Data-driven pattern recognition of seismic wind turbine emissions with machine learning

Master's thesis of

Marie Arnika Gärtner

at the Geophysical Institute (GPI)
KIT-Department of Physics
Karlsruhe Institute of Technology (KIT)

Date of submission:
16.05.2024

Supervisor:	Prof. Dr. Joachim R. R. Ritter
Co-supervisor:	Prof. Dr. Thomas Bohlen
Advisor:	Dr. René Steinmann (GFZ)
Co-advisor:	Dr. Laura Gaßner

Erklärung / Statutory declaration

Ich versichere wahrheitsgemäß, die Arbeit selbstständig verfasst, alle benutzten Hilfsmittel vollständig und genau angegeben und alles kenntlich gemacht zu haben, was aus Arbeiten anderer unverändert oder mit Abänderungen entnommen wurde sowie die Satzung des KIT zur Sicherung guter wissenschaftlicher Praxis in der jeweils gültigen Fassung beachtet zu haben.

I declare truthfully that I have written this thesis by myself, that I have fully and accurately specified all aids used, that I have correctly cited everything that was taken, either unchanged or with modification, from the work of others, and that I have complied with the current version of the KIT statutes for safeguarding good scientific practice.

Karlsruhe, 16.05.2024

Signature: Marie Arnika Gärtner

Acknowledgments

The roots of this work go back much further than the registration of this thesis. That is why I would like to thank you first, Laura Gaßner, for hiring me as a student assistant and for the many great hours we spent working together. This thesis would not have been possible without you. Thank you for proofreading my work and for the many helpful discussions we had! Joachim Ritter, I value your patience in the seminar and your mentorship. I have learned a lot from you during my master's studies, and I am very grateful for the probing questions that helped me to move forward. Thomas Bohlen, I appreciate that you accepted to be my co-supervisor and for the opportunity to present my work in your seminar. Special thanks to René Steinmann, who became my advisor for this work without knowing what he was getting himself into. Thank you for your help and our many enjoyable discussions. I thank Thomas Forbriger, Thomas Hertweck, and Leland McInnes for their contributions to my work. Jan-Phillip Föst, I am grateful for your support, our discussions, and your proofreading with all your helpful comments. Svenja Heyns, I would like to extend my sincere thanks for your careful proofreading and your outside perspective on my work. Yvonne Fröhlich thank you for your PyGMT tutorial and the delightful wind turbine markers. Rune Helk, it was a pleasure having you as an office partner. Thank you for standing by my side when I needed someone to listen to me debug my code. Sarah Mader thank you for trying to understand what I did, making me explain it understandably, and for the enjoyable walks. With that, I would like to thank the entire "Seismo" working group for all the great discussions and for making the effort to understand what I was doing or simply interpreting whether my UMAP atlases showed a dragon or perhaps a bear following a swarm of bees. You made my time at the seminar and GPI a pleasure, and I wouldn't want to miss it. And lastly, my parents, for their loving support and for believing in me, even if it was not always easy.

Abstract

Seismic emissions from wind turbines (WTs) affect the quality of seismological measurements, especially for the investigation of local earthquakes or for the monitoring of geothermal operations. Not all sources of the observed WT emissions are understood, and other related patterns may even be unknown. However, understanding these emissions is crucial to address and mitigate this problem.

This study presents a workflow employing unsupervised machine learning techniques to identify patterns in WT emissions utilizing ground motion data collected during four months in 2022 and 2023 in the Inter-Wind project near the Tegelberg wind farm in southwest Germany.

Aiming to extract known and unknown patterns of seismic WT emissions in a data-driven fashion, the hierarchical clustering algorithm HDBSCAN is applied, allowing a layered investigation of the clustering results. To ensure robust clustering of the dataset, a translation-invariant representation is essential. As the scattering transform proves effective in multiple seismological studies, it is applied before the clustering using the scatseisnet Python package.

The clustering results reveal distinct patterns correlating with wind direction, WT rotation rate, and wind speed, contributing to a better understanding of seismic WT emissions. Thus, the developed workflow is an important step toward the decorrelation of seismic WT emissions from meteorological conditions.

Contents

List of Figures

List of Tables

List of Abbreviations

mcs minimum cluster size. 45–47

BFO Black Forest Observatory. 1

BPF blade passing frequency. 1, 2, 14, 15, 21, 22, 29, 32, 34, 37, 39, 40, 59, 61, 63, 67, 68, 72, 79–81, 84

HDBSCAN hierarchical density-based spatial clustering of applications with noise. xii, xiii, 3, 40, 44, 45, 47, 48, 50, 53, 56–59, 61, 64–66, 68–70, 72, 73, 75, 76, 80, 81, 83, 84, 108

ML machine learning. 2, 3, 39, 79, 83

PC principal component. 40, 41

PCA principal component analysis. 40, 41

PSD power spectral density. xi, 22, 23

RMS root mean square. 21, 28, 99

rpm rotation per minute. 10, 14

SVD singular value decomposition. 41

UMAP uniform manifold approximation and projection. xi–xiii, xv, 3, 40–43, 47–59, 61, 68, 80, 81, 83, 84, 103–109

WF wind farm. xi, xii, 5–7, 9–12, 14, 15, 79, 83, 85, 86

WT wind turbine. xi, xiii, xv, 1–3, 5, 6, 8–15, 17, 21, 23, 29, 32, 37, 39, 40, 44, 50, 52, 53, 59, 61–64, 67, 68, 70, 72, 73, 76, 79–81, 83–85, 106, 107

Chapter 1

Introduction

The importance of wind energy in the transition to renewable energy cannot be overstated. The first section of the German Renewable Energy Act (EEG 2023) states that the share of electricity generated from renewable energy sources is to be increased to at least 80 percent by 2030 (§1 paragraph 2, EEG 2023, 2024). To achieve this, onshore wind turbine (WT) are to be expanded to a total installed capacity of 115 GW by 2030 (§4 paragraph 1d, EEG 2023, 2024). In 2023 alone, 745 new onshore WT were built in Germany, increasing their total number to 28 677 and the total installed capacity to 61 GW (Deutsche WindGuard GmbH, 2023).

The map insert, shown in Figure 1.1, illustrates the density of WTs in Germany in 2020. Compliance with EEG 2023 (2024) requires not only an increase in density but also an increase in the capacity of onshore WTs. The drawback of the increasing number of WTs, however, is that they not only generate acoustic noise but also constantly emit seismic waves due to the coupling of the foundation of the WTs with the subsurface. For seismic stations, or more generally for sensitive instruments, this leads to an increase in the noise level (Saccorotti et al., 2011; Styles et al., 2005; Zieger and Ritter, 2018). For instance, the permanent seismic measurements at Black Forest Observatory (BFO) are most affected in the frequency band between 1.1 Hz and 1.2 Hz (T. Forbriger, personal communication, May 4, 2024). Figure 1.2 shows the significant increase in the noise level in this frequency range over the past eleven years.

Projects like Inter-Wind or DB Miss are conducted to understand these emissions better, both to study the effects on humans themselves and to study the effects on seismic measurements (Gaßner et al., 2022; *Project DB Miss* 2024). The emitted ground motion velocity of WTs is far below $100\,\mu\mathrm{m\,s}^{-1}$, which is the human perceptibility according to Table 1 of DIN 4150-2, and therefore these emissions cannot be felt by humans. However, this does not apply to seismic sensors. The ground motion recordings in the vicinity of WTs are strongly dominated by mono-frequent signals corresponding to the eigenmodes of WT and multiples of their blade passing frequency (BPF). The dominant emissions are in the frequency range of 0.1 Hz to 20 Hz (Nagel et al., 2021). Furthermore, e.g., Gaßner et al. (2023), Nagel et al. (2021), and Saccorotti et al. (2011) showed that the WT signals are still measurable at a distance of several kilometers.

However, in the same frequency range we find signals of local earthquakes or earthquakes triggered by geothermal operations (Charléty et al., 2007; Hensch et al., 2019). The monitoring of the former is important to better understand the processes in the subsurface and the latter to monitor and steer down the geothermal energy production in case the subsurface activity becomes too severe. Both are challenging in the case of nearby WTs

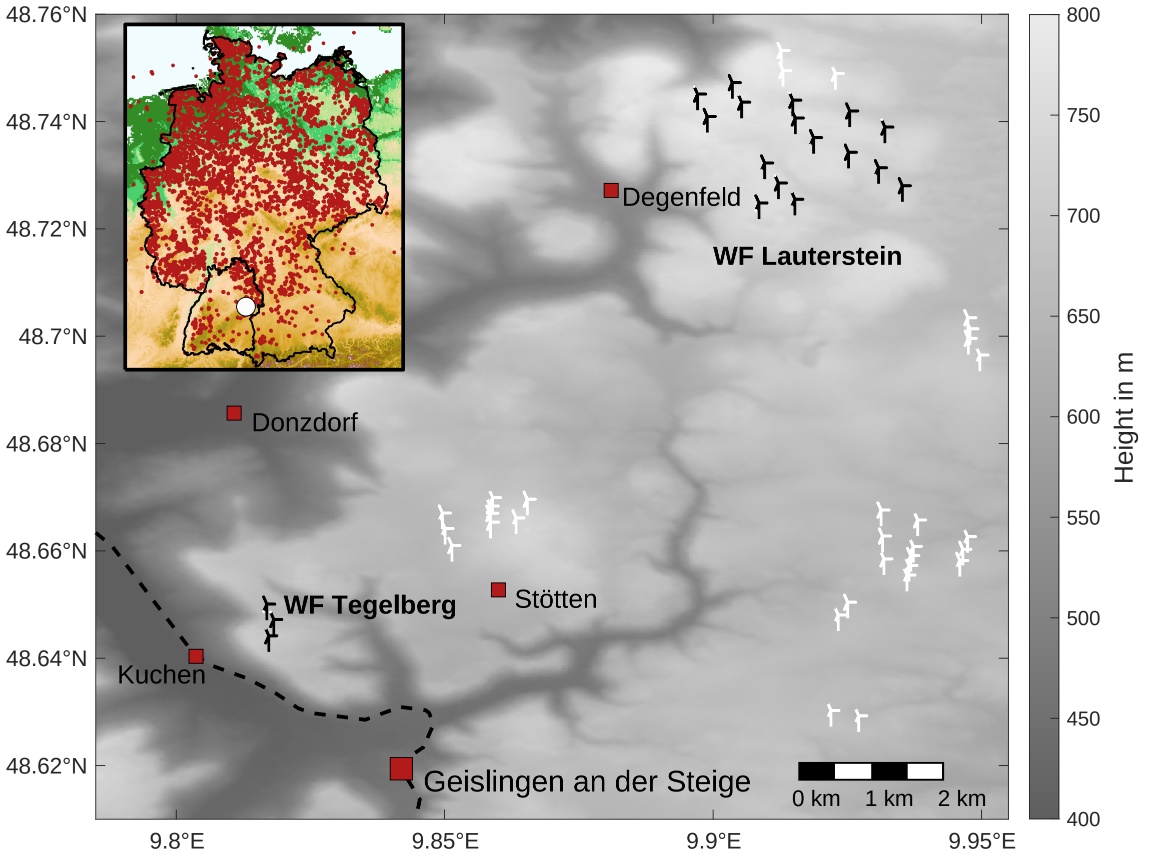


Figure 1.1: Map of the region investigated as part of the Inter-Wind project on the eastern Swabian Alb, southwest Germany. The black wind turbine (WT)s are associated with the investigated wind farms, in particular, Lauterstein (16 WTs) and Tegelberg (three WTs), from which we received operational data. The remaining WTs in the area are shown in white. The dashed line shows the main railroad line between Stuttgart and Ulm. The inset map provides an overview of all WTs in Germany (Bundesnetzagentur, 2022). Modified from Gaßner and Ritter (2023b).

due to the effect of seismic WT emissions on the quality of seismic recordings. Therefore, understanding the emissions of WTs is key to find a way to address this problem.

In my study, I develop a method to extract signal characteristics of seismic WT emissions in a data-driven fashion. My goal is to identify known seismic WT signal patterns such as the eigenmodes or multiples of the BPF, but also to better understand the signal sources of seismic WT emissions and to identify unknown features. For this, I apply machine learning (ML) methods on data recorded during the Inter-Wind project on the eastern Swabian Alb in southwest Germany (Fig. 1.1; Gaßner et al., 2022).

ML is a subfield of artificial intelligence, intended to find patterns and correlations within datasets. It includes three main approaches: reinforcement learning, supervised, and unsupervised learning (Bishop, 2006). Reinforcement learning is a trial and error algorithm evaluated, e.g., by a cost function to find an optimal output (Bishop, 2006). Reinforcement learning is used for instance in robotics. A labeled training dataset is used in supervised learning to train a function that classifies an unlabeled test dataset (Bishop, 2006). To produce the training dataset, preliminary information is required to label the dataset. The more data are available to train, the more reliable is the outcome. In this study, I want to characterize both known and unknown patterns. Therefore, a supervised approach is not suitable. Unsupervised ML techniques do not require a labeled training dataset, and procedures are the clustering of a dataset, dimension reduction, or density estimation to

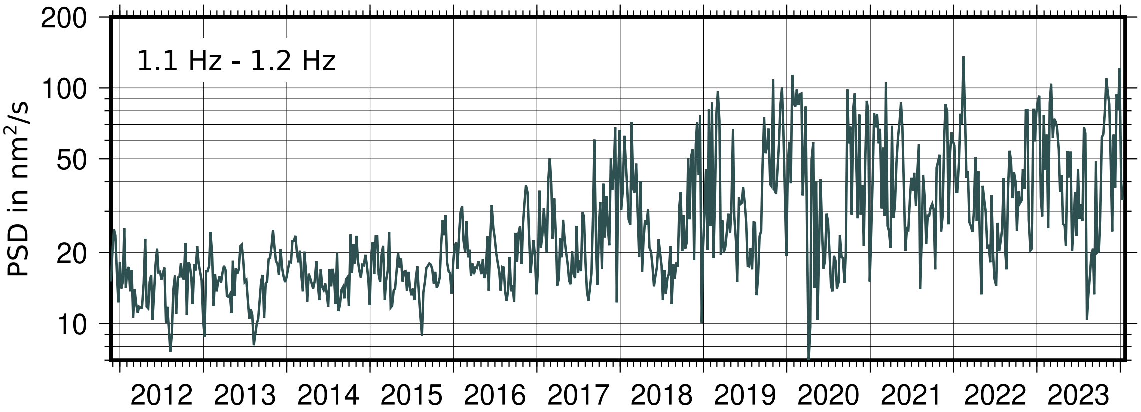


Figure 1.2: Noise increase in the frequency band 1.1 Hz to 1.2 Hz between 2012 and 2023 recorded at the Black Forest Observatory (BFO) in SW Germany. Shown is the vertical component of the STS-2 (BHZ GR) at BFO. The frequency range between 1.1 Hz and 1.2 Hz is most significantly influenced by WT emissions. Contributed and modified from Rudolf Widmer-Schnidrig, BFO.

determine the underlying distribution of a dataset (Bishop, 2006).

The application of ML techniques in the field of geophysics has become firmly established, as demonstrated, for example, by the development of algorithms for earthquake detection (e.g., Münchmeyer et al., 2022) and earthquake early warning algorithms (Kong et al., 2016; Ochoa et al., 2018; Reddy and Nair, 2013). Heuel and Friederich (2022) have recently presented a denoising method for seismic WT emissions. Nevertheless, the identification and extraction of signal sources from seismic WT emissions with ML remains an unsolved challenge to the best knowledge. This study aims to address this gap by using a scattering transformation prior to applying dimension reduction with UMAP and the clustering algorithm HDBSCAN (McInnes et al., 2017, 2018; Seydoux et al., 2020).

In Chapter 2, I will present the data used in this study and highlight the known signal characteristics of seismic WT emissions. Following this, I will extract the redundant signal sources from the seismic recordings using the scattering transformation (Chapter 3). Afterward, I will group and classify these signal sources by applying dimensional reduction and clustering techniques in Chapter 4. Finally, I will summarize and discuss my findings in Chapter 5 and present the conclusions and future work in Chapter 6.

Chapter 2

Data

This chapter presents the data analyzed within the scope of this work. The data was collected during the Inter-Wind project (Gaßner et al., 2022). I will present four of the eight campaigns conducted during this project and discuss which is best suited for developing a clustering workflow for seismic wind turbine (WT) emissions.

2.1 Project Inter-Wind

Inter-Wind was an interdisciplinary project between the Department of Psychology of the MSH Medical School Hamburg, the Stuttgart Wind Energy (SWE) chair of the University of Stuttgart, the Center for Solar Energy and Hydrogen Research Baden-Württemberg (ZSW), the Institute of Psychology of the Martin-Luther-University Halle-Wittenberg, and the Geophysical Institute (GPI) of the Karlsruhe Institute of Technology (Gaßner et al., 2022). It was supported by the Federal Ministry for Economic Affairs and Climate Action based on a decision by the German Bundestag (grants 03EE2023A-D). In addition to psychological surveys evaluating residents' complaints about WTs, meteorological data, acoustic and ground motion emissions from WTs were measured at and in the vicinity of two wind farms (WFs) in the eastern Swabian Alb, southwest Germany (Fig. 1.1). All publications within Inter-Wind and the corresponding measurement campaigns are summarized in Table 2.1. Between 2020 and 2023, a total of eight ground motion measurement campaigns at two wind farms, Lauterstein comprising 16 WTs, and Tegelberg, comprising three WTs, were conducted. The focus of the campaigns was on measuring the amplitude decay of the seismic WT emissions. The sensors were placed at different distances from one of the WTs, starting with a reference station inside the tower of the WT on the foundation (Gaßner et al., 2022, 2023; Gaßner and Ritter, 2023a,b). In the vicinity of WF Lauterstein, measurements were mainly performed in the surrounding forest. In contrast, the measurement sites during the campaigns at WF Tegelberg were more distributed, e.g., within residential buildings in the nearby municipality, next to a public swimming pool, or a federal road. Furthermore, the topography near WF Tegelberg is more complex (Fig. 1.1) and in addition, the closest station to the reference WT at Tegelberg was less than 100 m, while at WF Lauterstein the first station outside of the WT was set up in greater distances. Therefore, this work will focus on the data recorded during the Tegelberg campaigns, named IW02, IW05, IW07, and IW08 (Fig. 2.1).

Table 2.1: Publications regarding the ground motion measurements within the project Inter-Wind. In addition, the studied measurement campaigns (IW0x) and wind farms (WFs) are listed.

Publication	Campaign	WF
Gaßner et al. (2022)	IW02	Tegelberg
Gaßner and Ritter (2023a)	IW05	Tegelberg
	IW06	Lauterstein
Gaßner and Ritter (2023b)	IW02	Tegelberg
	IW03	Lauterstein
Blumendeller et al. (2023)	IW07	Tegelberg
Gaßner et al. (2023)	IW02	Tegelberg
	IW05	Tegelberg
	IW06	Lauterstein

2.2 Measurement campaigns

This section presents the measurement campaigns IW02, IW05, IW07, and IW08, conducted within Inter-Wind at WF Tegelberg. The associated WTs, from north to south, are referred to as *WT 1*, *WT 2*, and *WT 3* (Fig. 2.1). Table 2.2 presents an overview of the field experiments with their sensors and sampling rate. In addition to the ground motion measurements, acoustic measurements were conducted simultaneously at some sites. Since these measurements are not part of this work, I will not discuss them further. The ground motion measurements were conducted with three component sensors, vertical (Z), north-south (N), and east-west (E), orientated with an accuracy of ±0.3° with a gyrocompass. Each campaign included one station on the WTs' fundament at the outer edge of the basement to measure the source signal and with it the operating noise (Gaßner et al., 2022). The other stations, besides the ones at residential homes, are buried approximately 30 cm within the Earth. In case no electricity was available, the stations were operated with batteries. The sites were mainly placed towards the west and southwest of WF Tegelberg (Fig. 2.1). The stations of IW02 were placed towards the southwest from *WT 1* onward up to a distance of 1.8 km (Fig. 2.1b). IW05 combined a ring measurement around *WT 1* and an approximately 2.5 km long profile line towards the north (Fig. 2.1a). The set-up of IW07 was similar to IW02, but utilized *WT 2* as its origin (Fig. 2.1c). Furthermore, the furthest station, IW07Y, is positioned towards the west of *WT 2* next to the federal road B10. IW08 uses *WT 3* as its origin, and the stations are mainly distributed towards the west. Nevertheless, two stations, IW08H and IW08G, are set up towards the southwest at the public swimming pool of the municipality of Kuchen. Each campaign included measurements in residents' homes or a building. Within this work, I analyzed the recorded data from the station closest to the WT, as these recorded the strongest amplitudes of WT ground motion emissions and are not dominated by operational noise like the stations within the WTs.

2.2.1 Measurement campaign IW02

During the measurement campaign IW02 the closest station to the WT, IW02B, was deployed in a distance of 154 m to *WT 1* and 194 m to *WT 2* (Fig. 2.1 and Table 2.2). Three stations were set up in the southwest in the nearby forest and for two weeks in four houses in the municipality of Kuchen. This campaign is therefore, interesting for investigating WT emissions in buildings (Gaßner et al., 2022; Gaßner and Ritter, 2023a).

However, during quality control of the data, I observed a regular occurrence of local maxima

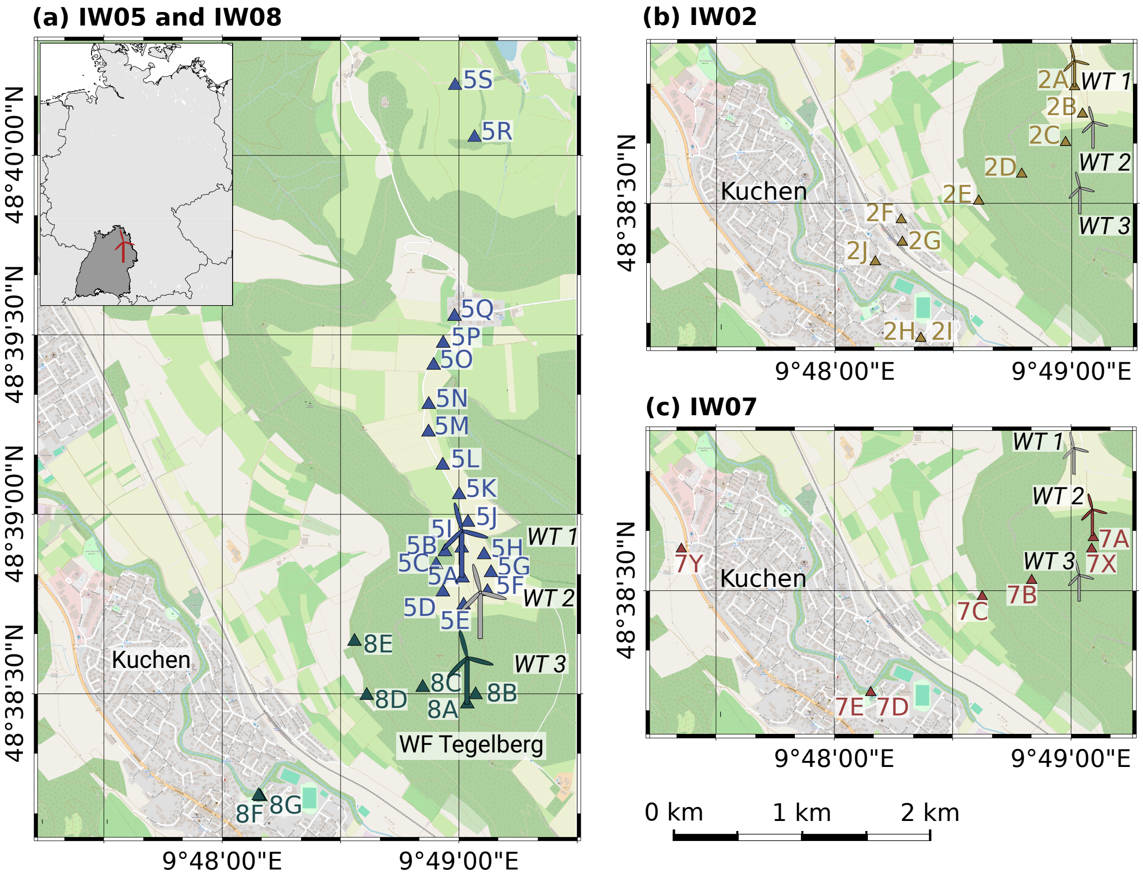


Figure 2.1: Map of the station sites of the analyzed measurement campaigns at the Tegelberg wind farm (WF) Tegelberg on the eastern Swabian Alb, SW Germany. The station sites are marked with triangles and the corresponding station code, IW0**, is written next to them. (a) shows the campaigns IW05 (blue) and IW08 (cyan) and an inset map of Germany in which Baden-Württemberg is highlighted. (b) depicts campaigns IW02 (brown). (c) shows campaign IW07 (red).

Table 2.2: Overview of the measurement campaigns presented in this work. In addition, the sensor type and sampling rate of the analyzed stations are listed. The used stations were equipped with a DATA-CUBE^3 digitizer and powered by a battery. LE-3D: Lennartz LE-3Dlite 1 s, L4-3D: MARK L-4C-3D 1 Hz, TC-P20: Nanometrics Trillium Compact Posthole 20 s, TC120: Nanometrics Trillium Compact 120 s.

ID	Date	Recording time	Station count	Station ID	Sensor type	Sampling rate
IW02	20.10.20 – 05.02.21	108 d	10	IW02B	LE-3D	100 Hz
IW05	19.11.21 – 17.12.21	29 d	9	IW05A – I	L4-3D	100 Hz
	25.11.21 – 17.12.21	23 d	10	IW05J – S	TC120	100 Hz
IW07	23.03.22 – 12.05.22	50 d	5	IW07B	TC-P20	400 Hz
	13.04.22 – 12.05.22	29 d	7	IW07X	TC-P20	400 Hz
IW08	19.10.22 – 21.02.23	125 d	7	IW08B	TC-P20	200 Hz

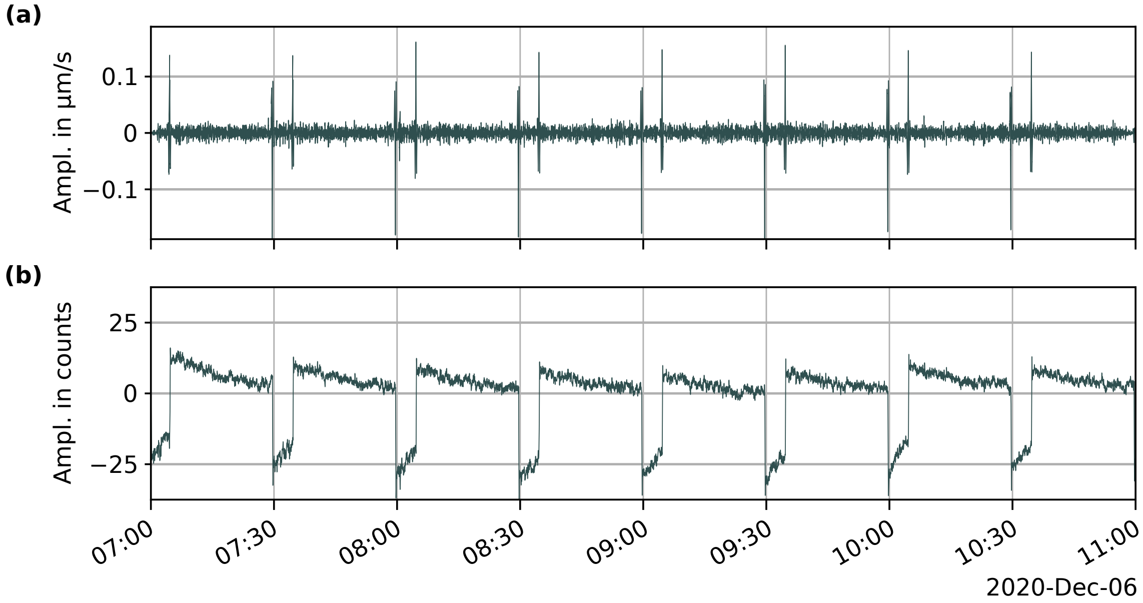


Figure 2.2: Ground motion recorded at station IW02B on 06.12.2020 between 7 a.m. and 11 a.m. (a) Ground motion velocity obtained by applying the ObsPy function `remove_response()` with a bandpass filter using the `pre_filt` option set to 0.01 Hz, 0.05 Hz, 45 Hz and 50 Hz and the default water level for the deconvolution of 60 dB. (b) 0.1 Hz low-pass filtered signal before removal of the instrument response and after application of a linear detrend and subtraction of the average signal value. The signal is equivalent to a boxcar function with a period of 5 min and a repetition period of 30 min, corresponding to GPS cycling (see text).

in the ground motion velocity data (Fig. 2.2a). Applying a 0.1 Hz low-pass filter to the data, unveiled a continuous low frequent boxcar signal with a period of 5 min and a repetition period of 30 min (Fig. 2.2b). This corresponds to the used GPS configuration, every 30 min the GPS searched for satellite signals for a period of 5 min.

Investigating this problem further showed that all Inter-Wind stations using the combination Lennartz LD-3D 1 s seismometer and DATA-CUBE[3] had the same problem. The peaks within the ground motion velocity signal can be visually removed by adjusting the options of the applied ObsPy function `remove_response()`. The corner frequencies of the bandpass filter were changed from 0.01 Hz, 0.05 Hz, 45 Hz and 50 Hz to 0.1 Hz, 0.5 Hz, 45 Hz and 50 Hz and the water level of the deconvolution, which is 60 dB in the default setting, was omitted. However, the Fourier transformation of a boxcar function is a sine cardinalis that extends its frequency content up to the Nyquist frequency

$$f_{\mathrm{Ny}} = 0.5 \cdot f_s, \tag{2.1}$$

with f_s the sampling rate. Consequently, this GPS noise affects the entire frequency band, rendering simple filtering ineffective for its elimination. Given these limitations, the data obtained at these stations was assessed as unsuitable for the automatic analysis of continuous data, and another measurement campaign was selected.

2.2.2 Measurement campaign IW05

Throughout the measurement campaign IW05, a ring and line measurement were conducted (Fig. 2.1 and Table 2.2). The ring measurement included eight stations, IW05B to IW05I, deployed with a distance of approximately 150 m to *WT 1* and with an inter-station azimuthal gap of 45°, starting with IW05B in the Northwest of the WT (Fig. 2.1a).

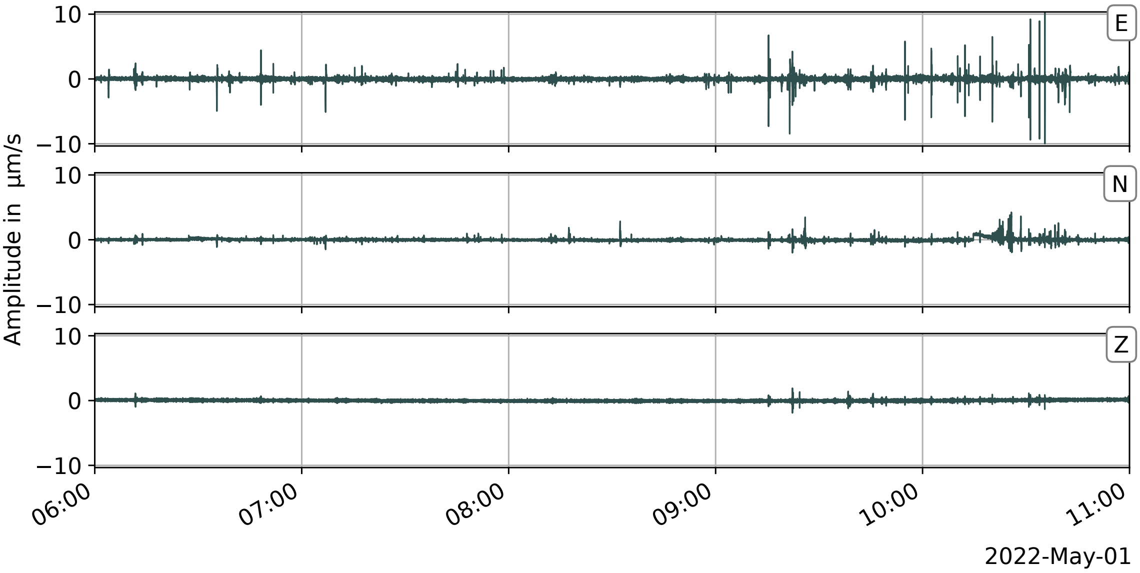


Figure 2.3: Ground motion velocity of station IW07X on May 01, 2022, between 6:00 and 11:00 a.m. The maximum absolute amplitude of the E-component is more than seven times larger than the other components.

This allowed an analysis of the direction-dependent radiation characteristics. The line measurement included ten instruments set up to the north of *WT 1* up to a distance of approximately 2.5 km. The measurement campaign IW05 is described in detail in the Geophysical Instrument Pool Potsdam (GIPP) report Gaßner et al. (2023). Due to the minimal height variations compared to the other campaigns, the line measurement is ideally suited to investigate the influence of the amplitude decay of the WT signal on the investigated methodology.

2.2.3 Measurement campaign IW07

The measurement campaign IW07 combines a variety of different measurement sites. Stations were not only set up in the vicinity of WF Tegelberg but also in the municipality of Kuchen or next to the federal road B10 (Fig. 2.1c). Hence, IW07 seemed to be the campaign to study the influence of the WTs emissions and other noise sources on the applied methodology. Station IW07X is set up in only 71 m distance to *WT 2* (Fig. 2.1c). This is the closest location to a WT installed during Inter-Wind up to this campaign up to this experiment, it seemed promising for an initial test of the method developed by Steinmann et al. (2022a,b). However, Figure 2.3 shows that the maximum absolute amplitude of the background noise of the E-component is more than seven times greater than that of the other components. This was measured during a period in which the WF was not in operation. The recorded data is classified as unreliable as the underlying problem could not be solved.

2.2.4 Measurement campaign IW08

The last major measurement campaign carried out at the WF Tegelberg was IW08 (Fig. 2.1 and Table 2.2). With 125 days it was the longest conducted experiment within the Inter-Wind project. In addition, different noise-reduced modes of the WT were tested, beginning on November 21, 2022 (Figs. 2.5 and 2.6a). Noise-reduced modes are modes in which a WT is not operated at maximum speed and consequently generates less noise. With *WT 3* being the center WT of this campaign, three of the seven stations were set up within the forest, one station was deployed on a field, and two next to a closed public outdoor swimming pool (Fig. 2.1a). The closest station to *WT 3*, IW08B, was deployed in 66 m distance and will be used for further analysis.

2.2.5 Meteorological and wind turbine data

To correlate the ground motion data with the operating phases of the WT and the meteorological conditions such as wind speed and direction, the operators of the WF Tegelberg provided the WT operational data recorded at the top of the WTs at hub height (Fig. 2.4). The operational data include the rotation rate of the WT in rotation per minute (rpm), wind speed, and nacelle position, among others, averaged to 10 min time windows. As the nacelle of the WTs aligns with the wind direction, I will refer to the nacelle position as wind direction throughout this work (Fig. 2.4). The WTs from WF Tegelberg are of the type General Electric (GE) 2.75-120 and have a hub height of 139 m, a rotor diameter of 120 m, and a power rate of 2.78 MW.

At full load, this WTs have a rotation rate of 12.5 rpm. Figure 2.6a shows the rotation rate as a function of the wind speed. At wind speeds between $3\,\mathrm{m\,s^{-1}}$ and $8\,\mathrm{m\,s^{-1}}$ the WT is operated at partial-load with 7.9 rpm and at wind speeds above $6\,\mathrm{m\,s^{-1}}$ in full load. If the WT runs in noise-reduced modes, the rotation rate deviates from 12.5 rpm at full load. The WT is switched off at the so-called cut-off wind speed, for this type above about $18\,\mathrm{m\,s^{-1}}$ (Fig. 2.6a).

IW08

Figures 2.5 and 2.6 show the rotation rate and an overview of the wind speed and direction recorded during the measurement campaign IW08. Detailed wind speed and direction representations can be found in the appendix (Figs. A.1 and A.2). Above, I describe that the WTs, primarily *WT 3*, were operated in various noise-reduced modes during IW08. This means that in addition to the partial load, which was not changed and remained at 7.9 rpm, the WTs ran at full load during the day at 12.5 rpm and at night at 12 rpm until November 21, 2022. From then on until December 5, 2022, the full load was reduced to 11 rpm and then changed to 12 rpm until January 16, 2023 (Figs. 2.5 and 2.6a). The rotational rate subsequently varied at full load. During the measurement period, the wind blew predominantly from the west and rarely from the north (Fig. 2.6b).

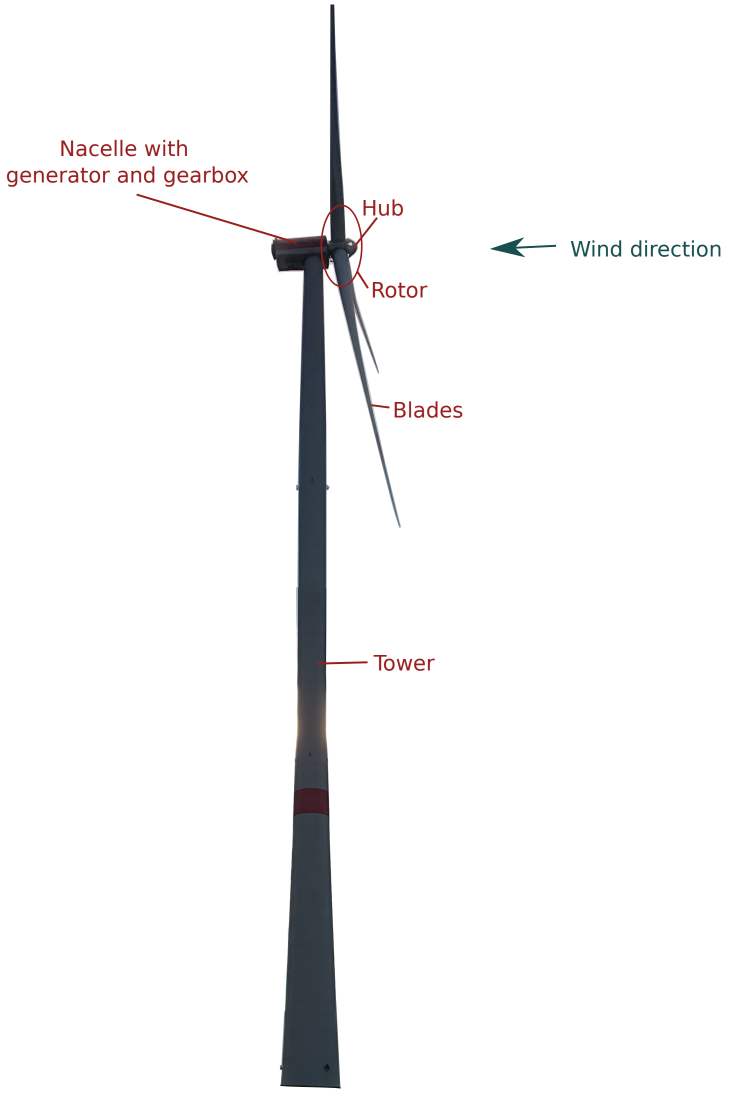


Figure 2.4: Northernmost WT of the WF Tegelberg, photographed by Laura Gaßner. The tower of an onshore WT is supported by a foundation, which is not shown in this figure. At the top of the tower is the nacelle, which is aligned in wind direction. Inside the nacelle is, amongst others, the generator, the gearbox, and the control system. Not highlighted in this illustration are the wind vane and the anemometer, which measure the wind speed and direction. The WT blades are attached to the front of the nacelle and face into the wind. They are held in position by the hub. The blades and the hub form the rotor.

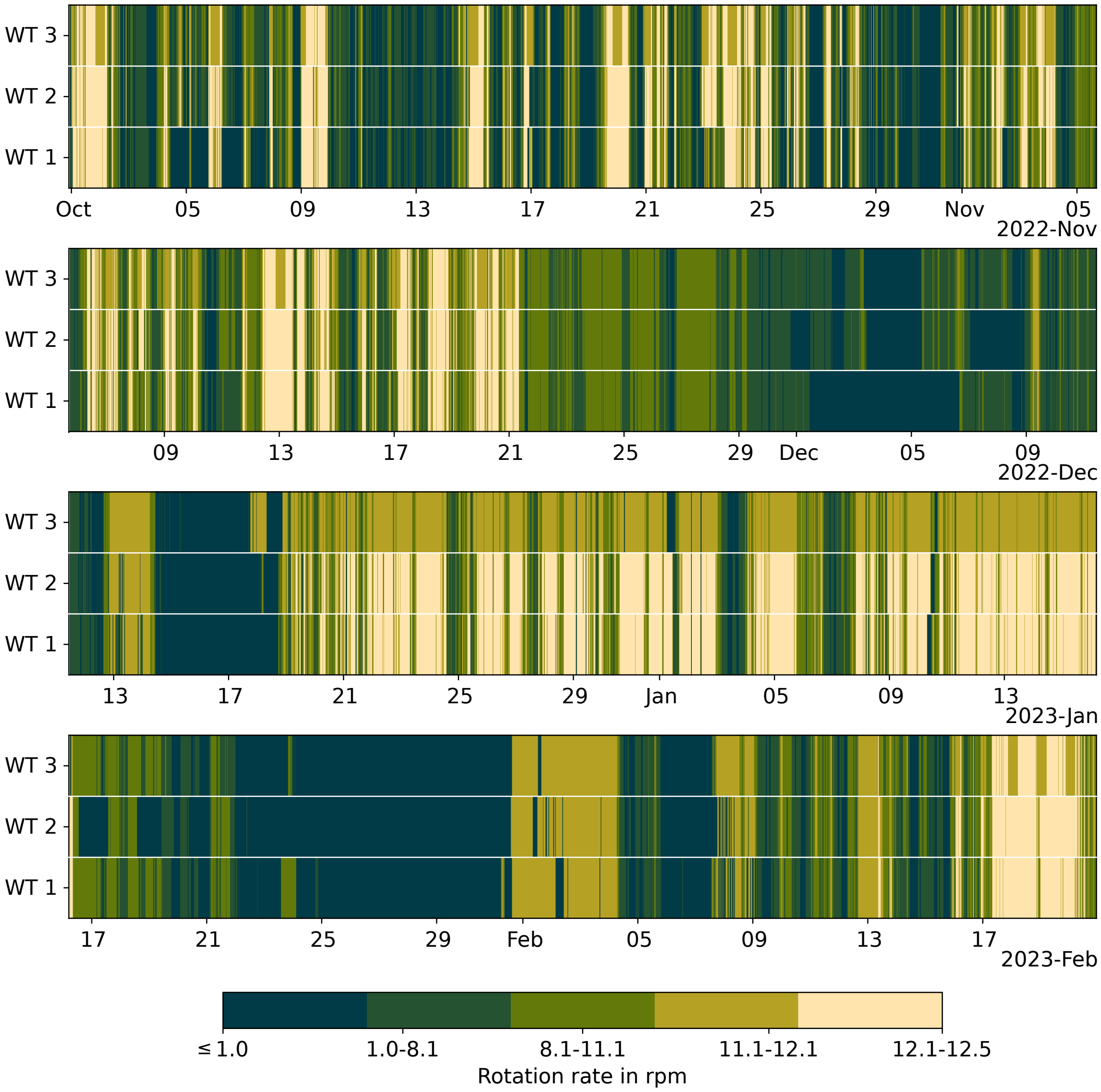


Figure 2.5: Rotation rate recorded at WF Tegelberg for wind turbines (WTs) *1* to *3* during the measurement campaign IW08 between November 1, 2022 and February 20, 2023. The color scale is divided into five parts, including the operating stage and the launch of the corresponding mode. ≤ 1 rpm – WT is not in operation or is just launched, 1 rpm to 8.1 rpm WT operates in partial-load (7.9 rpm), 8.1 rpm to 11.1 rpm WT operates in full-load with lower noise reduced setting (11 rpm), 11.1 rpm to 12.1 rpm WT operates in full-load with noise reduced setting (12 rpm), 12.1 rpm to 12.5 rpm WT operates in full-load (12.5 rpm).

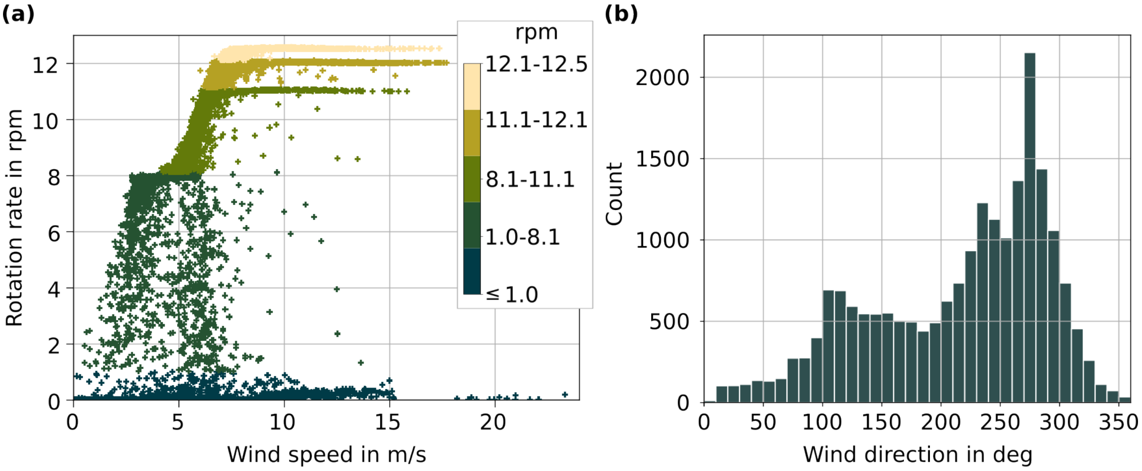


Figure 2.6: (a) Rotation rate as a function of the wind speed for *WT 3* (wind farm Tegelberg). The colors mark the different modes of *WT 3* from November 1, 2022 until February 20, 2023. The brighter the color, the higher the rotation rate. (b) Histogram over the wind direction during IW08.

2.3 Data processing

The recorded ground motion data was processed in Python using the ObsPy package (Beyreuther et al., 2010; Krischer et al., 2015; Megies et al., 2011). First, I removed any linear trend and subtracted the average of the data, to move the origin to zero, by applying the functions `detrend()` with the options `"linear"` and `"constant"`. Next, I applied a simulated instrument response and transformed the data to ground motion velocity using the `groundmotion()` function developed by Thomas Forbriger, setting the simulated eigenfrequency f_0 to 0.05 Hz (Appendix B).

With `groundmotion()`, the instrument response is simulated by the application of a recursive filter in the time domain (T. Forbriger, personal communication, November 20, 2023 and May 07, 2024). With this, the two lowest poles are shifted to f_0.

In contrast, the function provided by ObsPy to remove the instrument response, `response_remove`, feeds all poles and zeros into a numerical filter and with it deconvolves the instrument response. However, to flatten the instrument response, the amplitudes at the low- and high-pass flanks of the instrument response need to be increased. This factor increases as the flanks decrease and to stabilize possible oversimplifications a water level can be applied or the data can be tapered via the option `pre_filt` (The ObsPy Development Team, 2024). Problems appear, for instance, in case poles and zeros cancel each other out. In addition, the list contains poles and zeros that are irrelevant to the data but destabilize the filter. Furthermore, the application of the water level deconvolution can lead to an acausal response of the signal as phase information is lost.

Figure 2.7 illustrates this problem. The Z-component of the raw data contains a step just after 00:04:20 (Fig. 2.7a). This step leads to an acausal behavior of the signal after applying ObsPy's `response_remove` (Fig. 2.7a). The issue does not occur if the function `groundmotion()` developed by Thomas Forbriger is applied.

After removing the instrument response, a low-pass filter with 45 Hz was applied to minimize the data size and additionally the data was resampled to 64 Hz. Finally, the function `merge(method=1)` was utilized to remove possible overlaps and to fill gaps with `None`.

