AUTOMATIC INTERPRETATIVE SEISMIC IMAGE-FOCUSING ANALYSIS

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF GEOPHYSICS
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Joseph Stanley Jennings
March 2022

Printed as Stanford Exploration Project No. 186
by permission of the author

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Biondo L. Biondi) Principal Adviser

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Robert G. Clapp)

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Eric Dunham)

Approved for the University Committee on Graduate Studies

Abstract

The use of both the physical (midpoint and depth) and the offset/angle axes of seismic images during seismic image-focusing analysis enables a robust assessment of the errors within the migration velocity model. In fact, the use of the focusing information from all axes of the seismic image is crucial for imaging scenarios in which the subsurface is poorly illuminated. However, while the curvature information provided within the offset/angle axis is routinely used to assess velocity errors, the focusing within the physical axes of a seismic image is often neglected. This is due to the highly interpretative nature of the focusing of geological features within seismic images, therefore making focusing analysis within the physical space difficult to automate in a robust way.

I present a data-driven approach to automate interpretative seismic image-focusing analysis that is based on the recent success of supervised deep learning with convolutional neural networks (CNNs). I designed and developed a workflow that uses the output of a fault-focusing CNN as a focusing measure for geological faults within seismic images. As the CNN operates on prestack image patches, it makes use of both the spatial and angle/offset focusing provided by prestack seismic images. I demonstrate the effectiveness of this workflow by applying it to a 2D limited-aperture prestack image from the Gulf of Mexico and a 3D prestack image from the F3 block of the Dutch North Sea. For each application, I train the CNN on a small subset of real focused and unfocused image patches extracted from the images themselves, and also on synthetically generated focused and unfocused image patches to overcome the lack of real image patches for training. I then apply the trained CNN to each image

during focusing analysis and demonstrate that the CNN-based approach is able to robustly estimate focusing maps from each of the images even in the presence of limited illumination.

Using the estimated focusing maps, I am able to improve the focusing and automatic interpretation of faults present within the unfocused images through an image refocusing procedure. When comparing the resulting refocused images from the CNN-based approach to those obtained from a traditional semblance-based approach that only uses the curvature along the aperture-angle axis, I show on both 2D and 3D field data examples that the automatically interpreted fault surfaces within the refocused image from the CNN-based approach are more coherent and geologically consistent with the faults observed within the image.

Preface

The electronic version of this report[1] makes the included programs and applications available to the reader. The markings **ER**, **CR**, and **NR** are promises by the author about the reproducibility of each figure result. Reproducibility is a way of organizing computational research that allows both the author and the reader of a publication to verify the reported results. Reproducibility facilitates the transfer of knowledge within SEP and between SEP and its sponsors.

ER denotes Easily Reproducible and are the results of processing described in the paper. The author claims that you can reproduce such a figure from the programs, parameters, and makefiles included in the electronic document. The data must either be included in the electronic distribution, be easily available to all researchers (e.g., SEG-EAGE data sets), or be available in the SEP data library[2]. We assume you have a UNIX workstation with Fortran, Fortran90, C, X-Windows system and the software downloadable from our website (SEP makerules, SEPScons, SEPlib, and the SEP latex package), or other free software such as SU. Before the publication of the electronic document, someone other than the author tests the author's claim by destroying and rebuilding all ER figures. Some ER figures may not be reproducible by outsiders because they depend on data sets that are too large to distribute, or data that we do not have permission to redistribute but are in the SEP data library, or that the rules depend on commercial packages such as Matlab or Mathematica.

[1] http://sepwww.stanford.edu/public/docs/sep186
[2] http://sepwww.stanford.edu/public/docs/sepdatalib/toc_html/

CR denotes Conditional Reproducibility. The author certifies that the commands are in place to reproduce the figure if certain resources are available. The primary reasons for the CR designation is that the processing requires 20 minutes or more.

NR denotes Non-Reproducible figures. SEP discourages authors from flagging their figures as NR except for figures that are used solely for motivation, comparison, or illustration of the theory, such as: artist drawings, scannings, or figures taken from SEP reports not by the authors or from non-SEP publications.

Our testing is currently limited to LINUX 2.7 (using the Intel Fortran90 compiler) and the SEPlib-7.0.5 distribution, but the code should be portable to other architectures. Reader's suggestions are welcome. For more information on reproducing SEP's electronic documents, please visit *http://sepwww.stanford.edu/research/redoc/*.

Acknowledgments

While the title page of this thesis portrays a single author, the completion of this work would not have been possible without the help and support of many. It is with a humble spirit that I take the time now to express my deep gratitude for all of those that supported me on this long journey.

I first must thank my adviser, Biondo Biondi. He provided me with the opportunity to study and perform research within the Stanford Exploration Project (SEP) for nearly seven years. I initially arrived at Stanford intimidated by those around me and very unsure of myself. Through his positive outlook and unwavering optimism, Biondo helped me to find confidence in myself. Biondo also provides the students in SEP with a tremendous amount of research freedom. This helped me to develop the necessary research skills to and further develop my confidence to complete my PhD.

For the majority of my PhD, I was privileged to be co-advised by Shuki Ronen. I had the pleasure of working with Shuki on a number of different projects all of which involved the use of field data. I learned a tremendous amount from Shuki's practical yet technically sound approach to working with field data. This knowledge later became invaluable as I embarked on processing and imaging a 3D dataset on my own. Also, whenever I needed a second opinion on my work, Shuki always seemed to find the time out of his busy schedule to lend a critical eye to my reports and give me constructive feedback on my research.

Throughout my PhD, Bob Clapp was a wonderful source for ideas and always provided a unique way of solving a particular problem. In fact, the central idea of

this thesis occurred to me due to a conversation that I had with Bob. Bob has an incredibly extensive knowledge of geophysics and high-performance computing and without his guidance, the majority of the research I performed at Stanford would not have come to fruition. I also must thank Bob for recommending to me work on a machine learning side project with TotalEnergies that later became this thesis.

I was very fortunate to have Eric Dunham as my second project adviser. Eric is most certainly one of the best teachers I have ever had throughout all of my university training, and I learned a great amount from his course lectures as well as our one-on-on meetings. In addition to my work with Eric as part of my second project, he was always available for my annual reviews and also served on both my reading and defense committees. His input on faulting and faulting mechanisms was especially useful for me to consider when applying my method to the 3D image from the North Sea.

Two additional sources of help for me during my PhD were Jon Claerbout and Stew Levin. Jon's youthful spirit and attitude of learning new things was an inspiration to me throughout my PhD. Stew Levin was immensely helpful while he was at SEP during the first several years of my PhD. He paid close attention to each of my presentations during SEP seminar and would offer insightful questions and comments. Additionally, he spent a considerable amount of time helping me with the geometry of the Kietta data, which I later used in my thesis proposal as well as two publications.

The staff within the geophysics department was extremely supportive to me throughout my years at Stanford. Without the help of Rachael Madison, I certainly would not have been able to accomplish all of the milestones necessary for obtaining a PhD from the geophysics department. Liliane Pereira, Claudia Baroni and Denise Baughman were all very helpful in the administrative aspects of SEP.

SEP is a group of many students, many of whom were absolutely instrumental to the completion of this thesis. I first and foremost must thank my office-mates: Huy Le, Milad Bader, Badr Alrumaih and Chris Leader. I was very fortunate to

have incredible office-mates from whom I learned so much. I sincerely do not think I would have passed my qualifying exam without the help from Huy. Milad was always available for discussing research and his advice for imaging the 3D dataset was immensely helpful (I greatly missed our interactions during the stay-at-home order due to the pandemic). When I arrived at Stanford, Yang Zhang, Ali Almomin, Ohad Barak, and Musa Maharramov, who were the most senior students at the time, were very welcoming to me and offered a lot of help in getting started. As I progressed in my PhD, I learned a lot from discussing and sharing seminars with Yi Shen, Jason Chang, Taylor Dahlke, Eileen Martin and Gustavo Alves. I always enjoyed conversing with Yinbin Ma and he was extraordinarily helpful in offering suggestions for writing an efficient finite difference code. Both Guillaume Barnier and Ettore Biondi helped me to better develop my fundamental understanding of inversion and seismic imaging operators which proved to be very helpful in my research. Alejandro Cabrales-Vargas was always a great friend to me and I thoroughly enjoyed our discussions about WEMVA and the adjoint method. He also was always very patient with me when I would miscommunicate in Spanish. I arrived at Stanford with Fantine Huot and Rahul Sarkar and it has been a pleasure to collaborate and learn with them over the past seven years.

I also learned and benefited immensely from the students and post-docs who arrived after me at SEP. The machine learning discussions I had with Gabriel Fabien-Ouellet, Stuart Farris, Siyuan Yuan, Minjun Park and Francesco Picetti taught me a lot about research directions in deep learning and its use in inversion. When participating in the velocity estimation and imaging discussion groups, I learned a great deal from Ariel Lellouch, Rustam Akhmadiev, Miguel Ferrer-Avila and Julio Frigerio. Finally, it has been a pleasure to share seminars and participate in discussions with Jonathan Voyles, Paige Given, Bin Luo and Joe Stitt.

Outside of SEP, I also benefited greatly from working with several other colleagues. I very much enjoyed collaborating with both Leighton Watson and Kaiwen Wang during their second projects within SEP. While working on my second project, I worked closely with Martin Almquist who taught me quite a bit of mathematics and who I

also considered as a mentor. One of the highlights of my time at Stanford was collaborating with Marko Jakovljevic and Rehman Ali from the Dahl Ultrasound lab. My experience with them introduced me to the fascinating field of pulse-echo ultrasound imaging. Also, a special thanks to Jeremy Dahl for encouraging our collaboration and for chairing my defense.

The idea for this thesis originated from a project funded by TotalEnergies. I thank TotalEnergies for the financial support while working on this project and Russell Hewett, Rami Nammour and Mauricio Araya-Polo for the frequent meetings and very useful discussions. Mauricio also was extremely helpful to me throughout my PhD with very useful insights and ideas about deep learning approaches applied to the seismic imaging problem. He also served as a great mentor during two of my internships at Shell.

The writing of this thesis greatly benefited from the careful review of Evelin Sullivan with the technical communication program at Stanford. She painstakingly reviewed several chapters of my thesis and dramatically helped to improve the grammatical correctness of the text. I also thoroughly enjoyed our weekly interactions and her humorous character.

I would like to thank my family for supporting me throughout these seven years, and long before I started my PhD. My parents have always encouraged me to educate myself to the best of my ability and they made countless sacrifices to ensure that I had the best opportunity to do so. This thesis would not be possible without them.

Finally, I must thank my sweet and beautiful wife, Hongye. She has been with me long before I started at Stanford and remarkably, remained with me throughout this long journey. She not only provided me with loving companionship and support throughout my PhD, but financially she kept our family alive as living in off-campus housing solely on a graduate student salary is not possible in the San Francisco Bay Area. She and our cat Luna bring me the greatest joy in life, and I am deeply and forever grateful to her.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

PROBLEM OVERVIEW

Geophysicists use seismic images in order to better understand geologic structures and material properties located beneath the surface of the earth. To create a seismic image, geophysicists will first acquire reflection seismic data and then through a process known as seismic migration, reconstruct seismic images from the recorded seismic reflections. A key component of the reconstruction process performed during migration is a migration velocity model. The migration velocity model determines the positioning of the recorded seismic reflections in physical space and also controls the focusing of seismic images. The estimation of the migration velocity from seismic data is a challenging ill-posed inverse problem and therefore there often exist inaccuracies within the estimated migration velocity model.

When the migration velocity is incorrect, within their physical axes ($x-$midpoint, $y-$midpoint, and depth), seismic images lose reflector coherence and in the presence of faults and rugose geological features, exhibit unfocused/overly focused fault plane reflections and diffractions. The angle/offset axis within prestack seismic images provides an additional axis for analyzing seismic image focusing. When the migration velocity is correct, events within angle/offset gathers will be flat; when it is incorrect, they will either curve upward or downward. The use of the focusing information

within the physical axes as well as the angle/offset axis of a seismic image can provide a high resolution and robust assessment of the migration velocity model.

Seismic image-focusing analysis is a seismic imaging technique that analyzes focusing information within the physical and angle/offset axes of seismic images for the assessment of the accuracy migration velocity. While there exist a variety of forms of seismic image-focusing analysis, they share the common goal of analyzing focusing (i.e., velocity) errors from a collection of focused and unfocused seismic images. The first attempts at focusing analysis generated these images via downward continuation and extracted focusing information from 2D downward continued wavefields. Assuming small offsets and layered media, coordinate transformations could be performed to transform the depth/time extrapolation axis to velocity. Subsequently, depth/time-focusing analyses could be performed on the downward continued wavefields in order to obtain estimates of the prestack migration velocity (Yilmaz and Chambers, 1984; Faye and Jeannot, 1986; MacKay and Abma, 1992). Later, as computational resources improved, Audebert et al. (1996) introduced the concept of common-reflection point (CRP) scans using prestack Kirchoff depth migration. To create a collection of focused and unfocused 3D prestack seismic images, the migration velocity was scaled by a constant factor known as a migration scanning-parameter (i.e., focusing parameter). Sava (2003) then further extended the migration scanning technique to images obtained from 3D prestack wave-equation depth migration using prestack Stolt residual depth migration. While these approaches were successfully used for different applications of focusing and migration velocity analysis and laid the foundation for modern seismic image-focusing analysis, they all treated each image point independently (using only the offset/aperture-angle axis) and therefore neglected the use of the focusing information within the physical axes of the image. In fact, the use of only the curvature within offset/angle gathers and the neglect of the focusing information within the physical axes remains standard practice within seismic imaging workflows today. While a number of factors have led to this practice, one driving factor is that analyzing geological focusing within seismic images is a highly interpretative process that is difficult to automate.

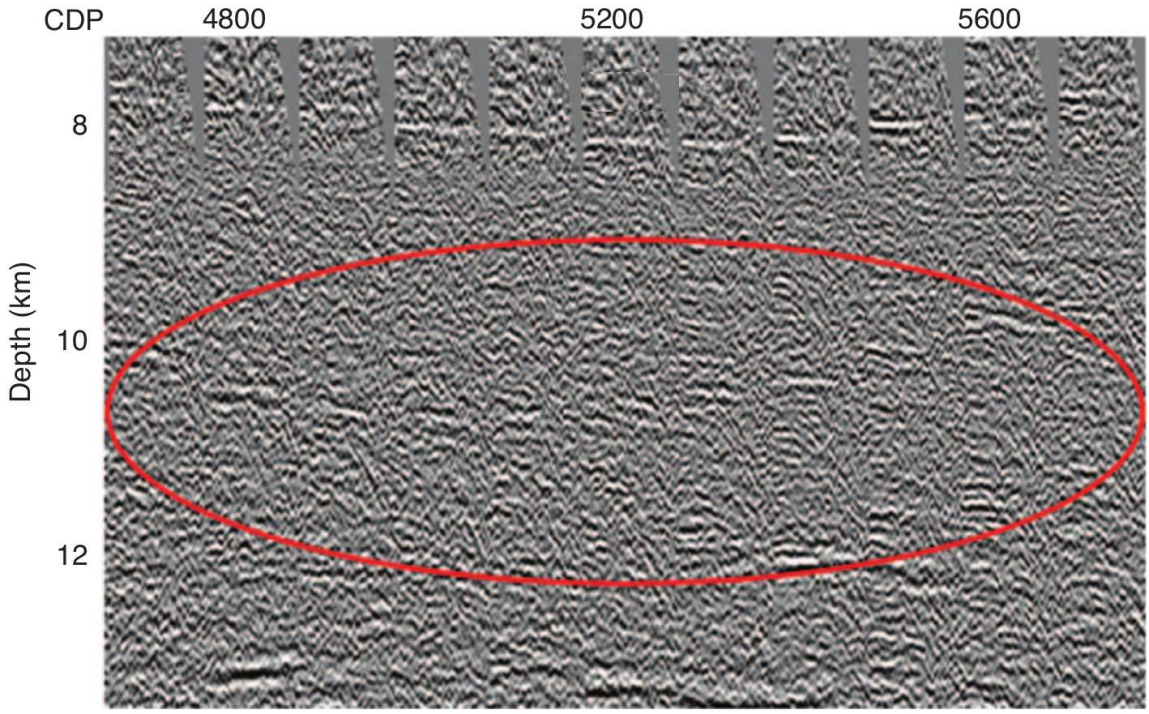


Figure 1.1: Angle gathers extracted from a subsalt seismic image. Note lack of coherent events within the angle gathers and low-signal to noise ratio due to the limited aperture-angle range caused by the presence of the salt. Figure modified from Wang et al. (2008). [**NR**]

The interpretative nature yet strong benefits of image-focusing analysis is perhaps best illustrated when attempting to assess focusing errors of seismic horizons and other geological features that are positioned beneath salt bodies. Due to the high velocity contrast and often irregular shape of salt bodies, reconstructing high fidelity subsalt images from reflection seismic data is a challenging task. Figure 1.1 shows several offset gathers extracted from image points positioned beneath a salt body located in the Gulf of Mexico. Note that within the red ellipse there is little to no signal of coherent events within the offset gathers and the gathers primarily consist of noise. This lack of signal within the gathers renders them unusable for typical tomography and focusing analysis workflows that operate gather-by-gather. However, by analyzing the coherence and focusing of the subsalt horizons within the physical axes of the image, reliable focusing information can be extracted and used to update the velocity model. This is illustrated in Figure 1.2 which shows three images extracted from the result of a migration scanning procedure. Panel (b) of Figure 1.2 shows the image obtained with the original migration velocity and panels (a) and (c) show the images obtained after migrating the data with velocity models that were scaled to be 20% slower and 15% faster than the original migration velocity (i.e., they were 80% and 115% of the original migration velocity). The colored points in the images correspond to image points of interest that were used for focusing analysis. Comparing the imaged reflectors positioned beneath the salt, it is apparent that a number of reflectors are more coherent and better imaged for the image obtained with the slower migration velocity (Figure 1.2(a)). This is especially apparent for horizons positioned between image points 6000 and 6250 as well as a small portion of a horizon positioned at approximately the 6750 image point and at 14 km in depth. While this qualitative analysis provided additional information on velocity errors within the image and led to a credible estimation of subsalt velocities, it was highly interpretative and required knowledge of the regional geology (Wang et al., 2008).

