will now be described in detail. Also, in table 5.1, are presented the chosen stations information and important values, such as latitude, longitude, and data centers regarding the seismic waveforms presented.

Station Code	**Station Name**	**Latitude**	**Longitude**	**Source Distance** (km)
ACRG	Accra, Ghana	5.64	-0.20	6886.59
II ABKT	Alibek, Turkmenistan	37.93	58.11	5985.87

Table 5.1: Table containing the station codes, station names, latitude and longitude values.

5.2.1 IRIS Syngine and ObsPy

The IRIS Synthetics Engine (Syngine), IRIS [2015], is a web service to generate and return fully three-dimensional synthetic seismograms for different one-dimensional earth models.
To generate the necessary synthetic data we created a Python script and took advantage of the ObsPy library, which implements, not just the “ObsPy.client.syngine” module to trans-

fer the seismograms created by the IRIS syngine service but also provides other modules for pre-processing seismic waveforms.

For the creation of this script, after initialising the syngine client, the first step was to perform a request to the IRIS service, using the previously initialised client object, to download the generated data. Next, a seismic data collection station, model and source parameters must then be specified. A full description and tutorial for using Syngine can be found online at Nissen-Meyer et al. [2015].

In our case, the seismic station ACRG was used with the ak135f_5s model. This returned the data shown in Fig. 5.1.

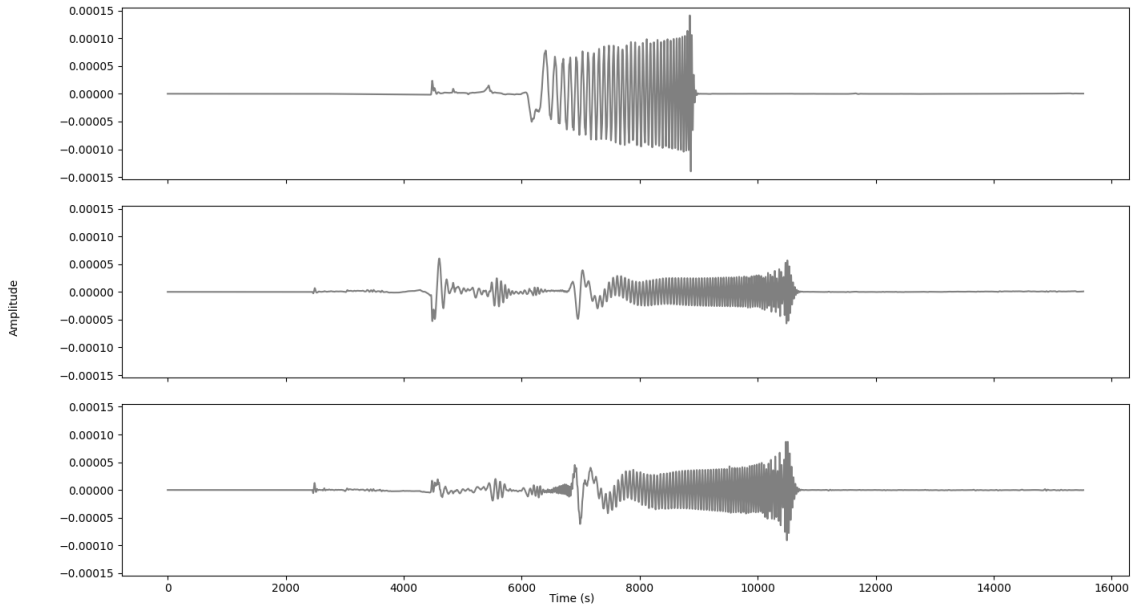


Figure 5.1: Seismic waveform, pertaining to the ACRG station, generated using the IRIS syngine service implemented in ObsPy.

Having downloaded the generated seismic waveform from IRIS, the second step was to rotate the horizontal components to change the signal orientation from north and east to radial and transverse to have the generated data in the same orientation as expected in the implemented methods.

The third step was to perform some pre-processing to the seismic signal generated by using the necessary ObsPy functions. In particular we began by rotating the horizontal components to change the signal orientation from north and east to radial and transverse in order to have the generated data in the same orientation as expected in the implemented methods. Next we applied trend removal on the signal by calling the detrend function, then, by calling the decimate function twice, we down-sampled the data, firstly, by an integer factor of 2 and, secondly, by a factor of 3, and lastly, the data was trimmed by calling the trim function, this produces the seismic waveform presented in Fig. 5.2.

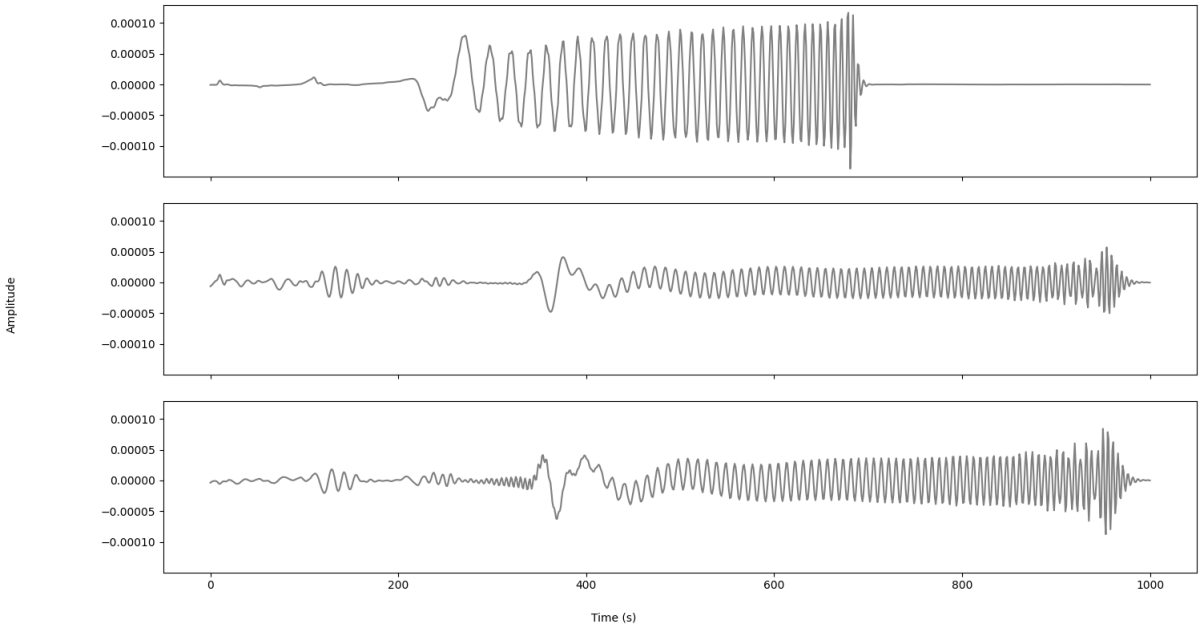


Figure 5.2: Seismic waveform, pertaining to the ACRG station, generated using the IRIS syngine service implemented in ObsPy after performing pre-processing.

Furthermore, after the signal pre-processing was finished, the data was converted from the ObsPy trace object to a NumPy array and saved in a *.mat* file.

5.2.2 SPECFEM3D Globe

This software, written in Fortran2003, is considered to be the most advanced in terms of generating complex synthetic data in seismology due to all the features it contains. It simulates three-dimensional global and regional seismic waves propagation and performs adjoint tomography or full waveform imaging (FWI) based on the spectral element method. Furthermore, SPECFEM3D Globe uses parallel programming based on the MPI, and it supports GPU acceleration, as well as OpenCL.
For this project, we took advantage of GPU acceleration by setting up two infrastructures supporting advanced GPU's. Both infrastructures were utilised to run the computationally demanding SPECFEM3D software.

- The first infrastructure was a server, provided by C4G laboratory, containing an NVIDIA Tesla v100 graphics to install a CentOS 7 operating system on a VM. At the time of writing this dissertation this model of GPU was state-of-art hardware for this specific numerical modelling.
- The second infrastructure was a machine running the CentOS 7 operating system put together using components facilitated by using components from the two PC's from the RELEASE laboratory each one containing an NVIDIA Titan Black graphics cards One PC was used with an external power supply from the second PC. The second PC's graphic card was removed to create a machine with two Titan Black graphics cards were wired together by using two Scalable Link Interface (SLI) cables to run in parallel.

For the software configuration and execution, the tutorial made by NVIDIA NVIDIA [2018] was followed on how to run SPECFEM3D Globe when using Tesla graphics cards. This tutorial was made specifically for running SPECFEM3D Globe on the NVIDIA Tesla GPU and so it had to be adapted to using the Titan graphics cards.
The configuration steps, although quite straightforward, require some attention as some steps are not explicitly mentioned in the tutorial. For example, the user, before executing the commands "make" and "./configure", has to define manually the environment variables shown in 5.1. The configuration steps used are the same for both GPUs.

Listing 5.1: Environment variables required in order to configure SPECFEM3D GLOBE for GPU acceleration.

```
export PATH=/usr/local/cuda-11.3/bin:$PATH
export LD_LIBRARY_PATH=/usr/local/cuda-11.3/
    lib64:$LD_LIBRARY_PATH
export CUDA=/usr/local/cuda
export CUDA_LIB=/usr/local/cuda/targets/
    x86_64-linux/lib
export PATH=$PATH:/usr/lib64/openmpi/
export PATH=$PATH:/usr/lib64/openmpi/bin
```

The execution, as a whole, of SPECFEM3D Globe, consists of a three step process. Firstly, the user must update a parameters file according to his needs. In this file the GPU model present in the machine that and number of processes that will execute the software are specified as well as the seismic source parameters and other parameters regarding the final solution. In this first step is crucial, if the user intends to utilise a GPU, to set the *GPU_MODE* parameter to *.true*. Secondly, the program needs to be rebuilt using the "make" command, since it uses static array sizes which then require the same to be rebuilt when any changes are made to the parameters file. The next stage consists of the creation of a computational mesh. This is followed by the final stage where the user can now run the solver to generate the final output. These two stages are achieved by running the commands listed in 5.2, for all of the commands presented the number of processes needs to be specified, this value will vary according to the number of GPUs.
In this project, we did not use the parallel processing feature of the MPI since neither of the machines could handle the toll imposed by this feature on its components. And so, the commands listed bellow 5.2 match the ones utilised on the execution of SPECFEM3D Globe.

Listing 5.2: Required commands for creating the mesh and running the main program.

```
mpirun -np 1 bin/xmeshfem3D
mpirun -np 1 bin/xspecfem3D
```

After generating the final solution the user is presented with files in the Seismic Analysis Code (SAC) format which contain the waveforms generated for each seismic station. To organise better the resulting waveforms it was then decided to use the Adaptable Seismic Data Format (ASDF) file format. The ASDF file format allows the gathering of all the

information inside a possibly “infinite” number of SAC or miniSEED data files and put it together in a new ASDF file. The advantage of this file format is not just the possibility of having to manage just one single file but also the fact that by making use of the pads we can get, more efficiently, not just the waveform but also specific metadata information, such as event date and name, stations list, and coordinates regarding each waveform Krischer et al. [a]. For the conversion of the SAC files to the ASDF format and for obtaining all the information regarding the generated event was written a Python script, that uses the pyasdf library, in order to automate the process. For plotting the converted ASDF file the ASDF-Sextant experimental Graphical User Interface (GUI) Krischer et al. [b] was used.

The Titan GPUs computer spectral elements had to be reduced, while still honing the earth curvature, to 96 due to the computation cost present in executing with the 128 defined in the Tesla GPU VM. This change might not seem significant but it changed the resolution and accuracy diminishing the quality of the generated waveforms as it is possible to see in Fig. 5.3 and 5.4. Due to these changes present in the data generated for this project was only used the seismic waveform obtained by the Tesla GPU virtual machine. Furthermore, both machines generated data for thirty stations with an, almost equal, execution time of approximately one hour and thirty minutes for the Tesla GPU and one hour and fifty minutes for the two Titan GPUs.

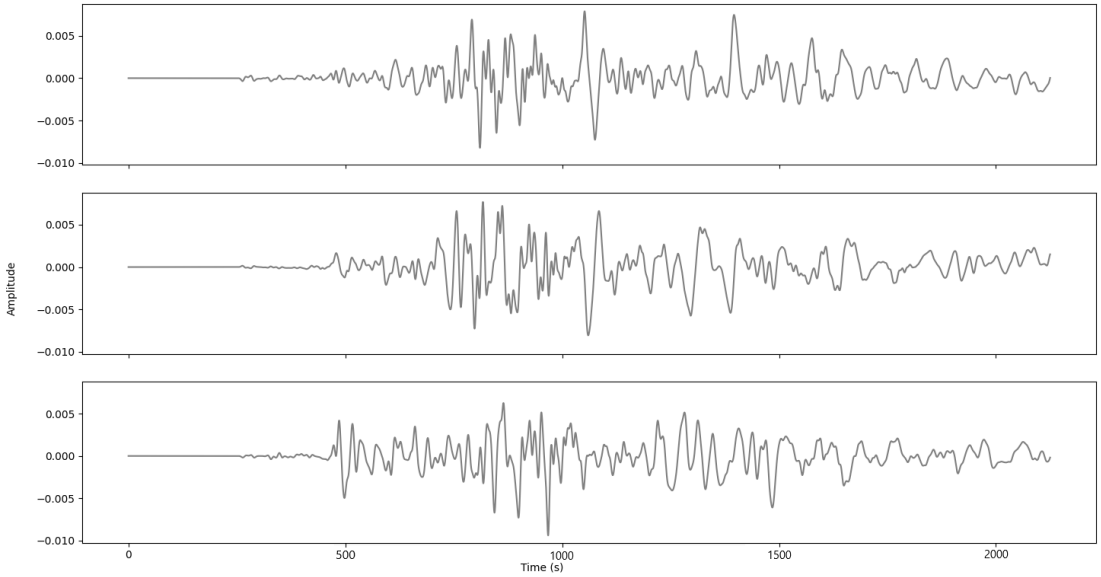


Figure 5.3: Seismic waveform, pertaining to the II.ABKT station, generated by SPECFEM3D Globe on the VM containing the NVIDIA Tesla GPU.

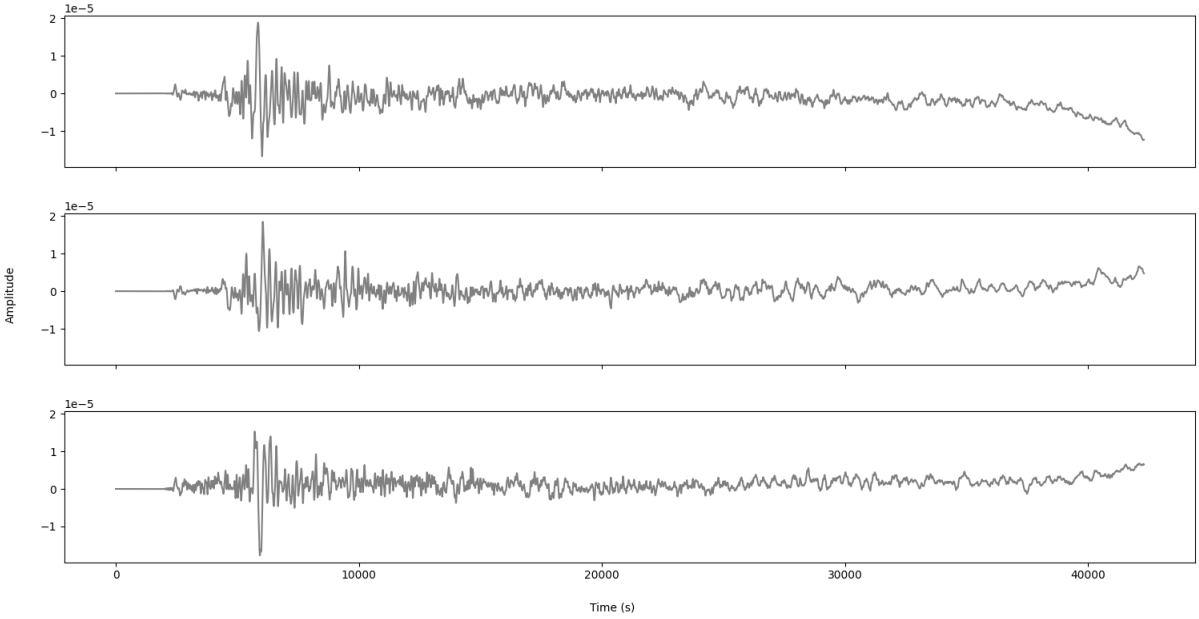


Figure 5.4: Seismic waveform, pertaining to the II.ABKT station, generated by SPECFEM3D Globe on the computer containing the two NVIDIA Titan Black GPU.

