Deep learning as a tool for seismic data interpolation

Volodya Hlebnikov

A thesis submitted for the degree of Philosophiae Doctor (Ph.D.)

Department of Geosciences
Faculty of Mathematics and Natural Sciences
University of Oslo
Oslo, Norway
2022
Preface

This thesis is submitted in partial fulfillment of the requirements for the degree of Philosophiae Doctor (Ph.D.) at the University of Oslo. The thesis is a collection of three papers and additional unpublished work. The research presented here was part of an Industrial PhD-scheme between the University of Oslo and CGG Services Norway AS and was funded by the Norwegian Research Council (NRC). The granted Ph.D. project (297180) was supervised by Dr. Vetle Vinje, Senior Scientist at CGG Services Norway AS, and Prof. Valerie Maupin from the University of Oslo. From January 2019 – December 2021, the author Volodya Hlebnikov was employed by CGG Services Norway AS. All the project work was conducted in the offices of CGG and at the University of Oslo.
Acknowledgements

Similarly to a seismic wavelet, this Ph.D. adventure had its peaks and troughs. It would not have been possible to complete this challenging journey alone. To get through these periods of failures and achievements, you need people to support you. I had the pleasure to work with different supervisors during this three-year project. Some of you have decided to change positions or move on to other projects because of the dynamic environment in the seismic industry. However, I will never forget your invaluable efforts, help, and guidance, for which I will always be grateful.

First of all, I would like to thank Dr. Thomas Elboth, Dr. Vetle Vinje, Prof. Leiv J. Gelius, and Prof. Valerie Maupin.

**Thomas**, in some way, you have been the main driving force behind the initialization of this project. Without you, this work would have been impossible. Thank you for believing in me and allowing me to be part of the Marine research and development environment at CGG. Although we worked for a short period together and the project has changed significantly from what we discussed more than three years ago, I am very grateful for your support. Your contagious optimism has been a great source of inspiration, especially during the more difficult moments of the Ph.D.

**Vetle**, despite the not-so-easy task to take over and transfer my project to the Seismic Imaging group at CGG once the Marine group was no more, you made this process flawless. During the two-year collaboration we had, you have been the ideal supervisor. You allowed me to work independently on my own, yet you have always been there when I had doubts about my work, giving me valuable feedback. Despite being busy, you always had the time for me when I had questions. It is impressive how you could always find the link between different results within a discussion. Your knowledge and understanding of the greater context have always inspired me.

**Leiv**, I want to thank you for all the helpful discussions. I have learned so much from you in all those meetings we had discussed various topics of my work. I consider myself highly privileged to be your student; I benefited enormously from your excellence as a teacher. Thank you for being so transparent about my work and for all the crucial help and feedback you provided during my Ph.D. work. Your perspective often helped me put things into the
proper context, see the bigger picture, and understand where my mistakes were.

Valerie, thank you for being so supportive during the last year of my project. Your help during the drafting and the writing stage of this thesis was vital for me. Without your knowledge and understanding, the preparation of this thesis for such a short period would have been impossible.

Secondly, I would also like to express my gratitude to CGG Services Norway AS for giving me access to as much seismic data as I could ask for, a workplace, resources for conducting this research, and to all personnel helping me in the realization of this project – directly or indirectly. I particularly want to thank Gordon Poole and Dr. Peng Zhao from CGG for their constructive criticism during our discussions. In addition, I am very grateful to Thomas Greiner from Lundin Norway AS for his excellent collaboration and contribution and to all the co-authors contributing to the papers included in this thesis.

Further, I would like to thank the Norwegian Research Council for funding this work through an Industrial Ph.D.-grant number 297180.

Finally, I am thankful to my wife Sofia for her support, patience, and understanding.

Volodya Hlebnikov

January, 2022
List of Papers

**Paper I**

**Paper II**

**Paper III**
Related publications and conference proceedings

In addition to the research papers listed in “List of Papers”, four conference abstracts based on the Ph.D. work have been disseminated to the research community. They have not been included in the thesis because they describe subsets of the work and results presented in the research papers.


Other scientific contributions


# Contents

Preface .............................................................................................................................................. I

Acknowledgements ............................................................................................................................ II

List of Papers ..................................................................................................................................... IV

Related publications and conference proceedings ................................................................................ V

Other scientific contributions .............................................................................................................. VI

Contents ............................................................................................................................................. VII

1. Introduction ..................................................................................................................................... 1
   1.1 Outline and contributions of the thesis ...................................................................................... 1
   1.2 Motivation, background, and scope ............................................................................................ 3

2. Seismic Data .................................................................................................................................. 7
   2.1 Marine seismic data acquisition ................................................................................................. 7
   2.2 Noise in marine seismic data ..................................................................................................... 12
   2.3 Paper I – Noise types and their attenuation in towed marine seismic: A tutorial .................. 14
   2.4 Seismic data processing and imaging ....................................................................................... 34

3. Deep Learning ............................................................................................................................... 36
   3.1 Deep learning fundamentals ....................................................................................................... 36
      3.1.1 The forward algorithm ......................................................................................................... 37
      3.1.2 Nonlinear activation functions ............................................................................................ 39
      3.1.3 Training Deep Neural Networks ......................................................................................... 40
      3.1.4 The backpropagation algorithm .......................................................................................... 42
      3.1.5 The optimization algorithm ............................................................................................... 44
3.2 Convolutional Neural Network .............................................................. 46
  3.2.1 Convolution layer ........................................................................... 47
  3.2.2 Pooling layer .................................................................................. 50
  3.2.3 Residual layer ................................................................................ 51
  3.2.4 Creating a convolutional neural network architecture .................... 51
3.3 Deep learning in seismic ........................................................................ 53
  3.3.1 Common problems of deep learning in seismic ................................. 53
  3.3.2 Overview of deep learning applications in seisms ............................ 54

4.  2D interpolation of seismic data using Deep Neural Networks .......... 57
  4.1 Paper II – Cross-streamer wavefield reconstruction through wavelet domain learning ................................................................. 59
  4.2 Shot point interpolation using deep neural network ......................... 75
      4.2.1 Data conditioning for multi-feature input to the CNN ................. 77
      4.2.2 Customized Convolutional Neural Network ................................ 79
      4.2.3 Field data experiments ................................................................. 81
      4.2.4 Discussion and challenges ........................................................... 85

5.  3D interpolation and regularization of seismic data using Deep Neural Networks ...... 88
  5.1 Problem statement ............................................................................. 89
  5.2 Replication of an existing interpolation and regularization method using deep neural network ................................................................. 92
      5.2.1 Anti-Leakage Fourier Transform as labeled data ......................... 92
      5.2.2 Supervised learning-based interpolation and regularization .......... 94
      5.2.3 Customized Convolutional Neural Network ................................ 96
      5.2.4 Strategies of supervised learning-based offset class interpolation and regularization using ALFT as labeled data ................................. 97
      5.2.5 Data conditioning for multi-feature input to the CNN ................. 98
      5.2.6 Field data experiments using ALFT as labeled data .................. 98
      5.2.7 Discussion and challenges ........................................................... 102
  5.3 Paper III – De-migration-based supervised learning for interpolation and regularization of 3D offset classes ........................................... 107
6. Discussion and outlook ........................................................................................................... 135

6.1 Noise considerations ......................................................................................................... 136
6.2 Sparsely sampled seismic data ....................................................................................... 137
6.3 High-frequency loss vs network architecture ................................................................. 139
6.4 Deep neural network with multi-feature seismic input/output vs video prediction? ........................................................................................................................................................................... 140
6.5 Physics-based methods vs deep neural networks ............................................................ 142

References .................................................................................................................................. 144
1. Introduction

1.1 Outline and contributions of the thesis

The first part of this thesis describes noise in marine seismic data, while the last and main part is on data interpolation using Deep Neural Networks (DNN). This work demonstrates: (i) some of the actual noise problems of the seismic data together with the state-of-the-art noise reduction techniques, and (ii) seismic data resolution improvements through a DNN-based interpolation. The thesis contains an introductory chapter, followed by four main chapters addressing the investigated topics in this thesis: noise in marine seismic data and data interpolation using DNNs. The articles are embedded in the relevant chapters. Finally, the last chapter focuses on the discussion and concluding remarks.

The first chapter introduces the outline and the motivation, background, and scope behind this thesis.

The second chapter introduces marine seismic acquisition and seismic data processing and imaging. The chapter gives a brief description of the characteristics of seismic data and elaborates on some of its processing problems/issues. These problems include noise and sparsity, which are the two topics covered in this thesis. Paper I, which is embedded in Chapter 2, covers the broad topic of noise in marine towed seismic as a tutorial and serves as a preface for the remaining work.

The third chapter introduces machine learning and some of its main concepts. The chapter then elaborates on some of the main issues of DNN methods in seismic and gives a short review of DNN seismic applications for noise attenuation and interpolation. This chapter provides background information introducing the reader to the problems investigated in the following sections.

The fourth chapter describes DNN applications for interpolation purposes in 2D cross-sections of the seismic data. The chapter aims to find a suitable way of training a DNN model by using the data itself rather than simply mimicking an existing algorithm. It consists of two main sections, which investigate different data domains, where one can train a DNN in one domain and utilize the trained model in another. This approach is possible because these two data domains share the same data features. After a brief problem formulation, the
chapter presents Paper II that explores training a DNN model from the densely sampled dimension on the shot gathers to interpolate data in the usually less well-sampled orthogonal dimension, i.e., across the streamers. The second problem investigates a similar approach of training a DNN from the densely sampled shot gathers to interpolate common receiver gathers; thus, interpolate shot gathers.

The fifth chapter presents DNN applications for interpolation of 3D seismic data. The chapter briefly introduces the problem of sparsely and irregularly sampled offset class data. Then it shares key findings and recommendations related to the use of DNNs for solving such a 3D problem. Firstly, when using conventional methods for labeling the data, the DNN can hardly outperform these conventional methods. Secondly, training approaches that use different 2D data orientations (vertical and horizontal cross-sections) or 3D sub-volumes affect the results. Based on this analysis, the conclusion is that a better training approach, not dependent on a conventional method, is needed. In addition, it is an advantage if the training is in 3D to avoid imprints in the data. Also, increasing the input feature space of the DNN by using higher offset classes improves the quality significantly. This work contributes to the development of Paper III, which presents results from training a DNN for 3D offset class interpolation and regularization that does not try to mimic (directly) an existing conventional method. Instead, Paper III, utilizes a novel training approach based on de-migration that aims to improve the quality of the 3D offset classes compared to conventional methods.

Finally, chapter six provides a discussion and concluding remarks of the work in this thesis.

The main scientific contributions of the thesis can be summarized as follows:

(1) Paper I in Chapter 2 gives a detailed description of a vast number of marine noise types on towed-streamer arrays. It demonstrates some standard techniques for their attenuation and explains some novel processing tricks developed in industrial de-noising.

(2) Chapter 4, including Paper II, contributes to developing a methodology for DNN-based interpolation of regularly missing traces for the cases of cross-streamer interpolation and shot point interpolation.
(3) Chapter 5, including Paper III, provides a DNN-based workflows for interpolation and regularization of irregularly and sparsely sampled 3D offset class data. The main outcome of this research activity is a novel interpolation approach currently tested in CGG’s standard workflow. A patent application No. 17/483,197 entitled “Modeling-based Machine Learning for seismic processing” was submitted in September 2021.

The thesis explores DNN-based interpolation workflows compared with state-of-the-art processing algorithms. This comparison is not a trivial task and is only possible by accessing large amounts of seismic data and modern seismic data processing software, which requires a significant amount of computational resources. Having access to data and computational resources was possible because of the industrial nature of this Ph.D.

The thesis also analyzes various CNN network designs, trying to find setup/parameters that are favorable for seismic data processing. This work brings new insights into how existing DNN can be improved to address seismic data processing.

1.2 Motivation, background, and scope
Seismic reflection imaging is an essential tool for subsurface exploration. Continuous advancements in seismic data acquisition, seismic equipment, and seismic data processing techniques since their first introduction in the 1920s have resulted in high-resolution subsurface images. However, marine towed-streamer data are never ideal because of physical, economic, and environmental constraints. Among others, these data suffer from two main issues addressed in this thesis. The first one is that the acquired seismic data are never noise-free; various types of noise have a negative impact on our ability to use the data optimally. In general, any recorded energy, apart from the primary reflections, could be considered noise. In addition to the noise problem, the marine towed-streamer data suffer from irregular spatial sampling and sparse sampling along at least one of the five dimensions (shot-x,y, receiver x,y, and time) of seismic data. The reason for this is twofold. Firstly, sea currents prevent positioning the seismic sources and the receivers at their planned locations. Secondly, the chosen acquisition geometry has cost- and -efficiency constraints for not acquiring densely sampled data in all five dimensions.
As of today, the marine seismic industry consists of a few service companies having the capacity to both acquire and process seismic data and few service companies focusing primarily on the processing side. With the ongoing focus on energy transition, it is becoming more and more critical for the service companies to improve their efficiency, deliver high-end products and reduce costs. Such improvements are indeed required when considering carbon capture, utilization, and storage (CCUS) projects, where the economic margins are likely to be much tighter than in the oil and gas exploration business. These goals can be achieved either by innovative equipment and solutions at the acquisition side or by improving seismic data processing and imaging. In 2019 I was employed by CGG Norway AS in collaboration with the University of Oslo as an Industrial Ph.D. candidate, and I have been working along with this second alternative.

My main task has been to study and implement machine learning applications for seismic data interpolation. The overall objective was to develop DNN-based methods for seismic data interpolation and implement them directly in the processing workflow. To achieve the objective, the work presented in this thesis has demonstrated the feasibility of the DNN-based methods for cross-streamer interpolation, shot point interpolation, 3D offset class interpolation and regularization, and the advantage of the DNN-based methods in processing efficiency compared to the existing conventional processing methods. Having to work on an industrial Ph.D. program had its benefits. Ideas that arose from this work were immediately implemented in real data testing. This means that the quality of the investigated DNN-based methods was always benchmarked to the seismic industry standards. Therefore, some of the ideas presented in this thesis are proven concepts and are currently being investigated to be fully commercialized and used daily.

In recent years, we have witnessed an unprecedented increase in our ability to analyze large amounts of data. Moore’s law brought about this increased computational capacity (e.g., doubling of computation capacity every two years). A similar advance in algorithm efficiency led to many open-source libraries like Scikit-learn (Pedregosa et al., 2011), TensorFlow (Abadi et al., 2016), PyTorch (Paszke et al., 2017), and Keras (Chollet, 2018). The combination of increased computing power and efficient algorithms have enabled DNN to accelerate progress in several fields of science, i.e., computer vision (Diba et al., 2017; Voulodimos et al., 2018), biology (Angermueller et al., 2016), medical image analysis (Litjens et al., 2017) and others.
DNN can be loosely formulated as a method that attempts to mimic the human brain – albeit not matching its ability – allowing it to learn and, to some extent, generalize from large amounts of data; thus, make approximate predictions about data. Therefore, DNN methods are being actively researched within the geophysical community too. Some examples can be automated noise detection (Rentsch et al., 2014), fault detection (Araya-Polo et al., 2017; Xiong et al., 2018), velocity model building (Mosser et al., 2018; Richardson, 2018), migration artifacts attenuation (Klocshikhina et al., 2020), and bandwidth extension (Aharchaou and Baumstein, 2020).

Many attempts have been made to solve various seismic noise problems using DNN. Some of these examples include seismic interference (Sun et al., 2020; Xu et al., 2020), swell noise (Zhao et al., 2019; Zhang et al., 2020), surface-related multiple elimination (Siahkoohi et al., 2019a), blending noise (Richardson and Feller, 2019; Zu et al., 2020; Wang et al., 2021), and source bubble noise (de Jonge et al., 2021).

A significant challenge for DNN-based supervised methods in seismics is selecting data for the training. Labeled (also known as target or ground truth) data are hardly available, especially for noise attenuation or interpolation purposes. Because seismic data are never noise-free (Paper 1), obtaining the labeled (noise-free) data usually are met using conventional noise attenuation methods or recorded noise. We can then use DNN methods to (almost) mimic the conventional approach (Siahkoohi et al., 2019a; Peng et al., 2021). However, these methods can hardly outperform the conventional approach used in the first place. Other methods added random noise and proposed DNN-based techniques of attenuating it both in supervised (Jin et al., 2018; Si and Yuan, 2018) and unsupervised manner (Zhang et al., 2019; Song et al., 2020). Pure random noise attenuation is significantly easier than real seismic noise, which can often be very complex.

As for DNN-based interpolation, some methods require fully sampled data (Siahkoohi et al., 2018; Mandelli et al., 2019), which normally are unavailable. Similarly to the DNN-based noise attenuation methods, one can replicate the conventional interpolation method, which results in sub-optimal results. Methods proposed to exploit similarities between the acquisition domains for interpolation have also been proposed. For instance, the similarity between the common shot gathers, and the common receiver gathers (Siahkoohi et al., 2018; Park et al., 2019; Wang et al., 2019b). In this case, the DNN-based interpolation
method does not rely on conventional interpolation but learns patterns of the seismic wavefield from one domain and predicts in another. Alternatively, Paper II proposed training from the densely sampled shot gathers to interpolate data orthogonally, i.e., across the streamers, Chai et al. (2020) used a combination of training on both synthetic and real data examples. Wang et al. (2020) have shown a transfer learning approach using synthetic data to initialize real shot gatherings training. Recently, Qu et al. (2021) demonstrated the potential of a DNN model trained exclusively on synthetic data for shallow data reconstruction before de-multiples. However, it is difficult to create realistic synthetic data that allow a trained network to generalize and handle real seismic data in a good way. In many circumstances, representative training data are challenging to obtain. Paper III in this thesis proposes overcoming this challenge for the case of supervised 3D offset class interpolation by generating a tailored training dataset. In Paper III, we de-migrate a stacked pre-stack depth migration (PSDM) image into two configurations: sparsely and densely sampled for supervised training of convolutional encoder-decoder, increasing the input feature space using higher offset classes. Other studies have shown promising results applying unsupervised DNN to the interpolation problem in 2D (Hu et al., 2019; Shi et al., 2020), 3D (Kong et al., 2020), and 4D (Greiner et al., 2021). Kong et al. (2020) utilize the deep image prior (DIP) (Ulyanov et al., 2018) approach to interpolate missing data on 3D shot-gathers. However, the DIP approach suffers from high computational costs due to reparameterization for every new prediction. Greiner et al. (2021) proposed an unsupervised approach, training an overcomplete convolutional autoencoder including a combined first- and second-order total-variation regularization to predict missing traces in 4D.

It is widely demonstrated that trained DNN-based methods are time and cost-efficient (Chai et al., 2020; Sun et al., 2020), which is an advantage over the conventional processing methods. However, these methods will only rarely be implemented in production until they meet or ideally exceed the quality of the conventional methods. In this context, the work demonstrated in this thesis aims to describe and develop the DNN-based methods in seismics that can further both reduce the processing costs and improve the data quality.
2. Seismic Data

This chapter introduces seismic concepts, including data acquisition, noise in the seismic data, and seismic data processing, to provide the basic background information required to understand the topics covered in this thesis. It serves as a bridge between the papers because: (i) noise is encountered at the acquisition phase; thus, one needs to know what an acquisition is, and (ii) it also highlights that interpolation is required because the acquisition geometries have limitations in terms of spatial sampling.

2.1 Marine seismic data acquisition

In marine seismic exploration, acoustic sources, generally in the form of air gun arrays, are used to produce a signal that penetrates the subsurface, see Figure 2-1. Part of that signal is reflected to the surface at various geological interfaces. That reflected signal is eventually picked up by receivers towed in the water column. A seismic trace in a receiver (or channel) is the summed response of a group of one or more individual hydrophones. This signal contains information about the reflection's arrival time, amplitudes, and characteristics. The received data are subsequently processed and finally used to produce a 3D image of the subsurface. This method is used to estimate the elastic properties of the subsurface, which then can help geoscientists pinpoint where to drill for oil and gas deposits or, in the future, identify and monitor subsurface CO2 storage sites and perhaps seabed minerals. However, estimating specific physical properties of the subsurface (e.g., acoustic impedance or density) is a long and expensive process, and its main steps are schematically summarized in Figure 2-2. The sketch shows that the process consists of four main steps: survey design, acquisition, processing and imaging, and interpretation. The survey design aims to determine the suitable survey parameters for the geological settings, usually using some modeling methods. Conceptually, this

![Figure 2-1. 2D cross section of the marine seismic method. The seismic vessel tows a streamer (receivers) that records the partially reflected energy emitted by the acoustic (sound) source. Two distinctive events are highlighted on the seismogram, the direct wave (red) and the water bottom reflection (yellow).]
step evaluates the trade-off between expected data quality and efficiency. Figure 2-3 shows some of the survey parameters for a conventional dual-source acquisition. The efficiency, i.e., the time needed to acquire the whole prospect area, is dependent on these parameters, which also will affect the data quality. The bin size $\Delta x$ and $\Delta y$ for example will affect the spatial resolution of the data. If the chosen bin size is too large, this will result in low resolution and reduce the data quality. Multi-component measurements allow for improved spatial sampling after tailored processing (Robertsson et al., 2008; Özbek et al., 2010; Vassallo et al., 2010). However, this thesis uses conventional hydrophone-only data.

Figure 2-2. Schematic illustrating the complex process of obtaining subsurface information. The main steps include survey design or modelling of the survey parameters, acquisition, processing and imaging, and finally interpretation that extracts qualitative and quantitative information of the subsurface.

To measure the spatial sampling, we use the term bin, which subdivides the acquisition area by a grid of lines. In practice, the bin size should be small to ensure high lateral resolution. It is directly linked to the survey layout parameters such that $\Delta y = DC/2NS$, and $\Delta x = RI/2$, where $RI$ denotes the receiver separation within a streamer. $RI$ is usually 12.5 m, for modern streamers, but different configurations also exist (Martin et al., 2000; Pan and Moldoveanu, 2001; Goujon et al., 2019). Among other parameters that depend on the acquisition layout and, more specifically, the shot point interval, we have the offset class size $Doff =$
and the fold $f = L/2DSx$. The smaller the shot interval, the higher the fold, which improves signal to noise ratio (S/N). In practice, this leads to blended acquisition (Barbier, 1983; Vaage, 2005), where consecutive shot points overlap. Although the blended acquisition is beneficial in cost-efficiency, a significant amount of processing is required to separate the blended energy.

The presented acquisition layout in Figure 2-3 corresponds to a conventional narrow azimuth (NAZ) survey, where azimuth represents the angle between the shot/receiver direction and the shooting direction. The nearest 3D offset classes can be problematic for conventional NAZ surveys. An offset class is defined as a 3D sub-volume of the recorded seismic data within a specific offset range. This range, as defined early, is determined by twice the distance between the shots along the line. As an example of an offset class volume formation, consider Figure 2-4, a NAZ acquisition, where the red asterisk indicates the active source with streamers (in orange) separated by 100 m and receivers (green dots) separated by 12.5 m along each streamer. The traces from the receivers within the 150 – 225 m highlighted red half-circle from the active source will fall into the first offset class. These traces are mapped into the bin cells associated with the source-receiver CMP, given as red

---

Figure 2-4. An illustration of NAZ acquisition overlaid with the acquisition bin grid illustrating the sparseness of the near offsets. Red dots indicate the CMP position formed by source-receiver combination, red asterisk, and green dots, respectively. Offset class definition indicated by the red half-circle with center $S_1$ for a single sail line in a), and for two adjacent sail lines in b).
dots in Figure 2-4. As we can see from this example, the first offset class is sparse, and only the central streamers contribute to its formation. Consequently, holes appear in the coverage. When we plot that offset class for two adjacent sail lines (Figure 2-4b), we notice that the gaps between sail lines for that offset class become even larger. The second offset class will also be sparse, and for the larger offset classes, the coverage will improve until eventually being nearly complete. The problem of the near offset exists for other acquisition layouts, and it is the main topic of Chapter 5, where it is explained in detail.

Figure 2-5 shows the azimuth-offset distribution for a series of acquisition layouts. The distance from the center of the circles is the offset, the angle represents the azimuth, while the color palette represents the fold. Within the figure, we have layout examples of NAZ, multi azimuth (MAZ) (Gaus and Hegna, 2003), wide azimuth (WAZ) (Sukup, 2002; Michell et al., 2006), and the combination of MAZ and WAZ leading to rich azimuth (RAZ) (Howard and Moldoveanu, 2006; Howard et al., 2007). Other acquisition layouts also exist, such as circular geometry (Moldoveanu et al., 2008) and full azimuth (FAZ) (Long et al., 2014). As shown in Figure 2-5, not only the diversity of azimuths, which provides complimentary target illumination, is directly impacted by the chosen acquisition layout but also the spatial sampling of the recorded seismic data. The source-over-streamer (Vinje et al., 2017) layout provides an increased near offset coverage with positive and negative offsets from the split-spread layout, which have the potential of improving both multiple modeling and imaging (Dhelie et al., 2018c). However, once these data are sorted to 3D offset classes, the near offsets are still not fully populated. Thus, these near offset classes will suffer from spatial aliasing at high-temporal frequencies from the coarse sampling across the inline and the crossline dimensions. Various interpolation methods exist to improve the data consistency along with the data five dimensions. Nevertheless, in the case of aliased data, the interpolation problem is ill-posed, and many possible solutions may explain the observed aliased data.

The seismic equipment's tow-depths, i.e., source and streamer, and the source volume, are other vital acquisition parameters that directly impact the data quality. Usually, sources with different volumes are arranged in arrays to create a source signature that is short in time, spiky, and has a large primary-to-bubble ratio (Ziolkowski et al., 1982).

The tow-depths define the source and receiver side ghost effect. Ghosts result from a reflection of the up-going wavefield at the sea surface, which interfere constructively and
destructively with the primary reflections. As a result, notches in the frequency spectrum appear, reducing the temporal resolution (Carlson et al., 2007). The notch frequency is defined as $f_n = \frac{v}{2\Delta z\cos \theta}$, where $v$ is the water velocity, $\Delta z$ is the source or receiver tow-depth, and $\cos \theta$ is the incidence angle. The classical solution to the notch phenomenon was to tow the source and streamer relatively shallow and use the data below the first notch. Other solutions were proposed, such as deconvolution (Jovanovich et al., 1983), and over-under-streamer acquisition (Hill et al., 2006), until Carlson et al. (2007) introduced the multi-component streamer, allowing streamers to be towed deeper, enhancing the low frequencies. Soubaras and Dowle (2010) also raised processing-based solutions using variable-depth streamers for conventional hydrophone-only streamers.

![Figure 2-5. Azimuth-offset distribution for conventional narrow azimuth survey (NAZ) in a), multi-azimuth survey (MAZ) in b), wide azimuth (WAZ) in c), and rich azimuth (RAZ) in d) (image courtesy of CGG).](image)

On the other hand, the temporal sampling rate at which the data are digitized governs the maximum frequency value that can be recorded without aliasing, named the Nyquist frequency, $f_N$, which is $f_N = \frac{1}{2\Delta t}$, where $\Delta t$ is the sampling rate.

Within the scope of this thesis, we deal with conventional marine NAZ data and source-over-streamer data using variable-depth hydrophone-only (single component) streamers with a receiver separation of 12.5 m.
2.2 Noise in marine seismic data

The recorded seismic data consists of a summation of reflected energy and various types of noise that negatively affect our ability to use these acquired data optimally. The nature of the seismic data may vary depending on many factors. These factors include geology, the size of the seismic source, the depth of the source and the streamer, the sensitivity of the equipment, and the amount of noise. The problem of noise in marine seismic can either add additional cost at the acquisition stage, often related to weather or operational stand-by (Smith, 1999; Elboth and Haouam, 2015), or can affect subsequent processing steps like ghost attenuation, multiple attenuation, and velocity analysis. Noise can also be one of the main factors that obscure deep targets and reduce the subsurface resolution. The origin of the recorded noise can either be related to the seismic instruments (e.g., bird or cross-feed noise), background (e.g., swell or seismic interference noise, engine noise or anthropogenic noise), or source generated (e.g., blending or air gun bubble noise). The noise can either be coherent or noncoherent from trace to trace in the observation data domain. Noise can also appear broadband or within a narrow range of the frequency spectrum. Figure 2-6 depicts different types of noise, i.e., seismic interference, swell, tugging, and streamer hits.

Figure 2-6. Various types of noise affecting the recorded data and obscuring the reflections of interest (image courtesy of CGG).

For example, the shot record in the top left corner in Figure 2-6 can be described as a background, coherent and broadband noise affecting the streamer from behind termed seismic interference caused by nearby seismic vessel activity. These examples show that noise
amplitude may vary significantly and sometimes obscure the seismic signal. Whenever that happens, the noise becomes problematic and specific techniques for noise attenuation are sought to improve the S/N ratio. Reducing the unwanted noise levels is achieved by two main approaches. The first approach implies that the recorded noise at the acquisition phase is as tiny as possible. This approach requires that the state of the seismic equipment be optimal, designed, and deployed so that it does not amplify specific unwanted energy. The second approach implies seismic data processing techniques to improve the S/N ratio further. The seismic industry combines these two approaches, where the latter consists of data transformation to a domain in which the noise and the desired signal separate. In such a way, the noise component is attenuated, and the signal is transformed back to the original data domain. Alternatively, the noise is also transformed back and adaptively subtracted from the data. Understanding the origin of noise plays a crucial role in selecting the proper technique for noise suppression. In addition, suppressing different noise types requires knowledge of the characteristics of the noise that can separate it from the signal. The following subchapter, i.e., Paper I, examines a wide range of real data examples contaminated by different types of noise. The aim of this tutorial paper is that, based on the provided insights, the geophysical community will be able to appreciate some of the most common types of noise encountered in marine towed seismics and understand some of the methods to deal with them.
2.3 Paper I – Noise types and their attenuation in
towed marine seismic: A tutorial

Hlebnikov, V., T. Elboth, V. Vinje, and L.-J. Gelius

Noise types and their attenuation in towed marine seismic: A tutorial

Volodya Hlebnikov¹, Thomas Elboth², Vetle Vinje², and Leiv-J. Gelius³

ABSTRACT

The presence of noise in towed marine seismic data is a long-standing problem. The various types of noise present in marine seismic records are never truly random. Instead, seismic noise is more complex and often challenging to attenuate in seismic data processing. Therefore, we have examined a wide range of real data examples contaminated by different types of noise including swell noise, seismic interference noise, strumming noise, passing vessel noise, vertical particle velocity noise, streamer hit and fishing gear noise, snapping shrimp noise, spike-like noise, cross-feed noise, and streamer-mounted device noise. The noise examples investigated focus only on data acquired with analog group forming. Each noise type is classified based on its origin, coherency, and frequency content. We then determine how the noise component can be effectively attenuated through industry-standard seismic processing techniques. In this tutorial, we avoid presenting the finest details of either the physics of the different types of noise themselves or the noise attenuation algorithms applied. Rather, we focus on presenting the noise problems themselves and find out how well the community is able to address such noise. Our aim is that, based on the provided insights, the geophysical community will be able to gain an appreciation of some of the most common types of noise encountered in marine towed seismic, in the hope to inspire more researchers to focus their attention on noise problems with greater potential industry impact.

INTRODUCTION

Marine seismic acquisition is a commonly used method for collecting information about the subsurface. However, various types of noise have a negative impact on our ability to use these data optimally. In general, any energy recorded, apart from the primary reflections, could be considered as noise. Nevertheless, several types of waves are considered as noise, whereas they simultaneously carry useful information about the subsurface. Multiples, for instance, could be considered as usable signals, although multiple attenuation is a major part of the standard seismic data processing sequence. Multiples carry information useful for the estimation of missing near offsets (van Groenestijn and Verschuur, 2006) and for improving image resolution (Lu et al., 2013; Ordoñez et al., 2016). Therefore, care should be taken when defining what is considered noise.

The problem of noise in marine seismic is twofold. First, the problem of noise adds extra costs at the acquisition stage, often related to either weather-associated delays (Smith, 1999) or operational time-sharing delays in the case of two or more vessels operating in close proximity (Elboth and Haouam, 2015), although both of these have become less common in recent years. Second, the problem of noise affects subsequent processing steps such as (sea surface) ghost attenuation, multiple attenuation, and velocity analysis. If the recorded noise is not properly attenuated, not only will the quality of the seismic image be degraded but also our ability to interpret and analyze the data. Consequently, one of the main (and early) tasks in seismic signal processing is noise attenuation. The various types of noise present in marine seismic records are never truly random. Instead, each type of noise normally has distinct characteristics. By identifying the origin of the noise and classifying it correctly, the optimal attenuation results can normally be achieved.

In this tutorial, we present, characterize, and classify real data examples of different types of marine seismic noise. Then, we demonstrate how seismic noise attenuation processing can improve the survey efficiency, by reducing the need for data reacquisition, and improve the overall image quality. Appendices A–H provide further...
details on the noise attenuation signal processing sequences used for each of the noise examples presented.

Each of the individual noise examples presented here will be classified according to:

1) The origin of the noise — background (e.g., swell), source generated (e.g., blending noise), or instrument noise (e.g., crossfeed), after Elboth et al. (2010).
2) The presence of any of these types of noise can be either coherent or noncoherent from trace to trace in the observation data domain.
3) The final feature is the characteristic frequency range of the noise, if any.

The motivation behind our work is the lack of recent literature combining and presenting the various types of noise expected during the acquisition of state-of-the-art towed marine seismic. Notable exceptions are the work presented by Olhovich (1964), Fulton (1985, 1993), and Dondurur (2018), in which various noise examples are analyzed.

How to enhance the signal-to-noise ratio (S/N) of seismic data has been an ongoing study for many years. Reducing the amount of unwanted noise is achieved by two approaches. The first approach is ensuring that the amount of recorded noise is as small as possible at the acquisition phase. This implies that the state of the seismic equipment should be optimal and operated correctly. Regular cleaning (to avoid barnacle growth) and proper maintenance of equipment mounts, which are known to induce vibration noise, are essential. Second, high noise levels can be reduced with various seismic signal processing methods. The industry normally uses a combination of algorithms, in which each of them targets a specific type of noise. The general concept for most of the existing noise attenuation methods consists of data transformation to a domain in which the noise and the desired signal more easily separate. Once the noise has been attenuated, the remaining signal is transformed back. Alternatively, the separated noise is transformed back, and it is (adaptively) subtracted from the original data. Stacking (Mayne, 1962) is also a classic way of improving the S/N. Noise attenuation is usually applied as part of the initial poststack processing of the sail line. At this stage, we try to attenuate the bulk of dominating noise to avoid contaminating the later ghost attenuation, multiple attenuation, and velocity analysis processing steps. As a further refinement, we may also apply additional noise attenuation techniques to attenuate residual noise. Further noise attenuation processing might also be considered at the offset processing stage, prior to stacking, or even poststack and even after migration.