Although the interpretative nature of using the physical axes of the image makes focusing analysis considerably more difficult to automate than the procedure of extracting velocity information from the curvature of the angle gathers, many successful attempts have been made to create a robust and automatic image-focusing analysis.

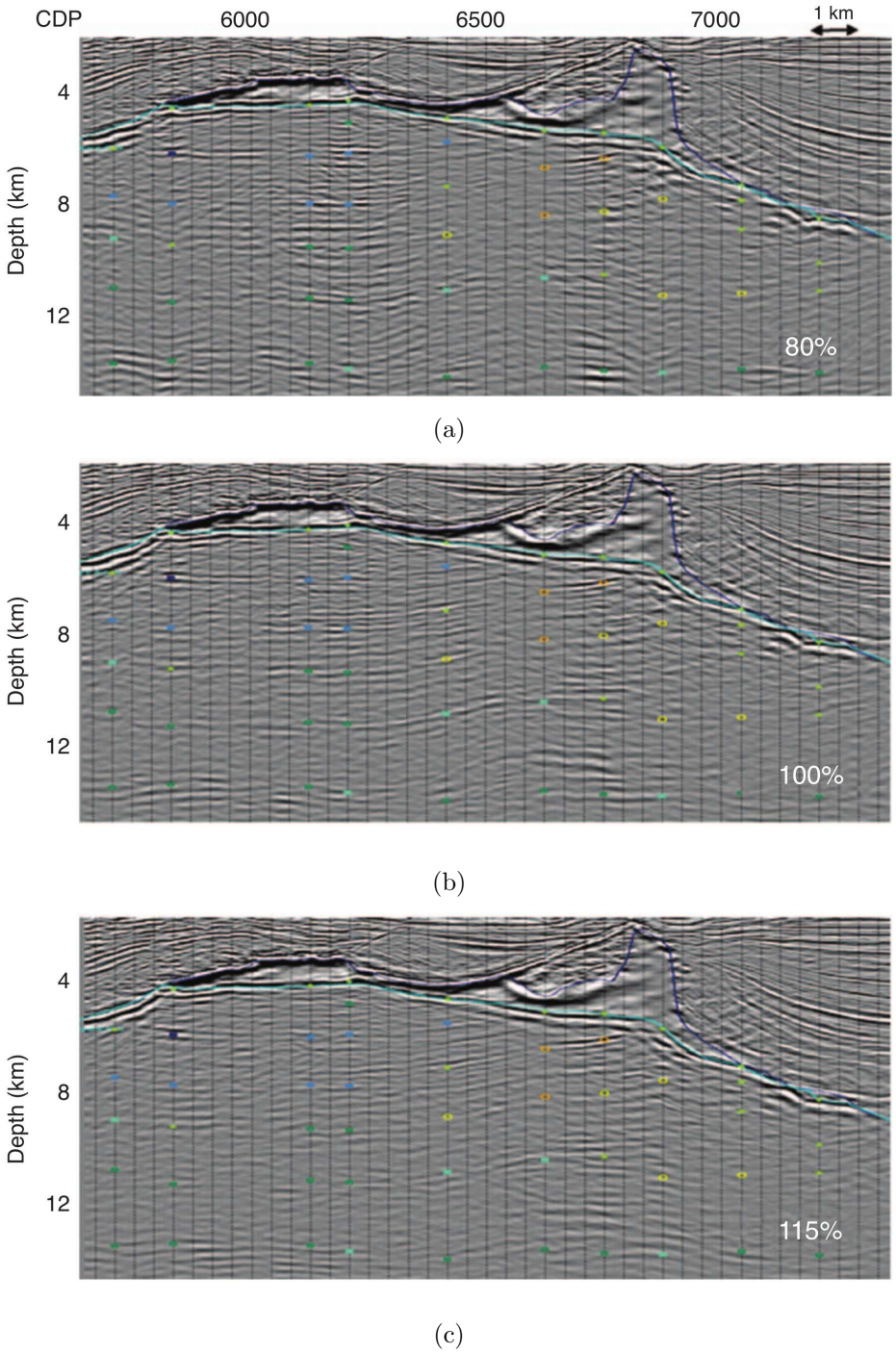


Figure 1.2: An example of a subsalt interpretative image-focusing analysis. The three panels show inline slices extracted from the result of a 3D image-focusing analysis. The panels correspond to images obtained from scaling the migration velocity by (a) 0.8, (b) 1.0 (the original image) and (c) 1.15. Note that between image points 6000 and 6250, there is a significant improvement in the coherence of the horizons beneath the salt in the 80 % image. Image modified from Wang et al. (2008). [**NR**]

Existing approaches can be split into two main categories. The first of these categories consists of diffraction-based approaches that analyze the focusing of diffractions separately from the focusing of reflections. These approaches have been well-studied and have been able to improve imaging and interpretation in a variety of geological settings such as near salt bodies, faults and also channels (De Vries and Berkhout, 1984; Harlan et al., 1984; Fomel et al., 2007; Montazeri et al., 2020). The majority of these approaches neglect the angle/offset axis and use an image entropy-based focusing measure for determining the optimal focusing of a diffraction.

Approaches that fall into the second category make use of coherence along the different axes of the image. Négron et al. (2000) extended the original CRP scan approach proposed by Audebert et al. (1996) to a more interpretative approach in which the physical axes of the image were also incorporated by analyzing the amplitudes and depths of horizons within the different migrated cubes from the scans. Wang et al. (2006) later extended the CRP-scan approach to a wave-equation-based migration algorithm and used only the maximum energy, resolution and spatial coherence of horizons as qualitative measures within the physical axes for the focusing analysis. They applied their approach to a sub-salt example in which there existed little to no aperture-angle information beneath the salt and they were able to successfully update the velocity and improve the image beneath the salt (Wang et al., 2008). Biondi (2010) also based their approach on wave-equation migration and developed a quantitative and automatic image and aperture angle coherence-based approach for performing the image-focusing analysis. Most recently, Whiteside et al. (2011) developed an automatic approach for image-focusing analysis but it was only for reverse-time migration (RTM)-based delayed imaging time (DIT) scans (Wang et al., 2009). They perform the automatic focusing analysis within the DIT gather and ensure geologic consistency via structural constraints such as horizons provided from automatic horizon extraction. They also show that their automated DIT scan approach has the added benefit of improving the interpretation of the salt.

While approaches in both diffraction and coherence categories have demonstrated success towards developing an automated interpretative image-focusing analysis, their

solutions are not all-encompassing. For one, the diffraction-based approaches rely on the detection and isolation of diffractions from reflections within seismic images. This can prove to be a challenging task in the presence of noise or limited illumination. For another ,these approaches are limited in that they only target point scatterers within stacked seismic images. Therefore, they neglect reflector coherence and curvature in addition to the curvature of the angle gathers available along the angle/offset axes of prestack seismic images.

While coherence based approaches have become routine procedures for subsalt imaging workflows, they, too, are not without limitations. While they incorporate structural constraints by performing the focusing analysis along extracted horizons, the picking of the optimal focusing parameter is performed on each image-point separately using a maximum amplitude criterion which can potentially lead to a spatially inconsistent assessment of the focusing errors. Note that, both the diffraction-based and the coherence-based approaches can be considered from an automation perspective as "rule-based" approaches that rely on the specification and extraction of particular features within focused and unfocused images in order to assess the velocity error. While these features can be extracted and used for a wide variety of seismic images, there are also many instances in which they do not apply (e.g., corner cases) and therefore other features need to be specified and the rule-based approaches need to be updated.

AUTOMATING SEISMIC IMAGE-FOCUSING ANALYSIS WITH CNNS

To overcome the limitations of the above-discussed approaches for performing automatic focusing analysis, this thesis presents a novel workflow for automatic interpretative image-focusing analysis that uses the powerful automatic feature extraction capabilities of supervised deep learning with convolutional neural networks (CNNs). In recent years, supervised deep learning with CNNs has shown remarkable abilities at automating a wide range of computer vision tasks . It has been especially useful

for tasks that are difficult to define formally (easy for humans to perform but difficult to automate with an algorithm). Within the computer vision community, examples of these tasks have been image classification, face recognition and object detection (Krizhevsky et al., 2012; Taigman et al., 2014; Liu et al., 2016).

At its core, supervised learning with deep CNNs is a data-driven, learning-based approach that has the primary goal of finding a model that explains the relationship between the elements of a dataset (e.g., image containing a chair) and their corresponding labels (e.g., a bounding box that surrounds the chair). While traditional supervised learning approaches required extensive manual extraction of features from data (commonly referred to as feature engineering) for complicated tasks, deep learning with CNNs is able automatically extract features from data via elaborate neural network architectures. This has enabled deep CNNs to learn incredibly complex relationships from datasets across a number of different fields.

The feature extraction and learning capabilities of CNNs have also recently begun to transform the seismic interpretation community. The ability of CNNs to learn features within seismic images has allowed them to detect subsurface geological structures such as horizons, faults, channels and salt bodies within seismic images from a wide variety of complex geological scenarios. (Waldeland et al., 2018; Wu et al., 2019; Pham et al., 2019; Tschannen et al., 2020). This advancement has aided in the improved characterization of reservoirs and a reduced turnaround time of seismic imaging and interpretation workflows (Wrona et al., 2021; Zhao et al., 2021).

The CNN-based automatic image focusing workflow I present in this thesis is similar to the work done in the seismic interpretation community. While the CNNs trained for seismic interpretation typically learn to detect the presence of geological features within seismic images, I desire to know whether the geological feature is focused within the image. I achieve this by training a CNN to classify prestack seismic patches as focused or unfocused. Assigning a label value of one to the focused patches, and zero of unfocused patches, the CNN is trained to provide a focusing score that will be larger for focused patches and smaller for unfocused patches. The unthresholded output of the CNN is then used as a focusing measure for determining

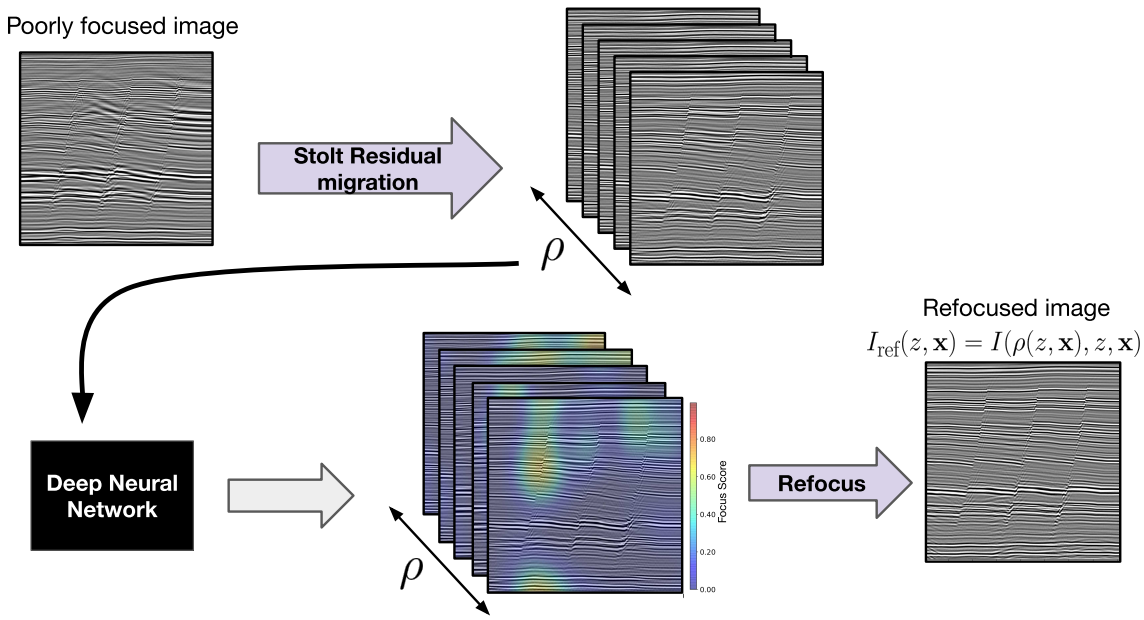


Figure 1.3: Summarized visualization of CNN-based focusing analysis and fault refocusing presented within this thesis. The input poorly focused image is first provided as input to prestack Stolt residual depth migration which results in collection of residually focused images. Each image results from a different migration scanning parameter ρ. These images are then divided into spatial patches which are then all provided to the CNN which computes a focusing score for each patch. Then, the ρ value that maximizes the focusing score is selected and assigned to all pixels within the patch which results in a spatial estimate of $\rho(z, \mathbf{x})$. Finally, a refocused image can be obtained from the residually migrated images using the estimated $\rho(z, \mathbf{x})$. [**NR**]

the focusing errors within prestack seismic image patches generated as part of a focusing analysis.

A summarized illustration of the workflow I developed for CNN-based seismic image-focusing analysis is shown in Figure 1.3. The input to the workflow is a prestack unfocused seismic image as is shown in the top left portion of the figure. The first step of the workflow is to create focused and unfocused seismic images as is done with any seismic image-focusing analysis. While there are many different approaches to creating focused and unfocused seismic images for image-focusing analysis, I choose to use the method of prestack Stolt residual depth migration (Sava, 2003) because it can create many prestack focused and unfocused images in a computationally efficient manner. I refer to these focused and unfocused images as residually focused images. Note that while my approach does assume images resulting from prestack Stolt residual depth migration, it can be readily adapted to focusing analysis from Kirchoff migration, wave-equation migration or RTM DIT scans. With the residually focused images created from residual prestack Stolt depth migration, the images are then provided as input to the fault-focusing CNN that has been trained with focused and unfocused seismic image patches. This is done by spatially splitting the images into overlapping patches and then providing all patches to the CNN, which computes a focusing score for each patch. Then, the migration scanning parameter associated with the maximum focusing score predicted for a patch is selected as the scanning parameter for all spatial locations within the patch. This is then repeated for all spatial patches in the image until a spatial map of the migration scanning parameter is estimated. This spatial map provides a root-mean square estimate of the velocity errors observed within the image.

While the central goal of this thesis is to demonstrate that the use of supervised learning with CNNs can provide a robust estimate of the focusing errors within a seismic image, I also use the estimated focusing errors in order to improve the image (this also helps to perform a quality control on the estimated focusing map). While the estimated focusing map can be used in different ways to improve the image, I follow the method described by MacKay and Abma (1989) and use it directly as a focusing

surface that indicates which residually focused image provides the best focusing for every point in the image. Performing this interpolation across all residually focused images then reconstructs a refocused image with faults with improved focusing, which are better suited for tasks such as automatic fault interpretation.