By using ObsPy modules, a set of pre-processing steps, similar to ones described in section 5.2.1, was applied on the generated seismic data (shown in Fig. 5.3) to attain the pre-processed waveform shown in Fig. 5.5. In this case, the down-sampling was applied by a factor of 8 to attain the final data by the sampling frequency of 0.5 Hz.

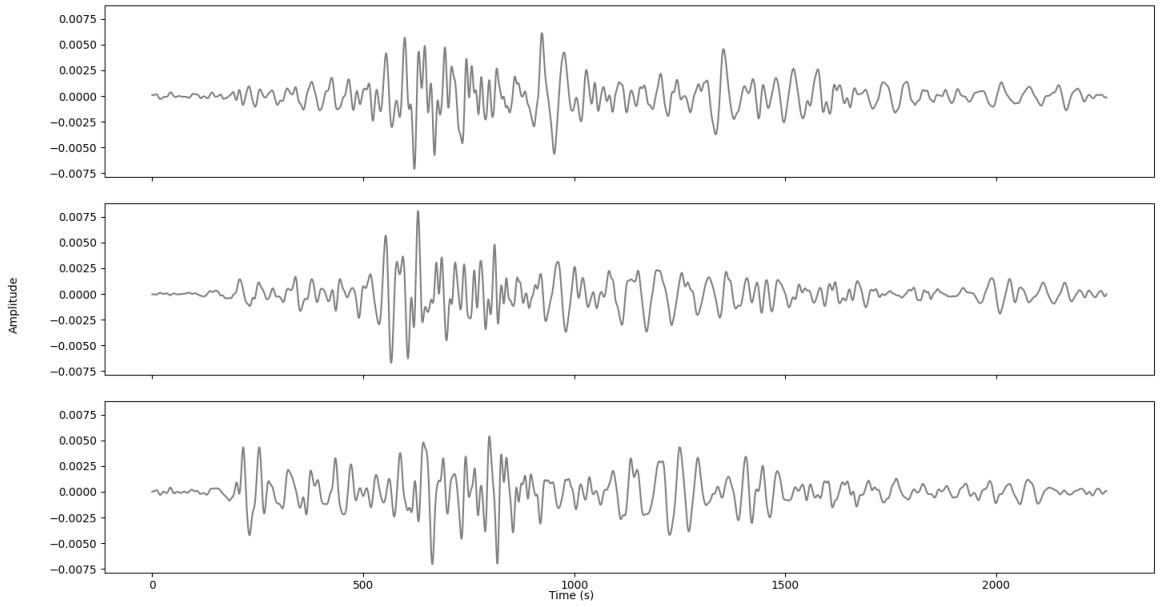


Figure 5.5: Seismic waveform, pertaining to the II.ABKT station, generated by SPECFEM3D Globe on the VM containing the NVIDIA Tesla GPU after performing pre-processing.

5.3 Obtained Results

The synthetic data generated in sections 5.2.1 and 5.2.2 is processed by implementing the written modules described in the previous chapter 4. The written library was installed on a VM made available by the C4G laboratory. This virtual machine contained the CentOS 8 operative system, NVIDIA Tesla V100 GPU with 16GB of Video Random Access Memory (VRAM), a 4 cores CPU (from which we took advantage of only one core for processing, since the implemented modules do not support multiprocessing) and 64GB of RAM. In tables 5.2 and 5.3, the execution time for the obtained results as well as the maximum RAM usage and CPU core usage for each executed module is discussed. In following, we briefly explain the obtained results.

SeisPolPy Modules	**Execution Time** (seconds)	**Maximum RAM Usage** (GB)	**Maximum CPU Usage** (%)
Flinn	1.75	3.85	40.05
Pinnegar	4.02	8.45	96.3
Vidale	2.85	2.85	81.8

Table 5.2: Table containing the execution times, maximum RAM usage and maximum CPU core usage regarding each module execution with the synthetic data generated using IRIS syngine service.

SeisPolPy Modules	**Execution Time** (seconds)	**Maximum RAM Usage** (GB)	**Maximum CPU Usage** (%)
Flinn	2.68	1.78	28.6
Pinnegar	4.34	2.85	81.8
Vidale	3.09	2.02	78.8

Table 5.3: Table containing the execution times and maximum RAM usage regarding each module execution with the synthetic data generated using SPECFEM3D Globe software.

Rstfr Module	**Execution Time** (seconds)	**Maximum RAM Usage** (GB)	**Maximum CPU Usage** (%)
STFT algorithm	14.98	14.7	100
sparse STFT algorithm	292.83	30.8	100

Table 5.4: Table containing the execution times and maximum RAM usage regarding the Rstfr module execution on the same data set as the one present in the article accepted by the IEEE Transactions on Geoscience and Remote Sensing (Mohammadigheymasi et al. [2021a]). This article is also present in annex B.

5.3.1 Results of Flinn Module

The time-domain rectilinearity and directivity information of the generated data by the syngine and SPECFEM3D software is extracted by implementing the Flinn module. The information is illustrated in Fig.s 5.6 and 5.7, respectively.

As the Rectilinearity panel of this Fig. 5.6 shows, a sequence of body waves has been recorded in the time interval around 30 to 200 seconds before the surfaces waves reach the sensor and become the dominant recorded waves.

The other panels, the Particle motion in Z, R, and T directions, give information on particle motion in the time-domain when the seismic wavefield arrives in the station. It shows that the particle motion is dominantly in the T direction, corresponding to the Love waves. Likewise, the direction of particle motion in the R and Z directions shows a combination of vertical and radial particle motion corresponding to the Rayleigh waves. As it is seen, by using the polarization attributes, one can effectively discriminate between different phases of seismic waves.

Similarly, the Rectilinearity panel of this Fig. 5.7 shows the rectilinearity information of the arrival wavefield in the station. As the synthetic data was generated by more sophisticated software, by considering more realistic assumptions of the propagating environment, including non-homogeneity and anisotropy, the seismic wavefield is more complicated and more similar to the waveforms being recorded by real sensors. There is no exact and sharp boundary between the arrival phases for this case. However, the obtained results can be utilised for designing adaptive filters to filter or amplify specific phases. This step is out of the scope of this research work and can be studied in future works.

The execution time for this module was 1.75 seconds for the syngine, and 2.68 for the SPECFEM3D Globe generated data.

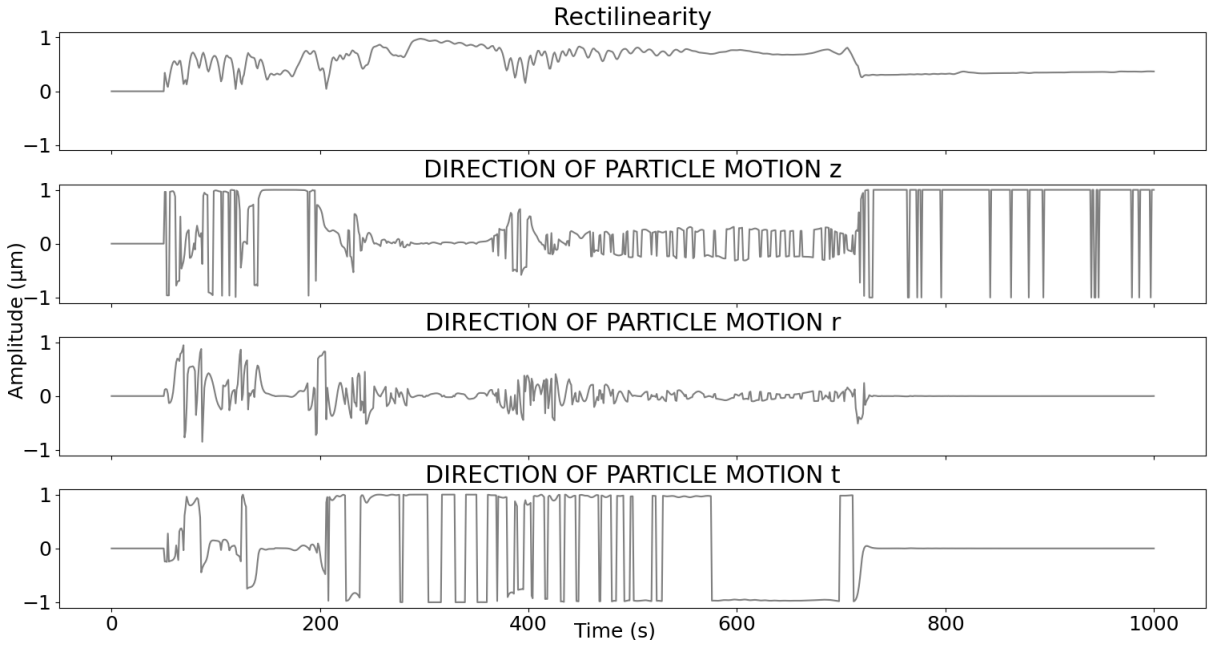


Figure 5.6: Flinn method results for the IRIS generated synthetic data.

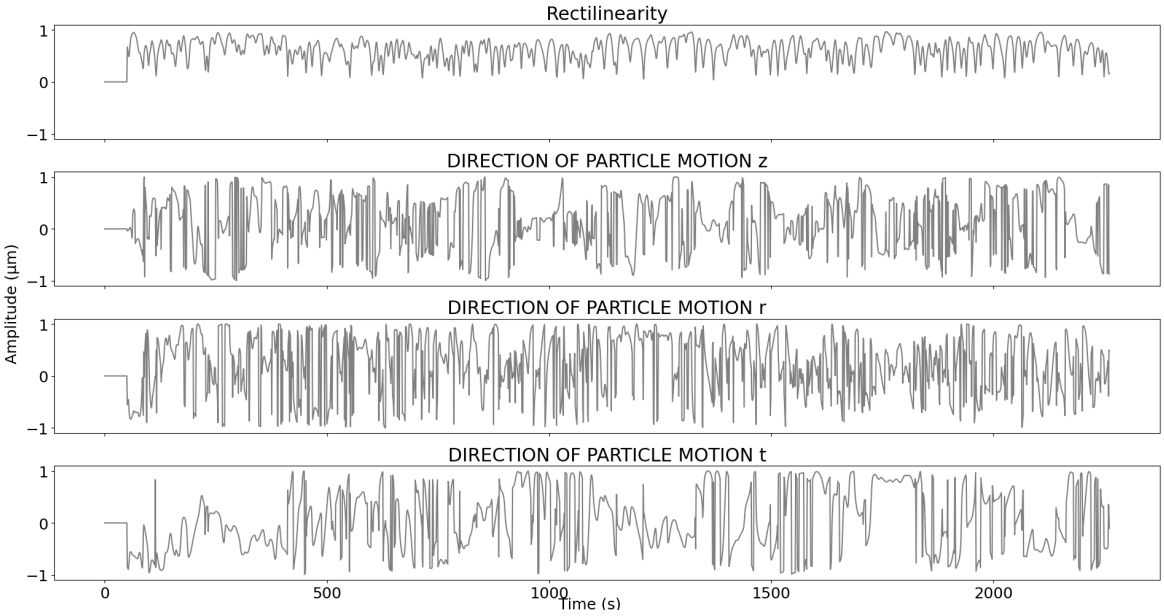


Figure 5.7: Flinn method results for the SPECFEM3D Globe generated synthetic data.

5.3.2 Results of Pinnegar Module on Synthetic Data

The results obtained by executing the Pinnegar module on the data set generated from the IRIS service are presented in Fig.s 5.8 and 5.9. The execution time for obtaining these results was of 4.02 seconds.

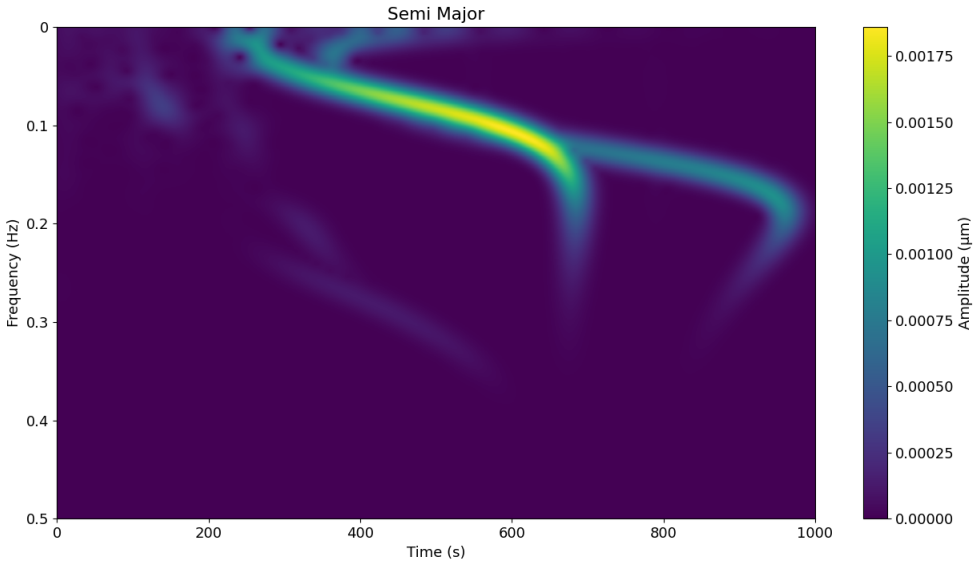


Figure 5.8: Pinnegar method semi-major results for the IRIS generated synthetic data.

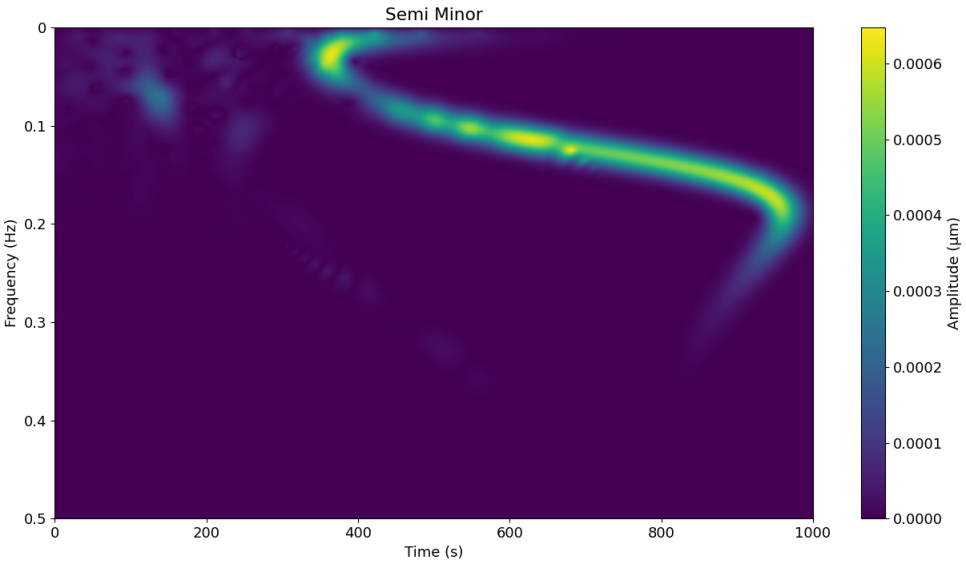


Figure 5.9: Pinnegar method semi-minor results for the IRIS generated synthetic data.

Figures 5.10 and 5.11, show the results from this module execution using the data set generated with SPECFEM3D. The execution time was 4.34 seconds.

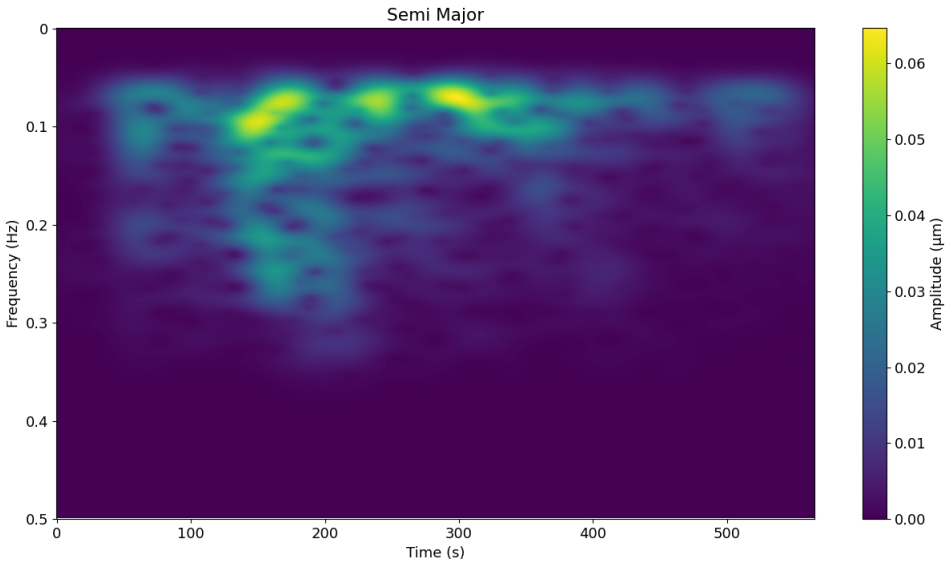


Figure 5.10: Pinnegar method semi-major results for the SPECFEM3D Globe generated synthetic data.