DATA EXAMPLES

The data examples have been collected throughout years of experience in the seismic industry onshore in processing centers and offshore on various seismic survey vessels. All examples are acquired by solid-filled streamers with a group spacing of 12.5 m, which means that every trace response represents the average signal of all of the sensors within the 12.5 m group. Averaging the signal within each group reduces the noise level (Martin et al., 2000) because the combined response acts as a spatial filter and it also removes information about the nature of the noise as recorded by each individual sensor. We present noise examples with two main origins — background and instrument. Source-generated noise such as blending noise and multiples is not addressed in this work. The interested reader is referred to Barberi (1983) and Vaage (2005) as the earliest known references for blending noise and, for some of the widely used approaches addressing surface-related multiples, to Verschuur et al. (1992), Berkhour and Verschuur (1997), and Verschuur and Berkhour (1997). Furthermore, we will not address the general ambient background noise recorded by the streamers. Wenz (1962) characterizes such noise, and the so-called Wenz diagram has been regularly updated in the subsequent years. According to Ross (2005), the low-frequency ambient noise due to shipping increases by one-half dB per year. The amplitude of this ambient background noise is in most cases low compared with the seismic reflection data that we record. A notable exception is at the lowest end of the exploration seismic spectra (below approximately 3–6 Hz), where the output of the seismic sources is limited.

Background types of noise

External noise is always present in seismic data. This type of noise has been an ongoing subject of study throughout the years. Schoenberger and Mifsud (1974), Fulton (1985), and Dondurur (2018) identify several different sources of noise. The most common types of background noise are swell noise and seismic interference (SI), which are easily recognizable. However, biological noise such as from marine biota can be more challenging to identify. Fulton (1993) presents examples of noise from shark bites, dolphins, and whales in seismic data.

Swell noise

Swell (rough sea) noise is one of the most common types of noise present in towed marine seismic. In our data example acquired by a solid-filled streamer with 12.5 m group spacing, this noise appears as vertical stripes, as shown in Figure 1a (data are courtesy of Frogro). Swell noise, if any.

1) The origin of the noise — background (e.g., swell), source generated (e.g., blending noise), or instrument noise (e.g., crossfeed), after Elboth et al. (2010).
2) The presence of any of these types of noise can be either coherent or noncoherent from trace to trace in the observation data domain.
3) The final feature is the characteristic frequency range of the noise, if any.
versal streamer vibrations propagating along the streamer generate more swell noise when the weather conditions deteriorate. This propagation is characterized by a very low apparent velocity of approximately 30–120 m/s (Teigen et al., 2012). However, during group forming (as per in our swell noise example), the relatively slow vibrational noise is highly aliased. Brink and Spackman (2004) show an example of this aliasing effect by analyzing data acquired with a solid-filled streamer with group spacing of 12.5 m and a solid-filled streamer with single hydrophone at an interval of 1.5 m during rough seas. In the same work, it is demonstrated that the swell noise more strongly affects the older fluid-filled streamers, although the higher number of hydrophones is used in each group for the older streamer generation.

Historically, seismic acquisition has been suspended when the swell noise level exceeded a predetermined level. Because of the increased computer capacity and available algorithms to tackle the problem of swell noise, modern seismic vessels are now more commonly limited by the risk of damaging their in-sea equipment in periods of bad weather, and not by swell noise on the recorded data. Nevertheless, seismic data processing still has to deal with these significant noise levels. The effect of the swell noise on the water column reduces with depth, which depends on the wavelength of the ocean swells (Trujillo and Thurman, 2005). Therefore, towing the equipment deeper reduces the impact of the noise. On the other hand, this deeper tow narrows the usable bandwidth of the data because the ghost notch is dependent on the depth of the streamer. The deeper the streamer is, the narrower the useful bandwidth is. Several approaches overcoming this bandwidth limitation are available. An early attempt is based on the use of dual streamers towed at two different depths (Hill et al., 2006). More recently, the variable-depth streamer (Soubaras and Dowle, 2010) and the dual-sensor streamer with pressure and velocity sensors (Carlson et al., 2007; Tenghamn et al., 2007) have been introduced. In addition, pure processing solutions based on hydrophone data only (Poole et al., 2018a) have also been developed.

A wide range of noise attenuation tools is available to address swell noise in the marine seismic environment for conventional group-formed data. Most of these methods are based on prediction error filtering (Canales, 1984) in the $f$-$t$ domain. Despite the effective attenuation of unwanted noise, this type of filtering does not preserve signal amplitudes. Several improvements have since been published; see, for example, Gülünay (1986), Soubaras (1995), Schonewille et al. (2008), Elboth et al. (2010), Bekara and van der Baan (2010), and recently Chen and Sacchi (2017). Attenuation methods in another group are either based on rank-reduction methods (Chen and Sacchi, 2015), dictionary learning (Vaezi and Kazemi, 2016), or deep convolutional neural networks (Zhao et al., 2019). Ozek (2000) introduces the linearly constrained adaptive noise attenuation method used on point receiver recorded data (Martin et al., 2000; Pan and Moldoveanu, 2001). A typical modern noise attenuation processing sequence, depending on the acquisition system, would combine some of these methods to better address the noise.

Figure 1a shows a shot gather contaminated with swell noise, i.e., a high-amplitude burst of energy. Although this high-amplitude noise affects several neighboring traces, we classify the swell noise as noncoherent from trace to trace for our group-formed data exami...
ple. This classification is because the apparent low coherency between the individually affected groups of traces is subject to change for every burst of swell noise and thus makes it challenging to fulfill a coherency criterion, such as moveout. Not attenuating such kind of noise would affect subsequent processing steps and deteriorate the final image. Figure 1b and 1c shows the result of swell noise attenuation and the difference plot, respectively. For further details about the noise attenuation technique used, refer to Appendix A. The corresponding frequency and f-k spectra are shown in Figure 1d–1g, in which we can observe that the frequency span of the noise is in the range of 2–15 Hz.

SI noise

SI is a type of background noise that occurs when two or more seismic vessels operate in the same general area. In most cases, SI noise consists of acoustic energy that propagates in the water column and it can often be visible for sources up to approximately 100 km away. Depending on the position of the source generating such noise, the SI appears differently in the shot domain, as illustrated in Figure 2.

SI noise can be classified to fall within three main categories:

1) Head-to-tail (H/T), in which the position of the noise source is ahead of the streamers, and the SI appears as “stripes” with a positive dip.
2) Tail-to-head (T/H), in which the position of the noise source is located behind the streamers, and the SI appears again as stripes, but with a negative dip.
3) Broadside (B/S), in which the position of the noise source is located beside the streamers. Depending on the distance to the noise source, the SI appears either with certain moveout when the source is relatively close or as nearly horizontal stripes when the source is relatively far.

Difficulties arise when the amplitudes and moveout (dip and/or curvature) of the SI noise exceed a certain threshold. Elboth and Haouam (2015) claim that, historically, it was not uncommon for vessels working in the North Sea to spend up to 30% of their available time on stand-by due to time sharing, which is operationally expensive and becomes challenging in areas where more than two vessels operate at the same time. However, in recent years, the industry has developed several methods for minimizing time sharing. Dhelie et al. (2013), Elboth and Haouam (2015), and Laurain et al. (2016) demonstrate that the costly time sharing due to SI noise could be reduced by proper control of the SI noise moveout and its arrival time on the seismic records. Most of the available algorithms for SI attenuation depend on randomizing the noise by sorting to other domains followed by prediction filtering (Gülünaçay, 2008; Elboth et al., 2010; Zhang and Wang, 2015). However, in the case of B/S SI being continuous (repetitive) in time and, therefore, coherent from shot to shot, the attenuation of such noise remains a challenge. According to Elboth and Haouam (2015), two steps are necessary to ensure that interference between vessels operating in the same area will not significantly affect the data quality. First, proper planning and intervessel coordination must be in place to ensure that the SI noise does not arrive at the same time on consecutive shots. Second, a suitable noise attenuation algorithm must be chosen at the seismic processing stage. Recent work by Elboth et al. (2017) and Shen et al. (2019) demonstrates that the use of a “line-mixing” approach potentially eliminates the need for time sharing and potential reacquisition due to any sort of SI. The line-mixing approach consists of borrowing shots from a neighboring sail line (source-cable pair) and mixing in these shots to break up any SI that is shot-to-shot coherent for the line in question. These borrowed shots are removed at the end of the SI attenuation process to respect the original sail line (source-cable pair).

When pressure and full particle velocity vector data are available, Vassallo et al. (2012) claim that the SI noise could be isolated, and thus attenuated. As an alternative approach to the conventional methods, Sun et al. (2020) demonstrate promising results of SI noise attenuation using deep convolutional neural networks.

Figure 3a shows interference from two different sources. The first one, labeled (1) in Figure 3a, is high-amplitude broadband interference from another vessel. This SI noise consists of a train of events of approximately 1.5 s, which is noncoherent from shot to shot. The noise labeled (2) in Figure 3a is observed as almost-horizontal stripes, and the vessel’s crew reported it to be produced by nearby rig activity. Although not generated by a seismic vessel, we can still classify such noise as SI. This rig noise is broadband, and its moveout overlaps with one of the reflections at the near offsets. Moreover, the noise train is continuous and affects the complete record length, which makes it shot-to-shot coherent, and thus challenging to attenuate. Figure 3b and 3c shows the results obtained after SI noise attenuation and difference plot, respectively. For further details about the noise attenuation technique used, refer to Appendix B. The corresponding frequency and f-k spectra are shown in Figure 3d–3g, in which we can clearly see that the noise removed has a broadband characteristic.
Strumming or tugging noise

Strumming or tugging noise is produced by longitudinal vibrations of a streamer (Teigen et al., 2012), which are caused by sudden variations in the streamer tension. This sudden change in the tension is normally a result of rapid movements of the seismic vessel due to the sea state conditions. This type of noise is often more visible on the outer streamers of the spread (Elboth et al., 2010). The reason is that these streamers are also connected to paravanes, which are floating devices that help maintain the desired separation between the streamers. These paravanes create additional tugging. Longitudinal waves typically propagate along the cable with velocities between approximately 1.5 and 3.0 km/s. The reason for having such a wide range of velocities for this noise is a result of the design of a modern solid streamer. If we look at a cross section of such a streamer, we have the outer skin made of polyurethane with an approximately longitudinal wave velocity of 1.5 km/s, and a polymer body with a similar wave velocity as the skin follows the outer skin. On the other hand, the strength members made of Kevlar are associated with a velocity of approximately 2.0 km/s. Finally, the highly conductive material (copper wires) is characterized with velocities of approximately 3.0 km/s or even higher. This background type of noise normally propagates along the first few hundred meters of the streamer, and it is coherent within the shot gather. In addition, the tail buoy is pulling the streamer from the back and similar noise is generated, but this time having opposite dip and propagation from the tail of the streamer. The example shown in Figure 4a has strumming noise from the T/H (the right side of the shot gather) and from the H/T (the left side) of the streamer. In this particular case, the amplitudes of the strumming noise from H/T are higher than usual. The noise also affects the whole length of the streamer as opposed to the first few hundred meters. This noise was most likely caused by debris caught on the separation ropes between the streamers, generating extra tension and vibrations. Figure 5a–5f shows a family of frequency panels computed from the shot gather contaminated with strumming noise, where we can observe that most of the noise is concentrated within the frequency range of 1–8 Hz. Several different methods may be used to obtain a target-oriented noise attenuation (e.g., f-k filtering or tau-p filtering). However, the application of these common methods used for coherent events resulted in signal loss due to the dip of the strumming noise partially overlapping with that of water bottom reflection (the results are not shown here). Because the noise was an issue at the acquisition phase, the onboard processing team used a model-based subtraction technique. The result of the complete noise attenuation sequence, including the model-based subtraction of the strumming noise, is shown in Figure 4b. As seen from the difference plot in Figure 4c, no damage to the signal is observed.

Passing vessel noise

Commercial ships can generate underwater noise that is sometimes recorded by the seismic receivers. The dominant noise source is usually propeller cavitation with frequencies below 200 Hz (Hildebrand, 2009). Depending on the propeller design (such as the number of blades) and the operating conditions (such as the vessel’s load and speed), the propeller noise may appear either as harmonic, broadband, or narrowband (Gorji et al., 2018).

Figure 3. The SI noise as seen on the shot gather with group spacing of 12.5 m. (a) Before SI noise attenuation; (b) after SI noise attenuation; (c) the difference plot (with the SI noise removed); (d) frequency spectra (normalized to the largest peak): red, blue, and green — before, after, and the difference (with the SI noise removed), respectively; and (e–g) f-k plots — before, after, and the difference (with the SI noise removed), respectively. Spectra are calculated for the entire record length.
Figure 6 depicts a triplet of shot gathers contaminated by noise generated by a nearby passing vessel. In the three noise-contaminated shot gathers (Figure 6a), the noise is highlighted with the black arrows. These shots are not consecutive, and, as seen in the example, the noise is coherent from trace to trace and has an appearance similar to SI noise. The ship noise is initially H/T, indicating that the passing ship is ahead of the recording streamers — the first shot gather (from left to right); as the passing ship moves alongside the streamers, the noise becomes B/S — second shot gather; and once the ship reaches the tail of the streamers, the noise becomes T/H (the third shot gather). Because the characteristics of the ship noise are similar to those of SI noise, a noise attenuation flow similar to the one used for the SI example was applied; refer to Appendix B. However, in that particular case, the line-mixing approach was not used. The resultant shot gathers after noise attenuation are shown in Figure 6b. Finally, the noise removed is shown in Figure 6c. The frequency spectra of one of the shot gathers from Figure 6 are displayed in Figure 7, in which the data before noise attenuation are in the red and the data after noise attenuation are in the blue. Note that, in our particular example, the ship noise is band limited and mostly falls between 70 and 80 Hz.

Vertical particle velocity-sensor noise

Conventional streamer systems are equipped with pressure sensors measuring hydrostatic pressure changes. Such pressure sensors record upward- and downward-traveling waves, the latter contributions caused by reflections at the air-water interface. These downward-traveling waves contain multiples and the sea surface reflected ghost waves. Because these ghost waves introduce notches in the frequency spectra, the recorded seismic signals for conventional marine towed streamers are degraded. The combination of different sensors allows
the joint measurement of pressure wavefields using hydrophones and vertical particle velocity fields using motion sensors (Tenghamn et al., 2007). The dual-sensor approach may lead to significant improvements within ghost attenuation. Robertsson et al. (2008) discuss the use of a multicomponent towed streamer that measures the pressure wavefield and the full particle velocity fields (in the 3D sense). The use of multicomponent measurements led to further improvements not only in ghost wave attenuation and interpolation (Robertsson et al., 2008; Özdemir et al., 2010; Bunting et al., 2013; Poole and Cooper, 2018b) but also in imaging (Orji et al., 2010; Ordoñez et al., 2016). However, motion sensors have their own noise issues. This noise is the result of longitudinal, transversal, or torsional streamer vibrations (Özdemir et al., 2012). Figure 8a depicts noise recorded along the vertical component of the particle velocity. Most of this noise observed in this vertical-component example is due to transversal streamer vibrations. Unlike hydrophones, which are made insensitive to transversal acceleration by mounting them in pairs (Dowle, 2006), a motion sensor cannot (by definition) be made insensitive to such movement (Teigen et al., 2012). Carlson et al. (2007) and Caprioli et al. (2012) demonstrate that, for dual-sensor streamers, the S/N is improved if the vertical component of the particle velocity data is not used for the low-frequency range below approximately 15–16 Hz. For the frequency range in which the noise recorded by the vertical motion sensor is excessive, the data may be predicted by the pressure sensor data, using the low-frequency conditioning (Carlson et al., 2007). The noise affecting motion sensors typically has much higher amplitudes compared with the useful seismic signal (Day et al., 2013). Because the noise propagates at very low velocities (Mellier et al., 2014), it is highly aliased for the 12.5 m conventional sensor group spacing. Recent work indicates that a maximum sensor spacing below 1 m is required if we want to capture the noise without aliasing (see Goujon et al., 2019).

An example of vertical component particle velocity data, with a 12.5 m sensor group spacing, contaminated by noise is shown in Figure 8a (the raw shot gather). We can see that the data are mostly dominated by noise with hardly any visible seismic reflections. One approach to address such noise is to filter out the low-frequency part of the data. In this example, the cut-off frequency was set to the conservative value of 20 Hz and the filtered result is shown in Figure 8b. The noise level is reduced significantly, and seismic reflections are now visible. However, further seismic processing is required to attenuate the remnant noise bursts present in the data. This remnant noise is typically generated by transversal vibrations/rattling caused by the streamer-mounted equipment. Some of the algorithms used to address that remnant noise energy are based on extensions of the random noise attenuation in the f-x domain tailored for multimeasurement data (Naghizadeh and Sacchi, 2012; Kamil et al., 2015) or introducing domain-based filters exploiting the theoretical signal cone limits (Rentsch et al., 2014). It is also possible to use joint noise attenuation methods using the hydrophone data to guide the attenuation of the velocity sensor data (Carlson et al., 2007; Peng and Huang, 2014; Sanchis and Elboth, 2014). Figure 8c and 8d shows the noise-attenuated vertical motion sensor data and the difference data (noise attenuated), respectively. For further details about the noise attenuation techniques used, refer to Appendix D. Alternatively, refer to Özdemir et al. (2012) for noise attenuation methods for multicomponent measure-

Figure 6. Example of ship noise as seen on a shot gather with group spacing of 12.5 m. (a) First triplet of shot gathers — before noise removal, (b) second triplet of shot gathers — after noise removal, and (c) last triplet of shot gathers — the difference plot (with the noise removed).

Figure 7. Frequency spectra calculated for the entire record length. Before (red) and after (blue) ship noise removal.
ment using point receiver data. The noise present on the vertical velocity sensor in our data example (Figure 8) mostly affects the low-frequency spectra up to approximately 20 Hz. However, weaker noise is still present at frequencies above 20 Hz. Because the noise in the observation domain has an appearance similar to swell noise (with sensor group forming), we classify the noise as noncoherent.

Streamer hit and fishing gear noise (strumming noise)

Another example of noise frequently observed in areas with high fishing activity is that due to fishing gear or debris in the water. It is common for a towed marine streamer to collide with various types of debris. In most cases, the receiver at the point of the collision between the streamer and the debris — which we call a streamer hit — records high-amplitude noise. Once the streamer hit occurs, extra tension is generated, which induces vibrational waves in the streamer. The result of this vibration is noise on the seismic records, i.e., strumming noise. Occasionally, debris at sea or fishing gear can tangle around the streamer. Tangling commonly occurs at the position of units mounted on the streamer, such as birds. The effect of adding extra weight on the streamer due to debris is an increase in tension, and as a result strumming noise may appear. Depending on the material of the debris, the streamer tends either to dive or to surface; thus, the adjacent streamer depth controllers try to compensate for the streamer movement to maintain the streamer at its nominal depth. This compensation may, in turn, generate additional noise.

Figure 9a shows an example of a shot gather with noise caused by tangled fishing gear on the streamer. Two distinctive noise events are observed. The first one, labeled (1), is high-amplitude, low-frequency energy that appears coherent within the affected part of the streamer. The second one, labeled (2), is strumming noise propagating in both directions along the streamer, which is probably caused by occasional hits on the streamer induced by the fishing gear. Such noise is often referred to as a “chevron” type of noise, where the hit occurred right at the apex of this chevron and with coherent noise propagating in both directions of the streamer. To preserve the signal, we addressed the noise in Figure 9a in several steps. For further details about the noise attenuation technique used, refer to Appendix E. Figure 9b and 9c shows results after the noise attenuation and difference plot, respectively. The frequency content of the noise removed is broadband, but with a stronger presence at the lower end of the frequency spectrum.

Snapping shrimp noise

Yet another source of ocean noise can be marine life, although these types of noise are generally rare, and at times somewhat exotic. The two dark spots in the wiggle-trace panels in Figure 10b, which is a magnification of the highlighted window in Figure 10a, are suggestive localized bursts of energy, most likely caused by snapping shrimp. These small creatures are capable of producing a loud sound by snapping their claws, and Everest et al. (1948) are probably the first to mention this phenomenon. Once the claw snaps, it emits cavitation bubbles capable of stunning a fish (Versluis et al., 2000).
receivers eventually record the cavitation. Snapping shrimp noise has the characteristics of a broadband burst of noise, affecting several adjacent traces. However, we still classify this noise as noncoherent from trace to trace. This background type of noise, although not very common, can be quite challenging to handle through processing. However, once the data are sorted to another domain, such as the channel or common receiver, it will appear as a collection of random individual bursts of energy. The middle image in Figure 10b shows the result after noise attenuation, where we can observe that the noise levels are significantly reduced. For further details about the noise attenuation technique used, refer to Appendix F.

**Spike-like noise**

In our next data example, we show noise that puzzled the seismic vessel crew for some time, having initially been misclassified. During acquisition of the seismic data, spikes appeared and disappeared on the data for several lines, as shown in Figure 11a, in which the highlighted dashed square represents a magnified area in wiggle-trace display in the upper right corner of the figure. The seismic vessel crew initially believed that the noise was due to an electrical problem. However, the root-mean-square (rms) plot in Figure 12 suggests that the individual spikes followed certain patterns in the source-receiver space, being close to stationary in the water and did not always appear on the same channels. Moreover, the amplitude and shape of the spikes varied. Combining all of these characteristics led to the belief that it was an external source — background type of noise, or potentially something colliding with the streamer. However, no debris were observed in the water. On the other hand, the presence of jumping “bait fish” and the significant barnacle growth on the streamers pointed to the hypothesis that larger predators such as tuna and barracuda might be attracted to feed around the area of data acquisition and occasionally hit the streamer. The noise attenuation results together with the difference plot are displayed in Figure 11b and 11c, in which the corresponding spectra are shown in Figure 11d–11g. For further details about the noise attenuation technique used, refer to Appendix F.
ation technique used, refer to Appendix G. The spectral plots suggest that the spikes are fairly broadband, but with peak values between approximately 2 and 25 Hz. Because the observed spikes appear as individual pulses of energy, we can classify them as noncoherent noise. This noise example demonstrates the importance of understanding the origin of the noise to avoid costly incorrect decisions such as replacing undamaged streamer sections.

**Instrument types of noise**

Seismic towed streamers contain large amounts of electronics, such as hydrophones and velocity sensors, incorporated inside a closed protective membrane. Streamers are designed in such a way that they can be deployed and retrieved from the water. This mechanical handling, however, could damage part of the streamer if not performed correctly. Moreover, all seismic equipments are operated in very dynamic marine environments; therefore, damage can be expected. Faulty hydrophones, for instance, may result in spiky traces.

**Cross-feed noise**

A typical example of instrument noise is cross-feed. This type of noise is defined as an electrical noise from channel to channel. Once apparent, it propagates to the other channels along the streamer almost instantaneously with velocity close to the speed of light. Therefore, such noise can occur on all channels at the same time within a shot gather. An example of this noise is shown in Figure 13a, in which the streamer was reported to have electrical leakage causing cross-feed noise, appearing as flat energy, highlighted with the black arrows.

A faulty sensor or group of faulty sensors is often the source of cross-feed. In most cases, the cause of this noise is due to bad electrical insulation, or water ingress, between the individual sections on the streamer. Some of the devices mounted on a streamer can also generate such electrical noise.

Historically, when strong electrical noise is apparent, as in the shot gather in Figure 13a, the streamer affected is typically flagged as “not to be processed” until the faulty equipment is repaired. To reduce the cost due to potential data reacquisition, the onboard processing team demonstrated that such noise could be effectively attenuated. The cross-feed levels were reduced through an approach of modeling the noise and subtracting it from the data. For further details about the noise attenuation technique used, refer to Appendix H. Figure 13b shows results after adaptively subtracting the noise, and the shot gather in Figure 13c represents the difference plot. The frequency content of our cross-feed example is broadband as shown in Figure 13d–13g.

**Streamer-mounted unit noise**

To keep a streamer neutrally buoyant and at a given depth, steering devices (birds) are mounted at regular intervals along the streamer. Mounted equipment may also include compasses, streamer recovery devices, and velocity meters, among other instrumentation. Although all of these devices have different purposes, for simplicity, we will use the common term “bird.” Birds mounted on the streamer induce noise, where the traces at the bird position can typically be three to four times noisier than adjacent traces (Schoenberger and Mifsud, 1974). Elboth et al. (2009a) suggest that noise peaks at the bird location could be the result of transversal vibrations transferred from the bird to the streamer or by interaction between the bird and the turbulent environment around the towed streamer. Figure 14a shows a magnified shot gather after 2.5 Hz low-cut filtering. The corresponding locations of the birds mounted on the streamer are above (series of spikes). Traces at each position where a bird is mounted show higher noise levels. Because the noise has a similar appearance to swell noise, i.e., vertical stripes of high amplitudes, a noise attenuation approach similar to the one used for swell noise can effectively attenuate the bird noise, as shown in Figure 14b (for more details on the approach used, refer to Appendix A). Steering-device-related noise attenuation (but for motion sensor data) based on dictionary learning is discussed by Turquais et al. (2017). Figure 14c represents the difference plot (noise removed), and no apparent damage of the signal is observed. The bird noise discussed in this example affects one or perhaps two adjacent traces; therefore, we classify it as noncoherent from trace to trace. The frequency spectrum of the difference plot shown in Figure 14d indicates that the dominating noise is mostly concentrated at approximately 5 Hz, as highlighted with a black arrow. This observation is supported by the f-k spectra shown in Figure 14e–14g.

**SUMMARY AND DISCUSSION OF TYPES OF NOISE**

We have presented various types of noise affecting marine towed seismic in this tutorial. Table 1 summarizes all types of noise presented together with their corresponding classification and techniques used to attenuate them. The proposed noise attenuation example workflows showed no apparent signal loss in the observation (shot) domain. However, to validate a noise attenuation workflow is not a straightforward process. Possible damages of the signal component may be more visible in other domains after sorting (e.g., common channel and common midpoint [CMP]), after stacking or even after migration. The rms maps extracted from

---

**Figure 10.** Example of snapping shrimp noise as seen on a shot gather with group spacing of 12.5 m. (a) Shot gather, where the noise is highlighted by the black arrows; (b) wiggle-trace magnification of the highlighted noise-contaminated area in (a), from top to bottom: before noise attenuation, after noise attenuation, and the difference of noise removed.
within the reservoir target or amplitude maps constructed along specific target horizons represent powerful ways of analyzing any potential signal loss. Such measures should be taken into consideration when choosing a specific noise attenuation sequence or when designing new tools to tackle the problem of noise. In this way, one can also analyze the consistency of the noise suppression as well.

Despite the apparent effectiveness of the proposed noise attenuation example flows, these methods may be further improved. In some cases, the algorithms are expensive to implement, whereas in other cases a precise parameterization is required and the chosen parameters typically have to be adjusted for some specific areas of the survey. More robustness is sought to reduce the manual factor of parameter testing.

Noise characteristics, and consequently the design of the noise attenuation algorithms used, sometimes depend on the type of streamer used at the acquisition phase. Fluid-filled streamers are more prone to transversal streamer vibrations, which are mostly contributing to the presence of swell noise on the seismic records. These streamers may work with larger sensor spacing if the vibration noise is attenuated sufficiently at high frequencies. Solid-filled streamers, on the other hand, are less sensitive to such types of vibration, and the swell noise is normally reduced. However, once the weather conditions deteriorate and more rattling of the streamer is observed, the swell noise increases significantly even for solid-filled streamers. We demonstrated in the swell noise example acquired with an analog group-formed (12.5 m) streamer that significant processing is required to target the noncoherent behavior of the swell noise. In case of point-receiver streamers with a 3.125 m interval (Pan and Moldoveanu, 2001), the swell noise appears more coherent compared to the group-forming case with a 12.5 m interval; still, significant processing is needed to attenuate the high levels of noise. Despite the more effective sampling of the swell noise when denser spatial sampling is used, the majority of the acquisition

Figure 11. Spike-like noise as seen on a shot gather with group spacing of 12.5 m, probably caused by feeding fish. (a) Before noise attenuation; (b) after noise attenuation; (c) the difference plot (with the spike-like noise removed); (d) frequency spectra (normalized to the largest peak): red, blue, and green — before, after, and the difference (with the spike-like noise removed), respectively; and (e–g) f-k plots — before, after, and the difference (with the spike-like noise removed), respectively. Spectra are calculated for the entire record length.
vessels (as of today) are still equipped with analog group-forming streamers. Depending on the coherency of the swell noise, different noise attenuation methods are used. In case of a point-receiver system, \( f-k \) dip filtering is a preferred choice, and correspondingly more “randomization”-like noise attenuation techniques for the analog group-formed type of acquisition. Motion sensors are much more exposed to vibration noise. Actually, a much denser spatial sampling down to 62.5 cm is required if we want to capture these transverse streamer vibrations (Paulson et al., 2015; Goujon et al., 2019).

One way of reducing the noise levels due to vibrations may be to tune the streamer stiffness to minimize the noise for different types of sensors. Another approach may require the introduction of some form of damping mechanism that will reduce the vibrations. This step will usually require the use of a fluid or oil that is placed within the sensor chamber. Primary concern here will be temperature variations that will change the viscosity and probably its
Figure 14. Bird noise as seen on shot gathers with group spacing of 12.5 m. The blue spikes on top of each shot gather show the position of the birds mounted on the streamer. (a) Shot gather after 2.5 Hz low-cut; (b) shot gather after noise attenuation; (c) the difference plot — noise removed; (d) frequency spectra (normalized to the largest peak): red, blue, and green — before, after, and the difference (with the bird noise removed), respectively; and (e–g) f-k plots — before, after, and the difference (with the bird noise removed), respectively. Spectra are calculated for the entire record length.

Table 1. Main characteristics of noise presented.

<table>
<thead>
<tr>
<th>Origin of the observed noise</th>
<th>Type of observed noise</th>
<th>Coherency in the observed domain</th>
<th>Frequency band of the noise (Hz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Background</td>
<td>Swell</td>
<td>□</td>
<td>2–15 Hz</td>
</tr>
<tr>
<td>Background</td>
<td>SI</td>
<td>✓</td>
<td>Broadband</td>
</tr>
<tr>
<td>Background</td>
<td>Strumming</td>
<td>✓</td>
<td>1–8 Hz</td>
</tr>
<tr>
<td>Background</td>
<td>Passing vessel</td>
<td>✓</td>
<td>70–80 Hz (in our example)</td>
</tr>
<tr>
<td>Background</td>
<td>Dual sensor</td>
<td>✓</td>
<td>Broadband, but stronger appearance on frequencies below 20 Hz</td>
</tr>
<tr>
<td>Background</td>
<td>Cable hit and fishing gear</td>
<td>✓</td>
<td>Broadband, but major energy on the low frequencies</td>
</tr>
<tr>
<td>Background</td>
<td>Snapping shrimp</td>
<td>□</td>
<td>Broadband</td>
</tr>
<tr>
<td>Instrument</td>
<td>Spike-like</td>
<td>□</td>
<td>Broadband</td>
</tr>
<tr>
<td>Instrument</td>
<td>Cross-feed</td>
<td>✓</td>
<td>Broadband</td>
</tr>
<tr>
<td>Instrument</td>
<td>Streamer mounted units</td>
<td>✓</td>
<td>5 Hz (peak value)</td>
</tr>
</tbody>
</table>
performance. New streamer designs were recently discussed by Goujon et al. (2019).

Apart from the types of noise included in this tutorial, one should also consider those generated by the marine seismic source itself such as multiples and simultaneous source acquisition (blending). Other types of noise not discussed in this tutorial include barnacle noise, marine mammal noise, turn noise, earthquake noise, lightning noise, and streamer power failure, among others.

CONCLUSION

Marine towed seismic processing benefit from knowledge of the types of noise that can be anticipated during seismic data acquisition. We have presented several typical data examples contaminated by various types of noise. These noise examples also represent typical noise problems that the industry is facing today. As such, they should be the benchmark for testing of new noise attenuation algorithms. Each type of noise was classified according to its main characteristics such as the noise origin, trace-to-trace coherency, and frequency band. By use of industry noise attenuation algorithms, we demonstrated that each type of noise presented could be effectively attenuated without apparent damage to the useful signal in the observation (shot) domain. Successful attenuation of such unwanted noise present in the marine tow environment has the capability of improving the data quality, reducing the seismic acquisition vessel downtime, and, therefore, increasing the overall productivity. We also demonstrated that accurate classification of noise could help in the decision-making process concerning line-acceptance criteria and in avoiding significant data reacquisition costs.

Such insights are important for onboard seismic quality control and processing because they can help the processing geophysicist tackle the noise problems at hand. Finally, we believe that the presented work in this paper will encourage new researchers to investigate complex noise problems and find solutions with potential significant industry impact.

ACKNOWLEDGMENTS

The authors would like to thank the editors, the reviewers, and S. McDonald for their helpful comments and suggestions that improved this work, as well as the CGG Data Library for permission to show these examples. This work is financially supported by the Research Council of Norway, project number 297180.

DATA AND MATERIALS AVAILABILITY

Data associated with this research are confidential and cannot be released.

APPENDIX A

SWELL NOISE REMOVAL

In our example flow of swell noise attenuation, we apply a conventional sliding-window approach (summarized in Table A-1). The swell noise must be seen as anomalous within the window chosen in step 1 (see Table A-1) of the attenuation flow. If the window is dominated by the swell noise, then the noise will not be seen as anomalous and thus will not be attenuated. Typical values for the temporal window length are between 1 and 2 s, but the spatial length depends on the width (in number of traces) of the swell noise stripes. To ensure that the noise is identified as anomalous, a common spatial window parameterization would be approximately three times larger than the width of the swell noise. To ease the discrimination between the signal and noise component, the swell attenuation process is run in frequency bands, as described in step 3 (see Table A-1). Commonly, these frequency bands are in the range of 4–6 Hz and may overlap. The user-defined threshold value in step 5 (see Table A-1) defines whether the trace is to be flagged as bad and reconstructed later or not. The higher the threshold values are, the less aggressive the attenuation process is. Typical values would be in the range of 2–4. Most of the time, we choose to apply the swell noise attenuation using more than one application window. In such a way, we can choose different parameters for the shallow and deep parts of the seismic section. Despite the given common parameters for swell noise attenuation in our example, these are to be considered illustrative and are subject to testing because they normally are data dependent.