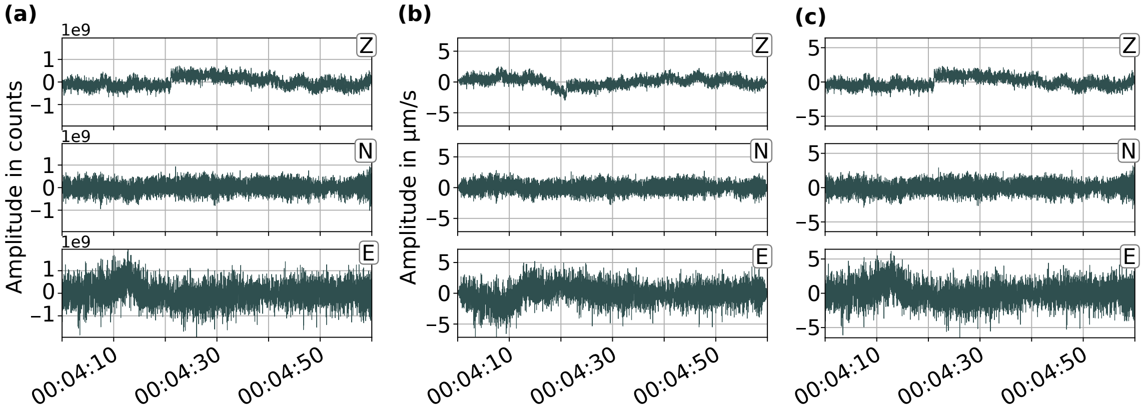


Figure 2.7: Ground motion recorded on February 3, 2023 between 00:04:00 and 00:05:00 at station IB08B. (a) Raw data in counts with average value removed. (b) Ground motion velocity. Instrument response removed with ObsPy's `response_remove`. No water level deconvolution and a prefilter with corner frequencies 0.1 Hz, 0.5 Hz, 45 Hz and 50 Hz was applied. (c) Ground motion velocity. Instrument response simulated with `groundmotion()` developed by Thomas Forbriger.

2.4 Seismological data

2.4.1 Wind turbine signals

WTs radiate seismic waves via the mechanical coupling of their foundation to the subsurface (Zieger and Ritter, 2018). The rotation of the blades, rotor, hub, and other components within the nacelle (Fig. 2.4), cause these emissions. But even when the WT is not in operation, the entire tower-nacelle system emits vibrations related to its eigenmodes (Nagel et al., 2019; Zieger, 2019). The frequencies emitted due to the rotation rate of the WT are the so-called BPF and its multiples. The blade passing frequency (BPF) depends on the number of rotor blades n, usually three, and is defined as n times the rotation rate. The WT's eigenfrequencies are related to the tower-bending modes and torsional oscillations of the tower-nacelle system. They are typically visible between 0.1 Hz and 20 Hz (Nagel et al., 2021). The highest eigenmode observed of this WT type at WF Tegelberg is 11.25 Hz (Table 2.3; Gaßner and Ritter, 2023b). The eigenmodes of the Tegelberg WTs are marked with dark red lines and labeled with Roman numbers in the spectrogram shown in Figure 2.8b. These mono-frequent signals appear as straight lines in the spectrogram, while the BPF and its multiples are gliding frequencies, which change with changing rotation speeds of the WT. The BPFs of the major operational modes of the Tegelberg WT are listed in Table 2.3. Figure 2.8 shows a change in rotation rate from 12 rpm to 7.9 rpm. The multiples of the BPF are marked in light red in Figure 2.8b.

Table 2.3: (a) Eigenmodes of the WTs tower. (b) Rotation rates in partial-mode (8 rpm) and full-load ($\gg 8$ rpm) and corresponding blade passing frequency, BPF = $\frac{3\times\text{rpm}}{60}$.

(a)

Eigen modes	Frequency in Hz
I	1.20
II	3.60
III	8.33
IV	11.25

(b)

Rotation rate in rotation per minute (rpm)	BPF in Hz
7.9	0.395
11	0.550
12	0.600
12.5	0.625

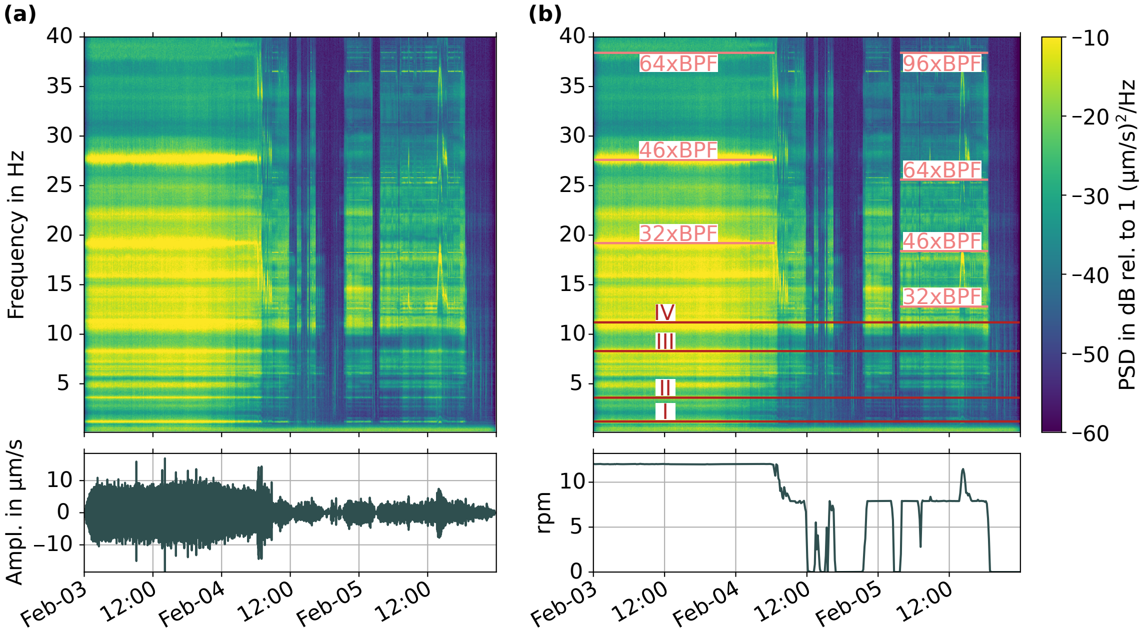


Figure 2.8: (a) Vertical ground motion velocity and its spectrogram recorded at station IW08B between February 3 and 5, 2023. (b) Rotation rate of the southernmost WT of WF Tegelberg and (c) identical spectrogram as (a), but with marked eigen frequencies of the WTs (dark red, roman numbers) and the gliding frequencies corresponding to the changing multiples of the BPF (light red). For 12 rpm the BPF is 0.6 Hz and for 7.9 rpm 0.395 Hz.

2.4.2 Other possible contributions

Station IW08B is located in a forest and therefore records not only WT emissions but also noise from trees, such as vibrating trees and falling branches, and other noise associated with the forest. Anthropogenic contributions are mainly limited to forest workers, as no residents live nearby and no roads are in the vicinity. Several teleseismic earthquakes occurred during the recording time, the largest of which was the M 7.8 Pazarcik earthquake in Turkey on February 6, 2023 at 01:17:34 (Fig. 2.9a; USGS National Earthquake Information Center, PDE, 2023). As Tegelberg is part of the Swabian Alb, one of the most seismically active regions in Germany, IW08B registered local quakes as well. For example, on January 27, 2023 at 20:46:16 with a magnitude of M_L 2.0 near Albstadt (Fig. 2.9a; *Erdbeben bei Albstadt, Zollernalbkreis, BW* 2023).

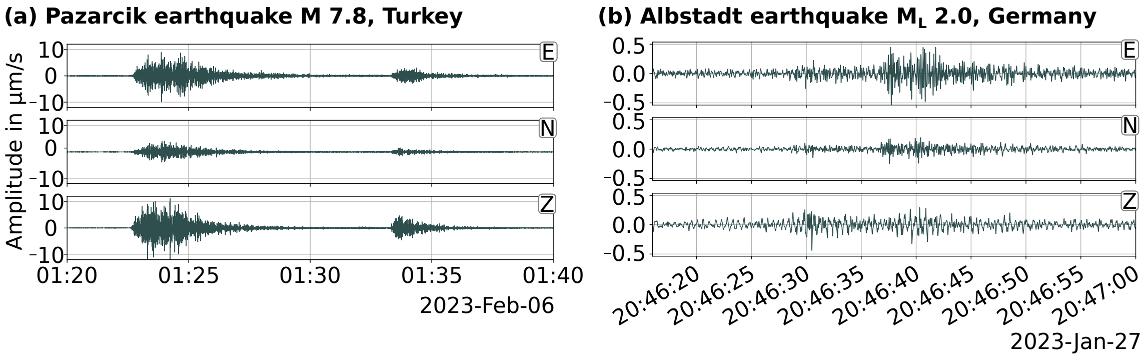


Figure 2.9: Earthquakes recorded at IW08B. Bandpass-filtered between 1 Hz and 10 Hz. (a) Teleseismic earthquake. Pazarcik earthquake with M 7.8 in Turkey on February 6, 2023 at 01:17:34. According to USGS National Earthquake Information Center, PDE (2023), the hypocenter was located at 37.23°S 37.01°E at a depth of 10 km. (b) Local earthquake M_L 2.0 on January 27, 2023 at 20:46:16. The hypocenter was located near Albstadt, 48.3°S 9.03°E at a depth of approximately 7 km according to *Erdbeben bei Albstadt, Zollernalbkreis, BW* (2023).

Chapter 3

Scattering network

This work aims to find known and hidden seismic WT emission patterns. Finding these patterns and classifying them according to their similarity requires extracting the most relevant information from our data. In other words, we want to reduce the variability of our data without losing the information that helps us to classify it (Mallat, 2010). To do so, we can, for instance, divide the data into time windows and group the time windows with similar waveforms into the same class. Similar to other object recognition tasks, such as identifying handwritten digits (Amit and Trouvé, 2007), translation or small deformations can alter the underlying pattern in seismic data.

Which implications does this have for the task at hand? Imagine two time windows with similar waveforms, one is slightly shifted and deformed compared to the other. However, to find a representation in which both time windows are still recognized as similar, it must be transformation invariant and stable against small deformations (Mallat, 2010). Examples of translation invariant representations are the Fourier transform and the autocorrelation, both of which are unstable against small deformations. While the Fourier transformation represents the signal by a linear combination of sinusoids of different frequencies and is only localized in frequency, the wavelet transformation covers the wanted frequency band, scale, by the multiple dilatation of a wavelet. Unlike continuous sinusoids, wavelets are localized in both time and frequency, making them stable towards small deformation (Bruna and Mallat, 2013). Nevertheless, they are covariant to translation. The application of a nonlinear operator makes the wavelet transform invariant to translation. The associated loss of information can be compensated for by repeated application of the wavelet transform and a non-linear operator (Mallat, 2010). This type of convolutional network is referred to as a scattering network (Bruna and Mallat, 2013; Mallat, 2012).

With this in mind, I will discuss the theory of the scattering network and the optimization of its architecture for seismic WT emissions in more detail, starting with the wavelet transform theory.

3.1 Wavelet transformation

A wavelet transformation is performed by dilating a mother wavelet $\psi(t)$ multiple times by a scaling factor λ to cover the desired scale or frequency range. Using the nomenclature of Anden and Mallat (2014), λ is defined as

$$\lambda = 2^{\frac{k}{Q}}, \quad \text{with } k \in \{0, 1, ..., J \cdot Q - 1\}, \tag{3.1}$$

which includes the number of octaves $J \in \mathbb{Z}$ the wavelets cover and the wavelets per octave $Q \in \mathbb{Z}$. Applying the scaling factors to the wavelet$\psi(t)$ results in to multiple wavelets referred to as a filter bank

$$\psi_\lambda(t) = \lambda \cdot \psi(\lambda t), \quad \lambda > 0. \tag{3.2}$$

Each wavelet is centered in time at $t = 0$ and frequency at different scales, or center frequencies f_{ctr}. They act as bandpass filters whose bandwidth is proportional to f_{ctr} and are also known as constant-Q filters (Brown, 1991; Mallat, 2009). Starting from the Nyquist frequency f_{Ny} (Eq. 2.1), the smallest wavelet in time and the largest bandwidth in frequency, the wavelets are dilated with each lower frequency in time. The wavelets cover, therefore, a frequency range between f_{Ny} and $2^{-J} \cdot f_{\text{Ny}}$ (Fig. 3.1a).

The wavelet transform is a convolution of the filter bank $\psi_\lambda(t)$ with the signal $x(t)$

$$W_\lambda(t) = x(t) * \psi_\lambda(t), \tag{3.3}$$

where $*$ represents the convolution operator. The wavelet coefficients W form a scalogram, depicting time on the x-axis and scale or the center frequencies f_{ctr} of the wavelets on the y-axis. The wavelets are of unit energy and therefore, have equal area beneath the filters, high frequent wavelets are broader in the frequency domain but have a smaller amplitude than lower frequent wavelets. Consequently, the amplitude of the scalogram is attenuated in higher frequencies (Fig. 3.1a).

There are numerous types of wavelets suitable for different applications. In accordance with Anden and Mallat (2014) and Bruna and Mallat (2013), this work utilizes the Morlet wavelet

$$\psi(t) = \exp\left(-i2\pi f_{\text{ctr}} t\right) \exp\left(-\frac{t^2}{a^2}\right), \quad a = \frac{d}{f_{\text{ctr}}} = \frac{d}{\lambda f_{\text{Ny}}}, \tag{3.4}$$

with a describing the exponential drop-off of a Gaussian window, the center frequency f_{ctr}, and the bandwidth d.

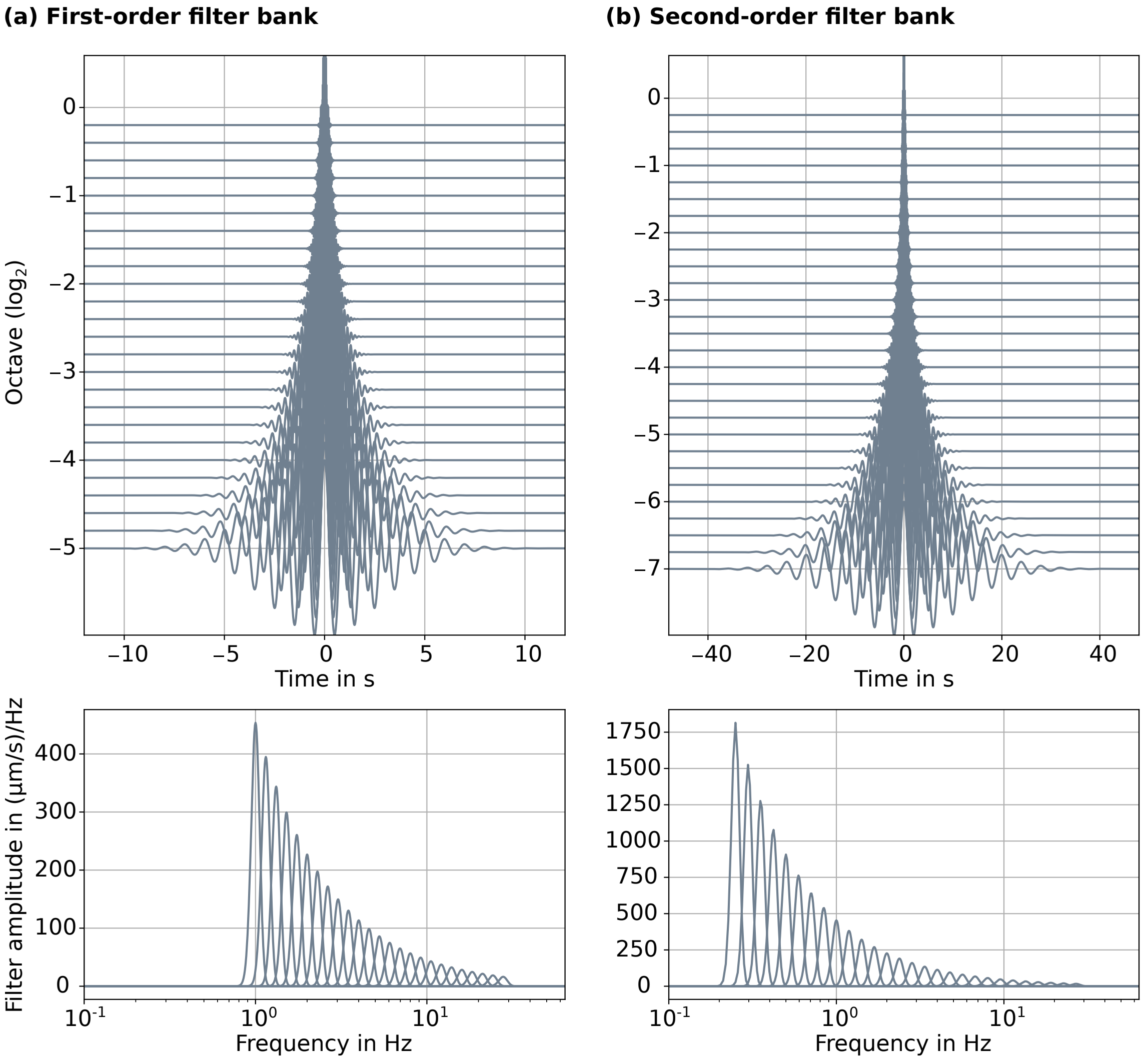


Figure 3.1: Filter banks of Morlet wavelets (top) and their frequency responses (bottom). (a) The first-order filter bank covers five octaves with a resolution of five wavelets per octave. (b) The second-order filter bank comprises seven octaves, including four wavelets per octave. Both filter banks have a quality factor of four and represent the initial filter bank used to test the scattering network.

3.2 Scattering transform and its application to seismic data

The scattering transform was introduced by Mallat (2010) and Mallat (2012) and first applied to image and audio processing by Bruna and Mallat (2013) and Anden and Mallat (2014). The initial step of the scattering transform is the calculation of the modulus of the wavelet transform. A non-linear pooling operator, $\phi(t)$, is then applied to ensure translation invariance of the result,

$$Sx(t,\lambda) = |x(t) * \psi_\lambda(t)| * \phi(t) = |W_\lambda(t)| * \phi(t) = U_\lambda(t) * \phi(t). \tag{3.5}$$

As such, the resulting scattering coefficients S are translation invariant and stable against small deformations but lose information due to pooling. By applying a second set of filter banks $\psi_{\lambda_2}(t)$ to the modulus wavelet coefficient of Equation 3.5, we reduce the loss of information due to pooling in the first layer. With this each scale, λ_1, of the first layer is further subdivided into the scales, λ_2, of the second layer,

$$S_2x(t,\lambda_1,\lambda_2) = |U_1 * \psi_{\lambda_2}(t)| * \phi(t) = ||x(t) * \psi_{\lambda_1}(t)| * \psi_{\lambda_2}(t)| * \phi(t), \tag{3.6}$$

with U_1 the modulus wavelet coefficients and $\psi_{\lambda_1}(t)$ the filter bank of the first layer. Hence, the scattering network resembles a convolutional network with a wavelet transform as filters and an output at each layer. Further iterations,

$$S_mx(t,\lambda_1,\ldots,\lambda_m) = ||x(t) * \psi_{\lambda_1}(t)| * \ldots | * \psi_{\lambda_m}(t)| * \phi(t), \tag{3.7}$$

further minimize the loss of information (Anden and Mallat, 2014; Bruna and Mallat, 2013). Nevertheless, Anden and Mallat (2014) and Bruna and Mallat (2013) showed that two layers are sufficient, as additional layers hardly contain energy.

Seydoux et al. (2020) first applied the scattering transformation to seismological data to cluster earthquake signals. Further seismological applications performed in Steinmann et al. (2022a,b) and Steinmann et al. (2023) form the base for the workflow of this thesis (Fig. 3.2). Further applications in seismology were conducted by Barkaoui et al. (2021), Morel et al. (2023), and Rodríguez et al. (2022). To perform the scattering transformation I utilized the python package scatseisnet developed by Seydoux and Steinmann (2023).

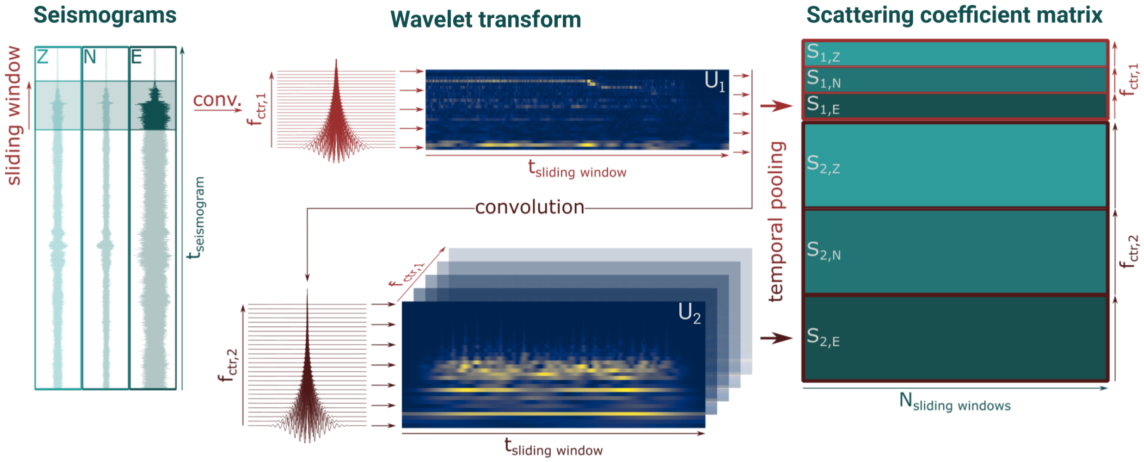


Figure 3.2: Application of a two-layer scattering network to continuous three-component seismograms. The sliding window is convolved with the first filter bank with center frequencies $f_{\text{crt},1}$. After calculating the modulus, the modulus wavelet coefficients U_1 are convolved with the second filter bank with center frequencies $f_{\text{crt},2}$. Subsequently, temporal pooling is applied to the modulus wavelet coefficients U_1 and U_2 to extract the first- and second-order scattering coefficients S_1 and S_2 respectively. These steps are repeated for all components and N moving windows and the resulting scattering coefficients are concatenated to obtain the displayed scattering coefficient matrix.

3.3 Design of the scattering network

This section discusses the parameter setup for the scattering network using a set of three distinct time windows from the recorded data obtained from station IW08B (Fig. 2.1 and Section 2.2.4) and in addition, four synthetic signals generated by superimposing sine signals (Table 3.1 and Fig. 3.3). The recorded signals, labeled $x_{\text{WT},(1,2,3)}(t)$, include a time window characterized by a constant rotation rate of 12 rpm, a period where the rotation rate transitions from 12 rpm to 8.5 rpm and finally to 0 rpm, and a phase where the WT is turned off, respectively. Correspondingly, three of the synthetic signals denoted as $x_{\text{syn},(1,2,3)}(t)$, represent these different operational states of the WT. The amplitudes of the superimposed sine functions of the synthetic signals are consistent with the root mean square (RMS) amplitudes of the eigenmodes and the 32nd multiple of the BPF present in the recorded signals $x_{\text{WT},(1,2,3)}(t)$ (Table 3.1). The 32nd multiple of the BPF is the first multiple observable in the data (Fig. 2.8)). Additionally, I created superimposed sine signals, $x_{\text{syn},0}(t)$, whose frequencies correlate with center frequencies $f_{\text{crt},1}$ of the first filter bank of the scattering network (Fig. 3.1a), to compare the amplitudes before and after the scattering transform.

3.3.1 Initial filter bank setup

As outlined above, each wavelet within the filter bank is characterized by a center frequency f_{ctr}, ranging from the Nyquist frequency f_{Ny} down to $2^{-J} \cdot f_{\text{Ny}}$ (Fig. 3.1). To capture the eigenmodes of the investigated WT, with the lowest frequency identified as 1.2 Hz (Table 2.3), and taking into account the 32nd multiple of BPF at maximum rotation rate the wavelets must cover a frequency range from 1.2 Hz to 20 Hz. The next higher power is 32 Hz, following the Nyquist criteria I resampled the data to 64 Hz. I set the number of octaves J covered by the first-order filter bank to $J = 5$. Consequently, the lowest center frequency is 1 Hz covered by the first-order filter bank. The wavelets per octave Q are set to five to maintain a sufficient frequency resolution. Following the methodology of Anden and Mallat (2014), the second-order filter bank is designed to cover a wider frequency range, but with a lower resolution Q. As such, the resolution Q is adjusted to four, and

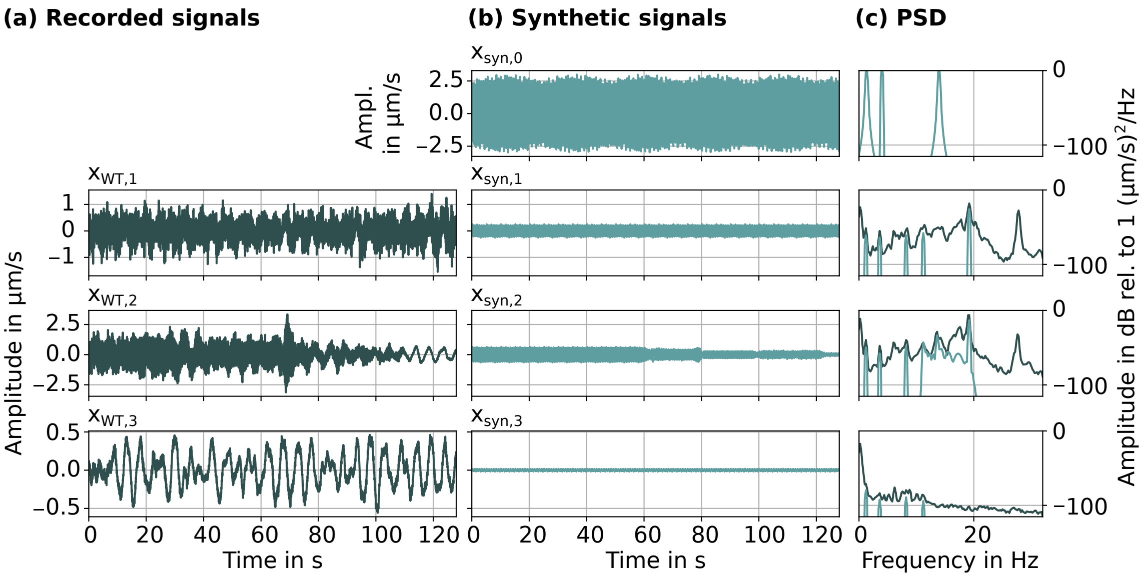


Figure 3.3: Scattering network test signals and their PSD. (a) presents the recorded signals $x_{\mathrm{WT},(1,2,3)}(t)$, while (b) illustrates the synthetic signals $x_{\mathrm{syn},(0,1,2,3)}(t)$, and (c) shows the corresponding PSD. Signal $x_{\mathrm{WT},2}(t)$ depicts a transition from 12 rpm to 8.5 rpm between 64 s and 754 s. During this time the BPF changes from 19.2 Hz to 13.6 Hz. This period of transition is simulated with a quadratic sweep in $x_{\mathrm{syn},2}(t)$. Further details are provided in Table 3.1.

the number of octaves J is set to seven, the maximum value in which the wavelets still fit into the time window. The resulting lowest frequency is 0.125 Hz, which the second-order filter bank covers. Initially, the quality factor q which regulates the number of wavelet oscillations and consequently the width of the frequency range, is set to four. This ensures that a single wavelet covers each frequency peak of interest.

Figure 3.1 illustrates the initial filter banks and Figure 3.4 the corresponding first-order scalogram U_1 and scattering coefficient S_1 of signal $x_{\mathrm{WT},2}$, as well as, the corresponding spectrogram and PSD. While the frequency resolution of the spectrogram is higher, especially at greater frequencies, U_1 shows a better time resolution, as the wavelets cover the entire time window. Signal $x_{\mathrm{WT},2}$ contains a change in the rotation rate and with it the change of the 32nd multiple of BPF from 19.2 Hz to 13.6 Hz. In the spectrogram, this

Table 3.1: Parameters of the scattering network test signals. To determine the amplitude of the sine signals with frequencies f_x that compose the synthetic signals $x_{\mathrm{syn},(1,2,3)}(t)$, the recorded signals $x_{\mathrm{WT},(1,2,3)}(t)$ of station IW08B are bandpass filtered at $f_x \pm 0.25$ Hz. Signal $x_{\mathrm{syn},0}(t)$ is the sum of three sine functions with frequencies f_x corresponding to frequencies of the first filter bank. The signals are each 128 s long.

Signal ID	Rotation rate in rpm	Start time
$x_{\mathrm{WT},1}(t)$	12	18.12.2022 00:00:00
$x_{\mathrm{WT},2}(t)$	12 to 8.5 to 0	18.12.2022 07:37:00
$x_{\mathrm{WT},3}(t)$	0	18.12.2022 09:58:16
	Frequencies f_x in Hz	Phase shift in s
$x_{\mathrm{syn},0}(t)$	1.32, 4.00 and 13.96	–
$x_{\mathrm{syn},1}(t)$	1.2, 3.6, 8.2, 11.2 and 19.2	0.0, 0.15, 0.3, 0.45 and 0.75
$x_{\mathrm{syn},2}(t)$	1.2, 3.6, 8.2, 11.2, 13.6 and 19.2	0.0, 0.15, 0.3, 0.45, 0.6 and 0.75
$x_{\mathrm{syn},3}(t)$	1.2, 3.6, 8.2 and 11.2	0.0, 0.15, 0.3 and 0.45

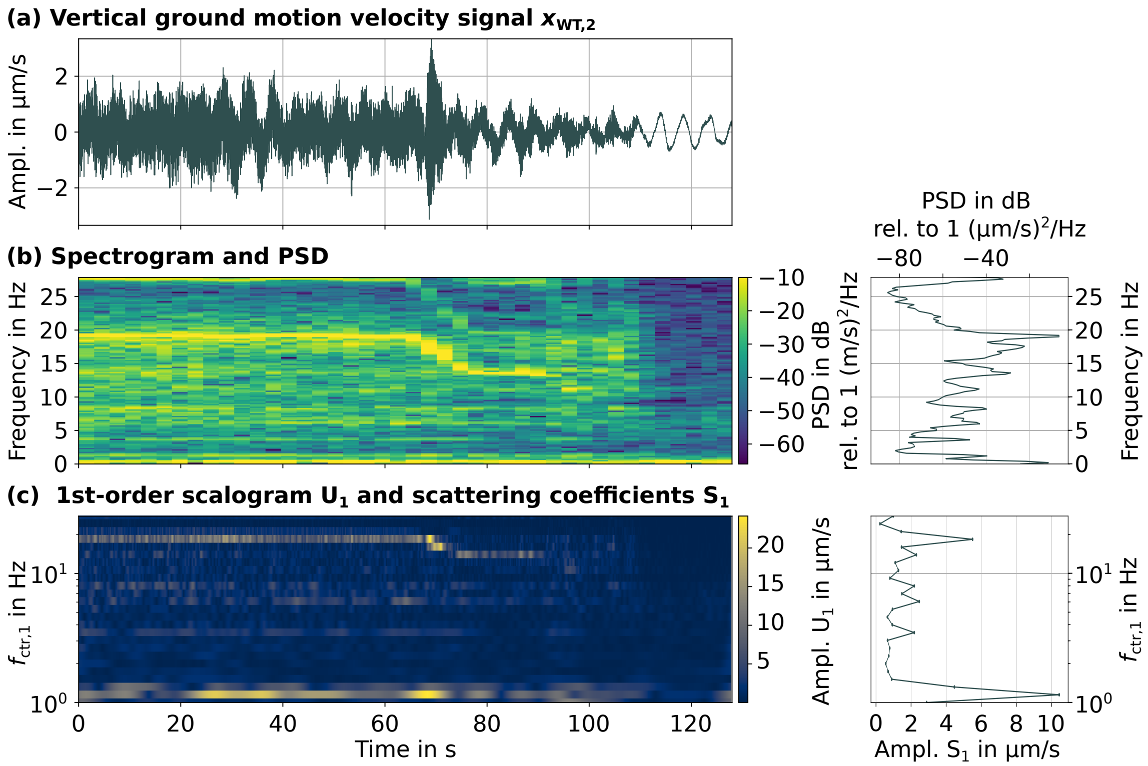


Figure 3.4: (a) depicts the recorded signal $x_{\mathrm{WT},2}$ and (b) its spectrogram on the left side and the power spectral density on the right. (c) shows its first-order scalogram U_1 calculated with the filter bank shown in Figure 3.1a on the left and its first-order scattering coefficients S_1 on the right. (b) is not plotted on a logarithmic scale, otherwise, it would hardly be possible to recognize something. Even if it makes it more difficult to compare the two diagrams, (c) is plotted on a logarithmic scale, since the center frequencies are naturally distributed logarithmically.

transition is blurred, while it is sharp in U_1 (Fig. 3.4). A comparison of the PSD with S_1 shows that the energy in S_1, as explained above, is shifted towards lower frequencies and thus we cannot compare the amplitude of the individual frequencies. Nevertheless, all peaks relating to the WT are visible in S_1. The PSD plot has an additional peak at 17.6 Hz, which is not associated with the WT. This peak is not resolved by U_1 or S_1 and is merged with the 19.2 Hz peak in the center frequency $f_{\mathrm{crt},1} = 18.38\,\mathrm{Hz}$. The second-order scalograms and scattering coefficients are discussed below in more detail (Section 3.3.6).

With this setup, the longest wavelet of the filter banks covers about 80 s (Fig. 3.1), dictating the minimum window size of the input data. Each window should be able to resolve changes in the WT's operation modes. These operational changes last a few seconds to approximately a minute, in case the WT is in full-load and turned off. In the duration of one minute the signal $x_{\mathrm{WT},2}$ changes from a rotation rate of 12 rpm to 8.5 rpm to 0 rpm (Fig. 3.3). Moreover, the convolution between the input data and the filter bank is performed in the frequency domain. To avoid artifacts, the number of samples nt within a window should be $nt = 2^n$, where $n = T_{\mathrm{signal}} \cdot f_s$. Given a sampling rate of $f_s = 64\,\mathrm{Hz}$ the window size is set to $T_{\mathrm{signal}} = 128\,\mathrm{s}$. To ensure that changes in the rotation rate are captured within each time window, the sliding windows overlap by 50 %.

3.3.2 Impact of signal tapering

Filtering a finite time series leads to artifacts at the beginning and end of the time window, which are referred to as edge effect artifacts (e.g., Mallat, 2009; Williams and Amaratunga, 1997). A common method to handle edge-effect artifacts is to taper the signal (Harris,

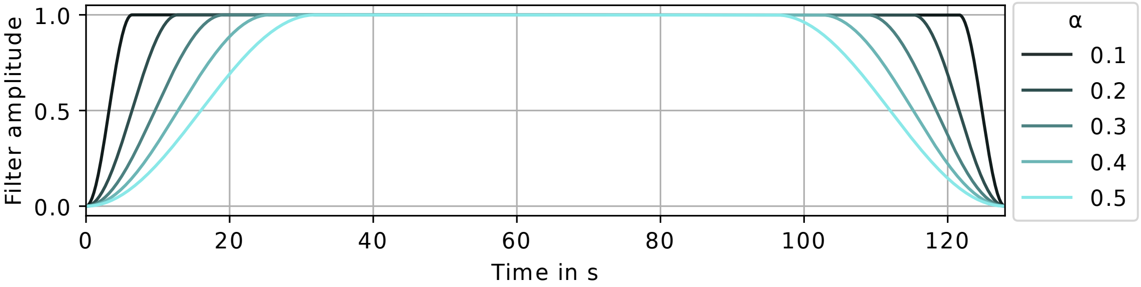


Figure 3.5: Tukey windows used to test the impact of tapering the input signal on the scattering network.

1978). In the following, I discuss the effects of tapering the signal with a Tukey window (Fig. 3.5).

Figure 3.6a depicts the first-order scalogram U_1 of signal $x_{\mathrm{syn},0}$. Since no taper was applied, edge-effect artifacts are noticeable at the boundaries of the shown scalogram, particularly in the lower frequency range. This can be attributed to the fact that low-frequency wavelets have a larger temporal width than high-frequency wavelets, which leads to a longer interference with the edges of the window.

The Tukey window

To mitigate these edge-effects, the input signal of the first filter bank is preprocessed using a Tukey window, also known as a cosine-tapered window. As per Harris (1978), the Tukey window is defined as

$$w(n) = \begin{cases} 1, & 0 \leq |n| \leq \alpha \frac{N}{2} \\ \frac{1}{2}\left[1 + \cos\left(\pi \frac{n - \alpha \frac{N}{2}}{2(1-\alpha)\frac{N}{2}}\right)\right], & \alpha \frac{N}{2} \leq |n| \leq \frac{N}{2} \end{cases} \tag{3.8}$$

$$n = -\frac{N}{2}, \ldots, -1, 0, 1, \ldots, \frac{N}{2}, \tag{3.9}$$

with the number of samples N, and the parameter $\alpha = [0, 1]$, which controls the transition of the Tukey window from a rectangular window, $\alpha = 0$, to a Hanning window, $\alpha = 1$. Applying a taper to a signal leads to the attenuation of the signal at the taper flanks. A compromise must thus be found between the mitigation of the edge effect and the loss of signal. For this purpose, I investigated the effects of the different Tukey windows, shown in Figure 3.5, on the signal $x_{\mathrm{syn},0}$. To limit the signal loss I chose the values of α to be smaller or equal 0.5.

Applying the Tukey window

Figure 3.6 shows the effects on the first-order scalogram S_1 applying the Tukey taper with values of α between 0.1 and 0.5 (Fig. 3.5). However, since the edge-effect artifacts are still significant for the tapers $\alpha = 0.1$ and 0.2, we will only consider the tapers with $\alpha = 0.3$, 0.4, and 0.5 in more detail (Fig. 3.7). For this purpose, let us also have a look at the second-order scalogram. Since the edge-effect is greatest at lower frequencies, I examined the artifact in the second-order scalogram S_2 at first-order center frequency $f_{\mathrm{crt},1} = 1.32\,\mathrm{Hz}$, the lowest frequency of $x_{\mathrm{syn},0}$ (Fig. 3.7b). The edge-effect artifacts are visible in the first and last 25 s of each example below 0.5 Hz. With an increasing value of α the amplitude of the edge-effect artifacts decreases. Nevertheless, to analyze the impact of the taper in more detail I plotted the $L2$ Norm

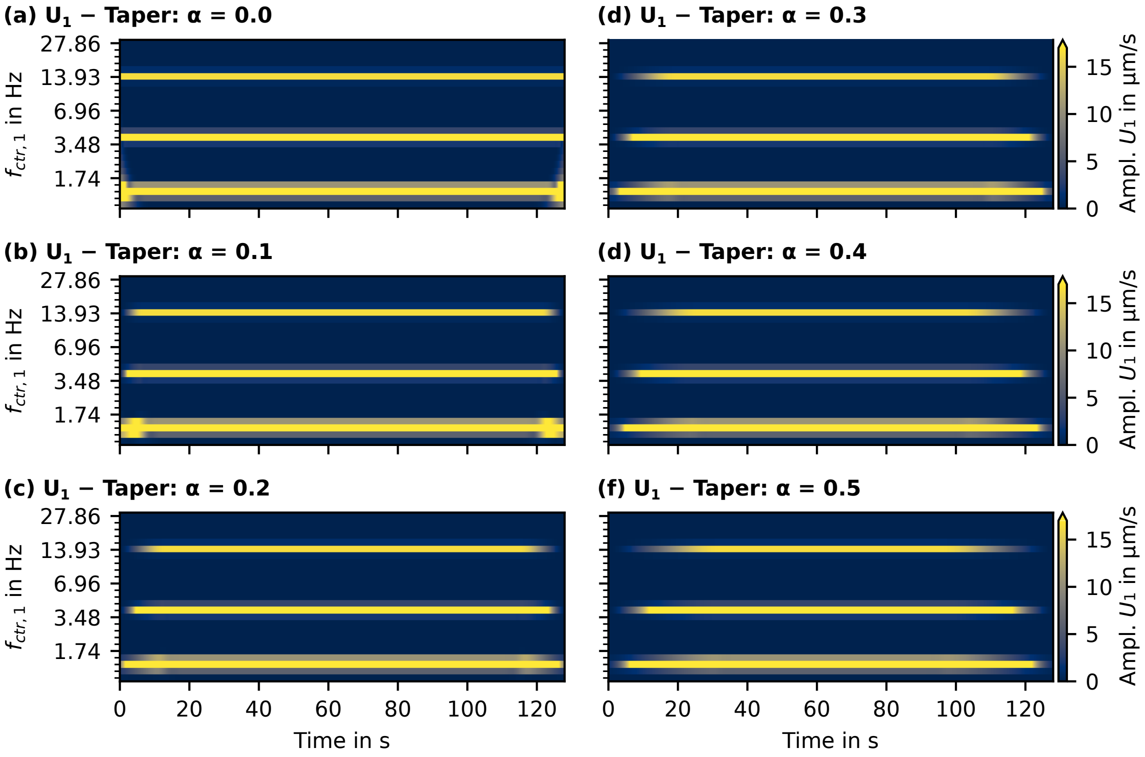


Figure 3.6: First-order scalograms U_1 of signal $x_{\text{syn},0}$ (Fig. 3.3). The signal is tapered by Tukey windows, Eq. 3.8, with values of α in a range from 0.0 to 0.5. The color scale was clipped to accentuate the visibility of the edge-effect artifacts, visible at the boundaries of the time windows, particularly evident in lower frequencies.

$$L2 = \sqrt{\sum_{k=1}^{n} |x_k|^2}, \quad x = \{x_1, x_2, \ldots, x_n\}, \tag{3.10}$$

of S_2 at $f_{\text{crt},1} = 1.32\,\text{Hz}$ using finer sampled values for α (Fig. 3.7c). After $\alpha = 0.4$, the $L2$ norm does not change significantly, thus $\alpha = 0.4$ is used for further analysis. To reduce the effect of amplitude attenuation on the flanks of the taper, only the part of the signal that is unaffected by the taper is used to calculate the scattering coefficients.

Edge-effect and tapering artifact

The second-order scalogram of the tapered signal still contains artifacts (Fig. 3.7). Therefore, I investigated whether these artifacts are due to the edge-effect or are caused by the attenuation of the signal at the edge of the taper. Figure 3.8a shows the first-order scalogram S_1 and the second-order scalogram S_2 at each of the frequencies of the non-tapered signal $x_{\text{syn},0}$. Figure 3.8b on the other hand shows the same plots for the Tukey tapered signal with $\alpha = 0.4$. While the taper artifacts in S_2 dominate at $f_{\text{crt},1} = 13.93\,\text{Hz}$ and $4.00\,\text{Hz}$, the edge effect artifacts seem to dominate at $f_{\text{crt},1} = 1.32\,\text{Hz}$. However, their amplitude is much smaller than that of the untapered signal scalogram, and, therefore, the result is in favor of applying the taper.

3.3.3 Impact of normalizing the wavelets

Previously, we discussed that the wavelets of the applied filter banks have unit energy, causing the attenuation of high frequencies (Fig. 3.9a). Consequently, the energy in the scalogram is shifted to lower frequencies. In turn, this affects the interpretation of the

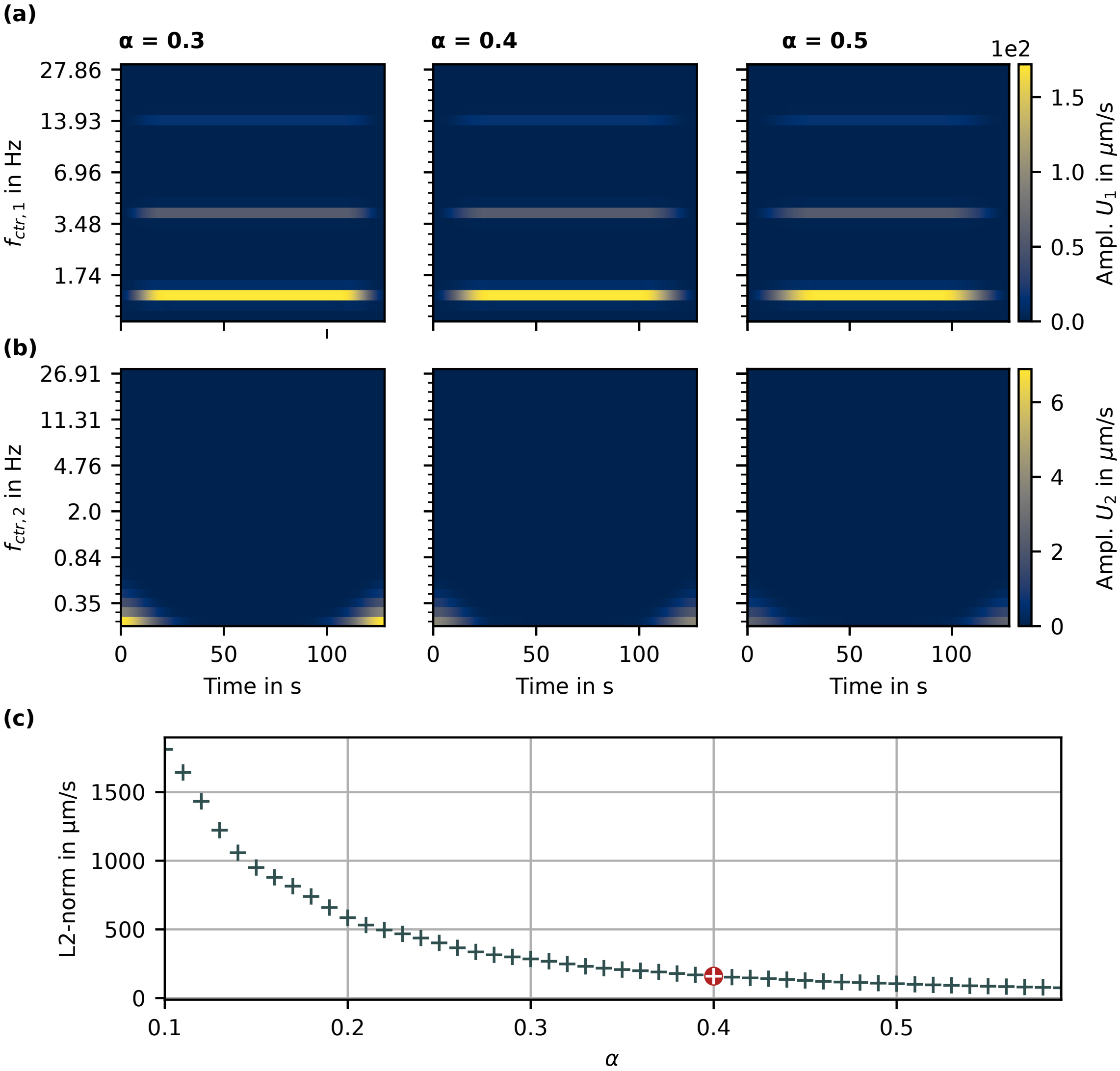


Figure 3.7: Influence of applying a Tukey window to the input signal of the initial filter bank on the first- (U_1) and second-order scalogram (U_2). (a) and (b) show the effect of changing the values of $\alpha = 0.3$, 0.4, and 0.5. (a) shows the first-order scalogram and (b) the second row the second-order scalogram of the center frequency of $f_{\mathrm{crt},1} = 1.32\,\mathrm{Hz}$. (c) displays the $L2$ Norm of the second-order scalogram of the center frequency of $f_{\mathrm{crt},1} = 1.32\,\mathrm{Hz}$. The red circle marks $\alpha = 0.4$ the taper is chosen for further analysis.