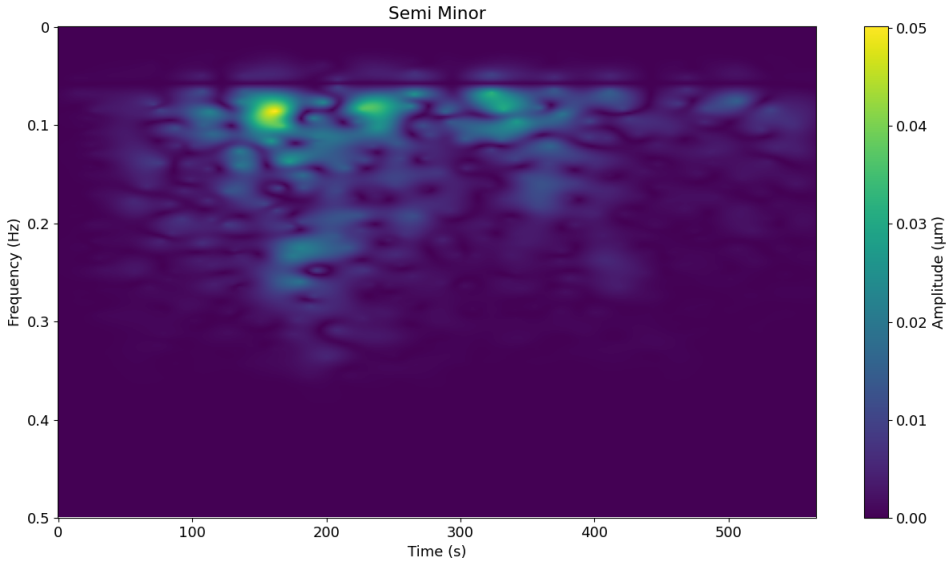


Figure 5.11: Pinnegar method semi-minor results for the SPECFEM3D Globe generated synthetic data.

In the previously mentioned figures, it is presented the results obtained for both generated synthetic data sets regarding the seismogram semi-major axis S spectrum, in Fig.s 5.8 and 5.10, and the semi-minor axis S spectrum, in Fig.s 5.9 and 5.11.

The semi-major S spectrum depicts the time–frequency dependency of the polarization ellipse's long axis. Moreover, the semi-minor S spectrum represents the time–frequency interdependence of the short axis of the polarization ellipse and thus can be conceived as the spectrum of the 'circular' component of the elliptical motion.

5.3.3 Results of Vidale Module

The Vidale module implemented when executed with the data from the IRIS service had a execution time of 2.85 seconds and the obtained results are shown in Fig.s 5.12 and 5.13. These, describe the results of polarization analysis for the three-component generated waveform.

In Fig. 5.12 the results show the dip, strike and elliptical component of polarization obtained through the defined equations (4.16), (4.15) and (4.14), respectively. Moreover, Fig. 5.13 describes the result obtained for the polarization strength of the signal (4.17).

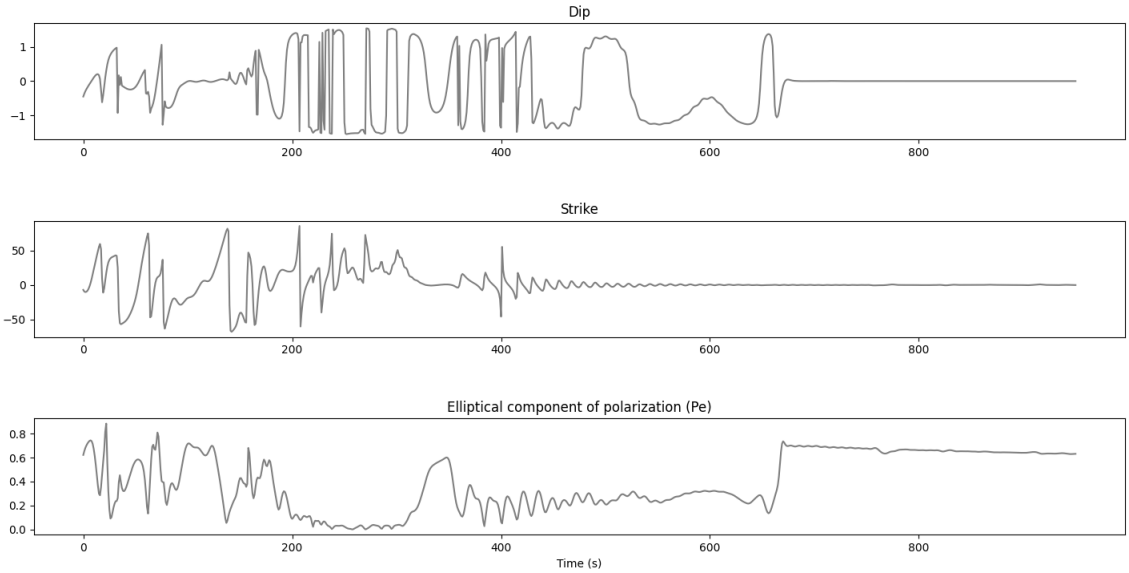


Figure 5.12: Vidale method results for the IRIS generated synthetic data regarding dip, strike an and elliptical component of polarization.

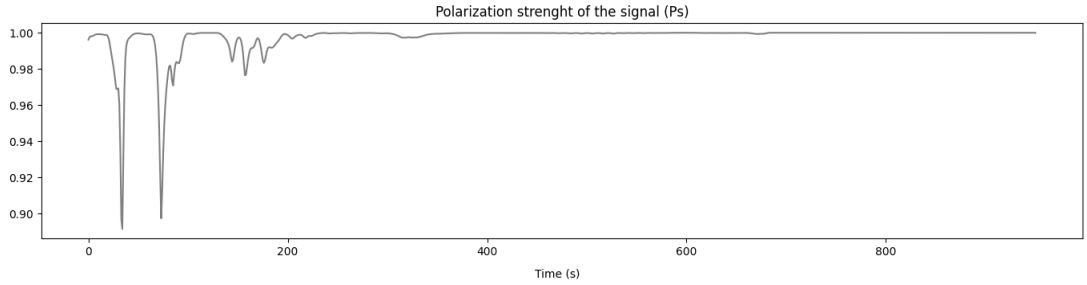


Figure 5.13: Vidale method result for the IRIS generated synthetic data concerning the polarization strength of the signal.

Furthermore, Fig.s 5.14 and 5.15 present the results attained through this module execution using the SPECFEM3D generated data. The module had a execution time of 3.09 seconds. The obtained results, in the same way of the ones described by Fig.s 5.12 and 5.13, present the seismogram polarization analysis results concerning dip, strike, elliptical component of polarization and polarization strength of the signal.

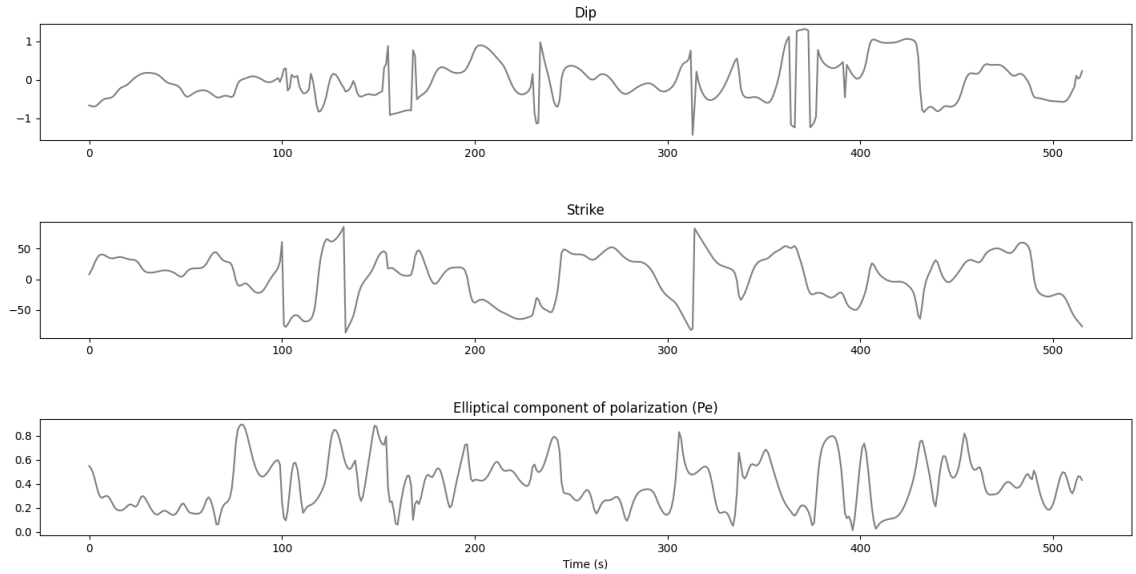


Figure 5.14: Vidale method results for the SPECFEM3D Globe generated synthetic data regarding dip, strike and and elliptical component of polarization.

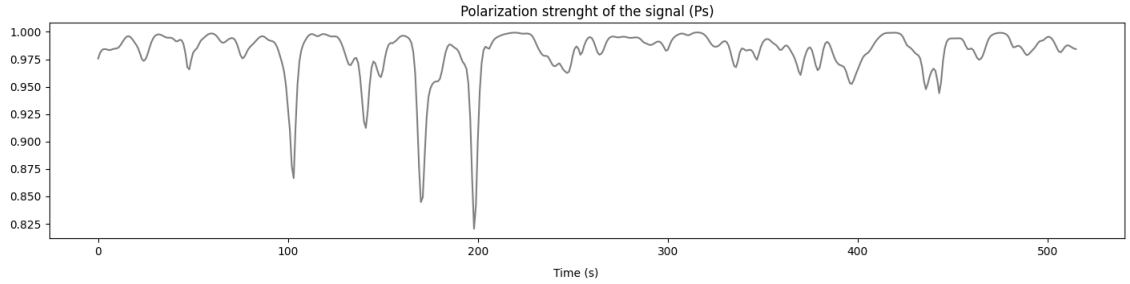


Figure 5.15: Vidale method result for the SPECFEM3D Globe generated synthetic data concerning the polarization strength of the signal.

5.3.4 Results of Rstfr Module

The Rstfr module, implemented in SeisPolPy, when executed with the synthetic data used in the published article (Mohammadigheymasi et al. [2021a]) regarding the seismic event that occurred in 101 km SSW of Tres Picos, Mexico with a source mechanism value of $M_w = 8.2$ resulting of a normal faulting at an intermediate depth of 47.4 km, the module had an execution time of 14.98 seconds with the STFT and 292.83 when using the sparse STFT.

In Fig. 5.16, the results obtained when executing the present module with the sparse STFT algorithm are presented. Through the obtained results it becomes possible to clearly identify the filtered Love and Rayleigh waves for each component of the signal (T, R, Z) and in this way perceive their arrival times, among other information there present. In panels (1), (3) and (5) the filtered Love wave in the transverse, radial and vertical components, respectively, are shown and panels (2), (4) and (6) describe the filtered Rayleigh wave

also in the transverse, radial and vertical components. The filtered waves in each panel are highlighted in black.

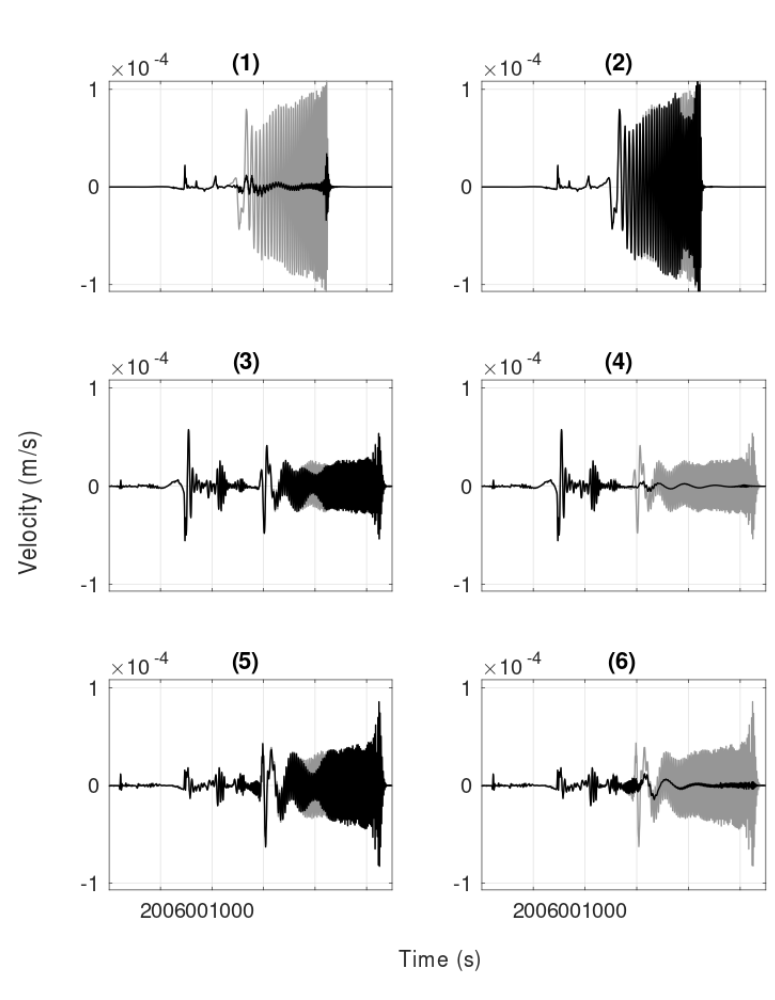


Figure 5.16: RSTFR method results presenting Love and Rayleigh waves trace and filter results for the sparse STFT algorithm. In (1), (3) and (5) correspond to the Love wave filtered components, and (2), (4) and (6) to the Rayleigh wave filtered components.

5.4 Library Test Module

The main goal of creating a module for software tests is to create a battery of tests that evaluates the methods in the developed library's current state of development and so that in the future, as this library scales, there is no concern for the integration of a testing library as it is already available.

In this way, there is no need for further configuration due to the implementation of new methods. All that is required is to write new tests and add them to the already existing battery. These tests also mean that whoever downloads and rebuilds the software library also has a set of tests that may be run that validate any new changes to the already existing codes.

For the testing phase of the implemented methods the Pytest Python library, described previously in 2.3.1.5, was used.

For the SeisPolPy library, it is only possible to write ten tests due to how the functions of the modules execute. Seven of these tests aim to assert that the implemented methods always return the same values if given the same data and parameters i.e that there are no deviations during the execution of the code. Furthermore, these seven tests also verify that the *if* conditions present in the Rstfr module perform as expected when verifying the choice made by the user in selecting the algorithm to be used (STFT or sparse STFT) and the waves to be filtered (Love or Rayleigh). For the last three tests, the objective is to verify if conditions raise the implemented exceptions.
The expected results for this battery of tests is for the first seven to pass with success, validating the consistency of the results, the choice of algorithms, and choice for the type of wave to be filtered, and the final three tests to fail, raising the implemented exceptions.

5.5 Conclusions

The present chapter was firstly discussed, in detail, two approaches to generate synthetic seismic data. Furthermore, the obtained results, validated by this dissertation co-supervisor and geophysics specialist, were presented along with their respective execution times. In this section, it is also shown two tables, in which are present the values of execution time, the maximum RAM usage and CPU (core) usage for each of the modules present in the SeisPolPy library. Finally, was described the tests module present within the SeisPolPy library.