Figure 1b depicts the resultant swell noise attenuation by using the above-explained conventional sliding-window approach, in which several passes of de-swell were used in the shot domain. Note that in Figure 1b nearly all of the swell noise has been successfully attenuated and the difference plot in Figure 1c shows no apparent damage to the signal. From the plots of the frequency and f-k spectra

Table A-1. Swell noise — example attenuation flow.

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Split input data into spatial and temporal windows in the t-x domain.</td>
</tr>
<tr>
<td>2</td>
<td>Transform the individual windows to the f-x domain.</td>
</tr>
<tr>
<td>3</td>
<td>Select a band of frequencies, and calculate the median amplitude of the window.</td>
</tr>
<tr>
<td>4</td>
<td>Calculate the ratio of the amplitude of each trace to the median amplitude of the window.</td>
</tr>
<tr>
<td>5</td>
<td>If the calculated ratio is greater than a user-defined value (commonly referred threshold), then flag the amplitude band for that trace in question as bad.</td>
</tr>
<tr>
<td>6</td>
<td>Repeat for all traces in the window, and repeat over all frequency bands to identify and flag bad data.</td>
</tr>
<tr>
<td>7</td>
<td>Construct an f-x decon filter (Gülünay, 1986) for every frequency and apply the filter to reconstruct and replace all bad data at each frequency.</td>
</tr>
<tr>
<td>8</td>
<td>Repeat over all spatial and temporal windows.</td>
</tr>
<tr>
<td>9</td>
<td>Transform back to the t-x domain.</td>
</tr>
</tbody>
</table>
shown in Figure 1d–1g, we can observe that the frequency span of the attenuated noise is in the range of 2–15 Hz.

Besides using a combination of algorithms, it is common practice to sort the data to different domains, such as the common receiver, channel, or CMP, in which the noise appears more random, which benefits most of the available noise attenuation tools.

APPENDIX B
SI NOISE REMOVAL

As a visual example of how an SI noise reduction method works, Figure 3a–3c shows, respectively, a shot gather from a recent survey, the same shot gather after SI attenuation, and the difference plot. This example was produced onboard the seismic vessel because SI evaluation was part of the line validation process. As discussed previously, Figure 3a shows interference from two different sources. We classified the noise labeled (1) in Figure 3a as broadband coherent energy from trace to trace but noncoherent from shot to shot. We classified the noise labeled (2) as broadband coherent energy from trace to trace and shot to shot. This combination of characteristics makes the attenuation of such noise challenging. Once transformed to the tau-p domain, the interference maps onto a narrow area because it is nearly linear in the time domain. The reason for this narrow area is because, in case of shot domain data, we obtain the tau-p transform by summing energy along straight lines, in which each line is defined by its slope (slowness) (p) and its intercept time or zero-offset crossing (tau). The summed energy along these lines maps to a single point in the tau-p domain. However, because the SI noise is continuous in the time domain, it remains continuous in the tau-p domain as well. To evaluate the noise effectively, the onboard processing team used the SI attenuation flow described in Table B-1. Once we break up the shot-to-shot coherency of the SI noise by borrowing shots from neighboring line (e.g., step 2 in Table B-1), the data are ready to be mapped to the tau-p domain. However, conventional tau-p transform suffers from data leakage among different slowness values. One solution proposed by Herrmann et al. (2000) is adding a sparseness constraint by fitting the input data with a sparse (spiky) tau-p model. Wang and Nimsaila (2014) extend this approach to a progressive fitting from stronger to weaker events through sparse tau-p inversion, which led to a more optimal tau-p model that better represents the input data.

APPENDIX C
STRUMMING NOISE REMOVAL

Strumming noise is typically addressed by the application of either f-k or tau-p filtering because the noise is coherent from trace to trace and it separates from the signal component in both transformed domains. However, in our example data (see Figure 4a), the dip of the abnormally high-amplitude H/T strumming noise is partially overlapping with that of the water bottom reflection (the results are not shown here) and the conventional filtering methods resulted in signal loss. Because the noise was an issue in the acquisition phase, the onboard processing team developed a model-based subtraction technique. Figure C-1a–C-1h visualizes the processing sequence used to address the noise (for detailed description, see Table C-1). The result of the model-based subtraction and the corresponding difference plot are shown in Figure 4b and 4c, respectively. The strumming noise is effectively reduced, and no apparent damage to the signal is observed.

As demonstrated in Figure 4b, these abnormal levels of strumming noise could be effectively attenuated by seismic processing. Therefore, no lines were rejected during the acquisition, which avoided the significant cost of reacquiring the data.

APPENDIX D
VERTICAL PARTICLE VELOCITY-SENSOR NOISE REMOVAL

In our example flow of vertical component noise attenuation, we apply a hydrophone-guided approach (summarized in Table D-1). The hydrophone and the vertical component data are first trans-

<table>
<thead>
<tr>
<th>Table B-1. SI noise — example attenuation flow.</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Step 1: Select a sail line (a source-cable pair) that has shot-to-shot coherent SI noise in the t-x domain (such as the SI noise labeled (2) in our data example in Figure 3a).</td>
</tr>
<tr>
<td>• Step 2: Borrow shots from a neighboring sail line acquired in the same direction, and select the same source-cable pair to break up the shot-to-shot coherency, e.g., line mixing (Elboth et al., 2017).</td>
</tr>
<tr>
<td>• Step 3: Transform to the tau-p domain by using the progressive sparse transform (Wang and Nimsaila, 2014), and sort the data in p-shot gathers.</td>
</tr>
<tr>
<td>• Step 4: Identify and scale down the tau-p coefficients corresponding to SI noise based on the median value in the sliding-window approach (Zhang and Wang, 2015).</td>
</tr>
<tr>
<td>• Step 5: Subtract the result in step 3 from that in step 4 to obtain the SI noise model.</td>
</tr>
<tr>
<td>• Step 6: Sort back to shot-p gathers and transform the SI noise model back to the t-x domain.</td>
</tr>
<tr>
<td>• Step 7: Remove the borrowed shots and subtract the SI noise model from the input data in step 1.</td>
</tr>
</tbody>
</table>
formed using a complex high-angular-resolution wavelet transform (Peng et al., 2013). Such a transform separates events in the shot gather based on their dip direction, frequency, and location. Once the transformation is used, noise can be distinguished from the desired common signal events by comparing the corresponding complex coefficients, which represent the input data (Peng and Huang, 2014). As demonstrated, the noise level in Figure 8c is significantly reduced. However, further noise attenuation processing could be considered because some remnant noise is visible in Figure 8c.

**APPENDIX E**

**STREAMER HIT NOISE REMOVAL**

The two distinctive noise events highlighted in Figure 9a are addressed using different attenuation routines. A summary of the attenuation flow used is summarized in Table E-1. The result of the complete example noise attenuation flow is displayed in Figure 9b. The noise caused by the fishing gear is well attenuated, without any apparent damage to the signal visible in the difference plot neither in Figure 9c nor in the frequency and f-k spectra plots shown in Figure 9d–9g.

**APPENDIX F**

**SNAPPING SHRIMP NOISE REMOVAL**

A summary of the snapping shrimp noise attenuation flow is given in Table F-1. Figure 10b (from top to bottom) shows the wiggle-trace magnified area of snapping shrimp noise before and after attenuation and the difference plot, respectively. It follows from this figure that no apparent damage to the signal is observed. Note that in areas where a large colony of such shrimp is present, which is typical in warm and shallow waters, we might experience more background noise compared with that shown in our example. In such a case, a combination of different attenuation algorithms may be beneficial, together with applications in more than one processing domain.

**APPENDIX G**

**SPIKE-LIKE NOISE REMOVAL**

The abnormally high level of observed spikes in Figure 11a is most likely caused by feeding fish that hit the streamer occasionally. The spiky samples were identified and reconstructed in the f-x domain and transformed back to the time domain, where the results before and after attenuation and the difference of the spike attenuation flow are shown in Figure 11a–11c, respectively. A fairly small sliding-window size was used in order to not harm any signal. The attenuation flow is described in Table G-1.

**APPENDIX H**

**CROSS-FEED NOISE REMOVAL**

Cross-feed noise tends to be challenging to attenuate because its amplitude, phase, and frequency content often overlap with those of the signal, especially at near offsets. Figure 13b shows results after adaptively subtracting the noise, and the shot gather in Figure 13c represents the difference plot. The cross-feed attenuation workflow used is described in Table H-1 and it shares similarities with the one applied in the case of strumming noise. Note that the noise is almost completely attenuated. For that reason, the streamer was not removed from the coverage and no data had to be reacquired, which in turn reduced the acquisition cost significantly.

Figure C-1. Strumming noise model building. (a) Shot gather affected by strumming noise, (b) shot gather after forward linear-moveout (LMO) correction, (c) gather after low-pass filter, (d) gather after f-k filter, (e) the difference (c) minus (d), (f) shot stack of (e), (g) stacked trace duplicated for all receivers in the shot group, and (h) reverse LMO correction — noise model.
Table C-1. Strumming noise — example attenuation flow.

• Step 1: A LMO correction, following the positive dip of the dominating noise, is applied to “flatten” it (see Figure C-1b).
• Step 2: The LMO-corrected shot gather is low-pass filtered keeping the dominating noise frequencies (see Figure C-1c).
• Step 3: An f-k filter is used, targeting only the “flat” energy (see Figure C-1d).
• Step 4: Subtraction of the result in step 2 from that in step 3 gives the initial noise model (see Figure C-1e).
• Step 5: To overcome possible signal leakage in the noise model obtained in step 4, the first 100 traces are stacked. Because the noise is flat, it stacks well, whereas the remnant signal does not. The resultant shot stack is displayed in Figure C-1f, in which each stacked trace represents one shot gather.
• Step 6: By duplicating the stacked trace for all receivers in a shot group, a shot gather is formed again (see Figure C-1g).
• Step 7: Reversing the LMO, the final noise model is obtained (see Figure C-1h).
• Step 8: The noise model is adaptively subtracted from the input data.

Table D-1. Dual-sensor streamer noise — example attenuation flow.

• Step 1: Remove frequencies of less than approximately 20 Hz from the velocity-sensor data.
• Step 2: Use hydrophone-guided noise attenuation based on the wavelet transform, in which the common components between the hydrophone and the vertical velocity data are distinguished and kept untouched, while any noise is attenuated (Peng and Huang, 2014).

Table E-1. Fishing gear/streamer hit noise — example attenuation flow.

• Step 1: The low frequencies (below 8 Hz) are attenuated using the same techniques as in the case of swell noise (shot domain).
• Step 2: Data are then sorted to the channel domain, and time-frequency noise attenuation (Elboth et al., 2010) is used for the whole spectrum, but it is limited to the noisiest traces (in our example, traces 315–385).
• Step 3: A second pass of noise attenuation in the shot domain is used by the same technique as in step 1, with the purpose of targeting remnant low-frequency noise up to 6 Hz.
• The steps from 1 to 3 attenuate the noise component labeled (1) in Figure 10a. To attenuate the second noise component labeled (2), the following step was added.
• Step 4: Application of conventional f-k filtering limited to 12 Hz is used.

Table F-1. Snapping-shrimp noise — example attenuation flow.

• Step 1: Sort data to the channel or common-receiver domain to enhance randomness.
• Step 2: Use a conventional technique such as time-frequency noise attenuation (Elboth et al., 2010) to attenuate the shrimp noise (and the additional swell noise).

Table G-1. Spike-like noise — example attenuation flow.

• Step 1: Use sliding-window technique in the f-x domain similar to the swell-noise case. However, to not harm any signal, small-sized windows are selected.
• Three different iterations were used with the following temporal window length (in ms) and spatial window length (in number traces): 200 ms and 24 traces, 100 ms and 24 traces, and last iteration 50 ms and 12 traces.
• Step 2: A local rms value is calculated from the data within a given window, and sample values exceeding the local rms value multiplied by a user-defined factor (in our example, equal to 2) are flagged as bad.
• Step 3: These flagged samples are reconstructed either from the use of f-x decon filtering (Güllüay, 1986) or the noise is transformed back and subtracted from the original data. Reconstruction was used in our data example.
Step 1: All of the shots for the streamer in question are stacked. Because the cross-feed noise appears as flat coherent energy in the shot domain, it stacks with high coherency.

Step 2: By forming a shot-stack section, the noise is represented as individual spikes (see Figure H-1a).

Step 3: Spike attenuation then can be performed using a flow similar to the one used in the case of spike-like noise in Appendix G.

Step 4: Subtraction of the result in step 2 from that in step 3 gives the noise model (see Figure H-1b).

Step 5: Each trace of the noise model is then duplicated to form a shot gather again.

Step 6: Each noise gather is adaptively subtracted from the original shot gathers.

Table H-1. Cross-feed noise — example attenuation flow.

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>All of the shots for the streamer in question are stacked. Because the cross-feed noise appears as flat coherent energy in the shot domain, it stacks with high coherency.</td>
</tr>
<tr>
<td>2</td>
<td>By forming a shot-stack section, the noise is represented as individual spikes (see Figure H-1a).</td>
</tr>
<tr>
<td>3</td>
<td>Spike attenuation then can be performed using a flow similar to the one used in the case of spike-like noise in Appendix G.</td>
</tr>
<tr>
<td>4</td>
<td>Subtraction of the result in step 2 from that in step 3 gives the noise model (see Figure H-1b).</td>
</tr>
<tr>
<td>5</td>
<td>Each trace of the noise model is then duplicated to form a shot gather again.</td>
</tr>
<tr>
<td>6</td>
<td>Each noise gather is adaptively subtracted from the original shot gathers.</td>
</tr>
</tbody>
</table>

Figure H-1. Cross-feed as observed in the shot stack, used for noise model building. (a) Shot stack affected by cross-feed and (b) the difference plot after despiking or noise model.
Noise types and their attenuation

W19

Hill, D., C. Combee, and J. Bacon, 2006, Over/under acquisition and data processing: The next quantum leap in seismic technology?: First Break, 24(1), 81–95.


Biographies and photographs of the authors are not available.
2.4 Seismic data processing and imaging

Seismic data are costly to acquire and sparse in nature. In addition, as we demonstrated, the data are rather noisy. Consequently, the recorded seismic data should go through a data processing sequence for noise suppression and interpolation onto a denser/regular grid.

Figure 2-7 shows a simplified processing workflow. The recorded raw data, usually in the format of SEGD, are first reformatted. Data that do not meet the contracted tolerances are edited/removed during this step. This edition may include whole lines, part of lines, specific shots, and or channels that should not be processed. The following step involves merging the seismic and navigation data that are usually in the format of P1-90. The foremost de-noise step is at the beginning of the sail-line processing, where processes like de-spike, swell noise attenuation, linear noise attenuation, external noise attenuation, de-blend, de-bubble, and de-ghost take place. The offline de-noise and offline processing usually involve specific offline workflows such as multiple model creation or velocity picking. Once the de-multiple is completed, any residual noise or even residual multiple are addressed in the last step of the sail-line processing. The simplified processing workflow continues with offset class processing and imaging. The first involves interpolation and regularization of seismic data. The latter aims to move the reflection events to their true subsurface location both in space and time. The post-migration processing usually involves additional de-noise and residual move-out correction before stack, followed by cosmetic signal enhancement.

Once the seismic data acquisition, processing, and imaging are completed, we can infer the geology at some depth from the processed seismic image (Figure 2-2). This product is finally used to extract subsurface properties of the Earth. The process is rather long, where processing and interpretation depend on each other. Firstly, proper parameterization of each processing step is required, which is time-consuming. Secondly, on data from large seismic surveys, specific processing steps can require a few weeks of CPU/GPU time to complete. This leaves little to no room for errors in a commercial setting because of the complexity and the tight deadlines. Mistakes in these processes are costly and might even lead to incorrect decisions further down the processing/interpretation pipeline. Thus, there is a demand for more time-efficient processing solutions. Ideally, DNN represents a possible solution that enables computers to learn complex non-linear mapping functions, potentially automizing specific processing tasks and consequently reducing cost. Therefore, the following chapters
will introduce DNN before presenting the developed applications as part of this thesis. This work may benefit seismic data processing, reduce costs, and improve data quality.

Figure 2-7. *A simplified processing workflow from raw data to final output.*
3. Deep Learning

This chapter introduces Deep Learning (DL) and covers some essential aspects required for a reader to understand the concepts discussed later. The chapter includes some of the basic fundamentals of DL, which consists of the forward algorithm, nonlinear activations, how to train a DL, the backpropagation algorithms, and the optimization algorithm. The focus is on Convolutional Neural Networks (CNNs), which are the type of networks used in this thesis. Then we discuss common problems of DL in seismics and give an overview of DL methods for noise attenuation and interpolation of seismic data. The theoretical framework in the following sections is based on the work of Nielsen (2015), Mehta et al. (2019), and Deisenroth et al. (2020).

3.1 Deep learning fundamentals

Machine Learning (ML) is the discipline or a collection of mathematical algorithms that make computers understand relationships between data based on automated learning from the data. In other words, ML is a technique for estimating an unknown underlying function using the data, i.e., a function approximator. Leshno et al. (1993) have shown that a neural network (NN) can uniformly approximate any continuous function under certain conditions.

Before moving forward, let us look at a simple example and introduce some of the main ML ingredients, which will be explained in more detail later. Let \( \{(x_1, y_1), (x_2, y_2), ..., (x_N, y_N)\} \) be our dataset, where the input matrix \( X := [x_1, x_2, ..., x_N] \in \mathbb{R}^{N \times D} \), for example, it represents a series of images of handwritten digits from 0 – 10 for convenience in vector form. Each entry in the vector represents a single pixel value in the image. Our label, or target, data \( y := [y_1, y_2, ..., y_N] \in \mathbb{R}^N \), denotes a scalar from 0 – 10 that defines the number in the images. For example, if a particular image, \( x_2 \), depicts 3, then \( y_2 = 3 \). We can then define a model \( f(X; \theta) \), which is a function \( f: X \rightarrow y \) of the parameters \( \theta \). Then we can use \( f \) to make a prediction from the input. To judge how well the model performs, we need to define a cost function \( C(y, f(X; \theta)) \), that measures the misfit between the label and our prediction. ML represents algorithms that can fit the model by finding a set of parameters \( \theta \), namely weights and bias, that minimize the cost function. The latter is achieved through an optimization algorithm. There are different ways of making a machine learn. ML methods include supervised learning, unsupervised learning, and reinforcement learning. The ML learns from labeled data in supervised learning, i.e., a collection of images labeled as, for
instance, a cat or not a cat. Everyday supervised learning tasks include classification and regression, where classification is the task of predicting a discrete class label, whereas regression predicts a continuous quantity. In this thesis, we explore the latter one. In unsupervised learning the algorithm finds patterns and structures, i.e., clustering or decision boundary. Reinforcement learning is about making suitable decisions to maximize reward in a particular situation, i.e., a delivery drone navigating through a complex environment being rewarded to actions helping the drone reach the desired destination.

Recently, Deep Neural Networks (DNN) or Deep Learning (DL), a subfield of ML, have emerged as one of the most widespread and powerful techniques used. DNNs have a long history (Bishop, 1995; Hinton and Salakhutdinov, 2006; LeCun et al., 2015; Mehta et al., 2019), but have caught the ML community’s attention as late as in 2012 when the AlexNet DNN model (Krizhevsky et al., 2012) reduced the error rate by twelve percent on the ImageNet Challenge (Deng et al., 2009). Further improvements followed with even lower error rates (Russakovsky et al., 2015; He et al., 2016). This challenge is considered the benchmark competition and database in computer vision. Ever since, DNNs have become a powerful toolkit for image and speech recognition tasks and led to many open-source libraries, like Scikit-learn (Pedregosa et al., 2011), TensorFlow (Abadi et al., 2016), PyTorch (Paszke et al., 2017), and Keras (Chollet, 2018), that makes it reasonably easy to deploy a DNN. The above improvements in computer vision and open-source packages contributed to the wide adaptation of DNNs in numerous fields of science, including geoscience.

3.1.1 The forward algorithm

A NN consists of neurons stacked into layers, where the output of one layer serves as input to the next layer. The first and the last layers usually are referred to as the input and the output layer, while any layer in between is called a hidden layer. If we have a vector of $n$ input features $\mathbf{x} = [x_1, x_2, \ldots, x_n]$, which we forward to a neuron that has one weight, $w_i$, for each value of $\mathbf{x}$, then the neuron will produce a linear combination that takes the form of a dot product of the input with a set of neuron-specific weights:

$$z = \sum_{i=1}^{n} w_i x_i + b,$$

(3-1)

where $b$ is a bias added to the output to reduce the chances of unwanted dead neurons. The
vectorized form follows as:

\[ z = w^T x + b \quad (3-2) \]

For the case of multi-neuron NN, we can rewrite the equation using matrix notations as:

\[ z = W^T x + b, \quad (3-3) \]

where \( W^T \) is the weight matrix. As the output \( z \) represents a linear operation, we need to introduce nonlinearity. To do so, we pass the linear output through an activation function. The output is then calculated as:

\[ a = \sigma(z), \quad (3-4) \]

where \( \sigma(z) \) is the nonlinear activation function. If we do not introduce nonlinearity, the entire NN can be represented as a collection of linear operations, i.e., linear function. Such functions have limited complexity, because the problems we aim to solve with NN in most cases do not have any good linear approximations, it is beneficial to introduce nonlinearity.

Figure 3-1 illustrates the forward algorithm of a NN. Figure 3-1a shows a node with three inputs, and one output passed through a nonlinearity. Figure 3-1b represents a NN where the neurons are stacked into layers, and the network has two hidden layers.

To summarize the forward algorithm, let the activation \( a^l_j \) of the \( j^{th} \) neuron in the \( l^{th} \) layer in the forward algorithm be related to the activities of the neurons in the layer \( l-1 \) by:

\[ a^l_j = \sigma \left( \sum_k w^l_{jk} a^{l-1}_k + b^l_j \right) = \sigma(z^l_j), \quad (3-5) \]

where we assume that there are \( L \) layers with \( l = [1,2, ..., L] \) indicating the layer, \( w^l_{jk} \) denotes the weight for the connection from the \( k^{th} \) neuron in layer \( l-1 \) to the \( j^{th} \) neuron in layer \( l \), and \( b^l_j \) denotes the bias of this neuron. In Equation (3-5), we assume that the nonlinear activation function \( \sigma(\cdot) \) is the same for every layer within the NN. However, different layers can use different nonlinearities.
3.1.2 Nonlinear activation functions

As discussed by the images of handwritten digits example in section 3.1, we had a highly complex problem representing a typical NN task. Linear models can hardly approximate such tasks. Because the problems we try to approximate with NN typically are highly nonlinear, it is beneficial to include nonlinear activation functions. Activation functions represent a crucial feature in NN (Sibi et al., 2013); without them, the whole NN would be a collection of linear operations.

Nonlinearities such as step-functions, sigmoids, and the hyperbolic tangent used to be commonly employed in the NN. Today, it has become more common to use the rectified linear unit (ReLU) and the leaky rectified linear unit (leaky ReLU), which are the nonlinearities used in this thesis. Choosing different nonlinearities leads to different training capabilities of the NN. This difference is because the training, and more specifically the backpropagation algorithm (explained later in 3.1.4), which adjusts the NN parameters, requires us to take the derivatives of these nonlinear functions with respect to the NN parameters. Therefore, the nonlinear functions have to be differentiable.

The rectified linear unit (ReLU) has become very popular and is ascribed to the recent great improvement in the performance of NN (Goodfellow et al., 2016). The function returns 0 if it receives any negative number but returns the input value for any positive value $z$. It can be written as:
\[
\sigma(z) = \begin{cases} 
z, & z \geq 0 \\
0, & \text{otherwise}
\end{cases}
\] (3-6)

Strictly speaking, the ReLU function is not differentiable at \( z = 0 \), which in theory is a shortcoming. A way of reducing the change of zero gradients, when \( z = 0 \) is to introduce a small slope for negative arguments, which is also known as the Leaky ReLU:

\[
\sigma(z) = \begin{cases} 
z, & z \geq 0 \\
\alpha z, & \text{otherwise}
\end{cases}
\] (3-7)

where \( \alpha \) is a hyperparameter that controls the slope. However, experience have shown that both the ReLU and the Leaky ReLU perform equally well in practice.

### 3.1.3 Training Deep Neural Networks

Assuming that we have defined our NN architecture (number of layers and type of nonlinear activations), which depends on the task, the amount of data, and the computational resources available, we need to train that NN. First, we need to divide our data into two portions, the dataset used for training and a smaller validation set. The training set is used for updating the NN parameters. In contrast, the validation set is used only to test the performance of NN on unseen data, thus checking the network's ability to generalize. We plot both the training and the validation error to monitor the NN. Ideally, both error metrics should decrease and converge during the training process. Before doing that, we need to define a cost function. The cost function is a way of determining how well the NN performs. Consider the label \( y_i \), and the corresponding prediction \( a^L_i \) given by the NN that makes based on the input \( x_i \). Then if our training set is defined as \( \{(x_1, y_1), (x_2, y_2), ..., (x_N, y_N)\} \), where \( X := [x_1, x_2, ..., x_N] \in \mathbb{R}^{N \times D} \) and \( y := [y_1, y_2, ..., y_N] \in \mathbb{R}^N \), we can calculate the cost function expressed in a matrix form by:

\[
C(W, b) = \frac{1}{N} \| y - a^L \|^2,
\] (3-8)

where \( W \) and \( b \) represent the weights and biases of all layers and and \( a^L := [a^L_1, a^L_2, ..., a^L_N] \in \mathbb{R}^N \) is the prediction of the NN. The cost function, in this case, is given by mean square error (MSE), which is sensitive to outliers because it penalizes high errors as it is based on the error squared. Alternatively, one can use the mean square error (MAE):
\[ C(W, b) = \frac{1}{N} \|y - a^t\| \]  

MAE is not so sensitive to outliers, thus will not penalize high errors. However, its gradient has a constant slope around 0 value, which might cause converging issues due to high gradients.

The cost function is often supplemented by additional terms that implement regularization on the NN parameters, more specifically on the weights term. Regularization, also known as the “penalty term”, reduces the risk of overfitting, a phenomenon when the NN fits too closely with the training data and does not generalize well on new data (Mitchell, 1997; Deisenroth et al., 2020). Overfitting is a major problem in NN, especially for modern networks with many parameters. An indication of overfitting is when the loss of the train set keeps decreasing, whereas the validation set one is stationary or even starts increasing. To add a penalty term to the cost function, in this case, the MSE, we modify Equation (3-8) to:

\[ C(W, b) = \frac{1}{N} \|y - a^t\|^2 + \lambda \|W\|^2, \]

where we added the second term that minimizes the weights, scaled by \( \lambda \in [0,1] \), namely the regularization parameter. Adding this second term to the cost function can be seen as compromising between finding small weights and minimizing the original cost function, with \( \lambda \) controlling the importance of the term. The regularization term reduces the risk of the NN to learn the effect of local noise in the data, for example, because the smallness of the weights ensures that the behavior of the NN will not change much if we change a few random inputs (Nielsen, 2015). Said in other words, the regularization constrains the NN to build a relatively simple model based on patterns seen often in the training data, making the NN resistant to learning peculiarities of the noise. In contrast, non-regularized NN can build a more complex model with large weights that carry information about the data's noise, making the NN more likely to change its behavior with small changes in the input.

Another way to reduce the risk of overfitting is by introducing dropout (Srivastava et al., 2014). Dropout works by randomly disabling a user-defined percentage of neurons and their corresponding connections. This technique prevents the NN from relying too much on single neurons and forces all neurons to learn, which results in better generalization of the
model.

Having defined a NN architecture and cost function, we can train the model. The NN optimizes its parameters in such a way as to minimize the cost function. This step requires calculating the gradients of the NN with respect to its parameters through a specialized algorithm called backpropagation. Once both the train and the validation set have gone through the forward and backpropagation algorithm, one epoch is completed. It is common to run multiple epochs to get satisfying results. When the model is trained, we usually use new data set, different from the train and validation sets, to check its performance and generalization abilities.

### 3.1.4 The backpropagation algorithm

The training process aims to find the NN parameters that minimize the cost function. To do so, we need to know how small changes in these parameters affect the cost function. These small changes are equivalent to calculating the partial derivatives of the cost function with respect to its parameters. Therefore, the backpropagation represents an ordinary partial differential chain rule that calculates the derivatives of the cost function with respect to all parameters of the NN, namely $\frac{\partial C}{\partial w_{jk}}$ and $\frac{\partial C}{\partial b_j}$.

Now let us recall Equation (3-5): $a_j^L = \sigma \left( \sum_k w_{jk}^L a_k^{L-1} + b_j^L \right) = \sigma(z_j^L)$. By definition, the cost function $C$ depends directly on the output layer $a_j^L$ and on the activities of all the neurons in the previous layers. If we denote the error of the $j^{th}$ neuron in the last layer $L$ by $\Delta_j^L$ as the change in the cost function with respect to the weighted input $z_j^L$:

$$\Delta_j^L = \frac{\partial C}{\partial z_j^L} \quad (3-11)$$

In the last expression, the often-used term error could be thought of as sensitivity instead. Similarly, we can define the error of neuron $j$ in layer $l$, $\Delta_j^l$, as the change of cost function with regards to the weighted input $z_j^l$. 
\[
\Delta_j^l = \frac{\partial C}{\partial z_j^l} = \frac{\partial C}{\partial a_j^l} \frac{\partial a_j^l}{\partial z_j^l} = \frac{\partial C}{\partial a_j^l} \sigma'(z_j^l),
\]

(3-12)

where \(a_j^l = \sigma(z_j^l)\), thus the term \(\frac{\partial a_j^l}{\partial z_j^l}\) becomes \(\sigma'(z_j^l)\).

This error function represents the change of cost function with regards to the bias input \(b_j^l\):

\[
\frac{\partial C}{\partial b_j^l} = \frac{\partial C}{\partial b_j^l} \frac{\partial b_j^l}{\partial z_j^l} = \frac{\partial C}{\partial z_j^l} = \Delta_j^l
\]

(3-13)

wherein Equation (3-13), we used the fact that \(z_j^l = \sum_k w^l_{kj} a^l_{k-1} + b_j^l\), and therefore \(\frac{\partial b_j^l}{\partial z_j^l} = 1\).

To propagate the error backward and calculate \(\Delta_j^l\) for all layers, we can use the fact that the error depends on neurons in layer \(l\) only through the activation of neurons in the next layer \(l+1\); thus, we need to rewrite Equation (3-11) in terms of \(\Delta_k^{l+1} = \frac{\partial C}{\partial z_k^{l+1}}\). Applying the chain rule we have:

\[
\Delta_j^l = \frac{\partial C}{\partial z_j^l} = \sum_k \frac{\partial C}{\partial z_k^{l+1}} \frac{\partial z_k^{l+1}}{\partial z_j^l} = \sum_k \Delta_k^{l+1} \frac{\partial z_k^{l+1}}{\partial z_j^l}
\]

(3-14)

To evaluate \(\frac{\partial z_k^{l+1}}{\partial z_j^l}\), we use Equation (3-5), which tells us that \(z_k^{l+1}\) and \(z_j^l\) are connected via:

\[
z_k^{l+1} = \sum_j w^l_{kj} a_j^l + b_j^l = \sum_j w^l_{kj} \sigma(z_j^l) + b_j^l
\]

(3-15)

Hence,

\[
\frac{\partial z_k^{l+1}}{\partial z_j^l} = w^l_{kj} \sigma'(z_j^l),
\]

(3-16)

which substituted in Equation (3-14), gives:
\[ \Delta_j^l = \sum_k \Delta_k^{l+1} w_{kj}^{l+1} \sigma'(z_j^l) \] (3-17)

The last step in the algorithm is differentiating the cost function with respect to the weight:

\[ \frac{\partial C}{\partial w_{jk}^l} = \frac{\partial C}{\partial z_j^l} \frac{\partial z_j^l}{\partial w_{jk}^l} = \Delta_j^l a_k^{l-1}, \] (3-18)

where in the last line, we have used the fact that by definition, \( \Delta_j^l = \frac{\partial C}{\partial z_j^l} \) and because \( z_j^l = \sum_k w_{jk}^l a_k^{l-1} + b_j^l \), we have \( \frac{\partial z_j^l}{\partial w_{jk}^l} = a_k^{l-1} \).

The combination of Equation (3-12) to (3-18) allows us to generate an efficient backpropagation algorithm and calculate the gradients with respect to all parameters as summarized in Table 3-1.

Table 3-1 – The backpropagation algorithm (after Nielsen, 2015; Mehta et al., 2019).

1. **Activation at input layer:** calculate the activations \( a_j^1 \) of all neurons in the input layer.

2. **Feedforward:** starting from the first layer, forward propagate through Equation (3-5) to compute \( z^l \) and \( a^l \) for each layer.

3. **Error at output layer:** calculate the error at the output layer using Equation (3-12).

4. **Backpropagate the error:** use Equation (3-17) to propagate the error backward and calculate \( \Delta_j^l \) for every layer.

5. **Calculate gradient:** use Equations (3-13) and (3-18) to calculate \( \frac{\partial C}{\partial b_j^l} \) and \( \frac{\partial C}{\partial w_{jk}^l} \).

Figure 3-2 visualizes the forward and backpropagation algorithms. The figure illustrates an arbitrary node in a network. The green arrows denote the forward algorithm, which goes from left to right, and the red arrows denote the backpropagation algorithm from right to left.

### 3.1.5 The optimization algorithm

Gradient descent is one of the most popular optimization algorithms used in NN. The basic idea behind it is straightforward: iteratively adjust the NN parameters in the opposite direction of the gradient of the cost function with regards to its parameters. In such a way, the algorithm
ensures that the parameters move towards a minimum of the cost function. That function is almost always unknown. The problem here is that the functions that we optimize are complex, rugged, non-convex in high dimensional space with many local minima.

There are three different ways of using gradient descent, which differ in how much data we use to compute the cost function gradient. One way is to calculate the gradient for the entire dataset at once to perform a single update of the parameters. This method is known as the “vanilla gradient descent” but is very slow. Another way is to calculate the gradient for each training example, referred to as stochastic gradient descent. Because of the frequent update of the parameters, the convergence of this approach is complicated. One of the most widely-adopted variants of the gradient descent is the stochastic gradient descent with mini-batches, where the size of the mini-batches is always smaller than the total number of data examples. The gradient descent is defined by:

$$\theta_t = \theta_{t-1} - \frac{\eta}{b} \left( \frac{\partial C}{\partial \theta_t} \right),$$

(3-19)

where $\theta$ denotes the NN parameters (weight or bias) at time step $t$, $\eta$ is the learning rate which
determines the step size at each time step in the gradient direction, $b$ is the batch size, and $C$ is the cost function. However, the gradient descent algorithm has limitations (Mehta et al., 2019). The method is sensitive to choices of the learning rate. A learning rate that is too small leads to slow convergence, whereas a learning rate that is too large can cause fluctuations around the sought minima. Additionally, the same learning rate applies to all parameter updates in all directions in the parameter space. The algorithm may get trapped in saddle points because of the complex and non-convex functions that we optimize (Dauphin et al., 2014). Therefore, numerous modifications exist that help the gradient descent overcome these limitations, like adding momentum (Qian, 1999) that helps accelerate the algorithm in the relevant direction, adaptive learning rates for each parameter (Duchi et al., 2011; Hinton et al., 2012; Zeiler, 2012), adaptive moment estimation (Kingma and Ba, 2014).