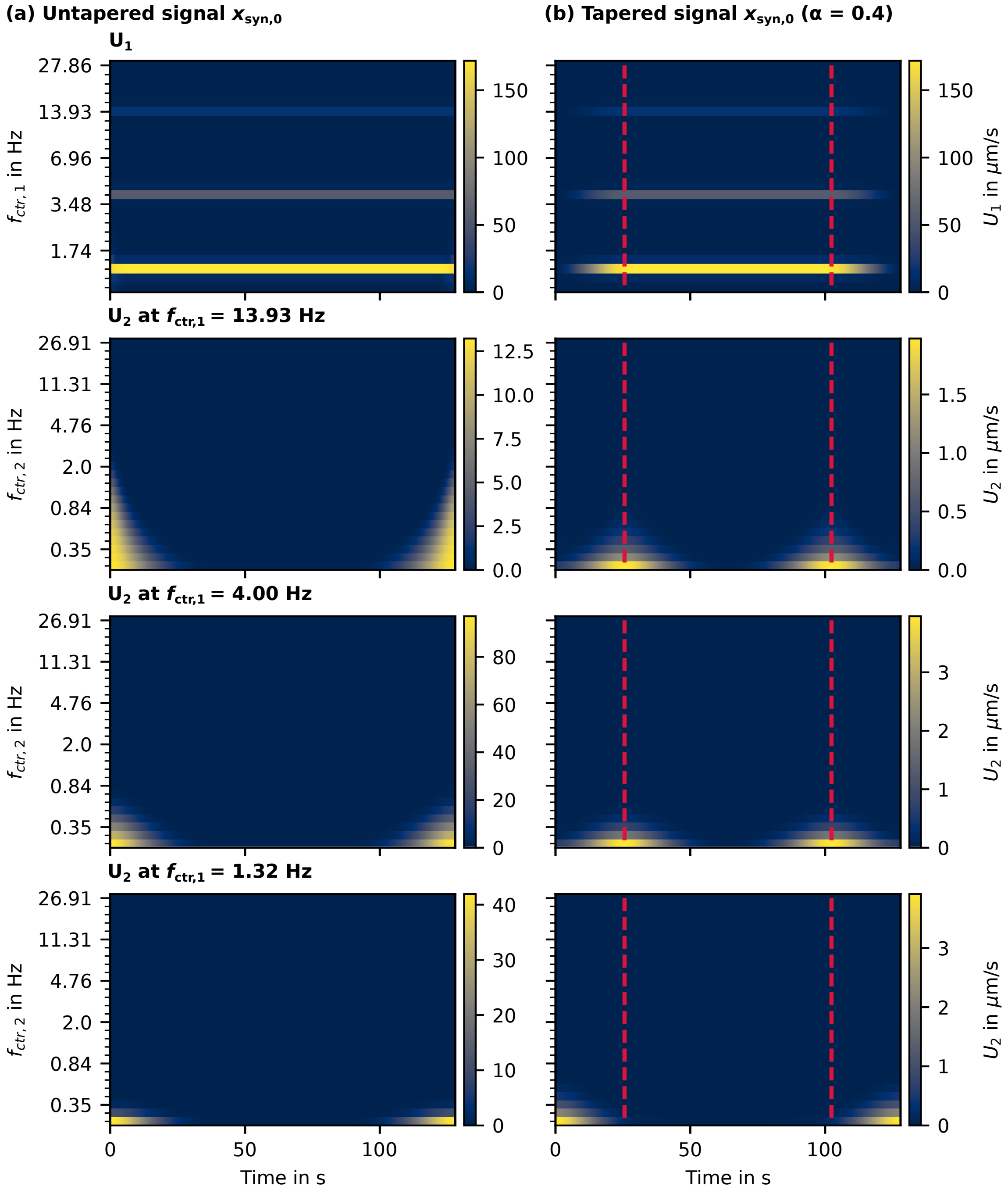


Figure 3.8: Comparison of the artifacts due to the edge-effect and due to tapering. In (a) no taper and in (b) a Tukey taper with $\alpha = 0.4$ is applied to signal $x_{\text{syn},0}$. The red dashed lines mark the region in which the taper does not affect the input signal. The first row shows the first-order scalogram and the other rows the second-order scalogram of frequencies 13.96 Hz, 4.00 Hz, and 1.32 Hz, respectively.

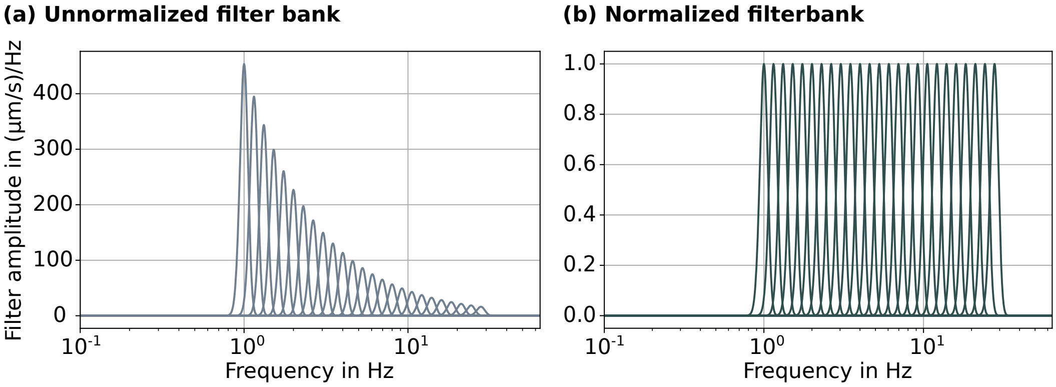


Figure 3.9: Amplitude of the first order filter bank. (a) without and (b) with $L1$ normalization applied to the wavelets in time.

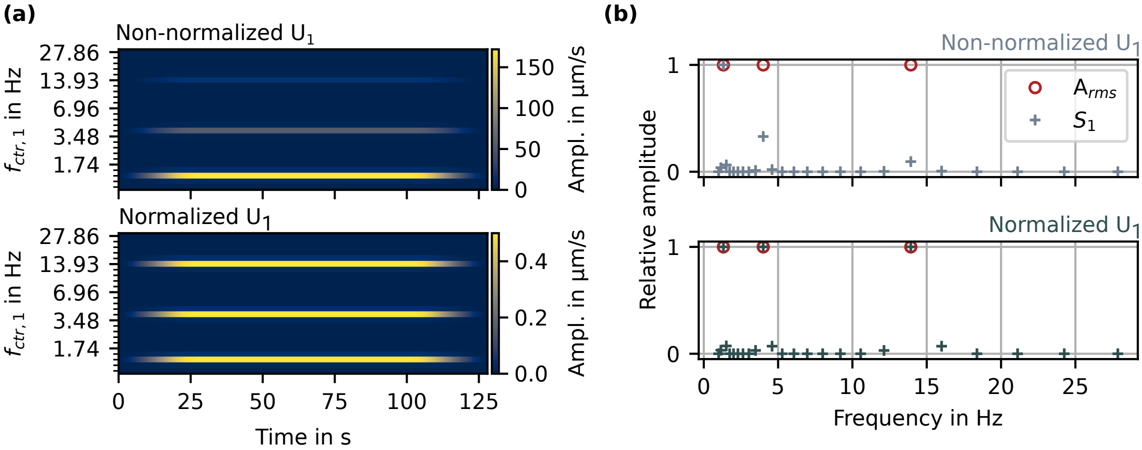


Figure 3.10: Effects of the $L1$-normalization of the filter bank on signal $x_{\text{syn},0}$. (a) shows the non-normalized first-order scalogram U_1 (top), and normalized U_1 (bottom), respectively. (b) depicts the relative root mean square amplitudes of the input signal, marked with red circles, compared to the relative first-order scattering coefficients S_1 (plus icon), not normalized (top) and normalized (bottom).

amplitudes in the scalogram. Since the signal $x_{\text{syn},0}$ is the superposition of three sine functions whose frequencies correspond to the center frequencies of the first filter bank, the interpretation of the amplitude decay over the frequency can thus only be attributed to the wavelet transform (Fig. 3.10a, top).

Corresponding to Morel et al. (2023), I normalized the filter banks in the time domain with their $L1$ Norm. Consequently, the frequency responses of the wavelet are normalized to one (Fig. 3.9b) and the relative amplitude of the signal is thus preserved (Fig. 3.10a, bottom). The bottom plot of Figure 3.10b compares the relative first-order scattering coefficients S_1 obtained by the normalized wavelets (plus symbol) with the relative RMS amplitudes of the signal (red circles). The relative RMS amplitudes of the signal and S_1 match perfectly. Even with a signal whose frequency content does not coincide with the center frequencies of the filters, the relative amplitudes are retained, although shifted to the center frequency, as shown in the appendix Figure C.1.

3.3.4 Impact of pooling

The next step of the scattering network involves the pooling operation applied to each of the scalograms. Through pooling, the scalogram of each time window is down-sampled to a single value per scale or center frequency, which leads to the network becoming translation invariant (Dumoulin and Visin, 2018; Mallat, 2010). The resulting values are referred to as the scattering coefficients. Common pooling methods include maximum, average, or median pooling, among others. The presented data is predominantly characterized by mono-frequent signals. As elaborated in Section 2.4.1, the only frequency changes of interest are caused by the change in the rotation rate of the WT. During these operational changes, the frequency glides from the current multiple of the BPF to the multiple of the adjusted BPF (Figs. 2.8 and 3.3, $x_{(\mathrm{WT,syn}),2}$). Consequently, the pooling operation has to reflect these data characteristics. Maximum pooling can be discarded as the focus is primarily on mono-frequent signals and not on events such as earthquakes, where maximum pooling has already proven to be beneficial (e.g. Steinmann et al., 2022a). If the data is normally distributed, average and median pooling are identical. However, seismic records typically deviate from the normal distribution, which means that the results of average pooling, in contrast to median pooling, are influenced by large outliers. These outliers can be short-term events, e.g., a passing car or a local earthquake, within the 128 s time window. The influence of these events is undesirable. On the other hand, a change in the rotation rate can last in the order of seconds, which is classified as a short-term event (Fig. 3.3, $x_{(\mathrm{WT,syn}),2}$). And these, in turn, are events of interest. Therefore, I evaluate the influence of both average and median pooling on the signals $x_{\mathrm{syn},1}$ and $x_{\mathrm{syn},2}$ based on the first-order scattering coefficients S_1 (Fig. 3.11). Signal $x_{\mathrm{syn},1}$ simulates an WT running at 12 rpm and includes no shift in frequency. As expected, the difference between both pooling operations is barely noticeable, and each of the expected frequency peaks highlighted with dashed lines in Fig. 3.11c match S_1. On the other hand, signal $x_{\mathrm{syn},2}$ includes a frequency shift from 19.2 Hz to 13.6 Hz and from 13.6 Hz to 11.2 Hz, although the latter is not included in S_1 as it is located below the flanks of the taper (Fig. 3.11b). Figure 3.11d confirms that these short-term events are identified with average pooling, but not with median pooling. The signal at 13.6 Hz, a multiple of BPF, which corresponds to 8.5 rpm, does not even appear utilizing median pooling. To include changes of the WT's rotation rate in the scattering coefficients, I, therefore, use average pooling from this point on. Nevertheless, it is important to take into account that other short-term events can influence the resulting S_1.

3.3.5 Impact of the quality factor

The quality factor q regulates the number of oscillations of the complex Morlet wavelet and thereby, its spectral bandwidth. A higher quality factor q results in an increased number of oscillations, leading to a narrower spectral width, whereas a lower quality factor q yields fewer oscillations and consequently a broader spectral width. The former results in a higher frequency resolution, while the latter improves the temporal resolution.

In the subsequent analysis, q_1 denotes the quality factor of the first filter bank and q_2 represents that of the second. I evaluate integer values between one and four for both layers, beginning with q_1. As shown in Figure 3.12a and b, an increase in the quality factor leads to a focusing of the energy on a narrower frequency range. For $q_1 = 1$ in particular, the bandpass filters overlap considerably, which leads to energy smearing across the three center frequencies of the signal $x_{\mathrm{syn},0}$. Conversely, an increase in the quality factor leads to more pronounced maxima at these frequencies.

Next, we investigate the influence of the quality factor on the resolution of synthetic seismic WT emissions utilizing signal $x_{\mathrm{syn},2}$, which encompasses frequency shifts from 19.2 Hz to 13.6 Hz and 11.2 Hz, along with four constant frequencies (Fig. 3.12c and d). At low quality factors, the energy of 11.2 Hz and 13.6 Hz merges with the peak of 19.2 Hz. The result improves with $q_1 = 2$ or with $q_1 = 3$, yet 11.2 Hz remains unresolved. With a quality factor

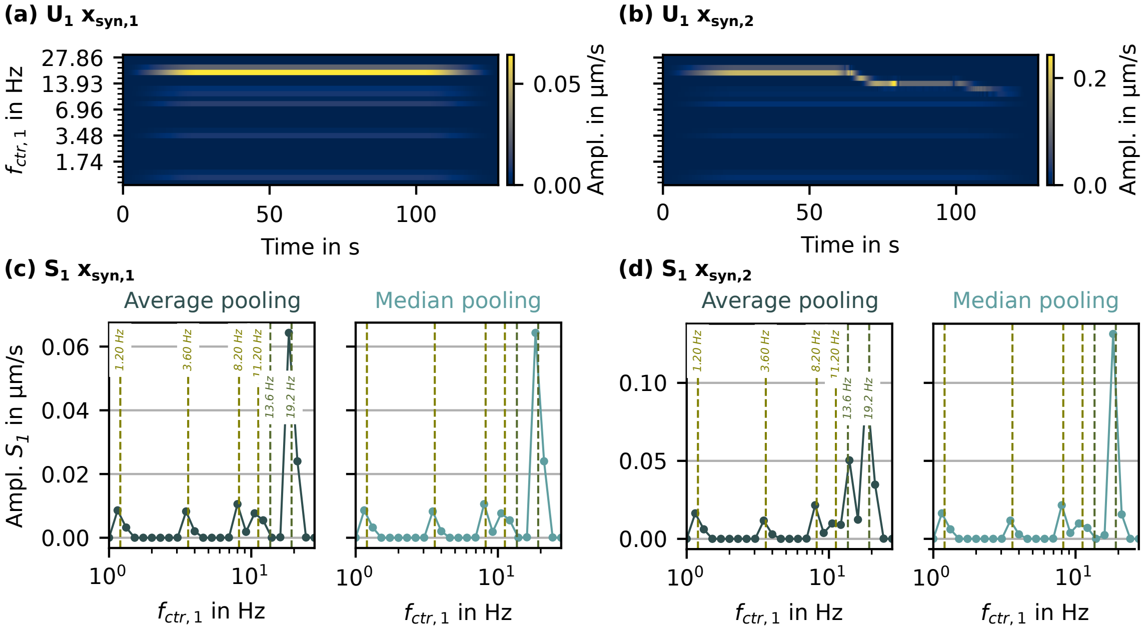


Figure 3.11: Effect of average and median pooling on the scattering coefficients. (a) and (b) depict the first-order scalogram of signals $x_{\text{syn},1}$ and $x_{\text{syn},2}$ (Fig. 3.3). (c) and (d) show the first-order scattering coefficients S_1 calculated with average (left) and median pooling (right) of these signals.

of $q_1 = 4$, all signal frequencies $x_{\text{syn},2}$ are distinctly resolved.

Subsequently, we focus on tuning the quality factor of the second filter bank q_2. As before, I examine quality factors between one and four. Figure 3.13b depicts the result of this for the second-order scalograms S_2 at $f_{\text{crt},1} = 12.13$ Hz. I will focus on two features within the area of the dashed lines. Firstly, spanning the interval from 65 s to 85 s below 0.84 Hz is a sharp triangular structure. This feature is attributable to the taper at the end of the sweep and the onset of the 13.6 Hz signal content. Notably, an increase in the quality factor leads to a reduction in the amplitude of this triangular feature. Secondly, we consider the range between 70 s and 105 s, and frequencies $f_{\text{ctr},2}$ ranging from 2.00 Hz to 4.76 Hz. With a quality factor of $q_2 = 1$, energy is distributed over the entire region. However, with higher quality factors, two prominent lines emerge at 9.15 Hz and 2.83 Hz, which are most distinct at $q_2 = 4$. Consequently, we achieve the best result with a quality factor of $q_2 = 4$.

3.3.6 Synthetic versus recorded signals

So far, I have discussed the design of the scattering network by using synthetic signals. In the following, I will discuss the application of this network to the recorded signals of IW08B, $x_{\text{WT},(1,2,3)}$, presented in the beginning of Section 3.3 and compare them with the results of the synthetic signals $x_{\text{syn},(1,2,3)}$ (Fig. 3.3). Furthermore, we will take a closer look at the second-order scalogram U_2 and the scattering coefficient S_2, as we have mainly discussed S_1 (e.g. Figs. 3.11 and 3.12) and only briefly analyzed an example from U_2 (Fig. 3.13).

First-order scalogram and scattering coefficient

The recorded signals $x_{\text{WT},(1,2,3)}$ and the synthetic signals $x_{\text{syn},(1,2,3)}$ derived from them are identical in their average amplitude at the frequencies listed in Table 3.1. However, some frequency content of the recorded signals is not incorporated into the synthetic signals. While the first-order scalograms U_1 of the synthetic signals only show maxima at the center frequencies corresponding to their frequency content $f_{\text{crt},1}$, the energy in U_1 of the recorded

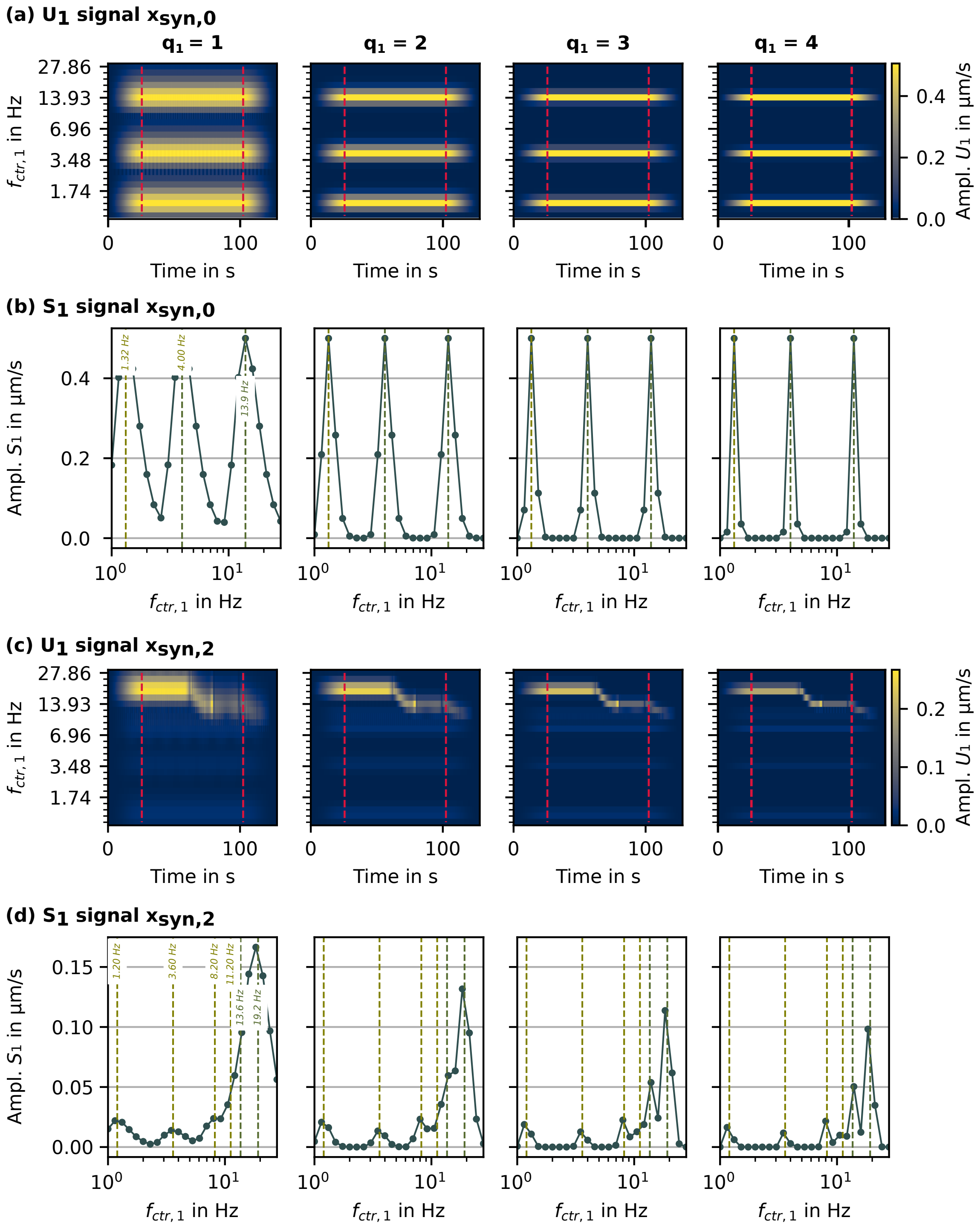


Figure 3.12: Impact of the quality factor of the first filter bank q_1. (a) shows the first-order scalogram and (b) the first-order scattering coefficients of signal $x_{\mathrm{syn},0}$. This also corresponds to (c) and (d), which display the results for signal $x_{\mathrm{syn},2}$. The columns from left to right correspond to the application of the quality factors that range from low to high. The red dashed lines in (a) and (c) mark the area in which the signal is not affected by the taper and the scattering coefficients are calculated. The dashed lines in (b) and (d) mark the center frequencies of the input signal.

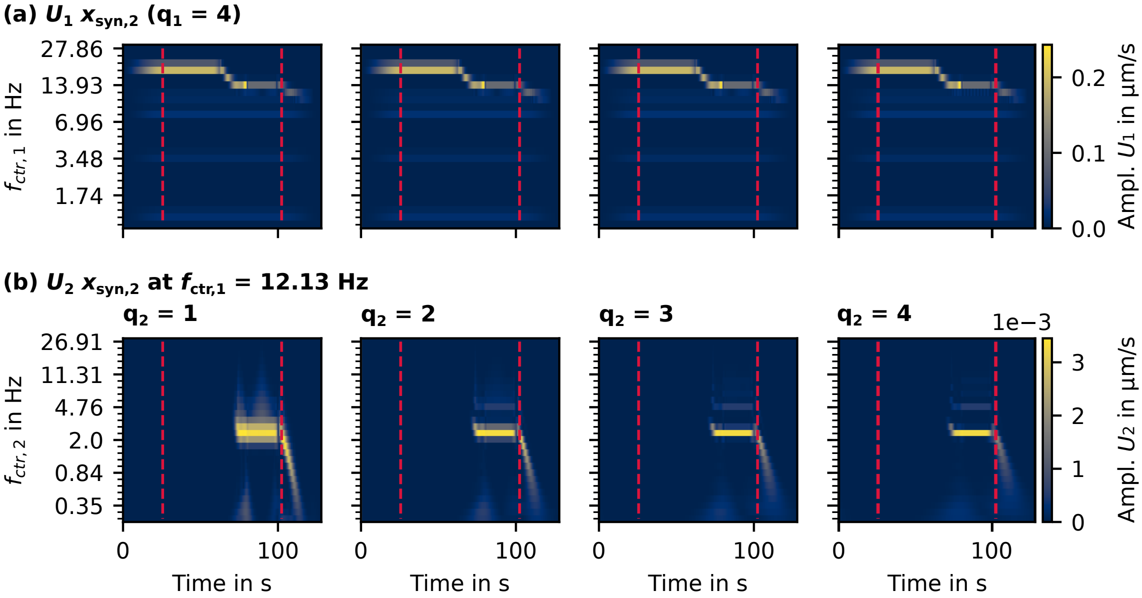


Figure 3.13: Impact of the quality factor of the second filter bank q_2. (a) shows the first-order scalogram utilizing a quality factor of $q_1 = 4$. (b) displays the second-order scalogram of $f_{\text{crt},1} = 12.13\,\text{Hz}$. The quality factors vary between integer values from one to four, applied from left to right. The red dashed lines mark the area in which the signal is not affected by the taper and the scattering coefficients are calculated.

signals is more broadly distributed (Fig. 3.14a and b). Especially in U_1 of $x_{\text{WT},3}$ the energy is distributed almost randomly with some local variations. Nevertheless, the maxima in U_1 of $x_{\text{WT},(1,2)}$ and $x_{\text{syn},(1,2)}$ match well, and even the change in the rotation rate between 60 s and 80 s looks very similar between $x_{\text{syn},2}$ and $x_{\text{WT},2}$.

The signal oscillations significantly differ between U_1 of the recorded and synthetic signals. While the synthetic signals show a constant maximum over time, which only shifts in the signal $x_{\text{syn},2}$ due to the change in the rotation rate, the recorded signals show oscillations in the order of microseconds (Fig. 3.15b). We will therefore take a closer look at the center frequency $f_{\text{crt},1} = 18.38\,\text{Hz}$ of the signals $x_{\text{syn},1}$ and $x_{\text{WT},1}$ (Fig. 3.15). Figure 3.15b represents the wavelet transform W_1 of both signals before calculating their modulus and Figure 3.15c after applying the modulus operation. When calculating the absolute value of a signal, negative values are folded up and thus lead to an increase in frequency. This is not the case with Figure 3.15c, where the modulus of the two signals is equal to their envelope. This is due to the sampling rate and leads to the modulus operation corresponding to an additional low-pass filter. However, why do we not see any oscillations in the synthetic signals? The wavelets filter the signals in a specific bandwidth centered at $f_{\text{crt},1}$. While the synthetic signals in the frequency range of a single wavelet filter are almost mono-frequent, the recorded signals include a wider range of frequencies. This in turn leads to the oscillations visible in Figure 3.15c and in the scalograms U_1 Figure 3.14.

After the application of average pooling, it is not surprising that the first-order scattering coefficients S_1 of the recorded signals and their synthetic counterpart are similar at the marked frequencies (Fig. 3.14, dashed lines). They differ only in their amplitude, attributable to the presence of frequency content that is not incorporated in the synthetic signals. For instance, the signals $x_{\text{WT},(1,2)}$ show a maximum value at 6.0 Hz and 27.6 Hz, the tenth and 46th multiples of BPF at 12 rpm (Fig. 3.3). These maxima are not visible in signal $x_{\text{WT},3}$, as this is an example when *WT 3* is not in operation.

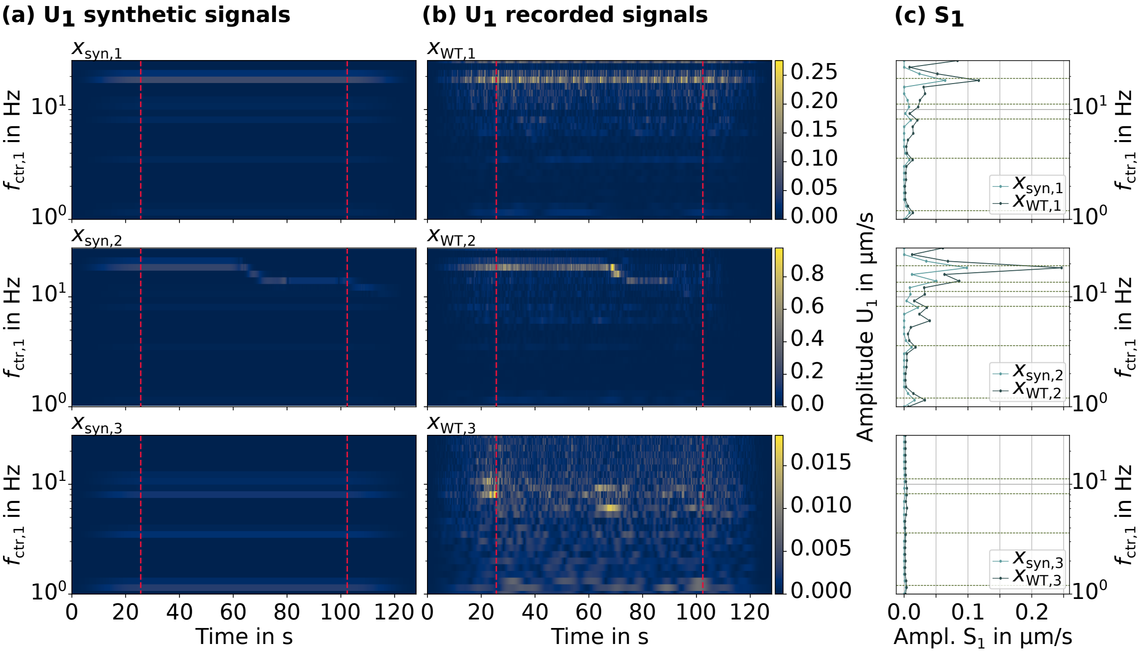


Figure 3.14: First-order scalograms U_1 of (a) the synthetic signals $x_{\mathrm{syn},(1,2,3)}$ and (b) of the recorded signals $x_{\mathrm{syn},(1,2,3)}$. (c) shows the corresponding first-order scattering coefficients S_1.

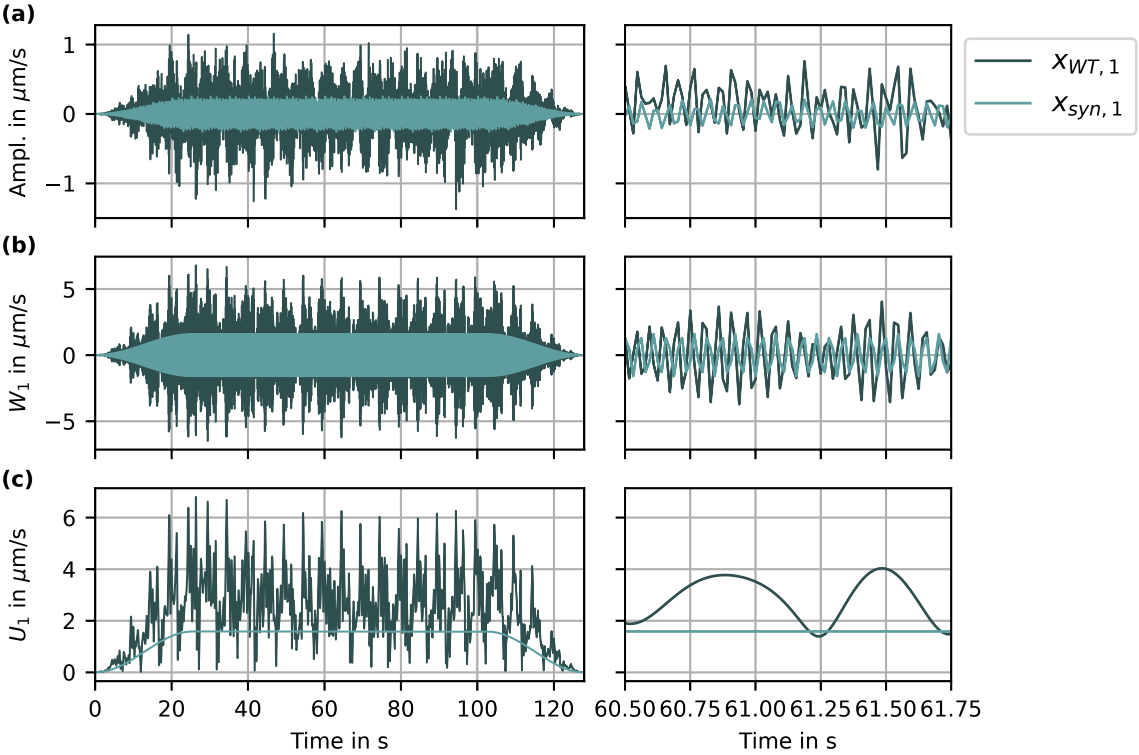


Figure 3.15: (a) Signal $x_{\mathrm{syn},1}$ and $x_{\mathrm{WT},1}$ before the wavelet transformation. (b) Wavelet transformed signal W_1 at the center frequency $f_{\mathrm{crt},1} = 18.38\,\mathrm{Hz}$. (c) The absolute value of the wavelet transforms U_1 visible in (b). The left column shows the entire signal, while the right depicts a zoomed part of 60.5 s to 61.75 s of the signal.

Second-order scalogram and scattering coefficient

The second-order scalogram U_2 and scattering coefficients S_2 illustrate variations within the first-order scalogram S_1. In acoustics, these variations may represent frequency intervals between harmonics or amplitude modulation (Anden and Mallat, 2014). In Figure 3.15 we discussed the oscillations observed in the first-order scalogram of the signal $x_{\mathrm{WT},1}$ at the center frequency $f_{\mathrm{crt},1} = 18.38\,\mathrm{Hz}$. Let us now discuss how these oscillations manifest themselves in U_2 and whether we can identify them in S_2. Subsequently, we will analyze the change in the rotation rate in the signal $x_{\mathrm{WT},2}$ and its synthetic counterpart, $x_{\mathrm{syn},2}$, and conclude the discussion of U_2 and S_2 with an example of signals $x_{\mathrm{syn},3}$ and $x_{\mathrm{WT},3}$.

At a center frequency of $f_{\mathrm{crt},1} = 18.38\,\mathrm{Hz}$, U_2 of the synthetic signal $x_{\mathrm{syn},1}$ predominantly exhibits weak amplitude signal changes caused by tapering (Fig. 3.16a, left). In contrast, the oscillations of the recorded signal $x_{\mathrm{WT},1}$, as shown in Figure 3.15, originate from two dominant frequencies, 0.21 Hz and 0.59 Hz, which correspond to periods of 4.76 s and 1.69 s, respectively. Additionally, in the range of 1 Hz to 2 Hz a region of increased energy including local maxima is characterized by smaller local oscillations within 1 s to 0.5 s. These effects are evident in the corresponding S_2, where the first two maxima of the scalogram U_2 are isolated and the range between 1 Hz to 2 Hz appears blurred (Fig. 3.17a right). The second-order scattering coefficients S_2 of signal $x_{\mathrm{syn},1}$ at $f_{\mathrm{ctr}} = 18.38\,\mathrm{Hz}$ primarily reflect tapering artifacts visible below 0.35 Hz (Fig. 3.17a left).

The second pair of signals, $x_{\mathrm{syn},2}$ and $x_{\mathrm{WT},2}$, contains the change in the rotation rate from 12 rpm to 8.5 rpm and thus the change in the observed 32nd multiple of BPF from 19.2 Hz to 13.6 Hz (Fig. 3.14b). In the first-order scalogram U_1, these changes are represented by the center frequencies $f_{\mathrm{crt},1} = 18.38\,\mathrm{Hz}$, 16 Hz and 13.93 Hz. The latter is constant between 74 s and 93 s and its second-order scalogram U_2 is shown in Figure 3.16b. We observe a maximum between the center frequencies $f_{\mathrm{ctr},2} = 0.59\,\mathrm{Hz}$ and 0.42 Hz located between 70 s to 95 s. This correlates with the period when the rotation rate was 8.5 rpm. The corresponding BPF is 0.425 Hz and thus a possible explanation for the lower limit of the maximum. However, U_2 of the synthetic signal $x_{\mathrm{syn},2}$ on the scale $f_{\mathrm{crt},1} = 13.93\,\mathrm{Hz}$, the effects of the tapering at the edges of the signal and the frequency changes are mingled. The two cone-shaped structures are caused by the tapered artifacts at the beginning and end of the signal component at 13.6 Hz and overlap the area in which the maxima corresponding to the frequency change should be located.

Considering the corresponding second-order scattering coefficients S_2 in Figure 3.17b, most of the energy of both signals, $x_{\mathrm{syn},2}$ and $x_{\mathrm{WT},2}$, is located between $f_{\mathrm{crt},1} = 13.93\,\mathrm{Hz}$ to 18.38 Hz. While the amplitudes of S_2 of the synthetic signal $x_{\mathrm{syn},2}$ increase between $f_{\mathrm{ctr2}} = 2.0\,\mathrm{Hz}$ and 0.13 Hz, the energy in S_2 of the recorded signal $x_{\mathrm{WT},2}$ is more widely scattered and contains local maxima that can be attributed to the more complex signal characteristics of $x_{\mathrm{WT},2}$.

With this, we move on to the last example, Figure 3.16c, the second-order scalogram U_2 at $f_{\mathrm{crt},1} = 9.19\,\mathrm{Hz}$ of signals $x_{(\mathrm{syn},\mathrm{WT}),3}$. U_2 of signal $x_{\mathrm{syn},3}$ contains, in addition to the taper artifacts below $f_{\mathrm{ctr},2} = 2\,\mathrm{Hz}$, a maximum which extends over the entire time window at $f_{\mathrm{ctr},2} = 2.83\,\mathrm{Hz}$, which corresponds to the frequency interval between 8.2 Hz and 11.2 Hz. The first-order filter at $f_{\mathrm{crt},1} = 9.19\,\mathrm{Hz}$ is already quite narrow, and the frequency content of the two eigenmodes at 8.2 Hz and 11.2 Hz is only picked up by the flanks of this filter and, therefore, attenuated in the amplitudes. The amplitudes decrease further with the application of the second-order filter bank, which leads to the low amplitudes in the order of 10^{-7}. Whereas U_2 of the signal $x_{\mathrm{WT},3}$ has much higher amplitudes, similar to its first order scalogram U_1, the energy below $f_{\mathrm{ctr},2} = 2.83\,\mathrm{Hz}$ is widely scattered and some local maxima are visible. Again, this is reflected in the second-order scattering coefficients S_2, the

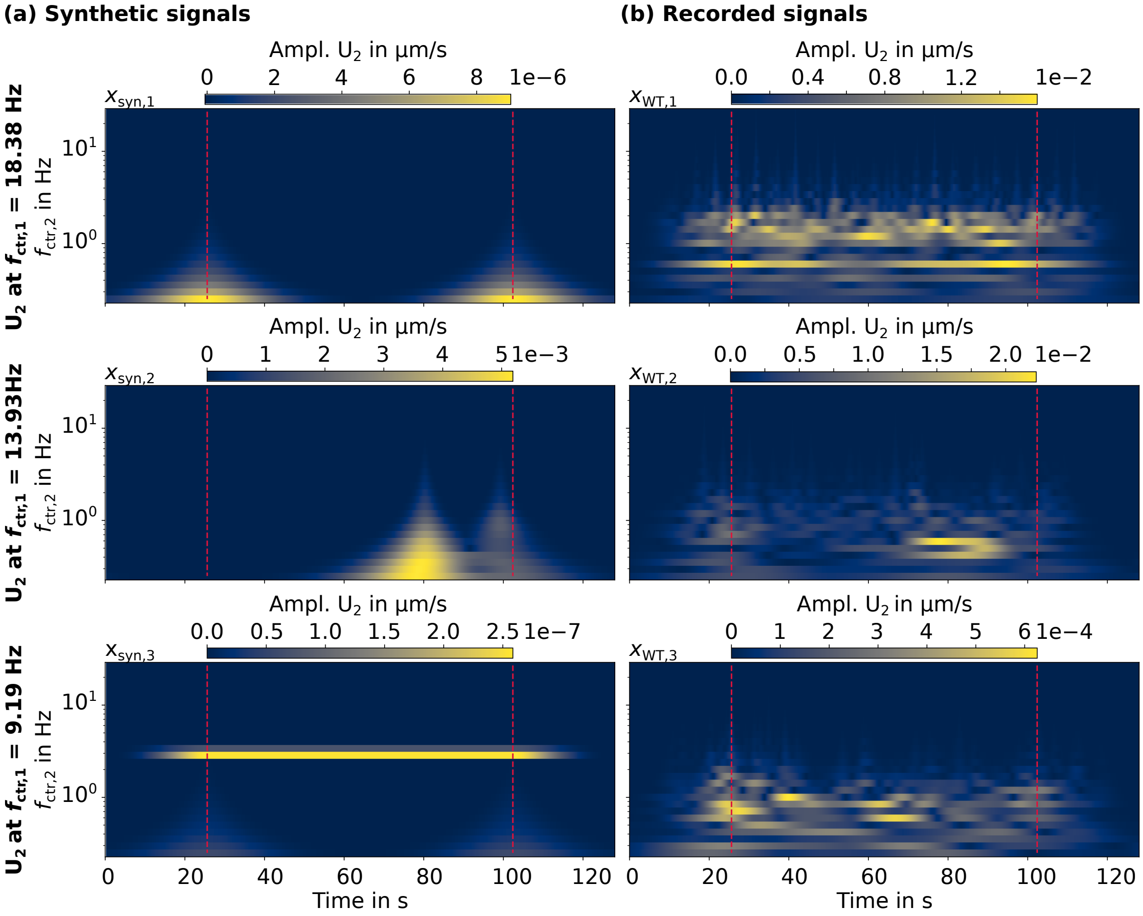


Figure 3.16: Second-order scalogram U_2 of (a) the synthetic signals $x_{\mathrm{syn},(1,2,3)}$ and (b) of the recorded signals $x_{\mathrm{syn},(1,2,3)}$. The first row shows the scalogram at center frequencies $f_{\mathrm{crt},1} = 18.38\,\mathrm{Hz}$, the second at 13.93 Hz, and the third row at 19.19 Hz. The red dashed lines mark the region where the taper did not affect the input signal.

energy accumulates under a hypothetical line that runs approximately from ($f_{\mathrm{crt},1} = 1.0\,\mathrm{Hz}$, $f_{\mathrm{ctr},2} = 0.25\,\mathrm{Hz}$) to ($f_{\mathrm{crt},1} = 27.86\,\mathrm{Hz}$, $f_{\mathrm{ctr},2} = 4.76\,\mathrm{Hz}$). The hypothetical line can likewise be found in Figure 3.16a and b in S_2 of the recorded signals. This is due to the fact that the first-order filter bank has filtered out the low frequencies and the second-order filter bank cannot in turn recover these. The synthetic S_2 does not show these patterns as the frequency content of the synthetic signals is limited. S_2 of signal $x_{\mathrm{syn},3}$, similar to the other examples, shows the artifacts of tapering and in addition the previously described maximum at $f_{\mathrm{ctr},2} = 2.83\,\mathrm{Hz}$, which correlates with the frequency interval between 8.2 Hz and 11.2 Hz.

Summarizing, the first-order scattering coefficients S_1 of the synthetic and the recorded signal are very similar except for some minor differences. Although we find some of the observed patterns in both second-order scalograms U_1, the second-order scattering coefficients are hardly comparable as the overall frequency content of the recorded signal and synthetic signal is not similar enough, thus these synthetic signals cannot be used to verify the second-order scattering coefficients S_1 of the recorded signals.

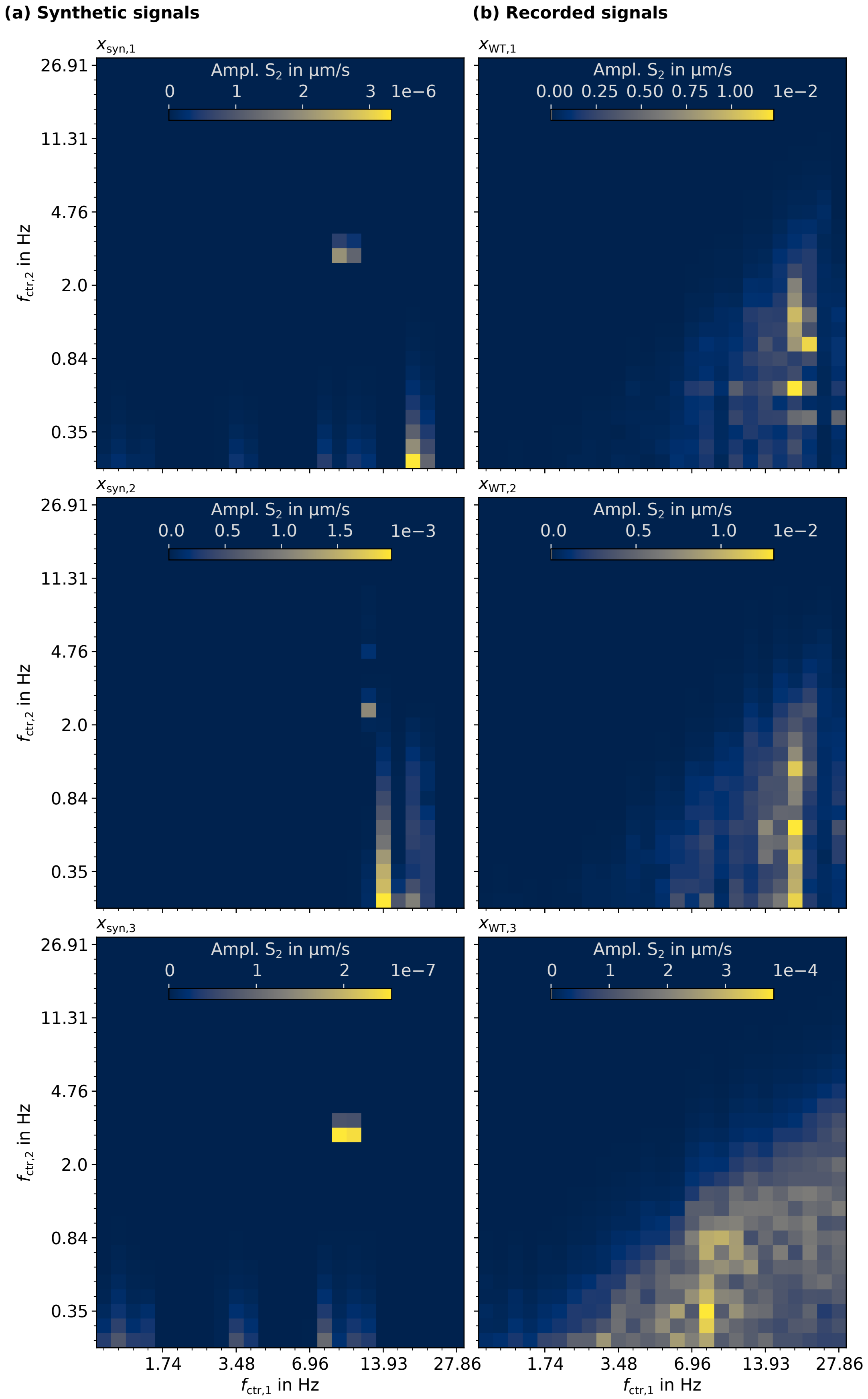


Figure 3.17: Second-order scattering coefficients S_2 of (a) the synthetic signals $x_{\text{syn},(1,2,3)}$ and (b) of the recorded signals $x_{\text{syn},(1,2,3)}$.

Table 3.2: Final setup of the filter banks. J is the number of octaves of the filter bank and Q is the filter bank resolution, the number of wavelets per octave. The quality factor q controls the oscillations of the wavelet. Each wavelet is normalized by its $L1$ Norm.

Filter banks	Octaves J	Resolution Q	Quality factor q	Normalization
1st-order	5	5	4	$L1$
2nd-order	7	4	4	$L1$

3.4 Concluding remarks

In this chapter, I discussed the design of the scattering network used to extract the most relevant patterns from the recordings of station IW08B. Henceforth, I will use the scattering coefficients matrix of the entire dataset recorded at station IW08B. Figure 3.18 shows the resulting first-order scattering coefficients S_1 of the Z-component. N- and E-components are in the appendix Figures C.2 and C.3. Each frequency related to the seismic WT emissions, the eigenmodes of WT and the 32nd multiples of BPF, except for the two noise-reduced modes at 17.6 Hz and 19.2 Hz, is covered by a separate wavelet of the first-order filter bank. By applying wavelet normalization, the S_1 amplitudes are comparable with the amplitude of the input signal. The final setup of the filter banks is summarized in Table 3.2. The sampling rate of the three component input signal is reduced to 64 Hz and cut into 128 s time windows with 64 s overlap. With this, the total number of time windows is 151 199, and the first-order scattering coefficients S_1 have a dimension of $151\,199 \times 3 \times 25$ and the second-order scattering coefficient S_2 $151\,199 \times 3 \times 25 \times 28$. The concatenated scattering coefficient matrix has a size of $151\,199 \times 2175$. While S_1 contains the primary frequency information of the input dataset, S_2 shows the variations within S_1, e.g., frequency intervals or amplitude modulation. With this, we move on to the next chapter of my study.