Chapter 6

Main Conclusions and Future Work

6.1 Main Conclusions

This document presented a thesis for a masters dissertation concerning the development of a Python library for the processing of seismic time series. An overview of the fundamental topics, such as basic concepts of seismology, signal processing and time series analysis was presented in chapters 2 and 3, as well as a literature review of important papers presented in a chronological order to be able to view the developments over time concerning seismic time series analysis. In chapter 1, an introduction is given, regarding the problems to be addressed by this project, the objectives expected to be achieved, and the contributions that were made. Even though some of the original goals seemed very challenging at first, as they involve several areas of science and engineering, some of which relatively unknown to the author, they were overcome with success. Chapter 4, describes, in detail, the implemented methods and functions existing in each developed module of the SeisPolPy software library, as well as how to build the package locally and install it locally or through the online Python repository. A selection of important methods were implemented, as well as a new method RS-TFR developed within the SHAZAM project. Finally, chapter 5, presents the techniques and processes used for generating synthetic data, in itself a non trivial procedure. The results obtained in each module present in the library, and the test module implemented to validate the codes written are also described.
In conclusion, this documents objective was to provide insights into the development of software for seismic time series analysis and provide a detailed description of the work developed. With the obtained results and tests performed, it is possible to ascertain that every objective proposed was achieved with success.

6.2 Future Work

Possible future work to be developed in order to improve and extend the library is as follows,

- Implementation of more methods for processing of seismic time series;
- Improve execution times by redesigning the algorithms to take better advantage of multiprocessing;
- Implementation of a pre-processing framework within the library to expand its capabilities.

Bibliography

USGS. v, 17

——, “Earthquake glossary,” [Online] https://earthquake.usgs.gov/learn/glossary/. Last accessed in 21 May 2021. v, 11, 16, 18

IRIS, “Seismic monitor,” [Online] http://ds.iris.edu/seismon/index.phtml. Last accessed in 2 December 2020. v, 18, 19

EPOS, “Epos integrated core services (ics),” 2019, [Online] https://www.ics-c.epos-eu.org/data/search. Last accessed in 2 December 2020. v, 18, 19

J. Hunter, D. Dale, E. Firing, M. Droettboom *et al.*, “Matplotlib,” 2012, [Online] https://matplotlib.org/. Last accessed in 11 November 2020. v, 22, 23

H. Mohammadigheymasi, P. Crocker, M. Fathi, E. Almeida, G. Silveira, A. Gholami, and M. Schimmel, “Sparsity-promoting approach to polarization analysis of seismic signals in the time-frequency domain,” *IEEE Transactions on Geoscience and Remote Sensing*, 7 2021. [Online]. Available: https://doi.org/10.36227/techrxiv.14910063.v1 vii, 8, 12, 33, 55, 61

E. Flinn, “Signal analysis using rectilinearity and direction of particle motion,” *Proceedings of the IEEE*, vol. 53, no. 12, pp. 1874–1876, 1965. 7, 12, 27, 28, 29, 30, 33, 35, 44

J. E. Vidale, “Complex polarization analysis of particle motion,” *Bulletin of the Seismological society of America*, vol. 76, no. 5, pp. 1393–1405, 1986. 7, 12, 28, 29, 33, 34, 39, 44

R. Pinnegar, “Polarization analysis and polarization filtering of three-component signals with the time—frequency s transform,” *Geophysical Journal International*, vol. 165, no. 2, pp. 596–606, 2006. 7, 12, 28, 31, 33, 36, 45

E. Almeida, “Seispolpy a python library for polarization analysis and filtering of seismic waves,” 2021, [Online] https://github.com/EduardoAlm/SeisPolPy. Last accessed in 28 June 2021. 8, 9, 12, 47, 48

E. Almeida, H. Mohammadigheymasi, M. Fathi, P. Crocker, and G. Silveira, “Eigenvalue decomposition polarization analysis: A regularized sparsity-based approach,” *EGU General Assembly 2021*, 2021. [Online]. Available: https://doi.org/10.5194/egusphere-egu21-15267 8, 12

P. Richards and K. Aki, *Quantitative seismology: theory and methods*. Freeman New York, 1980, vol. 859. 11

L. Hutchings and G. Viegas, "Application of empirical green's functions in earthquake source, wave propagation and strong ground motion studies," *Earthquake Research and Analysis-New Frontiers in Seismology*, 2012. 11

G. Bensen, M. Ritzwoller, M. Barmin, A. Levshin, F. Lin, M. Moschetti, N. Shapiro, and Y. Yang, "Processing seismic ambient noise data to obtain reliable broad-band surface wave dispersion measurements," *Geophysical Journal International*, vol. 169, no. 3, pp. 1239–1260, 2007. 11, 27

A. Wüstefeld and G. Bokelmann, "Null detection in shear-wave splitting measurements," *Bulletin of the Seismological Society of America*, vol. 97, no. 4, pp. 1204–1211, 2007. 11

T. Lecocq, C. Caudron, and F. Brenguier, "Msnoise, a python package for monitoring seismic velocity changes using ambient seismic noise," *Seismological Research Letters*, vol. 85, no. 3, pp. 715–726, 2014. 11, 24

L. Ermert, J. Igel, K. Sager, E. Stutzmann, T. Nissen-Meyer, and A. Fichtner, "noisi: A python tool for ambient noise cross-correlation modeling and noise source inversion," *Solid Earth Discussions*, vol. 2020, pp. 1–27, 2020. 11, 24

C. Jiang and M. Denolle, "Noisepy: A new high-performance python tool for ambient-noise seismology," *Seismological Research Letters*, vol. 91, no. 3, pp. 1853–1866, 2020. 11, 25

V. Ingle and J. Proakis, *Digital signal processing using matlab: a problem solving companion*. Cengage Learning, 2016. 11, 20

A. Bent, "Primary Wave (P-Wave)," in *Encyclopedia of Natural Hazards*, P. T. Bobrowsky, Ed. Dordrecht: Springer Netherlands, 2013, pp. 777–777. [Online]. Available: https://doi.org/10.1007/978-1-4020-4399-4_278 16

——, "Secondary Wave (S-Wave)," in *Encyclopedia of Natural Hazards*, P. T. Bobrowsky, Ed. Dordrecht: Springer Netherlands, 2013, pp. 901–901. [Online]. Available: https://doi.org/10.1007/978-1-4020-4399-4_311 17

V. Babuska and M. Cara, *Seismic anisotropy in the Earth*. Springer Science & Business Media, 1991, vol. 10. 17

IRIS, "Mission statement," [Online] https://www.iris.edu/hq/about_iris#vision. Last accessed in 2 December 2020. 18

M. Beyreuther, R. Barsch, L. Krischer, T. Megies, Y. Behr, and J. Wassermann, "Obspy a python framework for seismology," 2009, [Online] https://github.com/obspy/obspy/wiki. Last accessed in 11 November 2020. 19

M. Peixeiro, "The complete guide to time series analysis and forecasting," 2019, [Online] https://towardsdatascience.com/

the-complete-guide-to-time-series-analysis-and-forecasting-70d476bfe775. Last accessed in 11 November 2020. 20

N. Kehtarnavaz, "Chapter 7 - frequency domain processing," in *Digital Signal Processing System Design (Second Edition)*, N. Kehtarnavaz, Ed. Burlington: Academic Press, 2008, pp. 175 – 196. [Online]. Available: http://www.sciencedirect.com/science/article/pii/B9780123744906000076 20

Madisetti and K. Vijay, *The Digital Signal Processing Handbook-3 Volume Set*. CRC press, 2018. 21

L. Sevgi, *Electromagnetic Modeling and Simulation*. John Wiley & Sons, 2014. 21

R. G. Stockwell, "A basis for efficient representation of the s-transform," *Digital Signal Processing*, vol. 17, no. 1, pp. 371–393, 2007. 21

L. Chun-Lin, "A tutorial of the wavelet transform," *NTUEE, Taiwan*, 2010. 21, 22

T. Oliphant, "Numpy," 2010, [Online] https://github.com/numpy/numpy. Last accessed in 11 November 2020. 22

SciPy, "Scipy," 2012, [Online] https://www.scipy.org/. Last accessed in 11 November 2020. 23

——, "Scipy tutorial," 2012, [Online] https://docs.scipy.org/doc/scipy/reference/tutorial/. Last accessed in 11 November 2020. 23

S. Behnel, R. Bradshaw, C. Citro, L. Dalcin, D. Seljebotn, and K. Smith, "Cython: The best of both worlds," *Computing in Science & Engineering*, vol. 13, no. 2, pp. 31–39, 2010. 23

B. Okken, *Python testing with Pytest: simple, rapid, effective, and scalable*. Pragmatic Bookshelf, 2017. 23

M. Beyreuther, R. Barsch, L. Krischer, T. Megies, Y. Behr, and J. Wassermann, "Obspy: A python toolbox for seismology," *Seismological Research Letters*, vol. 81, no. 3, pp. 530–533, 2010. 24

K. Aki and P. Richards, *Quantitative seismology*. University Science Books, 2002. 27

A. Udias and E. Buforn, *Principles of seismology*. Cambridge University Press, 2017. 27

A. Jurkevics, "Polarization analysis of three-component array data," *Bulletin of the seismological society of America*, vol. 78, no. 5, pp. 1725–1743, 1988. 27, 28, 30, 31, 39

Yilmaz, *Seismic data analysis: Processing, inversion, and interpretation of seismic data*. Society of exploration geophysicists, 2001. 27

J. Montalbetti and E. Kanasewich, "Enhancement of teleseismic body phases with a polarization filter," *Geophysical Journal International*, vol. 21, no. 2, pp. 119–129, 1970. 27, 29, 30, 31, 39

A. Reading, W. Mao, and D. Gubbins, "Polarization filtering for automatic picking of seismic data and improved converted phase detection," *Geophysical Journal International*, vol. 147, no. 1, pp. 227–234, 2001. 28, 30

R. H. Herrera, J. B. Tary, M. Van der Baan, and D. W. Eaton, "Body wave separation in the time-frequency domain," *IEEE Geoscience and Remote Sensing Letters*, vol. 12, no. 2, pp. 364–368, 2014. 28, 32

C. Wang, Y. Wang, P. Sun, and Y. Li, "Discussions on the processing of the multi-component seismic vector field," *Applied Sciences*, vol. 9, no. 9, p. 1770, 2019. 28, 32

H. Mohammadi Gheimasi, H. R. Siahkoohi, and K. Lucas, "Improvement of temporal resolution of seismic data using singular spectrum analysis and autoregressive methods," *Journal of the Earth and Space Physics*, vol. 36, no. 3, 2010. 38

A. Gholami and H. M. Gheymasi, "Regularization of geophysical ill-posed problems by iteratively re-weighted and refined least squares," *Computational Geosciences*, vol. 20, no. 1, pp. 19–33, 2016. 38

H. M. Gheymasi, "Using singular spectrum analysis and autoregressive methods for improvement of temporal resolution of seismic data," in *Shiraz 2009-1st EAGE International Petroleum Conference and Exhibition*. European Association of Geoscientists & Engineers, 2009, pp. cp–125. 38

H. M. Gheymasi and A. Gholami, "A local-order regularization for geophysical inverse problems," *Geophysical Journal International*, vol. 195, no. 2, pp. 1288–1299, 2013. 39

H. M. Gheymasi, A. Gholami, H. Siahkoohi, and N. Amini, "Robust total-variation based geophysical inversion using split bregman and proximity operators," *Journal of Applied Geophysics*, vol. 132, pp. 242–254, 2016. 39

J. Kyparisis, "On uniqueness of kuhn-tucker multipliers in nonlinear programming," *Mathematical Programming*, vol. 32, no. 2, pp. 242–246, 1985. 39

M. Heravi, H. Mohamadi, and H. Hashemi, "A curvelet based wavefield separation in vertical seismic profiling," in *Istanbul 2012-International Geophysical Conference and Oil & Gas Exhibition*. Society of Exploration Geophysicists and The Chamber of Geophysical ..., 2012, pp. 1–4. 39

H. Mohammadigheymasi, M. R. Ebrahimi, G. Silveira *et al.*, "A sparsity-based adaptive filtering approach to shear wave splitting," in *EGU General Assembly Conference Abstracts*, 2021, pp. EGU21–13 988. 39

D. Li, D. Helmberger, R. Clayton, and D. Sun, "Global synthetic seismograms using a 2-d finite-difference method," *Geophysical Journal International*, vol. 197, no. 2, pp. 1166–1183, 2014. 49

R. B. Herrmann, "Computer programs in seismology: An evolving tool for instruction and research," *Seismological Research Letters*, vol. 84, no. 6, pp. 1081–1088, 2013. 49

M. Kaser, C. Castro, V. Hermann, and C. Pelties, "Seissol–a software for seismic wave propagation simulations," in *High Performance Computing in Science and Engineering, Garching/Munich 2009*. Springer, 2010, pp. 281–292. 49

M. Afanasiev, C. Boehm, M. Van Driel, L. Krischer, D. May, M. Rietmann, and A. Fichtner, "Salvus: a flexible high-performance and open-source package for waveform modelling and inversion from laboratory to global scales," in *EGU General Assembly Conference Abstracts*, 2017, p. 9456. 49

A. Reinarz, D. Charrier, M. Bader, L. Bovard, M. Dumbser, K. Duru, F. Fambri, A. Gabriel, J. Gallard, S. Koppel *et al.*, "Exahype: an engine for parallel dynamically adaptive simulations of wave problems," *Computer Physics Communications*, vol. 254, p. 107251, 2020. 49

IRIS, "Data services products: Synthetics engine," 2015. [Online]. Available: https://doi.org/10.17611/DP/SYNGINE.1 49

T. Nissen-Meyer, L. Van Driel, M.and Krischer, S. Stähler, K. Hosseini, A. Hutko, and E. Zürich, "Data services products: synginem," 2015, [Online] https://ds.iris.edu/ds/products/synginem/. Last accessed in 29 June 2021. 50

NVIDIA, "Gpu-accelerated specfem3d-globe," 2018, [Online] https://www.nvidia.com/en-au/data-center/gpu-accelerated-applications/specfem3d-globe/. Last accessed in 10 April 2021. 52

L. Krischer, P. Dirk, E. Andrade, C. Cui, J. Parker, M. White, C. Deil, and Z. Vasović, "pyasdf," [Online] https://github.com/SeismicData/pyasdf/. Last accessed in 24 May 2021. 53

L. Krischer, L. Neeley, and A. Cooper, "Asdf-sextant," [Online] https://github.com/SeismicData/asdf_sextant. Last accessed in 24 May 2021. 53

Appendix A

SeisPolPy Library File Structure

In this appendix, we present the files structure of the SeisPolPy library.

```
dist/SeisPolPy-0.0.1-py3-none-any.whl
dist/SeisPolPy-0.0.1-py3.8.egg
dist/SeisPolPy-0.0.1.tar.gz
docs/build/doctrees
docs/build/html
docs/source/_static/css/default.css
docs/source/_templates
docs/source/img/....png
docs/source/conf.py
docs/source/index.rst
docs/source/modules.rst
docs/source/SeisPolPy-Modules.rst
docs/requirements.txt
docs/Makefile
examples/flinn_example.py
examples/pinnegar_example.py
examples/rstfr_s_stft_example.py
examples/rstfr_stft_example.py
examples/vidale_example.py
examples/ACRG.mat
SeisPolPy/tests/outputb64files/....txt
SeisPolPy/tests/ACRG.mat
SeisPolPy/tests/test_flinn.py
SeisPolPy/tests/test_pinnegar.py
SeisPolPy/tests/test_rstfr.py
SeisPolPy/tests/test_vidale.py
SeisPolPy/__init__.py
SeisPolPy/Flinn.py
SeisPolPy/Pinnegar.py
SeisPolPy/Rstfr.py
SeisPolPy/Vidale.py
sharedClib/adjoint.so
sharedClib/diags.so
sharedClib/forw_op.so
.read_the_docs.yaml
LICENSE.md
pyproject.toml
README.rst
setup.cfg
setup.py
```

Appendix B

Sparsity-Promoting Approach to Eigenvalue Decomposition Polarization Analysis of Seismic Signals in the Time-Frequency Domain

In the present section of the appendix the submitted and accepted article for publication by IEEE Transactions on Geoscience and Remote Sensing is made available.