THESIS OUTLINE

In this chapter I have shown that developing a robust, automatic and interpretative approach to seismic image-focusing analysis is a challenging and as yet unachieved aim. I have also provided a high-level summary of the novel CNN-based focusing analysis that I have developed as an attempt at achieving this aim. In the remaining chapters of this thesis, I will describe in detail the underlying concepts upon which this approach is built and I will also demonstrate the effectiveness of my proposed CNN-based approach on one synthetic and two field data applications. The following sections provide an outline of what is described in Chapters 2 to 5.

Chapter 2: Measuring image focusing with a CNN

I introduce the concept of seismic image-focusing analysis using prestack Stolt residual depth migration as presented by Sava (2003). I also introduce my CNN-based approach to performing focusing analysis. This includes the creation of synthetic training data, the CNN training and the application of the CNN to the residually focused images to obtain an estimate of the focusing map.

Chapter 3: 2D synthetic subsalt and field data example

I present the application of the CNN-based focusing analysis presented in Chapter 2 to a 2D synthetic subsalt example and a 2D field data example from the Gulf of Mexico. For the synthetic example I use only the stacked images and for the field data example, I use prestack images for the CNN-based focusing analysis. For both

cases, I compare the results of CNN-based focusing analysis with a semblance-based analysis that uses only the curvature of the angle gathers for focusing analysis.

Chapter 4: Application to 3D Netherlands F3 data

I describe the extension of the 2D CNN-based fault-focusing analysis to 3D using 4D cross-correlation operators and apply it to a 3D prestack image from the F3 block of the North Sea. As for the 2D data examples, I compare the results of semblance-based focusing analysis with CNN-based focusing analysis and demonstrate the image refocused from the focusing map obtained from the CNN-based focusing analysis results in more coherent and geologically consistent segmented faults.

Chapter 5: Conclusion and future directions

I provide a general summary of the conclusions that can be drawn from the results presented from Chapters 2, 3 and 4. I also describe potential directions for future research. These involve the incorporation of the focusing of other geological features for CNN training such as horizons and river channels, the possible use of other methods for creating focused and unfocused seismic images and also how to improve CNN generalization given the availability of a large dataset consisting of focused and unfocused 3D prestack seismic images.

Chapter 2

Measuring image focusing with a CNN

In this chapter I introduce the theory and workflow of seismic image-focusing analysis of faults with CNNs. As its use in recent years has been limited, I begin by reviewing seismic image-focusing analysis with prestack Stolt residual depth migration. I then introduce semblance-based focusing analysis, which assesses the focusing of the image using only the flatness of the residually migrated gathers via the calculation of semblance. Following my description of semblance-based focusing analysis, I describe the design of the CNN used for focusing analysis as well as the generation of synthetic pre-training images and training of the CNN. Then after training the CNN, I show that the output of the trained CNN applied to a seismic image patch can be used as a seismic image focusing measure. Finally, through the visualization of the estimated CNN filter coefficients and computed feature maps as well as through CNN activation maximization and the calculation of sensitivity heat maps, I provide an interpretation of the inner-workings of the CNN. The interpretation of the results from each of these methods show that the CNN learns to extract the fault position and the strength of the fault discontinuity in order to provide an assessment of the focusing of a seismic image patch.

PRESTACK STOLT RESIDUAL DEPTH MIGRATION

As I mentioned in Chapter 1, while there exist a number of migration-scanning techniques for performing seismic image-focusing analysis, I use prestack Stolt residual depth migration for all of the focusing analyses shown in this thesis. As indicated by the name, prestack Stolt residual migration is based on a seismic imaging technique known as residual migration. While standard migration algorithms transform seismic data into an image, residual migration transforms an image that has been migrated with a particular velocity, into an image that has been migrated with another velocity. A major advantage of this image-to-image mapping is that as residual velocity errors are in general small, less accurate imaging algorithms may be utilized (Rothman et al., 1985). Typically, less accurate imaging algorithms are also less computationally demanding and therefore many residual migrations can be performed at relatively little cost.

One of these less accurate imaging algorithms that has been widely used in residual migration is Stolt migration. Beasley et al. (1988) demonstrated the utility of residual Stolt migration for poststack seismic images where they showed that they could achieve accurate imaging of a salt body using a series of Stolt residual migrations. Later, Stolt (1996) derived the residual Stolt prestack time operator extending Stolt residual migration to prestack images. Sava (2003) further extended the work of Stolt (1996) by deriving a residual prestack Stolt depth migration operator enabling the use of Stolt residual migration for depth images. Sava (2003) additionally demonstrated the use of the residual prestack Stolt depth migration operator for interpretative image-focusing analysis.

In comparison to other migration scanning methods used for focusing analysis, prestack Stolt residual depth migration offers two significant advantages. The first of these is computational efficiency. As prestack Stolt residual depth migration is based on the Stolt migration algorithm, all that is required to compute a residual migration image is the calculation of a forward fast fourier transform (FFT), an interpolation step, and then an inverse FFT. The cost of this method is comparable to that of

Kirchoff-based scanning approaches and is orders of magnitude cheaper than wave equation migration-based scanning approaches. The low cost of the method further facilitates the computation of many residually focused images allowing for a high-resolution focusing analysis. The second advantage offered by prestack Stolt residual depth migration is that it operates on prestack wave equation-migration images which enables the use of both residual moveouts and residual physical focusing to be used for assessing seismic image focusing. In contrast, more recent approaches based on RTM DIT gathers only operate on stacked images which generally results in a lower resolution focusing analysis.

In the following section I provide a derivation for the full prestack residual Stolt migration depth migration operator similar to what is presented in Sava (2003). I include it here in this chapter as it is central to the work described in this thesis. I begin from the definition of the double square root (DSR) operator used for downward continuation of prestack seismic data and first derive the prestack Stolt depth migration operator in midpoint-offset coordinates. From there I manipulate the dispersion relation that provides a mapping from $\omega \rightarrow k_z$ (Stolt migration) to a dispersion relation that provides a mapping from $k_{z_0} \rightarrow k_z$ (residual Stolt migration).

Derivation of the prestack residual Stolt depth operator

I begin by defining the recorded seismic data as pressure recordings measured at the earth's surface: $p(\mathbf{m}, \mathbf{h}, z = 0, t)$ where $\mathbf{m}$ is the midpoint coordinate, $\mathbf{h}$ is the offset coordinate, z is depth and t is time. To begin the derivation of the prestack Stolt dispersion relation, I compute the Fourier Transform of the data:

$$p(\mathbf{k}_m, \mathbf{k}_h, z = 0, \omega) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} p(\mathbf{m}, \mathbf{h}, z = 0, t) e^{i(-\omega t + \mathbf{k}_m^T \mathbf{m} + \mathbf{k}_h^T \mathbf{h})} d\mathbf{m}\, d\mathbf{h}\, dt, \quad (2.1)$$

where $\mathbf{k}_m$ is the midpoint wavenumber, $\mathbf{k}_h$ is the offset wavenumber and ω is the angular frequency. With the data in the frequency-wavenumber domain, they can

then be downward-continued with the DSR operator (Claerbout, 1985). In shot-geophone coordinates, the DSR operator can be expressed as

$$\begin{aligned} k_z &= DSR(\omega, \mathbf{k}_s, \mathbf{k}_g) = k_{z_s} + k_{z_g} \\ &= \sqrt{\frac{\omega^2}{v^2(\mathbf{s}, z)} - |\mathbf{k}_s|^2} + \sqrt{\frac{\omega^2}{v^2(\mathbf{g}, z)} - |\mathbf{k}_g|^2}, \end{aligned} \tag{2.2}$$

where $v(\mathbf{s}, z)$ is the velocity at the source location, $v(\mathbf{g}, z)$ is the velocity at the receiver location, $\mathbf{k}_s$ is the source wavenumber and $\mathbf{k}_g$ is the receiver wavenumber. I can transform this operator into midpoint-offset coordinates using the following coordinate transformation:

$$\mathbf{k}_s = \frac{\mathbf{k}_m - \mathbf{k}_h}{2}, \tag{2.3}$$

$$\mathbf{k}_g = \frac{\mathbf{k}_m + \mathbf{k}_h}{2}. \tag{2.4}$$

Substituting Equations 2.3 and 2.4 into Equation 2.2, I arrive at the following expression for the DSR operator in midpoint-offset coordinates

$$k_z = \sqrt{\frac{\omega^2}{v^2(\mathbf{s}, z)} - \frac{1}{4}|\mathbf{k}_m - \mathbf{k}_h|^2} + \sqrt{\frac{\omega^2}{v^2(\mathbf{g}, z)} - \frac{1}{4}|\mathbf{k}_m + \mathbf{k}_h|^2}. \tag{2.5}$$

To downward continue the data from the surface to a depth z, I can use the DSR operator as follows:

$$p(\mathbf{k}_m, \mathbf{k}_h, z, \omega) = p(\mathbf{k}_m, \mathbf{k}_h, z = 0, \omega)\, e^{-ik_z z}, \tag{2.6}$$

which provides a wavefield at the depth z. To form an image, $I(\mathbf{m}, \mathbf{h}, z)$, at this depth, I can integrate over frequency (zero-time imaging condition) and perform the inverse Fourier transform over the spatial axes:

$$I(\mathbf{m}, \mathbf{h}, z) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} p(\mathbf{k}_m, \mathbf{k}_h, z = 0, \omega) e^{-ik_z z}\; e^{-i(\mathbf{k}_m^T \mathbf{m} + \mathbf{k}_h^T \mathbf{h})} d\mathbf{k}_m d\mathbf{k}_h d\omega, \tag{2.7}$$

To derive the prestack Stolt operator, I consider the case of constant velocity. For this case, there is no need downward continue the wavefield, rather I can use the dispersion relation directly. To do this, I first re-express Equation 2.5 in terms of k_z, which after some algebra can be written as follows:

$$\omega^2 = v^2 \frac{[4k_z^2 + (|\mathbf{k}_m + \mathbf{k}_h| - |\mathbf{k}_m - \mathbf{k}_h|)^2][4k_z^2 + (|\mathbf{k}_m + \mathbf{k}_h| + |\mathbf{k}_m - \mathbf{k}_h|)^2]}{16k_z^2}, \tag{2.8}$$
$$\omega^2 = v^2 \frac{\mu}{16k_z^2}.$$

Note that I have introduced the additional variable μ as it greatly shortens some of the following expressions. Now with an expression for ω in terms of k_z, I can perform a change of variables in Equation 2.7 eliminating ω. This also requires calculating the Jacobian of the coordinate transformation which can be expressed as follows:

$$d\omega = J(\mathbf{k}_m, \mathbf{k}_h, k_z)\, dk_z, \tag{2.9}$$
$$J(\mathbf{k}_m, \mathbf{k}_h, k_z) = v \frac{16k_z^2 - (|\mathbf{k}_m + \mathbf{k}_h| - |\mathbf{k}_m - \mathbf{k}_h|)^2(|\mathbf{k}_m + \mathbf{k}_h| + |\mathbf{k}_m - \mathbf{k}_h|)^2}{8\sqrt{\mu}}\, dk_z.$$

With the coordinate transformation, Equation 2.7 changes to:

$$I(\mathbf{m}, \mathbf{h}, z) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} J(\mathbf{k}_m, \mathbf{k}_h, k_z)\, p\left[\mathbf{k}_m, \mathbf{k}_h, \frac{v}{4k_z}\sqrt{\mu}\right] e^{-i(k_z z + \mathbf{k}_m^T \mathbf{m} + \mathbf{k}_h^T \mathbf{h})} d\mathbf{k}_m\, d\mathbf{k}_h\, dk_z. \tag{2.10}$$

Examining Equation 2.10, we can recognize that it describes the inverse Fourier transform of the data evaluated at $v/4k_z\sqrt{\mu}$. Equations 2.1, 2.8 and 2.10 constitute a recipe for Stolt migration: we first compute the Fourier transform over all axes of the data. Then, using the dispersion relation provided by the DSR operator, we map the temporal frequency, ω, to the vertical wavenumber, k_z. Finally, we compute the inverse Fourier transform over all axes of the mapped data which provides a prestack depth image. Note that as this mapping via the dispersion relation does not occur on a regular grid, an interpolation step is required to evaluate $p(\mathbf{k}_m, \mathbf{k}_h, v/4k_z\sqrt{\mu})$ (Stolt, 1978).

To derive the operator for prestack Stolt residual depth migration, I need to find

a mapping between the vertical wavenumber of the original image migrated with velocity v_0, k_{z_0}, and the vertical wavenumber of the output residual migration image migrated with velocity v, k_z. This can be done by writing out Equation 2.8 in terms of k_{z_0} and then substituting in this expression for ω into the DSR operator in Equation 2.5:

$$k_z = \frac{1}{2}\sqrt{\frac{v_0^2}{v^2}\frac{\mu}{4k_{z_0}^2} - |\mathbf{k}_m - \mathbf{k}_h|^2} + \frac{1}{2}\sqrt{\frac{v_0^2}{v^2}\frac{\mu}{4k_{z_0}^2} - |\mathbf{k}_m + \mathbf{k}_h|^2}, \tag{2.11}$$

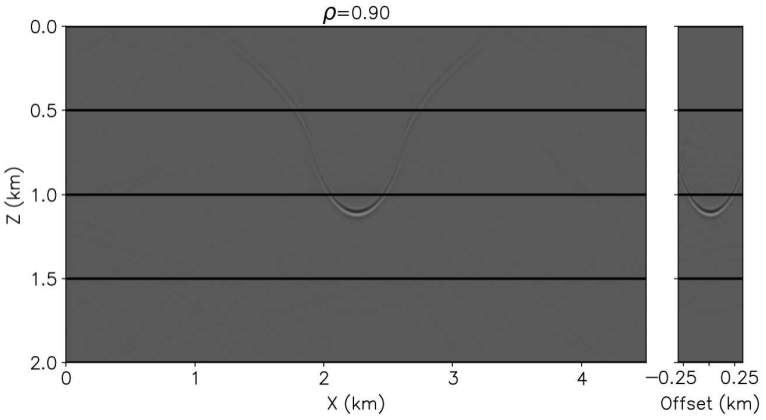
Note that in addition to removing ω from the Stolt operator, Equation 2.11 no longer depends on just the migration velocity, but rather on the ratio between the migration velocity, v_0 and the output residual migration or trial velocity v. For convenience, I denote this velocity ratio throughout this thesis as $\rho = v_0/v$ (Biondi, 2006). Note that while the expression for Stolt migration in Equation 2.10 is a constant velocity migration, Equation 2.11 describes a constant velocity ratio (ρ) method. This implies that while v_0 and v can vary spatially, they must be scaled versions of each other. Therefore, to perform a focusing analysis using Equation 2.11, v_0 remains fixed and v is set to be a scaled version of v_0. The residual migration will then transform the image into the equivalent image that would have been created if the migration velocity were v. Performing this for many different scaling factors, or equivalently for many different ρ values, then provides a collection of focused and unfocused images that can be used for focusing analysis. In this sense, performing prestack Stolt residual migration for a range of ρ values is very similar to performing wave equation migration/ common reflection point scans for focusing migration velocity analysis (Audebert et al., 1996; Wang et al., 2006).