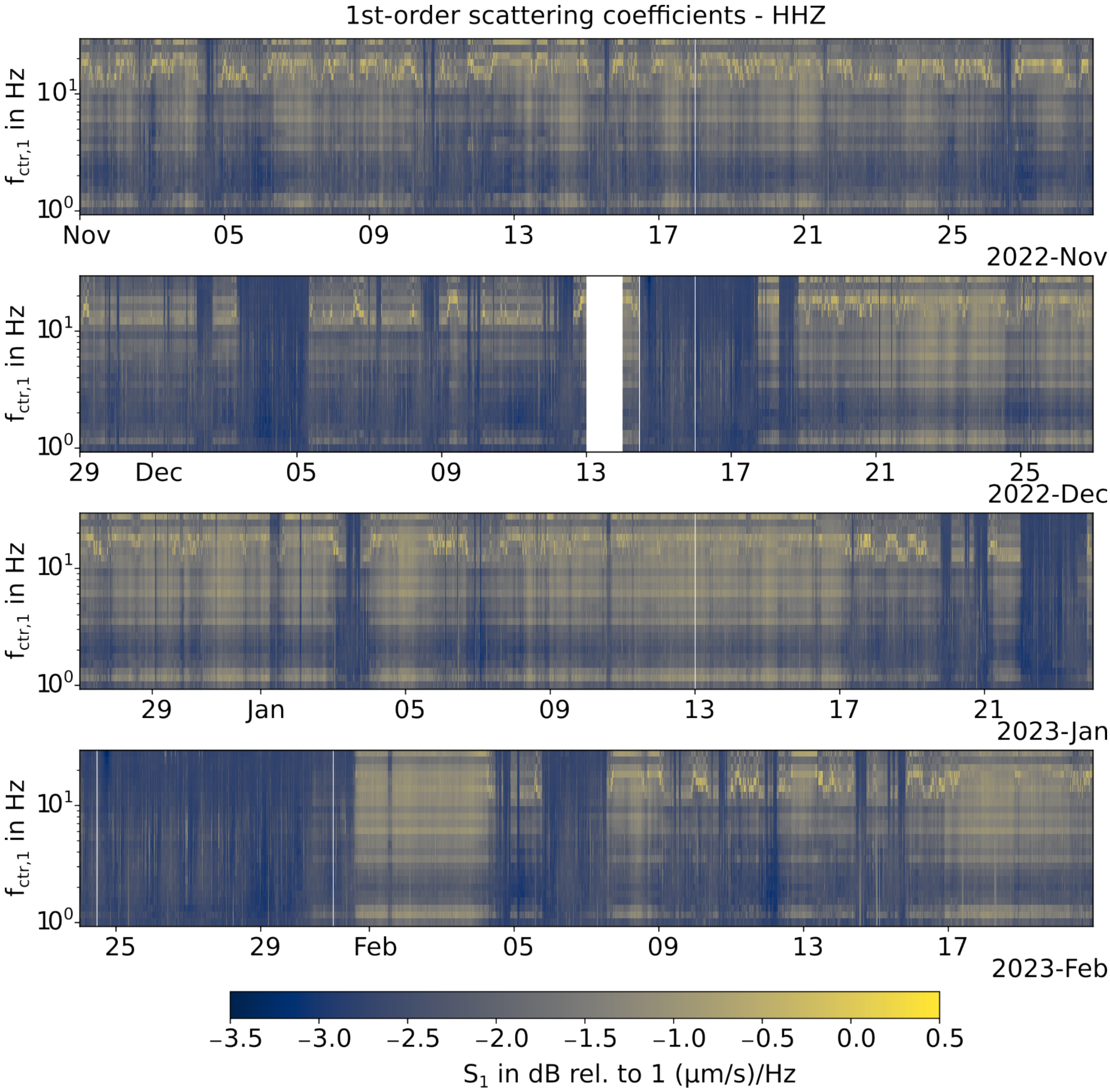


Figure 3.18: First-order scattering coefficients S_1 of the Z-component recorded at IW08B. There are two data gaps, during December 13, 2022, the station did not record any data and on December 14, 2022, between 11:02 and 11:06 we conducted station service. The white lines on November 13 and 23, 2022, and the ones beneath these in the other subplots are plotting artifacts.

Chapter 4

Data-driven pattern recognition of seismic wind turbine emissions

In the previous chapter, I introduced the scattering network a method to extract the underlying structure from a dataset. The inputs for the scattering network are continuous three-component seismograms recorded during the Inter-Wind project (c.f. Chapter 2). The seismograms are split into sliding time windows, with each component being transformed by the two-layer scattering network. As shown in Figure 3.2, the resulting scattering coefficients are concatenated to the scattering coefficient matrix S.

My next step is to analyze the underlying structure of S, by finding groups with similar data properties, to identify known and unknown patterns. For instance, the change in rotation rate of a WT can be determined by analyzing the spectrogram. The multiples of the BPF shift in frequency and the overall amplitude of the frequency content of the WT emissions increases with higher rotation rates (Fig. 2.8). Seismic WT emissions are not only related to the wind speed and with it the rotation rate of the WT but also to the wind direction. Results from Gaßner et al. (2023) and Neuffer et al. (2021) show that, depending on the eigenmode, Love waves are radiated in a downwind and Rayleigh waves in a crosswind direction and vice versa. However, the wind direction is not easily determinable if no reference data is available. Further sources or processes that influence the recording of ground movements are, for example, anthropogenic influences such as cars or other meteorological conditions such as temperature changes (Steinmann et al., 2022b). The manual separation of these various signal sources is very time-consuming and not always feasible. This study aims to develop a workflow that separates these signal sources in an unsupervised approach. The focus is set on the analysis of seismic sources associated with onshore WTs. I therefore test ML methods, in particular dimension reduction and clustering, for further analysis of the scattering coefficient matrix S of the IW08B dataset (Section 2.2.4 and Figs. 3.18, C.2 and C.3).

Clusters are groups of similar properties in a dataset. To determine the degree of similarity, measuring the distance L between elements x in a dataset is necessary. The most common way to measure the distance between two vectors x_a and x_b is to use a Minkowski distance,

$$L_p(x_a, x_b) = (\sum_{i=1}^{n} |x_{i,a} - x_{i,b}|^p)^{\frac{1}{p}} \quad \forall p \geq 1, p \in \mathbb{Z}^+, \tag{4.1}$$

especially the Euclidean distance with $p = 2$ (Murtagh and Contreras, 2012). In a high-dimensional space, however, the distance between any pair of points approaches the same

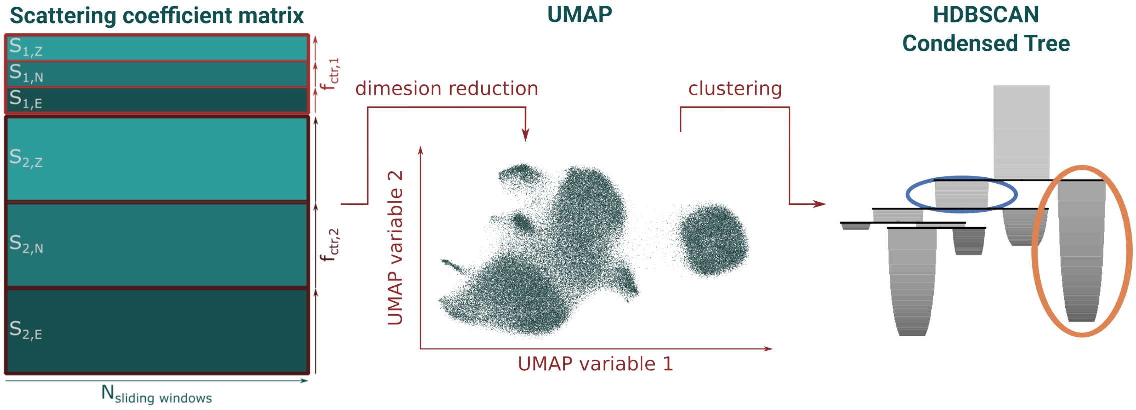


Figure 4.1: Workflow for dimension reduction and clustering. The high dimensional scattering coefficient matrix is reduced to two UMAP variables. Each point within the UMAP atlas represents one of the N sliding windows. The two UMAP variables, in turn, are utilized as input for the hierarchical density-based spatial clustering of applications with noise (HDBSCAN) clustering, here depicted as a dendrogram. The marked branches of the so-called condensed tree plot are the selected clusters. In this example, the clustering algorithm separated the data into two main clusters (blue and orange).

value. This is called the curse of dimensionality (Bellman, 1961). The scattering coefficient matrix S representing the IW08B dataset spans 2175 dimensions (Section 3.4). Hence, before clustering, I first need to reduce the dimensions of S (Fig. 4.1).

4.1 Dimension reduction

Reducing the dimensionality of a dataset has the advantage of avoiding the curse of dimensionality and reducing the memory requirements and computing time for subsequent processes. Dimension reduction aims to extract latent features from a dataset and remove redundant features, thus preprocessing the dataset for clustering or other tasks (McInnes et al., 2018).

There are many dimension reduction techniques and ways to group these different approaches. One way is to differentiate them according to how they handle distances, whether they tend to preserve pairwise distances and thus the global structure of the dataset or whether they focus on preserving the local structure (McInnes et al., 2018). For the seismic WT emissions discussed in this study, the global structure would be, for instance, whether *WT 3* is in operation or not, which is indicated by the amplitudes of the eigenmodes, amongst other features. Local structures are, e.g., the multiples of BPFs, which indicate the rotation rate of WT.

In this section, I present an example for each of these categories. Principal component analysis (PCA) preserves the global structure and is one of the most common techniques for dimension reduction (Garzon et al., 2022). Uniform manifold approximation and projection (UMAP), on the other hand, preserves the local distances in the dataset and is a recently developed neighborhood-based algorithm for manifold learning (McInnes et al., 2018).

4.1.1 Principal Component Analysis (PCA)

PCA is a linear dimension reduction technique that identifies orthogonal unit vectors along axes of maximum variance (Bloem, 2023). Initially, the axis of maximum variance is assigned to the first principal component (PC), followed by subsequent components representing the remaining variance in descending order. Hence, the PCs are the eigenvectors of the covariance matrix, and due to orthogonality, they are uncorrelated.

The PC can be derived from a matrix $X \in \mathbb{R}_n \times \mathbb{R}_m$ using singular value decomposition (SVD),

$$X = U\Sigma V^T, \tag{4.2}$$

where $U \in \mathbb{R}_n \times \mathbb{R}_n$ is an orthogonal matrix with the first p columns representing left singular vectors, V is an orthogonal matrix of size $\mathbb{R}_m \times \mathbb{R}_m$ with its first p columns being right singular vectors, and Σ is an $\mathbb{R}_n \times \mathbb{R}_m$ rectangular matrix with zeros entries except for its first p diagonal components, which are the singular values $s_1 \geq s_2 \geq \cdots \geq s_p \geq 0$. The PCs are the right singular vectors of V (Bloem, 2023).

While PCA effectively identifies linear structures in data by maximizing variance along orthogonal axes, manifold learning techniques offer a broader scope for capturing complex data structures beyond linear relationships. One such method is UMAP. In contrast to PCA, UMAP is based on capturing non-linear relationships within the data (McInnes et al., 2018). Recent studies, like Becht et al. (2019) and Cao et al. (2019), have demonstrated the performance of UMAP in extracting meaningful features and visualizing large datasets.

4.1.2 UMAP

UMAP is a graph-based algorithm employed for dimension reduction through manifold learning (McInnes et al., 2018). Manifold learning comprises non-linear dimension reduction algorithms aiming to describe datasets as "low-dimensional manifolds embedded in high-dimensional spaces" (VanderPlas, 2016). The mathematical foundation of UMAP is rooted in Riemannian geometry and topological data analysis (McInnes et al., 2018).

The algorithm of McInnes et al. (2018) is based on two primary steps. Initially, a k-neighborhood graph is constructed to represent the data manifold, followed by the learning of a low-dimensional representation of this graph, guided by three underlying axioms:

1. "There exists a manifold on which the data would be uniformly distributed."
2. "The underlying manifold of interest is locally connected."
3. "Preserving the topological structure of this manifold is the primary goal."

The high-dimensional graph is constructed with a so-called Čech complex (McInnes et al., 2018). A Čech complex is constructed from sets of n-simplices, which are basic components of polyhedra (Munkres, 1984). A 0-simplex is a point, a 1-simplex is a line segment between two 0-simplices, a 2-simplex is a triangle consisting of three 1-simplices, etc. To form a Čech complex, the topological structure is initially formed from 0-simplices. If these have a non-empty intersection, hence, if two 0-simplices are within a fixed radius to each other, they are combined with 1-simplices (Fig. 4.2a). If there is a non-empty triple intersection, three 1-simplices are combined to form a 2-simplex, etc. (McInnes et al., 2018). However, a fixed radius leads to strongly connected simplices in dense regions, and simplices may not connect at all in sparse regions (Fig. 4.2b). The former prevents the high-dimensional graph from having as few connections as possible and thus makes dimension reduction more difficult; the latter prevents axiom 2. from being fulfilled.

Instead, McInnes et al. (2018) used local distances in their algorithm using standard Riemannian geometry. If we stick to a three-dimensional manifold for explanation purposes, spheres with a given radius are formed around the 0-simplices i.e. data points. Using local distances, each of these spheres has a unit radius and extends to the nearest kth data point (Fig. 4.2c). A small value of k causes the graph to emphasize the local data structure, while a large value represents the global structure of the dataset. Each edge of the graph is weighted according to how far apart its points are. With this weighting scheme, McInnes et al. (2018) use a concept known as fuzzy coverage, which assigns values between zero and one to indicate the degree of inclusion in the spheres formed around the data points. In this approach, the center of the sphere is assigned the highest weight, which decreases towards the outer parts of the sphere.

In the manifold, each point must connect to at least one point (axiom 2.). To ensure this, the spheres extend to at least one neighboring point and continue fuzzy to the kth neighbor (Fig. 4.2d; McInnes et al., 2018). As up to two edges between points are possible, the high-dimensional graph is built with fuzzy simplicial sets (Fig. 4.2e) and merged into a fuzzy simplicial complex by merging two non-matching edges with weight a and b into a single edge with weight $a + b - a * b$(Fig. 4.2f).

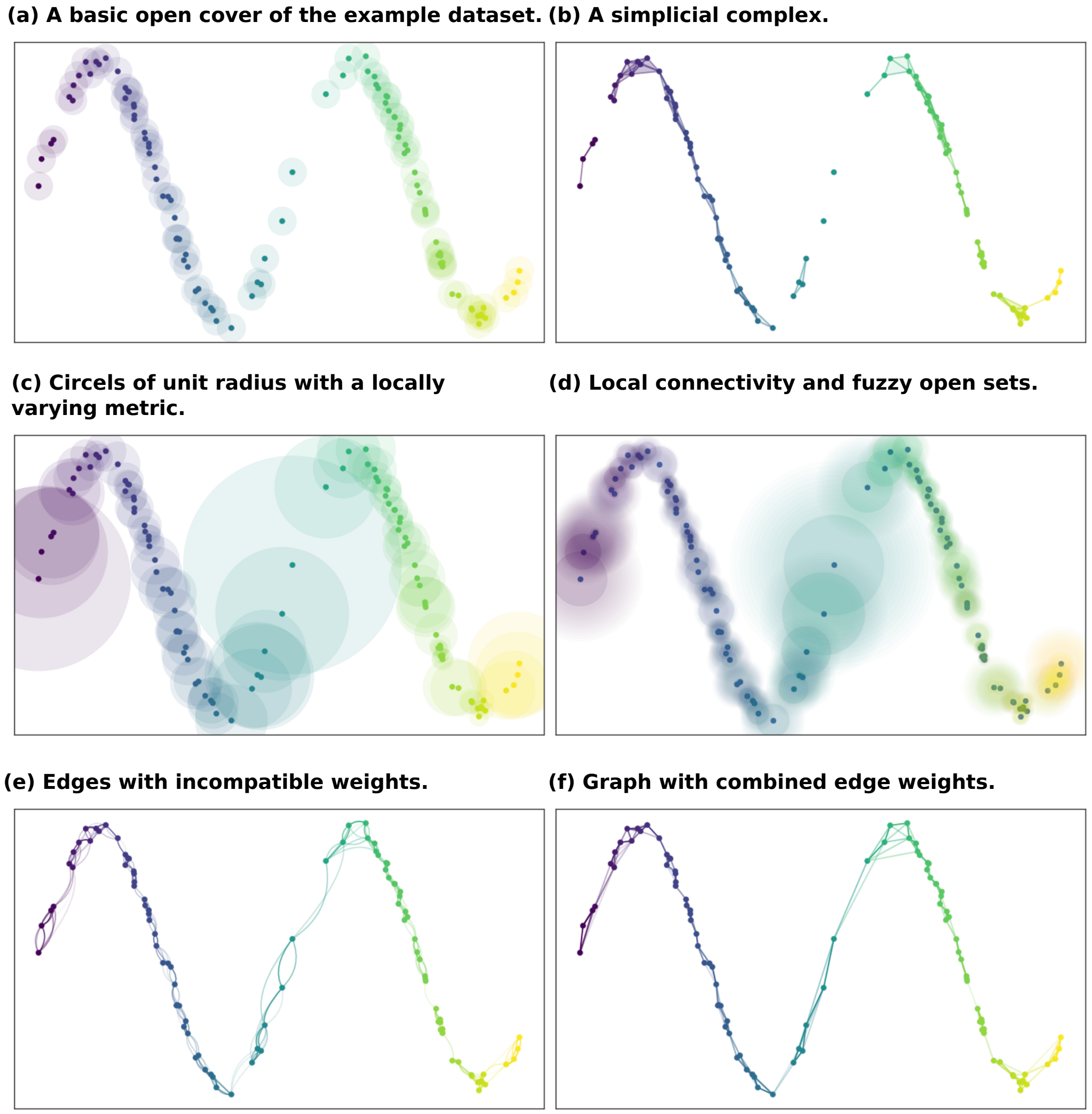
Thus, the construction of the high-dimensional graph representing the data involves the use of a fuzzy simplicial complex, which is projected onto a low-dimensional Euclidean manifold in the subsequent phase of the UMAP algorithm. Edges with higher weights correspond to a higher probability that the corresponding points remain close to each other in the resulting low-dimensional representation. Determining the optimal low-dimensional representation is facilitated by the application of stochastic gradient descent (McInnes et al., 2018).

Hyperparameter

To execute dimension reduction using UMAP, two essential parameters, namely `n_neighbors` and `min_dist`, must be set. I have kept the default settings for other parameters, including the dimensionality of the resulting low-dimensional representation, which is set to two.

The hyperparameter `n_neighbors` balance between local and global structures (McInnes et al., 2018). Its number determines the n nearest neighbors used to construct the high dimensional graph from the data. Choosing a small value preserves the local structure but looses information about the global structure of our data and vice versa. However, `n_neighbors` should not be set to a too small value as this could lead to spurious structures within the UMAP that do not represent the structure of the data (McInnes et al., 2018).

`min_dist` controls the points' density in the low-dimensional representation of the input data (McInnes et al., 2018). The lower the value the closer the points are packed together and the underlying structure of the high dimensional input data is better preserved.


(a) A basic open cover of the example dataset.

(b) A simplicial complex.

(c) Circels of unit radius with a locally varying metric.

(d) Local connectivity and fuzzy open sets.

(e) Edges with incompatible weights.

(f) Graph with combined edge weights.

Figure 4.2: The construction of a high-dimentional graph in UMAP. The example dataset used is a noisy sine wave. Each colored dot in the diagrams represents a sample. (a) Initially, each point is assigned a 0-simplex. The surrounding circles mark a fixed radius. As some of the circles do not overlap, this is an open cover of the entire dataset. (b) If these circles overlap, the 0-simplices are combined into 1-simplices and further to 2-simplices. Due to the open cover of this dataset not all points are connected. (c) To ensure that the data points are locally connected, McInnes et al. (2018) use local distances in a standard Riemannian geometry. Consequently, each circle has a unit radius and is connected to the kth nearest neighbor. (d) To ensure local connectivity, each of the circles includes at least one nearest neighbor and continues in a fuzzy cover to the kth nearest neighbor. This in turn makes the local distances compatible. (e) By using fuzzy simplices, up to two edges can connect two points. (f) Therefore, these edges are merged by combining their weights. The subfigures (a) to (f) are explained in more detail in the text. Based on McInnes (2018).

4.2 Introduction to hierarchical clustering

Clustering is something we do constantly in our daily lives, we group objects based on their similarity or dissimilarity. Most of the time we do this automatically when we classify objects according to their shape, color, or texture and recognize that, for example, an apple is an apple and not a pear, and sometimes deliberately when we sort our clothes in the closet, for example.

There are two main groups of clustering analyses, the partitional and the hierarchical approach (Ezugwu et al., 2022). The former leads to flat clusters, in the example of the closet this would mean that pants are sorted into pants and T-shirts into T-shirts. The hierarchical approach enables further subdivisions that are accessible at different levels. Thus, we further organize our T-shirts according to their color and on the next level, e.g., according to the type of neckline. Here we know the inherent structure of our "dataset", but in most real datasets this is not the case. Therefore, automatic clustering algorithms are applied to find these groups of similarity or dissimilarity in an unsupervised fashion.

In this study, I am looking for distinct clusters of signals that may correlate with the WT's operational phases, weather and wind conditions, and other anthropogenic activities. Since I do not know the dominant influences, I use hierarchical clustering to find not only the major differences but also subgroups of signals within a larger group.

Hierarchical clustering approaches can be separated into two groups: agglomerative and divisive hierarchical clustering (Fig. 4.3; Ezugwu et al., 2022). In the agglomerative approach, each data point initially forms a cluster. These are then combined step by step using a similarity or dissimilarity measure until only one cluster remains. Divisive hierarchical clustering works oppositely. In both cases, the similarity or dissimilarity is measured by a linkage metric that determines the pairwise distance. The most common linkage criteria for agglomerative clustering are single linkage, average linkage, and complete linkage (Ezugwu et al., 2022). These methods measure the distances between all points of a cluster and all points of another cluster. The single linkage measures the closest inter-cluster distance and is therefore also referred to as the nearest neighbor method. The average linkage measures the average or mean distance and the complete linkage measures the maximum inter-cluster distance. Since I will be applying agglomerative clustering, I will not go into further detail but refer to Ezugwu et al. (2022) for a more comprehensive overview of clustering algorithms.

Numerous clustering algorithms face the challenge of effectively coping with noise (McInnes and Healy, 2017). Noise is a well-known problem when analyzing seismic data. Therefore, I intend to use a hierarchical agglomerative clustering algorithm that can address noise, known as HDBSCAN.

4.2.1 HDBSCAN

HDBSCAN stands for Hierarchical Density-Based Spatial Clustering of Applications with Noise and was first introduced by Campello et al. (2013) and further developed by McInnes and Healy (2017). As the name already states, a dataset is clustered based on the density of its samples. Therefore, the density must be measured before the highest-density regions can be combined into clusters. McInnes and Healy (2017) use the analogy of islands in an ocean to describe the HDBSCAN algorithm. The islands are the dense regions within the dataset, which the algorithm should recognize as clusters and the ocean consists of noisy samples which should not be grouped with the other clusters. The sea level is therefore the threshold separating the clusters and the noisy data.

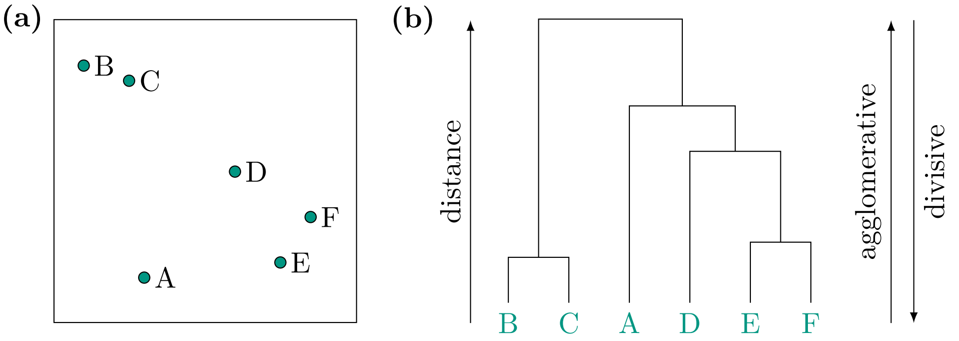


Figure 4.3: Hierarchical clustering example. a) Clustering input: Six random data points in a 2-dimensional space. b) The clustering result is summarized into a dendrogram. No linkage criterion was considered to create the dendrogram.

A measure of density

The density between values in a dataset $X = \{X_1, X_2, \ldots, X_N\}$ can be measured by the core distance ζ. A point X_i is called a core point if it contains k nearest points within a radius of ϵ, and only then is the core distance measured (Fig. 4.4; McInnes and Healy, 2017). The larger the core distance the sparser the region surrounding X_i. McInnes and Healy (2017) state that two core points X_i and X_j are *ϵ-reachable* if they are in the ϵ neighborhood of each other. They are also referred to as *density-connected* if connected directly or via other core points. The *density-connected* core points form a cluster.

To determine the distance between two points X_i and X_j McInnes and Healy (2017) introduced the mutual reachability distance,

$$d_{\text{mreach}}(X_i, X_j) = \begin{cases} \max\{\zeta(X_i), \zeta(X_j), d(X_i, X_j)\} & X_i \neq X_j \\ 0 & X_i = X_j \end{cases}, \tag{4.3}$$

with d the metric distance between the points X_i and X_j (Fig. 4.4). The mutual reachability distance pushes sparse points away from dense regions, while dense regions remain untouched. In other words, by utilizing the mutual reachability distance the algorithm ensures that the land stays land but that noisy data are pushed further into the ocean. The more k nearest points are included in the calculation of the core distance the more points are declared as noise.

The path to hierarchical clusters

Next, the hierarchical clusters of X are established through single linkage clustering, employing the metric space (X, d_{mreach}) (McInnes and Healy, 2017). Instead of utilizing the measure of minimum distance, the HDBSCAN algorithm aims to find the region of maximum density. In this approach, local density is defined by the inverse of ϵ,

$$\lambda = \frac{1}{\epsilon}. \tag{4.4}$$

This metric is utilized in constructing the cluster tree.

The condensed tree

The clustering tree of HDBSCAN is called a condensed tree (McInnes and Healy, 2017). At this point a new parameter has to be introduced, the minimum cluster size (*mcs*),

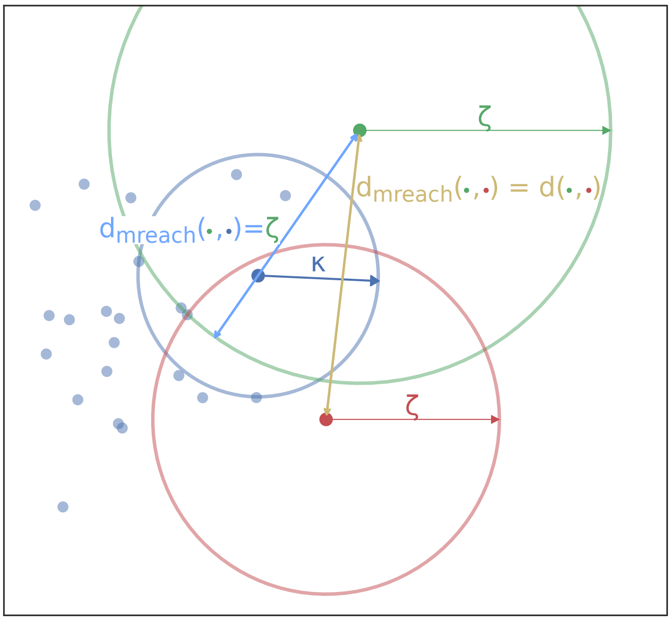

ζ
$d_{\text{mreach}}(\cdot,\cdot) = d(\cdot,\cdot)$
$d_{\text{mreach}}(\cdot,\cdot)=\zeta$
K
ζ

Figure 4.4: Mutual reachability distance. The symbol ζ represents the core distance, which is the radius of the circle surrounding a core point and enclosing k nearest points. In this instance, k is set to five. The core points are denoted in green, blue, and red. The mutual reachability distance d_{mreach} from blue to green is defined as the core distance of the green point, whereas between red and green, d_{mreach} equals the metric distance between the respective points. Adapted from McInnes et al. (2016).

which determines the number of points a cluster C_i has to have to be still a cluster. The condensed tree has the shape of an inverted tree and is constructed with the metric λ, starting at the root of the tree (Fig. 4.5).

In the first step, the parent cluster is the root. By increasing the value of λ, the parent cluster will eventually be split into two child clusters. In the case, that one of the children is smaller than the defined *mcs* it is declared a spurious split and this child falls out of the cluster (Fig. 4.5; McInnes and Healy, 2017). Hence, the root becomes smaller and smaller until a "true" split occurs where both of the children are larger than *mcs*. In this case, each of the children is now a parent cluster and the same process starts again (Fig. 4.5).

Once the last child smaller *mcs* drops out and no clusters larger than *mcs* are left, the condensed tree is constructed. Nevertheless, the main clusters still need to be chosen. For this, the measure of cluster stability σ is introduced (McInnes and Healy, 2017).

Selecting most stable clusters

The stability of cluster C_i is defined by

$$\sigma(C_i) = \sum_{X_j \in C_i} (\lambda_{max,C_i}(X_j) - \lambda_{min,C_i}(X_j)). \tag{4.5}$$

A point X_j in the cluster C_i first appears in the cluster at $\lambda_{min,C_i}(X_j)$, which also marks the birth of this cluster, and leaves C_i at $\lambda_{max,C_i}(X_j)$ in the event of spurious or "true" split (Fig. 4.5). The sum of the lifetimes of each point within cluster C_i describes its stability.

Suppose the stability of the parent cluster exceeds the combined stability of its child clusters. It is then labeled as the main cluster and the subsequent child clusters are no longer considered as independent clusters. Conversely, suppose that the combined stability

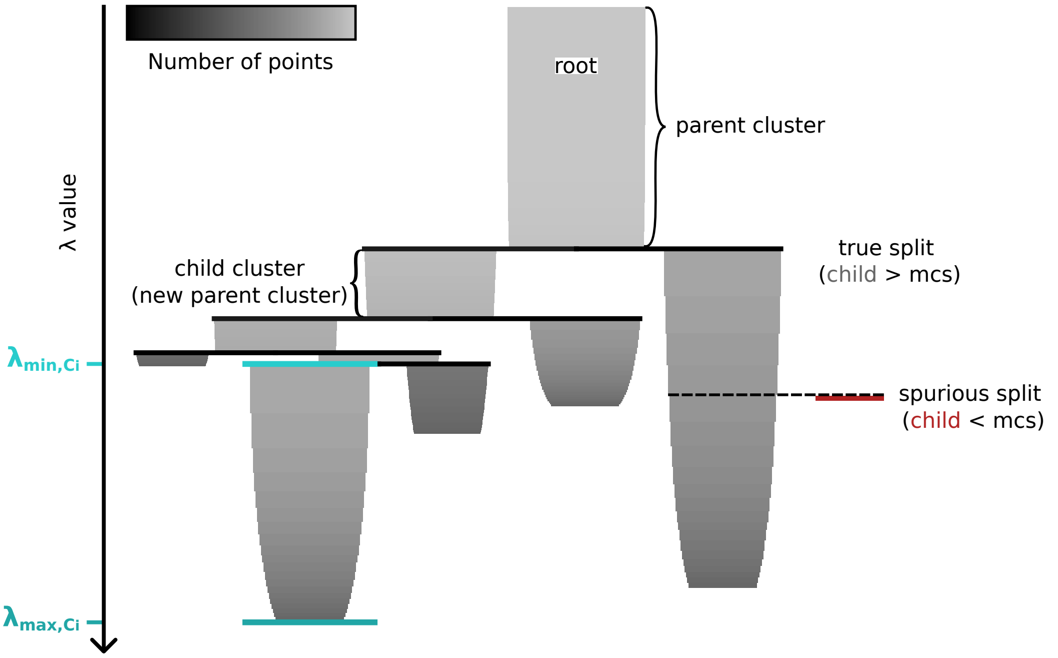


Figure 4.5: Example of a hierarchical clustering result from HDBSCAN, shown as a condensed tree. Detailed explanations of the highlighted terms can be found in the text.

of the child clusters exceeds that of the parent cluster. In this case, the analysis continues at the next level of the condensed tree, with the previous child clusters becoming parent clusters. However, the order of the cluster labels is arbitrary.

Hyperparameters

The major hyperparameters of HDBSCAN are `min_cluster_size` and `min_samples`, alongside `cluster_selection_epsilon` and `alpha` (McInnes et al., 2017). According to McInnes et al. (2017), it is recommended not to modify the latter. Additionally, the parameter `cluster_selection_epsilon` prevents clusters from breaking into smaller, micro-clusters. During the UMAP parameter testing, we observed no appearance of micro-clusters (Fig. 4.8), therefore, I will not discuss this parameter further.

More intuitive is the hyperparameter `min_cluster_size`, as described in Section 4.2.1 the *mcs* regulates how many samples a cluster must have to be declared a cluster. Clusters that are smaller than the `min_cluster_size` are declared as noise (McInnes et al., 2017). A small value focuses on the local structure while a large value focuses more on the global structure.

The hyperparameter `min_samples` controls the number of k-neighbors used to calculate the core-distance κ and with it the mutual reachability distance, Equation 4.3. Thus, a larger `min_samples` results in more points being declared as noise, and thus the clusters are defined by denser regions, ensuring more conservative clustering (McInnes and Healy, 2017). If small details within the dataset are important, this parameter must be set to a small value. In default, `min_samples` is set to the same value as `min_cluster_size`.

4.3 UMAP application

With the theory outlined, I will now discuss the application of UMAP to the scattering coefficient matrix S (Fig. 4.1). The inputs for the scattering network were the continuous three-component seismograms recorded during the Inter-Wind project at station IW08B (Fig. 2.1 and Section 2.2.4). The seismograms are split into sliding windows, with each component being transformed by the two-layer scattering network. As shown in Figure 3.2, the resulting scattering coefficients are concatenated to the scattering coefficient matrix S. The first step in Figure 4.1 shows the reduction of S to two UMAP variables discussed in this section.

4.3.1 Parameter tuning

This section describes the tuning process for the two main input parameters `n_neighborss` and `min_dist` of the UMAP algorithm (Section 4.1.2). Figure 4.6 shows the effects of these hyperparameters on the first- and second-order scattering coefficients of the IW08B data (Figs. 3.18, C.2 and C.3). The original 2175-dimensional scattering coefficient matrix is reduced to two UMAP variables (Fig. 4.1), with each point in the so-called UMAP atlas corresponding to a sliding window.

The rows in Figure 4.6 show the variation of the hyperparameter `min_dist` using values of 0.1, 0.2 and 0.4, the columns show different values of `n_neighborss` using 40, 80 and 160. I tested further parameter pairs. This set was chosen as it sufficiently shows the effects of the individual parameters.

Examining the rows of Figure 4.6 the effect of manipulating the `min_dist` parameter can be observed. Specifically, as this parameter value increases, the low-dimensional representation of the dataset expands. In contrast, alterations to the `n_neighborss` parameter exhibit less pronounced effects. Nevertheless, the points in the UMAP atlas draw nearer to one another as `n_neighborss` increases.

However, in the end, I want to cluster the data (Fig. 4.1), and with Figure 4.6 the optimal selection of parameter configurations is impossible. Therefore, I applied HDBSCAN clustering to gauge the sensitivity of the UMAP hyperparameters towards clustering, utilizing a `min_cluster_size` of 100 and leaving the other hyperparameters at their default values (Section 4.2.1). To facilitate the discussion of the UMAP atlas, I have marked the individual areas that can be delimited with the naked eye with Roman numerals (Fig. 4.7).

Effect of the UMAP parameter to HDBSCAN

By visually inspecting the data represented in Figures 4.6 and 4.7, one can discern the presence of up to seven distinct clusters. Intuitively, it would be expected that a smaller value of `min_dist`, which results in denser regions within the UMAP atlas, would yield a greater number of clusters. However, upon observing the rows in Figure 4.8 no discernible correlation of this nature is evident. This also applies to the influence of `n_neighborss`.

This lack of direct correlation can be attributed to the fact that UMAP is a non-linear dimension reduction method. Consequently, alterations in parameter values do not induce linear changes in the distribution of points within their low-dimensional representation.

Comparing the clustering results in Figure 4.8 with the labeled areas in Figure 4.7, the UMAP hyperparameters, which correspond to a `min_dist` set to 0.2 and `n_neighborss` set to 80, matches best with the labeled areas and are applied in the following.

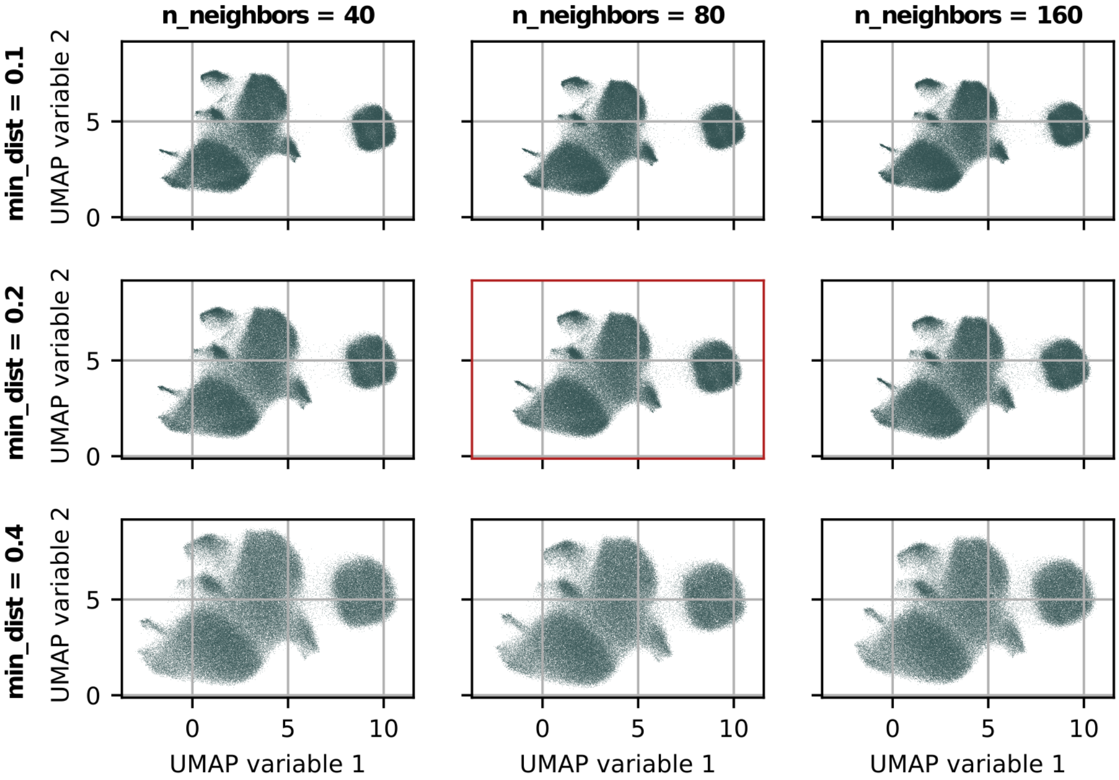


Figure 4.6: UMAP parameter test. The rows show the variation of the hyperparameter `min_dist`, while the columns show `n_neighbors`. Both, the first- and second-order scattering coefficients of the IW08B dataset were used as input. The result of the selected parameter set is highlighted in red.

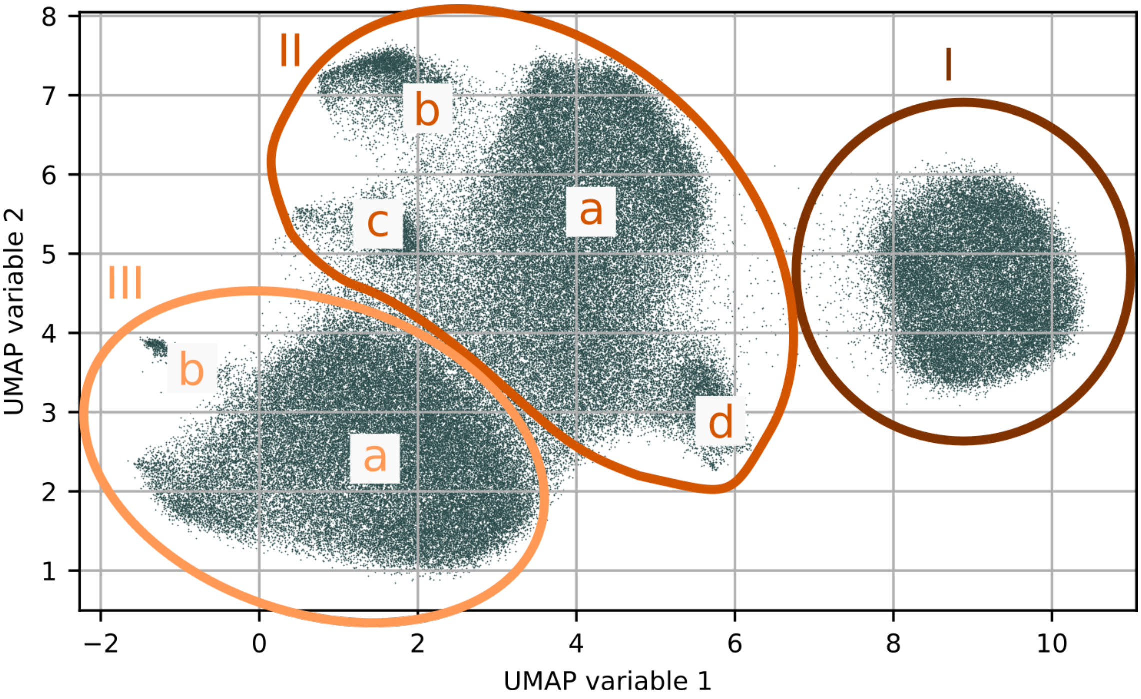


Figure 4.7: UMAP atlas of the scattering coefficient matrix of IW08B with annotated regions separable with the naked eye. Roman numerals denote the three primary regions, while smaller subdivisions are labeled with lowercase letters. The hyperparameter `min_dist` was set to 0.2 and `n_neighbors` to 80.

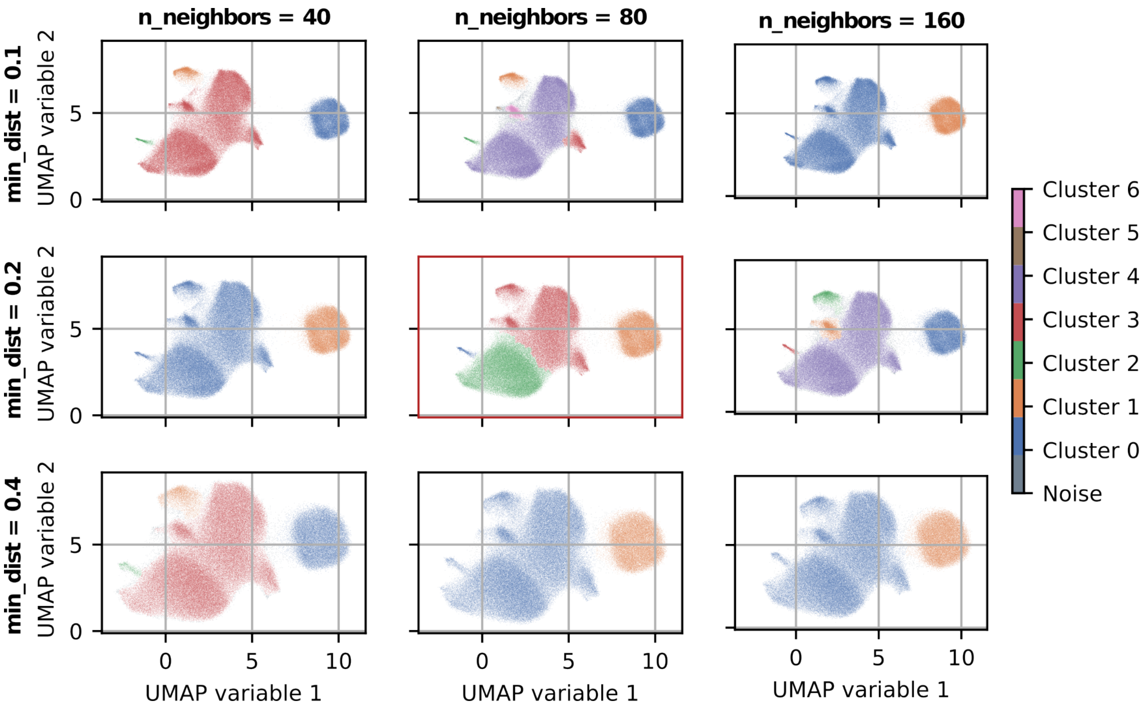


Figure 4.8: UMAP parameter test of the scattering coefficient matrix of IW08B color coded with clusters calculated with HDBSCAN. The rows show the variation of the hyperparameter `min_dist`, while the columns of `n_neighborss`. As input both, first- and second-order scattering coefficients were used. The result of the selected parameter set is highlighted in red.

4.3.2 Correlations with operational data

Initially, I investigated whether the regions within the UMAP atlas exhibit daily or hourly variations. As detailed in the appendix Figure D.1, the points within the UMAP atlas display no discernible temporal patterns. Subsequently, I color-coded the UMAP atlas with the operational data of *WT* 3 presented in Section 2.2.5 and in Figures 2.5, A.1 and A.2.

The UMAP areas marked in Figure 4.7 correlate with the wind speed and the rotation rate and are partially influenced by the wind direction (Fig. 4.9 and Table 4.1). The region *I* correlates with the times when *WT 3* is at a standstill and includes all recorded wind speeds (Table 4.1). It is well delimited from the areas *II* and *III*, which correlate with the time windows in which *WT 3* is in operation. The time windows related to different operating states, partial load, noise-reduced operation, and full load, merge into one another (Table 4.1 and Fig. D.4). The wind directions, on the other hand, show no clear correlation, except area *IIIb*, which only contains time windows with wind from E and SE. This is the only subarea directly related to meteorological conditions. The subareas *IIb* to *d*, on the other hand, show no clear indication of why they are separated from the main body indicated as *IIa*.

The representation of the individual UMAP variables in Figures 4.10a and D.2 shows that variable 1 inversely correlates with the rotation rate. The same is partly true for UMAP variable 2, although other unspecified effects affect this variable as well (Fig. 4.10b and D.3).

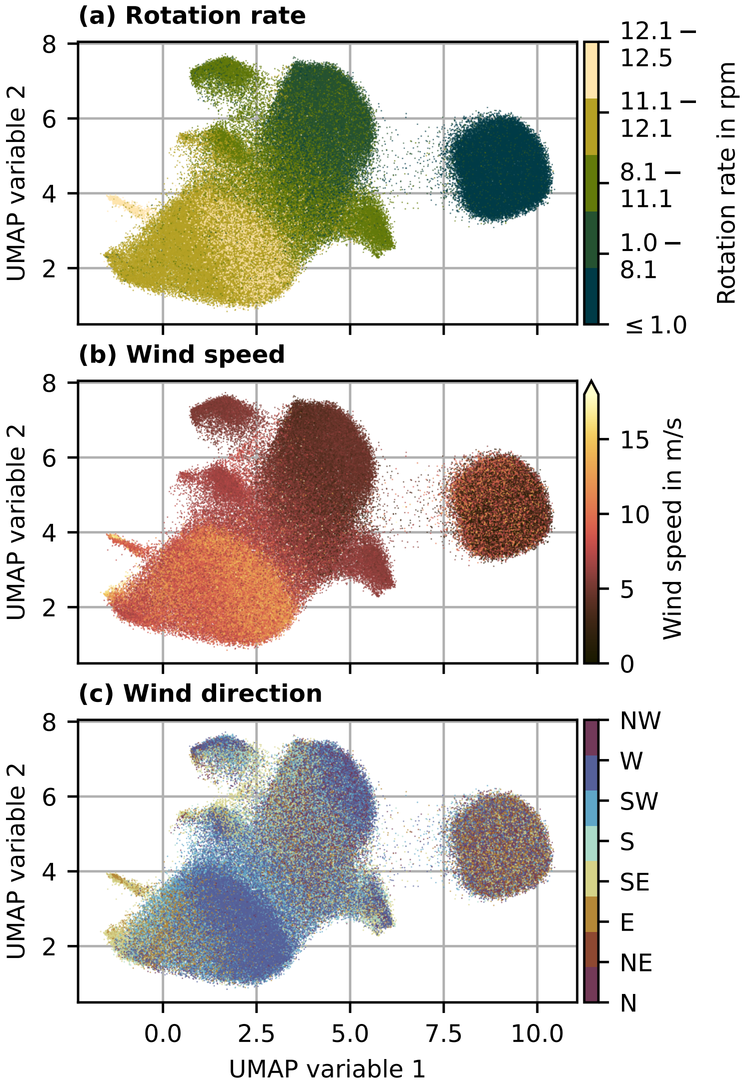


Figure 4.9: UMAP of the scattering coefficient matrix of IW08B color-coded with (a) rotation rate, (b) wind speed, and (c) direction. The hyperparameter `min_dist` was set to 0.2 and `n_neighbors` to 80.

Table 4.1: Correlation of wind speed and rotation rate with the UMAP areas shown in Figure 4.7. The WT loads represent the different operational stages of the *WT 3*, 0: WT not in operation or rotation rates ≤ 1 rpm, 1: rotation rates between 1 rpm and 8.1 rpm (partial load), 2: rotation rates between 8.1 rpm and 11.1 rpm (full load with lower noise-reduced mode), 3: rotation rates between 1.1 rpm and 12.1 rpm (full load with upper noise-reduced mode), 4: rotation rates between 12.1 rpm and 12.5 rpm (full load). See also Fig. D.4.