Sparsity-promoting approach to polarization analysis of seismic signals in the time-frequency domain

Hamzeh Mohammadigheymasi, Paul Crocker, Maryam Fathi, Eduardo Almeida, Graça Silveira, Ali Gholami, and Martin Schimmel

***Abstract*—Time-frequency (TF)-domain polarization analysis (PA) methods are widely used as a processing tool to decompose multi-component seismic signals. However, as a drawback, they are unable to obtain sufficient resolution to discriminate between overlapping seismic phases, as they generally rely on a low-resolution time-frequency representation (TFR) method. In this paper, we present a new approach to the TF-domain PA methods. More precisely, we provide an in-detailed discussion on rearranging the eigenvalue decomposition polarization analysis (EDPA) formalism in the frequency domain to obtain the frequency-dependent polarization properties from the Fourier coefficients owing to the Fourier space orthogonality. Then, by extending the formulation to the TF-domain and incorporating sparsity-promoting time-frequency representation (SP-TFR), we alleviate the limited resolution when estimating the TF-domain polarization parameters. The final details of the technique are to apply an adaptive sparsity-promoting time-frequency filtering (SP-TFF) to extract and filter different phases of the seismic wave. By processing earthquake waveforms, we show that by combining amplitude, directivity, and rectilinearity attributes on the sparse TF-domain polarization map of the signal, we are able to extract or filter different phases of seismic waves. The SP-TFF method is evaluated on synthetic and real data associated with the source mechanism of the $M_w = 8.2$ earthquake that occurred in the south-southwest of Tres Picos, Mexico. A detailed discussion on the results of these experiments is given, approving the efficiency of the technique in separating not only the Rayleigh from the Love waves but also to discriminate them from the body and coda waves.**

***Index Terms*—Eigenvalue decomposition, Polarization analysis and filtering, Sparsity-promoting time-frequency representation, Rayleigh and Love waves elimination, adaptive filtering**

H. Mohammadigheymasi and E. Almeida, are with the Department of Computer Sciences, University of Beira Interior, Covilhã, Portugal e-mail: (hamzeh@ubi.pt).

P. Crocker is with the Instituto de Telecomunicações and University of Beira Interior, Covilhã, Portugal

M. Fathi is with Islamic Azad University, Science and Research Branch, Tehran, Iran.

G. Silveira is with the Dom Luiz Institute, Faculty of Science, University of Lisbon, and Instituto Superior de Engenharia de Lisboa, Lisbon, Portugal.

A. Gholami is with the Institute of Geophysics, University of Tehran, 14155/6466 Tehran, Iran.

M.Schimmel is with the Institute Of Earth Sciences Jaume Almera, Barcelona, Spain

I. Introduction

A Seismic wavefield recorded as a seismogram is a superposition of overlapping direct, reflected, refracted, converted, and scattered body and surface waves, contaminated by various background sources and the signal generated noise [1]. It is well known that surface and body waves can carry considerable information about the subsurface structure. Depending on the research scope, any of these seismic phases can be studied, while their detection and extraction require advanced processing and analysis tools. Accordingly, multicomponent processing techniques have been developed to analyze the nonlinear and time-varying processes behind the seismic sources and the propagating environment [Refer to [2] as a rigorous survey]. Among these techniques, polarization analysis methods have attracted significant attention.

Generally, the polarization analysis methods can be divided into three broad categories: time, frequency, and time-frequency (TF) domain methods. As a pioneer, Flinn [3] introduced the eigenvalue decomposition polarization analysis in the time-domain. Likewise and in the realm of Hilbert transform, Vidale [4] introduced the analytic signal polarization technique. Although these methods are equipped by time-windowing to extract the non-stationary signal properties, they cannot discriminate between overlapping events with different frequencies. Comparatively, studies were carried out on achieving polarization properties of the seismic time-series in the frequency domain. Primarily developed by [5], the propagation direction of surface waves was obtained using the amplitude and phase spectrum of seismic waves. Likewise, Samson and Olson [6] proposed a technique to provoke the polarization state based on eigenvalue decomposition of the spectral matrix in different frequency narrow bands. However, these techniques are incapable of analyzing non-stationary signals in the time domain.

Due to the non-stationary nature of seismic signals with overlapping phases in time and frequency, pure time- or frequency-domain methods are often difficult to discriminate between the non-stationary seismic phases adequately. To alleviate this, Jurkevics [7] proposed filtering the signals into a series of narrow frequency bands, applying short sliding time windows, and then estimating the polarization ellipse from the covariance matrix in each window at each band. Despite providing a TF-domain insight to polarization analysis, the resolution was still

limited in the frequency domain due to the restricted functionality of the bandpass filtering. Therefore, by applying different time-frequency representation (TFR) methods [8, 9, 10] and using various polarization estimation criteria, several TF-based polarization analysis methods were proposed to analyze the non-stationary seismic signals. These TF-domain polarization estimation tools range from eigenvalue decomposition of the covariance matrix [11, 12] and complex trace analysis [4, 13, 14], to fitting the particle motion to a parametric polarization ellipse [15]. Despite attaining a TF-domain estimation these methods are not yet able to resolve closely-spaced overlapping seismic events in both time and frequency domains.

Obtaining a high-resolution TFR has always been a challenge for the scientific community and a variety of methods have been introduced, including synchrosqueezing transform (SST) [16], sparse transforms [17], SP-TFR [18, 19, 20], to name but a few. Remarkably, by formulating the TFR as an inverse problem, and taking advantages of sparsity-promoting (SP) regularization as a promising tool to obtain a high-resolution solution, several methods have been developed to improve the TFR resolution [18, 19, 20]. SP-TFR found many applications in seismology, including the denoising of microseismic data [20] and attenuation of seismic ground roll noise [19].

On the other hand, filtering or extracting different phases of seismic waves is a concerning challenge in the seismic community. Although in a laterally homogeneous structure, the Love wave appears mainly on the transverse and the Rayleigh wave on the vertical and radial components, due to the lateral heterogeneity and anisotropy, it is seldom the case in real data [4]. Likewise, this assumption fails in miss-oriented horizontal sensor components, which is a global problem even for the best-installed seismic networks [21]. Hence, it is always challenging to discriminate between the body and Rayleigh waves on the vertical and radial components as well as SH and Love waves on the transverse component. Accordingly, polarization filtering techniques have been developed to extract or filter a specific phase from other phases using the analyzed polarization information. As pioneers, Flinn [3], Montalbetti and Kanasewich [22], and Vidale [4] utilized time-domain rectilinearity and directivity attributes to amplify body wave phases teleseismic data. In like manner, Samson and Olson [6] applied these criteria to filter ultra low-frequency magnetic field signal fluctuations in the frequency domain. Similarly, Schimmel et al. [14] designed a TF domain filter based on the degree of polarization (DOP) extracted from the semi-major and semi-minor axis of the elliptical motion of the three-component ambient noise data to filter the elliptical particle motion. In an intuitive method, Pinnegar [15] utilized semi-major, semi-minor, inclination, and azimuth parameters to discriminate between the circular and linear polarization to filter the Rayleigh waves. Although the introduced filtering scheme is efficient in filtering the elliptically polarized phases like Rayleigh waves, it still faces challenges in filtering the linear phases because the method attains a null value of azimuth and inclination angles analyzing linear particle motions.

This article begins with an elaborative review of the eigenvalue decomposition polarization analysis EDPA; we formulate EDPA as a function of frequency. On this basis, we obtain high-resolution TF-domain polarization parameters by extending the formulation from frequency to the TF-domain and combining with the SP-TFR. Afterward, by extending the polarization filtering method of Pinnegar [15] to incorporate high-resolution TF-domain information of directivity, rectilinearity, and amplitude properties, we design suitably defined filters to accept (or reject) linear and elliptical seismic phases, including Rayleigh and Love, making it possible to separate them from body and coda waves. The main focus of the paper is to discriminate between the Love and Rayleigh from the body and coda waves.

This paper is organized in the following manner. First, in section II we elaborate the theory behind the EDPA in the time, frequency, and TF domains. We show that EDPA can be applied independently for every single frequency by taking advantage of the orthogonality of the Fourier domain. This property can be extended to the TF-domain, making it possible to obtain TF-domain polarization parameters. Then, by reviewing the SP-TFR and combining with EDPA, we obtain a high-resolution TF-domain polarization map of the signal. Afterward, by implementing TF-domain rectilinearity, directivity, and amplitude attribute and combining them with SP-TFR we introduce sparsity-promoting time-frequency filtering (SP-TFF) method to be used for filtering different phases of seismic signal. Next, in section III by conducting numerical experiments on synthetic and real earthquake data examples, we show that SP-TFF can efficiently extract and filter Rayleigh and Love waves and discriminate between linearly polarized seismic phases like Love and body and coda waves. Finally, in sections IV we discuss the results and conclude the paper.

II. Theory

To keep this paper self-contained, we rearrange the EDPA formalism in the frequency and TF domain to be combined by SP-TFR. Furthermore, to be consistent with the numerical algorithms, we exclusively present a discrete version of the mathematical equations.

A. Polarization analysis using eigenvalue decomposition

Suppose that

$$\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3] \in \mathbb{R}^{L\times 3}, \tag{1}$$

is a seismic time-series recorded with three components aligned with the base vectors of a right-handed coordinate system $\{\boldsymbol{e}_E, \boldsymbol{e}_N, \boldsymbol{e}_Z\}$ (east-north-vertical), $\{\boldsymbol{e}_T, \boldsymbol{e}_R, \boldsymbol{e}_Z\}$ (transverse-radial-vertical), or $\{\boldsymbol{e}_L, \boldsymbol{e}_Q, \boldsymbol{e}_T\}$ (tangent-normal-binormal or Frenet wave). The latter coordinate systems are obtained from the ordinary east-north-vertical system by rotation [see [23] for more details]. Each component, $\boldsymbol{x}_i \in \mathbb{R}^{L\times 1}$ $(i = 1, 2, 3)$, samples the wavefield arriving to the sensor along the time

axis on $t_k = k\delta t$, with δt being sampling interval and $k = 0, 1, ..., 2n$ being the time index assuming an odd length $L = 2n + 1$ of seismogram.

Then, the polarization properties of $\boldsymbol{X}$ can be extracted by the eigenvalue decomposition of the covariance matrix

$$\boldsymbol{V} = \begin{bmatrix} V_{11} & V_{12} & V_{13} \\ V_{21} & V_{22} & V_{23} \\ V_{31} & V_{32} & V_{33} \end{bmatrix} \in \mathbb{R}^{3\times3} \tag{2}$$

[22, 3, 7, 24]. The elements of the symmetric and real-valued matrix $\boldsymbol{V}$ in (2) are auto- and cross-variances of the components of the time-series defined as

$$V_{ij} = \left[\frac{1}{2n+1} \sum_{l=0}^{2n} (x_i(l) - \mu_i)(x_j(l) - \mu_j) \right], \quad i, j = 1, 2, 3. \tag{3}$$

In (3), μ is the mean or expected value of components and is defined as

$$\mu_i = \Psi\{\boldsymbol{x}_i\} = \frac{1}{2n+1} \sum_{l=0}^{2n} x_i(l) \tag{4}$$

[22]. Assuming a weakly stationary condition for all the components of the time series, $\Psi\{\boldsymbol{x}_i\} \cong 0,\ i = 1, 2, 3$, (3) can be rewritten as

$$V_{ij} = \left[\frac{1}{2n+1} \sum_{l=0}^{2n} x_i(l) x_j(l) \right] = C_{ij}, \tag{5}$$

which simplifies the definition of elements in (2) to auto- and cross-correlations,

$$\boldsymbol{C} = \begin{bmatrix} C_{11} & C_{12} & C_{13} \\ C_{21} & C_{22} & C_{23} \\ C_{31} & C_{32} & C_{33} \end{bmatrix} \in \mathbb{R}^{3\times3}, \tag{6}$$

or equally,

$$\boldsymbol{C} = \frac{\boldsymbol{X}^T \boldsymbol{X}}{N}, \tag{7}$$

where $(.)^T$ in (7) denotes the transposition operator. The weakly stationary assumption in (5) is satisfied by applying DC removal or detrending.

The quadratic matrix of correlation coefficient, $\boldsymbol{C}$, fits the particle motion ellipsoid in a least-squares sense; the parameters of this ellipsoid are obtained by solving the system of equations

$$(\mathbf{C} - \lambda_i \mathbf{I})\mathbf{u} = 0, \tag{8}$$

[7]. Geometrically, solutions to (8) give directions (eigenvectors, ($\boldsymbol{u}_1$, $\boldsymbol{u}_2$, $\boldsymbol{u}_3$)) that the linear transformation by operator $\boldsymbol{C}$ merely elongates or shrinks; the ratio of elongation/shrinkage is given by eigenvalues, (λ_1, λ_2, λ_3). Indeed, the eigenvectors direct principal axes of the polarization motion ellipsoid, and the eigenvalues are the size of those axes; eigenvalues are sorted such that $\lambda_j \geq \lambda_k$ for $j < k$.

1) Eigenvalue decomposition in the frequency domain: The first implementations of the eigenvalue decomposition on the frequency domain were proposed by [25, 6]; the spectral matrix corresponding to a perturbation around the central frequency $[\omega - \delta\omega, \omega + \delta\omega]$ was decomposed using eigenvalue decomposition to provoke the polarization states of the signal. This decomposition scheme is more compatible with natural signals, which are rarely composed of single-frequency polarized elements. However, by decomposing to a strictly polarized single frequency state, one can benefit from the orthogonality of the Fourier transform to extract frequency-dependent polarization properties. Here, we review and simplify the process.

The discrete Fourier domain counterpart of (1), $\boldsymbol{X}^f = \boldsymbol{\mathcal{FT}}\{[\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3]\} = [\boldsymbol{f}_1, \boldsymbol{f}_2, \boldsymbol{f}_3] \in \mathbb{R}^{L\times3}$, is obtained by modulating $\boldsymbol{x}_i$, $i = 1, 2, 3$, with pure sinusoids having discrete frequencies as follows

$$f_i(l) = \frac{1}{2n+1} \sum_{k=0}^{2n} x_i(k) exp(\frac{-2\pi \jmath kl}{2n+1}),\ i = 1, 2, 3. \tag{9}$$

Here, $l = -n, ..., n$, is the frequency index giving frequency content of the signal on discrete frequencies $\omega_l = \frac{l}{(2n+1)\delta t}$, and $\jmath$ is the imaginary unit. The original signal is reconstructed by applying the inverse Fourier transform, $\boldsymbol{X} = \boldsymbol{\mathcal{IFT}}\{[\boldsymbol{f}_1, \boldsymbol{f}_2, \boldsymbol{f}_3]\}$,

$$x_i(k) = \sum_{l=0}^{2n} f_i(l) exp(\frac{2\pi \jmath kl}{2n+1}),\ i = 1, 2, 3. \tag{10}$$

Accordingly, the frequency-domain counterpart of (5) is defined as

$$C_{ij} = ZL\{\mathcal{IFT}(\boldsymbol{f}_i \circ \boldsymbol{f}_j^*)\}, \tag{11}$$

in which, $(.)^*$, and $\circ$ are complex conjugate and Hadamard or element-wise product operator, respectively, and $ZL\{.\}$ is a function that picks the zero-lag element. By taking advantage of the orthogonality property of the Fourier domain, (11) can be written as

$$C_{ij} = \sum_{l=0}^{n} C_{ij}(l) = \sum_{l=0}^{n} ZL\{\boldsymbol{\mathcal{IFT}}(\hat{\boldsymbol{f}}_i^l \circ \hat{\boldsymbol{f}}_j^{l\,*})\}, \tag{12}$$

with

$$\hat{f}_k^l(j) = \begin{cases} f_k(j) & j = l \\ 0 & j \neq l \end{cases} \tag{13}$$

Hence, the auto- and cross-correlation terms for every single frequency can be decomposed via (13). Instead of applying operation in (13) to obtain $C_{ij}(l)$, it is attainable from $\boldsymbol{g}_{ij} = \boldsymbol{f}_i \circ \boldsymbol{f}_j^*$ as

$$C_{ij}(l) = \begin{cases} g_{ij}(-l) + g_{ij}(l) & l \neq 0 \\ g_{ij}(l) & l = 0 \end{cases},\ l = 0, 1, ..., n \tag{14}$$

gives the frequency-dependent auto- and cross-correlation elements. Solving system of equations (8) for the frequency-dependent elements $C_{ij}(l)$,

$$(\mathbf{C}(l) - \lambda_i(l)\mathbf{I})\mathbf{u}(l) = 0, \tag{15}$$

gives the eigenvectors, ($\boldsymbol{u}_1(l)$, $\boldsymbol{u}_2(l)$, $\boldsymbol{u}_3(l)$) and eigenvalues, ($\lambda_1(l)$, $\lambda_2(l)$, $\lambda_3(l)$), as a function of frequency.