### 3.2 Convolutional Neural Network

Convolutional Neural Network (CNNs) is a class of DNN that has become dominant in various computer image tasks. Any NN with at least one convolutional layer can be classified as CNN, according to Goodfellow et al. (2016). An image is a matrix of pixel values and is generally arranged into a one-dimensional vector and fed into the NN for a typical classification problem. Unfortunately, this vectorization throws away lots of the spatial information contained in the image (Mehta et al., 2019). In contrast, CNNs are better suited for images where neighboring pixels are connected in specific patterns (Goodfellow et al., 2016). Therefore, CNNs are designed automatically to learn spatial hierarchies of features, thus successfully capturing an image's spatial dependencies by applying relevant filter operations (LeCun and Bengio, 1995). That means that neurons from one layer are not connected to all neurons in the next layer but only to neurons within a local spatial patch, often called a receptive field. After the astonishing results of AlexNet (Krizhevsky et al., 2012) in the computer vision community, other fields of science started investigating CNNs for their needs, like physics (Caldeira et al., 2019; Mehta et al., 2019), medical field (Gulshan et al., 2016; Esteva et al., 2017), radiology (Yamashita et al., 2018), and many others. CNNs have therefore gained attention within the geophysical community too.

Three kinds of essential layers make up a CNN; convolution layers that compute the mathematical convolution operation of the input and a bank of filters, a re-sizing process that maintains locality and spatial structure, and fully connected layers that map the extracted
features into the final output. The first two, convolution and re-sizing layers, perform feature extraction and down – or – up-sampling, while the last, fully connected layer, is generally used for classification tasks. As described above, an image is a matrix of pixel values. A small sliding window called a kernel is applied at each image position. That kernel represents a grid of optimizable feature extractors parameters, which make CNNs highly efficient for image processing. As the output of one layer becomes the input for the following layer, the feature extraction process can become progressively more and more complex. The kernel parameters are updated during the training, where like in any other NNs, a CNN aims to minimize a cost function by an optimization algorithm adjusting these parameters.

The following subsections explain the basic building blocks of a CNN, and more specifically, those building blocks used in the thesis.

### 3.2.1 Convolution layer

Let a CNN consists of $L$ layers of linear convolutions and nonlinear transforms. The computational transition from an arbitrary layer $l \in \{0, 1, 2, ..., L\}$ to layer $l + 1$ within the network consists of computing a set of convolution operations on the values $a^l$ to give the linear output $z^{l+1}$, where $a^l$ and $z^{l+1}$ can be considered as 3D objects consisting of 2D features. The nonlinear output is then given by nonlinear transforms on $z^{l+1}$ to give the values $a^{l+1}$. Figure 3-3 is a simple illustration of the convolution operation where the input has two features.

Let $m$ denote the number of time samples, $n$ the number of traces in the target, and $c_l$ the number of features in layer $l$. A set of filters characterizes the CNN $\{w_k^{l+1} \in \mathbb{R}^{f \times f \times c_l|_{k=1}}^{c_{l+1}}\}$ and biases $\{b_k^{l+1} \in \mathbb{R}^{c_{l+1}}\}$ for each transition from one layer to the next. The dimension of the filters $f \times f$ denotes the size of the 2D convolution kernels, and $k$ is the feature number in the output layer $l + 1$. The linear output from the convolution operation of the feature $k$ can be written as:

$$z_k^{l+1} = w_k^{l+1} * a^l + b_k^{l+1},$$

where $*$ denotes convolution operation.
\[(w_{k}^{l+1} * a^{l})_{x,y} = \sum_{i=1}^{c_{l}} \sum_{u=-\lfloor f/2 \rfloor}^{\lfloor f/2 \rfloor} \sum_{v=-\lfloor f/2 \rfloor}^{\lfloor f/2 \rfloor} w_{u,v,i,k} a_{x-u,y-v,i} \]  

(3-21)

where the pair of indexes \((x, y)\) define each pixel position and the pair \((u, v)\) define the kernel position. We can summarize the convolution as a process where we take a small matrix of numbers (called kernel or filter), and we pass it over our image and transform it based on the kernel values and add bias to it. If \(f\) is odd, then \(f/2\) could be rounded down. The convolutional operation can perform different tasks depending on two additional parameters, stride and padding. The stride parameter \(s\), defines the jump size with which the convolutional kernel slides over the image. The padding parameter \(p\) defines how much padding is added to the input features. These two parameters define the specific dimensions of the output \(z_{k}^{l+1}\) and the specific way of determining the pairs \((x, y)\) and \((u, v)\). Figure 3-4 to Figure 3-6 illustrate three among all the possible configurations (Dumoulin and Visin, 2016). The common theme between these figures is that the 2D output \(z_{k}^{l+1}\) is in green, the convolutional 2D kernel \(w_{k}^{l+1}\) is in gray, the 2D input \(a^{l}\) is in blue, and the highlighted dashed area represents the padding. Figure 3-4 shows an operation of convolving a 3 × 3 kernel over a 5 × 5 input using half padding and unit strides, where the input and the output size of the 2D image remain unchanged. Figure 3-5 shows an operation of convolving a 3 × 3 kernel over a 5 × 5 input using 2 × 2 strides, where the input and the output size of the 2D image differ. The output size is down-sampled. The last of the
three figures (Figure 3-6) shows an operation of transposed convolution, which is the equivalent of convolving a $3 \times 3$ kernel over a $2 \times 2$ input (with 1 zero inserted between inputs) padded with a $2 \times 2$ border of zeros using unit strides. The output size of the 2D image, in this case, is up-sampled.

Figure 3-4. Plain convolution example. Convolving a $3 \times 3$ kernel over a $5 \times 5$ input using half padding and unit strides (after Dumoulin and Visin, 2016).

Figure 3-5. Down-sampling convolution example. Convolving a $3 \times 3$ kernel over a $5 \times 5$ input using $2 \times 2$ strides (after Dumoulin and Visin, 2016).

Figure 3-6. Up-sampling convolution example. The transpose of convolving a $3 \times 3$ kernel over a $5 \times 5$ input using $2 \times 2$ strides. It is equivalent to convolving a $3 \times 3$ kernel over a $2 \times 2$ input (with 1 zero inserted between inputs) padded with a $2 \times 2$ border of zeros using unit strides (after Dumoulin and Visin, 2016).

As shown in the last three examples (Dumoulin and Visin, 2016), the convolutional layer can be used to perform different operations; convolution without altering the size of the
input image, convolution that performs down-sampling, and convolution that performs up-sampling also known as transposed convolution.

### 3.2.2 Pooling layer

The pooling layer is another way of down-sampling that progressively reduces the spatial size of the features to reduce the number of trainable parameters. In contrast to the convolution layer that can also reduce the spatial size, the pooling layer has no trainable parameters. Instead, it is a fixed operator. Filter size, stride, and padding are hyperparameters in pooling operations, similar to convolution.

Down-sampling increases the receptive field of the subsequent convolutional kernels. This increase is because the network converts a high-resolution image to a low-resolution. It is achieved either through striding convolutional operations or through fixed pooling operations. This process helps the subsequent convolutional kernels “see” a larger context.

Down-sampling can help the network understand the main data features in seismic data. It acts as a compression, allowing the network to work on smaller data sizes. However, pooling particularly does come at the cost of losing some high-frequency fine-scale details.

Generally, pooling operations do not reduce the number of features of the convolutional layers because pooling is performed separately at each feature. The most common form of pooling operation is max pooling. In max pooling, the spatial dimensions are reduced by replacing a small patch from the input feature with a single value, the maximum value within that patch. Figure 3-7 (Mehta et al., 2019) illustrates a max pooling operation with a $2 \times 2$ filter and a $2 \times 2$ stride. The number of outputs is halved along each dimension. Other pooling operations exist, including average and min pooling.
3.2.3 Residual layer

A popular approach in deep neural networks is to introduce layers with residual mapping, where activations from a layer are skipped over and added or alternatively concatenated to a later layer. The main advantage of introducing residual layers is that it allows for training deeper networks because optimizing the residual is easier than optimizing the target by a plain stack of nonlinear layers (He et al., 2016). Let us focus on a small part of a given CNN, as shown in Figure 3-8, where we have our input feature \( a^l \) on the top. If we assume that the desired underlined mapping we want to obtain by learning is \( f(a^l) \), then in Figure 3-8a, the dashed-area needs to learn that mapping directly. In contrast, the dashed-area in Figure 3-8b needs to learn the residual mapping \( f(a^l) - a^l \). Figure 3-8b illustrates the residual block, where the solid line skipping over the dashed-area carries the layer input \( a^l \) to the addition operator, which represents the shortcut connection. In such a way, input features can forward propagate faster through the residual or skip connections across layers. It also has the effect of potentially saving/maintaining high frequency/detailed information that typically can be lost in pooling layers.

3.2.4 Creating a convolutional neural network architecture

Creating a network architecture using some open-source libraries is a relatively straightforward process. However, adjusting the network to the dataset and the specific task is challenging. This thesis used TensorFlow (Abadi et al., 2016) and PyTorch (Paszke et al., 2017) libraries to build our architectures. Numerous network designs were tested during this thesis, but they can be narrowed down to three structures:
No-downsampling CNN

As discussed, a CNN usually employs some down-sampling at specific layers to increase the receptive field of the subsequent convolutional kernels. Paper II uses an architecture that does not involve down-sampling operators. Instead, the network is trained in the wavelet domain (Mallat, 2009), which was demonstrated to increase the learning algorithm's performance due to the simplification of the image structure (Bae et al., 2017; Guo et al., 2017; Liu et al., 2018). The architecture we use consists of 11 layers (to avoid duplications of figures, please refer to Paper II for visualization of the network itself). The CNN comprises customized convolutional layers, some of which with residual layers. To go from the under-sampled space to the original 2D space, we employ a periodic up-sampling operation within the network referred to as spatial periodic resampling (SPR).

U-Net

In Chapters 4 and 5, we use a customized version of the U-Net (Ronneberger et al., 2015) architecture. We keep the main features of the original network like contracting path (encoder) consisting of convolutional layers and max pooling operations. After reaching the latent layer, an expansive path (decoder) follows, consisting of transposed convolutions for up-sampling and convolutional layers with residual layers. However, we modify the number of features and convolutional kernel sizes. These modifications showed no degradations in terms of quality but sped up the training process significantly. We also adjust the original architecture to accommodate for 2D and 3D implementations.

Overcomplete convolutional encoder-decoder

The majority of the architectures widely used in deep learning and seismics use a sort of encoder-decoder architecture. The encoding part tries to extract an abstract representation of the input data at a latent space at a lower-dimensional Hilbert space. According to Goodfellow et al. (2016), this latent space is called undercomplete. The decoder tries to learn how to take the latent low-dimensional representation back to a high-dimensional output. The latter is usually achieved through convolutional operations. As a consequence of the undercompleteness, such encoder-decoders may lose valuable information. Some
architectures include residual layers, like the U-Net, to overcome such limitations. Instead of U-Net, in Paper III, we employ an overcomplete convolutional encoder-decoder architecture.

The overcompleteness of this network arises from the fact that it maps a given input to a higher-dimensional latent Hilbert space. The architecture in Paper III uses a combination of striding, transposed, and regular convolutions for down-sampling, up-sampling, and plain convolutional layers, respectively. For every down-sampling layer, we down-sample by a factor of \( s^3 \), for a given 3D stride of size \( s \times s \times s \). That requires an increase of least \( s^3 \) in the number of features to ensure overcompleteness.

3.3 Deep learning in seismic

This section elaborates on some of the main issues of DNN methods in seisms and briefly reviews DNN seismic applications for noise attenuation and interpolation.

3.3.1 Common problems of deep learning in seismic

Because marine seismic data can be represented as images, i.e., a matrix, DNN-based methods can be adopted for seismic processing tasks. However, there are significant differences between conventional and seismic images. For instance, the dynamic range value for conventional images ranges from 0 – 255. In contrast, seismic data dynamic range varies significantly and is not limited only to positive values, but negative ones exist. Seismic data have a color depth of 1, i.e., greyscale, while conventional images have a color depth of 3 (RGB image). The frequency content of the two differs too. Seismic data have a much narrower spectrum compared to conventional images. Because of the existing differences between conventional and seismic images, the direct employment DNN methods tailored for conventional images will not work optimally for seismic data. Therefore, as part of the data preparation, the seismic data are usually amplitude normalized and (sometimes) compensated for spherical divergence.

Training a DNN model (supervised) requires available ground truth or labeled data. Hand-labeled data are openly available in the computer vision community. In contrast, obtaining ground truth data in seisms is often expensive, if not impossible. Seismic data are costly, sparse, and usually noisy. One way of obtaining labeled data is to use conventionally processed data and train the DNN model to replicate the process. For example, the noise-
contaminated data as input to the network and its label as conventionally attenuated data. Additionally, modeling can provide labeled data. However, modeling can implicitly include modeling assumptions in the data that differ from real data, which may be a problem. It is not uncommon to combine real and modeled data for better generalization of the trained model.

### 3.3.2 Overview of deep learning applications in seismics

DNN has drawn the attention of the geophysical community and has proven helpful for a vast number of applications in geoscience. Most DNN research in seismics corresponds to using already existing network tools rather than going into their fundamentals. During the course of this Ph.D. project, a significant amount of articles on DNN-based seismic noise attenuation and interpolation have been published.

Many DNN publications focus on random noise suppression within seismic noise attenuation, which is a significantly easier task than the problematic coherent noise and is not relevant to this thesis. In addition, many other topics were studied. Yu et al. (2019) showed the potential application of a CNN-based approach for different types of noise, like random, linear, and multiple. Ground-roll noise attenuation has also demonstrated the feasibility of CNNs for such tasks (Li et al., 2019; Kaur et al., 2020). Klochikhina et al. (2020) used a CNN-driven approach for migration swing suppression. De Jonge et al. (2021) addressed source bubble noise attenuation and showed promising results using a well generalized CNN. Applications towards swell noise attenuation (Zhao et al., 2019; Zhang et al., 2020) proved that CNNs could perform equally well for this task. Seismic interference studies (Sun et al., 2020; Xu et al., 2020) showed that a CNN could address this noise and perform attenuation in real-time at the acquisition phase. Attenuation of blended energy was also shown to be possible employing CNN (Richardson and Feller, 2019; Zu et al., 2020; Wang et al., 2021). Multiple attenuation techniques by training a CNN on a subset of conventional processing were proposed by Siahkoohi et al. (2019a). Similarly, Peng et al. (2021) used CNN for de-ghosting purposes by replicating an existing algorithm. Qu et al. (2021) developed a complete multiple attenuation workflow based on CNN trained on purely synthetic data.

CNNs have also served as an alternative way of seismic data interpolation. The ML-based on support vector regression (Jia and Ma, 2017) trains a regression function that is used to guide the interpolation. Mandelli et al. (2018) explored the capabilities of U-Net-based trace interpolation for irregularly sampled shot gathers. Similar U-Net architecture in
combination with texture loss was also proposed (Fang et al., 2021), claiming to capture the characteristics of the seismic data. Wang et al. (2019a) used a residual network with a bicubic pre-interpolator for seismic data interpolation. The concept of transfer learning using synthetic data to initialize real shot training (Wang et al., 2020) addressed the interpolation problem using a convolutional auto-encoder. A conditional generative adversarial network for the interpolation of post-stack data was investigated by Oliveira et al. (2018). Chai et al. (2020) used a combination of training on both synthetic and real data examples for CNN-based applications. Other studies investigated similarities between different domains and data orientation for CNN-driven interpolation (Siahkoohi et al., 2018; Park et al., 2019; Wang et al., 2019b; Paper II). Unsupervised CNN-based interpolation techniques for 2D, 3D, and 4D data have also shown that they can approximate the interpolation problem (Hu et al., 2019; Kong et al., 2020; Shi et al., 2020; Greiner et al., 2021).

The methods mentioned above for noise attenuation and interpolation purposes are just a small overview of the ones investigated in the seismic community. However, we can see many publications just within the last few years. Almost all of these publications have one thing in common; they are all CNN-based. CNNs have already proven their potential for seismic processing despite their relative novelty. One might say that autoencoder CNNs are not ideal for image processing (and seismics) as they typically try to represent the data at a compressed latent layer that has gone through several steps of down-sampling. For example, compression is a good way of removing noise from the data, but it also often results in signal loss. Therefore, when a down-sampling is achieved through fixed pooling operators, we usually employ residual layers to compensate for that unwanted signal loss. This approach is widely adopted in the CNNs methods used in geophysics, especially in those cases where the U-Net type of architecture (Ronneberger et al., 2015) is adapted (Mandelli et al., 2018; Park et al., 2019; Sun et al., 2020; Zhang et al., 2020; Fang et al., 2021). Rather than using a fixed down-sampling operator, overcomplete encoder-decoders that employ striding convolution operators learn the down-sampling operator and reduce the risk of signal leakage (Paper III in this thesis).

Another critical aspect, already discussed, is overfitting or biasing the trained model to the train set. Different techniques are employed to overcome that problem, i.e., regularization, dropouts, etc. We need to train using a lot of data, not just a few data examples; typically, thousands of examples to avoid overfitting. This high number of representative
training data examples increases the probability of good generalization of the NN. We can not expect that the NN will work on data that is too different from the one presented during the training. For example, a single “spike” of “blanked area” placed somewhere in the data can also totally “destroy” the results (unless the network has seen similar features before).
4. 2D interpolation of seismic data using Deep Neural Networks

In a marine towed-streamer seismic survey, the resolution of the seismic data are limited by many factors. Among these are the available bandwidth of the seismic equipment, the sampling rate, and the acquisition layout. Once the Nyquist sampling criterion is violated - aliasing occurs, which causes signal ambiguities. Figure 2-3 showed that the distance between the streamers (along the spatial crossline dimension) usually is significantly larger than between the channels (along the spatial inline dimension). That streamer distance can hardly go below 50 m because of operational reasons related to risk of tangling the seismic equipment. However, reducing it or increasing the number of sources, will eventually result in a smaller bin size because the bin size along the spatial crossline dimension is \( \Delta y = \frac{DC}{2NS} \), where \( DC \) is the streamer separation, and \( NS \) is the number of seismic sources. The orthogonal inline spatial sampling depends on the receiver spacing, which is usually fixed. However, the shot point interval is almost always larger than the corresponding group interval between each channel unit for two reasons; (i) to allow the compressors to fill the air-gun sources and (ii) to avoid blended energy contaminating the target zones. Because of this sparser shot point sampling, the performance of some of the main steps in seismic processing (e.g., blending noise attenuation, velocity model building, multiple attenuation, and migration) is suboptimal. The denser the shot point interval, the smaller the offset class size, thus increasing the nominal fold.

This chapter deals with the interpolation of 2D seismic gathers extracted from the 3D marine shot gathers. The focus relates to the interpolation of regularly distributed traces between the existing ones using CNNs. Since possible aliasing along the spatial inline or crossline dimension is delimited by the trace spacing, seismic data may show a different aliasing pattern depending on the data domain. For example, the common shot gather (CSG) and the common receiver gather (CRG) will have different aliasing patterns because they usually are sampled differently. To make this clearer, Figure 4-1 shows a sketch of the four commonly used domains for gather building for a single source (red asterisks) and streamer combination, where the shot point interval is 37.5 m, while the receiver (red triangles) interval is 12.5 m. Despite different trace intervals, once appropriately organized, we can use one observation domain to train a DNN model and utilize that trained model in another domain.
for interpolation purposes. More specifically, we investigate the potential use of DNNs by taking advantage of different data domains to densify; (i) cross-streamer trace increment and (ii) shot point increment. (i) is addressed in Paper II, while (ii) is included as a subsection of this chapter. The general idea behind this chapter is to find a suitable way of using the data to train a DNN rather than replicating an existing conventional interpolation algorithm.

Paper II explores the concept of supervised learning on regularly decimated CSG, intending to use that trained network in the orthogonal dimension and interpolate traces between the existing streamers. In other words, it uses the densely sampled inline dimension to train a CNN, which is utilized in the normally much coarser crossline dimension. Paper II demonstrates that this approach is feasible through synthetic and field data examples because these two domains share the same data features. We use the recorded CSG as labeled data, i.e., our densely-sampled data, to train the CNN for the field data example. We then decimate the CSG with three of four traces removed to create an under-sampled CSG. The CNN then uses these two datasets to learn the up-sampling function. Transforming the under- and densely-sampled examples into a feature space before learning, such as the wavelet domain (Mallat, 2009), increases the learning algorithm's performance due to the sparsification of the image structure (Bae et al., 2017; Guo et al., 2017; Liu et al., 2018). Therefore, in Paper II the CNN is trained in the wavelet domain. The synthetic example in Paper II indicates that the CNN can be trained in the inline dimension and applied in the crossline one while preserving the geologic model in the migrated section.

Figure 4-1. Schematic illustrating the available domains for gather building of a mono source-streamer configuration. The consecutive shot points are plotted one below each other for visualization purposes, starting from the top.
4.1 Paper II – Cross-streamer wavefield reconstruction through wavelet domain learning


Geophysics, 85, no. 6, V457-V471 (2020)
Cross-streamer wavefield reconstruction through wavelet domain learning

Thomas Andre Larsen Greiner†, Volodya Hlebnikov‡, Jan Erik Lie§, Odd Kolbjørnsen¶, Andreas Kjelsrud Evensen‖, Espen Harris Nilsen¶, Vetle Vinje‖, and Leiv-J Gelius§

ABSTRACT

Seismic exploration in complex geologic settings and shallow geologic targets has led to a demand for higher spatial and temporal resolution in the final migrated image. Conventional marine seismic and wide-azimuth data acquisition lack near-offset coverage, which limits imaging in these settings. A new marine source-over-cable survey, with split-spread configuration, known as TopSeis, was introduced in 2017 to address the shallow-target problem. However, wavefield reconstruction in the near offsets is challenging in the shallow part of the seismic record due to the high temporal frequencies and coarse sampling that leads to severe spatial aliasing. We have investigated deep learning as a tool for the reconstruction problem, beyond spatial aliasing. Our method is based on a convolutional neural network (CNN) approach trained in the wavelet domain that is used to reconstruct the wavefield across the streamers. We determine the performance of the proposed method on broadband synthetic data and TopSeis field data from the Barents Sea. From our synthetic example, we find that the CNN can be learned in the inline direction and applied in the crossline direction, and that the approach preserves the characteristics of the geologic model in the migrated section. In addition, we compare our method to an industry-standard Fourier-based interpolation method, in which the CNN approach shows an improvement in the root-mean-square (rms) error close to a factor of two. In our field data example, we find that the approach reconstructs the wavefield across the streamers in the shot domain, and it displays promising characteristics of a reconstructed 3D wavefield.

INTRODUCTION

Seismic processing and imaging of the subsurface in the Barents Sea is challenging for several reasons. The large uplift and erosion of the area manifest in rocks with high seismic velocities at shallow depth. These large velocity contrasts set up complex multiples and allow only a narrow cone of the reflected energy from the seismic sources to penetrate and illuminate the subsurface targets (Lie et al., 2018). The conventional 3D seismic spread lacks the near offsets, which are important for imaging shallow parts of the subsurface and are of great benefit for multiple attenuation (Vinje et al., 2017). Therefore, a conventional 3D acquisition does not represent an optimal setup in these environments. To address the shallow-target seismic-imaging issue, CGG and Lundin Norway proposed in 2017 a tailored acquisition solution, known as TopSeis (Vinje et al., 2017), yielding improved recording of the near offsets. This acquisition solution uses a split-spread, source-over-cable configuration, reduced streamer separation, wider source separation, and the deployment of two or more sources. Still, near offsets are sparse, and the data suffer from spatial aliasing due to high temporal frequen-
cies and coarse sampling across the streamers. The interpolation and reconstruction problem of aliased data is considered an underdetermined system because there are infinite solutions to which the aliased seismic data could fit the dense. Hence, a priori information about the wavefield, such as seismic velocities, or assumptions concerning local linearity, sparsity, or matrix rank are needed to reconstruct the wavefield. However, in the presence of severe aliasing, these assumptions may not hold. In this paper, we investigate the potential use of deep learning as a tool to reconstruct the wavefield across the streamers in the shallow part of the seismic record where the temporal frequencies are high and the spatial sampling is coarse.

Wave-equation-based methods (Fomel, 2003) may deal with irregular and regular sampling in the presence of aliasing but are computationally heavy and needs a velocity model. Some techniques assume local linearity and interpolate data in the frequency-space domain (Spitz, 1991; Porsani, 1999; Crawley, 2000; Naghizadeh and Sacchi, 2008). Some methods may reconstruct in the presence of a mild degree of spatial aliasing under the assumption that the interpolation problem becomes a matrix-completion (rank-reduction) problem (Trickett et al., 2010), minimum weighted norm inversion of the subsampling operator (Liu and Sacchi, 2004), or least-squares fitting of sinousoids (Ghaderpour et al., 2018; Ghaderpour, 2019). In the case of high temporal frequencies and coarse sampling in the near offsets, in which seismic events display conflicting dips and highly curved events, the linear assumption breaks down and these methods are therefore not optimal. Some methods handle the interpolation problem by means of predefined transforms, such as Fourier (Xu et al., 2005, 2010; Zwartjes and Sacchi, 2006; Schonewille et al., 2009; Naghizadeh and Sacchi, 2010a), Radon (Ibrahim et al., 2015), curvelet (Naghizadeh and Sacchi, 2010b), seislet (Gan et al., 2015), and focal (Kutscha et al., 2010). The interpolation problem is then solved in combination with sparse optimization algorithms such as matching pursuit (Mallat and Zhang, 1993), basis pursuit (Boyd and Vandenberghe, 2004), or projection onto convex sets (Abma and Kabir, 2006). However, it is challenging to find a single transform that sparsifies all events such as diffractions and reflections shallow and deep, and some events are not optimally sparse (Turquis et al., 2018). Rather than relying on a predefined sparse transform, in some sparse approximation methods, such as dictionary learning, we assume that the seismic data are a sparse linear combination of atoms defined from an overcomplete dictionary (Turquis et al., 2017, 2018; Zhu et al., 2017). Alternatively, complementary information from multicomponent data in combination with sparse optimization (Robertson et al., 2008; Özdemir et al., 2010; Vassallo et al., 2010) makes the crossline reconstruction beyond aliasing possible. An alternative path to the seismic interpolation problem is the use of data-driven approaches such as machine learning (Jia and Ma, 2017) and deep learning (Wang et al., 2018; Mandelli et al., 2019), which have drawn much attention recently. Deep learning based on CNN can be viewed as a special case of the traditional sparse approximation methods, but instead of optimizing each component separately, all components are jointly optimized (Dong et al., 2015), which allows for an efficient approach to the reconstruction problem.

Our approach to the seismic wavefield reconstruction problem may be seen as an analogy to inverse problems in image scaling in which low-resolution digital images are transformed into their corresponding high-resolution counterparts. These sets of techniques are commonly referred to as single-image superresolution (SISR), or image restoration, and date back to the mid-1980s and work of Tsai and Huang (1984). Popular SISR imaging methods can be categorized into four main types (Yang et al., 2014): prediction models (Irani and Peleg, 1991), edge-based methods (Sun et al., 2010), statistical methods (He and Sun, 2011; Efrat et al., 2013), and example/patch/learning-based methods (Chang et al., 2004). In recent years, the deep-learning-based methods have shown an increased popularity in image reconstruction problems because of their improved efficiency and state-of-the-art performance compared to the aforementioned methods (Yang et al., 2019). Deep-learning approaches are specifically designed to automatically learn an end-to-end mapping between the low- and high-resolution examples (Dong et al., 2015), in which the nonlinear representations are learned from large image databases.

In learning-based SISR, the low-resolution examples are commonly modeled from their corresponding high-resolution examples through blurring and downsampling followed by upsampling to the same size using a conventional interpolation method, such as bicubic interpolation (Dong et al., 2015; Guo et al., 2017). The SISR model is then applied to remove the blurring effect. To reduce the computational complexity and to accelerate learning and application, the suboptimal interpolation prior to learning can be skipped and the upsampling operation is moved to within or toward the end of the network (Dong et al., 2016; Shi et al., 2016). In addition, methods such as bicubic interpolation do not bring additional information to the reconstruction problem (Shi et al., 2016). However, if the image contains complex structures and patterns, state-of-the-art CNNs applied directly in the image domain are still inferior to other traditional methods (Bae et al., 2017). Transforming the low- and high-resolution examples into a feature space prior to learning, such as the wavelet domain, increases the performance of the learning algorithm due to the simplification of the image structure (Bae et al., 2017; Guo et al., 2017; Liu et al., 2018).

The proposed wavefield reconstruction approach, which is inspired from two deep-learning CNN-based techniques (Shi et al., 2016; Guo et al., 2017), considers the low- (subsampled) and high-resolution (target) seismic gather as training examples and transforms them into the wavelet domain where we learn the upsampling function. The key advantages of this method are (1) reduced computational complexity due to smaller size of the input in combination with time-space compression from the wavelet transform, (2) the wavelet domain simplifies the structure of the seismic gather, which could potentially increase performance in the presence of complex structural patterns, and (3) it is not dependent on any prior interpolation method to learn from. Our method is only limited by the available bandwidth in the domain from which we train the upsampling function. In addition, we hypothesize that the nonlinear relationships between the subsampled wavefields and their corresponding target wavefields can be learned along one direction — where the spacing between sensors is adequately dense — and then we use the trained function to reconstruct the wavefield in a coarser direction (Greiner et al., 2019). In the synthetic data example, we use offset gathers in the inline direction representing the dense cases and we attempt to reconstruct the much coarser crossline direction. In the field data example, we use marine shot gathers with split-spread geometry, illustrated in Figure 1, in the inline direction to reconstruct across the streamers where the sampling is much coarser.

This paper is organized as follows. First, we introduce learning-based SISR and discuss how SISR can be used within seismic wave-
LEARNING-BASED WAVEFIELD RECONSTRUCTION

Learning-based superresolution

The resolution limitation in seismic data is affected by the available bandwidth, the geometry, and density of the sampling array. If a subset of the data is not sampled according to the Nyquist sampling criterion, aliasing will take place, as in the case for the undersampled crossline data. To model the crossline resolution problem, consider the underdetermined system of linear equations

\[
DY = X, \tag{1}
\]

where \( Y \in \mathbb{R}^{M \times N} \) denotes the target, represented by a seismic gather of size \( M \times N \), where \( M \) is the number of time samples and \( N \) is the number of traces. The subsampled counterpart \( X \in \mathbb{R}^{rM/2} \) is decimated by the known operator \( D: \mathbb{R}^{M \times N} \rightarrow \mathbb{R}^{M/2} \), where \( r \) is the subsampling factor. To illustrate the action of the operator \( D \), let \( Y \in \mathbb{R}^{M \times N} \) represent a 2D seismic signal with four traces \([y_{10}, y_{11}, y_{20}, y_{21}]\) and three time samples \(i = 0, 1, 2\) downsampled by a ratio of \( r = 2 \). The decimation operation in equation 1, with \( Y \) represented in vector form, becomes

\[
\begin{pmatrix}
y_{00} \\
y_{01} \\
y_{02} \\
y_{10} \\
y_{11} \\
y_{12} \\
y_{20} \\
y_{21} \\
y_{22} \\
y_{30} \\
y_{31} \\
y_{32}
\end{pmatrix} =
\begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}
\begin{pmatrix}
x_{00} \\
x_{01} \\
x_{02} \\
x_{10} \\
x_{11} \\
x_{12} \\
x_{20} \\
x_{21} \\
x_{22} \\
x_{30} \\
x_{31} \\
x_{32}
\end{pmatrix}. \tag{2}
\]

It is obvious in equation 2 that the operator \( D \) is singular and has no inverse. Because seismic signals are highly structured and smoothly varying, we might still be able to estimate an inverse valid for this subspace. We try to establish this relation between the subsampled and the target seismic gathers by the functional relationship

\[
Y = f(X) + \varepsilon, \tag{3}
\]

where \( \varepsilon \) represents the noise. In our case, we seek an approximate function \( f(X): X \in \mathbb{R}^{M/2} \rightarrow Y \in \mathbb{R}^{M \times N} \) where \( Y \) is the predicted seismic output from the learned function \( \hat{f}(\cdot) \). Figure 2 shows an example of a 2D shot gather in the inline direction, which is extracted from a 3D marine split-spread shot gather and cropped in time and space.

In the case of shot-domain reconstruction across the cables, the 2D split-spread gathers along with the subsampled counterpart (Figure 2) will then define the training data \( \{(Y^{(i)}, X^{(i)})\}_{i=1}^{K} \) for \( K \) examples, from which we propose to learn the function \( f(\cdot) \). Our assumption is that the decimated wavefield in the inline direction is representative for the wavefield in the crossline direction. We expect this to be a reasonable assumption, especially considering split-spread geometry (Figure 1), which yield near-offset data giving similar inline-crossline characteristics, as observed in Figure 2b and 2d.