UMAP areas	*I*	*II*				*III*	
		a	*b*	*c*	*d*	*a*	*b*
Wind speed in $\mathrm{m\,s^{-1}}$	0 to 23	2 to 6	4 to 6	6 to 8	4 to 8	6 to > 14	
WT loads	0	1, 2	1, 2	2, 3	2	3, 4	4

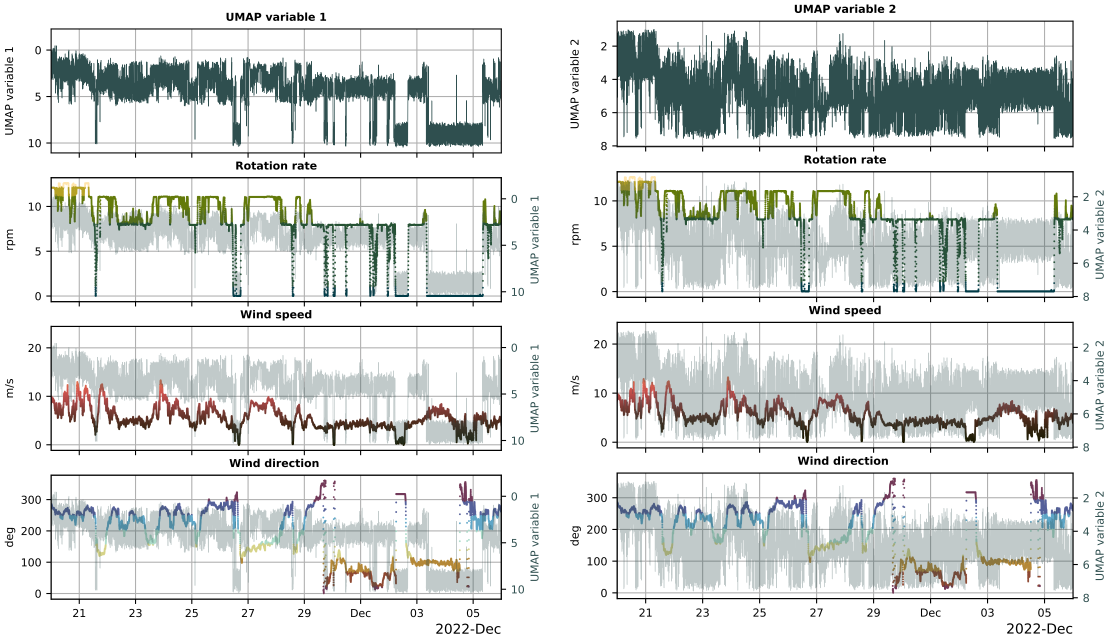


Figure 4.10: UMAP variables of the scattering coefficient matrix of IW08B in comparison to rotation rate, wind speed, and direction. Zoom to time range between November 20, and December 6, 2022. The UMAP variables are displayed inversely. (a) UMAP variable 1, (b) UMAP variable 2. The respective color scales are identical to those in Figure 4.9. The entire time range is plotted in the appendix in Figs. D.2 and D.3.

4.3.3 Influence of the UMAP input

In Sections 4.3.1 and 4.3.2, I analyzed the complete scattering coefficient matrix S as input for dimension reduction with UMAP. In this section, I further investigate the impact of incorporating both first- (S_1) and second-order scattering coefficients (S_2) separately, contrasting their influence with that of the averaged time windows themselves, which Anden and Mallat (2014) called zeroth-order scattering coefficient (S_0). Using the same UMAP hyperparameter as above, `min_dist` set to 0.2 and `n_neighborss` set to 80.

The output UMAPs of these four approaches are summarized in Figure 4.11 color-coded with the rotation rate of *WT 3*. In all these UMAP atlases, areas of different operating phases are distinguishable but overlap to varying degrees. In all the UMAP atlases but of S_0, the time windows in which *WT 3* is not in operation are separated from the ones in which the WT is in operation. Due to the poorer distribution of the time windows in their lower dimensional representation, S_0 as UMAP input is not further analyzed. The complete scattering coefficient matrix is denoted by $S_{(1,2)}$ in this section.

Comparing the UMAP atlases of S_1 and $S_{(1,2)}$ shows that the points of the former are more spread out and therefore less dense. Comparing the correlation with the wind direction, the individual wind directions are better separated in the UMAP atlas of S_1 than in $S_{(1,2)}$ (Fig. 4.12). S_1 contains amplitude and primary frequency information, while S_2 shows amplitude modulation and frequency intervals, among other things (Section 3.4). Figure 4.12 shows that the wind direction-dependent signal sources are most likely primarily present in S_1. However, S_1 has only 75 dimensions, which is only 3.45 % of the dimension in $S_{(1,2)}$. Therefore, the wind direction effect has only a secondary effect on the UMAP of the entire scattering coefficient matrix $S_{(1,2)}$. To determine how well the UMAP atlas of S_1 is clustered I performed a test with HDBSCAN using the same hyperparameter as in Figure 4.8. The UMAP atlas of S_1 is separated into two large and one very small cluster (Fig. D.6a). The two large clusters only separate the areas corresponding to whether *WT 3* is in operation or is not running.

The UMAP atlases of S_2, and $S_{(1,2)}$ are not distinguishable in Figure 4.11, the difference only becomes apparent when clustering these UMAPs (Fig. 4.8, red rectangle, and Fig. D.6b). While the clustering of the UMAP atlas of $S_{(1,2)}$ leads to the separation of areas *I*, *II*, *IIIa*, and *IIIb*, the clustering of the UMAP atlas S_2, combines area *IIa* and *IIIa*, while labeling area *I*, *IIb*, the upper part of *IIc*, *IIIb* as single clusters (Fig. 4.7).

Thus, I have shown that only the UMAP of the entire scattering coefficient matrix of IW08B leads to the desired clustering result most similar to the areas in Figure 4.7. However, this cannot be a definitive statement, as I did not test different UMAP parameters for S_1 and S_2, which is beyond the scope of this study.

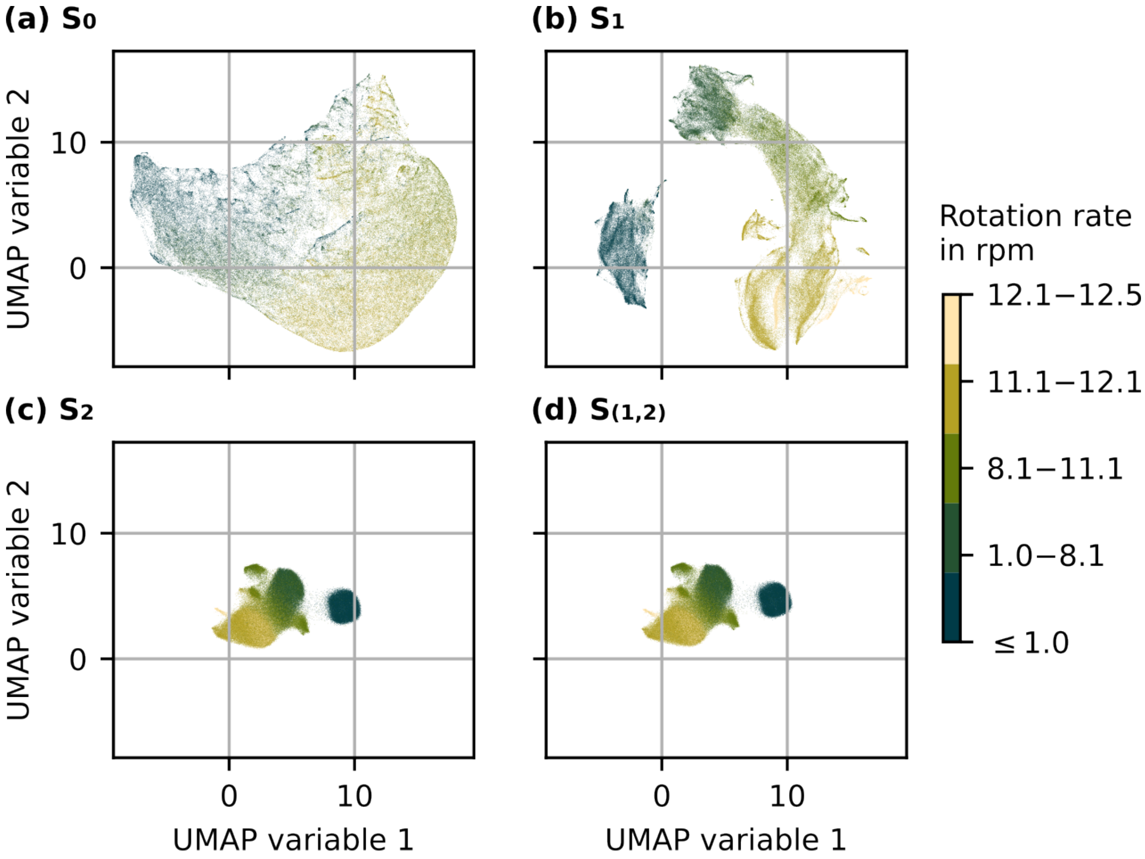


Figure 4.11: Impact of the different components of the scattering coefficient matrix of IW08B as input to UMAP. (a) averaged time windows, S_0, (b) first-order scattering coefficients, S_1, (c) seconds-order scattering coefficients, S_2, and (d) first- and second-order scattering coefficients, $S_{(1,2)}$.

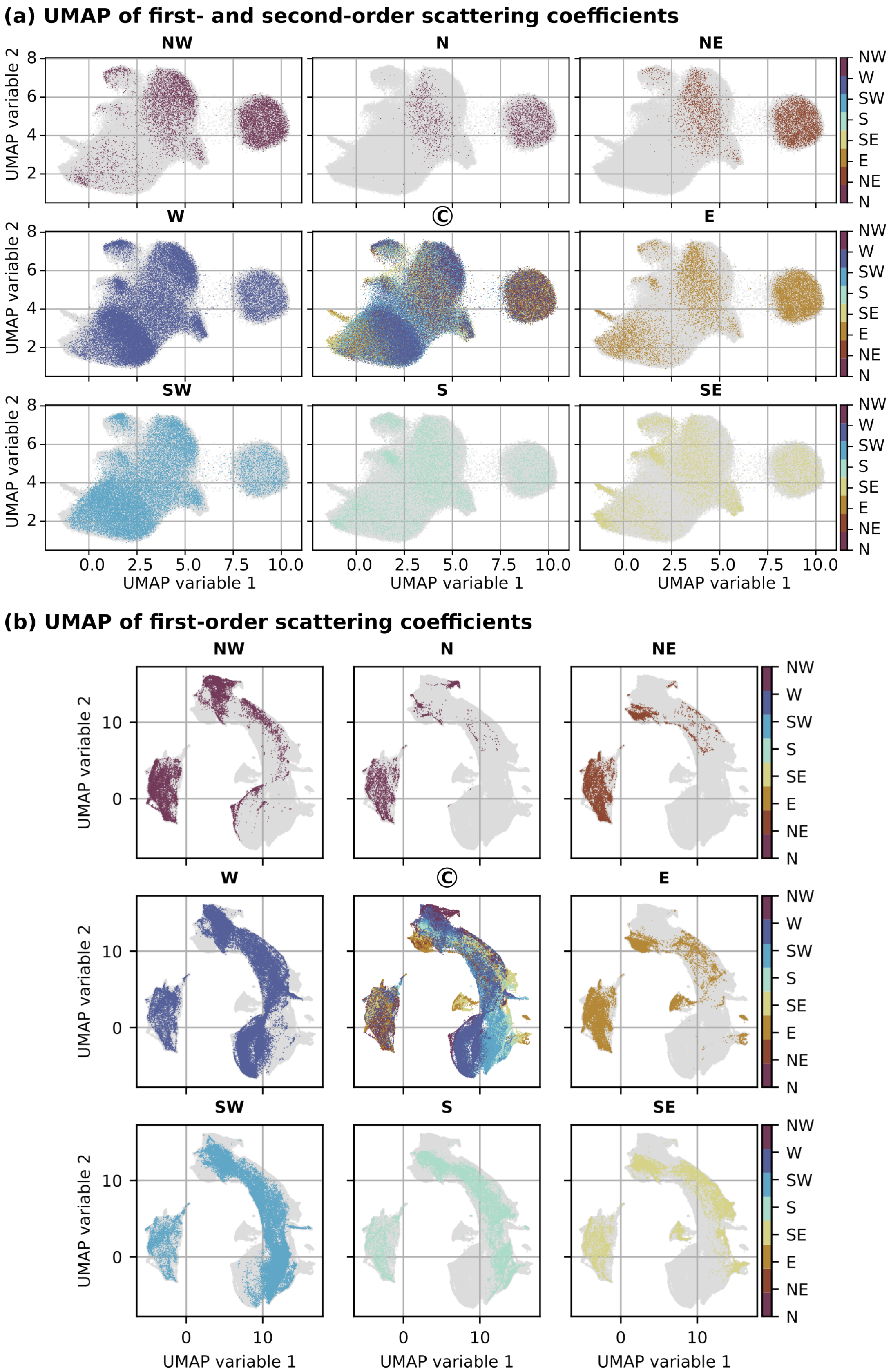


Figure 4.12: Influence of the wind direction on the time windows in the UMAP atlas of the scattering coefficient matrix of IW08B. The center plot, labeled **C**, shows the sum of the wind directions recorded during the IW08 experiment. The outer plots highlight the parts of the UMAP atlas that correlate with the respective cardinal direction. (a) showing the UMAP atlases of the first- and second-order scattering coefficients and (b) of the first-order scattering coefficients.

4.4 HDBSCAN application

I applied UMAP to reduce the high-dimensional scattering coefficient matrix of experiment IW08B to two UMAP variables. Subsequently, I identified different areas within the UMAP atlas (Fig. 4.7), that I aim to separate as clusters. To do this, I applied the clustering algorithm HDBSCAN (Fig. 4.1). In this section, I will first discuss the choice of hyperparameters for clustering the UMAP variables and describe the resulting clusters. I will continue by presenting the clustering of the first-order scattering coefficient matrix.

4.4.1 Parameter tuning

I tested several combinations of the HDBSCAN hyperparameters `min_sample` and `min_cluster_size` to find the most suitable set for the UMAP of the IW08B scattering coefficient matrix S (Section 4.2.1 and Figs. 4.13 and 4.14). I varied the parameter `min_sample` between 25, 50, 100, and 200, respectively. The clustering result did not change when higher and lower values were used. The same applies to the parameter `min_cluster_size`. Here I present values between 100 and 1000. However, this dataset is dominantly sensitive to the parameter `min_sample`, while `min_cluster_size` shows only a minor effect.

Clusters and UMAP areas

In the following, I will evaluate the results of the HDBSCAN parameter test shown in Figure 4.13, with the goal that the resulting clusters resemble the UMAP areas shown in Figure 4.7 as closely as possible. Additionally, I have plotted in Figure 4.14 the number of points in the individual clusters and the data points labeled as noise. I will not explain every visible feature, but highlight the most prominent ones. The cluster IDs are indicated with bold letters.

All parameter pairs have in common that region *I* is identified as a separate cluster. With the parameter combinations shown, regions *II* and *III* are either combined into a single cluster or separated into up to four clusters.

Setting `min_sample` to 200 and `min_cluster_size` to the values shown above results in the formation of two clusters (Figs. 4.13 and 4.14). The same occurs when `min_sample` is set to 25 and `min_cluster_size` is set to 100. While the former labels the domains *II* and *III* as cluster **0** and the domain *I* as cluster **1**, the latter leads to the opposite labeling. This is not correlated with the formation of the branches, as the order of the cluster labels is arbitrary (Section 4.2.1 and Fig. E.1; L. McInnes, personal communication, May 07, 2024).

With a `min_sample` set to 50, the smallest clusters and the highest number of clusters occur. Despite this, the *IIa* and *IIIa* areas are still combined in one cluster. Up to a `min_cluster_size` of 200, *IIIb* is further separated into two clusters, **0** and **1**. With higher `min_cluster_size` *IIIb* is combined in a single cluster with the ID **0**. In case *IIIb* is split into two clusters, cluster **0** has 356 inhabitants and cluster **1** has 634. This explains why we cannot observe the division of the area *IIIb* with a `min_cluster_size` greater than 356.

Only with a `min_sample` value of 100 the areas *IIa* and *IIIa* are split into separate clusters. The clustering results are almost identical for different `min_cluster_size` values. In addition to *IIa* and *IIIa*, the areas *IIIb* and *I* are clustered. However, cluster **0**, the cluster corresponding to *IIIb*, has only 989 inhabitants and from a `min_cluster_size` of 1000 it is classified as noise. As there are around 1000 points in this region, only a few points need to be declared as noise for this region to drop out of a spurious split (Fig. 4.5). Since the

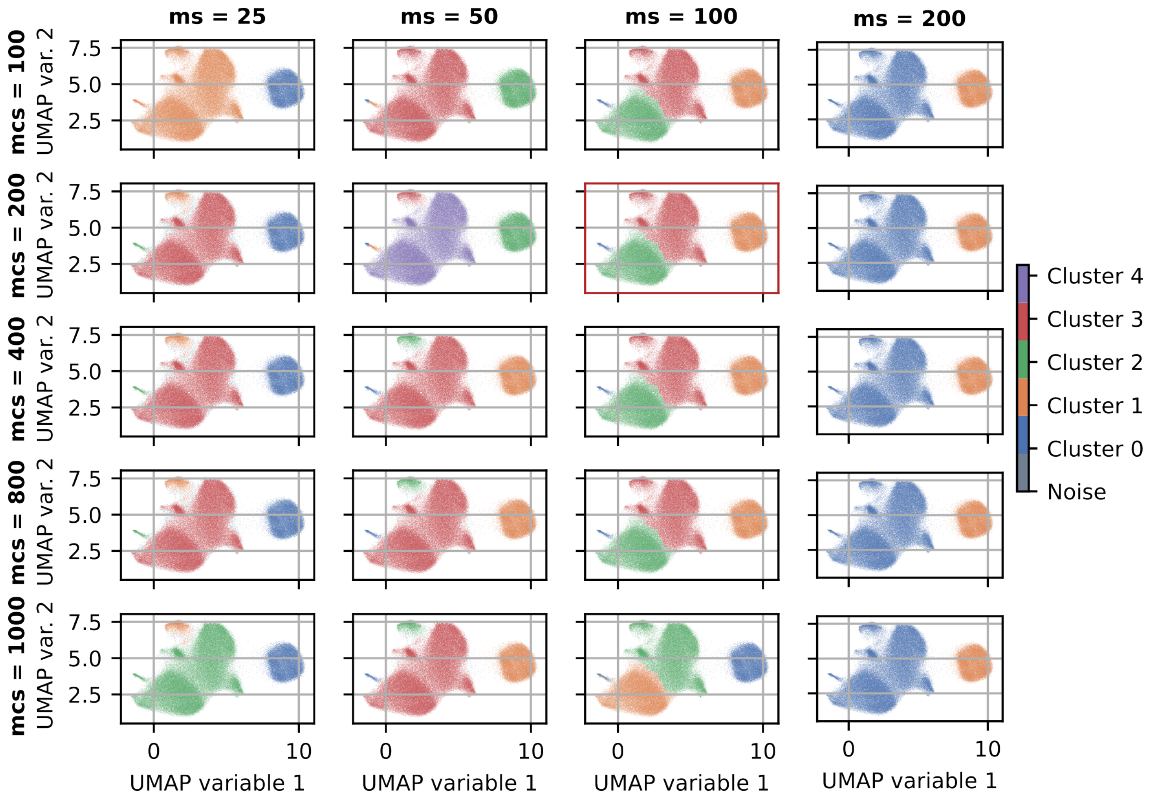


Figure 4.13: HDBSCAN parameter test using the the UMAP of the first and second-order scattering coefficients with `min_dist` set to 0.2 and `n_neighbors` set to 80 as input. The rows show the variation of the HDBSCAN hyperparameter `min_cluster_size` (mcs) and the columns of `min_samples` (ms). The resulting cluster labels are used as color-code in the UMAP. The result of the selected parameter set (ms=100, mcs=200) is highlighted in red.

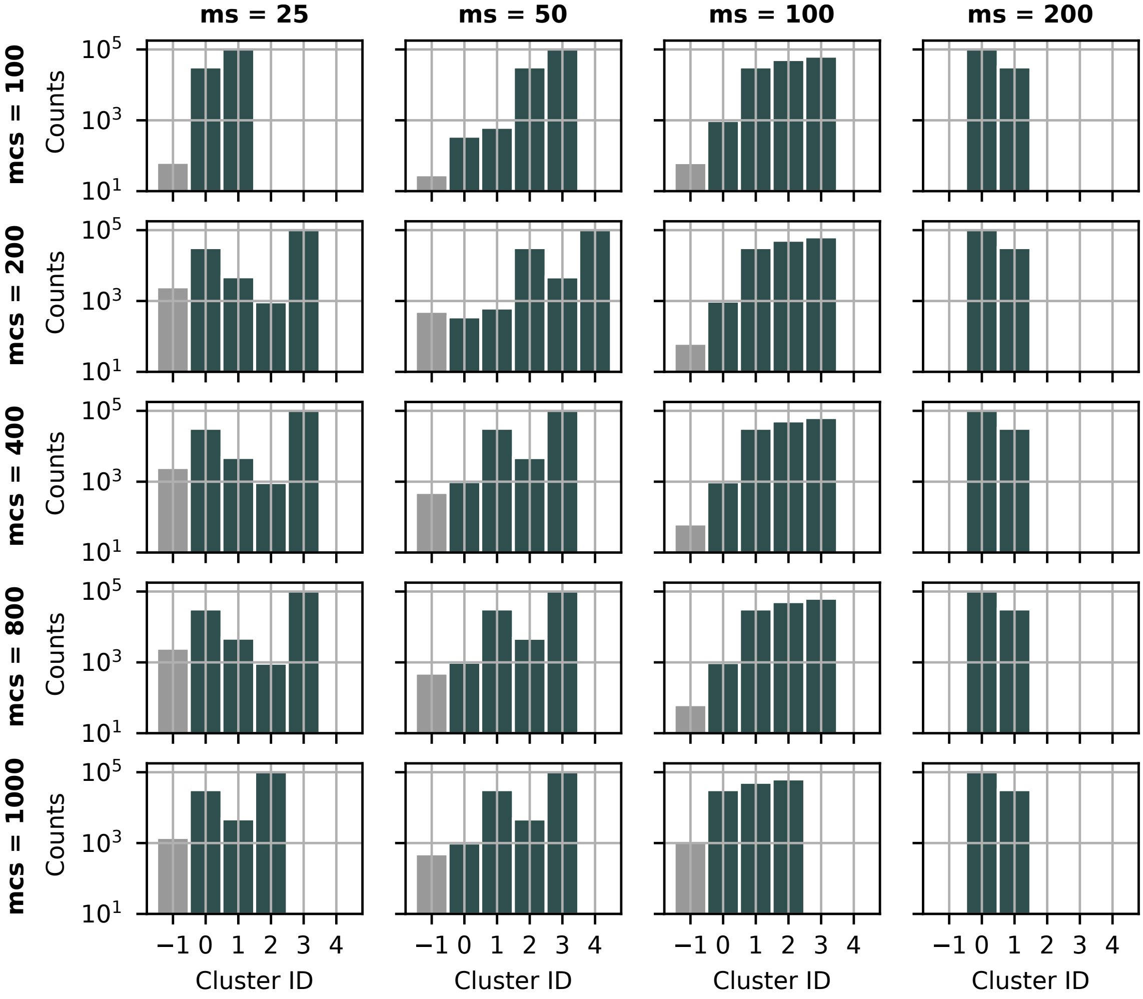


Figure 4.14: HDBSCAN parameter test using the UMAP of the first- and second-order scattering coefficients with `min_dist` set to 0.2 and `n_neighbors` set to 80 as input. The rows show the variation of the HDBSCAN hyperparameter `min_cluster_size` (mcs) and the columns of `min_samples` (ms). The histograms show the number of points filling each cluster of the tested HDBSCAN parameters. The cluster ID **-1** contains the points declared as noise and the bars are colored in gray.

clustering results of using `min_sample` of 100 is closest to the identified primary regions in Figure 4.7, I will further investigate the effects of `min_cluster_size` and thereby select the final parameter set.

Condensed tree

The clustering results shown in Figure 4.15a to d share the same patterns, cluster **0** (blue) corresponds to the UMAP area *IIIb*, cluster **1** (orange) with *I*, cluster **2** (green) with region *II*, and cluster **3** (red) with *IIIa*. In Figure 4.15e only three clusters were formed as outlined above. A comparison of the Figure 4.15e with the other subfigures shows that the splitting responsible for the cluster formation corresponding to region *IIIb* does not occur. In Figure 4.15a to d this cluster emerges from the first split of the root. This observation suggests that the samples from *IIIb* were disregarded during a spurious split and classified as noise.

The condensed tree plots vary in their branch thickness, which is related to the parameter `min_cluster_size`. Furthermore, fewer subclusters are formed with a larger `min_cluster_size`. A `min_cluster_size` of 100 or 200 exhibits significant branching of the condensed trees (Fig. 4.15a and b), compared to a `min_cluster_size` of 400 and 800 (Fig. 4.15c and d). However, I am primarily interested in the main clusters for this study, and hence, I will continue with the hyperparameter `min_samples` set to 100 and `min_cluster_size` to 200.

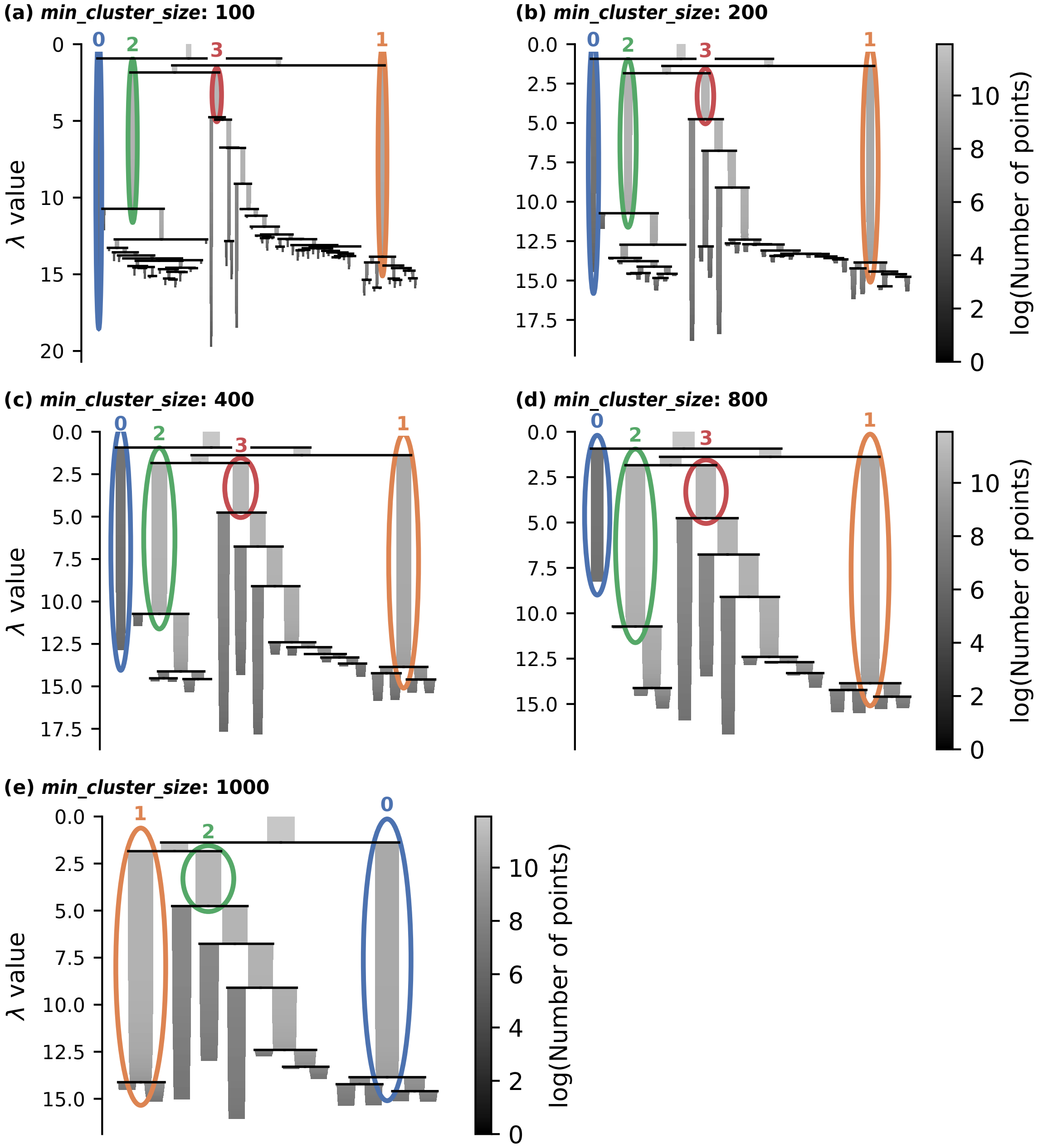
4.4.2 HDBSCAN cluster statistics

Now that we have set the hyperparameters for HDBSCAN, we can explore the properties of the individual clusters. Compared to Section 4.3.1, where I correlate the UMAP areas identified by the naked eye with the operational data of *WT 3*, I will evaluate the clusters separated with HDBSCAN now. Figure 4.16a shows the development of the individual clusters, and Figure 4.16b their distribution in terms of rotation rate, wind speed, and direction, recorded at *WT 3*. Both figures are consistent with the correlation of the UMAP domain summarized in Table 4.1 and Fig. 4.12a. Then why is it not enough to simply analyze the UMAP atlas?

Previously, I had to filter the operational data to see the correspondence, e.g., between WT loads and certain UMAP areas, since the different loads blend into each other in the UMAP atlas (Figs. 4.12 and D.4). To obtain a comprehensive overview of the activities in the individual clusters, it is only necessary to filter by the corresponding label of a cluster. This facilitates the analysis of overlapping regions as Figure 4.16b shows.

Furthermore, we can determine how the individual clusters relate to the scattering coefficients and whether it is possible, e.g., to determine the rotation rate from the clusters without the operational data. Figure 4.17 shows the first-order mean scattering coefficients S_1 of each cluster. The center frequencies of the first-order filter bank corresponding to the eigenmodes of *WT 3* and the 32nd multiple of its BPFs are marked. In Table 4.2, I have summarized how well the individual peaks are visible in the individual clusters.

Cluster **0** shows prominent peaks at the center frequency corresponding to 1.2 Hz and the 32nd multiple of the BPF at 12.5 rpm, while cluster **1** has no distinct maxima. Cluster **2** shows distinct maxima at the eigenmodes 1.2 Hz, 3.6 Hz, and 8.2 Hz and the 32nd multiple of the BPF corresponding to a rotation rate of 11 rpm or 12 rpm. These eigenmodes can likewise be identified in cluster **3**. However, a broad maximum extends over the possible peaks corresponding to the rotation rates of 11 rpm, 12 rpm and 12.5 rpm. This is


(a) ***min_cluster_size*****: 100**

(b) ***min_cluster_size*****: 200**

(c) ***min_cluster_size*****: 400**

(d) ***min_cluster_size*****: 800**

(e) ***min_cluster_size*****: 1000**

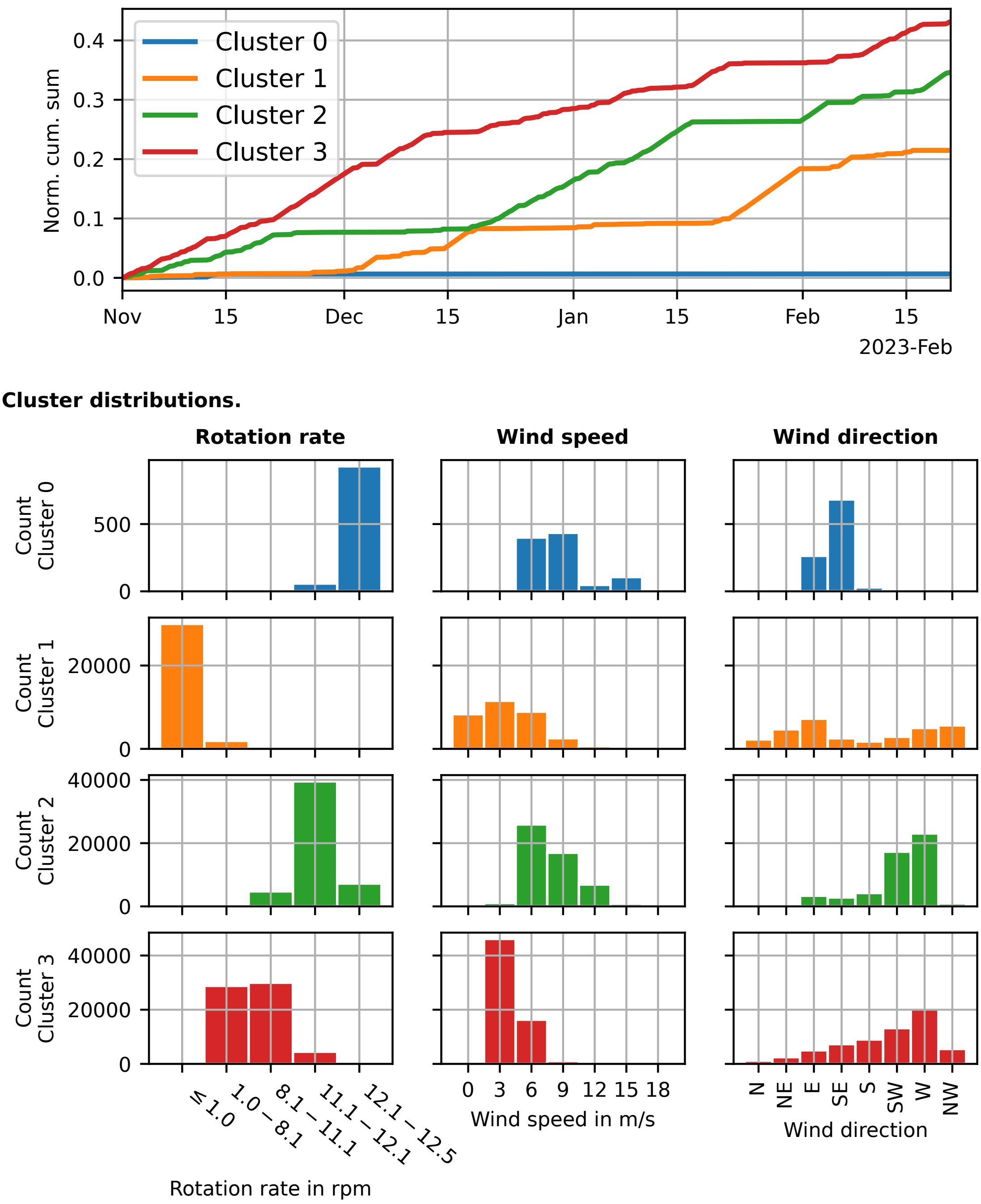
Figure 4.15: Condensed tree plot of the clustering results using `min_sample` set to 100 and varying values of `min_cluster_size`. The circled branches indicate cluster **0** to **3**.

Table 4.2: Prominence of the WT related peaks evaluated for the clusters shown in Figure 4.17. Prominent maxima are marked with ++, weak maxima with a +, and a - if there is no distinct maximum. The center frequencies of the first-order filterbank corresponding to the eigenmodes of the *WT 3* are marked with capital roman numbers and the 32 multiple of the BPF of the different operational modes of the WT are highlighted with small roman numbers. Listed are the center frequencies and in parentheses the corresponding frequencies emitted by *WT 3*: I - 1.15 Hz (1.2 Hz), II - 3.48 Hz (3.6 Hz), III - 8.0 Hz (8.2 Hz), IV - 10.56 Hz (11.2 Hz), i - 12.13 Hz (12.64 Hz), ii - 18.38 Hz (17.6 Hz and 19.2 Hz), iii - 21.11 Hz (20.0 Hz).

	I	II	III	IV	i	ii	iii
Cluster 0	++	-	-	-	-	-	++
Cluster 1	-	-	-	-	-	-	-
Cluster 2	++	++	+	+	-	++	-
Cluster 3	++	++	+	-	-	-	-

attributable to the cluster **3** being active when *WT 3* is operated at partial load or in one of the noise-reduced modes.

Hence, in case no operational data is available, the UMAP atlas and its cluster only partially present a possibility to identify different operational modes of *WT 3*. To further investigate the single UMAP areas one could have a look into the subclusters. However, the different WT loads blend in the UMAP atlas into each other, and therefore, it will not be possible to perfectly separate them. Except for the *IIIb* area, there is no correlation between the UMAP atlas and the wind direction (Fig. 4.12a). However, when I solely used S_1 as UMAP input, I observe not only the separation of the WT loads (Fig. D.5), but also a clearer division of the wind direction (Fig. 4.12b). Nevertheless, they still blend into each other making clustering difficult. This has given me the idea of clustering S_1 directly, because with 75 dimensions it is within the capabilities of HDBSCAN. The entire scattering coefficient matrix is too highly dimensional, as HDBSCAN can only process data with up to 50 to 100 dimensions (McInnes et al., 2017).



(a) Normalized cumulative sum of cluster detections.

(b) Cluster distributions.

Figure 4.16: Clustering statistics using the hyperparamerer `min_sample` set to 100 and `min_cluster_size` to 200. (a) The normalized cumulative sum of cluster detections plotted against the time. (b) Cluster distribution (colors as in (a)) with respect to the rotation rate, wind speed, and direction recorded at *WT 3*.

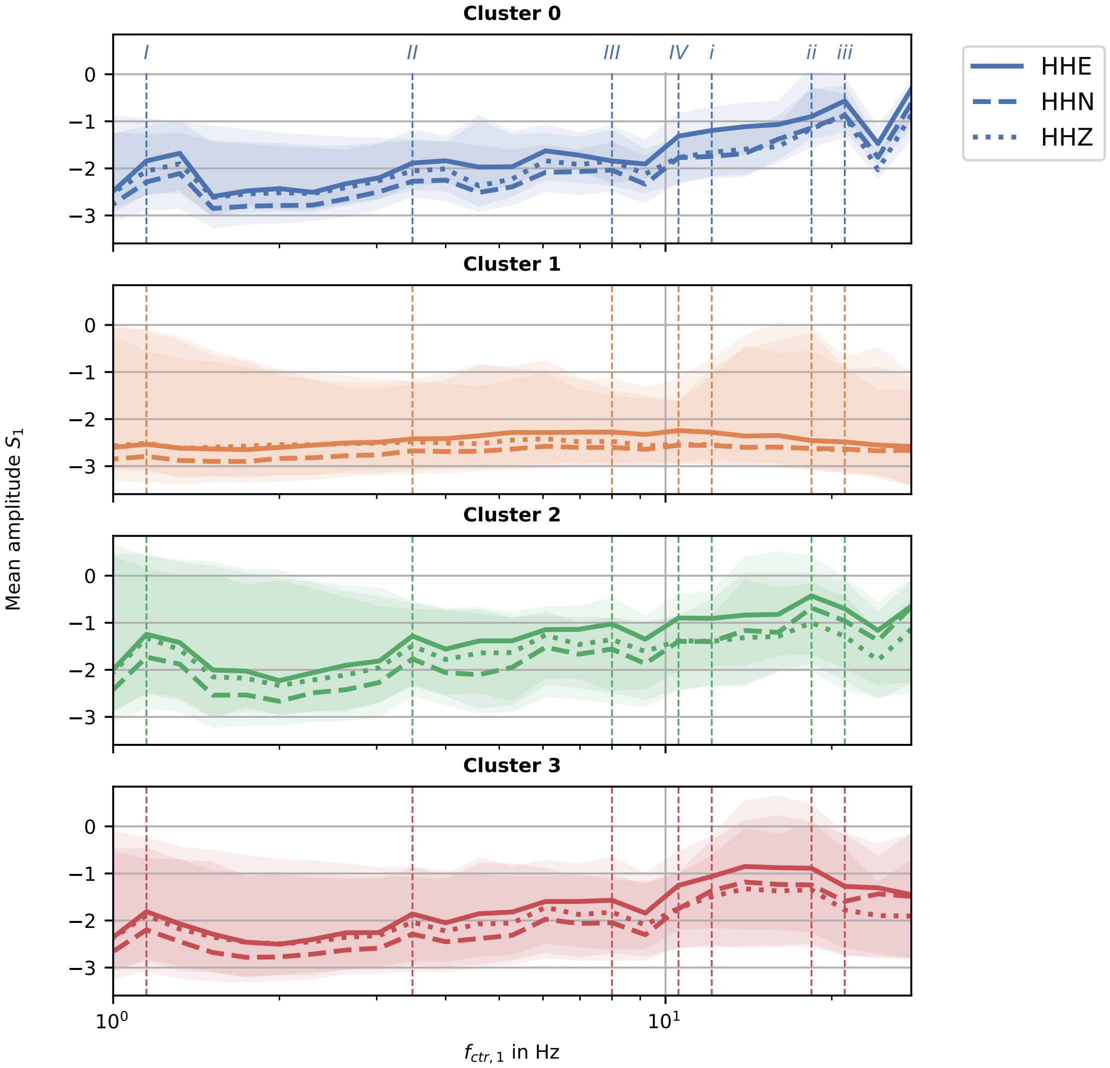


Figure 4.17: Mean amplitude of the first-order scattering coefficient S_1 of each seismometer component and cluster. The colored zones mark the range between minimum and maximum amplitude. The center frequencies of the first-order filterbank $f_{\mathrm{ctr},1}$ corresponding to the eigenmodes of the *WT 3* are marked with capital roman numbers and the 32nd multiple of the BPF of the different operational modes of the WT are highlighted with small roman numbers. Listed are the center frequencies and in parentheses the corresponding frequencies emitted by *WT 3*: I - 1.15 Hz (1.2 Hz), II - 3.48 Hz (3.6 Hz), III - 8.0 Hz (8.2 Hz), IV - 10.56 Hz (11.2 Hz), i - 12.13 Hz (12.64 Hz), ii - 18.38 Hz (17.6 Hz and 19.2 Hz), iii - 21.11 Hz (20.0 Hz).

4.4.3 HDBSCAN first-order scattering coefficients

In this section, I present the results of clustering the first-order scattering coefficient S_1 (Fig. 4.18). Since I use another input for HDBSCAN, I first have to perform the hyperparameter tuning.

Parameter tuning

First, a criterion for how parameter sets are considered adequately needs to be established. *WT 3* has five different operating modes: off, partial load, two noise-reduced modes, and full load (Section 2.2.5). Since the frequency content of the two noise-reduced modes falls in the same center frequency (Section 3.4), I expect to be able to distinguish at least four different clusters. In addition, these clusters should stand out from the others. Thus, there must be four clusters whose number of residents exceeds the median of all cluster residents. In cases where multiple parameter sets meet these criteria, priority is given to the hyperparameter set resulting in the selection of the smallest cluster with the highest number of points. This is intended to mitigate the formation of excessively small clusters.

I set the parameter `min_sample` to 25, 50, 100, and 200, respectively, and varried `min_cluster_size` varied between 25, 50 and 100 (Fig. 4.19). Lower or higher values did not meet the requirements. The median value is marked with a red line, and all hyperparameter sets, that satisfy the condition that at least four clusters must be larger than the median, are highlighted in red. Applying a `min_sample` set to 50 and `min_cluster_size` set to 100 resulted in the formation of the smallest cluster with the highest number of points and is highlighted with solid lines, while the results of the other parameter sets are highlighted with dashed lines.

Figure 4.20 shows the corresponding contest tree with cluster IDs. A total of eleven clusters are formed and compared to the condensed trees shown in Fig. 4.15, only a few of the clusters are further divided into subclusters and only up to a depth of two levels. However, this may be due to the large number of clusters.

Clustering statistics

I proceed with the cluster analysis and show the normalized cumulative sum of the cluster detections, the histograms of the operational data and the first-order mean scatter coefficient S_1 of each cluster. To do this, I group the eleven clusters into three groups. The first group

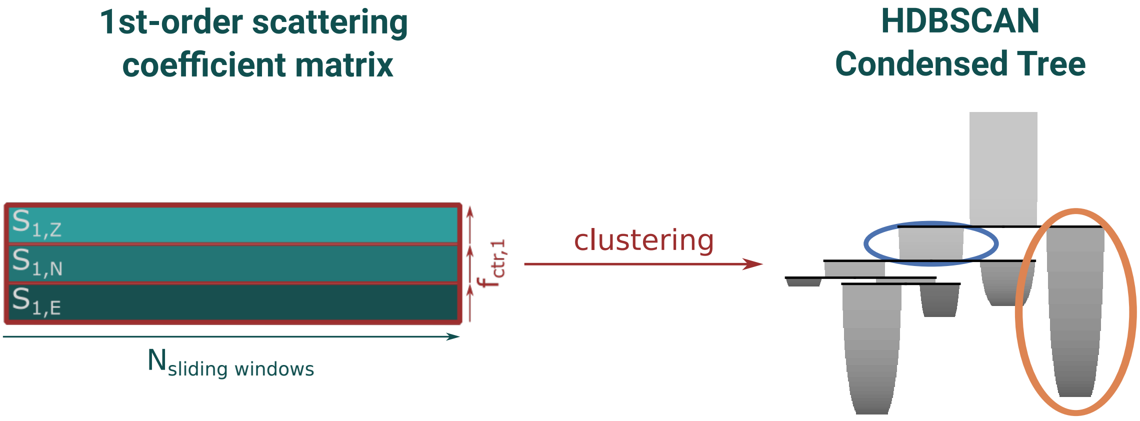


Figure 4.18: Workflow for clustering solely the first-order scattering coefficient matrix with HDBSCAN. In this example, the clustering algorithm separated the data into two main clusters (blue and orange).

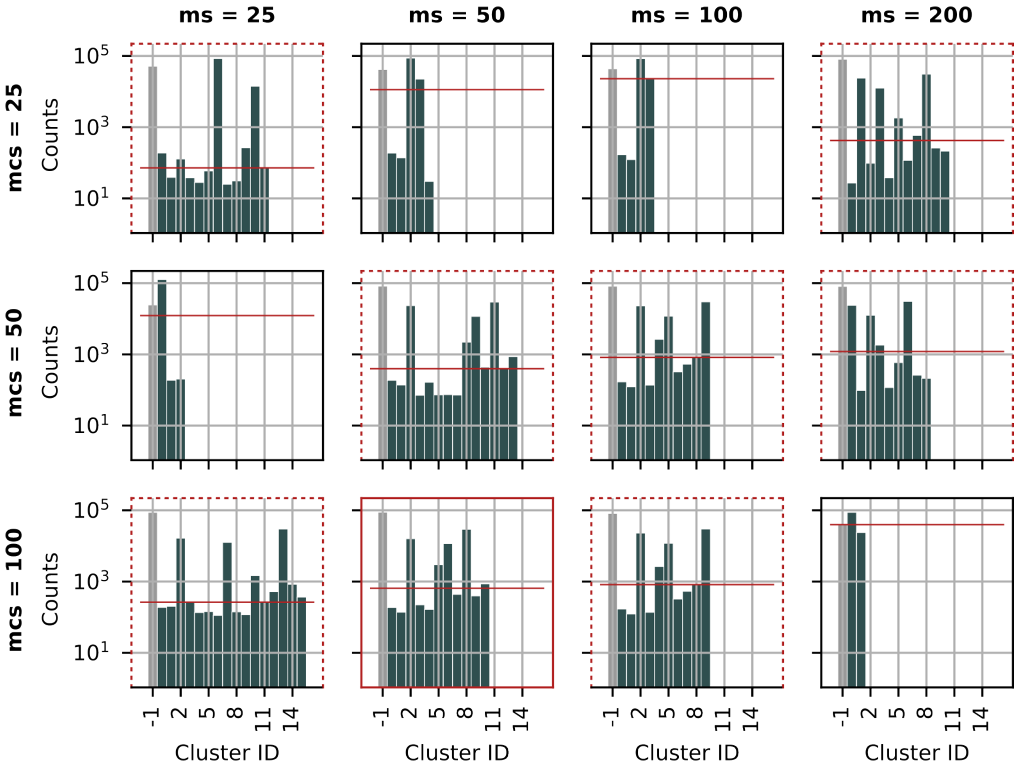


Figure 4.19: HDBSCAN parameter test utilizing the first-order scattering coefficient matrix of IW08B as input. The histograms show the number of points filling each cluster of the tested HDBSCAN parameters with the first-order-scattering coefficients as input. The cluster ID $-\mathbf{1}$ is for the points declared as noise and the bars are colored gray. The rows show the variation of the HDBSCAN hyperparameter `min_cluster_size` (mcs) and the columns of `min_samples` (ms). The horizontal red line marks the median value of each clustering result.