2) Eigenvalue decomposition in the time-frequency domain: The process described in section (II-A1) effectively decomposes stationary signals into it's frequency-dependent polarization components (eigenvectors and eigenvalues). Having to deal with the non-stationary nature of seismic data, the same definition is extended to give polarization components that depend on time and frequency by substituting a TFR in place of the ordinary Fourier transform. Much research has been devoted to finding efficient TF analysis methods and many powerful methods have been developed in the past decades, including the wavelet transform [26], the Wigner-Ville distribution [27], short time fourier transform (STFT) [8], Stockwell transform (ST) [9], etc. As an instance, the STFT representation of signal $\boldsymbol{x}_i$ in (1) is obtained by

$$TF_{STFT}(k,l) = \sum_{\hat{k}=0}^{2n} x(\hat{k}) w(\hat{k}-k) exp(\frac{-2\pi \jmath \hat{k} l}{2n+1}), \quad l = -n, ..., -1, 0, 1, ..., 2n, \;\; k = 0, 1, ..., 2n \tag{16}$$

with

$$w(\hat{k}-k) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(\hat{k}-k)^2/2\sigma^2} \tag{17}$$

being a Gaussian window with standard deviation σ, centered on the time index k. The definition in (16) can be extended to the ST [9] by applying a time–frequency spectral localization using a window function scalable with frequency as

$$w(\hat{k}-k, l) = \frac{l}{\sigma\sqrt{2\pi}} e^{-l^2(\hat{k}-k)^2/2\sigma^2} \tag{18}$$

with k and l are defined the same as (16). The parameter σ in (17) and (18) controls the resolution of the transform in the time and frequency domain; higher values of σ attains higher frequency resolution, while lower values improves the time resolution.

By obtaining TF of the 3-components of the signal, the TF-domain auto- and cross-correlation terms is obtained as

$$C_{ij}(k,l) = \begin{cases} g_{ij}(k,-l) + g_{ij}(k,l) & l \neq 0 \\ g_{ij}(k,l) & l = 0 \end{cases} \tag{19}$$

for time and frequency indexes $, k = 0, 1, ..., 2n+1,\ l = 0, 1, ..., n$. Consequently, the TF-dependent eigenvectors, ($\boldsymbol{u}_1(k,l)$, $\boldsymbol{u}_2(k,l)$, $\boldsymbol{u}_3(k,l)$) and eigenvalues, ($\lambda_1(k,l)$, $\lambda_2(k,l)$, $\lambda_3(k,l)$), is obtained by solving

$$(\mathbf{C}(k,l) - \lambda_i(k,l)\mathbf{I})\mathbf{u}(k,l) = 0, \tag{20}$$

giving a TF map of polarization state of signal. This decomposition process is similar to the method introduced by Pinnegar [15]. However, as we will discuss in the following sections, it can be used to filter the linearly polarized seismic phases, which is not able to be done with the Pinnegar [15] method.

B. Regularized sparsity-promoting TF decomposition

The system of equations for STFT and ST linear which allows us to define the TF coefficients as a solution of a linear system equations

$$\boldsymbol{x} = \boldsymbol{G}\boldsymbol{\alpha}, \qquad \boldsymbol{G} \in \mathbb{R}^{L \times L^2}, \qquad \boldsymbol{\alpha} \in \mathbb{R}^{L^2 \times 1} \tag{21}$$

where $\boldsymbol{\alpha}$ is a vectorized rearrangement of TF coefficients in (16) [see [20, 19] for more details about the structure of the forward operator $\boldsymbol{G}$]. Since the linear system in (21) in under-determined, there exists an infinite number of TF maps for representing the signal. The desired TF map can be obtained by using some form of a priori information under the frame of regularization techniques [20, 19]. A sparsity-promoting regularization enables selecting a TF model with a minimum number of non-zero coefficient by solving a constrained optimization problem

$$\boldsymbol{\alpha} = \arg\min_{\boldsymbol{\alpha}} \frac{1}{2}\|\boldsymbol{G}\boldsymbol{\alpha} - \boldsymbol{x}\|_2^2 + \mu\|\boldsymbol{\alpha}\|_1 \tag{22}$$

where $\|\boldsymbol{\alpha}\|_p = (\sum_i |\boldsymbol{\alpha}(i)|^p)^{1/p}$ is the ℓ_p norm of a vector $\boldsymbol{x}$ and $\mu > 0$ is the sparsity parameter [20, 19]. By choosing a proper μ, one can control the resolution of the TF map and allows us to being able to discriminate between closely spaced events in time and frequency, while reconstructing the data.

The optimization problem (22) can be solved by a variety of methods such as the split Bregman method [19] or fast iterative soft thresholding algorithm (FISTA) [28]. In this study, we utilized the FISTA method to solve (22).

The obtained SP-TFR through (22) is used to design an adaptive filtering for extracting (or filtering) different phases of seismic waves. In the next section, we briefly review the adaptive filtering approach in the TF domain.

C. Adaptive filtering in the TF domain

Adaptive filtering has been extensively applied in seismology [22, 4, 7, 14]. In an intuitive scheme, Pinnegar [15] utilized a combination of inclination, azimuth, and rectilinearity attributes in the TF domain to filter the Rayleigh waves. However, his method is not able to scrutinize the pure linear polarization because of an undefined inclination angle for linear particle motions [see section (III) for more details]. As a result, the method is not applicable to filter the Love wave or any other seismic phase with a linear polarity. Here, we extend his methodology by combining rectilinearity, directivity, and amplitude attributes [7, 22] with the TF-domain polarization parameters obtained from SP-TFR of 3-components of the signal to introduce SP-TFF method.

1) Rectilinearity attribute: Rectilinearity is a critical parameter for discriminating between the elliptical and linear particle motion states. The purely rectilinear ground motion is modeled by one nonzero eigenvalue in the TF plane

$$\lambda_i(k,l) = 0, \quad i > 1. \tag{23}$$

Nevertheless, due to the presence of contaminating noise, out-of-plane energy, and scattering distortions, it is seldom

the case for the real data [29]. To circumvent this, a degree of rectilinearity

$$Re(k,l) = 1 - \frac{\lambda_2(k,l) + \lambda_3(k,l)}{\lambda_1(k,l)}, \tag{24}$$

is defined as a rectilinearity measure to discriminate between the rectilinear motion of Love and body waves and elliptical motion of Rayleigh waves [7], [13], [30].

Accordingly, a rectilinearity filter is designed in the TF domain as

$$\Psi_{Re}(Re(k,l)) = \begin{cases} 1 & -1 < Re(k,l) < \alpha, \\ \cos(\frac{\pi(Re(k,l)-\alpha)}{2(\beta-\alpha)}) & \alpha < Re(k,l) < \beta, \\ 0 & \beta < Re(k,l) < 1, \end{cases} \tag{25}$$

while to avoid the Gibbs phenomenon caused by abrupt frequency cut-off, the accept or reject regions are cosine tapered in (23) by incorporating suitably defined adjusting parameters α and β. The proposed filter is similar to the TF filter introduced by Pinnegar [15], whereas it can be designed to filter the linear polarization particle motion.

2) Directivity attribute: Directivity is another crucial parameters to discriminate between different seismic phases based on the direction of particle motion. More precisely, a directivity measure is defined as the absolute value of the dot product of the first eigenvector by the base vectors

$$D_i(k,l) = |\boldsymbol{u}_1^T(k,l)\boldsymbol{e}_i|, \; i \in \{T,R,Z\}, \tag{26}$$

then normalizing the measure in the TF plane. Correspondingly, a directivity filter is designed in the TF domain as

$$\Psi_D(D_i(k,l)) = \begin{cases} 1 & 0 < D_i(k,l) < \gamma, \\ cos(\frac{\pi(D_i(k,l)-\gamma)}{2(\lambda-\gamma)}) & \gamma < D_i(k,l) < \lambda, \\ 0 & \lambda < D_i(k,l) < 1, \end{cases}$$
$$i \in \{T,R,Z\}. \tag{27}$$

The adjusting parameters γ and λ have the role of both cosine tapering to avoid the Gibbs phenomenon and threshholding as a percentage of the maximum measure. In the next section we present the combination of these attribute to filter seismic data.

3) Amplitude attribute: Although generally surface waves manifest themselves with higher amplitude than the body and coda waves [31], it is challenging in practice to work with amplitude attributes to discriminate them. Having a TF-domain insight to analyze the signal, enforced by SP-TFR to present it with a few sparse coefficients, accentuates the amplitude difference between the surface and coda waves. In other words, the energy of surface waves is extracted locally with a few sparse coefficients, while the body and coda waves are distributed to a broader range of coefficients due to having a broader frequency content. Hence, an amplitude attribute can be more efficiently used to discriminate surface waves from body and coda waves. Specifically, it can be employed as a tool to separate Love and SH waves, which have the same type of polarization directivity and rectilinearity. Defining an amplitude attribute as

$$A(k,l) = \frac{\sqrt{2}\lambda_1(k,l)}{L}, \tag{28}$$

a corresponding amplitude filter is designed in the TF domain as

$$\Psi_A(A(k,l)) = \begin{cases} 0 & 0 < A(k,l) < \zeta, \\ cos(\frac{\pi(A(k,l)-\zeta)}{2(\eta-\zeta)}) & \zeta < A(k,l) < \eta, \\ 1 & \eta < A(k,l) < 1, \end{cases} \tag{29}$$

normalized in the whole TF plane. The adjusting parameters ζ and η acts as a measure to pass (or reject) the coefficient in the TF plane, while applying the cosine tapering. In the next section we present the combination of these attribute to filter seismic data.

4) Regularized Sparsity-promoting Time Frequency Filtering : To combine the properties of different attributes, a similar methodology to [15] can be followed. More precisely, the total TF reject filter to reject a phase is obtained by combining the rectilinearity, directivity, and amplitude filters, as

$$\boldsymbol{\Psi}_R = 1 - \{1 - \boldsymbol{\Psi}_{Re}\} \circ \{1 - \boldsymbol{\Psi}_D\} \circ \{1 - \boldsymbol{\Psi}_A\}. \tag{30}$$

Similarly, a special seismic phase can be extracted by defining an extract filter as $\boldsymbol{\Psi}_E = 1 - \boldsymbol{\Psi}_R$. Finally, the filtering process is applied by element-wise multiplication of $\boldsymbol{\Psi}$ with the SP-TFR of the three components; then, the filtered signal is reconstructed in the time domain by applying (21) giving the SP-TFF.

Besides all the defined criteria, subjective information can be incorporated as a constraint to control the filtering domain while rejecting or extracting seismic phases. As an example, in the case of filtering or extracting the surface waves, an approximate initial time of the Love wave can be introduced to the algorithm to limit the domain of filtering in (30). It can be picked visually on the transverse component or estimated by using the standard global dispersion curves.

In the next section, we examine the application of the proposed filtering method to filter different seismic wave phases.

III. Numerical examples

To evaluate the SP-TFF method, we test it with synthetic and real data examples by extracting and filtering the Love and Rayleigh waves. The implementation results are compared with those of the method introduced in Pinnegar [15].

A. Synthetic examples

The synthetic data corresponds to the source mechanism of the $M_w = 8.2$ earthquake occurred in the 101km SSW of Tres Picos, Mexico, on September 8th, 2017, 04:49:19 (UTC), as a result of normal faulting at an intermediate depth of 47.4 km [see Table. I]. The source

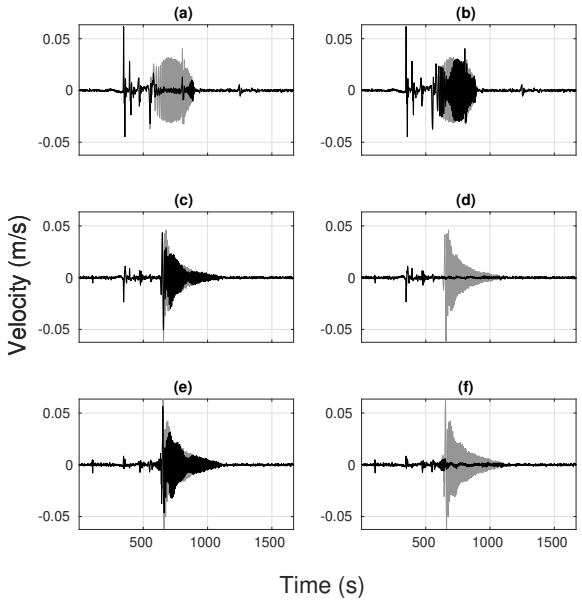


Figure 1. The background gray color waveforms in top, middle, and bottom panels corresponds to transverse, radial, and vertical components of a 3D synthetic seismogram generated for the source mechanism of $M_w = 8.2$ south-southwest of Tres Picos, Mexico [see text and Table. I]. The foreground black color diagram in panels (a), (c), and (e) are Love wave filtered transverse, radial, and vertical components by using the SP-TFF, and the panels (b), (d), and (f) are Rayleigh wave filtered waveforms of the corresponding components.

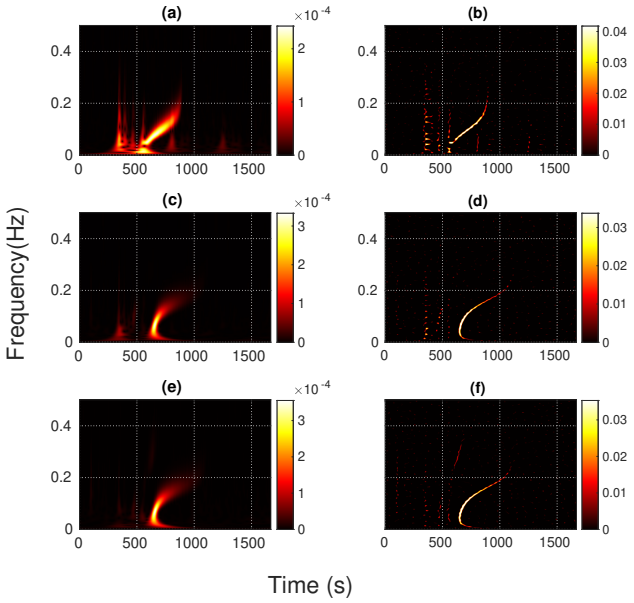


Figure 2. The TFR of the transverse, radial and vertical components of the synthetic data obtained by the ordinary ST (applied by Pinnegar [15]) are shown in the panels (a), (c), and (e). The corresponding TFR for the SP-TFR method are shown in the panels (b), (d), and (f) [Refer to the text for more explanations].

mechanism and the source-receiver geometry were chosen such that the amplitude of body and coda waves is almost comparable to the surface waves making separation more challenging.

A three-dimensional synthetic seismic data was generated through the 1D ak135f earth model [32] with spectral-element method assuming 3D (an-)elastic, anisotropic and acoustic wave propagation in spherical domains. The simulation was run by using the AxiSEM library through the IRIS Synthetics Engine (Syngine) client of ObsPy software [33, 34].

In the simulation, the seismic wavefield is recorded in the College Outpost, Alaska, USA [see Table. I for more details], at the azimuth of 61.56° to the epicenter.

The generated data were preprocessed; detrended and decimated by a factor of 8 to attain a data set with a sampling rate of 2 sec. Then, the traces were rotated to the transverse-radial-vertical coordinate system. To make the simulation more realistic, the data was contaminated by a Gaussian noises, $\boldsymbol{n} \in \mathbb{R}^{L\times 1}$ (bandpass-filtered in the range of $[0.02, 0.5]$ Hz) to give a signal to noise ratio (SNR) of 10. The Transverse, Radial, and Vertical components of the total motion are shown in gray color in Fig. 1.

To evaluate the efficiency of SP-TFF, the results are compared with those obtained by Pinnegar [15]. In this method, the ordinary ST is used as the TFR, and the TF-domain polarization parameters are obtained by fitting the particle motion to a parametric ellipse by incorporating the TFR of 3-components. More precisely, a set of the polarization parameters including $a(k,l)$ (the length of Semi-major (SM) axis of the parametric ellipse), $b(k,l)$ (the length of Semi-minor (Sm) axis of the parametric ellipse), $I(k,l)$ (the inclination of the ellipse to the horizontal), $\Omega(k,l)$ (the azimuth of the ascending node, $\omega(k,l)$ (the angle between the ascending node and the position of maximum displacement) and $\phi(k,l)$ (the phase, measured with respect to the time of maximum displacement) are obtained, with $k = 0, ..., 2n+1$ being the time and $l = 0, ..., n$ being the zero and positive frequency indices.

The TFR of the transverse, radial and vertical components of the synthetic data obtained by the ordinary ST (applied by Pinnegar [15]) are shown in the panels (a), (c), and (e) of Fig. 2; the corresponding TFR for the SP-TFR method are shown in the panels (b), (d), and (f). The SP-TFR attains a highly compact TFR with a maximum amplitude higher than the ST, while ST distributed the energy in the TF plane in a wider area. The up-chirp characteristics of surface waves are obvious in both TFRs.