Equation 2.11 describes the mapping for prestack Stolt residual migration for 3D images. For 2D images, the expression can be simplified by setting y-components of $\mathbf{k}_m$ and $\mathbf{k}_h$ to zero. This results in the following 2D residual prestack Stolt operator

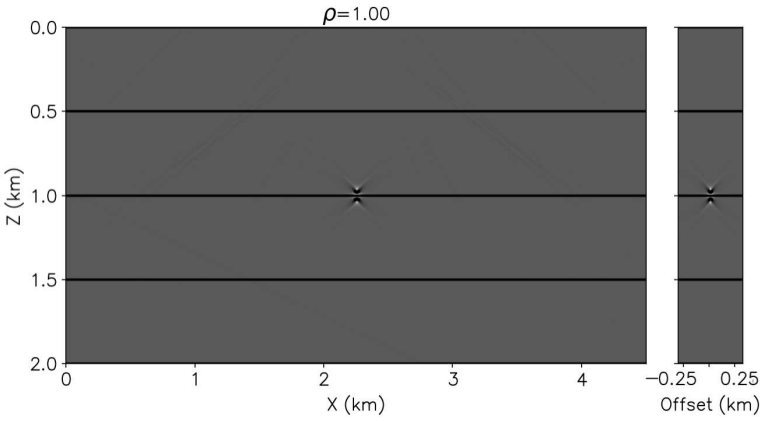
$$k_z = \frac{1}{2}\sqrt{\rho^2 \frac{\left[k_{z_0}^2 + k_{h_x}^2\right]\left[k_{z_0}^2 + k_{m_x}^2\right]}{k_{z_0}^2} - (k_{m_x} + k_{h_x})^2} + \frac{1}{2}\sqrt{\rho^2 \frac{\left[k_{z_0}^2 + k_{h_x}^2\right]\left[k_{z_0}^2 + k_{m_x}^2\right]}{k_{z_0}^2} - (k_{m_x} - k_{h_x})^2}, \tag{2.12}$$

where k_{m_x} and k_{h_x} are the x-components of the midpoint and offset wavenumbers respectively. As the focusing analyses I perform in this and the subsequent chapter are restricted to 2D images, I use Equation 2.12 for the migration scanning procedure for those examples. For the 3D example described in Chapter 4, I use a common-azimuth prestack Stolt residual migration and present the derivation for that residual migration operator in that chapter.

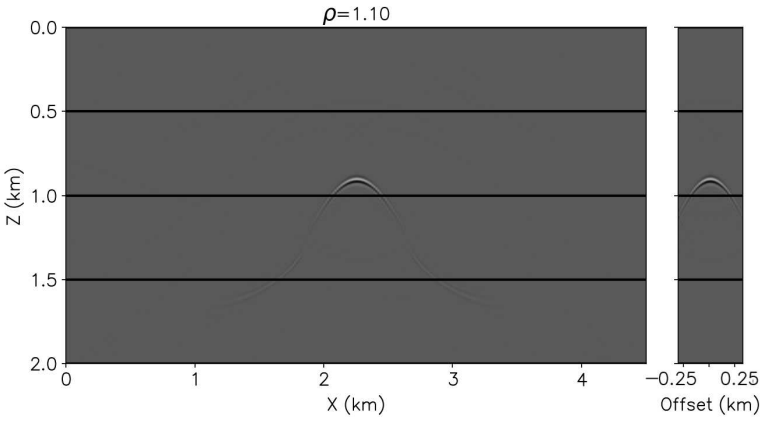
As with any linear operator, visualizing the impulse response of a migration operator is a useful way to better understand it. Figure 2.1 shows the application of the 2D prestack residual migration operator to a band-limited impulse positioned at $x = 2.25$ km, $z = 1.0$ km, and $h_x = 0$ km. The left figure within each panel shows the image at zero subsurface offset and the right figure shows the subsurface offset image extracted at $x = 2.25$ km. The ρ values selected for residual migration were $\rho = 0.9, 1.0$ and 1.1 which correspond to a residual migration velocity higher than the migration velocity, the residual migration equal to the migration velocity and the residual migration velocity lower than the migration velocity respectively. This is apparent in each of the panels of Figure 2.1 where we can observe that the impulse is overmigrated in Figure 2.1(a), the original impulse in Figure 2.1(b) and a diffraction hyperbola in Figure 2.1(c). The nature of these impulse responses indicates that the residual migration operator will either create overmigrated or undermigrated events within the image for ρ less than 1.0 and for ρ greater than 1.0 respectively. Therefore, if an image is unfocused due to predominantly high velocity errors, then residually



(a)



(b)



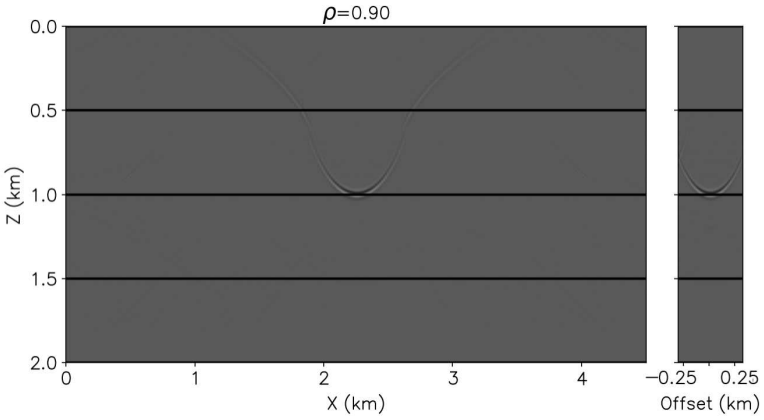
(c)

Figure 2.1: Impulse response of the 2D prestack residual migration operator for (a) $\rho = 0.9$, (b) $\rho = 1.0$ and (c) $\rho = 1.1$. The original impulse is shown for the $\rho = 1.0$ in panel (b). Note that the impulse response is shifted in depth for different values of ρ. [**ER**]

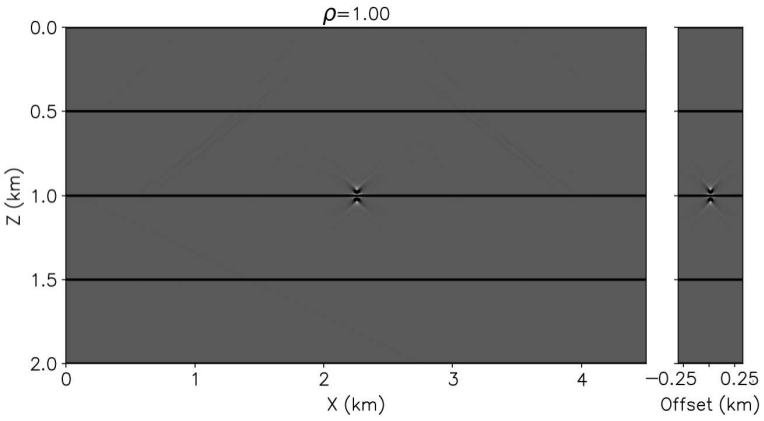
migrating this image with ρ greater than 1.0 will correct for those velocity errors. Similarly, for an image affected with low velocity errors, ρ values less than one will correct for those errors. In addition to the shape of the impulse, it is important to note that because residual prestack Stolt depth migration is a depth migration, migrated events will also be shifted in depth for different values of ρ. This is also clearly evident in Figure 2.1 where the impulse is shifted below 1.0 km in Figure 2.1(a) and above 1.0 km in Figure 2.1(c).

While this shift in depth is important in order to correctly position events in depth in addition to correct for the focusing of the event, it makes interpretative focusing analysis more challenging. Therefore, to eliminate this depth shift and preserve only the focusing of the images that is changed with residual migration, I convert each residual migration to a pseudo-depth (or time) in which they are all aligned. As each residual migration image corresponds to constant velocity scaling of the velocity model, the conversion of depth to pseudo-depth can be achieved with a scaling of the depth axis by the same ρ. Figure 2.2 shows the result of this conversion of depth to pseudo-depth for the impulse responses shown in Figure 2.1. Examining the vertical position of the impulse, it is clear the impulse no longer shifts in depth as ρ is changed. (Note that even for pseudodepth I also label the vertical axis as 'Z' to avoid confusion as the remaining figures in this thesis are shown in pseudodepth). This alignment of events greatly eases the task of interpretative focusing analysis and also is necessary for an image refocusing step that I will introduce in Chapter 3.

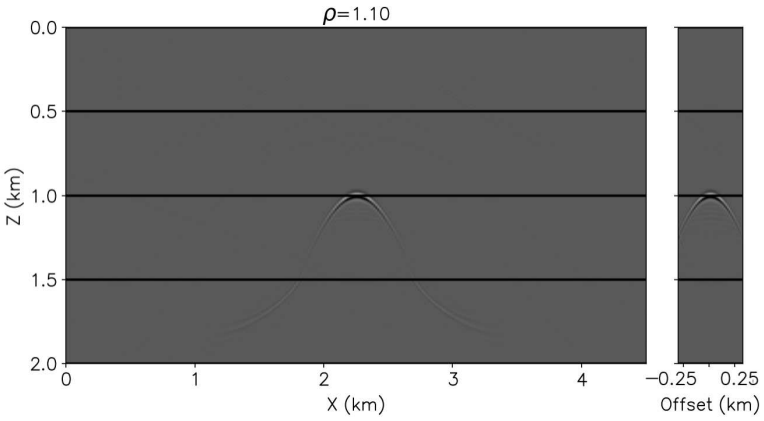
With the intuition provided by the impulse responses of the residual migration operator, I will now present an example that demonstrates the use of prestack Stolt residual depth migration for focusing analysis. Figure 2.3 shows a synthetic unfocused image migrated with a wave-equation depth migration algorithm. To create the image, I first performed linearized (Born) modeling using an extended split-step Fourier modeling algorithm to simulate seismic shot gathers (Kessinger, 1992). For the acquisition geometry, I used a streamer acquisition geometry with 148 shots and a maximum offset of 3.4 km. After modeling, I migrated the data with the same algorithm (adjoint of the modeling algorithm) and during the migration I introduced



(a)



(b)



(c)

Figure 2.2: Conversion of residual migration impulse responses in Figure 2.1 to pseudodepth. Comparing panels (a), (b) and (c) it is clear that the conversion to pseudodepth eliminates the depth shift between the panels within Figure 2.1. [**ER**]

a slow velocity anomaly that extended between 4 and 10 km in the x direction and between 0.5 and 1 km in the z direction. The migration velocity model is shown in Figure 2.4. The introduction of the velocity anomaly reduced the velocity by approximately 0.1 km/s in the shallow portion of the model and resulted in unfocused faults as are apparent in the region of interest of the image shown in Figure 2.5. Examining the faults within Figure 2.5, the truncated reflectors associated with the leftmost faults in the image (between 1 and 3 km) show diffracted events between depths of 1.25 and 1.75 km. Additionally, the three faults positioned between 4 and 6 km in the x direction also contain subtle diffracted energy along their fault trajectories.

Apart from examining the faults, we can also extract focusing information from the extended portion of the image. While velocity errors can be extracted from subsurface offset domain images, they are much easier to interpret in the reflection angle domain. Therefore, I converted the extended image from the subsurface offset domain to the reflection angle domain (Sava and Fomel, 2003) and extracted an angle gather at approximately $x = 4.5$ km and plotted this angle gather next to the stacked image in Figure 2.5. A single angle for all x and z points within the image represents an image that has been acquired for a fixed aperture (reflection opening) angle (Biondi, 2006). Therefore, as the image represents physical geological structures in depth, the depths of these geological structures should be the same for all angles. This then implies that the events seen within the angle gathers should be flat for all angles. Any curvature observed within the gather is indicative of velocity errors present within the image. If the velocity is too low, as is the case for the image shown in Figure 2.5, then the gathers will curve upward and the image is referred to as "undermigrated". Conversely, if the velocity is too high, the gathers will curve downward and the image is referred to as "overmigrated".

With the previous analysis in mind, I then applied prestack Stolt residual depth migration to the unfocused image. As the velocity error introduced into the migration was spatially-variant, a single residual migration image will not correct for the velocity error. Rather, I computed residual migration images for a range of ρ values in order to capture all focusing errors that were potentially present within the image. For this

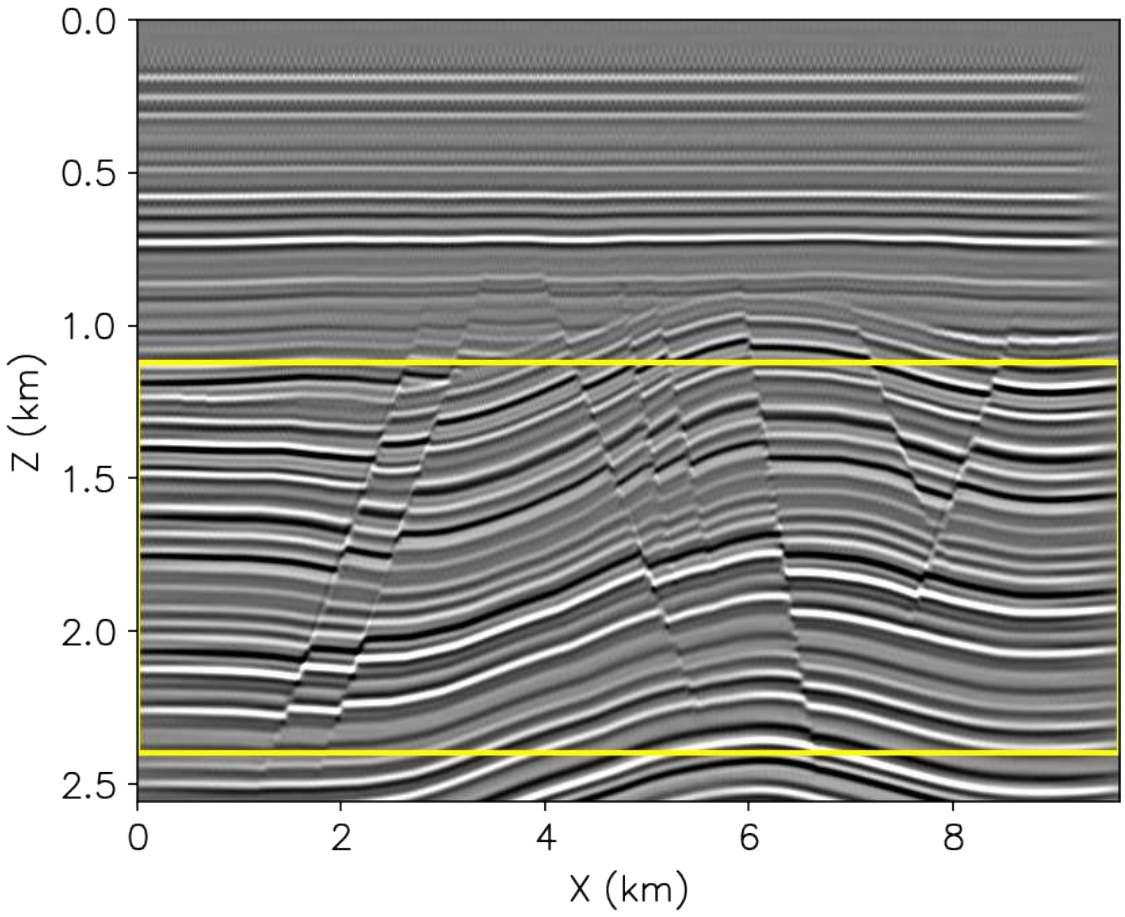


Figure 2.3: Synthetic unfocused image obtained by introducing a slow velocity anomaly in the overburden during wave-equation migration. The yellow box indicates the region of interest for focusing analysis. [**CR**]

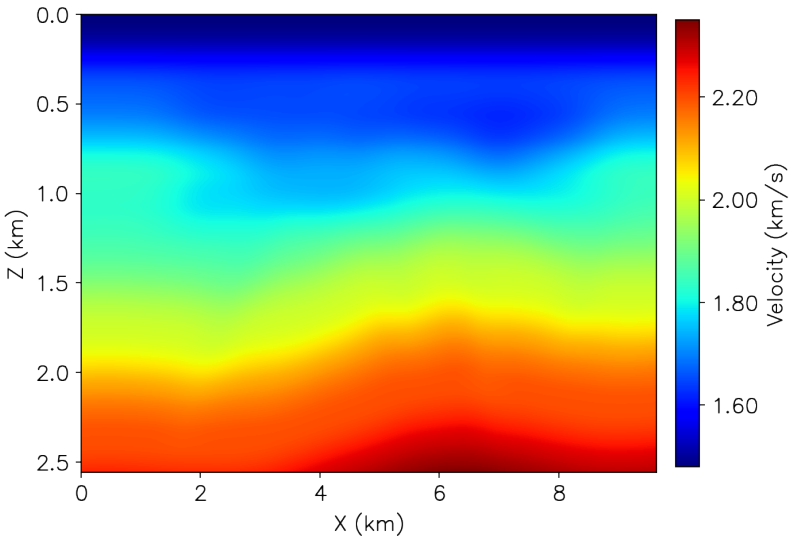


Figure 2.4: The migration velocity model used to create the unfocused image shown in Figure 2.3. The velocity anomaly extends between 4 and 10 km in the x direction and between 0.5 and 1 km in the z direction. [**ER**]

and the other 2D examples presented in this thesis, I chose a range of ρ from 0.9 to 1.1 with an interval of $d\rho = 0.00125$. This range of ρ is wide enough to capture velocity errors of various strengths. With the chosen sampling of $d\rho = 0.00125$, residual migration provides a total of 161 images for focusing analysis (including the $\rho = 1$ image) which allows for a higher resolution analysis of the focusing of the residual migration images. Figure 2.6 shows 5 images selected from the collection of 161 residually migrated images. Looking first at the images corresponding to $\rho = 0.94$ and $\rho = 1.06$, it is readily apparent from the strong diffracted energy as well as the large amount of curvature in the angle gathers that these two ρ values have grossly overcompensated for any velocity errors present within the image and have led to strongly unfocused images. The images corresponding to $\rho = 0.98$ and $\rho = 1.02$ contain much fewer obvious focusing errors and in fact, the $\rho = 0.98$ image has led to significant improvements in the focusing of the faults and the flatness of the gathers above depths of 1.5 km. Additionally, it has the highest amplitude of all the images which is indicative of a greater stack power due to the flatter angle gathers. However, observing deeper in the model, it is apparent that the faults are overmigrated and that the residual migration is overcorrecting for any velocity errors that may be present with in the image. This is especially apparent on the two leftmost faults between 1