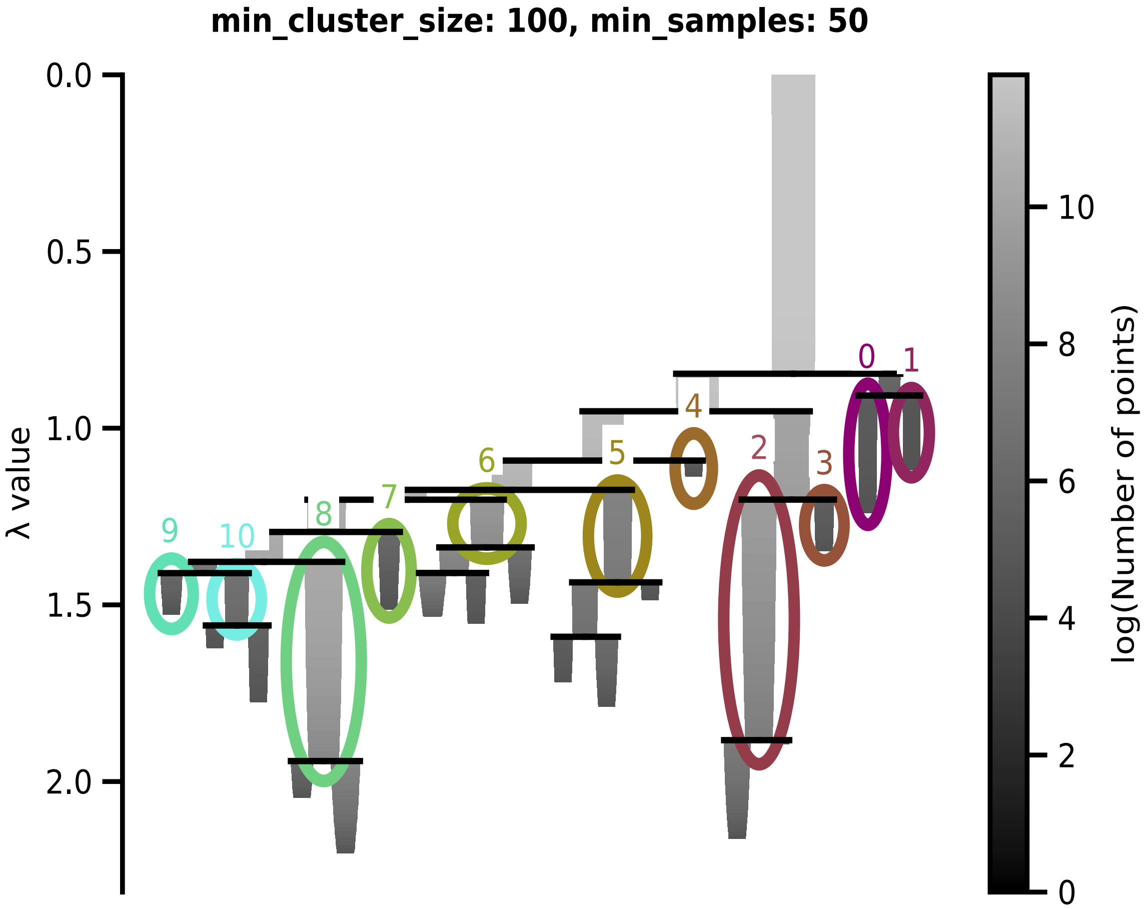


Figure 4.20: Condensed tree of the first-order scattering coefficients of IW08B with highlighted cluster IDs in color. The HDBSCAN hyperparameter `min_cluster_size` is set to 100 and `min_samples` to 50.

is displayed in Figures 4.21 to 4.23 and contains the clusters **0**, **1**, **2** and **3**. Figures 4.24 to 4.26 present the results of the second group and contains the clusters **4**, **5**, **8**. The clusters **6**, **7**, **9** and **10** form the third group and the results are shown in Figures 4.27 to 4.29. The analysis of the mean value S_1 is also summarized in Table 4.3.

The clusters of the first group are only filled with time windows in which *WT 3* was not in operation (Figs. 4.21 and 4.22). Out of the four clusters, cluster **2** is the largest and corresponds to the wind speeds between $0\,\mathrm{m\,s^{-1}}$ and $12\,\mathrm{m\,s^{-1}}$ but has no distinct wind direction. The time windows of clusters **0** and **3** correlate to wind speeds between $6\,\mathrm{m\,s^{-1}}$ and $12\,\mathrm{m\,s^{-1}}$ and differ in their wind directions. Cluster **1** contains the time windows with the largest wind speeds which are greater than $9\,\mathrm{m\,s^{-1}}$. There is no dominant wind direction. In all clusters but for cluster **2** the eigenmodes *WT 3* can be observed in the mean S_1 in Figure 4.23 and Table 4.3. Cluster **2** corresponds to the lowest wind speeds. If we only look at the maximum S_1 of this cluster the peaks of the WT's eigenmodes are visible (Fig. 4.23).

The second group combines clusters corresponding to *WT 3* running in the higher noise mode and at full load (Figs. 4.24 and 4.25). Cluster **4** is filled only during a few days in November when *WT 3* was running at full load. Cluster **8** is about ten times larger than cluster **5**. Both clusters are mainly filled during the times when *WT 3* was running at 12 rpm, the higher noise-reduced mode. The prevailing wind direction at the respective times distinguishes the two clusters, the wind comes from the E and SE in cluster **5** and from the S, SW, and W in cluster **8**. The 32nd multiple of the BPFs is prominent in all three clusters (Fig. 4.26 and Table 4.3). However, the peak at 18.38 Hz is less distinct in cluster **5** compared to cluster **8**. Furthermore, at this peak, the N-component is larger in cluster **8** compared to the Z-component, while the N- and Z-components are the same in cluster **5**. Cluster **4** correlates with winds from E, SE as well and N- and Z components also are similar at larger frequencies

The last group of clusters, cluster **3**, **7**, **9**, and **10**, combines time windows in which *WT 3* was running at partial load (cluster **3**) and at 11 rpm, the lower noise-reduced mode (Figs. 4.27 and 4.28). From the other three clusters, cluster **7** is the first one which is split off in the contest tree (Fig. 4.20). It is the only of these three clusters with wind coming from the SE. Clusters **9** and **10** have the same parent cluster, however, cluster **9** only correlates with wind speeds between $6\,\mathrm{m\,s^{-1}}$ and $9\,\mathrm{m\,s^{-1}}$ and westerly wind, while cluster **10** also correlates with higher wind speeds and wind from the S, SW, and W. Between the three clusters, the frequency peak corresponding to the 32nd multiple of the BPF is the least dominant in cluster **10**. Cluster **6** contains a few time windows in which *WT 3* was operated at 11 rpm and, therefore, its mean S_1 has two peaks corresponding to the 32nd multiple of the different BPFs.

Why do the noise-reduced modes separate into different clusters? The lower noise-reduced mode has a rotation rate of 11 rpm, with its corresponding 32nd multiple of the BPF at 17.6 Hz. Whereas, the higher noise-reduced mode operates at 12 rpm, which corresponds to the 32nd multiple of the BPF at 19.2 Hz. Consequently, the first-order filter bank, as shown in Fig. 3.9b, includes both multiples of BPF with a center frequency filter at $f_{ctr,1} = 18.37\,\mathrm{Hz}$. For the lower noise-reduced mode, some energy likewise is captured by the filter at $f_{ctr,1} = 16\,\mathrm{Hz}$, whereas, for the higher noise-reduced mode, energy is captured by the filter at $f_{ctr,1} = 21.11\,\mathrm{Hz}$, as well. Due to the broader high-frequency filters, more energy is distributed at a rotation rate of 12 rpm, resulting in a broad peak that covers frequencies at both 18.37 Hz and 21.11 Hz (Fig. 4.26). In contrast, at a rotation rate of 11 rpm, the peak at 18.37 Hz is more pronounced (Fig. 4.29). For the lower noise-reduced mode, some energy is captured by the filter at $f_{ctr,1} = 16\,\mathrm{Hz}$. For the higher noise-reduced mode some energy is captured by the filter at $f_{ctr,1} = 21.11\,\mathrm{Hz}$. As the high-frequency filters are broader, more energy is distributed at a rotation rate of 12 rpm leading to a broad peak covering $f_{ctr,1}$ at 18.37 Hz and 21.11 Hz (Fig. 4.26), while with a rotation rate of 11 rpm the peak at 18.37 Hz is more prominent (Fig. 4.29).

In all clusters, the E-component of S_1 exhibits the highest amplitudes (Figs. 4.23, 4.26 and 4.29). Additionally, for frequencies below approximately 10 Hz, the amplitude of the Z-component $S_{1,Z}$ surpasses that of the N-component $S_{1,N}$ (Figs. E.2 to E.4). For frequencies exceeding 10 Hz, $S_{1,Z}$ generally exhibits smaller amplitudes compared to $S_{1,N}$. However, in clusters **2** and **3** both components are equal (Fig. E.2). Similarly, in clusters **4** and **5**, the components are equal for the frequencies corresponding to the 32nd multiple of the BPF (Fig. E.3). In particular, these clusters share a common wind direction dominance, specifically from the NE, E, or SE. A similar wind direction pattern is observed in cluster **7**, where the mean values of $S_{1,Z}$ and $S_{1,N}$ are not identical, but fall within the maximum and minimum range of values (Fig. E.4). The other clusters have dominant wind directions from the S, SW, or W.

Concluding remarks

Clustering S_1 with HDBSCAN separates all five operating modes into individual clusters. In addition, these modes have been further subdivided into clusters showing dependencies on wind speed or wind direction. This was not possible using UMAP as clustering input. The influence of wind speed is indicated by its correlation with increased seismic emission amplitudes of *WT 3* (Figs. 4.22 and 4.23). Conversely, wind direction introduces variability in emissions, especially at frequencies above about 10 Hz. In particular, when prevailing winds are from the NE, E, or SE, the first-order scattering coefficients $S_{1,Z}$ are equal to $S_{1,N}$ at frequencies corresponding to the 32nd multiple of BPF. On the other hand, when the prevailing winds are from the S, SW, or W, $S_{1,N}$ is greater than $S_{1,Z}$ at frequencies corresponding to the 32nd multiple of BPF. This observation suggests the activation of separate vibration sources of *WT 3* corresponding to different wind directions. This will contribute to a better understanding of seismic WT emissions and will provide new opportunities for further investigations.

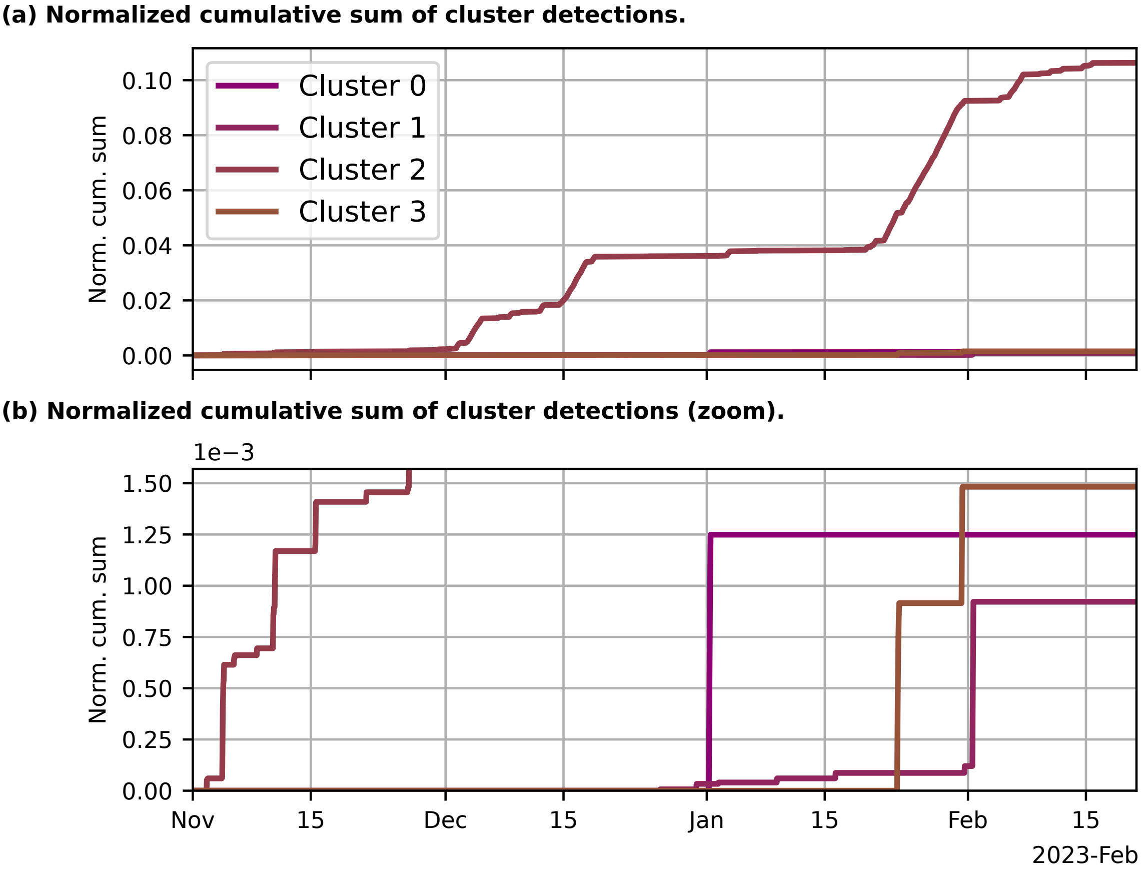


Figure 4.21: Clustering statistics of results using the HDBSCAN hyperparameters `min_sample` set to 50 and `min_cluster_size` to 100, utilizing the first-order scattering coefficient matrix of IW08B as clustering input. Out of the eleven clusters, only clusters **0** to **3** are depicted. (a) The plot illustrates the normalized cumulative sum of cluster detections over time, relative to all eleven clusters. (b) Zoom in on the cluster detections of clusters **0**, **1**, and **3** located in the lower part of (a).

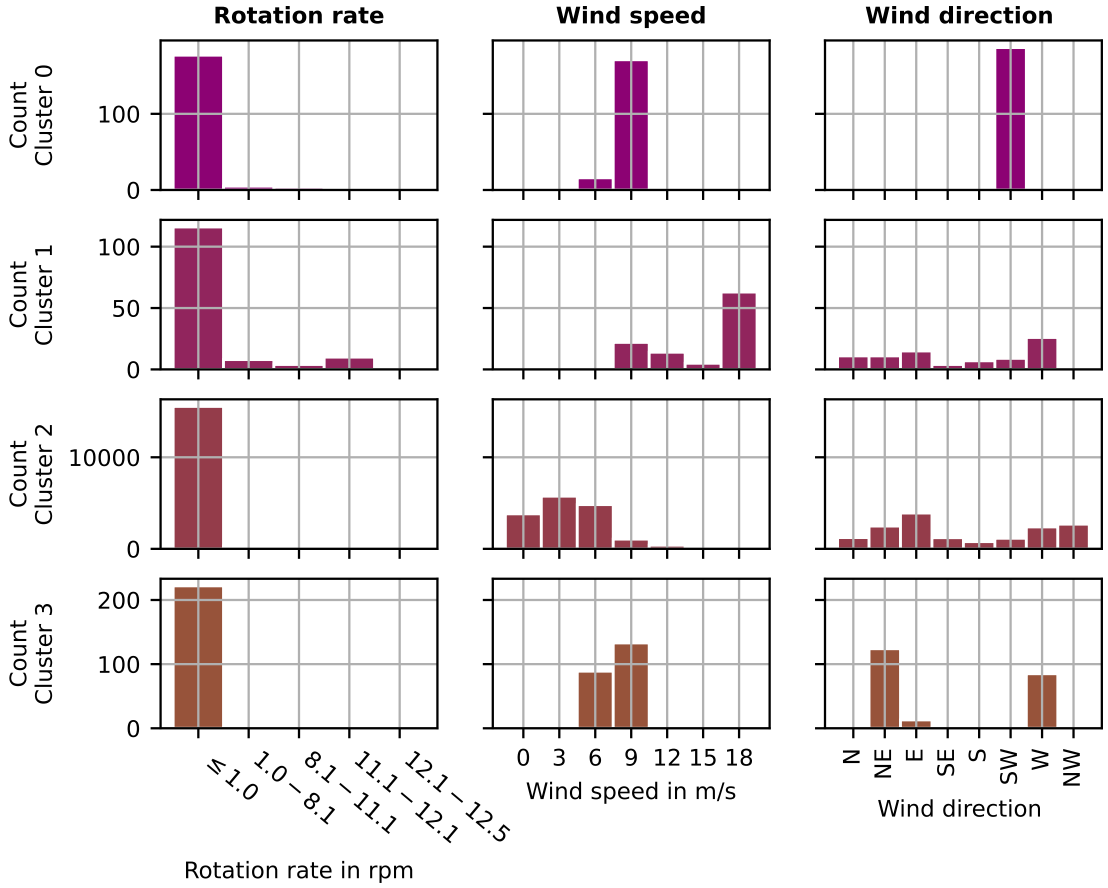


Figure 4.22: Clustering distribution of results using the HDBSCAN hyperparameters `min_sample` set to 50 and `min_cluster_size` to 100, utilizing the first-order scattering coefficient matrix of IW08B as clustering input. Out of the eleven clusters, only clusters **0** to **3** are depicted. The distribution of clusters is analyzed concerning the rotation rate of *WT 3*, wind speed, and direction.

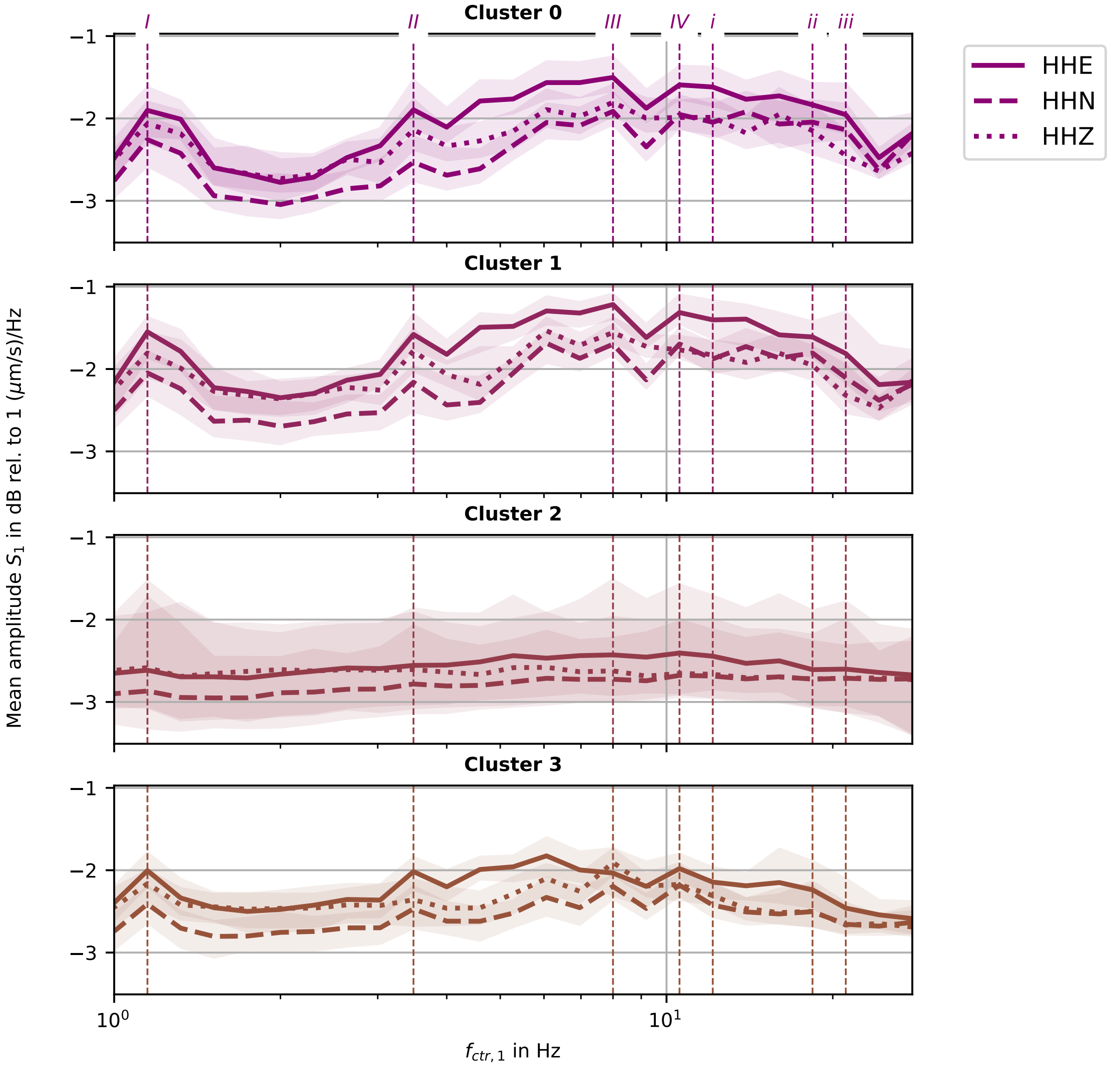


Figure 4.23: Mean amplitude of the first-order scattering coefficients S_1 of IW08B of each seismometer component and cluster utilizing S_1 as clustering input and the hyperparameters `min_sample` set to 50 and `min_cluster_size` to 100. The colored zones mark the range between minimum and maximum amplitude. Out of the eleven clusters, only clusters **0** to **3** are depicted.

Table 4.3: Prominence of the WT related peaks evaluated for the clusters shown in Figures 4.23, 4.26 and 4.29. Prominent maxima are marked with ++, weak maxima with a +, and a - if there is no distinct maximum. The center frequencies of the first-order filterbank corresponding to the eigenmodes of the *WT 3* are marked with capital roman numbers and the 32 multiple of the BPF of the different operational modes of the WT are highlighted with small roman numbers. Listed are the center frequencies and in parentheses the corresponding frequencies emitted by *WT 3*: I - 1.15 Hz (1.2 Hz), II - 3.48 Hz (3.6 Hz), III - 8.0 Hz (8.2 Hz), IV - 10.56 Hz (11.2 Hz), i - 12.13 Hz (12.64 Hz), ii - 18.38 Hz (17.6 Hz and 19.2 Hz), iii - 21.11 Hz (20.0 Hz).

	I	II	III	IV	i	ii	iii
Cluster 0	++	++	++	+	-	-	-
Cluster 1	++	++	++	+	-	-	-
Cluster 2	-	-	-	-	-	-	-
Cluster 3	++	++	++	++	-	-	-
Cluster 4	++	-	+	-	-	-	++
Cluster 5	++	++	+	-	-	+	-
Cluster 8	++	++	+	-	-	++	-
Cluster 6	++	++	+	-	++	++	-
Cluster 7	++	++	+	+	-	++	-
Cluster 9	++	++	++	+	-	++	-
Cluster 10	++	++	++	+	-	++	-

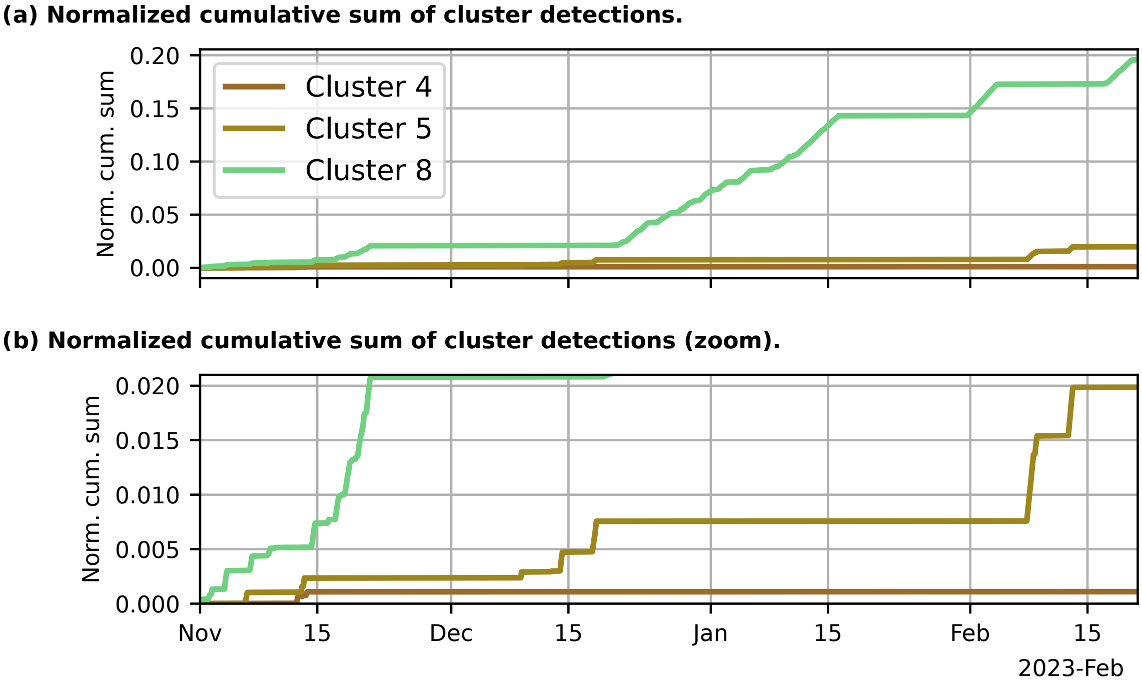


Figure 4.24: Clustering statistics of results using the HDBSCAN hyperparameters `min_sample` set to 50 and `min_cluster_size` to 100, utilizing the first-order scattering coefficient matrix of IW08B as clustering input. Out of the eleven clusters, only clusters **4**, **5**, and **8** are depicted. (a) The plot illustrates the normalized cumulative sum of cluster detections over time, relative to all eleven clusters. (b) Zoom in on the cluster detections of clusters **4** and **5** located in the lower part of (a).

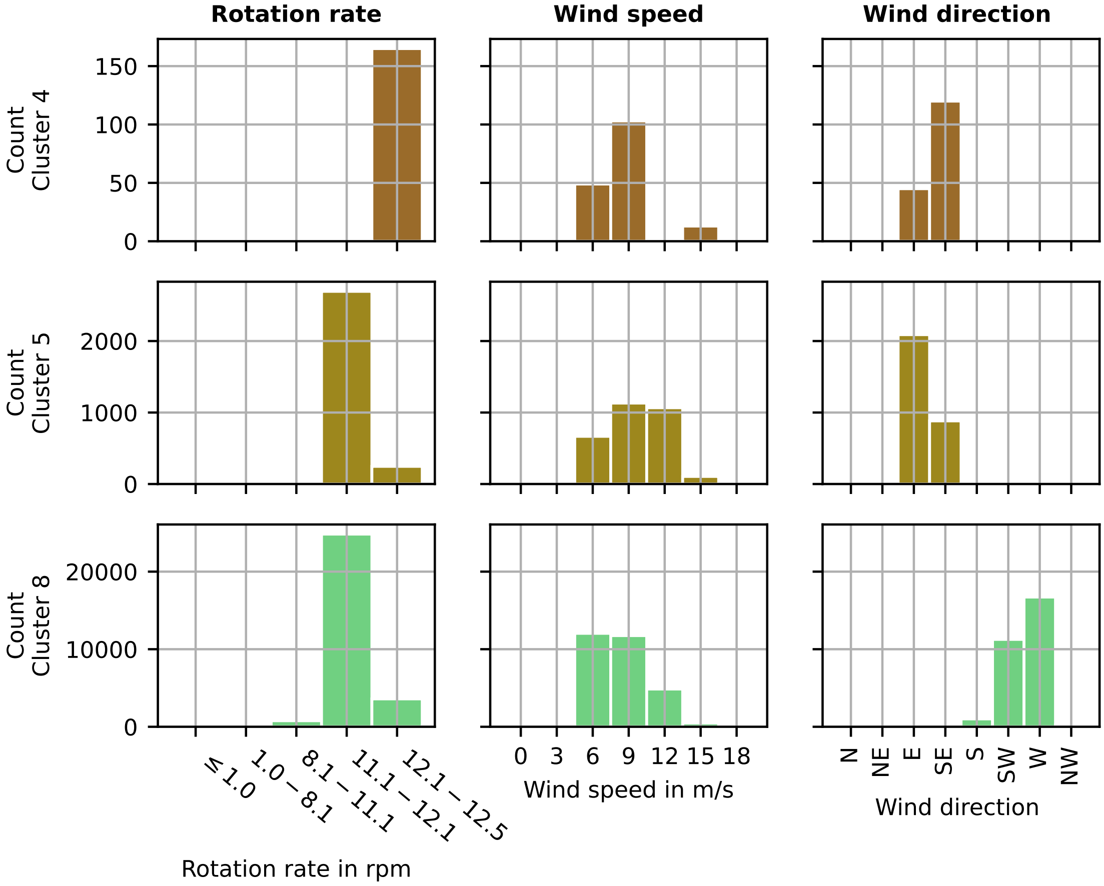


Figure 4.25: Clustering distribution of results using the HDBSCAN hyperparameters `min_sample` set to 50 and `min_cluster_size` to 100, utilizing the first-order scattering coefficient matrix of IW08B as clustering input. Out of the eleven clusters, only clusters **4**, **5**, and **8** are depicted. The distribution of clusters is analyzed concerning the rotation rate of *WT 3*, wind speed, and direction.

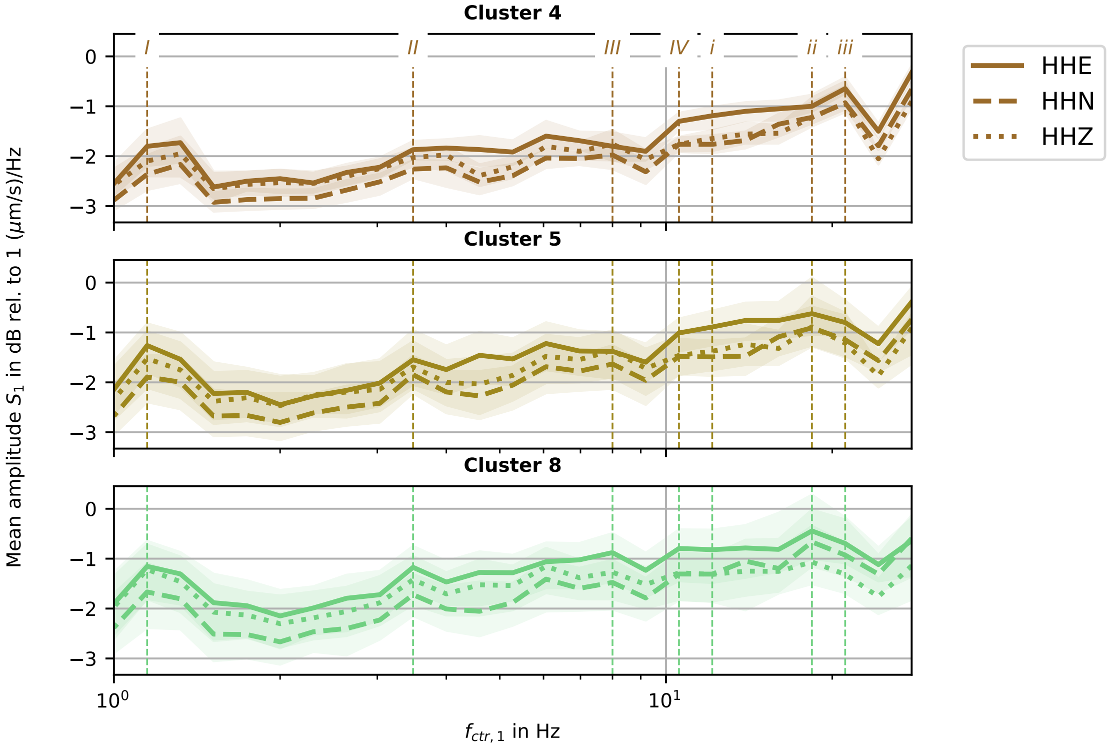


Figure 4.26: Mean amplitude of the first-order scattering coefficients S_1 of IW08B of each seismometer component and cluster utilizing S_1 as clustering input and the hyperparameters `min_sample` set to 50 and `min_cluster_size` to 100. The colored zones mark the range between minimum and maximum amplitude. Out of the eleven clusters, only clusters **4**, **5**, and **8** are depicted.

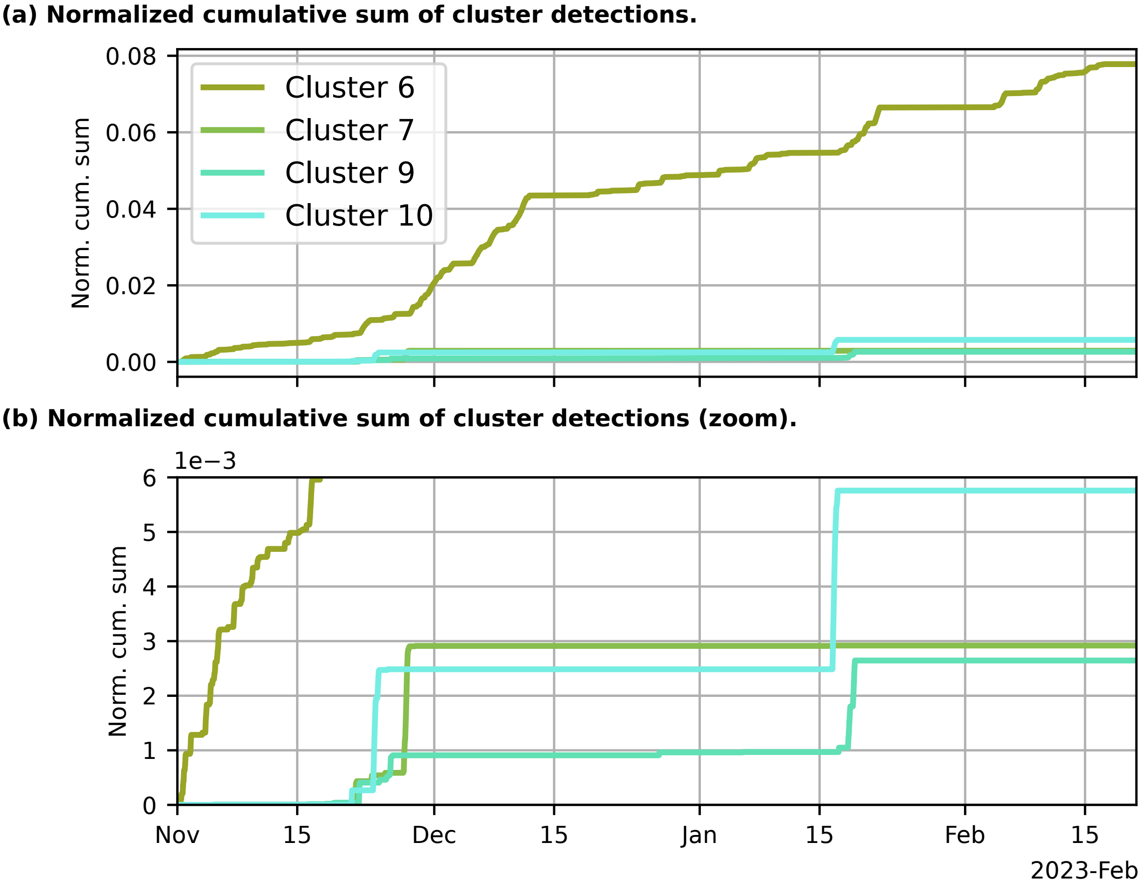


Figure 4.27: Clustering statistics of results using the HDBSCAN hyperparameters `min_sample` set to 50 and `min_cluster_size` to 100, utilizing the first-order scattering coefficient matrix of IW08B as clustering input. Out of the eleven clusters, only clusters **6**, **7**, **9**, and **10** are depicted. (a) The plot illustrates the normalized cumulative sum of cluster detections over time, relative to all eleven clusters. (b) Zoom in on the cluster detections of clusters **7**, **9**, and **10** located in the lower part of (a).

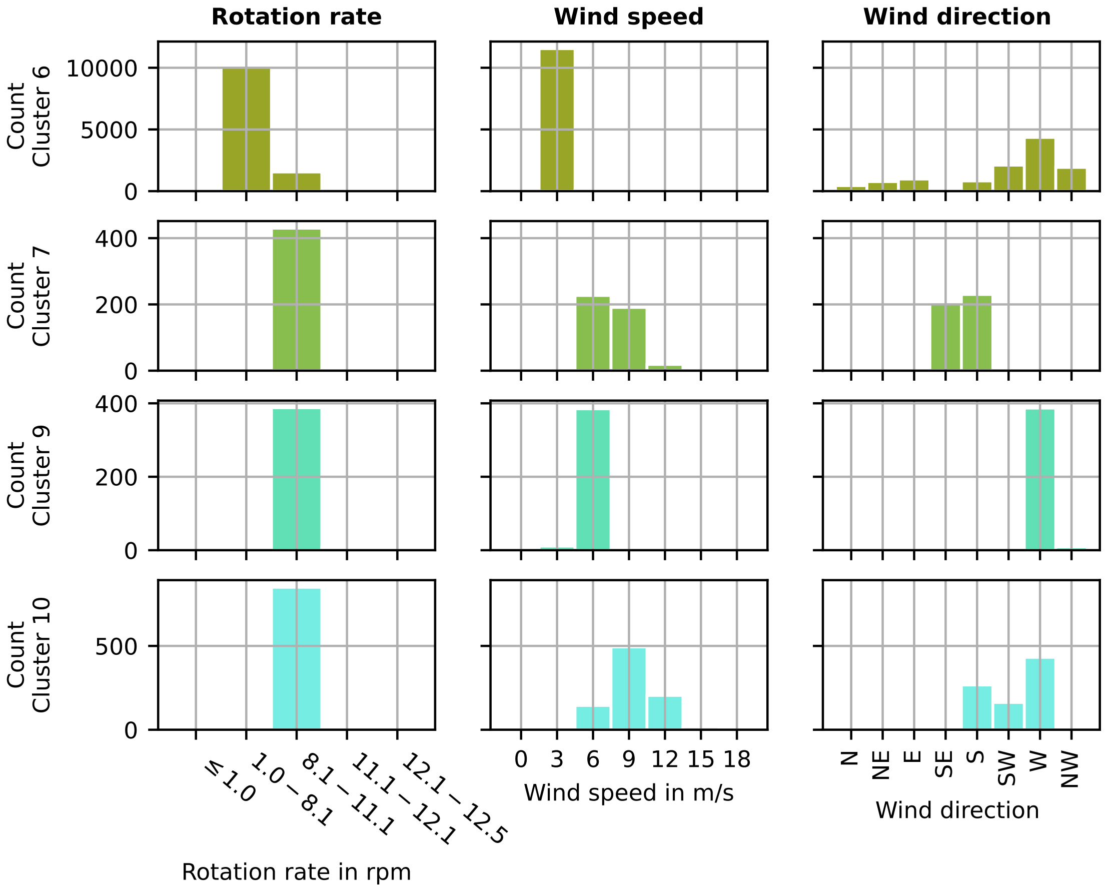


Figure 4.28: Clustering distribution of results using the HDBSCAN hyperparameters `min_sample` set to 50 and `min_cluster_size` to 100, utilizing the first-order scattering coefficient matrix of IW08B as clustering input. Out of the eleven clusters, only clusters **6**, **7**, **9**, and **10** are depicted. The distribution of clusters is analyzed concerning the rotation rate of *WT 3*, wind speed, and direction.

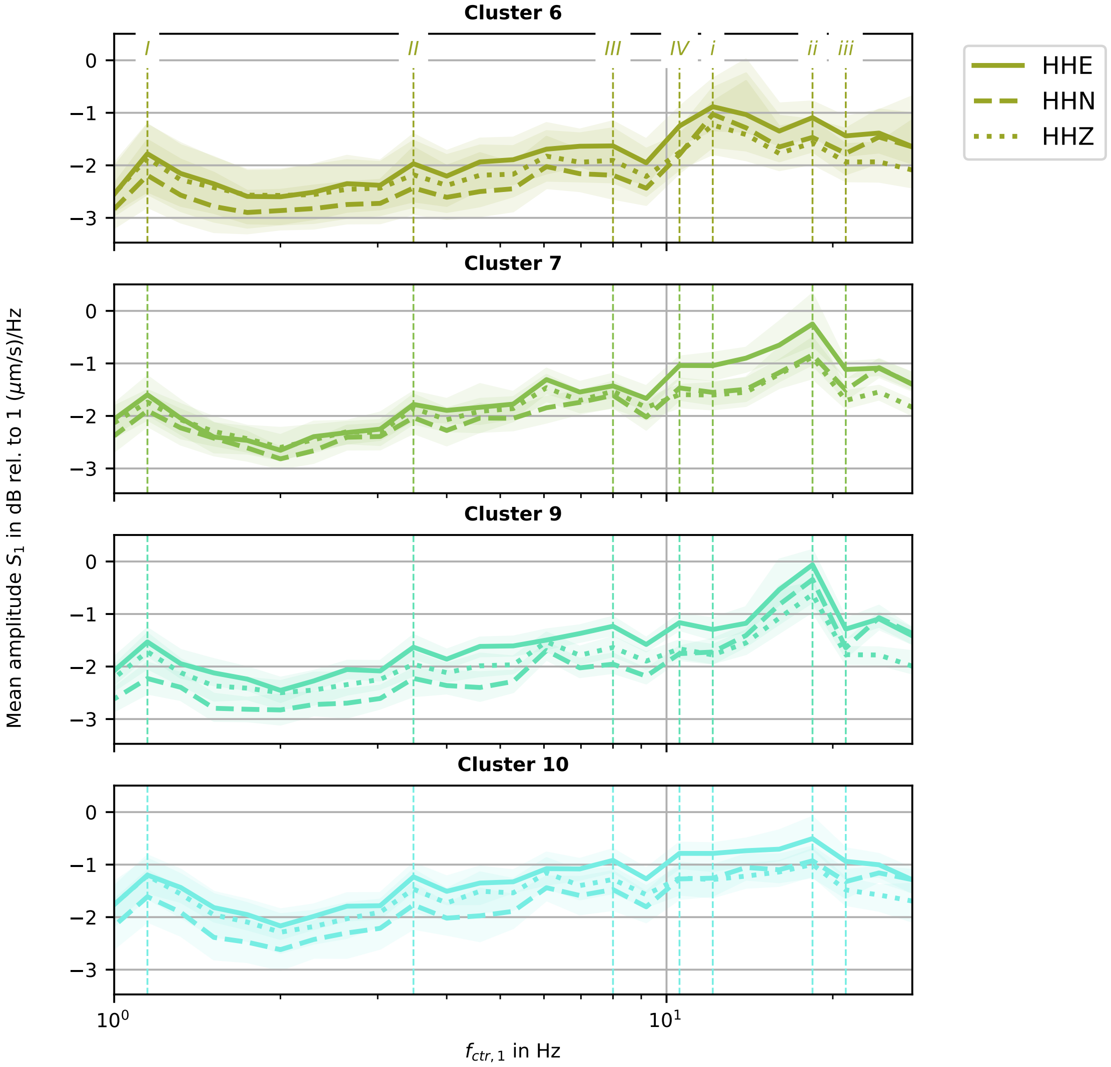


Figure 4.29: Mean amplitude of the first-order scattering coefficients S_1 of IW08B of each seismometer component and cluster utilizing S_1 as clustering input and the hyperparameters `min_sample` set to 50 and `min_cluster_size` to 100. The colored zones mark the range between minimum and maximum amplitude. Out of the eleven clusters, only clusters **6**, **7**, **9**, and **10** are depicted.

Chapter 5

Discussion

Seismic emissions from WTs affect seismological and other sensitive measurements within a radius of up to 11 km (Saccorotti et al., 2011). To enhance the understanding of signal sources associated with WTs, I employ ML methods. Specifically, I apply a clustering algorithm to ground motion data recorded during the Inter-Wind project IW08 experiment in the vicinity of *WT 3* at WF Tegelberg on the eastern Swabian Alb, southwestern Germany (Figs. 1.1 and 2.1), to identify known and unknown patterns of WT emissions.

Known patterns include the eigenmodes of the tower-nacelle system and five different modes in which *WT 3* was operated during the IW08 experiment: standstill, partial load at 7.9 rpm, a lower noise-reduced mode at 11 rpm, a higher noise-reduced mode at 12 rpm, and full load at 12.5 rpm. These modes can be distinguished by peaks corresponding to multiples of the BPF in a spectrogram. In the analyzed data, the highest visible peak related to the BPF is its 32nd multiple. Unknown patterns to be identified may include correlations with wind direction or other meteorological conditions.

To cluster seismic signals, it is not sufficient to divide a dataset into time windows and group them with a clustering algorithm. A waveform of the same shape can be stretched or slightly deformed, which can lead to the clustering algorithm no longer assigning these signals to the same group. Therefore, a translation invariant representation is needed that is stable against small deformations (Anden and Mallat, 2014; Bruna and Mallat, 2013). This representation can be obtained by applying the scattering transform and was first used by Anden and Mallat (2014) and Bruna and Mallat (2013) in acoustics and image processing. Barkaoui et al. (2021), Morel et al. (2023), Rodríguez et al. (2022), Seydoux et al. (2020), Steinmann et al. (2022b), and Steinmann et al. (2023) successfully applied the scattering transform to seismological data. Therefore, before applying the clustering algorithm, I first transform the IW08B dataset with a scattering network. To do this, I use the Python package scatseisnet (Seydoux and Steinmann, 2023).

5.1 Scattering network

Scatseisnet is based on the work from Seydoux et al. (2020) and Steinmann et al. (2022a,b) and has been applied to seismic data to analyze, e.g., earthquakes, ground freezing patterns, or seismo-volcanic activity, but to date not to mono-frequent signals comparable to seismic WT emissions.

For this reason, I selecte distinct time windows representing different operational modes of *WT 3* and additionally generated synthetic signals (Fig. 3.3). With these test signals, I tune the scattering network to the presented data.

Initially, I test the effect of tapering the input signal to reduce the edge-effect artifacts (Figs. 3.6 and 3.7). Following this, I examine the normalization of the filter banks. The application of the wavelet transform attenuates high amplitudes, but normalizing the filter banks with their $L1$ norm makes it possible to compare the amplitudes with each other (Fig. 3.10).

After the wavelet transform, the wavelet coefficients are pooled to obtain the scattering coefficients (Fig. 3.11). This ensures invariance to translation. I show that average pooling is the optimal method to capture, both, the mono-frequenct signal of the WT eigenmodes and the gliding frequencies corresponding to the multiples of the BPF. Finally, I tune the quality factor, which controls the spectral width of the wavelet, to fit the analyzed signal content Figs. 3.12 and 3.13.

With the final design of this scattering network, the redundant patterns are extracted from the IW08B recordings. The first-order scattering coefficients S_1 include the primary amplitude information of the seismic WT emissions. They contain both the mono-frequenct signal and the gliding frequencies corresponding to the 32nd multiples of BPF (Figs. 3.18, C.2 and C.3). The interpretation of the second-order scattering coefficients, S_2, of seismic emissions is a challenging task. I expecte to see, e.g., frequency intervals or amplitude modulation (Anden and Mallat, 2014). While it is possible to identify frequency intervals in the S_2 of the synthetic test signals, it is not possible to distinguish easily from the other signal contributions in the S_2 of the recorded signals and need to (Fig. 3.17).

5.2 Dimension reduction and clustering

The high dimensionality of the scattering coefficient matrix prohibits direct processing with a clustering algorithm, as in a high-dimensional space, all distances approach the same value. Therefore, I use the non-linear dimension reduction technique UMAP before clustering. I cho0se a hierarchical clustering algorithm to detect substructures in the data, called HDBSCAN.

Both UMAP and HDBSCAN require extensive parameter testing, as the clustering result of this dataset strongly depended on the chosen hyperparameters (Figs. 4.6, 4.8 and 4.13). The resulting UMAP atlas could be visually grouped into areas primarily correlated with the rotation rate of *WT 3* (Fig. 4.9). The major separation within the UMAP atlas was between the standstill of the WT and the times in which it was in operation. However, the different operational modes overlap in the UMAP atlas and could, therefore, not be separated into individual clusters (Figs. 4.9 and 4.16). The UMAP atlas has been grouped with HDBSCAN into four clusters:

- One cluster that correlates with time windows in which *WT 3* was at a standstill.
- A cluster of times when WT ran in partial load, the lower noise-reduced mode, and a few time windows when *WT 3* ran in the higher noise-reduced mode.
- Another cluster mainly groups time windows correlating with the higher noise-reduced mode but also includes time windows correlating with the lower noise-reduced modes and full load.
- The remaining cluster is filled with time windows in which *WT 3* was operated at full load and is the only cluster in which the wind direction is restricted.