Using wavelet domain learning

Following Bae et al. (2017), Guo et al. (2017), and Liu et al. (2018), we propose to learn the function \( f(\cdot) \) in the wavelet domain. In the 2D discrete wavelet transform (DWT), digital filters are introduced that are characterized by their scaling function \( \phi \) and wavelet basis \( \psi \). These are used to convolve a 2D signal \( Y \) into subband information represented by wavelet coefficients (Mallat, 1989). The simplest filters are the Haar filters, which could be represented by four 2D convolution kernels as

\[
g_{A} = \frac{1}{2} \begin{bmatrix} 1 & 1 \\ 1 & 1 \\ \end{bmatrix}, \quad g_{H} = \frac{1}{2} \begin{bmatrix} 1 & -1 \\ -1 & 1 \\ \end{bmatrix},
\]

\[
g_{V} = \frac{1}{2} \begin{bmatrix} 1 & -1 \\ 1 & -1 \\ \end{bmatrix}, \quad g_{D} = \frac{1}{2} \begin{bmatrix} 1 & 1 \\ -1 & -1 \\ \end{bmatrix}. \tag{4}
\]

In a wavelet decomposition of level 1, the target 2D signal \( Y \in \mathbb{R}^{M \times N} \) is convolved with \( g_{A}, g_{H}, g_{V}, \) and \( g_{D} \) yielding the four-band representations \( d^{A}, d^{H}, d^{V}, d^{D} \in \mathbb{R}^{M \times N} \), respectively, where \( m = M/2 \) and \( n = N/2 \). The four-band representations are schematically illustrated in Figure 3a with their respective convolutional kernels. In equation 4, we see that the filter \( g_{A} \) acts as a low-pass filter on \( Y \), and it is therefore commonly referred to as the approximation, whereas \( d^{H}, d^{V}, \) and \( d^{D} \) represent the horizontal, vertical, and diagonal detail, respectively. Figure 3b shows an example of a four-band split of a seismic source gather. We will use the following compact notation for the forward wavelet transform and inverse wavelet transform:

\[
\tilde{Y} = \psi_{H}Y \quad \text{and} \quad \tilde{X} = \psi_{H}X. \tag{5}
\]

Figure 1. An illustration of a split-spread design using a streamer vessel and a source vessel with three sources.
where the subscript \( H \) denotes the Haar filter, the superscript * denotes the adjoint, and \( \mathbf{Y} \in \mathbb{R}^{m \times n \times k} \) and \( \mathbf{X} \in \mathbb{R}^{m \times n \times k} \) denote, respectively, the decomposition of the target gather and the subsampled gather. In this case, \( \mathbf{X} \) denotes the input to the CNN and \( \mathbf{Y} \) denotes the target; we want to predict from the learned function \( f'(\cdot) \). Wavelet decomposition and reconstruction were performed using PyWavelet (Lee et al., 2019).

Convolutional neural network architecture

Convolutional neural networks (CNNs) are a special case of artificial neural network architectures that incorporate knowledge about the invariance of the object shapes by using local connection patterns and by imposing constraints on the learnable parameters (LeCun et al., 1998). A CNN consists of \( L \) layers of linear convolutions and nonlinear transforms. The computational transition from an arbitrary layer \( l \in \{0, 1, 2, \ldots, L\} \) to layer \( l + 1 \) within the network consists of computing a set of convolution operations on the values \( a' \) to give the linear output \( z^{l+1} \), where \( a' \) and \( z^{l+1} \) can be considered as 3D objects consisting of 2D features. The nonlinear output is then given by nonlinear transforms on \( z^{l+1} \) to give the values \( a^{'l+1} \). A simple illustration of the convolution operation is given in Figure 4 where the input has two features. We will now discuss our network architecture, which is schematically represented in Figure 5. There are four types of transforms in our design:

1) Standard convolutional layers. These layers do not alter the dimension of the 2D features due to zero padding, but might change the number of features.
2) Periodic upsampling to increase the 2D features size. This resampling operator upsamples the 2D feature space and reduces the number of features.
3) Nonlinear transforms, that is, activation functions.
4) Residual layers with single skip connections (He et al., 2016).

Let \( m \) denote the number of time samples, \( n \) the number of traces in the target, \( r \) the decimation factor, and \( c_i \) the number of features in layer \( l \). The CNN is characterized by a set of filters \( \{ w_{i+1}^{l+1} \in \mathbb{R}^{n \times m \times f} \}^{m+1}_{i=0} \) and biases \( \{ b_{i+1}^{l+1} \in \mathbb{R} \}^{m+1}_{i=0} \) for each transition from a layer to the next. The dimension of the filters \( f \times f \) denotes the size of the 2D convolution kernels, and \( k \) is the feature number in the output, that is, layer \( l + 1 \). The linear output from the convolution operation of feature \( k \) can be written as

\[
z^{l+1}_k = w^{l+1}_k \ast a^l + b^{l+1}_k,
\]

where \( \ast \) denotes the convolution operation

\[
(w^{l+1}_k \ast a^l)_{x,y} = \sum_{u=-f/2}^{f/2} \sum_{v=-f/2}^{f/2} w^{l+1}_{k,u,v} a^l_{x-u,y-v}.
\]

Figure 3. A simplified illustration of the wavelet decomposition, where (a) is a schematic representation of the four-band split with their respective convolution kernels (gray = 1/2, white = -1/2) and (b) an example where a seismic source gather is decomposed using the Haar filter.

Figure 2. An illustration of inline training data and the crossline reconstruction problem. (a and b) A single training example in the shot domain where (a) the target in the inline direction is subsampled to give (b) the crossline representation. The inline targets and the subsampled counterparts in the training data are then used to learn the function \( f'(\cdot) \) to reconstruct (c) the crossline target from (d) the undersampled crosslines using the learned function \( f(\cdot) \).

\[
Y = \psi_{H} X \quad \text{and} \quad X = \psi_{H}^\ast Y.
\]
where the pair of indexes \((x, y)\) defines each pixel position and the pair \((u, v)\) define the kernel position. We can think of the convolution equation 8 as a special case of convolution, where the \(k\)th output is computed by a sum of 2D convolutions, which implies using a convolution kernel on each input feature superimposed with the bias in each pixel position. Thus, for a given layer there are as many biases as there are features in the next layer, that is, \(c_{l+1}\). The number of weights is dependent on the kernel size and the number of input and output features. To simplify the notations, let \(W^{l+1} \in \mathbb{R}^{d_l \times d_l \times c_{l+1}}\) and \(b^{l+1} \in \mathbb{R}^{c_{l+1}}\) denote all of the weights and biases in layer \(l + 1\), respectively. As seen in Figure 5, the first-layer weights have the dimensionality \(7 \times 7 \times 4 \times 112\), where \(7 \times 7\) denotes the kernel size, 4 denotes the number of input features, that is, \(d^0\), \(d^1\), \(d^2\), \(d^3\), and \(d^4\), and 112 are the number of output features.

To introduce nonlinearity, \(a^{l+1} = [z^{l+1}_1, \ldots, z^{l+1}_{c_{l+1}}]\) is passed through an activation function

\[
a^{l+1} = \phi(a^{l+1}).
\]

The main advantage of introducing residual layers is that it allows for training deeper networks because optimizing the residual is easier than optimizing the target by a plain stack of nonlinear layers (He et al., 2016). In our case, the nonlinear function \(\phi(\cdot)\) is given by the leaky rectified linear unit (Maas et al., 2013) or leaky ReLU, which is defined as

\[
\phi(z) = \max(z, \alpha z),
\]

where \(0 < \alpha < 1\) is the slope for negative arguments. To go from the subsampled space to the original 2D space, we use a periodic up-sampling operation within the network. We will refer to this layer as the spatial periodic resampling (SPR) layer, which in our case can be mathematically defined as

\[
SPR(a)_{x,y,k} = a_{\lfloor x/r \rfloor, y, c \mod(x/r) + k},
\]

where \(x, y, k\) are the pixel coordinates starting at \(x, y, k = 0\), \(\lfloor x/r \rfloor\) is integer division, and \(\text{mod}(x/r)\) is the remainder of this division. From equation 12, we see that \(\text{SPR}(\cdot) : \mathbb{R}^{m \times n} \rightarrow \mathbb{R}^{m \times n}\) is an operator that rearranges the columns of \(a\) to upsample the 2D features by reducing the number of features, which is illustrated in Figure 6 resampling from \(m \times \frac{3}{2} \times 4\) to \(m \times 6 \times 2\).

### Optimization problem

In deep learning, the optimization problem is typically solved by minimizing and/or maximizing an objective function using first-order gradient-based optimization algorithms, that is, gradient descent/ascent, in combination with a learning-based procedure known as backpropagation (Rumelhart et al., 1988). Learning the weights \(W\) and biases \(b\) is then done by minimizing the following objective function:

\[
\min_{w,b} \{L(\hat{Y}, \hat{X}; W, b) + \lambda(R(W))\},
\]

where \(L(\hat{Y}, \hat{X}; W, b)\) is the loss term, \(R(W)\) is the regularization term, and \(\lambda\) is the regularization parameter defining the trade-off between the loss and the regularization. To minimize equation 13 by a
gradient-based method, we need to compute the partial derivatives with respect to each weight and bias within the network. The optimization problem was solved by stochastic gradient descent using subsamples (minibatches) using a version of the Adam optimizer defined in Loshchilov and Hutter (2018) as

$$W_{[t]} = W_{[t-1]} - \eta \nabla_W R(W)_{[t]} + \lambda \nabla_W R(W)_{[t-1]}$$

where \(\eta\) is the fixed learning rate, \(\nabla_W R(W)\) is the partial derivatives of the regularization term, and \(\nabla_W L(Y, \hat{X}, W, b)\) is the partial derivatives of the loss using the Adam optimizer rule (Kingma and Ba, 2014). In our case, the loss in equation 13 is given by the \(L_1\) norm \(L_1(\hat{Y}, \hat{X}, W, b) = ||\hat{Y} - f(\hat{X}, W, b)||\) and the regularization on the weights is given by the \(L_2\) norm penalty \(R(W) = \frac{1}{2} ||W||^2\). The biases are updated in the same manner excluding the regularization term. Neural network implementation and training were performed in Python using the NumPy and TensorFlow libraries (Abadi et al., 2016).

Network design, training, and regularization

The optimization problem in deep learning is challenging due to being nonconvex, overparameterized, and unstable. Careful selection of hyperparameters, parameter initialization, and network design through trial and error is necessary to get the model to train and converge to a proper local minimum. Initialization of the weights was done using He initialization (He et al., 2015), which is particularly designed for handling rectifier nonlinearities, and the biases were initialized to zero values. The hyperparameters of the Adam optimizer were set to the default because we observed no improvements by using other values. The network depth and width (the number of layers and the number of filters in each layer) were found by trial and error, along with the rectifier nonlinearities slope \(\alpha\) and learning rate \(\eta\), which were set to 0.1 and 0.001, respectively.

The computational model and network design used in our study — summarized and schematically illustrated in Figure 5 — consists of three single-skip connection residual layers in subsampled 2D space, an SPR operation after layer eight for upsampling followed by two nonlinear layers and a standard linear output. The first two convolution layers, in addition to the two convolution layers after upsampling, consist of \(7 \times 7\) followed by \(5 \times 5\) kernels. The other convolution layers have \(3 \times 3\) kernels. Similar to Dong et al. (2015), we find that using a larger kernel size in the start of the network improved the reconstruction and structural definition of the prediction. However, because larger kernels increase the computational complexity, the choice of kernel size should be a trade-off between performance and speed.

Prior to training, the training data are split randomly into three data sets, which we will refer to as the training, validation, and test sets. In each example, all of the sets were scaled to have a maximum absolute value of 1. The training set is used to fit the model to the data, the validation set is used for model selection, that is, selection of the regularization parameter \(\lambda\), and the test set is used for model assessment for the final evaluation.

For model selection, we used a cross-validation (CV) approach in combination with an early stopping algorithm. The goal of regularized and early stopping is to avoid overfitting in an overparameterized model. The early stopping approach avoids overfitting by storing the parameters that show the best performance in one training phase and stopping after a predefined number of iterations if the validation loss has not decreased. Regularization constrains the size of the weight parameters during training, which depends on the value of regularization parameter \(\lambda\). CV is performed by training with different \(\lambda\) values, that is, different models, and selecting the model that shows the best performance on the validation set. We used a selection of \(\lambda \in \{10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}, 0\}\), where \(\lambda = 0\) and \(\lambda = 10^{-3}\) denote the highest and lowest model complexity, respectively. We found \(\lambda = 10^{-4}\) to be a good choice in the synthetic example and the field example. An example from the loss during training from the field data example is shown in Figure A-1 in Appendix A.

SYNTHETIC DATA EXAMPLE

To evaluate the performance of our wavelet CNN and inline-learning to crossline-reconstruction approach, we have designed a synthetic 3D model and applied a diffraction modeling method similar to what is described in Jaramillo and Bleistein (1999) to create synthetic broadband data. For comparison, we use a 3D Fourier-based interpolation method (Xu et al., 2005), which is similar to
currently applied approaches in the industry. We will refer to this method as 3D Fourier in the following.

As can be seen from Figure 7, the model consists of a rugose water bottom from bathymetry in the Barents Sea and a 2D grid of equally spaced diffraction lines at a depth of 518 m. The water bottom and diffraction lines in this model will give a complex data set with strong aliasing and conflicting dips, which is a challenge to any interpolation algorithm.

In Figure 8a, we show the survey setup, where we model 64 constant-offset, zero-azimuth data cubes on a $6.25 \times 6.25$ m$^2$ grid with offset spacing of 12.5 m and a time sampling of 2 ms. The source signal is zero phase with a flat frequency spectrum from 4 to 175 Hz. We assume no external noise in this experiment, constant velocity of 1480 m/s and no anelastic attenuation.

In the following, we will refer to the densely sampled $6.25 \times 6.25$ m$^2$ cubes as the ground truth. We then decimate the ground truth to create a new subsampled data set, in which we have removed three of four traces to get a sampling of $6.25 \times 25$ m$^2$ as shown in Figure 8b. An example of the ground truth and the subsampled counterpart used for training in the inline direction, magnified at the water bottom and diffraction lines is displayed in Figure 9. The subsampled data define that the training data $D = \{Y^{(i)}, X^{(i)}\}_{i=1}^{K}$, where $Y$ is an offset class along one inline and $X$ is the subsampled counterpart, consist of 162 inlines, which gives $K = 10,368$ examples for the 64 offset classes. The training data size was doubled by augmentation, by flipping the data to give negative offsets. A subset of $K = 15,900$ examples was then used as the train-

Figure 8. Magnified section of the bin locations, position, and grid size of the modeled constant offset and zero-azimuth data. (a) Ground truth and (b) after decimation with three of four traces removed in the crossline direction.

Figure 9. Zoomed sections of the diffraction modeled data displayed in the inline direction where (a) and (b) displays, respectively, the shallow part of the ground truth and subsampled counterpart, and (c) and (d) displays, respectively, the deep part of the ground truth and subsampled counterpart.

Figure 10. Reconstruction results in the crossline direction from the shallow section of the wavefield, that is, the water bottom. From (a) the ground truth, we observe a complex wavefield setting, which follows that (b) the subsampled crossline direction yields a challenging wavefield to reconstruct. From (c) the 3D Fourier and (e) the wavelet CNN, we observe more residual energy within the (d) 3D Fourier interpolation error than the (f) wavelet CNN reconstruction error.
ing data, which were further split into a training set and a validation set of 14,900 and 1,000, re-

spectively. The crossline section will then re-

present the test set for final assessment. Using 400 time samples, the size of each training exam-

ple is $400 \times 640$ and $400 \times 160$ for the target and subsampled data, respectively. The wavelet CNN is then trained along the inline direction where the sampling is dense and is applied to the coarsely sampled crossline direction. The training phase runs for 58,700 iterations (19 epochs) before early stopping initiated, which took approximately 10 hours on a modern GPU (NVIDIA Tesla V100 SXM2 32 GB). When the training phase is complete, the run time for each crossline section ($600 \times 162$) takes approximately 0.2 s.

From the ground truth in Figures 9a and 10a, we observe that the rugose water bottom creates a complex reflected wavefield with a myriad of diffraction-like events causing strong interference. Deeper down, at approximately 0.7 s in Figure 11a, we see the regular hyperbolic events from the diffraction lines parallel to the inlines and the linear events from the diffraction lines parallel to the crosslines. The decimated data shown in Figures 10b and 11b are strongly aliased, and the 3D Fourier interpolation shown in Figures 10c and 11c struggles to resolve the conflicting dips and the aliasing. The wavelet CNN in Figures 10e and 11e does a much better job in reconstructing conflicting dips and aliased events, with some minor “striping” artifacts. These striping artifacts represent input traces where the wavelet CNN has not managed to learn a suitable pattern for reconstruction. In case of reconstruction of the deeper part, the diffraction

![Wavelet CNN reconstruction](image)

**Figure 12.** This figure shows the migration results from a crossline section and a time slice through 450 ms, where (a and b) represents the migration result of the ground truth, (c and d) represents the migration result of the 3D Fourier interpolation, and (e and f) represents the migration result of the wavelet CNN reconstruction.
lines are equal in the inline and crossline direction, and the model is therefore ideal for our assumptions of interchanging the inline and crossline and gives an optimistic impression in contrast to the top inference, which is realistic.

To compare the two different methods’ ability to reproduce and preserve the geologic features and patterns, we migrate the ground truth and the results from the 3D Fourier interpolation and wavelet CNN reconstruction, using Kirchhoff migration, which are displayed in Figure 12. We see that compared to the 3D Fourier interpolation, the wavelet CNN reconstruction shows fewer footprints, clearer structural definition, and less migration noise. For a quantitative measurement and comparison, we compute the root-mean-square (rms) error and the peak signal-to-noise ratio (PSNR) of the entire interval of the premigrated and postmigrated sections, which are listed in Table 1. The PSNR is computed as

$$\text{PSNR} = 10 \log_{10} \left( \frac{\text{max}(Y^2)}{\text{N} \text{M} Y - \tilde{Y}^2} \right).$$

(15)

where $Y$ is the target, $\tilde{Y}$ is the prediction, and $MN$ are the number of samples. Here, we see that the wavelet CNN rms error shows an improvement close to a factor of two compared to the 3D Fourier rms error, which implies a considerable uplift for the CNN approach.

FIELD DATA EXAMPLE

We tested the proposed method on TopSeis (Vinje et al., 2017) field data from the Barents Sea. The seismic data contain 901 3D shot gathers displaying complex wavefield settings in addition to a difficult noise setting. Each shot was recorded by 14 streamers separated by approximately 50 m, using a temporal sampling rate of 2 ms and receiver separation along the streamer of 12.5 m. Each 3D shot gather was split into 14 2D shot gathers, one for each cable. The 2D shot gathers were cropped to a size of $250 \times 80$ (the number of temporal and spatial samples) and decimated to adapt to the crossline sampling interval for input. The subsampled and target gathers define the training data $D = \{(Y[i],X[i])\}_{i=1}^{K}$ for $K = 12,614$ training examples. In addition, we used data augmentation to increase the size of the training set, first by horizontal translation and second by horizontal flipping, increasing the number of examples in the training set to $K_{\text{train}} = 72,648$. An example of subsampled versus target gathers with corresponding frequency-wavenumber plot is displayed in Figure 13. In Figure 13d, we see that the subsampled example with a 50 m spatial sampling interval gives rise to severe aliasing in the frequency-wavenumber domain compared to the target example in Figure 13c. The subsampled example in Figure 13b is displayed with nearest-neighbor interpolation, only for visual comparison. We extracted one example from the test set to visually compare the reconstructed and target gather. The target, subsampled, and reconstructed gathers along with the reconstruction error and the rms-amplitude spectrum are shown in Figure 14. The DWT of the target and predicted DWT from the test set in Figure 14 are displayed in Figure B-1 in Appendix B.

In this example, the model is capable of reconstructing a complex wavefield with difficult structural patterns. Overall, the reconstruction error increases with depth where the 2D structure becomes more complex. The model struggles with frequencies greater than 50 Hz, and at high wavenumbers, which we can observe in the corresponding frequency-wavenumber spectra in Figure 15. This is also present for the reconstruction in the crossline direction presented in Figure 16. Similar to the synthetic example, we observe striping artifacts in these examples, which are present in the deeper part of the seismic record.

Table 1. The rms error and PSNR from the pre- and postmigrated crossline section from the entire interval.

<table>
<thead>
<tr>
<th></th>
<th>3D Fourier</th>
<th>Wavelet CNN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>rms</td>
<td>PSNR</td>
</tr>
<tr>
<td>Premigration</td>
<td>0.0086</td>
<td>41.3434</td>
</tr>
<tr>
<td>Postmigration</td>
<td>0.0124</td>
<td>38.1423</td>
</tr>
</tbody>
</table>

Figure 13. An illustration of a single training example (a and b) in the shot domain and their corresponding frequency-wavenumber plot in (c and d), respectively.
The loss of frequencies greater than 50 Hz implies a smoother reconstruction than the target. However, we would expect the model to struggle with the high-frequency part due to both complex wavefield patterns and noise settings. To have a quality measure between the different data sets, we computed the rms error and the PSNR. The average rms error and PSNR of the three data sets is listed in Table 2. The rms error and PSNR of the example in Figure 14 are 0.0497 and 26.0673, respectively. The rms error and PSNR for all shots in the test set are displayed in a scatterplot in Figure C-1 in Appendix C. The rms error from the test set reconstruction is close to the rms error in the validation set, which implies a good generalization from the proposed method. The training set shows a lower rms error, which we expected because the model is fitted on this data set.

The method manages to unwrap the signal at high frequencies and wavenumbers for curved events, linear events, and conflicting dips, in addition to preserving the characteristics of the wavefield in three dimensions. The characteristics of the 3D wavefield are more evident on time slices. We sorted the 3D shot gather presented in Figure 16 into time slices, and we extracted four slices corresponding to \( t \in [460,480,500,520] \) ms. The time slices — before and after reconstruction — are shown in Figure 17, where we observe in Figure 17e–17h a high-resolution wavefield with clear wavefield patterns implying a promising 3D wavefield reconstruction from the proposed method. Still, we observe the same striping artifacts and loss of high frequencies as seen in the inline case and the synthetic cases.

### DISCUSSION

One advantage of the proposed method is that there is no need for manual labeling, preinterpolation to learn from or any interpretation, leading to minimal human interference. Once the wavelet domain CNN is learned, reconstruction of the 3D wavefield is achieved automatically by applying to individual 2D crossline sec-

---

Table 2. The average rms error and PSNR comparison of the different sets.

<table>
<thead>
<tr>
<th></th>
<th>Test set</th>
<th>Validation set</th>
<th>Training set</th>
</tr>
</thead>
<tbody>
<tr>
<td>rms</td>
<td>0.04484</td>
<td>0.04487</td>
<td>0.04418</td>
</tr>
<tr>
<td>PSNR</td>
<td>27.1471</td>
<td>27.1318</td>
<td>27.2670</td>
</tr>
</tbody>
</table>

---

Figure 14. Reconstruction result from the inline direction from one arbitrary example in the test set where (a) is the target gather, (b) is the subsampled, (c) is the reconstruction, (d) is the reconstruction error, that is, the difference between (a) and (c), and (e) is the rms amplitude spectrum of (a-c).
tions. In our approach, the introduction of the wavelet transform adds additional complexity to the CNN approach because the wavelet basis of choice is to be considered an additional network hyper-parameter. We experimented with different types of wavelets from the DWT filter bank, such as Haar, coiflets, and symlets, and the Haar basis gave more stable results than the other wavelet basis. The stability of the Haar basis might relate to its short support, that is, the length of the wavelet, because the crossline section in the shot domain has only 14 traces for input. We could also consider using different wavelet basis in each direction, that is, Haar in the space domain and another discrete — or continuous — wavelet in the temporal direction, which could potentially improve the method in presence of a low signal-to-noise ratio (S/N).

The method is not dependent on any prior knowledge about the wavefield, such as velocities, and can therefore be trained on data comprising multiples and ghost reflections. This implies a broader use of the method, such as a tool for wavefield reconstruction in an earlier stage of the processing flow; therefore, it can potentially be used in demultiple and dehosting workflows and prior to migration. The early stage processing potential might be a topic for further investigation for application purposes. However, for the method to work optimally, it requires that the statistical properties of the wavefield in the inline direction are representative for the crossline direction.

As shown in the field data example in Figure 14 (inline) and Figure 16 (crossline), the prediction suffers from a loss of energy at high frequencies. From our tests, the introduction of wavelet-domain learning contributed slightly to restoring more of the high frequencies. In other tests, by using different objective functions such as $L_2$ or $L_1$ loss in combination with $L_1$ or $L_2$ regularization, the $L_1$ loss gave the most significant improvement in reconstruction at high frequencies. Zhao et al. (2016) report similar results when they investigate the difference between $L_2$ and $L_1$ loss in image restoration problems. They argue that the $L_2$ loss gets more easily stuck in a local minimum whereas the $L_1$ loss may be guided toward a better minimum, most likely due to the smoothness of the function and local convexity properties of the $L_2$ versus $L_1$. If the problem is related to a difficult noise...
setting, a potential solution could be to introduce more data to re-
duce uncertainty and increase robustness in presence of noise. Al-
ternatively, preprocessing of the data to remove some of the noise 
before training could also be a potential option.

Another potential challenge is in situations in which streamer 
feathering and/or streamer fanning causes large deviations in 
streamer spacing. The CNN approach is dependent on the learned 
characteristics of the wavefield given by the relationship between 
the target and the subsampled counterpart. Augmenting the training 
data by subsampling with different trace spacing can potentially im-
prove its robustness in these circumstances. Another potential chal-
lenge is the well-known problem in deep learning for image 
superresolution of reconstruction artifacts caused from instabilities 
due to carefully constructed noise, that is, adversarial attack (Antun 
et al., 2019). Because we have not considered problems related to 
the aforementioned challenges, this might be a potential topic for 
further research.

Concerning our hypothesis — using the densely sampled inline 
direction to learn nonlinear representations for reconstruction of 
the wavefield in the crossline direction — we consider the seismic 
wavefield as highly structured and smooth 3D signals represented 
by local geometrically shaped patterns. Even though the wavefield 
represents high complexity in terms of aliasing, conflicting dips, the 
seismic structure is determined by wave phenomena rather than 
geology. In this case, even though the inline and crossline differ 
in terms of geologic features and patterns, we consider it reasonable 
that the wavefield patterns and features are locally similar in both 
directions. A challenge in field data is where the wavefield struc-
tures and patterns could be damaged by a low S/N and processing 
artifacts from various workflows down the processing pipeline. 
Differences in anisotropy, dispersion, and other wave phenomena 
effects in addition to large differences in structural dip could also 
potentially play a role for the inline-to-crossline application. All of 
these effects, which are not considered in our synthetic example, 
increase the complexity and difficulty of learning a robust represen-
tation for wavefield reconstruction in different directions. To ad-
dress the challenges related to inline-to-crossline application, a 
much more comprehensive synthetic example should be used, by 
introducing realistic noise, more wave phenomena complexity, 
and strong structural differences in the inline compared to the cross-
line, followed by a complete processing flow, to produce a total 
imaging impact of the approach. However, this type of study is be-
yond the scope of this paper, and it is therefore planned for future 
research.

CONCLUSION

We propose using a deep CNN, trained in the wavelet domain, as 
a tool for seismic wavefield reconstruction beyond aliasing, in 
which nonlinear wavefield representations are learned on densely 
sampled seismic gathers — in the inline direction — to learn 
an upsampling function, which is applied on the undersampled 
wavefield in the crossline direction. We tested the wavelet-domain 
CNN method on synthetic data and field data where it manages to 
unwrap high frequencies at high wavenumbers. In the synthetic 
case, we compared the wavelet CNN to a Fourier-based industry 
standard method, where the wavelet CNN approach gave an im-
provement in rms error close to a factor of two. The deep-learning 
approach in this paper has demonstrated to be an effective tool for 
seismic wavefield reconstruction and an approach that allows for 
fast and efficient reconstruction of aliased seismic data. However, 
to improve the robustness of the model, more data should be in-
cluded by introducing more seismic gathers in combination with 
different augmentation approaches. In addition, solutions related 
to the loss of high frequencies and wavenumbers are yet to be 
solved and should be considered in future research.

Figure 17. Time-slice view of the input and reconstructed 3D shot gather displayed in Figure 16, where (a-d) are the inputs with 14 cables and 
(e-h) are the reconstruction results output with 53 cables.
ACKNOWLEDGMENTS

The authors would like to thank Lundin for allowing us to use the TopSeis data and P. Zhao in CGG for valuable discussions. This research is financially supported by the Research Council of Norway, project number 287664.

DATA AND MATERIALS AVAILABILITY

Data associated with this research are confidential and cannot be released.

APPENDIX A

TRAINING AND VALIDATION LOSS DURING TRAINING FOR FIELD DATA

An example from the loss during training from the training set batches (the black curve) and validation set (the red curve) is displayed in Figure A-1 with logarithmic scale on the x-axis. In this example, early stopping was set to train for 20,000 iterations without improvement in the validation loss before initiating.

Figure A-1. The $L_1$ loss during training up to early stopping for the training set batches and the validation set on the field data.

Figure B-1. The DWT of the target data and predicted DWT from the test set displayed in Figure 14.

Figure C-1. Scatterplot of the (a) PSNR and (b) rms error of all shots in the test set. The blue triangles represent the PSNR and rms for the test set example shown in Figure 14.
APPENDIX B
WAVELET DOMAIN

The DWT of the target and predicted wavelet domain model from the test set shown in Figure 14 is displayed in Figure B-1.

APPENDIX C
RMS AND PSNR PLOTS FOR FIELD DATA

The PSNR and rms for all shots in the test set are displayed in Figure C-1, with the larger blue triangles representing the test set example shown in Figure 14.

REFERENCES


Abma, R., and N. Kabir, 2006, 3D interpolation of irregular data with a POCS

V470 Greiner et al.

Bae, W., J. Yoo, and J. Chul Ye, 2017, Beyond deep residual learning for

Dong, C., C. C. Loy, K. He, and X. Tang, 2015, Image super-resolution using

Crawley, S., 2000, Seismic trace interpolation with nonstationary prediction-


Antun, V., F. Renna, C. Poon, B. Adcock, and A. C. Hansen, 2019, On in-

Ghaderpour, E., W. Liao, and M. P. Lamoureux, 2018, Antileakage least-

Ghaderpour, E., 2019, Multichannel antileakage least-squares spectral

He, H., and W.-C. Siu, 2011, Single image super-resolution using Gaussian

Antun, V., F. Renna, C. Poon, B. Adcock, and A. C. Hansen, 2019, On in-

Robertsson, J. O., I. Moore, M. Vassallo, K. Özdemir, D.-J. van Manen, and

Jaramillo, H. H., and N. Bleistein, 1999, The link of Kirchhoff migration and
degression to Kirchhoff and Born modeling: Geophysics, 64, 1793–1805, doi: 10.1190/1.144685.


Kingma, D. P., and J. Ba, 2014, Adam: A method for stochastic optimiza-

Kutscha, H., D. J. Verschuur, and A. J. Berkhout, 2010, High resolution
double focal transformation and its application to data reconstruction:

LeCun, Y., L. Bottou, Y. Bengio, and P. Haffner, 1998, Gradient-based learn-
ing applied to document recognition: Proceedings of the IEEE, 86, 2278–
2324, doi:10.1109/59.726791.


Lie, J. E., V. Danielsen, P. E. Dhelie, R. Sabileon, R. Siligie, C. Grubb, V. Vinje, C. J. Nilsen, and R. Setubas, 2018, A novel source-overlay-cable solution to address the Barents Sea imaging challenges: Marine Acquisition Work-
shop, 1–5.


Liu, P., H. Zhang, K. Zhang, L. Lin, and W. Zuo, 2018, Multi-level wavelet-
CNN for image restoration: Proceedings of the IEEE Conference on Com-
puter Vision and Pattern Recognition Workshops, 773–782.


Mallat, S. G., 1989, A theory for multiresolution signal decomposition: The wavelet representation: IEEE Transactions on Pattern Analysis and Ma-
chine Intelligence, 11, 674–693, doi:10.1109/34.192463.

Mallat, S. G., and Z. Zhang, 1993, Matching pursuit with time-frequency


Mandelli, S., V. Lipari, P. Bestagini, and S. Tubaro, 2019, Interpolation and

Naghizadeh, M., and M. D. Sacchi, 2008, fx adaptive seismic-trace inter-

tolation: Geophysics, 74, no. 6, V9–V16.

Naghizadeh, M., and M. D. Sacchi, 2010a, On sampling functions and

Fourier reconstruction methods: Geophysics, 75, no. 6, WB137–
WB151, doi:10.1190/1.350577.

Naghizadeh, M., and M. D. Sacchi, 2010b, Hierarchical scale curvelet

Özdemir, K., A. Özbek, D.-J. van Manen, and M. Vassallo, 2010, On data-
dependent multicomponent interpolators and the use of priors for optim-
ial reconstruction and 3D up/down separation of pressure wavefields:

Porsani, M. J., 1999, Seismic trace interpolation using half-step prediction

Robertsson, J. O., I. Moore, M. Vassallo, K. Özdemir, D.-J. van Manen, and A. Özbek, 2008, On the use of multicomponent streamer recordings for reconstruction of pressure wavefields in the crossline direction: Geophys-
ics, 73, no. 5, A45–A49, doi:10.1190/1.2953388.

Rumelhart, D. E., G. E. Hinton, and R. J. Williams, 1986, Learning repre-
sentations by back-propagating errors: Nature, 323, 533–536, doi:10.1038/323533a0.


Shi, W., J. Caballero, F. Huszar, J. Totz, A. P. Aitken, R. Bishop, D. Rueck-
tert, and Z. Wang, 2016, Real-time single image and video super-resolu-


Sun, Z., X. Zhu, and H.-Y. Shum, 2010, Gradient profile prior and its appli-
cations in image super-resolution and enhancement: IEEE Transactions
4.2 Shot point interpolation using deep neural network

The shot point interval in towed-streamer seismic exploration usually is significantly larger than the corresponding interval between each channel spacing. This subchapter investigates DNN as a tool for seismic shot point interpolation. Some publications exist on this topic, but none of them explore the case of complex marine seismic data. Thus, this observation has motivated the case study presented here. The proposed method is based on the U-Net architecture and is trained on densely sampled common shot gathers. When employed on under-sampled common receiver gathers, the trained network can interpolate shot points between existing ones. We have evaluated the efficiency of the proposed method on complex field data from a commercial marine towed dual-front-source survey in the North Sea. More specifically, we consider field data with an original flip-to-flip shot point interval of 37.5 m and decimate it by a factor of two. We aim to interpolate that decimated shot point and compare the interpolation quality with the corresponding real ones. In addition, we compare our method with the conventional f-x interpolation technique and demonstrate that the DNN approach has overall better performance. The field data example finds that the approach can reconstruct the seismic wavefield except for very high frequencies, and the interpolated shot points display promising features.

In some learning-based reconstruction and super-resolution techniques, the under-sampled input fed to the DNN is initially up-sampled to the same size as its label/target using a conventional interpolation method, such as bicubic spline. The neural network model is then trained to remove the blurring effect caused by the sub-optimal interpolation. Employing bicubic interpolation of the input was proposed in one of the first deep learning-based super-resolution techniques applied to natural images (Dong et al., 2015). Geophysical applications followed soon within shot domain interpolation (Wang et al., 2019a) and missing shot reconstruction (Wang et al., 2018, 2019b). However, pre-interpolation of the input using bicubic interpolation does not bring any additional information to the reconstruction problem (Shi et al., 2016).