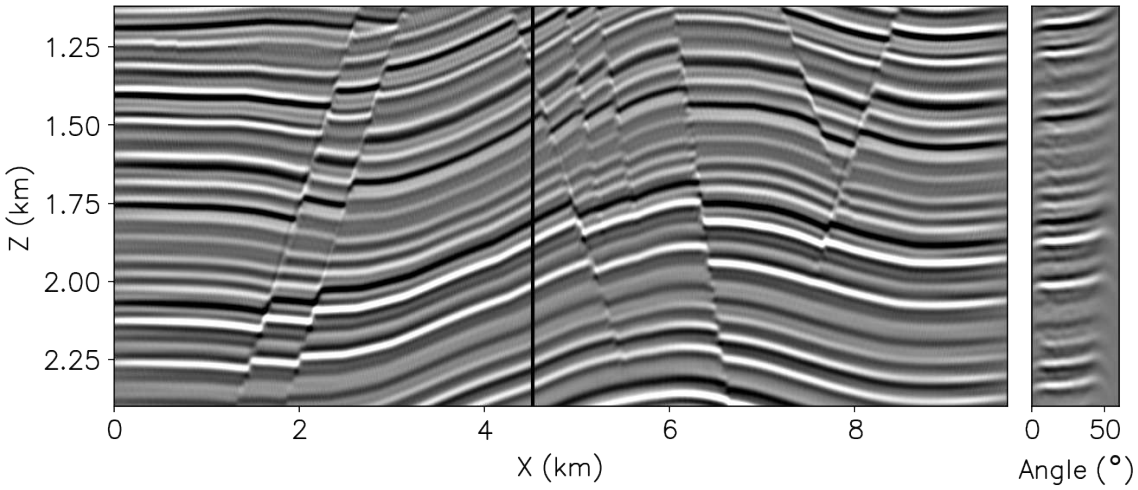


Figure 2.5: The region of interest for the synthetic unfocused image. The diffractions present on the truncated reflectors of the two leftmost faults indicate signs of undermigration. The upward curvature of the angle gathers also clearly indicate that the image is undermigrated. [**ER**]

and 2 km in the x direction and below 2 km in depth where there exist overmigrated "smiling" events associated with truncated reflectors created by the faulting.

The qualitative analysis of the residual migration images shown in Figure 2.6 leads to two key aspects of focusing analysis with prestack Stolt residual migration. The first is that for spatially-varying velocity errors, a single ρ will not suffice to focus the image. Rather, portions of the image will be focused for different ρ values. This observation then leads to the second key aspect of focusing analysis, which is that focusing criteria need to be established in order to extract spatially-variant focusing information from residual migration images. With focusing criteria established, the focusing of the image can then be assessed for all points within the image. In the following section, I will describe the most straightforward focusing criterion which is to examine the flatness of the gathers.

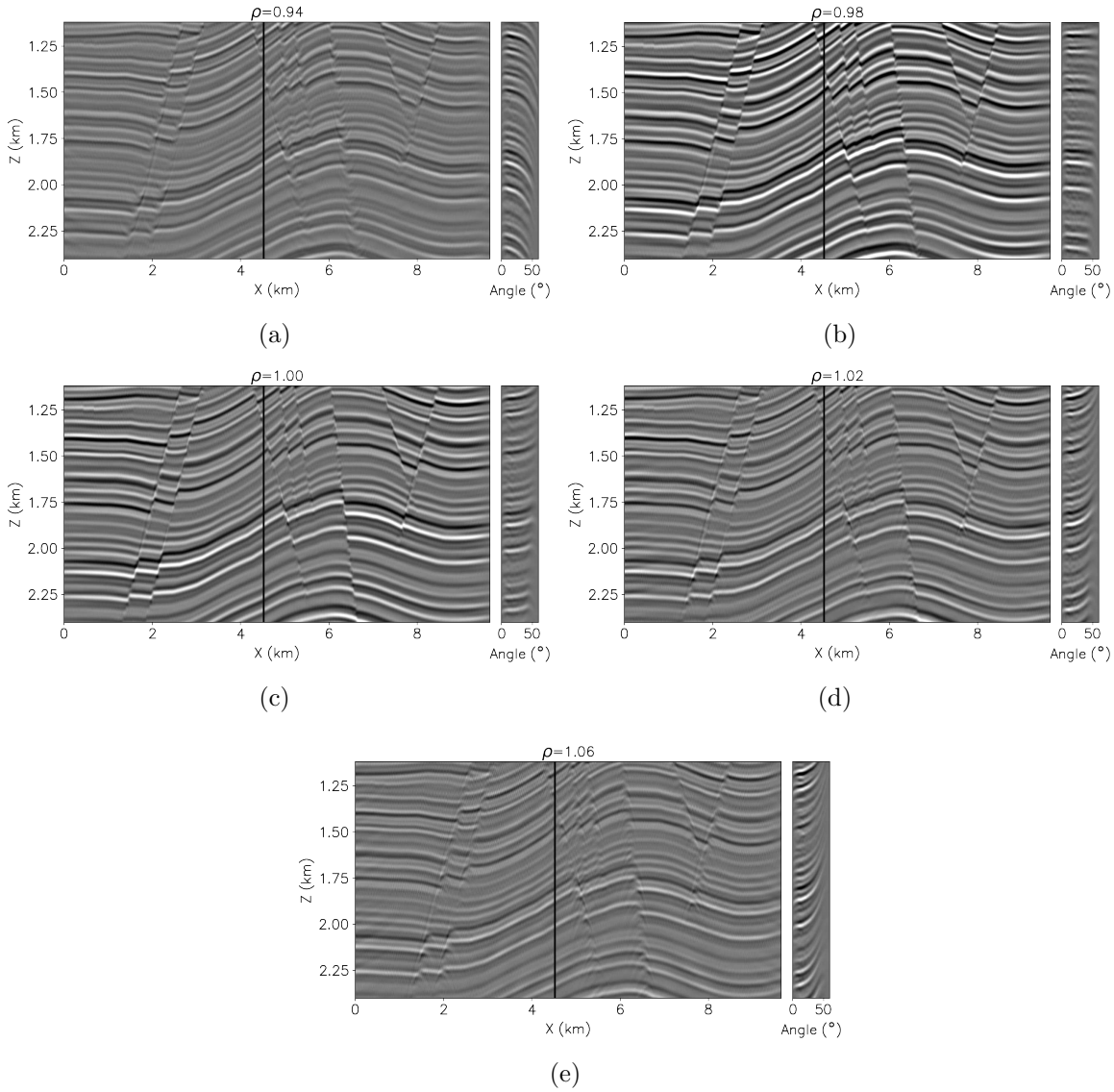


Figure 2.6: A selection of residual migration images taken from the application of prestack Stolt residual depth migration to the unfocused image shown in Figure 2.5. The image in panel (b) corresponding to $\rho = 0.98$ qualitatively provides the best focusing of the faults and flattening of the angle gather in the shallow portion of the image. However, observing the focusing of the faults and the angle gather extracted at $x \approx 4.5$ km for all images, it is apparent that single ρ will not suffice to correct for all velocity errors. [**CR**]

SEMBLANCE-BASED FOCUSING ANALYSIS

As the residual migration images in Figure 2.6 show, the flatness of the gathers is directly linked to the focusing of the images. Therefore, the most straightforward approach to assess image focusing for all image points is to measure the flatness of the angle gathers for different values of ρ. This can be done with the following coherence measure (Neidell and Taner, 1971):

$$s_i = \frac{\sum_{j=i-M}^{i+M} \left(\sum_{k=1}^{N} g_{j,k} \right)^2}{N \sum_{j=i-M}^{i+M} \sum_{k=1}^{N} g_{j,k}^2} \tag{2.13}$$

where g is a residually migrated angle gather for a single ρ, i and j are depth indices, k is the current trace index within the angle gather, n is the number of traces within the gather M is the length of the smoothing filter in depth. Ignoring the outer summands in both numerator and denominator and just analyzing the inner summands, we can observe that the these two summands compute the square of the sum over all traces in the gather (numerator term) and the sum of the squared traces (denominator term) for a single depth point. This value provides the coherence measure over the angle gather. The outer two summands then perform an averaging operation that smooths the computed coherence measure over a depth window of size M. Performing this calculation for all depths and a range of ρ values will result in an image that will indicate for all depths that ρ values that provide the flattest gathers. The maxima of this semblance panel can then be picked resulting in an estimate of $\rho(z)$ for the x position at which the angle gather was extracted.

Figure 2.7 shows an example of a semblance panel calculated from the angle gather extracted at $x \approx 4.5$ km. The left panel of the figure shows the extracted angle gather residually migrated for ρ values between 0.9 and 1.1 and the right panel shows the computed semblance panel. There is a very clear trend on the semblance panel that allows for relatively easy picking of the maxima. I picked the maxima of the semblance panel using an automatic picker that picks the maxima via the solution of a variational problem (Fomel, 2009). The computed pick is displayed as the black

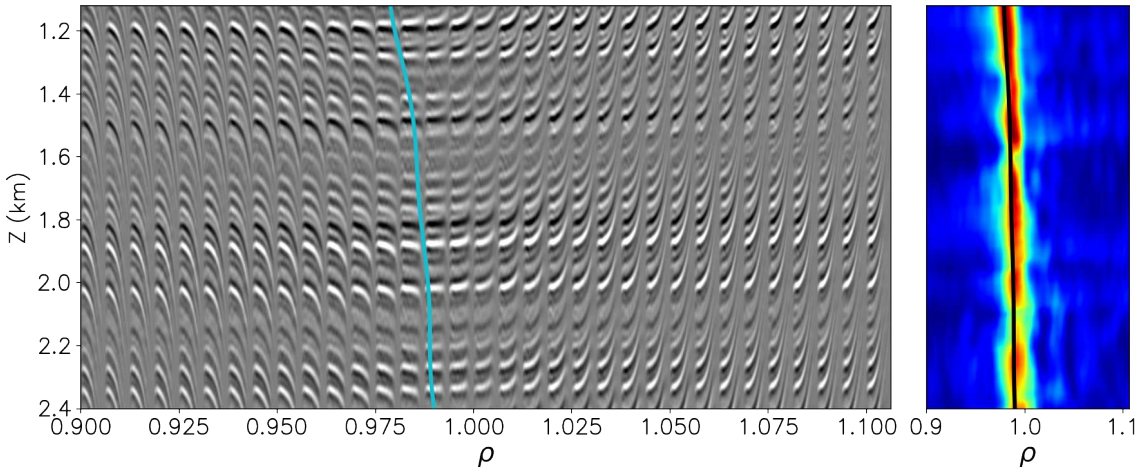


Figure 2.7: Computed ρ semblance from an angle gather extracted at $x \approx 4.5$ km. The left panel shows the angle gather residually migrated for ρ values between 0.9 and 1.1 and the right panel shows the computed semblance. The black and cyan curves show the pick of the maxima of the semblance panel. [**CR**]

curve plotted on top of the semblance panel and as the cyan curve plotted on the residually migrated angle gather. Examining the picked ρ values, it is clear that the picked $\rho(z)$ is gradually increasing with depth. This increasing trend is consistent with the focusing of the faults and the flattening of the gathers that can be observed in Figure 2.6. In order to provide an estimate of ρ for every image point, this process can be repeated for all angle gathers. The picks for each angle gather can then be collected and smoothed providing an estimate for $\rho(z, x)$. This estimate of $\rho(z, x)$ can then be used to spatially assess the focusing of the image. Furthermore, as I will describe in Chapter 3, this spatial estimate of ρ can be used to further update the velocity model as part of a tomographic migration velocity analysis or can be used directly to correct for focusing errors within the image.

Limitations of focusing analysis with Stolt residual migration

While I have demonstrated the effectiveness of the use of prestack Stolt residual depth migration for performing focusing analysis on an unfocused seismic image, it is

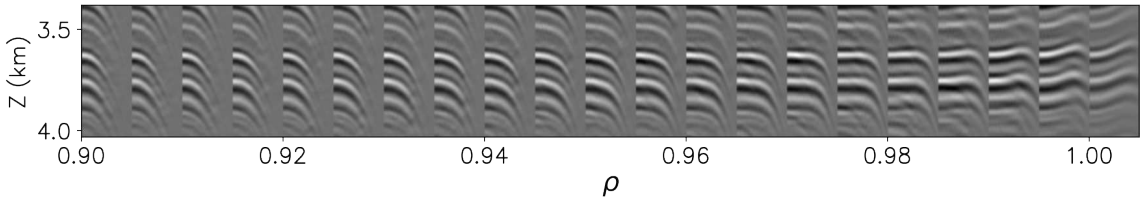


Figure 2.8: Residual migration of an angle gather that has been migrated with a strong local velocity anomaly ($\sim$ 15% of the background velocity). The rightmost gather was extracted from the $\rho = 1$ image. Note the non-hyperbolic nature of the events in the gather. Regardless of the ρ value used, the events cannot be completely flattened with Stolt residual migration. [**CR**]

important to understand two key limitations of this method for assessing or correcting the focusing errors within a seismic image. The first limitation is that Stolt residual migration cannot correct for arbitrary velocity errors, but rather only for velocity errors that are global or smoothly-varying. The nature of the velocity error can be easily seen from the hyperbolic curvature of the angle gather shown in Figure 2.5. When strong and local velocity errors are present within the migration velocity model, the gathers deviate from this hyperbolic curvature into more complicated curvatures. When seismic data are imaged with these velocity errors, Stolt residual migration cannot fully correct for the focusing errors present within the image.