The TF domain SM and Sm axes of the particle motion obtained by Pinnegar [15] method are depicted in (a) and (c) panels of Fig. 3; the corresponding TF domain SM and Sm axes for EDPA using SP-TFR are shown in in (b) and (d) panels. The SM and Sm axes for EDPA is obtained as

$$\begin{aligned} SM(k,l) &= ||\boldsymbol{SM}(k,l)||_2 = \tfrac{\sqrt{2}\lambda_1}{L}||\boldsymbol{u}_1(k,l)||_2, \\ Sm(k,l) &= ||\boldsymbol{Sm}(k,l)||_2 = \tfrac{\sqrt{2}\lambda_2}{L}||\boldsymbol{u}_2(k,l)||_2, \end{aligned} \tag{31}$$

with k and l are defined similar to (16). By considering the left panels of Figs. 2 and 3 it is evident that by applying the ordinary ST, the Rayleigh and Love waves are inseparably overlapping both in time and frequency. It can

Table I
INFORMATION OF SYNTHETIC AND REAL DATA EXAMPLE CORRESPOND TO $M_w = 8.2$ EARTHQUAKE OCCURRED NEAR COAST OF CHIAPAS, MEXICO RECORDED AT COLA STATION, IU NETWORK ALASKA, USA.

Date	Time	Hypo-lat	Hypo-Lon	Hypo-dep	Station-lat	Station-lon	Station-ele	Dis	Azimuth
2017-09-08	04:49:20(UTC)	15.022	-93.899	47.4km	64.87°	-147.86°	200m	61.56°	-22.98°

be deduced either from the TFR maps or from the SM and Sm maps. In contrast, for the SP-TFR (Right panels), the high-resolution TFR successfully separated the wavefields, giving the possibility of discriminating between different seismic wave phases. It is also seen in the SM and Sm maps in Fig. 3. Another interesting feature is the SH and Love wave pattern in the TFR. The Love wave has been concentrated around the dispersion curve; however, the SH wave has been spread in a wider frequency bandwidth around the time 800 secs. It makes using amplitude attribute more efficient in separating them. Subsequently, by incorporating the obtained TF domain polarization parameters, we process the data to filter different seismic phases.

1) Love and Rayleigh wave filtering using SP-TFF: The TF domain polarization parameters obtained form SP-TFR, are used to define and adaptive filter according to section (II-C) to filter Love and Rayleigh waves.

To extract and filter the Love waves, we design a directivity filter by defining the directivity measure with respect to the transverse axis $\boldsymbol{e}_T$ in (26), and a set of adjusting parameters $\gamma = 0.13$ and $\lambda = 0.16$ for amplitude threshholding and cosine tapering of the directivity measure. An amplitude filter by the set of parameters $\zeta = 0.26$ and $\eta = 0.23$ was combined to define a Love-reject filter accroding to (30). The results of applying the filter on the SP-TFRs of the transverse, radial and vertical components are shown in panels (a), (c), and (d) of Fig. 4. As is shown, the energy corresponds to the Love wave in the TF plane has been significantly removed, and only scattered energy remains, which corresponds to the body and coda waves, and noise (top panel). The SP-TFRs of the radial (panel (c)) and vertical (panel (e)) components have not been affected by filtering. The black color waveform in (a), (c), and (e) panels of Fig. 1 depict reconstructed transverse, radial and vertical components after filtering in the time domain; the Love wave is almost entirely removed in the time domain, while the other phases, including the body and coda wave, and also the noise has remained in the seismogram. It is a promising feature of the SP-TFF algorithm compared to the Pinnegar [15] method, which attains a null value for the inclination and azimuth parameters corresponding to a linear particle motion. The filtering process has not affected other phases in the radial and vertical components, except a minor effect on the Rayleigh phase around the time 700s, in which the SP-TFR of the Rayleigh and Love phases fully overlaps. An interesting result of applying SP-TFF in this example is that a SH phase around 800 sec masked by high-amplitude Love waves has been recovered after filtering.

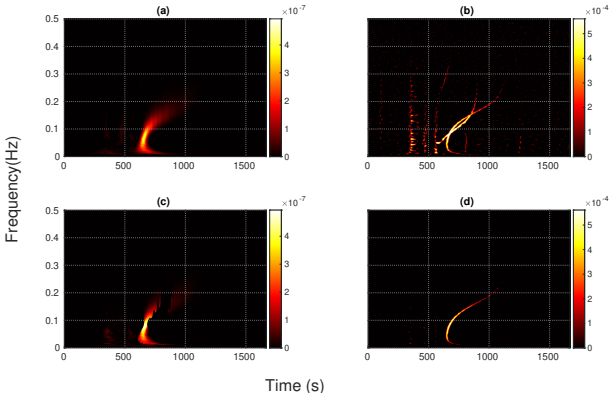


Figure 3. Panels (a) and (c): TFR of measure of SM and Sm axis of particle motion obtained by using the Pinnegar [15] method. Panels (b) and (f): The corresponding TFR of measure of SM and Sm axis of particle motion obtained by by implementing EDPA on the SP-TFR. [Refer to the text for more explanations].

To filter the Rayleigh phase, the directivity measure is computed with respect to the radial-vertical plane computed as

$$D(k,l) = \sqrt{D_R(k,l)^2 + D_Z(k,l)^2}. \tag{32}$$

The adjusting parameters are set to $\gamma = 0.25$ and $\lambda = 0.3$. Furthermore, a rectilinearity filter is defined by setting the parameters $\alpha = 0.1$ and $\beta = 0.12$. The results of applying the filter on the SP-TFRs of 3-components are shown in the panels (b), (d), and (f) of Fig. 1. Similar to the Love wave filtering, the filtered SP-TFR only contains scattered energy of the noise, body, and coda waves in the radial and vertical components. The reconstructed filtered components are shown in the right panels shown in the right panel of Fig. 1. As shown, SP-TFF successfully filtered the Rayleigh wave without affecting the body and coda waves in the radial and vertical components and substantially affected the other phases in the transverse components. The same as for the Love wave filtering, around 800s, the Love wave has slightly been filtered. The results obtained from the SP-TFF are superior to those from Pinnegar [15] by having a very high-resolution TFR enable to separate the Rayleigh and Love waves, while in the ordinary ST the TF resolution is limited.

As a final test to assess the SP-TFF method, the Love and Rayleigh phases are extracted by applying (30) on the SP-TFR of three components. The extracted Love wave is shown in the (a) panel of Fig. 5. Similarly, panels (b) and (c) of this figure show the radial and vertical components of Rayleigh waves. Both the Love and Rayleigh phases have been cleanly extracted from the entire waveform. The extracted surface waves can be used as an input to other

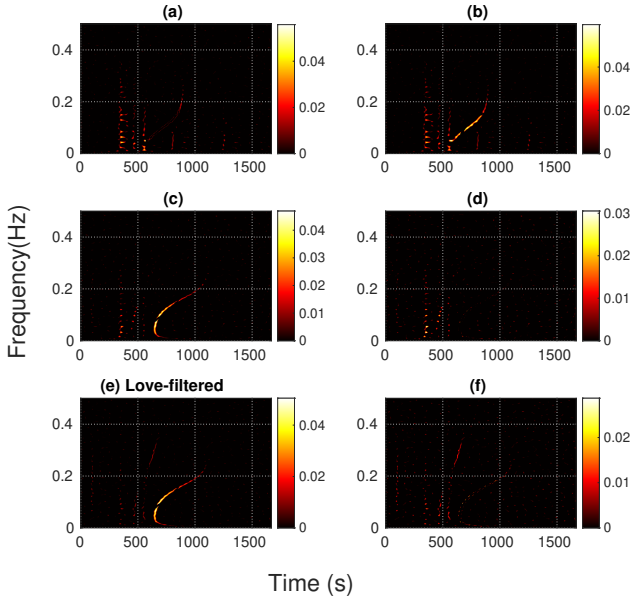


Figure 4. Left panel: Adaptively Filtered SP-TFRs of the transverse, radial, and vertical components of synthetic data to eliminate the Love wave. Right panel: Adaptively Filtered SP-TFRs of components to eliminate the Rayleigh wave.

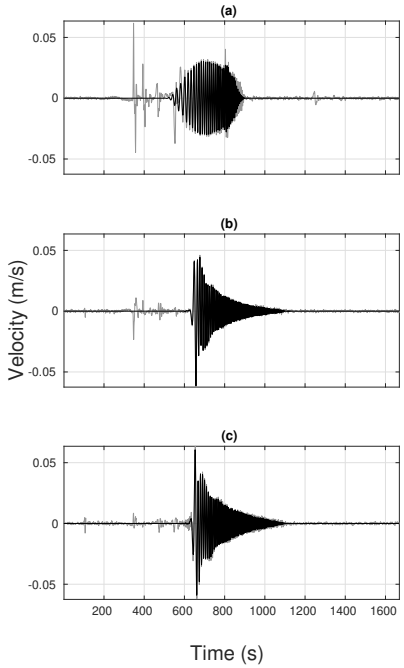


Figure 5. Panel (a): extracted Love wave by applying SP-TFF method on the synthetic data. Panel (b) and (c): the radial and vertical component of extracted Rayleigh wave by applying SP-TFF method.

processing methods like dispersion curve inversion. In the following subsection, we examine the performance of the method on real data.

B. Real data example

The real seismogram corresponds to the real data recorded for the same earthquake and station of the synthetic model. The waveform was pre-processed including detrending, decimation, deconvolving the instrument response, and converting to velocity with a sampling rate of 2 sec. The traces were mapped to the transverse-radial-vertical coordinate system. The pre-processed transverse, radial, and vertical components are shown in gray color in the top, middle, and bottom panels of Fig. 6.

The obtained TFR by applying [15] and SP-TFR methods are shown in the left and right panels of Fig. 7; similar to the synthetic example, the SP-TFR (right panels) presents a highly compact TFR comparing to the ordinary ST implemented by Pinnegar [15] (left panels). Although there is no sharp up-chirp pattern for the surface waves like in the synthetic data, there are still two separate energy panels in the SP-TFR of different components showing an increasing value of frequency by time. These two panels marked by dash-dot and continuous line ellipsis correspond to Love and Rayleigh waves, respectively. Two other panels shown by dashed ellipsis contain mostly body and coda waves. On the other hand, the TFR obtained by ST and shown in the left panel of Fig. 7 depicts a mixed and inseparable pattern of Love and Rayleigh waves. The distinct polarization pattern between the Love and Rayleigh waves is better visible in the TF domain SM and Sm axes of particle motion as shown in the right panel of Fig. 8. The elliptical particle motion of Rayleigh waves is separable from the Linear particle motion of Love, body, and coda waves in the SM and Sm axes figures. Contrarily, they have been mixed in time and frequency in the results obtained by Pinnegar [15] method, as shown in the left panel of Fig. 8. In the following subsection, we perform the adaptive filtering method to reject or extract Love and Rayleigh waves.

1) Love and Rayleigh wave filtering using SP-TFF: The TF domain polarization parameters obtained from SP-TFR are used to design an adaptive filter to filter Love and Rayleigh waves. For the real data, similar adjusting parameters of $\gamma = 0.1$, $\lambda = 0.13$, $\alpha = 0.03$, and $\beta = 0.04$ to the ones defined for the synthetic example were set to design directivity and rectilinearity attributes. We only slightly changes the for amplitude filtering parameters by setting $\zeta = 0.16$ and $\eta = 0.19$. A visual assessment chose an initial time of 630 seconds as a constraint to limit the filtering region. It affects to discriminate between the high amplitude of SH waves and Love waves. domain while rejecting or extracting seismic phases The (a), (c), and (e) panels of Fig. 9 show the filtered SP-TFRs of the transverse, radial, and vertical components of the data, respectively. As it can be seen, the focused energy corresponding to the Love waves [shown by blue oval in

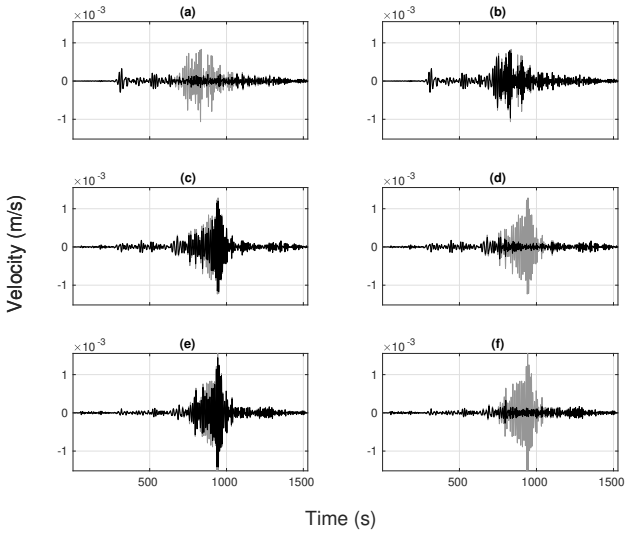


Figure 6. The background gray color waveforms in top, middle, and bottom panels corresponds to transverse, radial, and vertical components of $M_w = 8.2$ earthquake [see text and Table. 1]. The foreground black color diagram in panels (a), (c), and (e) are Love wave filtered transverse, radial, and vertical components by using the SP-TFF, and the panels (b), (d), and (f) are corresponding Rayleigh wave filtered components.

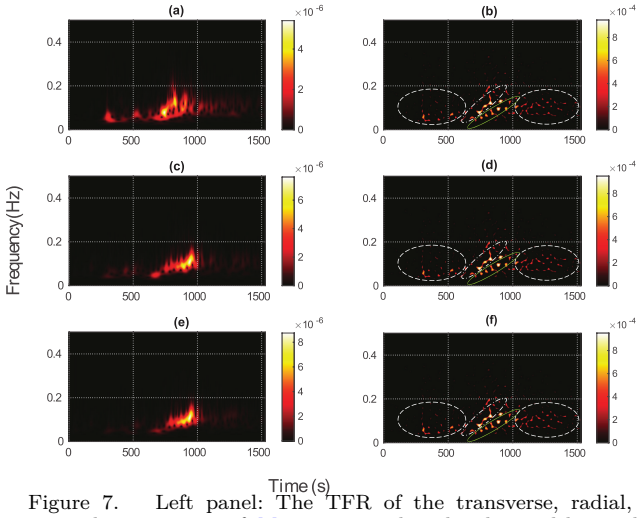


Figure 7. Left panel: The TFR of the transverse, radial, and vertical compenents of $M_w = 8.2$ earthquake obtained by applying conventional ST. Right panel: The SP-TFR of the components.

Figs. 8 and 7] are highly damped as a result of filtering, and only a scattered signal corresponding to body and coda waves remained in the transverse component. The SP-TFRs of the radial (panel (c)) and vertical (panel (e)) components have not significantly affected by filtering. The black color waveforms at the (a), (e), and (e) panels of Fig. 6 depict reconstructed signal for the transverse, radial, and vertical components in the time domain. The results confirm that the SP-TFF filtering significantly canceled the Love wave in the time domain without affecting other phases, including the body and coda waves.

To filter the Rayleigh phase, the adjusting parameters

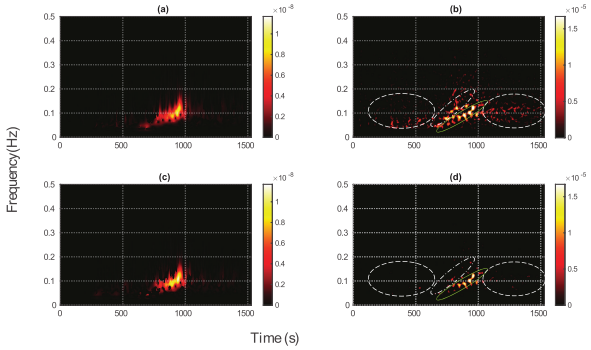


Figure 8. Left panel: The TFR of the length of the SM and Sm axis of particle motion of the transverse, radial, and vertical components of $M_w = 8.2$ earthquake obtained by using the Pinnegar [15] method. Right: SM and Sm axes by implementing EDPA on the SP-TFR.

of the directivity measure are set to $\gamma = 0.1$ and $\lambda = 0.13$, and the rectilinearity filter is set to have $\alpha = 0.03$ and $\beta = 0.04$ adjusting parameters. The results of applying the filter on the SP-TFRs of 3-components are shown in the right panels of Fig. 9. Like the Love wave filtering, the filtered SP-TFR only contains scattered energy of the noise, body, and coda waves in the radial and vertical components. The reconstructed filtered components are shown in the right panels shown in the right panel of Fig. 6. As shown, SP-TFF successfully filtered the Rayleigh wave without affecting the body and coda waves in the radial and vertical components and substantially affected the other phases in the transverse components.