Furthermore, by training correlation functions, CNNs applied to seismic data reconstruction and super-resolution problems are meant to correlate spatiotemporal data in a high-dimensional non-linear space. For two reasons, the correlation between traces is
challenging when trained on sparse and under-sampled seismic data. The first is that the distances between consecutive traces within the gather domain are large. The second arises from the fact that if we only consider the 2D wavefield structure (2D shot gathers, for example, extracted from the complete 3D shot gather), then the solution to the reconstruction problem suffers from a lack of information about the full 3D wavefield patterns. In the first case of large distances between the traces within the gather, the pure feed-forward CNNs with small kernels, such as residual networks (Wang et al., 2018, 2019b), do not represent optimal designs. The reason is that the receptive field increases only linearly in the forward pass of the network. Instead, this issue can be solved by other network architectures, for example, by skipping the pre-interpolation stage and using the sub-sampled data directly as input to the network. Thus, letting the up-sampling operation be integrated as part of the network (Paper II) or using encoder-decoder architectures (Park et al., 2019). These approaches provide a larger effective size of the convolutional kernels, i.e., smaller kernels correlate over larger spaces.

To address the seismic wavefield's 3D nature, Siahkoohi et al. (2019b) trained a Generative Adversarial Networks (GAN) using pairs of fully-sampled and under-sampled single-receiver frequency slices. They showed promising results for synthetic ocean bottom node data. Similarly to previous works (Park et al., 2019; Siahkoohi et al., 2019b; Wang et al., 2019b), we take advantage of the existing reciprocity between common shot gather (CSG) and common receiver gather (CRG) (Knopoff and Gangi, 1959). This case study demonstrates an extension to the original U-Net design for shot point interpolation. We replace the sub-optimal bicubic pre-interpolation stage with a zero-masking operation. In addition, to take into account the 3D nature of the seismics, we propose to include adjacent gathers to the input space, which provide additional information about the 3D structure of the wavefield. Therefore, the proposed method is an encoder-decoder hybrid 3D/2D approach with by-pass connections for seismic super-resolution, applied to under-sampled receiver gathers.

In this demonstration of shot point interpolation, we consider complex commercial marine field data from a towed dual-front-source survey with an original shot point interval of 37.5 m and decimate it by a factor of two. The quality of the interpolated shot points can then be directly compared with the corresponding real ones. We also compare the network interpolated shot points with a conventional f-x interpolation approach (Spitz, 1991). The proposed method interpolates the shots in two steps by training separate CNN models.
employing the same network structure. Step 1 uses three common shot gathers (CSGs) as input feature space to the network and trains on CSGs. This trained network is then employed on common receiver gathers (CRGs) to interpolate shot points between existing ones. However, after completing step 1, the spatial distance between each trace in the interpolated shot point is the same as the new shot point interval, i.e., 37.5 m. To recover the original trace spacing in the interpolated shot gather, we include step 2. Step 2 uses three CSGs as input to the network (one original shot gather on each side of the interpolated one), and it is trained to fill in these missing traces. The motivation for this work is to give an overall comparison between CNN and f-x based shot point interpolation methods and evaluate which of the two improves the quality by directly comparing them side by side with the ground truth shot gather. The performance of the proposed method is tested using field data from a 10 km × 10 km swath from a commercial marine towed dual-front-source survey in the North Sea.

4.2.1 Data conditioning for multi-feature input to the CNN

To quantify the performance of the DNN-based shot point interpolation method, we decimated conventional field data by discarding every other shot point, which will later serve as the ground truth. The original flip to flip shot point interval (Δs) is 37.5 m with channel spacing (Δr) of 12.5 m. After decimating every other shot point, we have Δs=75 m. The actual investigated interpolation procedure is a two-step approach. In step 1, we train the CNN model in the common shot domain and predict in the common receiver domain. Since the proposed method exploits the similarities between a CSG and a CRG, we precondition our data so that the distance between consecutive traces in both domains is equal, i.e., Δs=Δr. Not doing so will result in different aliasing patterns in the two domains. We intend to interpolate one shot point between two existing ones, therefore the desired Δs=37.5 m. Since the original 2D CSGs are spatially sampled at 12.5 m, with a total of 512 channels along the streamer, we decimate each CSG and select a subset of channels that are 37.5 m apart.

Figure 4-2 shows several consecutive shot point gathers plotted one below the other. The asterisks represent a shot point and the triangles receivers, where a shot-receiver combination will be a seismic trace. The grey shot channel combinations refer to the
decimated data and the red colored ones to the kept data. In the same figure, a vertical combination of receiver-shot pairs represents a CRG. We consider the red source-channel pairs as sampled traces and the grey ones we aim to fill in. Note that the adjacent CSG gathers in the top left image in Figure 4-2 have different patterns of sampled traces because of the chosen $\Delta s$ and $\Delta r$. Closer inspection shows that this pattern repeats every second gather. The first row in the same figure represents the training phase of step 1 with three consecutive gathers ($n-1$, $n$, $n+1$) as input feature space to the CNN to the left and its target gather ($n$) to the right. The second row denotes the prediction (inference) step, where the green shot channel combinations are interpolated by the trained network employed on data sorted to CRGs. After completing step 1, we interpolate one new shot point between two existing ones, but only for a channel interval of 37.5 m (the decimation interval chosen in step 1). Therefore, step 2 is designed to fill in channels in each of the new CSGs corresponding to a channel interval of 12.5 m. Figure 4-3 shows the training phase of step 2 in the first row of images. This trained network is then applied to the new interpolated shot gathers from step 1 to achieve a final channel spacing of 12.5 m shown in the second row of images in Figure 4-3. Similar to step 1, we have three consecutive input gathers to the left and the middle-inferred gather to the right.

Figure 4-2. Schematic representation of step 1 of the DNN-based shot point interpolation workflow. The red-colored traces represent the sampled ones, the grey ones we aim to fill in, and the green ones the interpolated traces after completing step 1. The first row represents the training phase employing decimated shot points, and the actual interpolation applied in the common-receiver domain is sketched in the second row.

Figure 4-4 visualizes how we organized the adjacent seismic gathers for the two interpolation steps in the data domain. The gathers in Figure 4-4a represent step 1 of the
workflow. With zero traces at the locations that we want to fill in, these masked gathers represent either CSGs at the training phase or CRGs at the prediction phase, while the corresponding target is shown to the right in Figure 4-4a. The data size is $M \times N \times k$ in Figure 4-4a, where $M$ is the number of time samples, $N$ is the number of traces (channels in the case of CSGs or different shot-channel pairs in the case of CRGs), and $k$ is the number of features (consecutive gathers), which corresponds to $1000 \times 171 \times 3$ samples with $\Delta r=37.5$ m. Figure 4-4b shows the combination of CSGs, which represents step 2 of the workflow. As in Figure 4-4a, the input gathers are displayed to the left and their target to the right. Note that the input to the network has one original CSG on each side of the masked one that is to-be-interpolated. The data size $M \times N \times k$ in Figure 4-4b corresponds to $1000 \times 512 \times 3$ samples with $\Delta r=12.5$ m.

Figure 4-3. Schematic representation of step 2 of our DNN-based shot point interpolation workflow. The first row represents the training phase, and the actual interpolation is sketched in the second row.

Figure 4-4. Arrangement of 3 consecutive gathers fed into the network. a) Three input gathers and their target to the right (step 1 of the DL-based interpolation flow); b) Three input CSGs and their target to the right (step 2 of the DL-based interpolation flow).

### 4.2.2 Customized Convolutional Neural Network

A set of training data consisting of an input and corresponding target data representing the
labeled are provided to the customized CNN, based upon the original U-Net proposed by Ronneberger et al. (2015). The supervised learning approach aims at finding a non-linear function that can map the data from one representation to another more desired one. Figure 4-5 illustrates the modified network architecture. It consists of a contracting path (encoder) and an expansive path (decoder). Each down-sampling block uses $3 \times 3$ convolutions, except the first layer that uses a kernel size of $5 \times 5$. We included padding to preserve the size of the image. Each convolution is followed by an activation function of the “ReLU” type, and at the end of the down-sampling block a $2 \times 2$ max pooling operation with stride 2 is included. Each expansive block consists of an up-sampling of the feature maps through $2 \times 2$ transposed and padded convolutions with a stride of 2. The output of each transposed convolution is concatenated with its corresponding feature maps from the contracting path before moving to the next convolutional layer. Each up-sampling goes through two $3 \times 3$ padded convolutions followed by the ReLU activation function, except for the last up-sampling block, whose last layer uses a kernel size of $5 \times 5$. Similarly, to the original proposed U-Net architecture, at the final layer, a $1 \times 1$ convolution is used to map each 64-component feature map to the desired output image. The U-Net uses the full data bandwidth in the first layers, which are not down-sampled and later concatenated with the last layers. U-Nets are good because they can learn the structure of the seismic data (in the encoder-decoder part) while using skip-connections to retain the detailed high-frequency information. Once the mapping function is learned by identifying specific patterns in the given data examples, the trained network can be applied to unseen data and provide predictions according to these learned patterns.

Figure 4-5. Architecture of our customized U-Net. The network consists of several layers where the numbers on top of each of them represent a collection of 2D features.
Compared to Paper II in this work, we did not use the wavelet domain and employed different NN architecture. Moreover, the difference in the formulation lies in the way we define the observed seismic gather:

$$D \odot Y = X,$$

(4-1)

where $Y \in \mathbb{R}^{M \times N \times k}$ denote a seismic gather, or target of size $M \times N \times k$, where $M$ is the number of time samples, $N$ is the number of traces (channels in the case of CSGs or different shot-channel pairs in the case of CRGs) and $k$ is the number of features (consecutive input gathers), $X \in \mathbb{R}^{M \times N \times k}$ is the masked counterpart of $Y$ by the known operator $D \in \mathbb{R}^{M \times N \times k}$. The $D$ operator here is not a decimation operator but a masking operator, represented by ones at sampled traces and zeroes everywhere else, and $\odot$ is the Hadamard product.

The remaining learning-based methodology follows the one described in Paper II. In contrast with Paper II, the loss in this study is computed with the $L_2$ norm.

### 4.2.3 Field data experiments

The field data used for demonstration purposes of the proposed shot point interpolation are from the North Sea with an average water depth of around 150 m. This seismic survey was a conventional 3D narrow azimuth towed dual-front-source acquisition with 12 streamers; shot point interval of 18.75 m (or $\Delta s=37.5$ m flip-to-flip); streamer length of 6400 m with 12.5 m channel group-forming; 75 m streamer separation; record length of 4 s with 4 ms sample interval and natural bin size of $6.25 \times 18.75$ m.

The proposed method was employed on a 10 km $\times$ 10 km swath. The test area consisted of 26 adjacent sail-lines. The distance between the individual sail-lines was 450 m. We trained the network on a single sail-line positioned at the center of the test area. The training data consist 10000 gathers. 8500 randomly selected gathers were used as a training set and the remaining for validation. As already discussed, the customized network is trained in two steps. The training phase for step 1 ran for all the requested 100 epochs, while step 2 ran for 89 epochs before early stopping was initiated due to no further improvement. The two training steps took approximately 10 and 24 hours to finish on the GPU configuration NVIDIA GTX1080TI. The prediction of a single subline (one source-streamer combination) with 275 CSG as input took a total of approximately 84 s.
In comparison, $f$-$x$ interpolation of the same amount of data took 102 s on CPU configuration Intel Xeon E3-1240 v5. We extracted one example from the test data set to visually compare the interpolated gathers from the $f$-$x$ interpolation and the proposed method with the ground truth CSG.

Figure 4-6 and Figure 4-7 show a series of frequency panels for the shallow and deep parts of the field data. The individual octave panels are normalized to the same value for visualization purposes. The magnified figures highlight the data recorded up to 3.5 s and the first 1 km offset along the streamer. The individual rows from top to bottom in these figures represent the ground truth data, the $f$-$x$ interpolated data, the DNN-based interpolated data, and the difference of the two methods with respect to the ground truth. The columns represent the corresponding frequency panels for each of the tests, wherefrom left to right, we have the full frequency band; 2-4 Hz; 4-8 Hz; 8-16 Hz; 16-32 Hz; 32-64 Hz, and 64-125 Hz, respectively. It is difficult to discriminate between the different results from a judgment based on the full-band data (e.g., column 1 in Figure 4-6 and Figure 4-7). However, decomposing the data into various frequency bands makes such analysis easier. Comparing the low side of the frequency spectra up to 8 Hz, or columns 2 and 3 from left to right in Figure 4-6, we cannot observe any significant differences between the two interpolated gathers. The minor discrepancies again support this observation compared with the ground truth for both interpolated gathers. Similar results are achieved for the deeper part of the section shown in Figure 4-7. The frequency range from 8-32 Hz, or columns 4 and 5 from left to right in Figure 4-6 and Figure 4-7, shows less difference with the ground truth in favor of the DNN-based interpolation method, especially for both negative and positive dipping energy. The last two-octave panels show similar quality in interpolation by the two methods, both in the shallow and in the deep part of the examined CSG.

We can also note that the error difference for the DNN-based technique in the shallower window is larger than that in the deeper section. Figure 4-6 and Figure 4-7 also show that the proposed method reconstructs the signal for curved, linear, and conflicting dipping events. The investigated CSG’s corresponding amplitude and $f$-$k$ spectra in Figure 4-6 and Figure 4-7 are shown in Figure 4-8 and Figure 4-9, respectively. We observe that the proposed method suffers from some high-frequency loss starting from approximately 40 and 30 Hz for the shallow and deep windows, respectively. We have noticed around 1 dB weakening in the amplitude spectra for a reference frequency of 40 Hz in the shallow section.
and approximately 3 dB weakening for the same reference frequency in the deeper area. Higher frequencies showed even more weakening. Table 4-1 lists the Root Mean Squared Error (RMSE) and the Peak Signal to Noise Ratio (PSNR) accuracy metrics for the analyzed CSG, computed over the full gather and the full bandwidth. Both metrics indicate that the proposed DNN based interpolation performs well.

Figure 4-6. Frequency panel display of a CSG taken from a selected part of the shallow section. a) Ground truth; b) f-x interpolation; c) DL-based interpolation; d) Difference a) minus b); e) Difference a) minus c).
The characteristics of the interpolated wavefield and, more specifically, its coherency are more evident on common offset gathers. Such gathers are shown in Figure 4-10, where channel number five is selected corresponding to an approximated offset of 212.5 m. One can notice that nearly flat events dominate the shallower part. However, within the time window of approximately 1-2 s, the data show a more complex character with diffraction-like events crossing the weaker near-horizontal ones. The originally sampled data at $\Delta s=37.5$ m, represented by Figure 4-10a, are already slightly aliased. Once we increase $\Delta s$ and simulate a dataset with $\Delta s=75$ m, as shown in Figure 4-10b, these diffracting events become strongly

Figure 4-7. Frequency panel display of a CSG taken from a selected part of the deep section. a) Ground truth; b) f-x interpolation; c) DNN-based interpolation; d) Difference a) minus b); e) Difference a) minus c).
aliased and represent a significant challenge to any interpolation tool. Figure 4-10c and Figure 4-10e show the results of the f-x interpolation-based method and the proposed DNN-based interpolation method, respectively.

Table 4-1 – PSNR and RMSE from the analyzed CSG computed over the full gather and bandwidth.

<table>
<thead>
<tr>
<th></th>
<th>f-x-based method</th>
<th>DL-based method</th>
</tr>
</thead>
<tbody>
<tr>
<td>RMSE</td>
<td>0.0173</td>
<td>0.0125</td>
</tr>
<tr>
<td>PSNR</td>
<td>35.22</td>
<td>38.07</td>
</tr>
</tbody>
</table>

The difference plot with respect to the ground truth for each of these two methods is displayed in Figure 4-10d and Figure 4-10f. We can observe that the nearly horizontal events have good continuity for both the f-x-based and the proposed interpolation methods. However, Figure 4-10e shows slightly better coherency of the diffraction-like energy than Figure 4-10c. The difference plot also suggests less residuals for the DNN-based approach are observed. Even though the proposed interpolation technique is a two-step approach, the character of the wavefield is well preserved.

### 4.2.4 Discussion and challenges

The presented strategies for CNN learning utilizing the reciprocity of CSGs and CRGs of non-linear representations for interpolation of shot points perform better than a standard f-x interpolation. The method demonstrated good signal preservation and better performance than the f-x method, especially for the vital mid-frequency range between 8-32 Hz. When sorted
to the common channel domain, the DNN results showed better defined diffraction-like energy than the $f$-$x$ data. However, a noticeable loss of high-frequencies was observed for the proposed DNN method. Paper II reported a similar loss of high-frequency energy when investigating wavefield reconstruction in the crossline dimension. Rahaman et al. (2019) and Xu et al. (2019) emphasize a general learning bias of DNN towards low-frequency functions. Therefore, we tried splitting our input data into different frequency bands, trained individual CNN models for each frequency band, and then superimposed the results. However, we did not observe much difference from conventional learning based on the full-frequency band. More discussion on handling the high-frequency loss is addressed in the Discussion and outlook section of the thesis.

Figure 4-9. CSG $f$-$k$ spectra (normalized to the largest peak). a) and b) Ground truth shallow and deep, respectively; c) and d) $f$-$x$ interpolation shallow and deep, respectively; e) and f) DNN-based interpolation shallow and deep, respectively.
Figure 4-10. Common offset gather with $\Delta s = 37.5m$. a) Ground truth; b) Decimated ground truth or input to interpolation; c) f-x interpolation; d) Difference a) minus c); e) DNN-based interpolation; f) Difference a) minus e).
5. 3D interpolation and regularization of seismic data using Deep Neural Networks

Chapter 4 investigated two cases of using CNN to interpolate 2D gathers extracted from the 3D marine shot gathers. Instead of working with 3D shot gathers, we deal with 3D offset class data in this chapter to explore the CNN capabilities of working directly on 3D volumes of data.

Regularization and interpolation of 3D offset classes before imaging is an essential and challenging task. The challenge arises because the minimum offset for marine towed-streamer data is sparsely and irregularly sampled, which results in severe aliasing for the high frequencies. This chapter investigates how to perform this task using a deep neural network. As in any other DNN approach in seismics, the problem of suitable training data remains. Target or labeled data for training are hardly available because such data are not acquired. Therefore, this chapter investigates two strategies for meeting the criterion of getting labeled data. The first and the most straightforward way is to train a DNN that directly replicates an existing conventional method, using a subset of the conventionally processed data. This trained DNN model can then be utilized to predict data outside the training area. This supervised method uses a multi-feature approach that includes adjacent 3D offset classes. The target data have been obtained by employing a multi-dimensional anti-leakage Fourier transform (ALFT) (Xu et al., 2005, 2010). In this context, this chapter aims to give valuable insight into improving DNN-based interpolation methods used for seismics. We investigate 2D and 3D convolutional kernels to train the DNN using vertical, horizontal 2D cross-sections or 3D volumes of data, respectively. This direct replication of the conventional method shows that the DNN-based process can speed up the interpolation and regularization step of the processing workflow at the expense of reduced accuracy. These findings led to a novel approach investigated in Paper III, which would have been impossible without the preceding work. Paper III describes how a tailored training dataset can be generated by de-migrating stacked pre-stack depth migration (PSDM) images. For each offset class volume, a de-migration of the PSDM stacked image into two configurations is proposed: (i) the original survey configuration consisting of the recorded source/receiver positions, and (ii) an idealistic
survey configuration with constant offset and azimuth for each offset class volume. 3D convolutional encoder-decoder is then trained to map each offset class from (i) to (ii) and use this trained model on the recorded data. The proposed DNN-based method in Paper III reduces the computational cost and improves the quality of the interpolated and regularized data compared to the ALFT. However, the technique is still implicitly dependent on some prior interpolation technique to get an initial image of sufficient quality to be used for de-migration. This chicken-and-egg problem is shared with a range of other processing and imaging techniques. For instance, an initial velocity model is required to obtain a subsurface velocity model using Full Waveform Inversion.

The main difference here is that only a subset of the recorded data needs to go through migration and de-migration for training data generation, ideally from geology-specific areas of the survey for better model generalization. The trained DNN model can then be utilized for the whole survey area. An alternative way will be to use a PSDM image from another nearby survey or combine several surveys with different geology and geometries to make the DNN more generalized and robust.

5.1 Problem statement

All 3D marine towed acquisition geometries show poor sampling along with one or more data dimensions. In addition, changing sea currents, wind, and waves make it nearly impossible to position the sources and the receivers at their planned locations. Thus, the acquired data are also irregularly sampled. Processing techniques, such as migration, rely on regularly and densely sampled data to achieve constructive interference to image the subsurface and destructive interference to suppress migration noise (Abma et al., 2007). Even in areas without dipping reflectors, missing data or irregularities of the recorded CMP position, offset, or azimuth will result in sub-optimal migration results (Poole and Lecerf, 2006). Multiple elimination methods assume a uniform distribution of source and receivers (Verschuur and Berkhout, 1997; Weglein et al., 1997). Extending these multiple elimination methods to 3D requires data regularization and interpolation of missing data (Van Dedem and Verschuur, 1998). Therefore, it is common in seismic data processing to employ interpolation and regularization of data. The interpolation process fills in the missing data, and the regularization maps the data from the irregularly recorded locations to CMP locations on a regular grid. For migration methods using offset classes as input, like Kirchhoff, the
interpolation and regularization of offset classes data plays an essential part in the processing workflow. As discussed in Chapter 2, an offset class is defined as a 3D sub-volume of the recorded seismic data within a specific offset range, usually determined by twice the distance between the shots along the line. Figure 5-1 shows the offset class formation for a triple source-over-streamer (Vinje et al., 2017) acquisition, where the red asterisk indicates the active source with streamers (in orange) separated by 50 m and receivers (green dots) separated by 12.5 m along each streamer. The traces from the receivers within the half-circle with 50 m radius from the sources will fall into the first offset class 0 – 50 m. These traces are mapped into the bin cells associated with the source-receiver CMP. Although we have indicated the CMPs with red dots for the active source $S_3$ and in gray for the inactive sources $S_1$ and $S_2$ in Figure 5-1a, all these CMPs contribute to the offset class formation. As we can see from this example, the first offset class is sparse, and only the central streamers contribute to its formation. Consequently, holes appear in the coverage. Adding another sail line (Figure 5-1b) and plotting all CMPs (all red dots) contributing to the offset class, we notice that the coverage will have even larger gaps between sail lines. The second offset class will also be sparse, and for the larger offset classes,
the coverage will improve until eventually being nearly complete. In the case of an actual acquisition, the problem is even more complex. Figure 5-2a shows the observed recording positions overlaid by the acquisition bin grid for a given offset range. The data are not positioned at the center of the bins, and the number of traces within each bin also varies (irregularly sampled). Some bins are either under-populated (no traces) or over-populated (more than one trace). These missing and irregularly sampled data are due to two main factors: (i) acquisition layout and shot point interval, and (ii) misplaced position of the seismic sources and streamers due to a dynamic marine environment. The desired input to pre-stack 3D offset classes migration is more in-line with Figure 5-2b. The data are interpolated (no empty bins) and regularized (regularly sampled along with the spatial dimensions) with source and receiver positions that give constant offset and constant azimuth for each trace. Offset-based migration, as mentioned, relies on regularly and densely sampled data, but other processing methods, such as surface-related multiple elimination (SRME) (Verschuur et al., 1992), are also known to be sensitive to the availability and quality of the reconstructed near-offset data.

This chapter deals with field data acquired in the Barents Sea. The water depths are relatively shallow (approximately 350 m) with a very hard and rugose seafloor, which sets up complex multiple patterns that are difficult to attenuate (Dhelie et al., 2018a). In addition, the significant velocity contrast due to erosion and uplift of the area close to the seafloor allows only a narrow cone of energy to penetrate and illuminate the subsurface targets (Lie et al., 2018). The source-over-streamer acquisition set-up (Vinje et al., 2017) significantly improves both modeling of multiples and imaging of the challenging Barents Sea subsurface (Dhelie et al., 2018c) because it provides marine split-spread near – and zero – offset seismic data. However, once these data are sorted to 3D offset classes, the near-offssets are still not fully populated. Thus, these near-offset classes suffer from spatial aliasing from coarse sampling across the inline and crossline dimensions. To mitigate this problem, various interpolation methods exist to improve data consistency. Nevertheless, the interpolation problem is ill-posed in the case of aliased data. Thus, many possible solutions may explain the observed
aliased data. Consequently, a priori information about the wavefield, such as subsurface velocities, or assumptions like linearity or sparsity, is required to reconstruct the missing seismic data more uniquely. In this chapter, the use of DNN is investigated as a tool to interpolate and regularize 3D offset classes. Like conventional methods, we provide a priori information to assist the DNN algorithm in finding a more optimal solution. Use of such physical constraints improved the achieved results significantly.

5.2 Replication of an existing interpolation and regularization method using deep neural network

This subchapter focuses on building a DNN tool for interpolation and regularizing 3D offset classes. The motivation behind this study is to determine if a trained DNN can replicate a widely used algorithm in the seismic industry and produce high-quality seismic images at a reduced cost. The conventional method is given by the state-of-the-art anti-leakage Fourier Transform (ALFT) (Xu et al., 2005, 2010), which serves as the target for the DNN. The section briefly describes the customized network employed in this study. It focuses on analyzing various strategies to improve the accuracy of the offset class interpolation and regularization by the DNN. The techniques include DNN training using multi-feature vertical, horizontal 2D cross-sections, or 3D volumes of data. These different data orientations are to find out which suits the practical problem best.

5.2.1 Anti-Leakage Fourier Transform as labeled data

The ALFT (Xu et al., 2005, 2010) is an interpolation and regularization method using an iterative approach to compute a frequency-wavenumber representation of irregularly sampled data. It then uses an inverse Fourier transform to reconstruct data at new locations. The inverse Fourier transform is straightforward and is implemented using an inverse discrete Fourier transform (DFT). However, the forward transform using DFT is not optimal and suffers from so-called “spectral leakage”. Think of a single sinusoid signal (finite) sampled at given positions to understand this latter term. If that sampling is regular in space (or time), its spectrum will be filled by zeroes for all wavenumbers except for the spectral element corresponding to its wavenumber. However, when sampled irregularly, the spectrum will have additional non-zero spectral amplitudes corresponding to other sinusoids, which arises because the Fourier basis functions are no longer orthogonal (Xu et al., 2005). The additional
non-zero components (or noise) in the spectrum are referred to as spectral leakage. The goal of the ALFT is to estimate the Fourier coefficients from irregularly sampled data with reduced wavenumber leakage.

To reduce the leakage of the Fourier coefficients for a 1D signal \( \mathbf{x}_\ell = [x_1, x_2, ..., x_{N_p}] \) with known irregular sampling positions \( \ell \), we can consider the normalized Fourier summation for a single wavenumber value \( k \) of the Fourier transform variable. The forward and inverse Fourier transforms are defined as follows (Xu et al., 2005, 2010):

\[
\hat{f}(k) = \frac{1}{\Delta X} \sum_{\ell=1}^{N_p} \Delta x_\ell f(x_\ell) e^{-2\pi i k x_\ell},
\]

\[
f^k(x_\ell) = \hat{f}(k) e^{2\pi i k x_\ell},
\]

where \( \hat{f}(k) \) denotes the Fourier coefficient associated with \( k \), \( \Delta X \) is the summation range, where \( \Delta x_\ell \) is the data weight in the summation, which for 1D case is the distance between two adjacent irregular samples, and \( f^k(x_\ell) \) denotes the component of \( k \) in the input data. Xu et al. (2005) proposed the ALFT, which works by estimating the Fourier coefficients recursively, starting with the one with the maximum energy and proceeding downward in energy until the minimum energy has been reached. The reason for starting with the maximum is that large coefficients have more leakage than smaller ones. After each step of estimation, they reset the calculated \( \hat{f}(k) \) to zero by updating the input data, which is equivalent to removing the \( f^k(x_\ell) \) component from the input data:

\[
f^{u}(x_\ell) = f(x_\ell) - f^k(x_\ell),
\]

where the algorithm can be summarized as:

1. Use Equation (5-1) to compute all Fourier coefficients of the input data.
2. Select the coefficient with the maximum energy.
3. Subtract the contribution of this coefficient from the input data using Equation (5-2).

The last step becomes the input for solving the next coefficient, until all the values in the updated input tend to zero. The ALFT algorithm can be used to calculate the Fourier
coefficients of higher-dimensional data sets (Xu et al., 2005, 2010; Poole, 2010). The ALFT in this study made use of inline, crossline, offset, and time dimensions.

5.2.2 Supervised learning-based interpolation and regularization

Consider an Offset Class (OFC) sorted into a binned 3D cube \( d_{r,h} \), where \( r \) stands for recorded, and \( h \) denotes the offset class index, where \( d_{r,1} \) is the first (smallest) offset class, \( d_{r,2} \) is the next one etc. \( d_{r,h} \) is a three-dimensional function \( d_{r,h} = d_{r,h}(x, y, t) \), where \( (x, y) \) denotes the spatial coordinates of the bin centers, and \( t \) denotes time. In the binning process, each recorded trace is mapped into the bin cell in \( d_{r,h} \) closest to the CMP position of the trace. The result is a series of OFCs with irregularly located traces and some empty bins, as shown in the left part of Figure 5-2a.

Now, suppose that \( n_h \) denotes the number of OFCs, where each OFC has \( n_x \) crosslines, \( n_y \) inlines, and \( n_t \) temporal samples. Then the discrete samples of \( d_{r,h}(x, y, t) \) form a 3D tensor, which defines the set of OFCs as \( \{d_{r,h}\}_{h=1}^{n_h} \). For notational convenience, let the recorded data \( d_{r,h} \in \mathbb{R}^n \) represent a vector according to its lexicographic order of size \( n \times 1 \), where \( n = n_x n_y n_t \).

In principle, we would like to solve the following inverse problem:

\[
d_{r,h} = Mm_{r,h} + \varepsilon_{r,h},
\]

where \( d_{r,h} \in \mathbb{R}^n \) is the sparsely sampled recorded data, \( M: \mathbb{R}^n \to \mathbb{R}^n \) denotes the mapping operator, \( m_{r,h} \in \mathbb{R}^n \) denotes the Ideal data and \( \varepsilon_{r,h} \) is the observation noise. Now, assume that there exists a non-linear functional representation of the Ideal data that takes the sparsely sampled data as input, i.e.,

\[
d_{r,h} = Mf(d_{r,h}; W, b) + \varepsilon_{r,h},
\]

where \( f(\cdot; W, b) \) denotes an arbitrary neural network, where the network is parameterized by \( W \) and \( b \) denoting the convolutional kernels and biases, respectively. The goal is then to find an approximation of \( f(\cdot; W, b) \), such that:
\[
\hat{m}_{r,h} = f(d_{r,h}; \hat{W}, \hat{b}),
\] (5-5)

where \( f(\cdot; \hat{W}, \hat{b}) \) can be considered a generalization of the approximate inverse of \( M \), and \( \hat{W} \) and \( \hat{b} \) denote the trained parameters. However, the Ideal data \( m_{r,h} \) is practically unknown. Therefore, to construct training data set, the ALFT (Xu et al., 2005, 2010) method is used to create an approximation of the Ideal data using the recorded data \( d_{r,h} \):

\[
\hat{m}_{r,h} = A_h(d_{r,h}),
\] (5-6)

where \( A_h \) represents the ALFT method. In this case, the \( \hat{m}_{r,h} \) inherits a similar feature space as the unknown OFC \( m_{r,h} \). Then, by utilizing the ALFT approximation of the Ideal data \( \hat{m}_{r,h} \), we can solve the following inverse problem instead of Equation (5-3):

\[
d_{r,h} = M\hat{m}_{r,h} + \varepsilon_{r,h},
\] (5-7)

where \( M \) is the mapping operator extracted from Equation (5-3). Compared to Equation (5-3), we now have access to a representative Ideal labeled data \( \hat{m}_{r,h} \) to be used to train our supervised deep learning model. To learn the parameters deep learning model parameters, we minimize a misfit function, including a regularization penalty term, given by the following objective function:

\[
\min_{W,b} \left\{ L \left( \hat{m}_{r,h} - f(d_{r,h}; W, b) \right) + \lambda R(W) \right\},
\] (5-8)

where \( L \left( \hat{m}_{r,h} - f(d_{r,h}; W, b) \right) \) is the loss term, \( R(W) \) is the regularization term, and \( \lambda \) is the regularization parameter controlling the trade-off between the loss and the regularization. Minimizing expression (5-8) is achieved by solving an optimization problem. First, the sparsely sampled and the Ideal data are divided into smaller patches (mini cubes), with \( T \) as the number of training examples (or mini cubes) \( \{\left(\hat{m}_{r,h}^{(i)}, d_{r,h}^{(i)}\right)\}_{i=1}^{T} \). These patches are then grouped into randomized batches of size \( B \), such that the input to the optimization algorithm consists of mini-batches of randomized training examples. In our case, we employed a first-order stochastic gradient descent algorithm with mini-batches (subsamples), known as decoupled weight decay (Loshchilov and Hutter, 2017), which is a weight decay version of the Adam optimization (Kingma and Ba, 2014). The decoupled weight decay algorithm for updating the weights reads:
\[ W_n = W_{n-1} - \eta \left( \frac{1}{B} \sum_{i=1}^{B} (\nabla_W C(W, b)_{n-1}) + \nabla_W R(W)_{n-1} \right), \]  

(5-9)

where \( \eta \) is the learning rate, \( n \) is the iteration, \( \nabla_W C(W, b)_{n-1} \) is the partial derivatives of the misfit function with respect to the weights using the Adam update rule \( (\text{Kingma and Ba, 2014}) \), \( \nabla_W R(W)_{n-1} \) are the partial derivatives of the regularization term, and \( B \) denotes the batch size. In this case, the loss in Equation (5-8) and the regularization of the weights are both given by the \( L_2 \) norm. Updating the bias parameters is done similarly, but without the regularization term.

### 5.2.3 Customized Convolutional Neural Network

A set of training data consisting of an input and corresponding target data representing the labeled are provided to the customized CNN. The supervised learning approach aims at finding a non-linear function that can map the data from one representation to another more desired one. For more details about the architecture of the CNN shown in Figure 5-3 please refer to Section 4.2 Shot point interpolation using deep neural network, where we used the same network. For the 3D application of the network, we increased the dimensionality of each convolutional kernel or pooling operator, i.e., \( 3 \times 3 \) becomes \( 3 \times 3 \times 3 \).