Figure 2.8 shows an example of an angle gather that has deviated from the hyperbolic curvature that can be corrected with Stolt residual migration. The angle gather was extracted from the same seismic image shown in Figure 2.3 but was migrated with a strong slow local velocity anomaly introduced into the migration velocity. The anomaly was $\sim$ 15% slower than the background and had a lateral extent of 7 km and a vertical extent of approximately 0.5 km. The rightmost gather shows the original angle gather (corresponding to the $\rho = 1$ image) and the gathers were computed after applying Stolt residual migration for ρ values between 0.9 and 1.0. Examining the events within the angle gather, it is apparent that while they curve upward due to the low velocity errors, at the larger angles they have deviated from a hyperbolic curvature and have flattened slightly. For lower values of ρ, it is clear that while

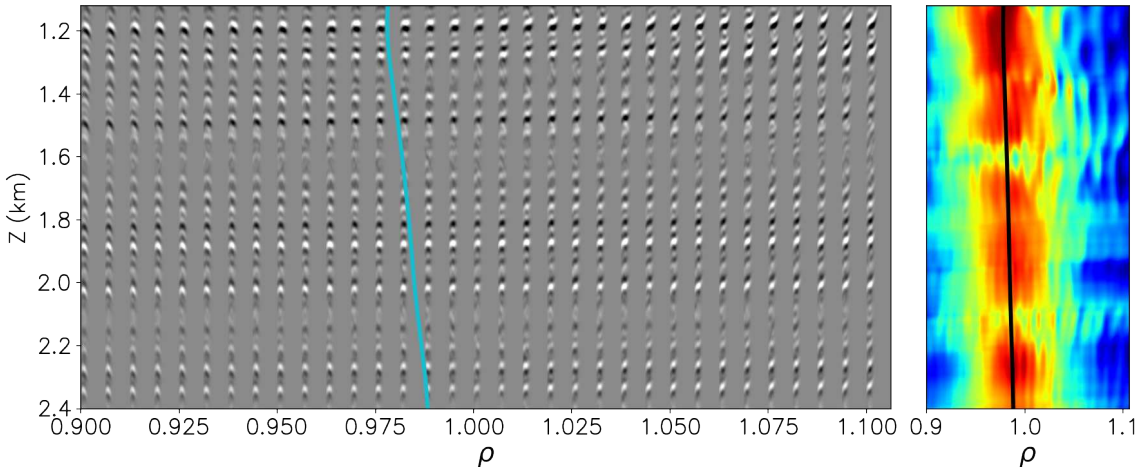


Figure 2.9: Computed ρ semblance of the angle gather extracted at $x \approx 4.5$ km, but now a mute has been applied to the angle gather simulating as if it had been acquired with a maximum offset of 1 km. Note in the left panel the lack of sensitivity in the residually migrated angle gathers to ρ which resulted in a very broad semblance trend in the right panel. [**CR**]

Stolt residual migration does correct for the hyperbolic curvature within the gather for $\rho \approx 0.98$, the far angles still curve downward and the events are not flat. This is an inherent limitation of the constant velocity ratio approach of Stolt residual migration and to overcome it, the velocity model must be updated via tomography or higher-order terms must be introduced into the residual Stolt operator. In spite of this limitation, Stolt residual migration has repeatedly demonstrated to be a useful tool for focusing velocity analysis and velocity inversion (Sava and Biondi, 2004b; Sava et al., 2005).

The second limitation of focusing analysis with prestack Stolt residual migration arises for data that have been acquired with limited illumination. A key element to the success of the estimation of $\rho(z)$ shown in Figure 2.7 is the sensitivity of the angle gather with a change in ρ. Observing the residually migrated angle gathers, in Figure 2.7, it is apparent that the larger angles are more sensitive to changes in ρ than smaller angles (thus leading to the hyperbolic curvature of the events). Therefore, a reduction in the range of illuminated reflection angles will reduce the sensitivity of

the angle gathers to velocity errors and lead to a lower resolution ρ semblance. To demonstrate this, I applied a mute to the angle gathers to simulate data that had been acquired with a maximum offset of 1 km. I then repeated the semblance calculation on these muted gathers and show the resulting computed semblance in Figure 2.9. Observing the angle gathers in the left panel, it is apparent that there is very little moveout sensitivity for different ρ values. This lack of sensitivity has resulted in a very broad trend in the computed ρ semblance shown in the right panel. The broad trend introduces considerable uncertainty when picking and can lead to errors in the estimation of $\rho(z)$. However, unlike the inherent limitation of Stolt residual migration to strong local velocity errors, this limitation can be overcome using the focusing of geological features within the x and z axes (the physical space of the image). For this particular example, the focusing of the diffractions along the fault trajectories provide key information for overcoming illumination issues. However, while it is possible to extract focusing information from the faults within the physical space of the image, it is considerably more difficult to automate than using only the moveout information provided by the angle gathers. The aperture-angle based approach is advantageous in that it is largely independent of the subsurface geology and therefore angle gathers will not drastically differ. Conversely, the physical-space focusing approach is entirely dependent on the subsurface geology and dipping layers, faults, salt bodies and other geological features must be considered. Due to this large variability of focusing within the physical space, it can be difficult to construct robust criteria for determining the relative focusing of an image. In the remaining sections of this chapter, I describe and provide an intuition behind a data-driven approach that I have developed that uses the feature-extracting capabilities of CNNs to determine the relative focusing of seismic images.

SEISMIC IMAGE-FOCUSING ANALYSIS WITH A CNN

For the remaining sections of this chapter, I describe a novel approach to extracting focusing information from prestack seismic images using a deep CNN. In recent years, CNNs have shown remarkable abilities at automating a wide range of computer vision

tasks (Krizhevsky et al., 2012; Taigman et al., 2014; Liu et al., 2016). They have been especially useful for tasks that are difficult to define (easy for humans but difficult to automate with an algorithm). Even more recently, CNNs have shown to be of use within the seismic interpretation community. Their powerful ability to learn features within seismic images has allowed them to detect subsurface structures in a wide variety of complex geological scenarios.

I extend this idea of geological feature detection to aid in the task of image-focusing analysis. I do this by training a deep CNN to classify geological features as focused or unfocused. Using a supervised-learning based approach, I provide prestack image patches to the CNN and train it to provide a score from zero to one indicating the focusing of the patch. While semblance provides a focusing measure for each image point using only the flatness of the angle gathers, the CNN uses all available information within the prestack image patch to provide a focusing measure of the image patch. In the following sections, I first describe the design of the CNN used for extracting focusing features necessary for determining the focusing of seismic images. Then, as I use a supervised-learning based approach for determining the filter coefficients of the CNN, I describe the training data creation and collection procedure used to create training datasets. With a training set, I then estimate the CNN filter coefficients by attempting to fit the training data using a binary cross entropy loss function as measure of the data fit. Finally, with the trained CNN, I perform image-focusing analysis on the unfocused image shown in Figure 2.5.

2D Neural network structure

Of the many possible choices for designing the architecture of the CNN, I chose a relatively simple feed-forward (i.e., a sequential chain of intermediate operations) architecture that is composed of two primary components: a feature extraction component and a classification component. The feature extraction component takes as input a prestack image patch (which in 2D has dimensions of (nγ, nz, nx), γ representing the reflection angle) and provides as output a feature vector consisting of 128

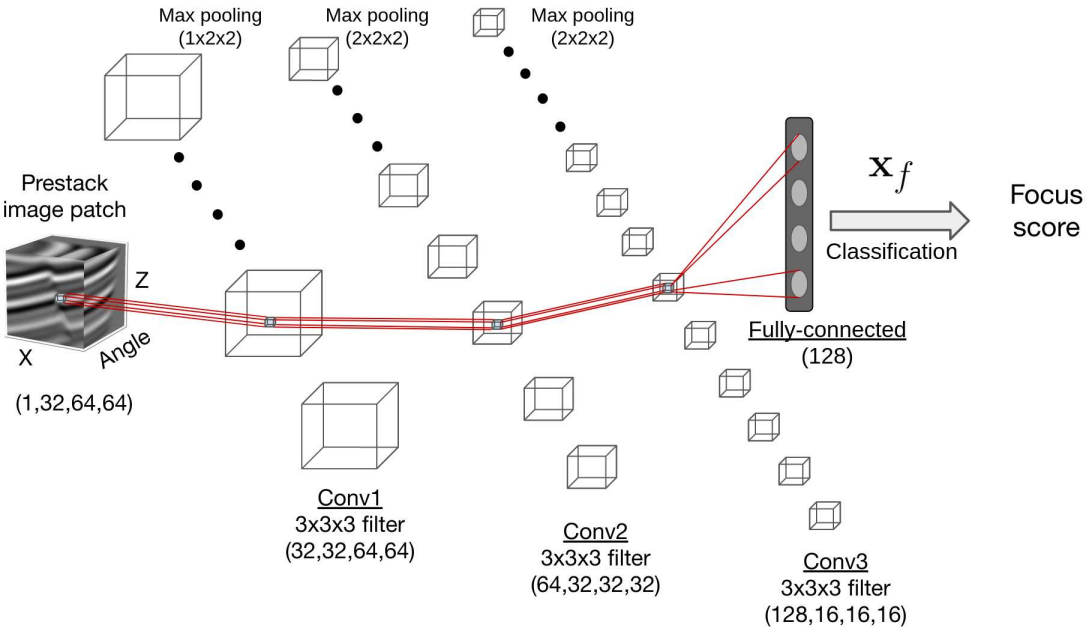


Figure 2.10: A schematic of the feature extraction component of the fault-focusing CNN. The shapes written beneath each of the operations denote the shapes of the outputs of the operation. [**NR**]

elements. Mathematically, I can describe this component as the following non-linear transformation:

$$\mathbf{x}_f = \mathbf{f}(\mathbf{I}; \mathbf{w}, \mathbf{b}), \tag{2.14}$$

where $\mathbf{x}_f$ is the output feature vector of length 128, $\mathbf{I}$ is an input prestack image patch and $\mathbf{w}$ and $\mathbf{b}$ are vectors that, respectively, contain the weights (i.e., CNN filter coefficients) and biases to be estimated and that parameterize the CNN. Figure 2.10 shows a schematic of the architecture of this component of the CNN. This component consists of three 3D convolutional, max-pooling and rectified linear unit (ReLU) blocks. All convolutional operations were performed using filters of size $3 \times 3 \times 3$ and apart from the first layer, all max-pooling operations were performed with a kernel of size $2 \times 2 \times 2$ which act to halve each dimension of the input features. (For the first block, I chose to not perform pooling along the angle axis). I will show in the final section of this chapter, these three blocks serve to extract the features from the image

that are most relevant for determining the focusing of the image patch. The output of these blocks is then reshaped and transformed into the output feature vector $\mathbf{x}_f$ via a fully-connected layer. The classification component then maps $\mathbf{x}_f$ to a scalar via an inner-product and then a sigmoid activation function is used to provide an output score:

$$s(\mathbf{I}; \mathbf{w}, \mathbf{b}, \boldsymbol{\theta}, \theta_0) = \frac{1}{1 + e^{-\boldsymbol{\theta}^T \mathbf{x}_f(\mathbf{I};\mathbf{w},\mathbf{b}) + \theta_0}}, \tag{2.15}$$

where $\boldsymbol{\theta}$ is a 128-element vector of parameters and θ_0 is a bias parameter to be estimated during training. s is then the output image focus score (bounded between 0 and 1) that is used as a measure for the focusing of the image patch.

While there are many possible choices for designs for CNNs, I chose the design of the neural network wanting to take advantage of all of the information provided within the full prestack image patch and used the VGG16 CNN (Simonyan and Zisserman, 2014) (a commonly-used architecture for computer vision tasks) as a rough guide. I found that the CNN could robustly do this by extracting three-dimensional features with 3D convolutional and 3D max-pooling operations. While I could add layers and complexity to the CNN, I desired to keep the total number of parameters relatively small. The feature extraction component consists of a total of 8,666,304 parameters (number of parameters in $\mathbf{w}$ and $\mathbf{b}$). Including the 129 parameters (128 weights from the final classification layer, $\boldsymbol{\theta}$, and the bias term), the CNN has a total of 8,666,433 trainable parameters. While maintaining minimal complexity does potentially sacrifice the feature extraction capability of the CNN, with fewer layers and computed features, the CNN is easier to troubleshoot and the intermediate feature maps are easier to interpret. Additionally, a small number of parameters in the CNN helps to prevent overfitting of the training samples and improve generalization, an especially important property for seismic imaging tasks for which typically the size of training sets is relatively small. Finally, maintaining a small number of parameters also greatly facilitates the extension to 3D images which as I will show in Chapter 4, requires the application of custom 4D convolutional layer operations which incur a significant computational cost when compared to 3D convolutional operations.

While there are a number of benefits of the chosen architecture as explained above,

there is certainly room for improvement as the architecture chosen for the task of image-focusing analysis is by no means the optimal architecture. One particular failing of this particular architecture is that it is entirely local (patch-based) and therefore does not consider the image as a whole nor the position of the patch within the image. While it has been shown for tasks such as image segmentation and video classification that CNNs tend to perform better when taking into account global contextual information (Wang et al., 2018; Lin et al., 2020), for the examples shown throughout this thesis, I did not find this to be necessary for the CNN to perform well at assessing the focusing of an image patch. Additionally, providing global contextual information into the CNN could help to reduce the dependency of the CNN on the choice of an adequate patch size for image-focusing analysis. However, I found that a patch size of 64 samples in each spatial direction and 32 angles was adequate for the images encountered in this thesis. While choosing an optimal CNN architecture was not a primary focus of the research presented in this thesis, it most likely would aid the CNN in generalizing to images that contain more complex geology and/or focusing errors and could be a direction for future research.

Training data creation

In order to estimate the parameters of the CNN ($\mathbf{w}$, $\mathbf{b}$, $\boldsymbol{\theta}$, θ_0), I use a supervised learning approach. This approach requires providing labeled training samples to the CNN and the CNN parameters are updated via an optimization procedure with the goal of minimizing a misfit between the CNN prediction on the image patches and the ground-truth labels. As at the time of writing this thesis, there do not exist any publicly available datasets that contained labeled focused and unfocused seismic image patches, for all of the examples in this thesis, I needed to create my own training datasets. While for many tasks in seismic imaging, creating labeled training sets can be a difficult task, it is relatively straightforward to create large labeled training sets of focused and unfocused seismic image patches from both field and synthetic seismic training images. This is because many focused and unfocused seismic images can be created from Stolt residual migration at relatively little computational cost.

As the end goal is to use the CNN for focusing analysis on real seismic images, ideally all training image patches would be taken from real seismic images that have undergone focusing analysis with Stolt residual migration. Creating labeled training patches from images that have undergone interpretative focusing analyses is not too difficult of a task, as the focusing analysis will either be performed directly by an experienced seismic processing geophysicist, or at least undergo a quality control (QC) step performed by an experienced seismic processing geophysicist. As I will demonstrate in Chapter 4, from a single 3D volume and with an appropriate patch size, thousands of labeled training focused and unfocused prestack image patches can be created.

However, there are challenges associated with creating labeled training image patches from real focused and unfocused images. Perhaps the most significant challenge is that of accurately labeling the images as focused and unfocused. Each seismic processing geophysicist contains their own biases when analyzing focused and unfocused images which can create inconsistently labeled patches (Liu et al., 2021). Without consistent and accurate labeling, the CNN will fit the inaccuracies in the training data and will not generalize well to patches not included in the training set. Additionally, since the majority of seismic images and data are owned and managed by large energy companies and geophysical contractors and are not publicly available, creating a large training dataset containing focused and unfocused images from multiple geologic regions is next to impossible for academic and other researchers. Therefore, for the field data examples in both Chapters 3 and 4 of this thesis, I perform focusing analyses on the entire images and manually label a subset of the patches generated from the focusing analyses. While this does provide labeled patches for training, the manual labeling procedure is time-consuming and additionally, using only a subset of the patches from the entire image may not provide enough training samples to adequately train the CNN (this is especially the case for 2D images).

To overcome these challenges associated with creating labeled focused and unfocused patches from real seismic images, I augment the field labeled training patches with unfocused and focused patches generated from synthetic seismic images. With

synthetic training images, I can establish an exact ground truth and create an accurately labeled training dataset. However, when using synthetic training images and then attempting to use the CNN on real seismic images, there can exist a large domain (covariate) shift between the two datasets (Gretton et al., 2009). This domain shift can significantly hinder generalization unless it is minimized by producing synthetic training samples that closely resemble the real training samples (i.e., that the differences between the training data distribution and test data distribution are minimized) (Shrivastava et al., 2017; Murez et al., 2018; Hoffmann et al., 2019). To accomplish this, for all CNN-based focusing analyses shown in this thesis, I create tailored synthetic focused and unfocused prestack training image patches. Using Stolt residual migration applied to focused synthetic images, I am able to create thousands of training image patches in a computationally-efficient manner. I describe the workflow I have developed for creating these image patches in the following section.