Finally, we extracted the Love and Rayleigh phases for the real data set by applying (30) on the SP-TFR of three components. The extracted Love wave is shown in the (a) panel of Fig. 10. Similarly, panels (b) and (c) of this figure show the radial and vertical components of Rayleigh waves. Both the Love and Rayleigh phases have been cleanly extracted from the entire waveform, without any inclusion of body and coda waves. In the following subsection, we present a discussion on the SP-TFF method.

IV. Discussion and Conclusions

We presented a SP-TFF method by combining SP-TFR and EDPA methods as a robust seismic processing tool to separate different phases of seismic waves according to their polarization state. Taking advantage of SP-TFR, high-resolution polarization information is attained to be analyzed by TF-domain polarization attributes for resolving closely spaced seismic events in time and frequency. Conducting numerical examples on synthetic and real earthquake data, we showed that the SP-TFF can be used as a sophisticated tool to filter elliptical and linear particle motion by designing suitably defined directivity, rectilinearity, and amplitude attributes. Remarkably, not only SP-TFF is efficient to filter Love and Rayleigh waves from the other seismic phases, but it also can handle a more challenging problem of discrimination between the Love and SH and coda waves. This is a promising feature of this method.

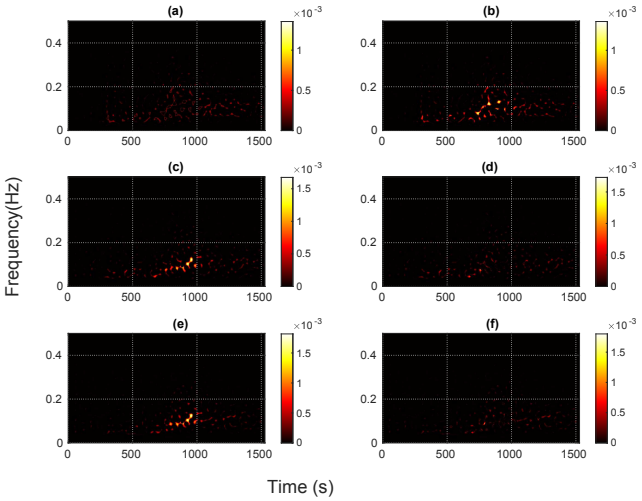


Figure 9. Left panel: Adaptive Filtered SP-TFRs of the transverse, radial, and vertical components of the $M_w = 8.2$ earthquake to eliminate the Love wave. Right panel: Adaptive Filtered SP-TFRs of components to eliminate the Rayleigh wave.

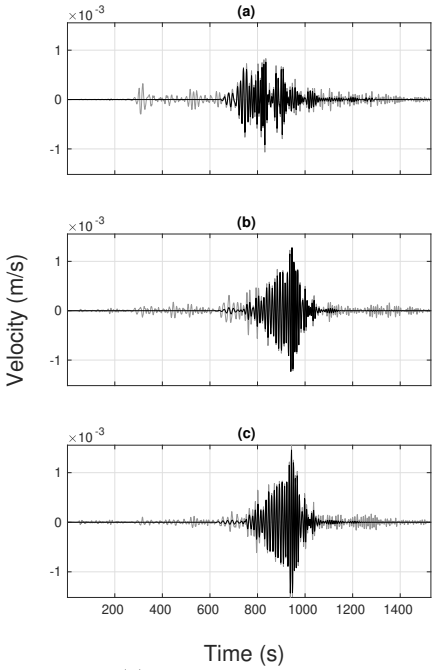


Figure 10. Panel (a): extracted Love wave by applying SP-TFF method on the synthetic data. Panel (b) and (c): the radial and vertical component of extracted Rayleigh wave by applying SP-TFF method.

SP-TFF can find application in various seismic processing methods like anisotropy parameters estimation using the Shear Wave Splitting method [35], surface waves extraction for dispersion curve inversion [36] and sensor miss-orientation test [21], and elimination of the surface waves to extract the coda waves.

The highest computational cost of the algorithm is due to solving the regularized inverse problem (22). As a result, considerable memory space and high computation time are required for a massive input data set. Furthermore, the weakly stationary condition assumption in (5) violates for very low frequencies. Notwithstanding, these are drawbacks of SP-TFF methods.

This paper intended to present the methodology of SP-TFF; only processing earthquake waveforms evaluated the efficiency. The research is ongoing with studying other critical issues, including (a) stability of SP-TFF at different SNR levels, (b) evaluation of the efficiency of SP-TFF for processing ambient noise data, (c) application of SP-TFF for extraction of low amplitude seismic phases, and (d) extraction of surface wave dispersion curves using SP-TFF.

V. Code and data availability

The numerical results from the synthetic and real data examples presented in this paper are reproducible by running a set of computer codes available at the Github account ("Will be inserted when the manuscript is accepted").

VI. Acknowledgments

This research was funded by Fundação para a Ciência e a Tecnologia (FCT) in the content of SHAZAM (Ref. PTDC/CTA-GEO/31475/2017) project. This work is funded by FCT/MCTES through national funds and when applicable co-funded EU funds under the project UIDB/EEA/50008/2020.

References

[1] K. Aki and P. G. Richards, *Quantitative seismology*, 2002.

[2] C. Wang, Y. Wang, P. Sun, and Y. Li, "Discussions on the processing of the multi-component seismic vector field," *Applied Sciences*, vol. 9, no. 9, p. 1770, 2019.

[3] E. Flinn, "Signal analysis using rectilinearity and direction of particle motion," *Proceedings of the IEEE*, vol. 53, no. 12, pp. 1874–1876, 1965.

[4] J. E. Vidale, "Complex polarization analysis of particle motion," *Bulletin of the Seismological society of America*, vol. 76, no. 5, pp. 1393–1405, 1986.

[5] R. Simons, "A surface wave particle motion discrimination process," *Bulletin of the Seismological Society of America*, vol. 58, no. 2, pp. 629–637, 1968.

[6] J. Samson and J. Olson, "Some comments on the descriptions of the polarization states of waves," *Geophysical Journal International*, vol. 61, no. 1, pp. 115–129, 1980.

[7] A. Jurkevics, "Polarization analysis of three-component array data," *Bulletin of the seismological*

society of America, vol. 78, no. 5, pp. 1725–1743, 1988.

[8] J. Allen, "Short term spectral analysis, synthesis, and modification by discrete fourier transform," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 25, no. 3, pp. 235–238, 1977.

[9] R. G. Stockwell, "A basis for efficient representation of the s-transform," *Digital Signal Processing*, vol. 17, no. 1, pp. 371–393, 2007.

[10] S. Ventosa, C. Simon, M. Schimmel, J. J. Dañobeitia, and A. Mànuel, "The *s*-transform from a wavelet point of view," *IEEE Transactions on Signal Processing*, vol. 56, no. 7, pp. 2771–2780, 2008.

[11] M. Kulesh, M. Diallo, M. Holschneider, K. Kurennaya, F. Krüger, M. Ohrnberger, and F. Scherbaum, "Polarization analysis in the wavelet domain based on the adaptive covariance method," *Geophysical Journal International*, vol. 170, no. 2, pp. 667–678, 2007.

[12] Y.-Y. Tan, C. He, Y.-D. Wang, and Z. Zhao, "Ground roll attenuation using a time-frequency dependent polarization filter based on the s transform," *Applied Geophysics*, vol. 10, no. 3, pp. 279–294, 2013.

[13] M. Schimmel and J. Gallart, "The use of instantaneous polarization attributes for seismic signal detection and image enhancement," *Geophysical Journal International*, vol. 155, no. 2, pp. 653–668, 2003.

[14] M. Schimmel, E. Stutzmann, F. Ardhuin, and J. Gallart, "Polarized earth's ambient microseismic noise," *Geochemistry, Geophysics, Geosystems*, vol. 12, no. 7, 2011.

[15] C. Pinnegar, "Polarization analysis and polarization filtering of three-component signals with the time—frequency s transform," *Geophysical Journal International*, vol. 165, no. 2, pp. 596–606, 2006.

[16] I. Daubechies, J. Lu, and H.-T. Wu, "Synchrosqueezed wavelet transforms: An empirical mode decomposition-like tool," *Applied and computational harmonic analysis*, vol. 30, no. 2, pp. 243–261, 2011.

[17] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM review*, vol. 43, no. 1, pp. 129–159, 2001.

[18] O. Portniaguine and J. Castagna, "Inverse spectral decomposition," in *SEG Technical Program Expanded Abstracts 2004*. Society of Exploration Geophysicists, 2004, pp. 1786–1789.

[19] A. Gholami, "Sparse time–frequency decomposition and some applications," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 51, no. 6, pp. 3598–3604, 2012.

[20] I. Vera Rodriguez, D. Bonar, and M. Sacchi, "Microseismic data denoising using a 3c group sparsity constrained time-frequency transform," *Geophysics*, vol. 77, no. 2, pp. V21–V29, 2012.

[21] A. O. Ojo, L. Zhao, and X. Wang, "Estimations of sensor misorientation for broadband seismic stations in and around africa," *Seismological Research Letters*, vol. 90, no. 6, pp. 2188–2204, 2019.

[22] J. F. Montalbetti and E. R. Kanasewich, "Enhancement of teleseismic body phases with a polarization filter," *Geophysical Journal International*, vol. 21, no. 2, pp. 119–129, 1970.

[23] A. Plešinger, M. Hellweg, D. Seidl *et al.*, "Interactive high-resolution polarization analysis of broadband seismograms," *Journal of Geophysics/ IF 32.18*, vol. 59, no. 1, pp. 129–139, 1986.

[24] J. P. Jones, D. W. Eaton, and E. Caffagni, "Quantifying the similarity of seismic polarizations," *Geophysical Journal International*, vol. 204, no. 2, pp. 968–984, 2016.

[25] J. Samson, "Descriptions of the polarization states of vector processes: applications to ulf magnetic fields," *Geophysical Journal International*, vol. 34, no. 4, pp. 403–419, 1973.

[26] S. Mallat, *A wavelet tour of signal processing*. Elsevier, 1999.

[27] P. Flandrin, *Time-frequency/time-scale analysis*. Academic press, 1998.

[28] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM journal on imaging sciences*, vol. 2, no. 1, pp. 183–202, 2009.

[29] H. Sato, M. C. Fehler, and T. Maeda, *Seismic wave propagation and scattering in the heterogeneous earth*. Springer Science & Business Media, 2012.

[30] M. Schimmel and J. Gallart, "Degree of polarization filter for frequency-dependent signal enhancement through noise suppression," *Bulletin of the Seismological Society of America*, vol. 94, no. 3, pp. 1016–1035, 2004.

[31] B. A. Bolt and B. A. Bolt, *Inside the Earth: Evidence from earthquakes*. WH Freeman San Francisco, 1982.

[32] J.-P. Montagner and B. Kennett, "How to reconcile body-wave and normal-mode reference earth models," *Geophysical Journal International*, vol. 125, no. 1, pp. 229–248, 1996.

[33] M. Beyreuther, R. Barsch, L. Krischer, T. Megies, Y. Behr, and J. Wassermann, "Obspy: A python toolbox for seismology," *Seismological Research Letters*, vol. 81, no. 3, pp. 530–533, 2010.

[34] M. van Driel, A. Hutko, L. Krischer, C. Trabant, S. Stähler, and T. Nissen-Meyer, "Syngine: on-demand synthetic seismograms from the iris dmc based on axisem & instaseis," in *EGU General Assembly Conference Abstracts*, 2016, pp. EPSC2016–8190.

[35] P. G. Silver and W. W. Chan, "Shear wave splitting and subcontinental mantle deformation," *Journal of Geophysical Research: Solid Earth*, vol. 96, no. B10, pp. 16 429–16 454, 1991.

[36] M. Maraschini and S. Foti, "A monte carlo multimodal inversion of surface waves," *Geophysical Journal International*, vol. 182, no. 3, pp. 1557–1566, 2010.

Appendix C

Eigenvalue Decomposition Polarization Analysis: A regularized sparsity-based approach

In the present appendix it is shown, in Fig. C.1, the poster used for presenting the submitted and accepted abstract for the 2021 EGU conference.

EGU General Assembly 2021

Eigenvalue decomposition polarization analysis: A regularized sparsity-based approach

Almeida, E., Mohammadigheymasi, H., Fathi, M., Crocker, P., and Silveira, G.

Universidade da Beira Interior

Polarization analysis is a signal processing tool for decomposing multi-component seismic signals to a set of rectilinearly or elliptically polarized elements. Theoretically, time-frequency (TF)-domain polarization methods are the most compatible tools for analyzing the intrinsically non-stationary seismic signals. The TF-domain polarization methods decompose the signal to a superposition of polarized elements with well defined polarization properties, localized in the time and frequency domains. On the other hand, in practice, it suffers from a lack of resolution in the time and frequency, unable to discriminate between interfering seismic phases, and instability in the presence of noise, as they rely on a generally an underdetermined and low-resolution time-frequency representation (TFR) method. In this study, we propose a method to circumvent this problem.

TF domain cross-correlation parameters is obtained by incorporating TFD of the 3-components of the signal, xi (i = 1, 2, 3)

$$C_{ij}(k,l) = \begin{cases} g_{ij}(k,-l) + g_{ij}(k,l) & l \neq 0 \\ g_{ij}(k,l) & l = 0 \end{cases} k = 0, 1, ..., 2n+1,\ l = 0, 1, ..., n \quad (1)$$

where,

$$g_{ij} = TF_i \circ TF_j^* \quad (2)$$

then,

$$(\mathbf{C}(k,l) - \lambda_i(k,l)\mathbf{I})\mathbf{u}(k,l) = 0, \quad (3)$$

attains, $(u_1(k,l), u_2(k,l), u_3(k,l))$ $(\lambda_1(k,l), \lambda_2(k,l), \lambda_3(k,l))$, TF-domain eigenvalues and eigenvectors.

By using Wavelet transform as TFR, we can obtain TF coefficients as a set of linear of equations:

$$x = W\alpha, \quad (4)$$

The system above can be solved as a constrained optimization problem imposing sparsity [Portniaguine and Castagna(2004)]:

$$\alpha = \arg\min_{\alpha} \frac{1}{2}\|W\alpha - x\|_2^2 + \mu\|\alpha\|_1 \quad (5)$$

Rectilinearity and Directivity attribute:

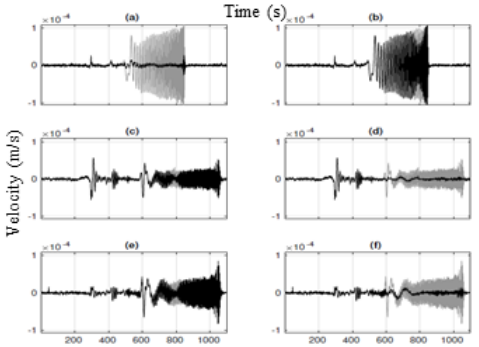
$$Re(k,l) = 1 - \frac{\lambda_2(k,l) + \lambda_3(k,l)}{\lambda_1(k,l)}, \quad D_i(k,l) = |u_1^T(k,l)e_i|,\ i \in \{T, R, Z\}, \quad (6)$$

Ciências ULisboa

$$\Psi_{Re}(Re(k,l)) = \begin{cases} 1 & -1 < Re(k,l) < \alpha, \\ \cos(\frac{\pi(Re(k,l)-\alpha)}{2(\beta-\alpha)}) & \alpha < Re(k,l) < \beta, \\ 0 & \beta < Re(k,l) < 1, \end{cases} \quad (7)$$

$$\Psi_D(D_i(k,l)) = \begin{cases} 1 & 0 < D_i(k,l) < \gamma, \\ \cos(\frac{\pi(D_i(k,l)-\gamma)}{2(\lambda-\gamma)}) & \gamma < D_i(k,l) < \lambda, \\ 0 & \lambda < D_i(k,l) < 1, \end{cases} \quad (8)$$

$i \in \{T, R, Z\}$.



Love (Left panel) and Rayleigh waves (Right panel) elimination by sparsity-based approach.

Applications of the proposed method:

- Extracting surface waves as an input to surface wave dispersion curve inversion methods.
- Eliminating surface waves to prepare the earthquake data for coda waves studies.
- Extracting Rayleigh waves for sensor miss-orientation tests.

References:

O. Portniaguine and J. Castagna, "Inverse spectral decomposition," in SEG Technical Program Expanded Abstracts 2004. Society of Exploration Geophysicists, 2004, pp. 1786–1789.

ISEL

Figure C.1: Poster for the 2021 EGU conference.