This result could have been obtained without clustering. A simple analysis of the spectrogram of the recorded data would have been sufficient. However, I reduce the previous 2175 dimensional scattering coefficient matrix to only two UMAP variables, and therefore a loss of information can be expected. However, it may be possible to preserve, for example, the signal content correlating with wind direction by using more UMAP components. In the scope of this work, I analyze only two UMAP variables, but the values of the UMAP components can be varied between two and 100.

5.3 Clustering the first-order scattering coefficient matrix

While testing different inputs for the UMAP algorithm, I find that utilizing only the first-order scattering coefficient matrix S_1 leads to a better separation of the wind direction than using the entire scattering coefficient matrix (Fig. 4.12). While S_1 contains the primary amplitude information of the seismic WT emissions and has only 75 dimensions, the second-order scattering coefficient matrix S_2 shows amplitude modulation or frequency intervals and has 2100 dimensions. Since the UMAP variables of the entire scattering matrix or of S_2 show almost no correlation with wind direction, the main signal content correlating with wind direction must be in S_1. With HDBSCAN being able to cluster data with up to about 100 dimensions (McInnes et al., 2017), I, therefore, utilize S_1 directly as HDBSCAN input.

The clustering result includes not only individual clusters corresponding to the five operating modes of *WT 3*, but the clusters also depend on wind speed and wind direction and leading in total to eleven clusters (Figs. 4.22, 4.25 and 4.28), four corresponding to the standstill of *WT 3*, one to partial load, and one to full load. Three clusters correlate to the lower noise-reduced mode and two to the higher noise-reduced mode.

While the first four clusters indicate that the main influence for the grouping is the standstill of *WT 3*, the other separations predominantly correlate with the wind direction, and the rotation rate appears to be of secondary importance.

The reason for this could be that the wind direction affects the ratios between the Z-component of S_1 ($S_{1,Z}$) and its N-component ($S_{1,N}$) at the frequency corresponding to the 32nd multiple of BPF, if the WT is operating, otherwise all frequencies above about 10 Hz are affected. If the wind is blowing from the east, $S_{1,N}$ and $S_{1,Z}$ are equal, otherwise the amplitude of $S_{1,N}$ is greater than $S_{1,Z}$ (Figs. E.2 to E.4).

IW08B is set up in the NW of *WT 3* (Fig. 2.1). In the case the wind is blowing from the SW, WT and the station are in line with the downwind direction. With northeasterly wind, *WT 3* and IW08B are in line with the crosswind direction. With the $S_{1,N}$ being larger than the $S_{1,Z}$ with wind from the NE, this could imply that Rayleigh waves are radiated in a crosswind direction. When the wind is blowing from the SW, $S_{1,N}$ and $S_{1,Z}$ are equal in amplitude, implying that Love waves are radiated in a downwind direction. Neuffer et al. (2021) observed a similar radiation pattern but for the eigenfrequencies of the WT and not the multiples of the BPF. For the eigenfrequencies 0.3 Hz, 3.25 Hz, and 6.0 Hz they observed Rayleigh waves being radiated in crosswind direction and Love waves in a downwind direction, similar to my observation. However, to confirm the radiation pattern considerations regarding the WT emissions recorded at IW08B, the respective particle motion must be analyzed.

The development of the method for characterizing and clustering seismic WT emissions requires extensive parameter testing before the presented results can be obtained. As I have only applied the workflow to one dataset, further tests need to be performed to prove the reliability of this method. Nevertheless, the results are already promising, and further investigation may lead to a better understanding of the radiation characteristics of WTs.

Chapter 6

Conclusion and outlook

In this study, I develope a workflow to analyze seismic WT emissions with unsupervised ML, utilizing three component ground motion data recorded during the project Inter-Wind at the WF Tegelberg on the eastern Swabian Alb, southwestern Germany. The initial workflow include three processing steps:

(i) the extraction of the most relevant signal features from the recorded ground motion data of IW08B with the application of a two-layered scattering network utilizing the Python package scatseisnet (Seydoux and Steinmann, 2023).
(ii) Following this, the high-dimensional scattering coefficient matrix is reduced to two dimensions by applying the non-linear dimension reduction technique UMAP (McInnes et al., 2018).
(iii) Afterward the UMAP variables are clustered with HDBSCAN (McInnes et al., 2017).

Since this workflow does not lead to the desired clustering results of separating the known and unknown patterns of the seismic WT emissions, I need to adapt it. During the development of the initial workflow, I have become aware that the first-order scattering coefficient matrix S_1 contains signal patterns that correlate with wind direction, one of the lesser understood influences on seismic WT emissions. Based on this, I adapte the first two steps of the initial workflow and use only S_1 as clustering input:

(i) Extracting the most relevant signal features from the recorded ground motion data of IW08B with the application of a one-layered scattering network utilizing the Python package scatseisnet (Seydoux and Steinmann, 2023).
(ii) Clustering of the resulting first-order scattering coefficients S_1 with HDBSCAN (McInnes et al., 2017).

The application of my final workflow to ground motion data of IW08B generates eleven distinct clusters, whose correlations differ not only in the rotation rate of the WT but also in the wind speed and wind direction. Furthermore, by utilizing only a one-layered scattering network followed by the HDBSCAN clustering I not only reduce the total computation time but also minimize the number of hyperparameters needed to be adjusted. The scattering network can be directly applied to other ground motion recordings containing seismic WT emissions, as I tune the network to seismic WT emissions in general. The hyperparameters of HDBSCAN, however, need to be adjusted with each new dataset, as it is dependent on its size. With only two parameters, this is a formidable task.

With this work I have contributed to the scatseisnet library, in particular, I have optimized the tapering procedure and implemented the wavelet normalization.

Furthermore, with my final workflow, I have successfully extracted various known and previously unknown patterns related to the rotation rate of the WT, the wind speed, and the wind direction. I have shown a dependence of the WT radiation pattern on the wind direction that can be observed at the 32nd multiple of BPF. However, the letter requires further investigation before general conclusions can be drawn. I have thus paved the way for a greater understanding of seismic WT emissions.

6.1 Outlook

Before applying the developed workflow to other datasets, some processing steps in the scattering network can be revised. In particular, possible improvements or a better understanding of the scattering network can be achieved by reviewing the tapering of the input signal to the scattering network and by fine-tuning its filter bank.

The area in the scalogram affected by the edge-effect artifacts is also called the "cone of influence" (Torrence and Compo, 1998). A Tukey taper is implemented in the Python package scatseisnet to handle the edge effect. Other studies, such as Lilly (2017) and Liu et al. (2007), use different approaches to handle the edge-effect artifacts that could be compared to the Tukey taper.

In the final workflow of my thesis, I apply clustering only to the first-order scattering coefficients S_1 of IW08B. If only S_1 is utilized as clustering input, the hyperparameters controlling the frequency content of the filter banks, the number of octaves J, and the wavelets per octave J, can be altered. In case higher multiples of the BPF should be analyzed, the upper-frequency limit of the filter bank needs to be adjusted. It must be ensured that S_1 does not have more than 100 dimensions, for HDBSCAN to be able to process S_1.

My final workflow does not utilize the second-order scattering coefficients S_2 of IW08B. To analyze the clustering results of both, S_1 and S_2, the dimensions of S_2 can be reduced with UMAP and the result concatenated with S_1. This way one would still include S_2 in the analysis.

To prove the reliability of the developed method, it is necessary to apply it to other datasets. For this, the line measurement of experiment IW05 is well suited and allows to study up to what distance the clustering algorithm can separate the signals of the seismic WT emissions (Section 2.2.2). Depending on these results, this workflow can also be applied to other recording stations of the IW08 experiment to investigate whether the WT emissions can be isolated from anthropogenic noise.

In the IW05 experiment, a ring measurement was performed, but not at all stations all three components are reliable (Gaßner and Ritter, 2023a). To further investigate the radiation characteristics of WTs, an analysis of a ring measurement with three-component seismometers would be beneficial.

Appendix A

Wind speed and direction during measurement campaign IW08

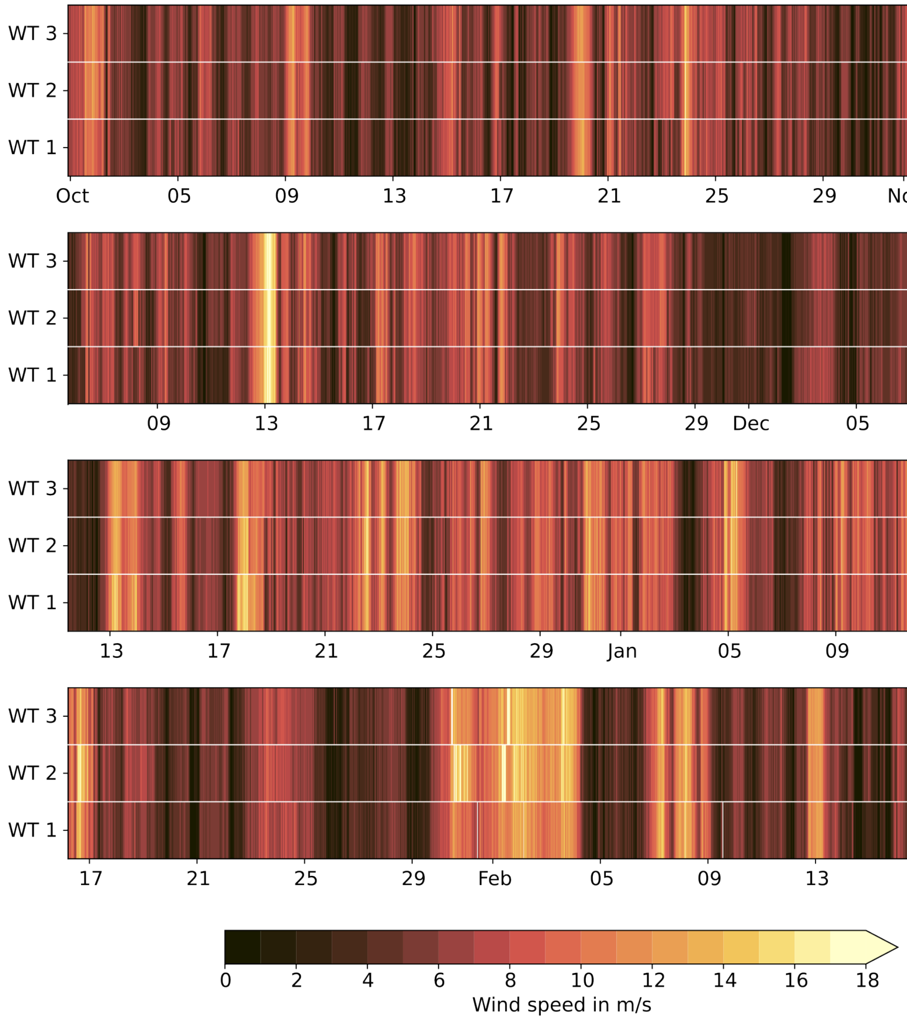


Figure A.1: Wind speed recorded at WF Tegelberg for wind turbines (WTs) *1* to *3* during the measurement campaign IW08 between November 1, 2022 and February 20, 2023. The color scale is clipped at the cut-off wind-speed of the WT at 18 m s^{-1}.

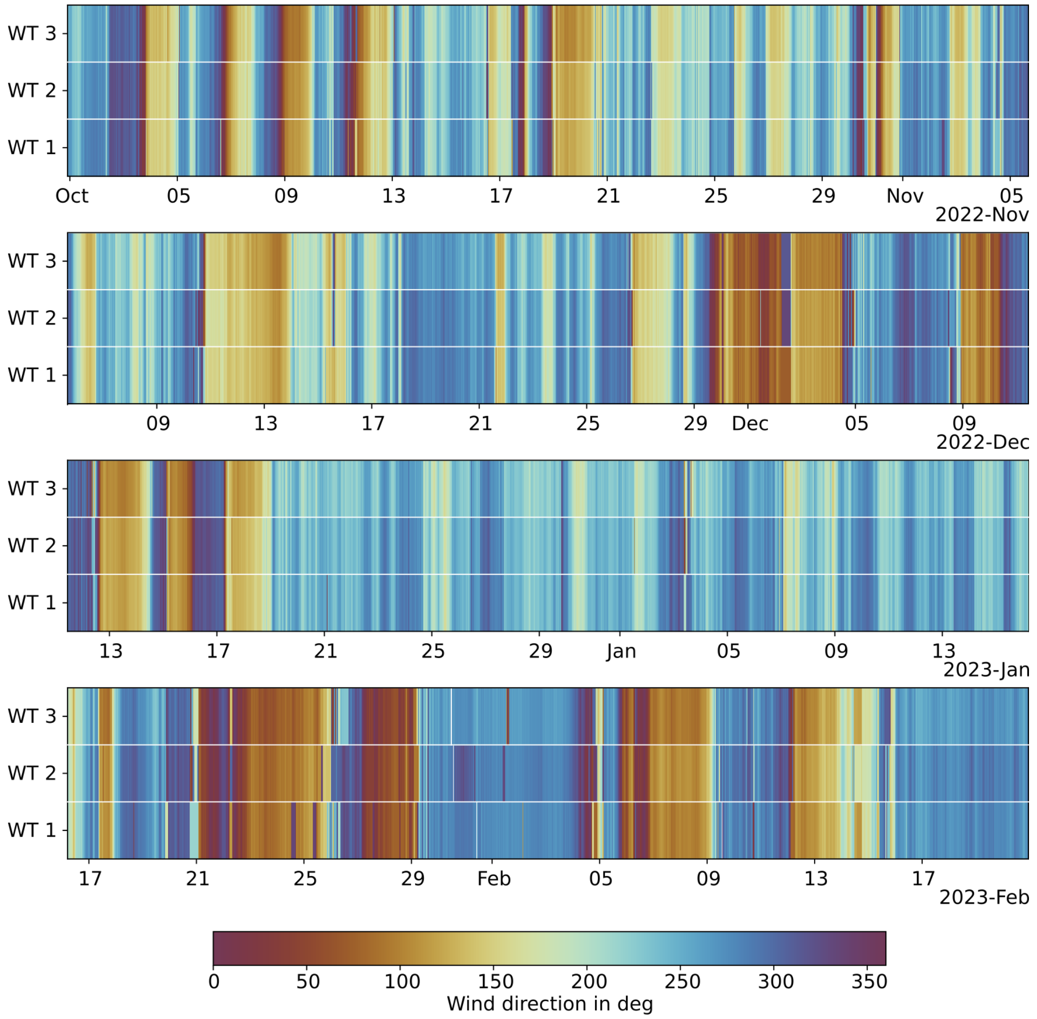


Figure A.2: Wind direction recorded at WF Tegelberg for wind turbines (WTs) *1* to *3* during the measurement campaign IW08 between November 1, 2022 and February 20, 2023.

Appendix B

Simulation of the instrument response by Thomas Forbriger

B.1 VLPtools

```python
#!/usr/bin/env python
# this is <VLPtools.py>
# ------------------------------------------------------------------------
#
# Copyright (c) 2023 by Thomas Forbriger (KIT, GPI, BFO)
#
# a module for handling VLP signal analysis
#
# REVISIONS and CHANGES
#    30/09/2023   V1.0   Thomas Forbriger
#
# ========================================================================
#
import modules.pazfilter as paz
from modules.invresponse import dumpresp
import numpy as np

def groundmotion(stin, inv, f0, quantity="displacement", verbose=True):
    """
    convert raw recordings to ground motion in a specific frequency band

    The two lowermost poles of the instrument are moved to f0.
    The signal values are converted to the specified quantity.

    parameters
    ----------
        stin : obspy.Stream
            three-component time series data
        inv : obspy.Inventory
            station metadata
        f0 : float
            frequency to which lowest poles of response are shifted / Hz
        quantity : str
            kinematic quantity to represent ground modtion
            may be:
            - acceleration
            - displacement
            - velocity
        verbose : bool
            be verbose
    returns
    -------
        obspy.Stream
            converted time series data
    """

    st=stin.copy()
    st.remove_sensitivity()

    if quantity == "displacement":
        st.integrate()
        units="displacement / $\mu$m"
        unitfac=1.e6
    elif quantity == "acceleration":
        st.differentiate()
        units="acceleration / nm s$^{-2}$"
        unitfac=1.e9
    elif quantity == "velocity":
        units="velocity / $\mu$m s$^{-1}$" #nm
        unitfac=1.e6
    else:
        print("ERROR: undefined quantity %s" % quantity)
        exit(3)

    pazsim=paz.PAZBWHP(1./f0,2)
    pazsim["zeros"]=[]

    for tr in st:
        if verbose:
            print("convert %s" % tr.id)
        tr.data *= unitfac
        tr.stats.units=units
        resp=inv.get_response(tr.id, datetime=tr.stats.starttime)
        if verbose:
            dumpresp(resp)
        seispaz=resp.get_paz()
        funits=0.
        if seispaz.pz_transfer_function_type == "LAPLACE (HERTZ)":
            funits=2.*np.pi
        elif seispaz.pz_transfer_function_type == "LAPLACE (RADIANS/SECOND)":
            funits=1.
        else:
            print("ERROR: unexpected type of response parameters!")
            exit()
        if verbose:
            print(seispaz)
        poles=paz.sortpz([(x*funits) for x in seispaz.poles])
        if verbose:
            print(poles)
            print(poles[0:2]) # Change to three if using lennarz seimometer
        paz.printpz(poles[0:2])
        pazseis={’zeros’: [],
                 ’poles’: poles[0:2],
                 ’gain’: 1.0}
        pazfilt=paz.concatenate([pazsim,paz.reciprocal(pazseis)])
        if verbose:
            paz.printsys(pazfilt)
        tr.data=paz.sosfilter(tr.data, pazfilt, tr.stats.delta)

    return st

# ----- END OF VLPtools.py -----
```

B.2 pazfilter

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Functions to handle poles and zeros, and PAZ-dictionaries
(part of CONRAD)

PAZ dictionaries represent transfer functions in the sense of the Laplace
transform of the impulse response function. The poles and the zeros of the
rational function are given by complex values in units of rad/s. Poles and
zeros must either be real or appear in complex conjugate pairs. For a
system to be stable, the real part of the poles must be negative.

For seismometers the gain is understood to be in counts*s/m

For barometers the gain is understood to be in counts/hPa

The dictionary uses three keys:
    zeros
    poles
    gain
"""
#
# Copyright 2021 by Thea Lepage
#
## @package pazfilter
# Functions to handle poles and zeros and PAZ dictionaries
#
# Definitions in file pazfilter.py
#
## @file pazfilter.py
# Functions to handle poles and zeros and PAZ dictionaries
#
# ----
# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General Public License
# along with this program.  If not, see <https://www.gnu.org/licenses/>.
#
# ----
#
# REVISIONS and CHANGES
#  - 27/04/2022   thof  switch from CC0 to GPL
#  - 19/08/2022   thof: import modules as submodules of crd
#

import numpy as np
import matplotlib.pyplot as plt
from operator import itemgetter
from scipy import signal

# =========================================================================
# functions to treat poles and zeros
# ==================================
# -------------------------------------------------------------------------
def sosfilter(data,paz,dt):
    """
    Apply sos-filter on data

    Parameters
    ----------
    data : array of data
    paz : paz-dictionary of filter
    dt : sampling interval

    Returns
    -------
    res : filtered data
    """
    z,p,k=signal.bilinear_zpk(paz["zeros"],paz["poles"],paz["gain"],1./dt)
    sos=signal.zpk2sos(z,p,k)
    res=signal.sosfilt(sos,data)
    return res

# -------------------------------------------------------------------------
# all-pass filter
PAZallpass={"zeros":[], "poles":[], "gain":1.}

# -------------------------------------------------------------------------
# PAZ for taing the derivative
def PAZderivative():
    PAZ={"zeros": [0.],
         "poles": [],
         "gain":1.}
    return PAZ

# -------------------------------------------------------------------------
# PAZ for signal integration
def PAZintegrate():
    PAZ={"zeros": [],
         "poles": [0.],
         "gain":1.}
    return PAZ

# -------------------------------------------------------------------------
# PAZ dictionary for Butterworth high-pass
def PAZBWHP(T0,npoles):
    BWHP={"zeros":np.array([0]*npoles),
          "poles":(2*np.pi/T0)*np.array(signal.buttap(npoles)[1]),
          "gain":1.}
    return BWHP

# -------------------------------------------------------------------------
# PAZ dictionary for Butterworth low-pass
def PAZBWLP(T0,npoles):
    BWHP={"zeros": [],
          "poles":(2*np.pi/T0)*np.array(signal.buttap(npoles)[1]),
          "gain": np.power(2*np.pi/T0,npoles)}
    return BWHP

# -------------------------------------------------------------------------
## Reduce common poles and zeros
def reduce_factors(z,p):
    """
    reduce common poles and zeros after sorting
    Parameters
    ----------
    z : zeros.
    p : poles.

    Returns
    -------
    zclean : reduced zeros.
    pclean : reduced poles.
    """
    z=sortpz(z)
    p=sortpz(p)

    wl=1.e-4 # threshold
    # indexes to reduce
    z_trash=[]
    p_trash=[]
    for k in range(len(z)):
        for j in range(len(p)):
            if ((np.abs(z[k])-np.abs(p[j]))<wl
                and (np.real(z[k])-np.real(p[j]))<wl and j not in p_trash):
                # common values are sorted out
                z_trash.append(k)
                p_trash.append(j)
                break

    zclean=np.delete(z,z_trash[:])
    pclean=np.delete(p,p_trash[:])
    return(zclean,pclean)

# -------------------------------------------------------------------------

## Compute inverse of an LTI system (reciprocal transfer function)
def reciprocal(pazdict):
    """
    compute the inverse of an LTI system, i.e. the reciprocal transfer
    function

    pazdict:    a poles and zeros dictionary

    return: reciprocal system (poles and zeros dictionary)
    """
    reciprocal_system={"poles": pazdict["zeros"],
         "zeros": pazdict["poles"],
         "gain": 1./pazdict["gain"]}
    return reciprocal_system

# -------------------------------------------------------------------------

## Concatenate multiple PAZ dictionaries into one
def concatenate(pazdict):
    """
    concatenate multiple paz-dictionaries into one

    Parameters
    ----------
    pazdict : list of poles and zeros dictionaries.

    Returns
    -------
    RES : poles and zeros dictionary.
    """
    z=np.array([])
    p=np.array([])
    k=1
    for i in pazdict:
        z=np.append(z,i["zeros"])
        p=np.append(p,i["poles"])
        k*=i["gain"]

    z,p=reduce_factors(z,p)

    RES={"zeros":z,
         "poles":p,
         "gain":k}

    return RES

# ========================================================================
# functions to report filter properties by plotting a diagram
# ==========================================================

## Plot poles and zeros of a transfer function
def plot_paz(z,p,station):
    """
    plot poles and zeros of analog transfer function

    Parameters
    ----------
    z : zeros.
    p : poles.
    """
    z_Hz=1./(2.*np.pi)*np.array(z)
    p_Hz=1./(2.*np.pi)*np.array(p)

    # find largest magnitude of any of the complex values to be used to set
    # axes limits; apply a waterlevel
    axis_limit=np.max(np.abs(list(p_Hz)+list(z_Hz)+[1.e-4]))

    # open new plot an display values
    plt.figure()
    plt.xlim(-1.1*axis_limit,1.1*axis_limit)
    plt.ylim(-1.1*axis_limit,1.1*axis_limit)
    plt.plot(np.real(z_Hz), np.imag(z_Hz), 'ob')
    plt.plot(np.real(p_Hz), np.imag(p_Hz), 'xr')
    plt.legend(['zeros', 'poles'], loc=0)
    plt.grid()
    plt.xlabel("real part in Hz")
    plt.ylabel("imaginary part in Hz")
    plt.title('Pole-zero plot at %s' %station)
    plt.tight_layout()

# ------------------------------------------------------------------------

## Plot a bode diagram for a transfer function given by a PAZ dictionary
def bode_plot(pazdict):
    """
    plot a bode diagram

    Parameters
    ----------
    pazdict : a poles and zeros dictionary
    """
    sys=signal.ZerosPolesGain(
        pazdict["zeros"],
        pazdict["poles"],
        pazdict["gain"]
    )
    w, mag, phase=signal.bode(sys)
    plt.figure()
    plt.subplot(211)
    plt.semilogx(w/(2*np.pi), mag)    # Bode magnitude plot
    plt.ylabel("gain")
    plt.grid()
    plt.subplot(212)
    plt.semilogx(w/(2*np.pi), phase/(360.))  # Bode phase plot
    plt.ylabel("phase / $2\pi$")
    plt.xlabel("frequency / Hz")
    plt.grid()
    plt.tight_layout()

# ========================================================================
# functions to analyse poles and zeros
# ====================================

## Compute the value of H(s) for s=i*omega=i*2*pi*f
def H(pazdict, f):
    """
    Compute the value of the transfer function H(2*pi*i*f).

    Parameters
    ----------
    pazdict : poles and zeros dictionary
    f : frequency in Hz

    return : complex value
      value of H(s) at s=2*pi*i*f
    """
    s=2.j*np.pi*f
    H=1.
    for z in pazdict["zeros"]:
        H=H*(s-z)
    for p in pazdict["poles"]:
        H=H/(s-p)
    H=H*pazdict["gain"]
    return H

# ------------------------------------------------------------------------

## Compute damping as a fraction of critical
def damping(pz):
    """
    return damping as a fraction of critical for a pole (or zeros) existing
    in a pair of poles (or zeros) for a transfer function H(s)

    pz: pole or zero of a transfer function H(s) - a complex number

    return: damping as a fraction of critical - a real number
    """

    # threshold for value comparison of floating point values
    # if difference is below the threshold, bost are considered equal
    wl=1.e-10

    # set default in case zero or pole is at the origin
    h=1.
    # compute damping if pole or zero is not at origin
    if np.abs(pz) > wl:
        h=-np.real(pz)/np.abs(pz)
    return h

# -------------------------------------------------------------------------

## Compute eigenfrequency
def eigenfrequency(pz):
    """
    return eigenfrequency for a pole (or zeros) existing in
    a transfer function H(s)

    pz: pole or zero of a transfer function H(s) - a complex number

    return: eigenfrequency / Hz - a real number
    """
    f0=np.abs(pz)/(2.*np.pi)
    return f0

# -------------------------------------------------------------------------

## Pretty print a complex number
def printcomplex(z, u):
    """
    print a complex number in a nice way

    z: complex number
    u: units
    """

    # sign of imaginary part
    if np.imag(z) < 0.:
        isign="-"
    else:
        isign="+"
    print("  (%10.5f %s i%10.5f) %s"
            % (np.real(z), isign, np.abs(np.imag(z)), u))

# -------------------------------------------------------------------------

## Pretty print a complex number
def complex_to_str(z, u):
    """
    return a string representing a complex number in a nice way

    z : complex number
    u : units

    return : str
    """

    # sign of imaginary part
    if np.imag(z) < 0.:
        isign="-"
    else:
        isign="+"
    retval=("(%10.5f %s i%10.5f) %s" % (np.real(z), isign,
                                       np.abs(np.imag(z)), u))
    return retval

# -------------------------------------------------------------------------

## Sort a list of poles or zeros
def sortpz(pz):
    """
    sort a list of poles and zeros by increasing frequency

    Parameters
    ----------
    pz : list of poles or zeros of a transfer function H(s) - complex numbers

    Returns
    -------
    retval : sorted list
    """
    # setup a sort list
    # we sort by absolute value and real value (in case of absolute value
    # being equal)
    pzlist=[]
    for ipz in pz:
        pzlist.append((ipz, np.abs(ipz), np.abs(np.real(ipz))))

    # sort the list
    pzsorted=sorted(pzlist, key=itemgetter(1,2))

    # extract complex number of poles as a return value
    retval=[]
    for ipz in pzsorted:
        retval.append(ipz[0])

    return retval

# ------------------------------------------------------------------------

## return a pretty string representation of a list of poles or zeros
def pz_to_str(pz):
    """
    return a pretty string representation of a list of poles or zeros
    of a transfer function H(s) by specifying eigenfrequncy and damping

    pz : list of complex poles or zeros of a transfer function H(s)
    return : list of strings
    """

    retval=list()
    # threshold for value comparison of floating point values
    # if difference is below the threshold, bost are considered equal
    wl=1.e-10

    # skip if list is empty
    if len(pz) > 0:
        # sort list
        spz=sortpz(pz)

        # iterate through list
        k=0
        while k < len(spz):
            # compute eigenfrequency and damping
            f0=eigenfrequency(spz[k])
            h=damping(spz[k])
            # if damping is not 1, i.e. pz-value is not on real axis, we
            # expect a pair of complex conjugate values
            expectpair=(np.abs(1.-h) > wl)
            # is there a partner for this value
            if ((expectpair) and ((k+1) >=len(spz))):
                print("ERROR in printpz: values do not appear in pairs!")
                exit(3)
            # check if partner matches and print
            if ((expectpair)
                    and (np.abs(f0-eigenfrequency(spz[k+1])) < wl)
                    and (np.abs(np.imag(spz[k]+spz[k+1])) < wl)):
                # report second order system
                if (f0 > wl):
                    retval.append(
                        "pair at   f0=%10.5fHz   T0=%10.5fs   h=%10.5f" %
                        (f0, 1./f0, h)
                    )
                else:
                    retval.append("pair at   f0=%10.5fHz" % f0)
                # skip the next entry
                k=k+1
            else:
                # report first order system
                if (f0 > wl):
                    retval.append("single at f0=%10.5fHz   T0=%10.5fs" %
                            (f0, 1./f0))
                else:
                    retval.append("single at f0=%10.5fHz" % f0)

            # proceed to next entry
            k=k+1

    return retval

# -------------------------------------------------------------------------

## Pretty print a list of poles or zeros
def printpz(pz):
    """
    print a list of poles or zeros of a transfer function H(s) specifying
    eigenfrequncy and damping

    pz: list of poles or zeros of a transfer function H(s) - complex numbers
    """

    listofpz=pz_to_str(pz)
    for l in listofpz:
        print("  %s" % l)

# -------------------------------------------------------------------------

## create a pretty print report of a poles and zeros dictionary
def pazdict_to_str(pazdict):
    """
    create a pretty print report of a poles and zeros dictionary

    Parameters
    ----------
    pazdict : dictionary
        PAZ dictionary defining an LTI system transfer function

    return : list of strings
    """

    retval=list()
    # extract poles and zeros
    sz=sortpz(pazdict["zeros"])
    sp=sortpz(pazdict["poles"])
    sk=pazdict["gain"]
    # print summary values
    retval.append("number of zeros:  %d" % len(sz))
    retval.append("number of poles:  %d" % len(sp))
    retval.append("numerator factor: %f" % sk)

    # dump complex zeros and poles
    if len(sz) > 0:
        retval.append("")
        retval.append("complex zeros (numerator zeros):")
        for z in sz:
            retval.append(complex_to_str(z, "rad/s"))

    if len(sp) > 0:
        retval.append("")
        retval.append("complex poles (denominator zeros):")
        for p in sp:
            retval.append(complex_to_str(p, "rad/s"))

    # dump complex zeros and poles as eigenfrequency and damping
    if len(sz) > 0:
        retval.append("")
        retval.append("zeros (numerator zeros):")
        retval=retval+pz_to_str(sz)

    if len(sp) > 0:
        retval.append("")
        retval.append("poles (denominator zeros):")
        retval=retval+pz_to_str(sp)

    return retval

# -------------------------------------------------------------------------

## Pretty print a poles and zeros (PAZ) dictionary
def printsys(pazdict):
    """
    print parameters of a transfer function H(s)

    Parameters
    ----------
    pazdict : dictionary
        PAZ dictionary defining an LTI system transfer function
    """

    listofstrings=pazdict_to_str(pazdict)
    for l in listofstrings:
        print("  %s" % l)

# ----- END OF pazfilter.py -----
```

B.3 invresponse

```
#!/usr/bin/env python
# this is <invresponse.py>
# -------------------------------------------------------------------------
#
# Copyright (c) 2023 by Thomas Forbriger (KIT, GPI, BFO)
#
# module to handle inventory data
#
# REVISIONS and CHANGES
#    22/09/2023   V1.0   Thomas Forbriger
```

```
#
# ==========================================================================
#

import numpy as np
from obspy import read, read_inventory
from obspy.core import AttribDict, UTCDateTime
from obspy.geodetics.base import locations2degrees
import matplotlib.pyplot as plt

def dumpresp(resp):
    """
    resp: obspy response object obtained through get_response function of
    inventory
    """
    print(resp)
    seis_paz=resp.get_paz()
    print(seis_paz)
    print("'"+seis_paz.pz_transfer_function_type+"'")
# factor to scale to units of Hertz (set zo 0, if units are unknown)
    funits=0.
    if seis_paz.pz_transfer_function_type == "LAPLACE (HERTZ)":
        funits=1
    elif seis_paz.pz_transfer_function_type == "LAPLACE (RADIANS/SECOND)":
        funits=0.5/np.pi
    else:
        print("ERROR: unexpected type of response parameters!")
    paz_dict={"zeros": seis_paz.zeros,
              "poles": seis_paz.poles,
              "gain": seis_paz.stage_gain}
    print(paz_dict)
    for t in ("zeros", "poles"):
        print("\nlist of %s:" % t)
        for x in paz_dict[t]:
            f=np.abs(x)*funits
            if ((f < 1.e-10) or (f >= 1.)):
                fstring="%10.4f Hz" % f
            else:
                fstring="%10.4f s" % (1./f)
            print(fstring, "at", x*funits, "Hz")