Synthetic training data creation

The first step I take in creating synthetic seismic images is to create a synthetic velocity model using the approach described by Clapp (2014). This method takes on a simplified basin-modeling approach to create geological models. Following this approach, I first create a sequence of depositional events of a specific thickness and of constant velocity. Then, I introduce small random variations within the layer to simulate fine random layering. Repeating this for many depositional events following where the chosen velocity is smoothly varied will result in a layered velocity model that follows a $v(z)$ velocity model but with small random variations within the different deposited layers. An example of a layered velocity that was created with this strategy is shown in Figure 2.11(a). Note that I created deposition events until $z = 1$ km and above that depth I have padded with zeros for display purposes. Then, continuing with the basin-modeling approach, I introduce a compressional event on the deposited layers that creates folding and undulation in the layers. To accomplish this I first create a smooth random function based on Perlin random noise (Perlin, 1985) and then compute depth shifts for all pixels that are then applied to the image via spline

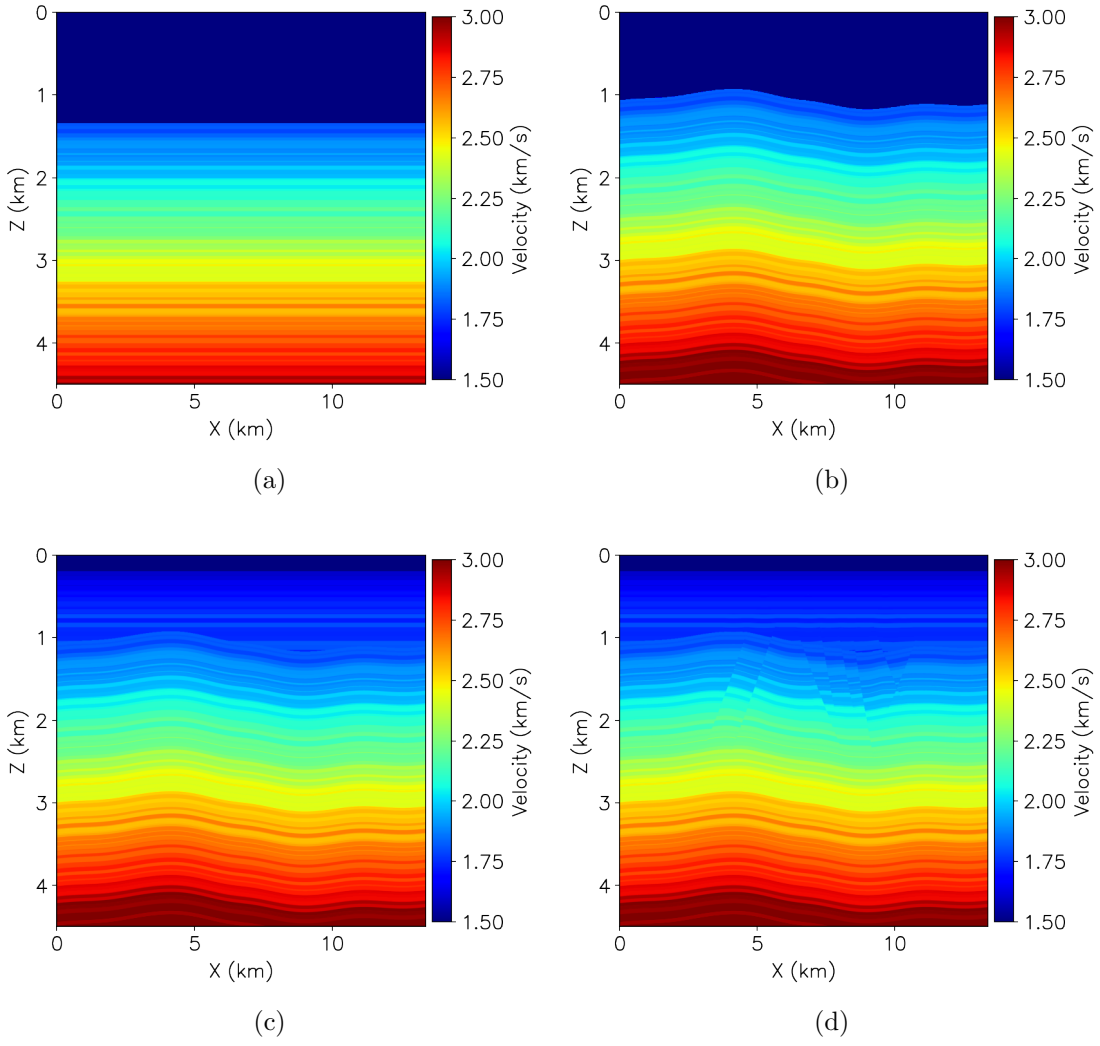


Figure 2.11: Example of building a synthetic velocity model using the simplified basin-modeling approach from Clapp (2014). (a) A sequence of constant velocity depositional events create a layered velocity model until 1 km depth, (b) a compressional event is introduced into the velocity model to simulate folding (c) more depositional events are introduced until reaching the water bottom and (d) a sequence of faulting events is introduced between 1 and 2.5 km depth. [**ER**]

interpolation. Figure 2.11(b) shows the velocity model after applying the shifts. Note now that the events are now undulating, mimicking geological folding. Then, I continue to simulate depositional events until the sea bottom which results in an angular unconformity at approximately 1 km as can be observed in Figure 2.11(c). Finally, I introduce a number of faulting events that occur within 1 - 2.5 km depth as are apparent in Figure 2.11(d).

To simulate faulted reflectivity within the models, I also use the approach described in Clapp (2014). This approach works by modeling the fault plane as a segment (arc) of a circle (cylinder in 3D) and the faulting process as a rotation about the center of the circle. This is illustrated in Figure 2.12. The fault is parameterized by the center point on the segment (x_f, z_f), an angle φ_0 (measured from vertical) and the radius of the circle r. Then, an angular shift is computed in order to rotate the points along the segment. This can be expressed as follows:

$$\varphi_s = \begin{cases} \varphi_c - \Delta\varphi(r_c, \varphi_c), & r_c > r\, p(\varphi_c), \\ \varphi_c + \Delta\varphi(r_c, \varphi_c), & r_c < r\, p(\varphi_c), \end{cases} \tag{2.16}$$

where φ_s is the angle of the new shifted point $(x_s,\, z_s)$, $\varphi_c = \tan^{-1}(z_c/x_c)$ is the angle of the original point (x_c, z_c), $r_c = \sqrt{x_c^2 + z_c^2}$ is the radius of the original point, r is the radius of the circle and $p(\varphi_c)$ is a smooth random function bounded between 0.9 and 1.1. The smooth random function is also calculated via the method of Perlin noise and I introduce it in order to allow the fault surface to deviate from strictly linear and circular trajectories. Figure 2.12 shows an example of this smooth random variation that perturbs the radius and therefore the trajectory of the fault segment. Note that the introduction of this smooth random function perturbs the original fault segment from the point (x_f, z_f) to the new point (x'_f, z'_f) as shown in the figure.

The angular shift $\Delta\varphi(r_c, \varphi_c)$ is constructed from the following quadratic decay function:

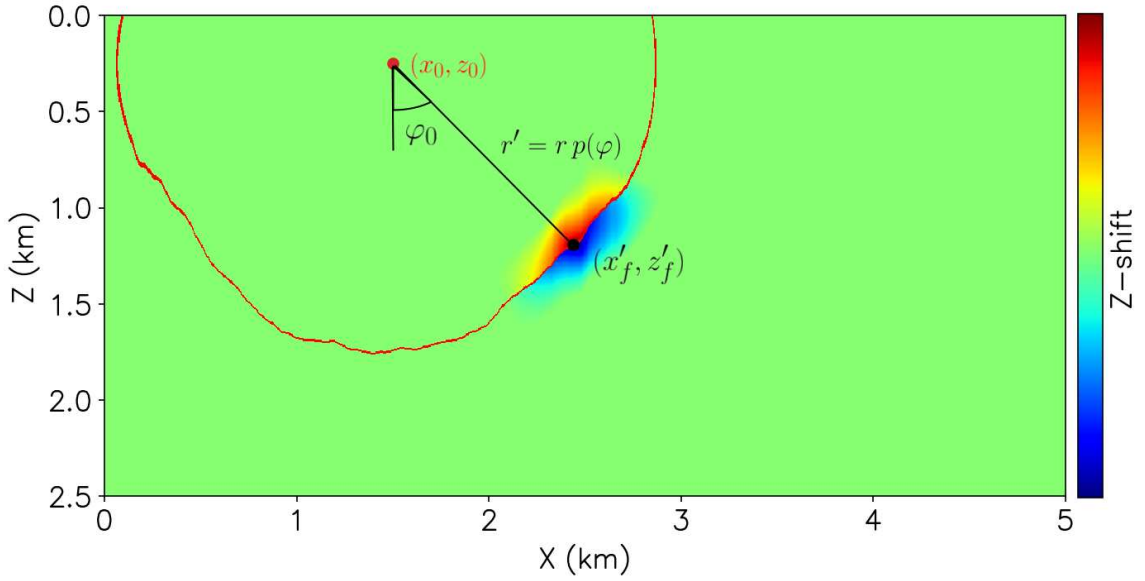
$$\Delta\varphi(r_c, \varphi_c) = \begin{cases} \Delta\varphi_0 \left(1 - \frac{|r_c - r\,p(\varphi_c)|}{r_d}\right)\left(1 - \frac{|\varphi_c - \varphi_0|}{\varphi_d}\right), & |r_c - r\,p(\varphi_c)| < r_d,\ |\varphi_c - \varphi_0| < \varphi_d \\ 0, & |r_c - r\,p(\varphi_c)| \geq r_d,\ |\varphi_c - \varphi_0| \geq \varphi_d \end{cases} \tag{2.17}$$

where r_d and φ_d are parameters that control the extent and the rate of the decay of $\varphi(r_c, \varphi_c)$ from the fault surface. Once the new angle φ_s has been obtained, the coordinates of the shifted point (x_s, z_s) can be computed as follows:

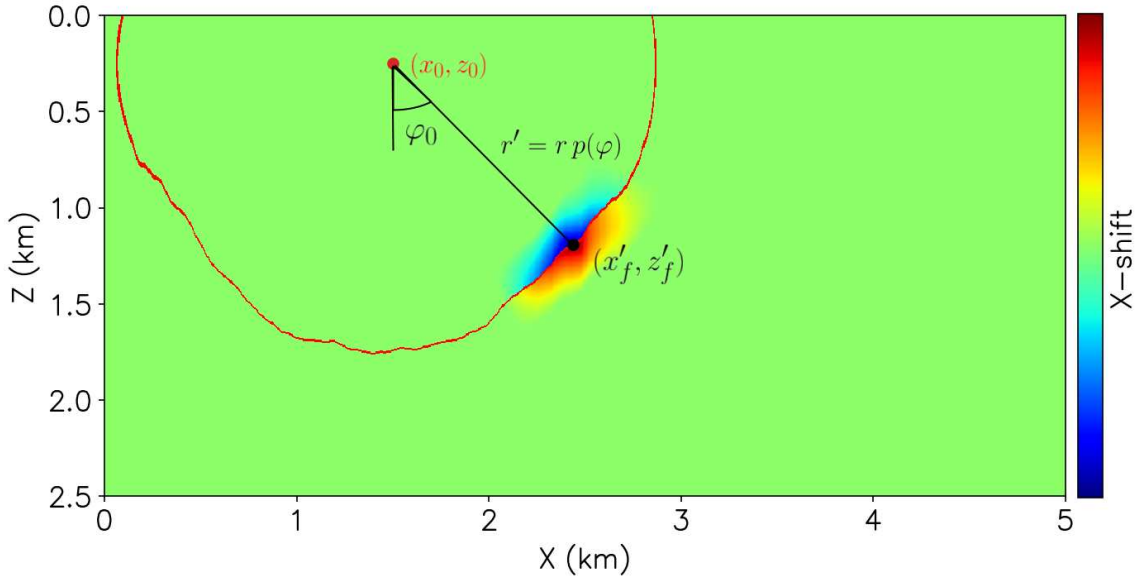
$$\begin{aligned} x_s &= r_c \cos\varphi_s + x_0 \\ z_s &= r_c \sin\varphi_s + z_0, \end{aligned} \tag{2.18}$$

where x_0 and z_0 are the origin of the circle as shown in Figure 2.12. With the coordinates of the shifted point (x_s, z_s), shifts from the original point to the new point can be computed and provided to a spline interpolator which numerically will perform the mapping between the points as was done for the compressional events in Figure 2.11(b). Panel (a) of Figure 2.12 shows the z-component and panel (b) shows the x-component of the computed shifts for a single fault example. While this approach for introducing faulting into the velocity model does not consider the geomechanics controlling actual faulting and fracturing in the subsurface, and therefore is limited in what it can model, it provides a computationally efficient approach to creating synthetic 2D and 3D faults within velocity and reflectivity models. As I will demonstrate in Chapters 3 and 4 of this thesis, this approach is accurate enough to create synthetic fault trajectories that can be used to train a CNN for segmenting faults within real seismic images. Moreover, the computationally efficient aspect of this approach enables the creation of many synthetic velocity models at low computational cost.

With the synthetic velocity model created, the next step in the synthetic training data creation workflow is to create a reflectivity model. To obtain this model, I simply compute a derivative along the depth axis of the velocity model using a first order forward finite-difference operator. This removes the zero frequency component



(a)



(b)

Figure 2.12: (a) Z-component and (b) X-component of shifts computed for a 2D fault example. (x'_f, z'_f) is the original fault position perturbed by the random function $p(\varphi)$ r' is the perturbed radius, r is the original radius of the circle and φ_0 is the unperturbed dip angle of the fault (in this case 45°). [**NR**]

of the velocity model and leaves the interfaces of which I desire to create an image. While ideally, the normal-incidence reflectivity would be obtained using a derivative computed along the axis normal to the dips of the geology present within the model, the majority of the synthetic models created in this thesis contained relatively shallow dips and I found that the depth axis was a reasonable approximation to the normal axis for computing the reflectivity. Now, with the computed reflectivity model I can create a simple synthetic band-limited seismic image by convolving with a Ricker wavelet. Depending on the frequency range of the real seismic image on which I desire to apply CNN-based focusing analysis, I modify the central frequency of the Ricker wavelet. For the synthetic example in this chapter, I use a central frequency of 20.0 Hz.

At this point in the workflow, I have created a synthetic stacked, focused seismic image. For training, I desire to train on focused and unfocused prestack seismic images. For a focused prestack image, the velocity model does not contain any errors and all energy will be focused at zero subsurface offset. Therefore, to create a prestack focused image, I can create a subsurface offset axis by padding the stacked image with zeros. Again, this padding will depend on the field data application, but for all examples in this thesis, I pad with 20 positive and 20 negative subsurface offsets. Then I convert this subsurface offset-domain image to the reflection angle domain which yields an image with fully-illuminated and flat angle gathers. As a final step, I introduce a mute along the angle gathers in order to mimic the effects of illumination that occur during migration. To create the unfocused image, the procedure is nearly the same as for the focused image, apart from one key step, which is to residually migrate the prestack focused subsurface offset domain-image with $\rho \neq 1$. This process simulates a migration with a constant velocity error and will move energy away from zero-subsurface offset and additionally will introduce diffraction events at the terminated reflectors created by the faulting in the model. Figure 2.13 shows an example of the stack of an unfocused training image obtained by residually migrating the focused prestack image for $\rho = 0.965$. The presence of the diffractions are obvious along the faults. Since this process of creating focused and unfocused images requires little computational cost, I can create many focused and unfocused synthetic seismic

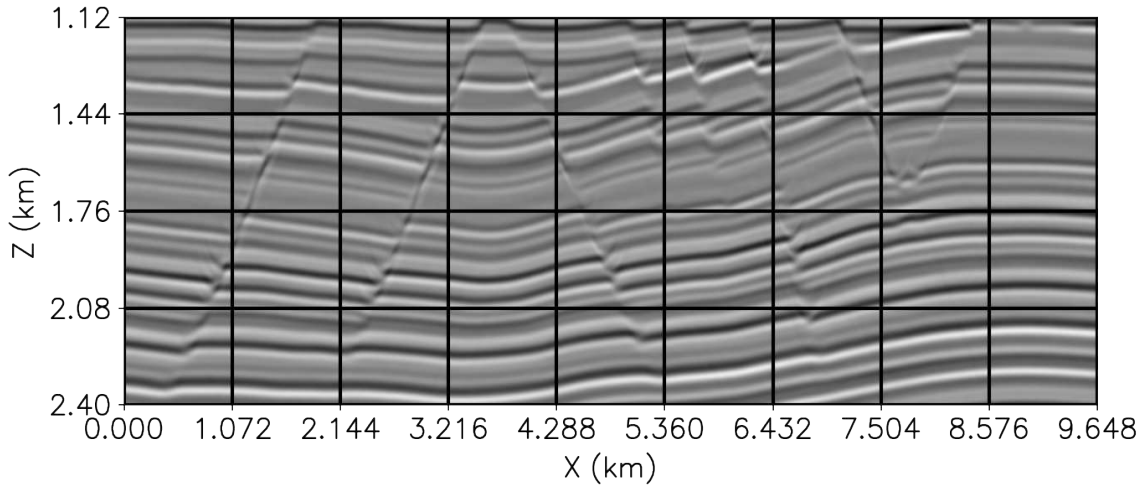


Figure 2.13: An example of synthetic unfocused image created by convolving the reflectivity with a wavelet and residually migrating the image with $\rho = 0.965$. Note that at the fault locations there exist overmigration smiles due to $\rho < 1$. The patch grid used for creating image patches is plotted on the image. This grid is the principal (unstrided grid). The strided grids would be shifted by half patch length in either x or z or in both directions. [**CR**]

images. This enables the creation of tailored training sets for different field data examples, which further minimizes the domain shift that is inevitable when training on synthetic data with the intent to predict on real data. To create pseudo-random training images, I randomly perturb the reflectivity (e.g., the compressional event), the positions, dips, throws of the faults and also the ρ value used for creating an unfocused image. The perturbations introduced into these parameters are bounded by the faults, geological dips and velocity errors observed and expected within the real seismic image.