![Figure 5-3. Architecture of the customized U-Net. Note the highlighted multi-channel input shown at the bottom left corner of the figure. The input corresponds to vertical or horizontal cross-sections for the 2D experiments and volume data for the 3D experiments.](image)
5.2.4 Strategies of supervised learning-based offset class interpolation and regularization using ALFT as labeled data

A conventional CNN reconstruction based on 2D convolutions of an incomplete input requires the selection of a 2D vertical cross-section (e.g., inline or crossline) from the 3D offset class volume. Such an incomplete vertical cross-section will serve as input data to the CNN. The desired label will be its complete interpolated and regularized version derived from other reference interpolation methods. Seismic events of consecutive offset volumes are correlated, with the degree of correlation determined by the number of offset volumes selected and the complexity of the geology. Therefore, we can train the network by employing multi-feature input data that includes adjacent offset volumes following the to-be-predicted offset volume as additional features. Adding extra features as input feature space to the CNN, corresponding to increasing offset representations of the same cross-section, will force the convolutional kernels of the first layer to use this additional information. Forming a multi-feature vertical cross-section is the most intuitive approach because the seismic events are coherent in the $t$-$x$ domain. However, these vertical cross-sections are still oriented along one spatial dimension only (e.g., either inline or crossline). In the case of real data, the missing data or the holes in the 3D offset classes are irregular in size and may be large in one dimension but reasonably small in other dimensions. Consequently, a CNN approach employing such multi-feature vertical cross-sections in the $t$-$x$ domain will not be able to “see” and utilize any data in the orthogonal dimension. One can use time slices as an alternative approach, where the seismic signal is still coherent. In this way, the simple 2D convolutional kernels can see higher-offset classes data from both the inline and the crossline spatial dimensions.

In addition to an interpolation based on multi-feature cross-sections, this multi-feature method can be further extended from 2D to 3D by employing multiple offset classes sorted in 4D tensors (e.g., inline, crossline, time, and offset) for training and prediction. Compared with an interpolation based on multi-feature cross-sections, a multi-feature 3D volume-based interpolation uses 3D convolution kernels instead of 2D convolution kernels in operation. The advantage of multi-feature 3D volume-based interpolation is that multi-
dimensional wavefields are directly utilized in data reconstruction. However, the main disadvantage of this method is the costly computation and large memory requirement due to the 3D convolution calculations. We give a detailed discussion of the approaches mentioned above in the field data experiments section.

5.2.5 Data conditioning for multi-feature input to the CNN

The input data to the network are sorted into 3D offset classes (Figure 5-1a) with static binning applied, i.e., a single trace per offset class per bin is selected. The selection criterion is based on the minimum distance to the bin center. This trace selection can also be extended to include an azimuth-based selection to handle multi-azimuth-based interpolation. Similarly to conventional interpolation and regularization methods, additional information from longer offsets improved the results significantly. Consequently, we use a multi-feature input to the CNN, where each feature represents increasing offsets with a Normal Move Out (NMO) correction included based on stacking velocities. In addition, we applied a standard T2 gain to balance the amplitudes of later arrivals. Both the NMO and the T2 gain are easily reversible after the interpolation.

5.2.6 Field data experiments using ALFT as labeled data

We used field data from the Barents Sea to demonstrate the CNN-based interpolation and regularization of 3D offset classes. The average water depth of the studied area was 375 m, and the seismic survey was acquired using a source-over-streamer configuration (Vinje et al., 2017). This set-up (Figure 5-1a) included the use of 3 air-gun sources towed over the middle of a 12-streamer spread with a total length of 7050 m and a 12.5 m channel group-forming. The survey had a flip-to-flop shot point interval of 8.33 m and a streamer separation of 50 m. The natural bin size was 6.25 m × 8.33 m. The data were output from the complete sail-line processing and resampled to 4 ms. Each 3D offset class bin center started at −3475 m and ran to 3475 m with 50 m increments.

We carried out a series of experiments to analyze the actual network performance. First, a 2D CNN approach was considered with the following data sorting: (i) training using
vertical sections, i.e., inline or crossline sorted data, and (ii) time slice approach. We modified
the network to 3D to take into account 3D mini-patch volumes extracted from the same
training area as the 2D approach by keeping the feature size of each layer of the network intact
(Figure 5-3). The difference between these approaches is summarized in Table 5-1. To
quantify the quality of the experiments, we computed the Root-Mean-Square Error (RMSE)
and the Peak Signal-to-Noise Ratio (PSNR). The PSNR reads:
\[
\text{PSNR} = 10 \log_{10} \left( \frac{\max (\mathbf{\hat{m}}_{r,h})^2}{\text{MSE}(f(d_{r,h}; W, b) - \mathbf{\hat{m}}_{r,h})} \right)
\]
(5-10)

Table 5-1 – Summary of the trained CNNs.

<table>
<thead>
<tr>
<th>Trained CNN</th>
<th>Data sort</th>
<th>Tensor input size (offset × crossline × inline × time)</th>
<th>Tensor output size (offset × crossline × inline × time)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNN1</td>
<td>Vertical section (crossline)</td>
<td>1×1×256×480</td>
<td>1×1×256×480</td>
</tr>
<tr>
<td>CNN2</td>
<td>Vertical section (crossline)</td>
<td>5×1×256×480</td>
<td>1×1×256×480</td>
</tr>
<tr>
<td>CNN3</td>
<td>Horizontal section (time-slice)</td>
<td>5×256×256×1</td>
<td>1×256×256×1</td>
</tr>
<tr>
<td>CNN4</td>
<td>3D (inline, crossline, time)</td>
<td>5×128×128×128</td>
<td>1×128×128×128</td>
</tr>
</tbody>
</table>

We observed the largest holes in the data in the nearest offset classes. Therefore, the
training data consisted of examples in the range of the first 10 offset volumes (positive and
negative). We applied mute above the water bottom arrival time to avoid training on data pair
examples where we observed no signal. The training data for the vertical and the horizontal
cross-sections consisted of 10000 examples. 85% of the training data were randomly selected
for the training set and with the remaining ones forming the validation set. The training data
for the 3D approach consisted of 3060 patches. These patches were extracted from the training
data using a sliding window approach with an overlap of one-third. Similarly, 85% of the
training data were randomly selected for the training set, and the remaining ones were left for
the validation set. Figure 5-4 visualizes a single data training pair for the time slice sorted
case. The input data in Figure 5-4a represents a 3D tensor with five consecutive offset classes,
with the first OFFC of 25 m and the fifth OFFC of 225 m. The labeled data in Figure 5-4b is
a 2D tensor that is the desired interpolated and regularized data of the first OFFC of 25 m.
From the same figure, one can notice that large holes are observed at the nearest offsets, which
tend to decrease as the offset increases. Note that the unique position of each trace is not input
to the CNN. It is discarded because each input trace is assumed to be at the center of the bin.
However, the CNN modifies the input traces and aligns them with the newly interpolated traces to form coherent seismic events. This modification is equivalent to regularization. The 2D networks were trained for 100 epochs on a single GPU, while the 3D networks used 2 GPUs to finish 45 epochs. We used the same hyperparameters for all experiments, which we believe to be optimal, where we set the learning rate to \( \eta = 10^{-4} \), the batch size to \( B = 4 \), and the regularization parameter \( \lambda = 10^{-5} \). The corresponding graphs of the loss curves are displayed in Figure 5-5.

Figure 5-6 and Figure 5-7 visualize a crossline and an inline example, respectively taken outside the training area, for the positive offset value of 25 m. Results are shown for all test configurations described in Table 5-1. The inline example in Figure 5-7 corresponds to the case of no recordings for this offset range between two adjacent sail lines (Figure 5-1b); thus, input data do not exist. Analyzing these results indicates an improvement employing the multi-feature approach. This observation is valid for both the inline and crossline examples when comparing CNN1 and CNN2.

However, noticeable ‘jitter’ (misalignment between the events) is visible for CNN1 and CNN2 when inspecting the inline data example in Figure 5-7. The
reason is that these two networks were trained using vertical cross-section data-oriented along the crossline dimension. This jitter indicates that the network has biases towards the training dimension. The horizontal cross-section (time-slice) approach (CNN3) shows further improved coherency for the nearly horizontal-like energy compared to CNN2. On the other hand, a slight reduction of some diffraction-like energy is observed compared to the vertical cross-section training. The 3D training example (CNN4) seems to capture both the horizontal-like and diffraction-like energy well. The extracted time slice from the test area shown in Figure 5-8 supports the above observations. However, weak amplitude dimming is seen even for the 3D network for the areas falling right in the center of the gaps. Figure 5-9 shows the calculated RMSE and PSNR for the presented data examples. All traces and the complete record length are used in these calculations. The plots suggest that the 3D network CNN4 performs the best overall.

![Figure 5-6. Crossline data example for the positive offset value of 25 m employing the different trained networks as shown in Table 5-1](image)

Figure 5-6. Crossline data example for the positive offset value of 25 m employing the different trained networks as shown in Table 5-1

To further evaluate the performance of the trained networks, for the analyzed crossline and inline data examples, the amplitude and $f-k$ plots are displayed in Figure 5-10
and Figure 5-11. These plots suggest that the vertical cross-section training struggled to reconstruct the data properly along the inline dimension and introduced noise visible in the $f$-$k$ spectrum in Figure 5-11. The horizontal cross-section (time-slice) training (CNN3) preserves more of the horizontal-like energy, but it can be seen from the $f$-$k$ plots that some dipping-like energy is not well preserved. Weak amplitude decay is observed for all network predictions at the higher frequencies, with CNN4 closest to the target amplitude spectrum. The 3D network achieved the best performance among these experiments.

![Figure 5-7](image)

Figure 5-7. Inline data example for the positive offset value of 25 m employing the different trained networks as shown in Table 5-1.

### 5.2.7 Discussion and challenges

The presented strategies for CNN learning of non-linear representations for interpolation and regularization of 3D performed adequately, except the vertical cross-section case. Despite the benefit of the multi-feature approach, these networks showed bias to the chosen dimension of training, and therefore, introduced artifacts in the orthogonal dimension during the prediction. The horizontal cross-section approach showed improved performance over the vertical-
section ones. Once sorted back to the $t$-$x$ domain, the field data results demonstrated that the time-slice method preserved the seismic wavefield’s coherency characteristics. In addition, the horizontal cross-section training is not limited to any single spatial dimension, using simple 2D convolutions, the approach combines data from both inline and crossline dimensions. However, we observed improvement when employing a 3D convolutional network at the expense of more costly training.

Figure 5-8. *Time slice at 1224 ms for the positive offset value of 25 m employing the different trained networks as shown in Table 5-1.*

Figure 5-9. *RMSE and PSNR plots calculated for both the inline (in yellow) and the crossline (in purple) data examples. Note that all traces and complete record length are used in these calculations.*
Table 5-2 summarizes the time of the training step for all the presented experiments. Note that more powerful hardware had to be used for the 3D CNN, but still, the computational time increased significantly. The 2D networks were trained for 100 epochs on a single GPU, while the 3D networks used 2 GPUs to finish 45 epochs. The time needed to predict a small cube of data with a size of $1 \times 256 \times 256 \times 480$ (offset $\times$ inline $\times$ crossline $\times$ time) is shown in Table 5-3. Although we do not show the results, the horizontal cross-section and 3D trained CNNs showed a more negligible difference with respect to the target at larger offsets and after

Figure 5-10. Amplitude and f-k spectra of the presented crossline data example. All traces and complete record length are used in these calculations.

Figure 5-11. Amplitude and f-k spectra of the presented inline data example. All traces and complete record length are used in these calculations.
Following the field data examples, we can see that when we have large holes in the data (e.g., near 3D offset classes), the target results are more accurate in terms of reconstructed diffraction-like structures than the DNN methods but are more time-consuming. On the other hand, the computational cost for the ALFT method depends on the number of wavenumbers, the number of spatial dimensions, and the number of input traces in a given processing block (Xu et al., 2010). Some methods to accelerate the ALFT algorithm are also available (Whiteside et al., 2014; Jahanjooy et al., 2016). In the case of DNN, the computational cost is more related to time spent on network training. The actual cost for interpolation and regularization employing a DNN approach is low. For large-size surveys, the ALFT can be utilized for selected parts of the survey area first, and the remaining amount is left to the DNN method. In such a case, the DNN methods can speed up the interpolation and regularization step of the processing workflow at the expense of reduced accuracy. However, since the presented DNN is dependent on ALFT, the time required for proper parameterization of ALFT should also be included.

Table 5-2 – Training times.

<table>
<thead>
<tr>
<th>Trained CNN</th>
<th>Epochs</th>
<th>Training completed in (hh:mm:ss)</th>
<th>NVIDIA GPU hardware used</th>
<th>Number of GPUs used</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNN1</td>
<td>100</td>
<td>17:00:00</td>
<td>RTX2080</td>
<td>1</td>
</tr>
<tr>
<td>CNN2</td>
<td>100</td>
<td>15:15:00</td>
<td>RTX2080</td>
<td>1</td>
</tr>
<tr>
<td>CNN3</td>
<td>100</td>
<td>08:20:00</td>
<td>RTX2080</td>
<td>1</td>
</tr>
<tr>
<td>CNN4</td>
<td>45</td>
<td>100:03:00</td>
<td>Quadro RTX6000</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 5-3 – Prediction times for a cube of seismic data with a size of 1 × 256 × 256 × 480 samples (offset × inline × crossline × time).

<table>
<thead>
<tr>
<th>Method</th>
<th>NVIDIA GPU hardware used</th>
<th>Number of GPUs used</th>
<th>Prediction completed in (hh:mm:ss)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNN1</td>
<td>RTX2080</td>
<td>4</td>
<td>00:00:27</td>
</tr>
<tr>
<td>CNN2</td>
<td>RTX2080</td>
<td>4</td>
<td>00:00:35</td>
</tr>
<tr>
<td>CNN3</td>
<td>RTX2080</td>
<td>4</td>
<td>00:00:42</td>
</tr>
<tr>
<td>CNN4</td>
<td>RTX2080</td>
<td>4</td>
<td>00:02:33</td>
</tr>
<tr>
<td>ALFT</td>
<td>RTX2080</td>
<td>4</td>
<td>00:10:03</td>
</tr>
</tbody>
</table>

As in every other supervised learning problem, the availability of labeled or target data (i.e., fully regularly sampled images in our case) remains a challenge. The presented DNN methods utilized a subset of the available ALFT data to overcome this issue. The challenge with such an approach is that the quality of the target data will limit the network
performance. Moreover, we also observed that in some of the examples, the DNN approaches struggled to reconstruct steep dipping events with some high-frequency leakage. The deep part in the same crossline example, together with the extracted time slice, showed some weak amplitude dimming for the reconstructed data at the center of the missing data holes. A possible solution to improve the reconstruction of diffraction-like events is to include more training examples with such patterns. Another approach might be to develop a new network structure or, for example, to take advantage of recent developments in the so-called transformer-based networks (Chen et al., 2020) to better deal with the complex nature of seismic data. An alternative way of obtaining target data is to follow unsupervised learning (Kong et al., 2020; Greiner et al., 2021), which aims to identify patterns and relationships in a data set without labeled data. In any case, obtaining relevant and high-quality training data for the interpolation and regularization of offset classes remains challenging. The following subchapter, i.e., Paper III, aims to solve the problem of training data availability. The method proposes a de-migration of the PSDM stacked image into two configurations for each offset class. The first configuration follows the original survey consisting of the recorded source/receiver positions. The second configuration is an idealistic survey with constant offset and azimuth for each offset class volume. We propose to train a 3D convolutional encoder-decoder to map each offset class from configuration one to two and use this trained model on the recorded data. This technique of obtaining tailored synthesized data could be employed for other steps of the processing workflow after proper modifications.
5.3 Paper III – De-migration-based supervised learning for interpolation and regularization of 3D offset classes

Hlebnikov, V., T. L. A. Greiner, V. Vinje, J. E. Lie, and G. Poole

Geophysical Prospecting, Resubmitted after minor revision in January 2022.
ABSTRACT

Regularization and interpolation of 3D offset classes prior to imaging is an important and challenging step in the marine seismic data processing flow. Here we describe how to perform this task using a deep neural network (DNN), and we explain how to overcome the challenge of creating a suitable training dataset. The training dataset is generated by de-migrating stacked pre-stack depth migration (PSDM) images. For each offset class volume, we de-migrate the PSDM stacked image into two configurations; (i) the original survey configuration consisting of the recorded source/receiver positions and (ii) an “Ideal“ survey configuration with constant offset and azimuth for each 3D offset class. The training creates a 3D convolutional encoder-decoder (CED) model that will regularize and interpolate seismic data. The CED is trained on 3D sliding-windows in each 3D offset cube to map from (i) to (ii), i.e., to map the original survey configuration with irregular and sparse sampling into the fully sampled regular offset cubes suitable for offset-based migration, such as Kirchhoff migration. Such migration algorithms rely on regular and sufficiently dense sampling to achieve constructive interference to image the structures and destructive interference to suppress migration noise. We test the new method on one synthetic and one field data example and show that it performs better than a standard regularization/interpolation method based on anti-leakage Fourier Transform, especially for the smallest offset classes. On the synthetic data, we also demonstrate that the CED method preserves the amplitude versus offset (AVO) as well as the standard method.

INTRODUCTION

All 3D marine acquisition geometries suffer from poor sampling along with one or more spatial directions. In addition, sea currents and navigation challenges make it difficult to put the sources and receivers in their planned locations. This is a problem for migration algorithms based on regular and sufficiently dense sampling to achieve constructive interference to image the structures and destructive interference to suppress migration noise (Abma et al, 2007). Therefore, it is common in seismic data processing to employ interpolation and regularization of data prior to migration. Conceptually, the interpolation process fills in the missing data and the regularization maps the data from the irregularly recorded locations to common midpoint (CMP) locations on a regular grid. For migration methods using constant offset volumes (or offset classes) as input (Sattlegger et al., 1980;
Deregowski, 1990, Ehinger et al., 1996), the regularization/interpolation forms an important part of the processing workflow. An offset class (OFC) is defined as a 3D volume being a subset of the seismic data within a certain offset range, the range typically defined by twice the distance between the shots along a shot line.

Interpolation and regularization issues are not new; various methods exist to tackle them either independently or at the same time. Some examples are: prediction filtering (Naghizadeh and Sacchi, 2009; Liu and Fomel, 2011), rank-reduction methods (Trickett et al., 2010; Gao et al., 2015), tensor completion techniques (Kreimer et al., 2013), minimum-weighted norm interpolation (MWNI) (Liu and Sacchi, 2004; Zwartjes and Sacchi, 2007), anti-leakage Fourier transform (ALFT) (Xu et al., 2005, 2010), other transform-based methods like Curvelet (Naghizadeh and Sacchi, 2010), Seislet (Gan et al., 2015) and Radon (Trad et al., 2002; Wang et al., 2010; Ibrahim et al., 2015), data-driven methods based on dictionary learning (Yarman et al., 2017; Zhu et al., 2017; Turquais et al., 2018), and techniques based on the use of the wavefield attributes (Höcht et al., 2009; Xie and Gajewski, 2017; Zhao et al., 2020). Machine learning (ML) is an alternative data-driven way of interpolating seismic data (Jia and Ma, 2017); in this article, we will focus on deep learning (DL), which represents a broad family of ML methods based on artificial neural networks (ANNs), where nonlinear feature representations and patterns can be automatically learnt from the use of training data (LeCun et al., 2015).

Convolutional neural networks (CNNs) are among the most used DL architectures in image processing, and they have gained attention within the geophysical community. A common problem shared by these methods is the selection of data for the training process. For interpolation purposes, some CNN methods require fully sampled data (Siahkoohi et al., 2018; Mandelli et al., 2019), which are commonly unavailable for field data. Some methods proposed to exploit similarities between the acquisition domains, for instance, the common shot gather and the common receiver gather for interpolation purposes (Siahkoohi et al., 2018; Park et al., 2019; Wang et al., 2019). Alternatively, Greiner et al. (2020) proposed training from the densely sampled direction on shot gathers intending to interpolate data in the orthogonal direction, i.e., across the streamers, Chai et al. (2020) used a combination of training on both synthetic and real data examples, and Wang et al. (2020) have shown a transfer learning approach using synthetic data as initialization for training on real shot gathers. Recently, Qu et al. (2021) demonstrated the potential of a DL model trained exclusively on
synthetic data for shallow data reconstruction prior to de-multiple. Other studies have shown promising results applying unsupervised DL to the interpolation problem in 2D (Hu et al., 2019; Shi et al., 2020), 3D (Kong et al., 2020), and in 4D (Greiner et al., 2021). Kong et al. (2020) utilize the deep image prior (DIP) (Ulyanov et al., 2018) approach to interpolate missing data on 3D shot-gathers. However, the DIP approach suffers from high computational costs due to reparameterization for every new prediction. Greiner et al. (2021) proposed an unsupervised approach, training an overcomplete convolutional autoencoder including a combined first- and second-order total-variation regularization to predict missing traces in 4D with state-of-the-art results compared to industry workflows. In many circumstances, representative training data are challenging to obtain.

The focus of our work is to build a representative data set, synthesized from recorded data, that will be utilized in the supervised training of a DL model in order to improve on the interpolation and regularization application for binned 3D offset classes. This will demonstrate the use of CNNs to reconstruct sparsely sampled data. Our ultimate goal is to improve the quality of the 3D near offset classes compared to conventional methods. This paper is organized as follows. First, we briefly define the methodology used for training data generation. Then we present the CNN architecture and further extend the methodology. The approach is then tested on synthetic data and real data. Finally, we discuss our findings and provide a set of conclusions.

**METHODOLOGY**

**Workflow overview**

Migration and de-migration are approximately adjoint processes (Santos et al., 2000), and they have been used by Li et al. (2020) for dictionary learning and inversion-based denoising. Here, we propose to use a migration/de-migration routine to train a CNN for interpolation and regularization of sparsely and irregularly sampled OFCs. A flow chart of the workflow can be seen in Figure 1. Consider an OFC sorted into a binned 3D cube, where its corresponding spatial CMP locations are shown in Figure 1 (top left). An example of a nominal acquisition layout that will produce such a sparsely sampled near OFC data set is shown in Figure 2a. In reality, it is difficult to place the sources and receivers in their planned locations due to the dynamic marine environment; thus, the observed irregular pattern of the recorded CMP locations are shown in Figure 1 (top left). An example of a nominal acquisition
layout that will produce such a sparsely sampled near OFC data set is shown in Figure 2a. In reality, it is difficult to place the sources and receivers in their planned locations due to the dynamic marine environment; thus, the observed irregular pattern of the recorded CMP locations in Figure 1 (top left). Interpolation and regularization aim to map the original survey configuration with sparse and irregular sampling into the regularly sampled offset cubes suitable for offset-based migration, such as Kirchhoff migration, because such data will produce a high-quality Kirchhoff PSDM image in the absence of velocity errors. Such approximated data can be obtained, for example, by the ALFT method (Xu et al., 2005, 2010), and its densely-sampled, regular CMP locations are shown in Figure 1 (top right). These data have constant offset and azimuth. An example of an acquisition layout that will produce such a data set with one trace per bin per offset with a constant azimuth is shown in Figure 2b. In the following, we will

Figure 1 – Flow chart diagram of the synthesized driven training. We start from top left with a birds-eye view of a binned sparsely sampled near OFC with its approximated Ideal version (top right). The stacked PSDM is then used to de-migrate into two configurations; the original survey configuration consisting of the recorded source/receiver positions (bottom left) and (ii) an Ideal survey configuration with constant offset and azimuth (bottom right), which are used as input and target to train our CED.
refer to this acquisition layout as “Ideal”. We then stack the migrated OFCs to reduce the migration noise caused by the imperfect interpolation of the sparse near OFCs. De-migration is done using the partial stacked PSDM image limited to near offsets, where most of the acquisition gaps occur, and the migration velocity to generate two sets of data for a given OFC, each with a different acquisition layout. The first de-migrated data set is sparsely sampled, following the actual acquisition spatial sampling (Figure 1 bottom left), while the second de-migrated data set has an Ideal geometry (Figure 1 bottom right). We propose using the sparsely sampled and the Ideal de-migrated OFCs as input/target pairs to train our CNN. The synthesized training data will be similar to the recorded data, as can be seen in Figure 1, a comparison between the recorded data and the sparsely sampled synthesized data set (de-migration following the recorded acquisition) for a given inline. Apart from a difference in

Figure 2 – An example acquisition layout for a real and an Ideal survey. a) – an example survey nominal geometry with 16 streamers and 6 sources, where the near OFCs are sparsely sampled. b) – survey geometry for a fully sampled data set, where each bin center is modelled with a constant offset and azimuth. Note that the plots are not up to scale.

Figure 3 – Inline comparison between the recorded sparsely sampled data in a) and the de-migrated (synthesized) sparsely sampled data following the recorded acquisition in b) for the OFC of 337.5 m.
the noise level and some weaker diffraction tails for the de-migrated data, the two data sets are similar, encouraging the possibility that DL-based methods using networks trained on synthesized data can be employed on recorded data.

The proposed training strategy for regularization/interpolation still depends implicitly on the conventional regularization/interpolation, because a subset (e.g., a range of near offsets) of the available data needs to be processed all the way up to migration to create a stacked PSDM image model for de-migration.

Workflow for training data generation

Consider an OFC sorted into a binned 3D cube \(d_{r,h}\), where \(r\) stands for recorded and \(h\) denotes the offset class index, where \(d_{r,1}\) is the first (smallest) offset class, \(d_{r,2}\) is the next one etc. \(d_{r,h}\) is a three-dimensional function \(d_{r,h} = d_{r,h}(x,y,t)\), where \((x,y)\) denotes the spatial coordinates of the bin centers and \(t\) denotes time. In the binning process, each recorded trace is mapped into the bin cell in \(d_{r,h}\) closest to the CMP position of the trace. The result is a series of OFCs with irregularly located traces and some empty bins, as shown in the upper left part of Figure 1.

Now, suppose that \(n_h\) denotes the number of OFCs, where each OFC has \(n_x\) crosslines, \(n_y\) inlines, and \(n_t\) temporal samples. Then the discrete samples of \(d_{r,h}(x,y,t)\) form a 3D tensor, which defines the set of OFCs as \(\{d_{r,h}\}_{h=1}^{n_h}\). For notational convenience, let the recorded data \(d_{r,h} \in \mathbb{R}^n\) represent a vector according to its lexicographic order of size \(n \times 1\), where \(n = n_x n_y n_t\).

In principle, we would like to solve the following inverse problem:

\[
d_{r,h} = Mm_{r,h} + \varepsilon_{r,h},
\]

where \(d_{r,h} \in \mathbb{R}^n\) is the sparsely sampled recorded data, \(M: \mathbb{R}^n \rightarrow \mathbb{R}^n\) denotes the mapping operator, \(m_{r,h} \in \mathbb{R}^n\) denotes the Ideal data and \(\varepsilon_{r,h}\) is the observation noise. Now, assume that there exists a nonlinear functional representation of the Ideal data that takes the sparsely sampled data as input, i.e.,

\[
d_{r,h} = Mf(d_{r,h}; W, b) + \varepsilon_{r,h},
\]
where $f(\cdot; \mathbf{W}, \mathbf{b})$ denotes an arbitrary neural network, where the network is parameterized by $\mathbf{W}$ and $\mathbf{b}$ denoting the convolutional kernels and biases, respectively. Our goal is then to find an approximation of $f(\cdot; \mathbf{W}, \mathbf{b})$. In our case, $f(\cdot; \mathbf{W}, \mathbf{b})$ is trained exclusively on synthesized data, in which case the solution to Equation 1 takes the form

$$\tilde{\mathbf{m}}_{r,h} = f(d_{r,h}; \hat{\mathbf{W}}_s, \hat{\mathbf{b}}_s),$$

(3)

where $f(\cdot; \hat{\mathbf{W}}_s, \hat{\mathbf{b}}_s)$ can be considered a generalization of the approximate inverse of $\mathbf{M}$, and $\hat{\mathbf{W}}_s$ and $\hat{\mathbf{b}}_s$ denote the trained parameters with subscripts $s$ implying synthesized data. In this case, we assume that $f(\cdot)$ does not depend strictly on $d_{r,h}$ and $\mathbf{m}_{r,h}$, but merely depends on the structural pattern and frequency content of the seismic data it has been trained on. Hence, we hypothesize that a suitable approximation to $f(\cdot)$ can be found by utilizing training data that show similar characteristics to the recorded data $d_{r,h}$ and $\mathbf{m}_{r,h}$, i.e., data that contain optimally all or most of the features that the complete data $\mathbf{m}_{r,h}$ possess.

To construct such a training data set, we first create an approximation of the subsurface reflectivity $\mathbf{R}$ from the recorded data and apply de-migration to $\mathbf{R}$. In our case, the construction of $\mathbf{R}$ was done by utilizing conventional processing on a subset of the recorded data. The reflectivity data were constructed through the formula

$$\mathbf{R} = \frac{1}{n_h} \sum_{h=1}^{n_h} \mathcal{G}_h(\mathcal{A}_h(d_{r,h})), \quad (4)$$

where $\mathcal{A}_h$ represents the ALFT method (Xu et al., 2005, 2010) used to interpolate and regularize the recorded data $d_{r,h}$, $\mathcal{G}_h$ is a migration operator and the summation represents the stacked response with a normalization factor of $1/n_h$, where $n_h$ denotes the total number of OFCs used to create the reflectivity. We then create Ideal OFCs by de-migration:

$$\mathbf{m}_{s,h} = \mathcal{G}_h^\dagger \mathbf{R}, \text{ for } h = 1, \ldots, n_h, \quad (5)$$

where $\mathcal{G}_h^\dagger$ represents a de-migration operator. In this case, the $\mathbf{m}_{s,h}$ inherits a similar feature space as the unknown recorded OFC $\mathbf{m}_{r,h}$. Then, by utilizing the synthesized data $\mathbf{m}_{s,h}$, we can solve the following inverse problem instead of Equation 1:
\[ d_{s,h} = Mm_{s,h} + \varepsilon_{s,h}, \]  

where \( M \) is the mapping operator extracted from Equation (1) and \( d_{s,h} \) denotes the sparsely sampled synthesized data, and \( \varepsilon_{s,h} \) represents noise arising from the imperfect reflectivity model used for de-migration. Compared to Equation 1, we now have access to a representative Ideal target data \( m_{s,h} \), to be used to train our supervised deep learning model. To learn the parameters \( W_s \) and \( b_s \), we minimize a misfit functional in a least-squares sense including a regularization penalty term, given by following objective function

\[
\min_{W_s, b_s} \left\{ \frac{1}{2} \| m_{s,h} - f(d_{s,h} + e; W_s, b_s) \|_2^2 + \frac{\lambda}{2} \| W_s \|^2_2 \right\}, \tag{7}
\]

where \( \| \cdot \|^2_2 \) denotes the L2-norm, \( \lambda \) is the regulariztion parameter defining the trade-off between the misfit and the penalty term, and \( e \sim N(\mu, \sigma) \) is a noise variable that performs stochastic corruption of the input by introducing additive Gaussian noise with varying mean \( \mu \) and variance \( \sigma \). Such stochastic corruption of the input is commonly referred to as jittering in machine learning research. It has shown to be a crucial factor for improving the generalization, stability, and accuracy in deep learning for inverse problems (Genzel et al., 2020).

Minimizing expression (7) is achieved by solving an optimization problem. First, the sparsely sampled and the Ideal data are divided into smaller patches (mini cubes), with \( T \) as the number of training examples (or mini cubes) \( \{(m_{s,h}^{(i)}, d_{s,h}^{(i)})\}_{i=1}^{T} \). These patches are then grouped into randomized batches of size \( B \), such that the input to the optimization algorithm consists of mini batches of randomized training examples. In our case, we employed a first-order stochastic gradient descent algorithm with mini batches (subsamples), known as decoupled weight decay (Loshchilov and Hutter, 2017), which is a weight decay version of the Adam optimization (Kingma and Ba, 2014). The decoupled weight decay algorithm for updating the weights reads:

\[
W_n = W_{n-1} - \eta \left( \frac{1}{B} \sum_{i=1}^{B} (\nabla_W C(W, b)_{n-1}) + \nabla_W R(W)_{n-1} \right), \tag{8}
\]
where $\eta$ is the learning rate, $n$ is the iteration, $\nabla_W C(W, b)_{n-1}$ is the partial derivatives of the misfit function with respect to the weights using the Adam update rule (Kingma and Ba, 2014), $\nabla_W R(W)_{n-1}$ are the partial derivatives of the regularization term, and $B$ denotes the batch size. Updating the bias parameters are done in a similar manner, but without the regularization term.

**CNN architecture**

The convolutional neural network (CNN) is a special type of neural network architecture. Compared to dense neural networks, which have a full connection between the weight matrices and input vectors from one layer to the next, the CNN architecture instead constrains the linear operations to convolutions. The majority of the architectures widely used in deep learning and seismics use a sort of encoder-decoder architecture. The encoding part tries to extract an abstract representation of the input data at a latent space at a lower-dimensional Hilbert space. According to Goodfellow et al. (2016), this latent space is undercomplete.

On the other hand, the decoder tries to learn how to take the latent low-dimensional representation back to the original high-dimensional output. As a consequence of the undercompleteness, such encoder-decoders may lose valuable information. Some architectures include residual layers, like the U-Net (Ronneberger et al., 2015), to compensate such limitations.

The CNN we employ in our study comprises an overcomplete 3D convolutional encoder-decoder (CED) architecture, where an OFC denotes an input feature space. In such a way, the 3D convolutional kernels can “see” information from the higher offset classes, which makes the method 4D. Overcomplete CED networks are typically built as expansive (wide) architectures in combination with a down-sampling of features. In contrast to undercomplete networks, the latent representation of an overcomplete CED lies on a higher-dimensional Hilbert space with respect to its input. Expansive networks have been shown to yield good generalization ability (Han et al., 2017) and have the capacity to provide perfect reconstruction. The capacity of overcomplete networks makes them susceptible to overfitting, which we reduce by employing an $L_2$-norm regularization penalty on the convolutional kernels. In addition, to avoid low-pass filtering from down-sampling operators such as max- or mean-pooling, we employ strided convolutions (down-sampling) and transposed
convolutions (up-sampling), in which the down-sampling and the up-sampling are learned instead of being fixed operators.