# ----- END OF invresponse.py -----
```

Appendix C

Scattering network

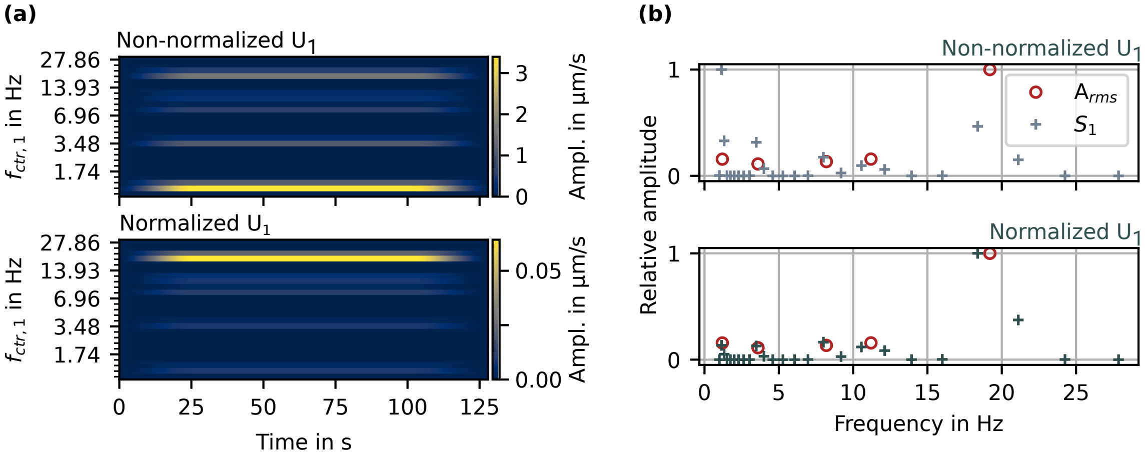


Figure C.1: Effects of the $L1$-normalization of the filter bank on signal $x_{syn,1}$. (a) shows the unnormalized first-order scalogram U_1 (top), and normalized U_1 (bottom), respectively. (b) depicts the relative root mean square amplitudes of the input signal, marked with red circles, compared to the relative first-order scattering coefficients S_1 (plus icon), not normalized (top) and normalized (bottom). In both non-normalized subplots, the higher amplitudes are attenuated. Whereas, the relative amplitudes of the normalized U_1 match the relative amplitudes of the input signal. However, the frequency content in the lower plot of (b) does not perfectly match the frequency content of the signal. This is due to the fact that the center frequencies of the filter bank do not perfectly match the frequencies of the signal. For more details see Section 3.3.3.

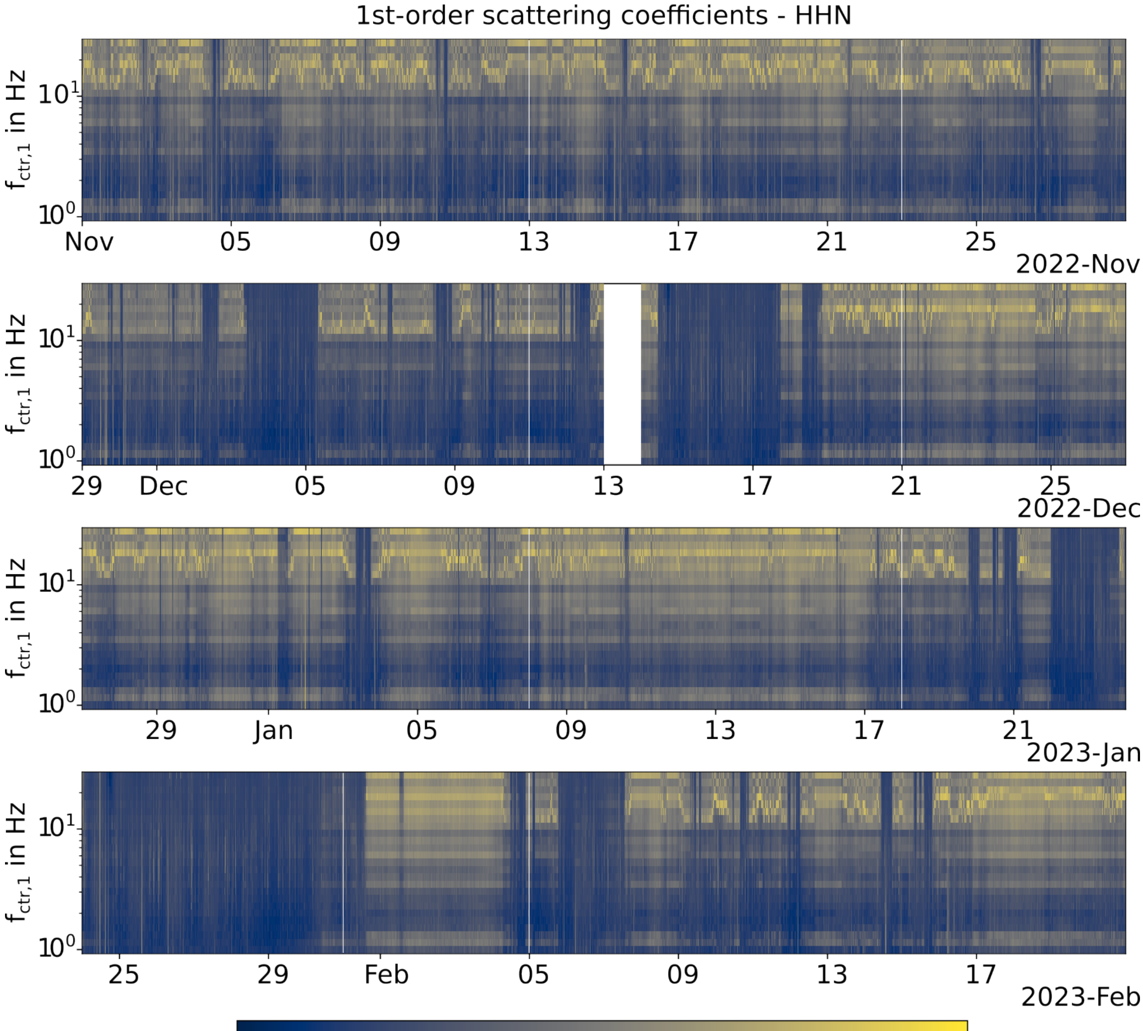


Figure C.2: First-order scattering coefficients S_1 of the N-component recorded at IW08B, for further explanation see Fig. 3.18.

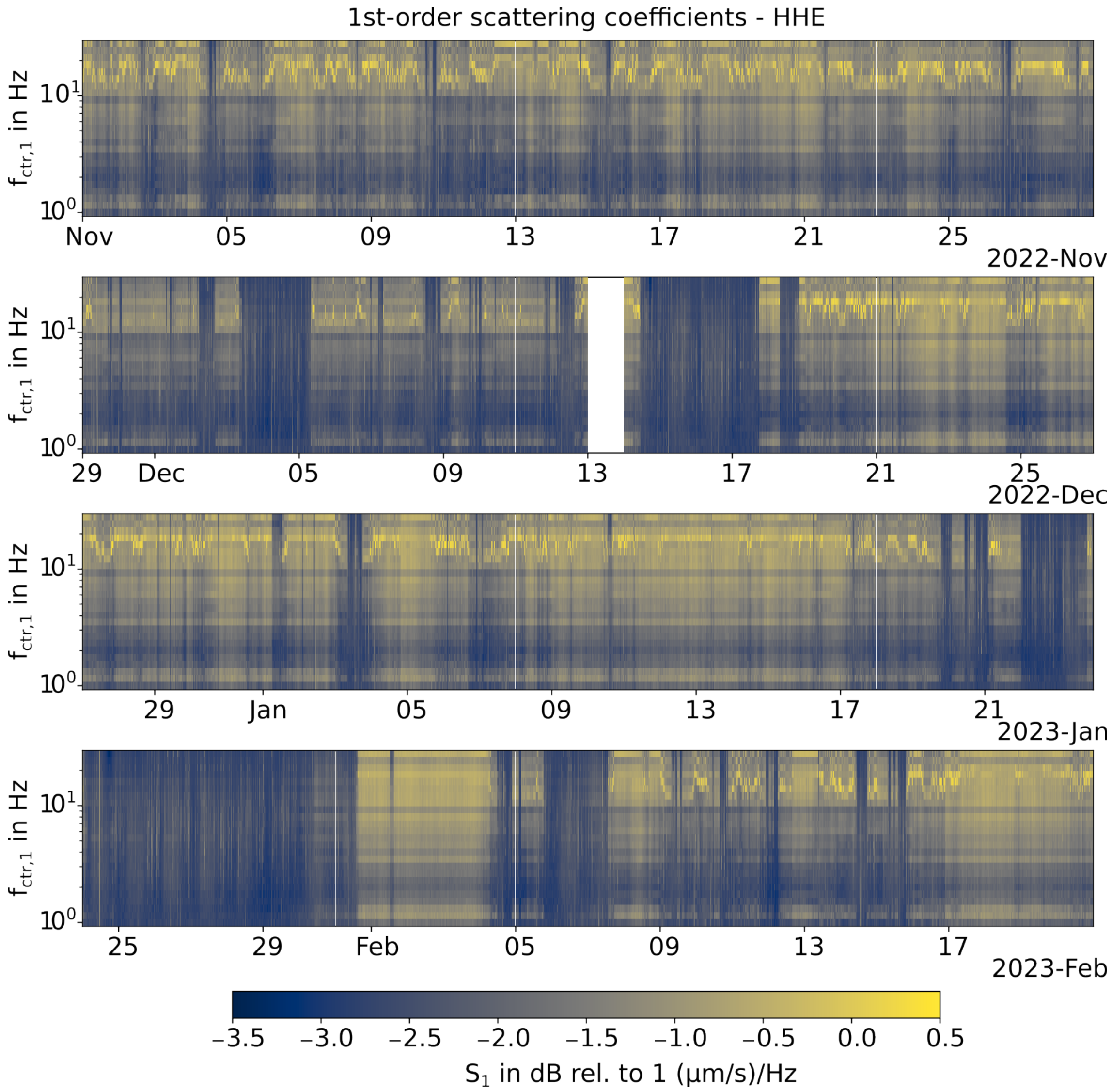
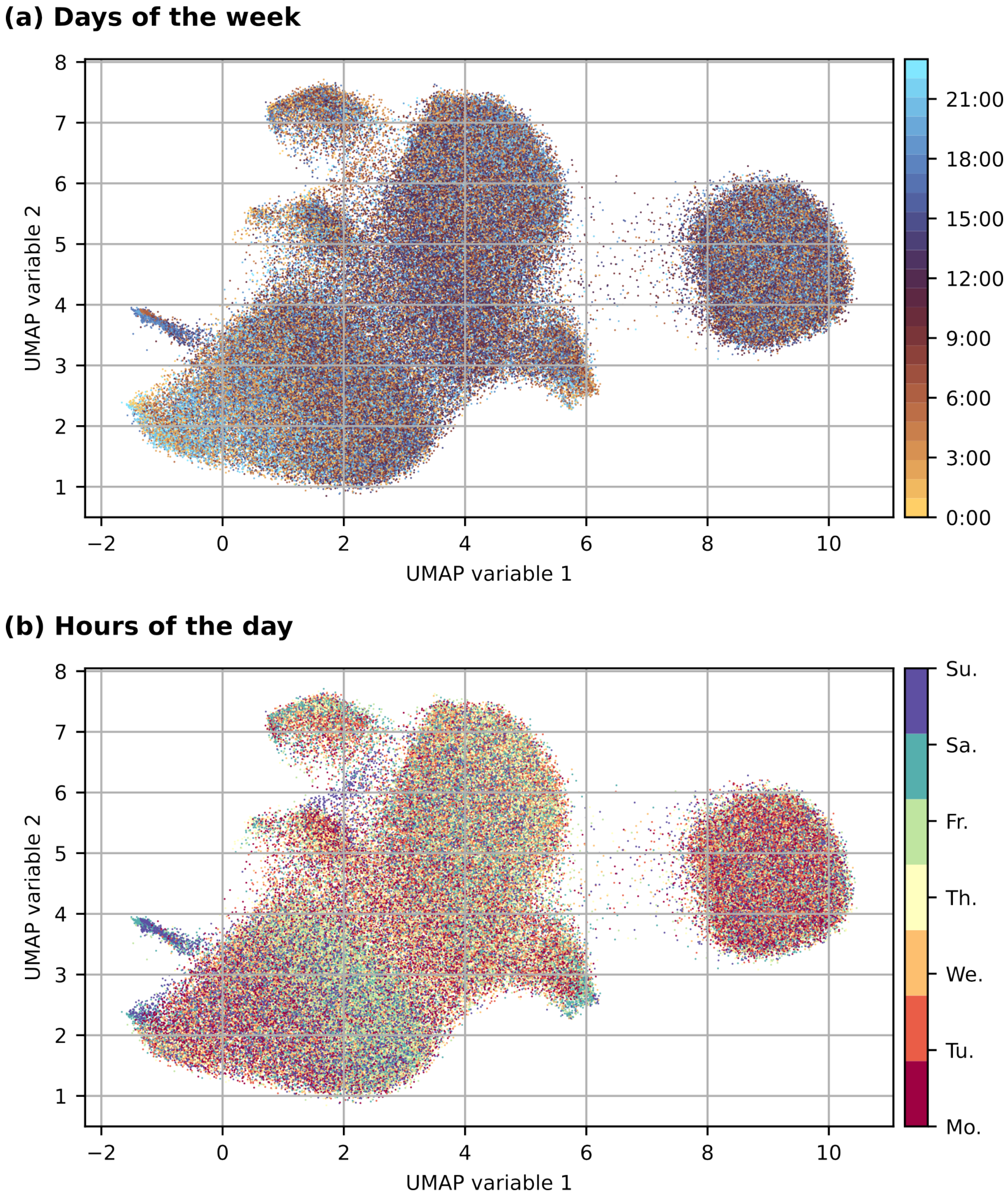


Figure C.3: First-order scattering coefficients S_1 of the E-component recorded at IW08B, for further explanation see Fig. 3.18.

Appendix D

UMAP



(a) Days of the week

(b) Hours of the day

Figure D.1: UMAP color-coded with (a) days of the week and (b) hours of the day.

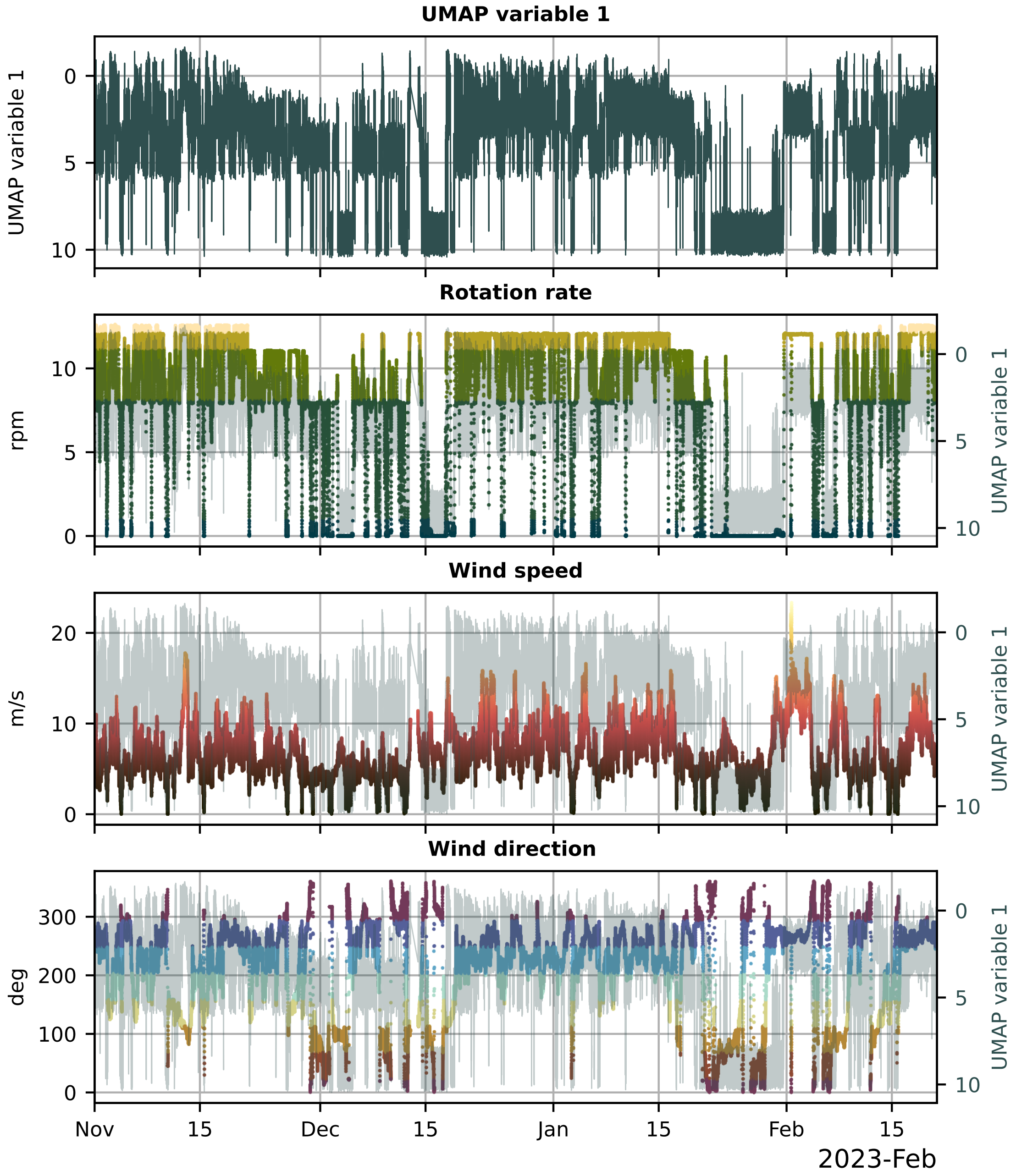


Figure D.2: UMAP variable 1 in comparison to rotation rate, wind speed, and direction. The UMAP variable are displayed inversely.

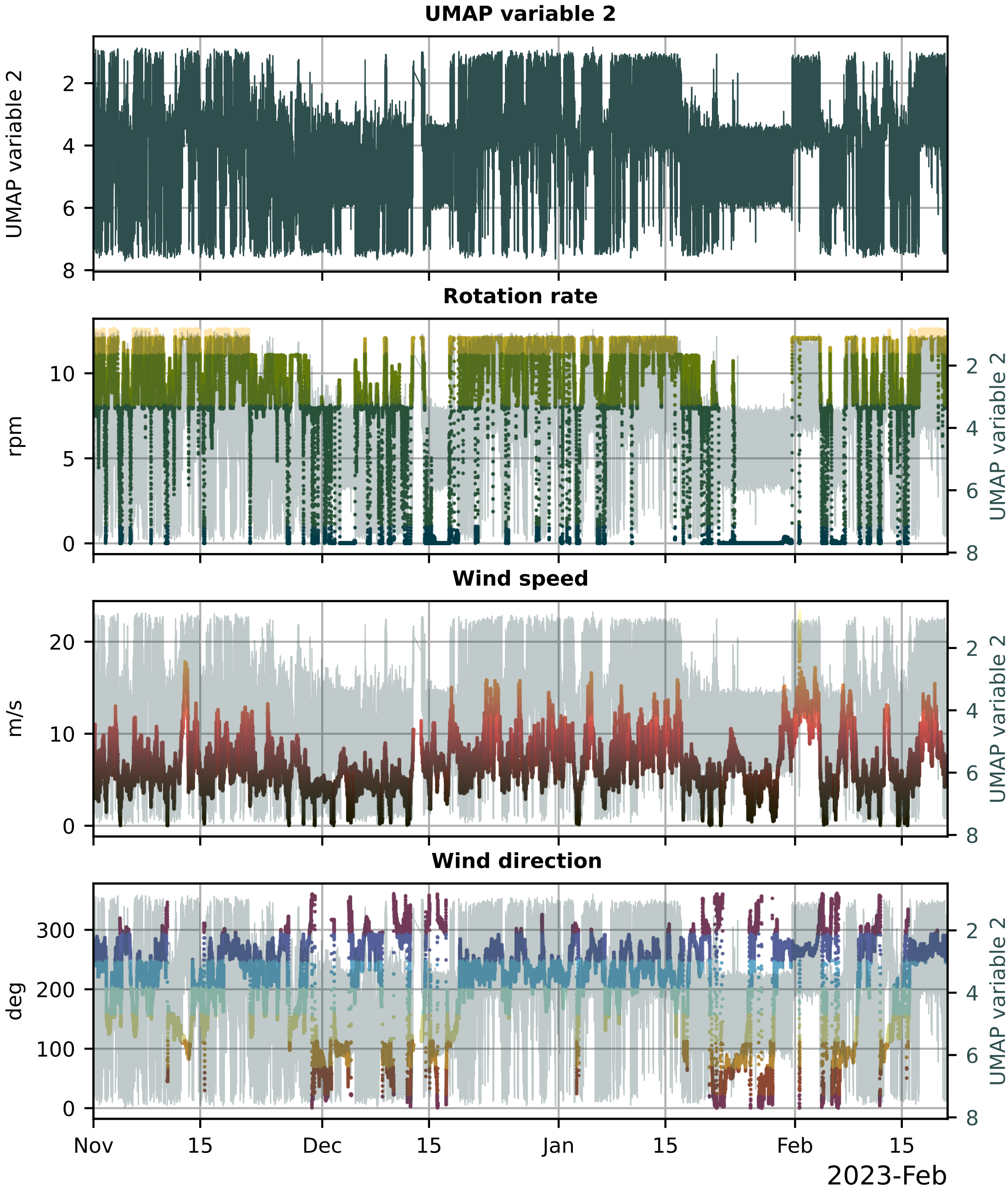


Figure D.3: UMAP variable 2 in comparison to rotation rate, wind speed, and direction. The UMAP variable are displayed inversely.

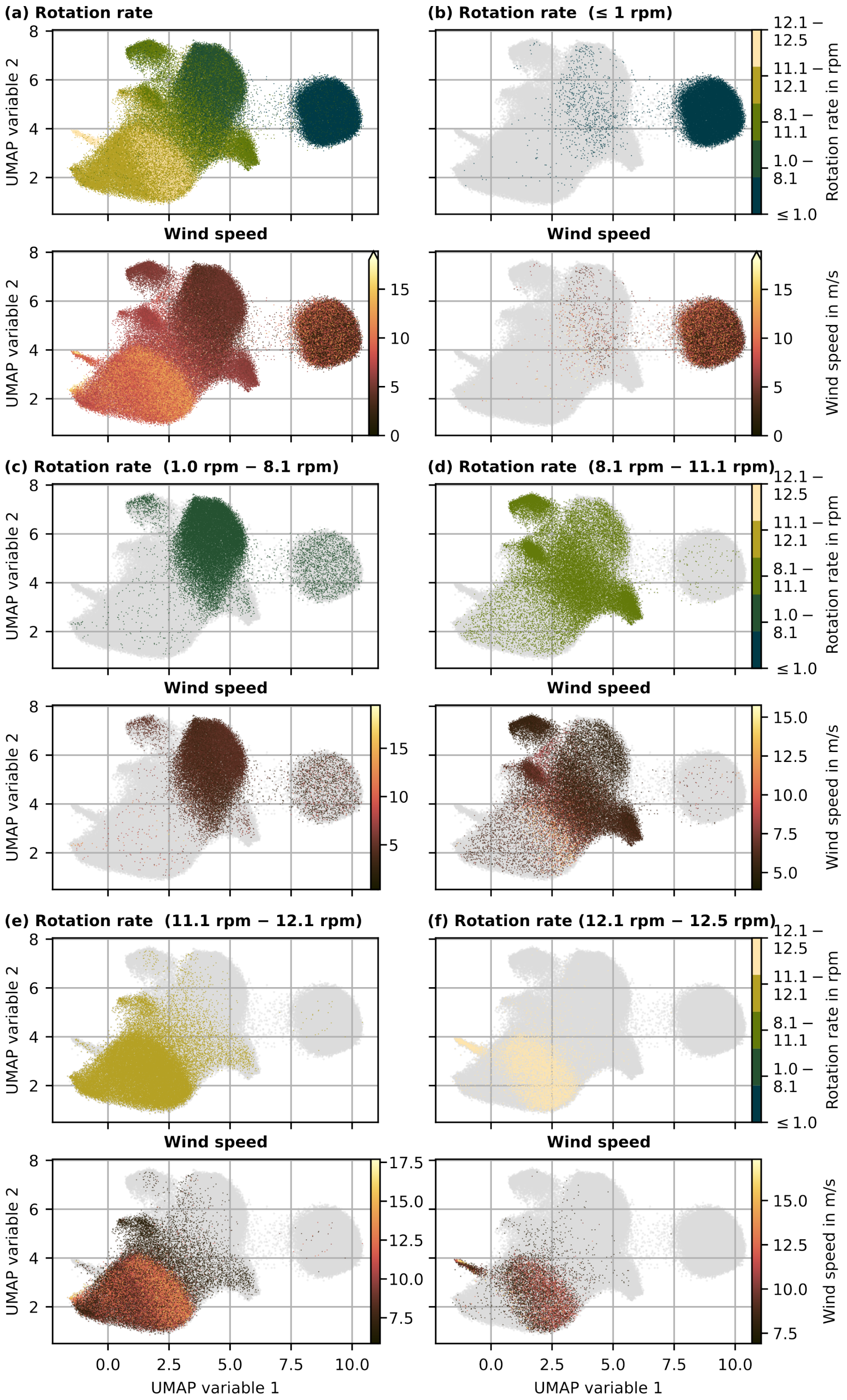


Figure D.4: UMAP atlas color-coded with rotation rate, wind speed, and direction, and filtered by the different operation modes of *WT 3*. (a) Entire range of rotation rates. (b) Rotation rate between 0 rpm and 1 rpm, i.e. *WT 3* was not in operation. (c) Rotation rate between 1 rpm and 8.1 rpm, i.e. *WT 3* was operated at partial load. (d) Rotation rate between 8.1 rpm and 11.1 rpm, i.e. *WT 3* was operated at full load with lower noise reduced mode. (e) Rotation rate between 11.1 rpm and 12.1 rpm, i.e. *WT 3* was operated at full load with high noise reduced mode. (f) Rotation rate between 12.1 rpm and 12.5 rpm, i.e. *WT 3* was operated at full load.

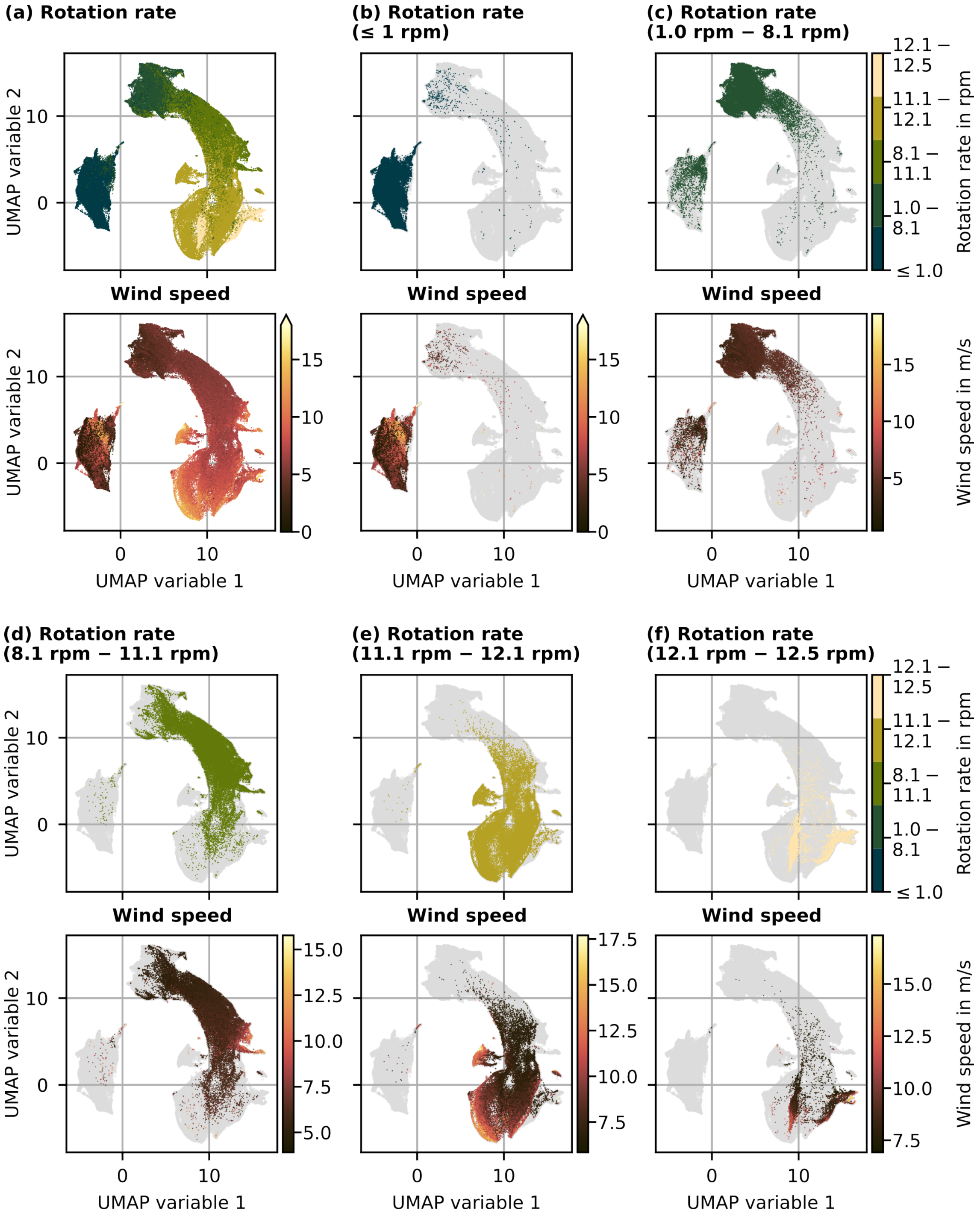


Figure D.5: UMAP atlas of the first-order scattering coefficient color-coded with rotation rate, wind speed, and direction, and filtered by the different operation modes of *WT 3*. (a) Entire range of rotation rates. (b) Rotation rate between 0 rpm and 1 rpm, i.e. *WT 3* was not in operation. (c) Rotation rate between 1 rpm and 8.1 rpm, i.e. *WT 3* was operated at partial load. (d) Rotation rate between 8.1 rpm and 11.1 rpm, i.e. *WT 3* was operated at full load with lower noise reduced mode. (e) Rotation rate between 11.1 rpm and 12.1 rpm, i.e. *WT 3* was operated at full load with high noise reduced mode. (f) Rotation rate between 12.1 rpm and 12.5 rpm, i.e. *WT 3* was operated at full load.

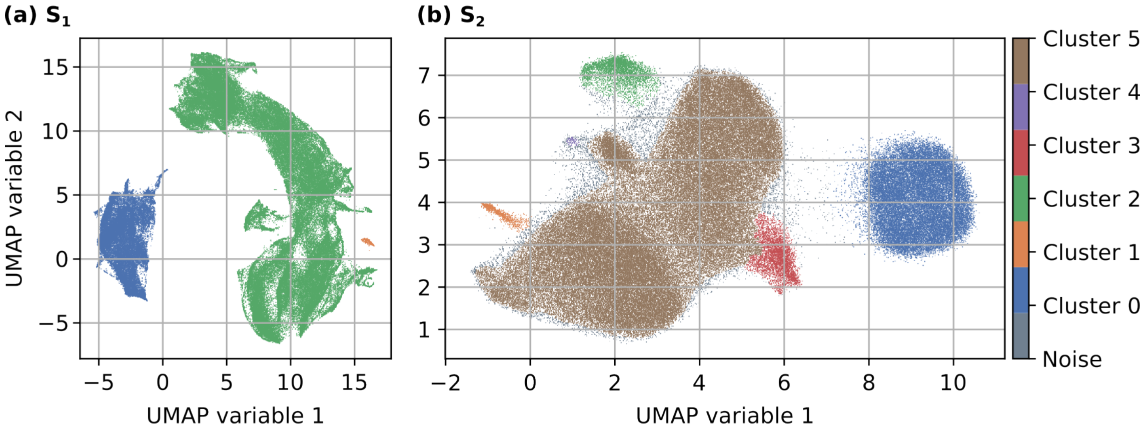


Figure D.6: UMAP atlas of first- (a) and second-order scattering coefficients (b) color coded with clusters calculated with HDBSCAN using `min_cluster_size` of 100.

Appendix E

HDBSCAN

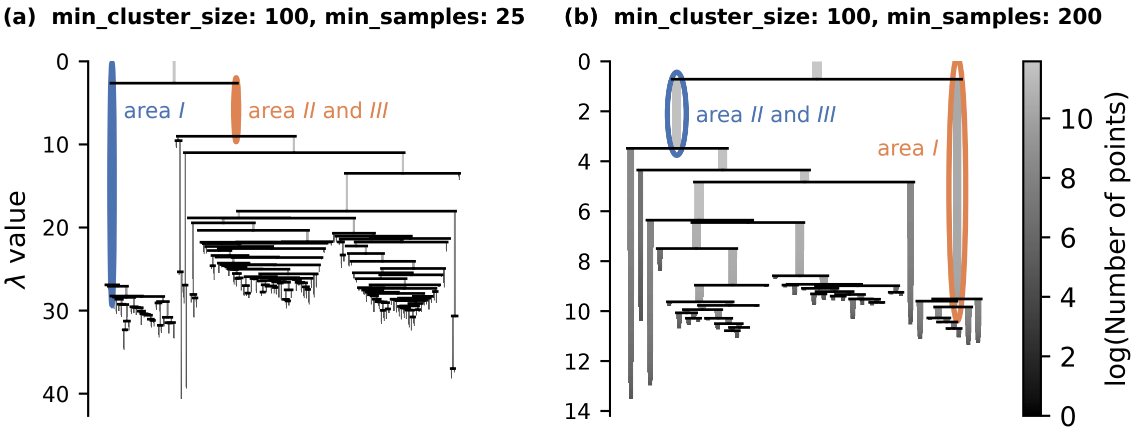


Figure E.1: Example of the same areas within the UMAP atlas being grouped into two clusters but labeled differently. A detailed explanation can be found in the text.

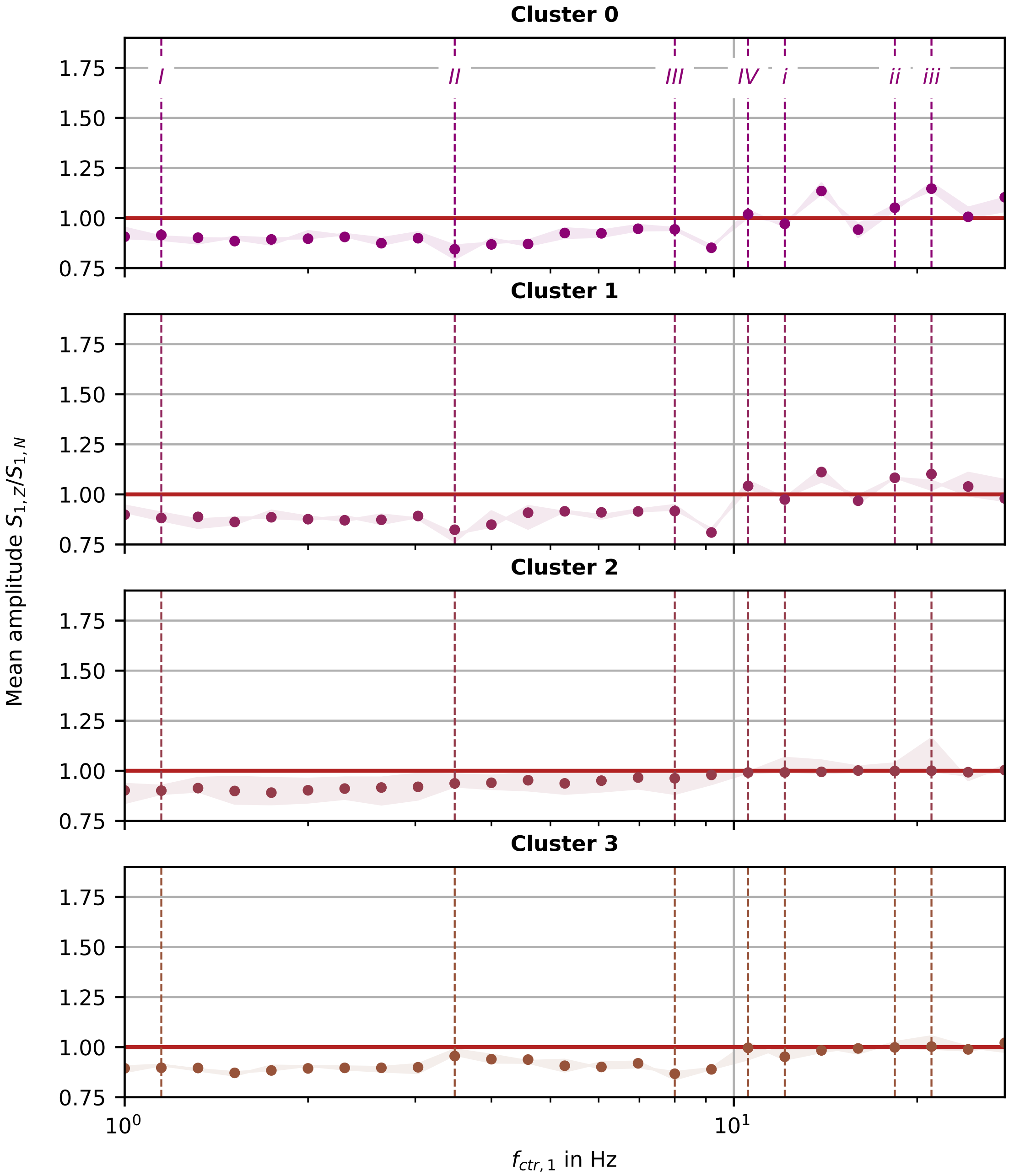


Figure E.2: Mean amplitude of the Z-component of the first-order scattering coefficients $S_{1,Z}$ compared to the N-component $S_{1,N}$. A value less than one indicates that $S_{1,Z}$ has a larger amplitude than $S_{1,N}$, and values greater than one indicate that $S_{1,N}$ is larger. Out of the eleven clusters, only clusters **0**, **1**, **2**, and **3** are depicted.

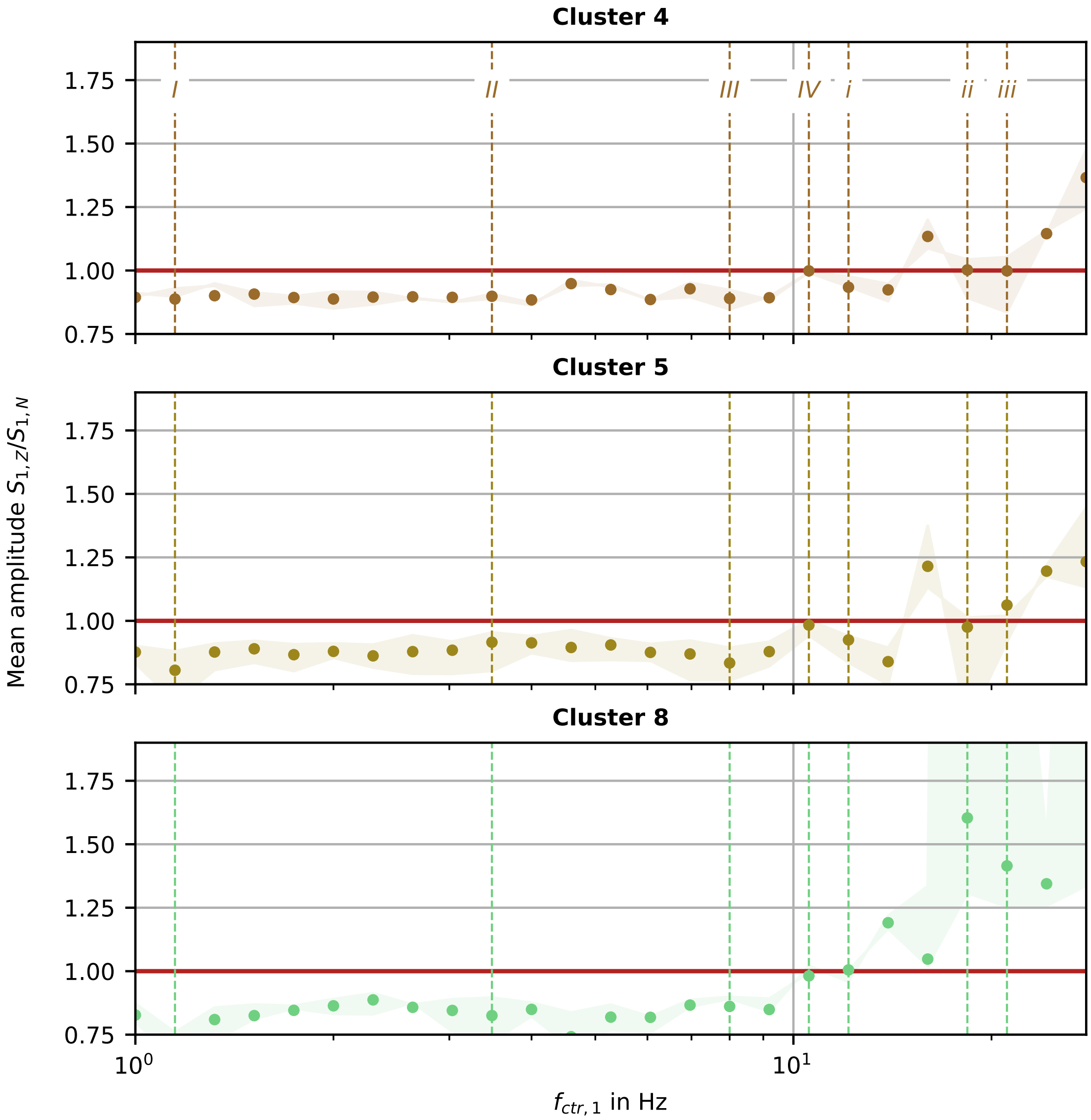


Figure E.3: Mean amplitude of the Z-component of the first-order scattering coefficients $S_{1,Z}$ compared to the N-component $S_{1,N}$. A value less than one indicates that $S_{1,Z}$ has a larger amplitude than $S_{1,N}$, and values greater than one indicate that $S_{1,N}$ is larger. Out of the eleven clusters, only clusters **4**, **5**, and **8** are depicted.

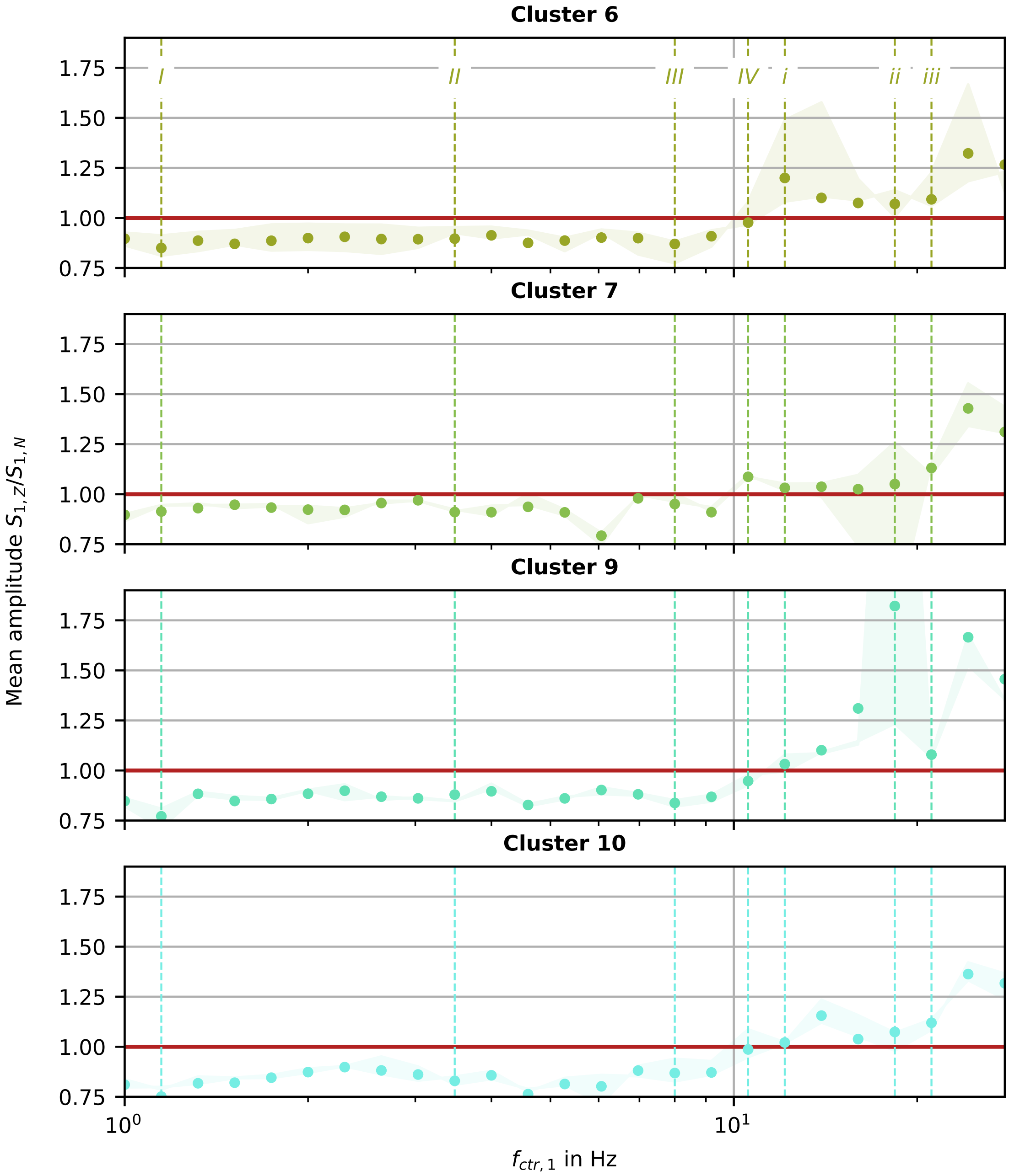


Figure E.4: Mean amplitude of the Z-component of the first-order scattering coefficients $S_{1,Z}$ compared to the N-component $S_{1,N}$. A value less than one indicates that $S_{1,Z}$ has a larger amplitude than $S_{1,N}$, and values greater than one indicate that $S_{1,N}$ is larger. Out of the eleven clusters, only clusters **6**, **7**, **9**, and **10** are depicted.

References

Amit, Y. and Trouvé, A. (Nov. 2007). “POP: Patchwork of Parts Models for Object Recognition”. *IEEE Transactions on Signal Processing* 75, pp. 267–282. DOI: 10.1007/s11263-006-0033-9.

Anden, J. and Mallat, S. (Aug. 2014). “Deep Scattering Spectrum”. *IEEE Transactions on Signal Processing* 62.16, pp. 4114–4128. DOI: 10.1109/tsp.2014.2326991.

Barkaoui, S., Lognonné, P., Kawamura, T., et al. (Dec. 2021). “Anatomy of Continuous Mars SEIS and Pressure Data from Unsupervised Learning”. *Bulletin of the Seismological Society of America* 111.6, pp. 2964–2981. DOI: 10.1785/0120210095.

Becht, E., McInnes, L., Healy, J., Dutertre, C.-A., Kwok, I. W. H., Ng, L. G., Ginhoux, F., and Newell, E. W. (Jan. 2019). “Dimensionality Reduction for Visualizing Single-Cell Data Using UMAP”. *Nature Biotechnology* 37.1, pp. 38–44. DOI: 10.1038/nbt.4314.

Bellman, R. E. (1961). “Adaptive Control Processes”. *Princeton University Press.*

Beyreuther, M., Barsch, R., Krischer, L., Megies, T., Behr, Y., and Wassermann, J. (May 2010). “ObsPy: A Python Toolbox for Seismology”. *Seismological Research Letters* 81.3, pp. 530–533. DOI: 10.1785/gssrl.81.3.530.

Bishop, C. M. (2006). *Pattern recognition and machine learning.* Information science and statisticsComputer science. New York, NY: Springer. ISBN: 0387310738; 9780387310732.

Bloem, P. (June 2023). *Unraveling Principal Component Analysis.* ISBN: 979-8850607159.

Blumendeller, E., Gaßner, L., Müller, F. J., Pohl, J., Hübner, G., Ritter, J., and Cheng, P. W. (2023). “Quantification of amplitude modulation of wind turbine emissions from acoustic and ground motion recordings”. *Acta Acustica* 7, p. 55. DOI: 10.1051/aacus/2023047.

Brown, J. C. (Jan. 1991). “Calculation of a Constant Q Spectral Transform”. *The Journal of the Acoustical Society of America* 89.1, pp. 425–434. DOI: 10.1121/1.400476.

Bruna, J. and Mallat, S. (Aug. 2013). “Invariant Scattering Convolution Networks”. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35.8, pp. 1872–1886. DOI: 10.1109/TPAMI.2012.230.

Bundesnetzagentur (2022). *Marktstammdatenregister.* URL: marktstammdatenregister.de/MaStR/ (visited on 06/10/2023).

Campello, R. J. G. B., Moulavi, D., and Sander, J. (2013). “Density-Based Clustering Based on Hierarchical Density Estimates”. In: *Advances in Knowledge Discovery and Data Mining.* Ed. by D. Hutchison, T. Kanade, J. Kittler, et al. Vol. 7819. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 160–172. ISBN: 978-3-642-37455-5 978-3-642-37456-2. DOI: 10.1007/978-3-642-37456-2_14.

Cao, J., Spielmann, M., Qiu, X., et al. (Feb. 2019). “The Single-Cell Transcriptional Landscape of Mammalian Organogenesis”. *Nature* 566.7745, pp. 496–502. DOI: 10.1038/s41586-019-0969-x.

Charléty, J., Cuenot, N., Dorbath, L., Dorbath, C., Haessler, H., and Frogneux, M. (Dec. 2007). “Large Earthquakes during Hydraulic Stimulations at the Geothermal Site of Soultz-sous-Forêts”. *International Journal of Rock Mechanics and Mining Sciences* 44.8, pp. 1091–1105. DOI: 10.1016/j.ijrmms.2007.06.003.

Deutsche WindGuard GmbH (Jan. 2023). *Status of Onshore Wind Energy Development in Germany – Year 2023*. Tech. rep.

Dumoulin, V. and Visin, F. (2018). *A guide to convolution arithmetic for deep learning*. arXiv: 1603.07285 [stat.ML].

EEG 2023 (Feb. 2024). *Gesetz für den Ausbau Erneuerbarer Energien*. URL: https://www.gesetze-im-internet.de/eeg_2014/ (visited on 04/17/2024).

Erdbeben bei Albstadt, Zollernalbkreis, BW (Jan. 2023). URL: https://erdbeben.led-bw.de/erdbeben/230127_2045 (visited on 04/25/2024).

Ezugwu, A. E., Ikotun, A. M., Oyelade, O. O., Abualigah, L., Agushaka, J. O., Eke, C. I., and Akinyelu, A. A. (2022). “A comprehensive survey of clustering algorithms: State-of-the-art machine learning applications, taxonomy, challenges, and future research prospects”. *Engineering Applications of Artificial Intelligence* 110, p. 104743. DOI: https://doi.org/10.1016/j.engappai.2022.104743.

Garzon, M., Yang, C.-C., Venugopal, D., Kumar, N., Jana, K., and Deng, L.-Y., eds. (2022). *Dimensionality Reduction in Data Science*. Cham: Springer International Publishing. DOI: 10.1007/978-3-031-05371-9.

Gaßner, L., Blumendeller, E., Müller, F. J., Wigger, M., Rettenmeier, A., Cheng, P. W., Hübner, G., Ritter, J., and Pohl, J. (Apr. 2022). “Joint analysis of resident complaints, meteorological, acoustic, and ground motion data to establish a robust annoyance evaluation of wind turbine emissions”. *Renewable Energy* 188, pp. 1072–1093. DOI: 10.1016/j.renene.2022.02.081.

Gaßner, L., Gärtner, M. A., and Ritter, J. (Nov. 2023). “Simulation of ground motion emissions from wind turbines in low mountain ranges: implications for amplitude decay prediction”. *J Seismol.* DOI: 10.1007/s10950-023-10172-6.

Gaßner, L. and Ritter, J. (Jan. 2023a). *Wind turbine emissions: Interdisciplinary analysis and mitigation approaches – Project Inter-Wind. Description of datasets “Inter-Wind“ and "Inter-Wind (recorder log Files)"*. Tech. rep. Karlsruhe Institute of Technology (KIT), Geophysical Institute (GPI).

Gaßner, L. and Ritter, J. (July 2023b). “Ground motion emissions due to wind turbines: observations, acoustic coupling, and attenuation relationships”. *Solid Earth* 14.7, pp. 785–803. DOI: 10.5194/se-14-785-2023.

Harris, F. (1978). “On the Use of Windows for Harmonic Analysis with the Discrete Fourier Transform”. *Proceedings of the IEEE* 66.1, pp. 51–83. DOI: 10.1109/PROC.1978.10837.

Hensch, M., Dahm, T., Ritter, J., Heimann, S., Schmidt, B., Stange, S., and Lehmann, K. (Mar. 2019). “Deep Low-Frequency Earthquakes Reveal Ongoing Magmatic Recharge beneath Laacher See Volcano (Eifel, Germany)”. *Geophysical Journal International* 216.3, pp. 2025–2036. DOI: 10.1093/gji/ggy532.

Heuel, J. and Friederich, W. (June 2022). “Suppression of Wind Turbine Noise from Seismological Data Using Nonlinear Thresholding and Denoising Autoencoder”. *Journal of Seismology* 26.5, pp. 913–934. DOI: 10.1007/s10950-022-10097-6.

Kong, Q., Allen, R. M., Schreier, L., and Kwon, Y.-W. (Feb. 2016). “MyShake: A Smartphone Seismic Network for Earthquake Early Warning and Beyond”. *Science Advances* 2.2, e1501055. DOI: 10.1126/sciadv.1501055.

Krischer, L., Megies, T., Barsch, R., Beyreuther, M., Lecocq, T., Caudron, C., and Wassermann, J. (May 2015). “ObsPy: a bridge for seismology into the scientific Python ecosystem”. *Computational Science &amp Discovery* 8.1, p. 014003. DOI: 10.1088/1749-4699/8/1/014003.

Lilly, J. M. (Apr. 2017). “Element Analysis: A Wavelet-Based Method for Analysing Time-Localized Events in Noisy Time Series”. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 473.2200, p. 20160776. DOI: 10.1098/rspa.2016.0776.

Liu, Y., San Liang, X., and Weisberg, R. H. (Dec. 2007). “Rectification of the Bias in the Wavelet Power Spectrum”. *Journal of Atmospheric and Oceanic Technology* 24.12, pp. 2093–2102. DOI: 10.1175/2007JTECHO511.1.

Mallat, S. G. (2009). *A Wavelet Tour of Signal Processing: The Sparse Way.* 3rd ed. Amsterdam ; Boston: Elsevier/Academic Press. ISBN: 978-0-12-374370-1.

Mallat, S. (Aug. 2010). “Recursive Interferometric Representation”. In: *18th European Signal Processing Conference (EUSIPCO-2010)*, pp. 716–720.

Mallat, S. (2012). “Group Invariant Scattering”. *Commun. Pure Appl. Math.* 65.10, pp. 1331–1398. DOI: 10.1002/cpa.21413.

McInnes, L. (2018). *How UMAP Works.* URL: https://umap-learn.readthedocs.io/en/latest/how_umap_works.html#id8 (visited on 03/21/2024).

McInnes, L. and Healy, J. (Nov. 2017). “Accelerated Hierarchical Density Based Clustering”. In: *2017 IEEE International Conference on Data Mining Workshops (ICDMW).* New Orleans, LA: IEEE, pp. 33–42. DOI: 10.1109/ICDMW.2017.12.

McInnes, L., Healy, J., and Astels, S. (2016). *How HDBSCAN Works.* URL: https://hdbscan.readthedocs.io/en/latest/how_hdbscan_works.html (visited on 03/21/2024).

McInnes, L., Healy, J., and Astels, S. (Mar. 2017). “hdbscan: Hierarchical Density Based Clustering”. *The Journal of Open Source Software* 2.11, p. 205. DOI: 10.21105/joss.00205.

McInnes, L., Healy, J., and Melville, J. (2018). *UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction.* arXiv: 1802.03426 [stat.ML].

Megies, T., Beyreuther, M., Barsch, R., Krischer, L., and Wassermann, J. (Apr. 2011). “ObsPy – What Can It Do for Data Centers and Observatories?” *Annals of Geophysics* 54.1. DOI: 10.4401/ag-4838.

Morel, R., Rochette, G., Leonarduzzi, R., Bouchaud, J.-P., and Mallat, S. (2023). *Scale Dependencies and Self-Similar Models with Wavelet Scattering Spectra.* arXiv: 2204.10177 [physics.data-an].

Münchmeyer, J., Woollam, J., Rietbrock, A., et al. (Jan. 2022). “Which Picker Fits My Data? A Quantitative Evaluation of Deep Learning Based Seismic Pickers”. *Journal of Geophysical Research: Solid Earth* 127.1, e2021JB023499. DOI: 10.1029/2021JB023499.

Munkres, J. R. (1984). *Elements of Algebraic Topology.* Redwood city (calif) [etc.]: Addison-Wesley Publishing Company. ISBN: 978-0-201-04586-4.

Murtagh, F. and Contreras, P. (Jan. 2012). “Algorithms for Hierarchical Clustering: An Overview”. *WIREs Data Mining and Knowledge Discovery* 2.1, pp. 86–97. DOI: 10.1002/widm.53.

Nagel, S., Zieger, T., Luhmann, B., Knödel, P., Ritter, J., and Ummenhofer, T. (June 2019). “Erschütterungsemissionen von Windenergieanlagen”. *Stahlbau* 88.6, pp. 559–573. DOI: 10.1002/stab.201900039.

Nagel, S., Zieger, T., Luhmann, B., Knödel, P., Ritter, J., and Ummenhofer, T. (June 2021). “Ground motions induced by wind turbines”. *Civil Engineering Design* 3.3, pp. 73–86. DOI: 10.1002/cend.202100015.

Neuffer, T., Kremers, S., Meckbach, P., and Mistler, M. (Apr. 2021). “Characterization of the seismic wave field radiated by a wind turbine”. *Journal of Seismology* 25.3, pp. 825–844. DOI: 10.1007/s10950-021-10003-6.

Ochoa, L. H., Niño, L. F., and Vargas, C. A. (Jan. 2018). “Fast Magnitude Determination Using a Single Seismological Station Record Implementing Machine Learning Techniques”. *Geodesy and Geodynamics* 9.1, pp. 34–41. DOI: 10.1016/j.geog.2017.03.010.

Project DB Miss (Jan. 2024). URL: https://www.uni-muenster.de/Physik.GP/dbmiss/project.html (visited on 05/11/2024).

Reddy, R. and Nair, R. R. (Oct. 2013). “The Efficacy of Support Vector Machines (SVM) in Robust Determination of Earthquake Early Warning Magnitudes in Central Japan”. *Journal of Earth System Science* 122.5, pp. 1423–1434. DOI: 10.1007/s12040-013-0346-3.

Rodríguez, Á. B., Balestriero, R., De Angelis, S., Benitez, M. C., Zuccarello, L., Baraniuk, R., Ibanez, J. M., and De Hoop, M. V. (2022). “Recurrent Scattering Network Detects Metastable Behavior in Polyphonic Seismo-Volcanic Signals for Volcano Eruption Forecasting”. *IEEE Transactions on Geoscience and Remote Sensing* 60, pp. 1–23. DOI: 10.1109/TGRS.2021.3134198.

Saccorotti, G., Piccinini, D., Cauchie, L., and Fiori, I. (Mar. 2011). “Seismic Noise by Wind Farms: A Case Study from the Virgo Gravitational Wave Observatory, Italy”. *Bulletin of the Seismological Society of America* 101.2, pp. 568–578. DOI: 10.1785/0120100203.

Seydoux, L., Balestriero, R., Poli, P., de Hoop, M., Campillo, M., and Baraniuk, R. (Aug. 2020). “Clustering earthquake signals and background noises in continuous seismic data with unsupervised deep learning”. *Nature Communications* 11.1. DOI: 10.1038/s41467-020-17841-x.

Seydoux, L. and Steinmann, R. (2023). *scatseisnet.* URL: https://scatseisnet.readthedocs.io/en/latest/index.html (visited on 10/24/2023).

Steinmann, R., Seydoux, L., Beaucé, É., and Campillo, M. (Jan. 2022a). “Hierarchical Exploration of Continuous Seismograms With Unsupervised Learning”. *Journal of Geophysical Research: Solid Earth* 127.1. DOI: 10.1029/2021jb022455.

Steinmann, R., Seydoux, L., and Campillo, M. (Aug. 2022b). “AI-Based Unmixing of Medium and Source Signatures From Seismograms: Ground Freezing Patterns”. *Geophysical Research Letters* 49.15. DOI: 10.1029/2022gl098854.

Steinmann, R., Seydoux, L., Journeau, C., Shapiro, N. M., and Campillo, M. (June 2023). “Machine learning analysis of seismograms reveals a continuous plumbing system evolution beneath the Klyuchevskoy volcano in Kamchatka, Russia”. DOI: 10.22541/essoar.168614505.54607219/v1.

Styles, P., Stimpson, I., Toon, S., England, R., and Wright, M. (2005). *Microseismic and infrasound monitoring of low frequency noise and vibrations from windfarms. Recommendations on the siting of windfarms in the vinicity of Eskdalemuir, Scotland.* Appliedand Environmental geophysics research group, School of Physical and Geographical science, University of Keele,

The ObsPy Development Team (May 2024). *obspy.core.trace.Trace.remove_response.* URL: https://docs.obspy.org/master/packages/autogen/obspy.core.trace.Trace.remove_response.html (visited on 05/07/2024).

Torrence, C. and Compo, G. P. (Jan. 1998). “A Practical Guide to Wavelet Analysis”. *Bulletin of the American Meteorological Society* 79.1, pp. 61–78. DOI: 10.1175/1520-0477(1998)079<0061:APGTWA>2.0.CO;2.

USGS National Earthquake Information Center, PDE (Apr. 2023). *M 7.8 - Pazarcik earthquake, Kahramanmaras earthquake sequence.* URL: https://earthquake.usgs.gov/earthquakes/eventpage/us6000jllz/origin/detail (visited on 04/25/2024).

VanderPlas, J. (Nov. 2016). *Python Data Science Handbook.* O’Reilly Media, Inc. ISBN: 978-1-4919-1205-8.

Williams, J. R. and Amaratunga, K. (1997). “A Discrete Wavelet Transform without Edge Effects Using Wavelet Extrapolation.” 3, pp. 435–449. ISSN: 069-5869.

Zieger, T. (2019). “Experimental quantification of seismic signalsinduced by wind turbines”. PhD thesis. Karlsruher Instituts für Technologie.

Zieger, T. and Ritter, J. R. R. (Sept. 2018). “Influence of wind turbines on seismic stations in the upper rhine graben, SW Germany”. *Journal of Seismology* 22.1, pp. 105–122. DOI: 10.1007/s10950-017-9694-9.