The 3D CED employed in our studies is illustrated in Figure 4. The same figure shows that the CED is made of two encoder blocks, a latent layer, two decoder blocks, and an output layer. The input tensor has a size of $96 \times 96 \times 96 \times m_h$ and the output size is $96 \times 96 \times 96 \times 1$, hence, the neural network $f(\cdot) : \mathbb{R}^m \rightarrow \mathbb{R}^n$ maps $m_h$ number of OFCs to a single OFC, where $m = n_x n_y n_z m_h$. We found that setting $m_h = 5$ was a good fit in our experiments. Each of the encoder blocks consists of two cascaded 3D convolutions. For example, the first 3D convolution of the first encoder block has a dimensionality of $7 \times 7 \times 7 \times 5 \times 80$, where $7 \times 7 \times 7$ denotes the 3D convolutional kernel size, $5$ denotes the number of input features, that is $5$ consecutive OFCs, and $80$ is the number of output features. Similar to Dong et al. (2015) and Greiner et al. (2020), we found that using larger kernel sizes at the start of the network improved the results. To introduce nonlinearity, the output of that convolution is passed through an activation function. In our case, the activation function is given by the rectified linear unit or ReLU. A second 3D convolution with activation function then follows that is within the same encoder block with the dimensionality of $5 \times 5 \times 5 \times 80 \times 96$. The output size of each encoder block is reduced by a factor of 8, since the 3D stride of the first convolution in each encoder block is equal to $2 \times 2 \times 2$. To ensure overcompleteness, that requires an increase of the number of features by at least a factor of 8. Once the latent layer of the CED is reached, where a single convolutional operation is performed, we start increasing the size of the image by a factor of 8 using transposed convolutions in each of the decoder blocks. At the final layer, a 3D convolution with a kernel size of $1 \times 1 \times 1$ without nonlinear activation is used to map each 80-component feature map to the desired single output image, which is the first of the 5 OFCs used on input. The network depth (number of layers) and the input data sizes were found by trial and error. However, the design was also driven by ensuring few down-sampling operations that fit the available hardware memory.

Note that the unique position of each trace is not input to the CED so that, in a sense, each input trace is assumed to be at the center of each bin. However, the CED modifies the input traces and aligns them with the newly interpolated traces to form coherent seismic events. This modification is equivalent to regularization. Therefore, once the prediction of the
OFCs is completed, we update the trace locations to the corresponding bin centers. That is why we call our method a CED-based interpolation and regularization.

---

Figure 4 – *Our example CED for interpolation and regularization. CED’s input is a fourth-order tensor on the left with size of $96 \times 96 \times 96 \times 5$. The network consists of two encoder blocks, a latent layer, two decoder blocks, followed by an output layer without nonlinearity. The corresponding target tensor on the right has size of $96 \times 96 \times 96 \times 1$. The input/output tensor dimensions correspond to inline (IL), crossline (XL), time, and offset class (OFC).*

SYNTHETIC AND REAL DATA EXAMPLES

We tested the performance of the CED-based interpolation and regularization of 3D OFCs on a 3D synthetic data set and a real data set. Real data from a marine wide-towed penta-source-over-streamer (*Poole et al., 2020; Salaun et al., 2020*) survey is used for demonstration purposes. Even though this source-over-streamer setup enabled the acquisition of short offsets, the ultra-near OFCs were still sparsely sampled. The wide source tow setup reduced the empty gaps in coverage between the sail lines, but at the expense of strong azimuth changes in the crossline direction as varying source-streamer trace contributions provided coverage. The combination of the survey layout and the rugose hard water bottom in the Barents Sea led to a challenging and complex wavefield to be reconstructed. Because of the above-mentioned reasons, the synthetic data set that follows were modelled with a similar wide-tow configuration.
Synthetic data example

To evaluate the performance of our CED approach, we designed a synthetic 3D model. We applied a diffraction modelling-based method similar to the one described by Jaramillo and Bleistein (1999) to generate synthetic broadband data. The model consisted of a rugose water bottom from bathymetry measurements in the Barents Sea (Figure 5). Such a water bottom gave a complex data set with severe aliasing and multiple events with conflicting dips, challenging any interpolation algorithm. The source wavelet had a zero-phased flat frequency spectrum from 4 to 180 Hz. We assumed no external noise, constant velocity in the water layer of 1480 m/s, and no anelastic attenuation.

![Figure 5 – Barents Sea water bottom bathymetry used in the modelling. The water depth varies from 320 to 360 m. The glacial scour marks give a complex data set with strong aliasing and conflicting dips, which is a challenge to any interpolation algorithm.](image)

We used the Ideal survey layout (Figure 2b) to model 10 constant offset, zero-azimuth data classes \( m_{r,h} \) on a 6.25 m × 6.25 m grid. The bin center locations of each of the 3D constant OFCs ranged from 37.5 m to 712.5 m with an increment of 75 m. The size of each OFC was approximately 4 × 13 km with 750-time samples using a 2 ms temporal sampling. We then modeled a new sparsely sampled data set \( d_{r,h} \), in which we had the source-over-streamer (Vinje et al., 2017) survey layout shown in Figure 2a. This layout gave sparsely populated near OFCs with varying offset and azimuth. Our training methodology implies that the input data to our network are sorted to 3D OFCs with static binning applied, i.e., a maximum of one trace per bin was selected in each OFC based on the minimum distance to the bin center. A subset of the study area of that sparsely sampled data was then passed through the proposed workflow (Figure 1) to generate the de-migrated sparsely sampled and Ideal OFCs, \( d_{s,h} \) and \( m_{s,h} \), respectively.
As shown in Figure 4, our CED expects tensors of size $96 \times 96 \times 96 \times 5$. To extract the mini cubes out of the modelled data sets, we used a sliding window approach with an overlap of one-third between the windows. This led to a total number of 1080 mini cubes, where 20 percent were randomly selected as a validation data set, and the remaining were used as a training data set. The CED was then trained for 60 epochs. Figure 6 displays the loss during the training from the training set (black curve) and the validation set (red curve).

From the modelled Ideal data example for the nearest OFC of 37.5 m in Figure 7b, we observe that the rugose water bottom created complex data with diffraction-like events causing strong interference. The modelled sparsely sampled OFC in Figure 7a shows the challenging data to be reconstructed. Figures 7d and 7f demonstrate the pre-migration reconstruction for both the ALFT and the CED-based methods. The yellow arrows in the figure highlight some of the differences between the two methods, and we can note an improved reconstruction of the diffraction-like events with CED. These improved diffractions contributed to a better coherency of the water bottom reflection than the ALFT method after migration, as shown in Figures 7e and 7g. This visual inspection also suggests that the noise levels for both ALFT and CED are comparable. Finally, the estimated amplitude spectra for the three data sets after migration displayed in Figure 7h indicate that the ALFT and the CED followed the shape of the modelled and Ideal data set closely, with a noticeable sharper cut-off for the CED at approximately 175 Hz.

One of the main concerns of the proposed method is AVO fidelity, which is lost during the stacking process. To test the AVO fidelity of our method, we performed a careful analysis of the synthetic data, which were modelled with negative AVO gradient (Shuey, 1985), resulting in amplitude decay with offset.
To test the AVO fidelity of the method, we used the synthetic data example presented above and generated sub-stacks for the whole study area. Each individual sub-stack consisted of the stacked response of 3 consecutive migrated OFCs. There was no overlap of offset among the sub-stacks; we refer to these sub-stacks as the near, mid, and far. We then picked the peak amplitude along the water bottom for all sub-stacks and for the Ideal, ALFT, and

![Figure 7 – Synthetic crossline data example for OFC of 37.5 m. a) – Modelled sparsely sampled; b) – Modelled Ideal and its corresponding migration in c); d) – the ALFT reconstructed and its corresponding migration in e); f) – the CED reconstructed and its corresponding migration in g); h) Amplitude spectrum of the migrated results, normalized to the largest peak.](image)

To test the AVO fidelity of the method, we used the synthetic data example presented above and generated sub-stacks for the whole study area. Each individual sub-stack consisted of the stacked response of 3 consecutive migrated OFCs. There was no overlap of offset among the sub-stacks; we refer to these sub-stacks as the near, mid, and far. We then picked the peak amplitude along the water bottom for all sub-stacks and for the Ideal, ALFT, and
CED datasets. Figure 8 shows the extraction of that amplitude picking for the near stack. We notice that, even for the modelled Ideal data in Figure 8a, an imprint of the scour marks is still visible due to the imperfect migration. These scour marks are more apparent for the ALFT and the CED examples in Figures 8b and 8c, where the histogram of the CED appears narrower than that of the ALFT. We also calculated the mean and the standard deviation of the picked peak amplitude for all sub-stacks. These values for the whole study area are displayed in Figure 9, where we observe that the AVO fidelity of the proposed approach is comparable with that of the ALFT method. The corresponding standard deviations suggest more clustering around the mean value for the CED method than the ALFT; this means that the CED is on average closer to the true reflection coefficient than the ALFT in this synthetic model.

As shown in Figure 7, the improved reconstruction of diffraction events obtained by the proposed method led to a better coherency of the water bottom reflection after migration. To evaluate the noise level, we calculated Normalized Root Mean Square Error (NRMSE) between the Ideal data set and
the two reconstruction methods. The estimation was done within a $+/-80$ ms window around the modelled water bottom horizon. We calculated the NRSME for the 3 sub-stacks over the whole study area using the following equation:

$$\text{NRMSE} = \frac{\text{MSE}(m_{r,h} - \hat{m}_{r,h})}{\max(\text{abs}(m_{r,h}))},$$

where the MSE stands for Mean Square Error. The calculated values are shown in Figure 10. As seen from this figure, the CED NRMSE values are lower overall, which can be explained by the combination of the improved water bottom imaging and less migration noise around the modelled horizon.

Figure 9 – Mean and standard deviation of the peak amplitude along the modelled water bottom horizon for the simulated near, mid, and far sub-stacks.

Figure 10 – NRMSE estimated in a window around the modelled water bottom horizon for the simulated near, mid, and far sub-stacks.

**Real data example**

Our real data example is from the Barents Sea with a natural bin size for the configuration of $6.25 \times 6.25$ m. The data were processed through sail-line processing with a temporal sampling interval of 2 ms. With a source over streamer acquisition, negative offsets are recorded, and therefore the offset center of each of the 3D OFCs started from $-4087.5$...
m to 4087.5 m with an increment of 75 m. However, since our focus was mainly on improving the reconstruction of the sparsely sampled near OFCs, we concentrated on the first 9 positive OFCs only. The study area was approximately 10 km × 10 km. From that area, we selected a subset of the data, which were used for the training data generation and fed into the training workflow (Figure 1). After slicing through the de-migrated sparse and Ideal data sets, in a similar approach to the synthetic example, i.e., a sliding window with an overlap of one-third, we ended up with 3600 data training pairs. 20 percent of these were randomly selected as a validation data set, and the remaining were used as a training data set. The CED was then trained for 60 epochs. We then inferred the regularized data on the full study area from the trained model. Figure 11 displays the loss during the training from the training set (black curve) and the validation set (red curve).

Figure 12a shows a crossline from the recorded sparsely sampled OFC for a 37.5m offset. Both the ALFT and the CED-based methods shown in Figures 12b and 12c successfully managed to reconstruct this challenging data set, with slightly different characteristics. As observed for the synthetics example, the CED better reconstructs the diffraction-like events and reduces the noise content than the ALFT reconstruction. The amplitude spectrum in Figure 12d for the two methods are very similar, with the CED-based reconstruction showing an amplitude decrease on the high frequencies starting from approximately 170 Hz.

To compare the two different methods’ abilities to reproduce and preserve the geological features and patterns of the data, we migrated the ALFT and the CED reconstructions, using a Kirchhoff depth migration. Figure 13 shows these results stretched back to the time domain. As expected, the improved reconstruction of diffraction-like events in the CED approach led to enhanced coherency not only for the water bottom but also for deeper events. Overall, less migration noise is visible for the CED, and an improved
coherency is seen for most of the events on the crossline. The extracted time slices in Figure 14 corresponding to $t = 600 \text{ ms}$ and $900 \text{ ms}$ support these observations, where we see improved scour mark patterns on $t = 600 \text{ ms}$ and fault definition on $t = 900 \text{ ms}$ implying a promising 3D wavefield reconstruction from the proposed method.

![Real crossline data example for OFC of 37.5 m. a) – Sparsely sampled; b) – the ALFT reconstruction; c) – the proposed CED reconstruction; d) – Amplitude spectrum, normalized to the largest peak.](image-url)
At larger offsets and after stacking near OFCs, we observed more similarity between the ALFT and the CED-based methods than what was observed on the near section. Figures 15 and 16 compare crossline from a near stack data and time slices corresponding to \( t = 900 \) ms. The CED shows sharper fault patterns and lower noise levels and is still better, even after stack.

**DISCUSSION AND CHALLENGES**

The proposed CED-based approach works on NMO-corrected data (as does the ALFT method), and thus depends on some prior knowledge of the subsurface velocities. An inaccurate velocity field will not correctly precondition the data, and the offset-to-offset coherency of the seismic events will be violated, which may lead to a decreased accuracy of the method. A possible correction to this problem is to perform data augmentation by applying perturbations to the velocity field, such that the CED have seen both correct and incorrect NMO-corrected data examples, which may improve the generalization performance of the model. We have not investigated to what degree the velocity inaccuracies are tolerated by the
CED, but we can conclude that if stacking velocities are available, the method holds, especially when limited to near offsets.

Another potential challenge is the presence of noise that will increase the uncertainty of the proposed method. Seismic noise is never truly random but exhibits specific patterns (Hlebnikov et al., 2021). Therefore, in the presence of a low signal-to-noise ratio, the results can be biased by such noise patterns and compromise the quality of the interpolated primary

![Figure 14 - Real time slice data examples for OFC of 37.5 m after Kirchhoff depth migration stretched back to the time domain. a) and c) the ALFT reconstruction for t = 600 and 900 ms, respectively; b) and d) the proposed CED reconstruction for t = 600 and 900 ms, respectively.](image)
events. Pre-processing with the aim of noise attenuation (as in our real data example) is, therefore, an option.

Figure 15 – Real crossline stack data example after Kirchhoff depth migration stretched back to the time domain. a) – the ALFT reconstruction; b) – the proposed CED reconstruction.

Figure 16 – Real time slice stack data examples corresponding to $t = 900$ ms after Kirchhoff depth migration stretched back to the time domain. a) – the ALFT reconstruction; b) – the proposed CED reconstruction.
We have not analyzed any 4D fidelity of the proposed work, which will be part of future research. However, we believe that the method will hold based on the AVO analysis presented in the synthetic data example.

Even though the proposed approach does not try to mimic an existing conventional reconstruction method, our approach makes the CED implicitly dependent on an initial interpolation method, such as ALFT. We used a subset of the available data processed all the way up to migration to create a reflectivity model, which was then used for the de-migration step. Therefore, proper parameterization of the ALFT is an important consideration. However, once the network is trained, the complete wavefield is reconstructed automatically with no further need for the initial interpolation method.

One of our main goals was to achieve a better quality of the 3D near offset classes compared to conventional methods. However, the compute (GPU/CPU) efficiency is also important. Table 4 shows compute times based on the synthetic test with the following normalized times per unit survey area: (i) the time for the ALFT, (ii) the time for training the CED, including generating the PSDM images, the de-migrations, and the training process, and (iii) time per unit area of CED prediction. All the compute times are normalized to that of ALFT.

<table>
<thead>
<tr>
<th>Method</th>
<th>Normalized times</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALFT</td>
<td>1</td>
</tr>
<tr>
<td>CED training</td>
<td>6.1</td>
</tr>
<tr>
<td>CED prediction</td>
<td>0.069</td>
</tr>
</tbody>
</table>

Table 4 – Approximate normalized GPU times to predict the complete synthetic data for the ALFT and the CED methods using a single GPU node. The CED training includes ALFT, migration, de-migration, and training using the complete synthetic dataset.

As shown in Table 4, the training process requires additional steps and is time-consuming. However, these steps are needed only for the subset of the survey area where we train the network. Once the CED model is trained, the computational cost for prediction is only 6.9% of the ALFT. This means that if we use a subset of the entire survey area for training, the CED may be more CPU/GPU effective than the ALFT.
CONCLUSIONS

In this study, we investigated the use of a 3D CED for interpolation and regularization of sparsely sampled near offset class volumes. A new strategy for training data generation was proposed, that is employing migration and de-migration. For each offset class, we de-migrated the stacked PSDM image into two configurations; (i) once using the original survey configuration consisting of the recorded source/receiver positions (ii) a target survey configuration with constant offset and azimuth for each 3D offset volume. These two synthesized data sets were then used to train our 3D CED to map the offset classes from the actual acquisition layout to the “Ideal“ one, and therefore perform interpolation and regularization. We demonstrated that the trained 3D CED performs well both in a synthetic and a real data example. Moreover, we showed examples of AVO preservation of the method using modelled synthetic data. The migrated results showed clear wavefield patterns implying a promising 3D wavefield reconstruction from the proposed method.

ACKNOWLEDGEMENTS

The authors thank CGG, Lundin, and the University of Oslo for providing the work environment for this research. We are grateful to the Multi-Client groups in CGG and TGS for providing show rights. We acknowledge the Research Council of Norway, CGG, and Lundin for funding this study through industrial PhD-grants, project numbers 297180 and 287664. Finally, we gratefully recognize the help of Céline Lacombe and Samuel Gray from CGG for reviewing the manuscript.

DATA AVAILABILITY STATEMENT

Research data are not shared.

REFERENCES


6. Discussion and outlook

The first part of this thesis describes noise in marine seismic data, while the last and main part is on data interpolation using Deep Neural Networks (DNN). It demonstrates some of the actual noise problems of the seismic data together with the state-of-the-art noise reduction techniques and seismic data resolution improvements through a deep neural network (DNN)-based interpolation.

The main scientific contributions can be summed up as:

(1) Chapter 2 and Paper I give detailed descriptions of a wide range of marine noise types on towed-streamer arrays, some widely adopted techniques for their attenuation. The work also presents some novel processing tricks developed for industrial de-noising.

(2) Chapter 4 and Paper II develop a methodology for DNN-based interpolation of regularly missing traces for the cases of cross-streamer interpolation and shot point interpolation with promising results.

(3) Chapter 5, including Paper III develops a number of DNN-based workflows for interpolation and regularization of irregularly and sparsely sampled 3D offset class data. The main outcome is a novel interpolation approach currently tested in CGG’s standard workflow, with a patent application No. 17/483,197 entitled “Modeling-based Machine Learning for seismic processing” submitted in September 2021.

Throughout the whole thesis, we have compared the DNN-based workflows with the results of existing state-of-the-art processing algorithms. This comparison is not a trivial task and was only possible by having access to large amounts of seismic data and modern seismic data processing software. The computational resources required to obtain these results are also significant. Not only in terms of CPU/GPU time, but also in terms of storage capacity and disk/network input-output.

We have also done extensive testing on CNN network designs, trying to find setup/parameters that were favorable for seismic data processing. The last bit is time-
consuming but has resulted in new insights into how existing DNN software can be improved to support better processing of seismic data.

The following subsections overview some of the observed challenges during this study and give suggestions for future research with the aim of overcoming them.

### 6.1 Noise considerations

The presence of noise in marine seismic data has been an issue ever since the towed streamer arrays were introduced. The recorded seismic data contains both signals that we are interested in and unwanted noise. We would ideally like to separate these without compromising the signal's integrity. These needs can be met through new streamer designs or optimized processing solutions, or a combination of both. The seismic industry is moving towards multicomponent streamers, hydrophones and particle motion sensors, where transversal vibrations are the primary noise source in the motion sensors. There are currently three types of multicomponent streamer systems available (Tenghamn et al., 2007; Özdemir et al., 2012; Mellier et al., 2014). Each system uses different mechanics, electronics, and processing solutions to attenuate the unwanted noise. Goujon et al. (2019) discussed a newer generation of streamers that would combine optimized mechanical properties that reduce the vibration with a new single sensor configuration capable of reducing noise. The most plausible solution to the noise problem will probably involve a combination of new streamer designs and optimized processing solutions. The latter will always be required as inevitable external noise will be recorded independently of the type of equipment used.

Despite the apparent effectiveness of the available noise suppression techniques through processing, these methods can be improved in quality and efficiency. DNN can be a possible solution to some of the noise problems, and as we have witnessed within the past few years, this is an active area of research. Based on the presented work, I believe a promising approach would be to investigate further the cost function that can help develop these methods. Modifying the cost function may represent an improved way of teaching the network which data features are essential. For example, consider a de-noise problem where rather than measuring the misfit between the labeled and target data directly in the $t$-$x$ domain, we map the data to a domain where the two more easily separate. This modification of the cost function may improve the performance of the model.
A suitable way of doing so will be to add an additional term in the cost function. Let us recall Equation (3-10) and modify it such that the input and predicted data denote images:

\[ C(W, b) = \frac{1}{N} \| Y - A^L \|^2 + \lambda \| W \|^2, \]

that represents the MSE with the additional regularization term. Tau-p mapping (Diebold and Stoffa, 1981; Stoffa et al., 1981) is often used in de-noise processing; therefore, a modified cost function may take the form:

\[ C(W, b) = \frac{(1 - \alpha)}{N} \| Y - A^L \|^2 + \lambda \| W \|^2 + \frac{\alpha}{N} \| \mathcal{T}(Y) - \mathcal{T}(A^L) \|^2, \]

(6-1)

where \( \mathcal{T} \) is the tau-p mapping operator, and \( \alpha \in [0,1] \) controls the importance between the MSE and tau-p terms. This additional term may be beneficial because the unwanted noise may separate more easily than in the t-x; thus adding more constraints on the DNN solution.

In seismic data processing, we often exploit different and more suited domains for a given task at hand. For that reason, I think it is important not to throw away this knowledge. Instead, use it and find a way to incorporate it in DNN methods.

### 6.2 Sparsely sampled seismic data

The problem of sparsely sampled seismic data discussed in this thesis exists primarily because of physical, environmental, and efficiency constraints. 3D acquisition layouts usually do not tow streamers too close to each other. The smaller the streamer separation, the smaller the bin size in the crossline direction; thus, the higher the resolution. In contrast, that increases the risk of tangling the towed equipment and the cost of the seismic survey. Therefore, streamer separations below 50 m are usually avoided. Another way of increasing the resolution in the crossline direction is by employing more and wider-towed sources. Traditionally, marine seismic sources used to be towed in front of the two inner-most streamers. However, increasing the number of the deployed sources which are towed wider and outside the two inner-most streamers together with positioning them closer to the streamer front-ends or even on top of the streamers allows for improved resolution in the crossline direction and better near offset coverage (Vinje et al., 2017; Vinje and Elboth, 2019; Widmaier et al., 2020). Increasing the number of sources leads to smaller source volumes due to seismic vessel compressor capacity, limiting penetration depth. However, it was demonstrated that source volumes of approximately 1000 m³ are sufficient for imaging down to 3 s of data (Dhelie et al., 2018b). Another problem with increasing the number of sources is its time from flip-to-
flip, unless one introduces blending, which comes with its own set of challenges. Implementing wider-tow approaches is an active area of research and requires modifications and improvements in the hardware solutions and the processing ones. Changes in the hardware usually are relatively expensive and take a significant amount of time to be fully commercialized. Besides, such improvements are not possible for vintage data that one may want to re-process. For that reason, we proposed DNN driven approaches for data resolution improvements both in 2D and 3D. The presented techniques in Chapters 4 and 5 demonstrated the potential of DNN as an alternative way of seismic data interpolation of missing data. However, these methods require future research to address some of the problems observed in this thesis. The future acquisition may combine improvements on the equipment side, i.e., more tailored acquisition layouts, and new or improved processing solutions through conventional or DNN algorithms.

Paper II explored training a DNN model from the densely sampled dimension of the shot gathers to interpolate data in the usually less well-sampled orthogonal dimension, i.e., across the streamers. The paper demonstrated the feasibility of the method and showed promising results. This method requires the use of source-over-streamer (Vinje et al., 2017) acquisition because the two spatial orientations, i.e., inline and crossline, share a split-spread geometry. Applying the proposed technique will be significantly more challenging for conventional front source acquisition because the inline and crossline geometry differs. The data's inline geometry or arrival pattern follows hyperbolic curves, while the crossline geometry is a split-spread. Addressing this issue (at least partially), we can apply NMO, flatten the data, and make the two domains more similar.

In Paper III, we discussed the issue of sparsely sampled 3D offset classes. As discussed in this work, the chicken-and-egg problem is shared with a range of other processing and imaging techniques, i.e., Full Waveform Inversion. One way to divorce the proposed method from such implicit dependency is to use small subsets of data from different vintage surveys. These subsets can then be de-migrated onto the original and idealistic survey configurations (as shown in the real data example in Paper III) and augmented for different sparsely and irregularly sampled ones. This combination of various datasets and augmented survey layouts will improve the generalization of the method and make it more robust to apply it to data it has never seen before. Alternatively, when moving onto a new survey area, one can use transfer learning to initialize new training (Wang et al., 2020). In such a way, the
trained model will be retrained just for a few epochs before deploying it in production, which will reduce the training time significantly.

6.3 High-frequency loss vs network architecture

In Chapters 4 and 5, we have used a customized version of the U-Net (Ronneberger et al., 2015) architecture for shot point interpolation, and interpolation and regularization of 3D offset class data. These two methods revealed a common issue: the interpolated traces suffered from high-frequency loss. Although the residual layers (skip connections) in theory should compensate for such losses as part of the U-Net, we still noted bias towards the low-frequencies.

We first investigated the possibility of introducing different cost functions. Inspired by improvements in natural image processing (Zhao et al., 2016) by the use of Structural Similarity Index Measure (SSIM) (Wang et al., 2004) as a cost function, we tried addressing the challenges mentioned above by implementing this cost function as a measure for the misfit for our U-Net investigated cases. We tried different combinations of hyperparameters, deeper or shallower U-Net architectures, with and without dropouts. However, no improvements were observed.

A possible explanation discussed by Ye et al. (2018) suggested that the U-Net-based architectures might emphasize low-frequencies because the fixed down-sampling pooling operations with skip-connections do not always guarantee perfect results. Yasarla et al. (2020) also suggested U-Nets do not focus on local features (high-frequencies) because the deeper layers of these networks have large receptive fields, thus emphasizing high-level features (low-frequencies).

As a result, Paper II and III used different network architectures, where the loss of high-frequencies was lower for Paper II and nearly negligible for Paper III. In Paper II, combining a customized CNN without a fixed down-sampling operator, i.e., pooling, but employing the wavelet decomposition performed by PyWavelet (Lee et al., 2019) instead contributed to restoring more high frequencies. Other studies also demonstrated an increase in the learning algorithm's performance when employing wavelet decomposition to simplify the image structure (Bae et al., 2017; Guo et al., 2017; Liu et al., 2018).
Paper III used an overcomplete convolutional encoder-decoder with striding convolutional operations for down-sampling instead of fixed operators, i.e., pooling. Combining trainable striding convolutions (Dumoulin and Visin, 2016) and overcompleteness restored more high-frequencies. Other fields of study like medical image segmentation (Valanarasu et al., 2020, 2021) and image de-raining (Yasarla et al., 2020) have recently explored overcomplete representations and reported improved CNN performance.

Based on the conducted studies in this thesis, we observed that the network architecture had a significant influence on the restoration of high-frequencies for our specific tasks. The U-Net struggled more with high-frequencies compared to the network architectures used in Paper II and III. Therefore, it will be interesting to investigate an overcomplete 3D approach for the examples presented in Chapter 4, utilizing the full 3D shot gather instead of 2D representations.

**6.4 Deep neural network with multi-feature seismic input/output vs video prediction?**

In Chapters 4 and 5, we have employed multi-feature input to the DNN. For the shot point interpolation case, we provided adjacent 2D gathers as an input feature space to the DNN. We used higher offset classes as an input feature space for the 3D offset class interpolation and regularization. This technique has proved to be beneficial in terms of quality. The improvement is because the adjacent 2D gathers or 3D offset classes correlate, which provides the convolutional kernels with additional information.

In computer vision, video prediction represents a significant challenge. The general task in these methods can be formulated to use a set of frames from a video recording system to train a DNN and predict future frames (Ranzato et al., 2014; Patraucean et al., 2015; Srivastava et al., 2015; Lotter et al., 2016). Suppose we ignore the fact that these methods use different network architectures, primarily unsupervised, while we deal with supervised methods in this thesis. In that case, one might ask, what are the similarities then? If we arrange several consecutive 2D or 3D shot gathers, for example, they can be considered as a short video, where each gather illustrates a frame from a video. Of course, there will be differences, but these seismic “frames” will not change too much for such short distances. The same applies to the 3D offset classes.
The idea behind this comparison of seismics and natural videos is that in video prediction, it is demonstrated that combining the standard loss with a gradient difference loss (GDL) sharpens the predicted frames because it penalizes the differences of image gradient predictions (Mathieu et al., 2015; Xu et al., 2018; Sarkar et al., 2021). To make this approach feasible for the studies conducted in this thesis, we should change the output feature space of our methods. As formulated in Chapters 4 and 5, the input feature space differs from the output. We should first make this two equal. That means if we have five consecutive offset classes on input, we should have five consecutive offset classes on output. The same rule applies to the shot point interpolation discussed in Chapter 4. This modification is required to ensure a multi-feature output space to allow the GDL to use this information. The expression of GDL for our purposes can then be formulated as finite difference approximation:

$$GDL = \frac{1}{hwc} \sum_{i,j,k} |\nabla_i Y - \nabla_i A^L|^\gamma + |\nabla_j Y - \nabla_j A^L|^\gamma + |\nabla_k Y - \nabla_k A^L|^\gamma,$$  \hspace{1cm} (6-2)

where $\nabla_i Y = [Y_{i,j,k} - Y_{i,j,k-1}]$, $\nabla_j Y = [Y_{i,j,k} - Y_{i,j-1,k}]$, $\nabla_k Y = [Y_{i,j,k} - Y_{i,j,k-1}]$, analogously and for $\nabla_i A^L, \nabla_j A^L, \nabla_k A^L$. The subscripts $i, j, k$ denote the pixel coordinates with $h$ and $w$ the height and the width of the image, and $k$ the feature space, i.e., 2D gather or higher offset class representation. The hyper-parameter $\gamma$ of the GDL loss can be 1 or 2.

The combined MSE cost function then becomes:

$$C(W, b) = \frac{(1 - \alpha)}{N} \| Y - A^L \|^2 + \lambda \| W \|^2 + \frac{\alpha}{N} GDL,$$  \hspace{1cm} (6-3)

where $\alpha \in [0,1]$ controls the importance between the MSE and GDL terms.

This modification of the cost function may lead to better performance because it will introduce additional constraints and also ensure that the data are coherent along with the $k$ feature space. Therefore, I think the GDL loss should be investigated for seismic data interpolation applications.
6.5 Physics-based methods vs deep neural networks

The primary aim of any seismic survey is to obtain a detailed image of the subsurface. This subsurface, however, is a highly complex media whose dynamics are interactively governed by physical, chemical, and geological processes. Therefore, a significant amount of effort is required to understand these processes, usually simulated by solving partial differential equations using, e.g., finite differences.

In contrast, DNNs can explore massive amounts of data, find multi-dimensional correlations and manage ill-posed problems, which in part makes them “ideal” for solving seismic-related tasks. However, DNNs, as a purely data-driven method, may fit well the observations, but prediction may be inconsistent or even completely wrong, i.e., overfitting or missing small but important details. In this thesis, we have discussed some of the techniques available for overcoming the issue of overfitting, like regularizations and dropouts. However, it is important to find a way of integrating physical laws in DNN. In such a way, we can “guide” the DNN by providing additional physical constraints that may lead to better performance and better generalization. Physics informed neural networks (PINNs) (Raissi et al., 2019) are practical and efficient for ill-posed and inverse problems because they seamlessly integrate data and mathematical physics models. Raissi et al. (2019) demonstrated a scheme for solving differential equations governing a physical system where a DNN represents a solution of the system. They proposed to use these underlying equations as training constraints. Moseley et al. (2020) used PINN for solving the wave equation. They improved the performance and concluded that the physics loss allows “extrapolation” outside the training set, i.e., achieved significantly better generalization. The challenge in seismic processing with DNNs is to describe the appropriate physics, which still requires future research to find a way to do that.

Today, DNN-based seismic data processing like de-noising and interpolation treats the seismic gathers similarly to conventional natural images. On the other hand, seismic wavefield images are in a limited space, i.e., they represent strict wavefield patterns compared to natural images. Therefore, seismic wavefield may be optimal for DNN-based algorithms. However, it is very challenging to grasp whether the DNN actually learns any physical relationships from the seismic data. Therefore, PINNs should be further investigated for geophysical problems as they may represent a more suited approach. One should explore combining the “standard” DNN with some physical information to guide the training, like
including the wave equation. Ideally, one could hope to exceed the DNN seismic processing results today that generally struggle to match the best results available by a conventional physics-based method.
References


Chollet, F., 2018, Keras: The python deep learning library: Astrophysics Source Code Library,


Dhelie, P. E., V. Danielsen, J. Lie, A. Wright, N. Salaun, G. Henin, and V. Vinje, 2018c, A novel source-over-cable solution to address the Barents Sea imaging challenges; Part 2, processing and imaging results: 80th EAGE Conference and Exhibition, 1–5.


Dumoulin, V., and F. Visin, 2016, A guide to convolution arithmetic for deep learning: ArXiv
Elboth, T., and F. Haouam, 2015, A seismic interference noise experiment in the central North Sea: 77th EAGE Conference and Exhibition.


Hill, D., L. Combee, and J. Bacon, 2006, Over/under acquisition and data processing: The
next quantum leap in seismic technology? First Break, 24, 81–95.


Long, A., S. Campbell, S. Fishburn, S. Brandsberg-Dahl, N. Chemingui, and V. Dirks, 2014,


Moldoveanu, N., J. Kapoor, and M. Egan, 2008, Full-azimuth imaging using circular
geometry acquisition: The Leading Edge, 27, 908–913.


Peng, H., J. Messud, N. Salaun, I. Hammoud, P. Jeunesse, T. Lesieur, and C. Lacombe, 2021,
Proposal of the DUnet neural network architecture: Deghosting example and theoretical analysis: 82nd EAGE Annual Conference & Exhibition, 1–5.

Poole, G., 2010, 5D data reconstruction using the anti-leakage Fourier transform: 72nd EAGE Conference and Exhibition, Expanded Abstracts B, 46.


Ronneberger, O., P. Fischer, and T. Brox, 2015, U-net: Convolutional networks for biomedical


Wang, Z., A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, 2004, Image quality assessment:
from error visibility to structural similarity: IEEE Transactions on Image Processing, 13, 600–612.


Ye, J. C., Y. Han, and E. Cha, 2018, Deep convolutional framelets: A general deep learning