The final step for creating training images used to train the fault-focusing CNN is to create image patches of size $32 \times 64 \times 64$ from the pseudo-random seismic images. The first step in the patch creation is to window the full focused and unfocused pseudorandom images to the region of interest as shown in Figure 2.3. Then, to create overlapping patches, I specify four different patch grids each with a stride specified in either the x or the z direction or in both x and z directions. Figure

2.13 shows a patch grid plotted on top of an unfocused pseudorandom image. For the remaining grids, I specified a stride of half the patch dimension (32 samples) in both x and z directions. Therefore, these grids would have been offset by half a patch length in both x and z directions. Note that in order to provide the CNN with all computed angles within the angle gather, I do not perform any patching along the angle axis. Additionally, as I desire that the CNN assess the focusing of image patches using both faults and the angle gathers, I keep only the patches that contain segments of the fault trajectory. I do this by using the fault trajectories computed during the model building process and selecting the patches that contain a specific number of pixels associated with a fault trajectory in the patch. After patching and filtering the patches based on the positions of the faults, a single focused or unfocused image will provide approximately 80 patches for the 2D synthetic example. Repeating this procedure for 150 pseudorandom unfocused and focused images yielded 10,240 patches in total for training. With all patches extracted from the images, each patch has its mean subtracted and is normalized by its standard deviation. Finally, all image patches from a focused image receive a label of one and all patches from an unfocused image receive a label zero. Figure 2.14 shows examples of focused and unfocused image patches used for training the CNN.

CNN Training

With a training set created, the weights of the CNN could then be estimated via an optimization procedure. This step is also referred to as the training of the CNN. Essentially the goal of training is to update the weights of the CNN such that when provided an input image from the training dataset, it can accurately predict the label. The first step in accomplishing this is to specify a loss function that provides a measure of the errors in the CNN prediction. While there are many choices for loss functions, a typical choice for a binary classification task (i.e., the labels are either zero or one) is a loss function known as a binary cross entropy loss function (also

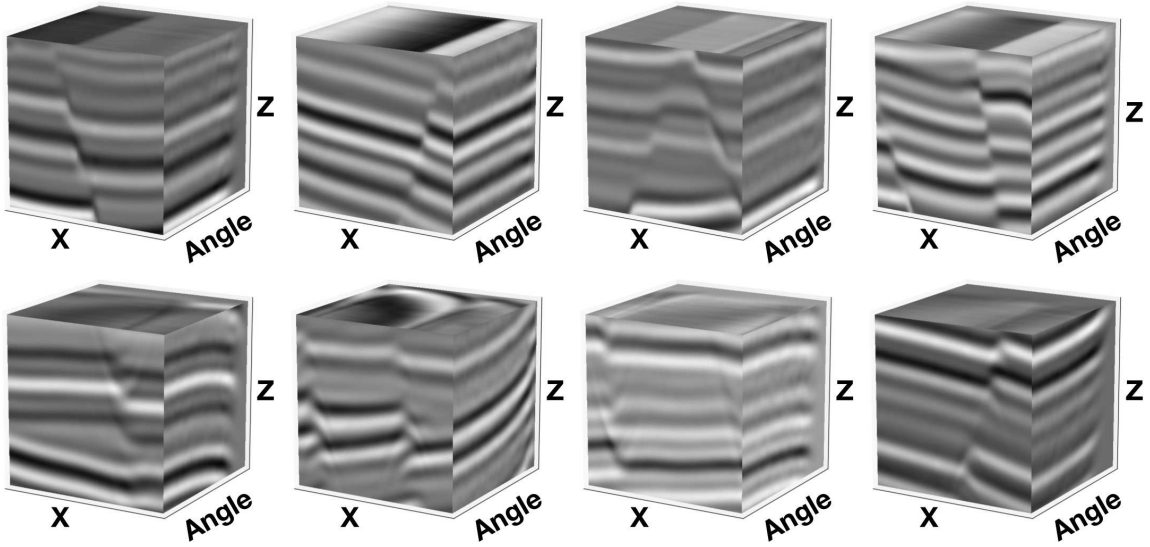


Figure 2.14: Focused (top row) and unfocused training patches used for training the fault-focusing CNN. [**NR**]

referred to as a log loss). It can be expressed as follows:

$$\mathcal{L}(\mathbf{w}, \mathbf{b}, \boldsymbol{\theta}, \theta_0) = -\sum_{i=1}^{M} y_i \log(s_i(\mathbf{I}_i; \mathbf{w}, \mathbf{b}, \boldsymbol{\theta}, \theta_0)) + (1 - y_i) \log(1 - s_i(\mathbf{I}_i; \mathbf{w}, \mathbf{b}, \boldsymbol{\theta}, \theta_0)), \tag{2.19}$$

where y_i is the label for the i-th training example and M is the total number of examples used for training. To provide a better intuition for this loss function, recall that the entropy of a discrete distribution $q(y)$ can be expressed as:

$$H(q) = -\sum_{i=1}^{N} q(y_i)\ \log(q(y_i)). \tag{2.20}$$

The entropy measure is commonly used in the field of information theory and is maximal for a uniform distribution and minimal for a deterministic distribution. In the case of a binary discrete random variable, where y can either be zero or one and $N = 2$, Equation 2.20 can be expressed as

$$H(q) = -q \log q - (1 - q) \log(1 - q). \tag{2.21}$$

The binary cross entropy is an entropy measure between two binary distributions and changes Equation 2.21 to:

$$H_r(q) = -q \log r - (1 - q) \log(1 - r), \tag{2.22}$$

where $H_r(q)$ is the binary cross entropy between discrete binary distributions q and r. We can now recognize that the summand of Equation 2.19 is the binary cross entropy measure between the binary distribution given by the labels and the binary distribution given by the CNN predicted focusing score. It is important to note that the cross entropy between two distributions will always be larger than the entropy of either of the distributions (Murphy, 2012). Therefore, the only way to minimize Equation 2.19 will be that the CNN predicted focusing scores mimic the labels associated with the training samples. This is equivalent to updating the CNN weights such that the CNN accurately predicts the training samples.

In order to find the optimal CNN parameters that minimize Equation 2.19, I follow what is considered standard practice in the fields of data science and deep learning and use a stochastic gradient-based optimization algorithm. These algorithms all rely on some form of the following update rule:

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \alpha \hat{\nabla}_\theta \mathcal{L}(\mathbf{w}, \mathbf{b}, \boldsymbol{\theta}, \theta_0), \tag{2.23}$$

where $\boldsymbol{\theta}_k$ are the estimated parameters at the kth iteration, α is known as the learning rate (i.e., step size) and $\hat{\nabla}\mathcal{L}$ is an estimate of the gradient of the loss. Note that for simplicity I only consider the gradient and the parameter updates associated with the final output layer. When I perform the actual optimization, $\mathbf{W}, \mathbf{b}, \boldsymbol{\theta}$ and θ_0 are all updated. While the exact gradient can be used to update the parameters, it requires a forward prediction over all samples in the training set. This is illustrated in the following expression for the exact gradient:

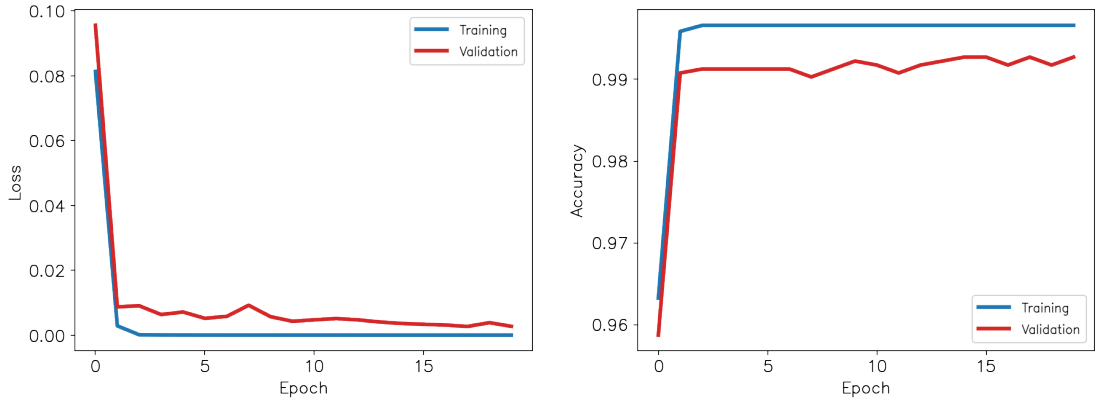
$$\nabla_\theta \mathcal{L} = \sum_{i=1}^{M} (s_i(\mathbf{I}; \mathbf{w}, \mathbf{b}, \boldsymbol{\theta}, \theta_0) - y_i) \mathbf{x}_{f_i}(\mathbf{I}; \mathbf{w}, \mathbf{b}). \tag{2.24}$$

While this is the gradient that mathematically should be used to update the parameters $\boldsymbol{\theta}$, the cost of computing the error term for each and every sample before updating the weights can be prohibitive for large datasets. Instead, an approximate gradient can be computed as follows:

$$\hat{\nabla}_{\theta}\mathcal{L} = \sum_{i=1}^{B}(s_i(\mathbf{I};\mathbf{w},\mathbf{b},\boldsymbol{\theta},\theta_0) - y_i)\mathbf{x}_{f_i}(\mathbf{I};\mathbf{w},\mathbf{b}), \tag{2.25}$$

where B is known as a batch size and is greater than zero and much smaller than M. In addition to the computational cost advantages provided from using this inexact gradient, this approach also has some additional advantages. As the parameters are updated much more frequently, albeit with inexact gradients, stochastic gradient descent algorithms can exhibit faster rates of convergence than gradient descent algorithms that use the exact gradient (Keskar et al., 2016). Moreover, while all gradient-based methods face the problem of falling into local minima, using a noisy gradient may aid to escape these local minima (Kleinberg et al., 2018).

Of course, as stochastic gradient descent uses the inexact gradient for updating the parameters, there are also a number of convergence-related disadvantages to using this approach. Perhaps the most significant of these issues is that even in the case of strong convexity, stochastic gradient descent algorithms do not guarantee convergence to the solution, but rather only within a neighborhood of the solution (Bottou et al., 2018). Practically, this also makes it difficult to establish a stopping criterion during CNN training. Additionally, due to the use of an inexact gradient, an optimal step size cannot be estimated at each iteration of the optimization, but rather a choice for a learning rate must be provided as input to the algorithm. Naively, one then can use this fixed learning rate to update the parameters at each iteration of the optimization until a stopping criterion has been satisfied. Due to this reliance on an adequate choice of a learning rate, the rate of convergence and stability of the optimization become largely dependent on the choice of this parameter as large learning rates can lead to highly oscillatory and unstable convergence and small learning rates can lead to very slow convergence. To help avoid this issue, I use the ADAM optimization



(a) (b)

Figure 2.15: The "learning curves" obtained during the training of the CNN on synthetic training image patches computed from convolution and residual migration. (a) The training and validation loss vs training epoch and (b) the training and validation accuracy vs training epoch. The CNN is able to fit both the training and validation datasets with 99% accuracy indicating that it can distinguish between focused unfocused seismic image patches with faults. [**CR**]

algorithm (the name originating from the term "adaptive moment estimation") which incorporates previous gradients and previous squared gradients in addition to the current gradient into the update rule (Kingma and Ba, 2014). While this does help to considerably improve convergence of the stochastic gradient descent, it is still sensitive to the learning rate which thus requires tuning for each new model to be trained.

It is also important to note that an "iteration" of a stochastic gradient descent algorithm is not the same as an iteration of a traditional gradient descent algorithm due to the use of the inexact gradient. Typically, when training a model using stochastic gradient descent, an iteration or "epoch" of training has occurred when the model has encountered each sample in the dataset. Each update within an epoch is referred to as a training step. For the training dataset described above, consisting of 4,096 focused and 4,096 unfocused samples, I trained the CNN for 20 epochs with the ADAM optimization algorithm. I chose a batch size of $B = 20$ training images and a learning rate $\alpha = 1 \times 10^{-4}$. Figure 2.15(a) shows the computed loss at the end of each epoch

during training. To measure the generalization of the CNN (i.e., the extent to which it is overfitting the training data), at the end of each epoch, I evaluate the loss on 1024 focused samples and 1024 unfocused samples that are not included as part of the training. I denote this dataset as a validation set and the loss resulting from evaluating the CNN at the end of each epoch on this dataset is also plotted in Figure 2.15(a).

In addition to computing the loss during training, I also compute the accuracy of the predictions for all training samples as well as validation samples. The accuracy is computed using the following expression:

$$\text{Accuracy} = \frac{N_{11} + N_{00}}{N_{11} + N_{00} + N_{10} + N_{01}}, \tag{2.26}$$

where N_{11} is the number of samples that the network predicted as focused that were actually focused (true positives), N_{00} is the number of samples that the network predicted as unfocused and were actually unfocused (true negatives), N_{10} is the number of samples that the network predicted as focused that were actually unfocused (false positive) and N_{01} is the number of samples that the network predicted as unfocused that were actually focused (false negative). Figure 2.15(b) shows the computed accuracy at the end of each epoch on both the training and validation datasets. From these curves, it is clear that the CNN is able to accurately distinguish unfocused seismic image patches from focused image patches.

While the accuracy is generally informative of the ability CNN to accurately fit the training and validation data, other metrics are often employed to further understand the predictive capabilities of the CNN. Two of these metrics that I compute here are the true and false positive rates (also referred to as recall and specificity). These two metrics can further help to understand where the network is failing to correctly classify certain image patches and are computed as follows

$$\text{TPR} = \frac{N_{11}}{N_{11} + N_{10}}, \tag{2.27}$$

$$\text{FPR} = \frac{N_{00}}{N_{00} + N_{01}}, \tag{2.28}$$

where TPR and FPR are the true and false positive rates respectively. For the validation dataset, I found that the $TPR = 1.0$ and that the $FPR = 0.995$ which indicates that the CNN is able to accurately distinguish focused seismic image patches 100% of the time, and 99.5% of the time can distinguish the unfocused seismic image patches. While for certain patches it can be difficult to assess the exact cause of the inaccurate prediction, in general this is due an image patch containing a small portion of the fault, a large mute, or weak velocity error therefore resulting in insufficient focusing information for the CNN to accurately assess the focusing of the image patch.

While the previously computed metrics above the overall performance of the CNN on synthetic focused and unfocused seismic images created with convolutional modeling and residual migration, the main purpose of the CNN will be to perform focusing analysis on real seismic images. As real seismic images contain noise, complex geology and illumination effects that can be very difficult to model in synthetic images, for each of the focusing analysis examples shown in this thesis, I also perform a secondary stage of training on a dataset that is much more representative of the real seismic image patches that will be provided to the CNN for focusing analysis. For the migrated image shown in Figure 2.5 on which I will perform CNN-based focusing analysis in this chapter, I created an additional training data set composed of unfocused and focused seismic image patches, created from images obtained via wave-equation depth migration. To create the images, I first created synthetic velocity and reflectivity models using the same method as described above. Then, for each pseudorandom reflectivity and velocity model, I performed linearized modeling and migration and converted the image to angle to create a focused migrated seismic image. To create an unfocused image, I took an approach similar to the convolutional approach in that I applied prestack Stolt residual migration for $\rho \neq 1$ which created a constant velocity defocused image. I did this for a total of 100 images and formed patches from each of these images which resulted in a total of 600 focused and 600 unfocused prestack image patches. During training, I used a total of 1,000 of these images for training and 200 for validation.

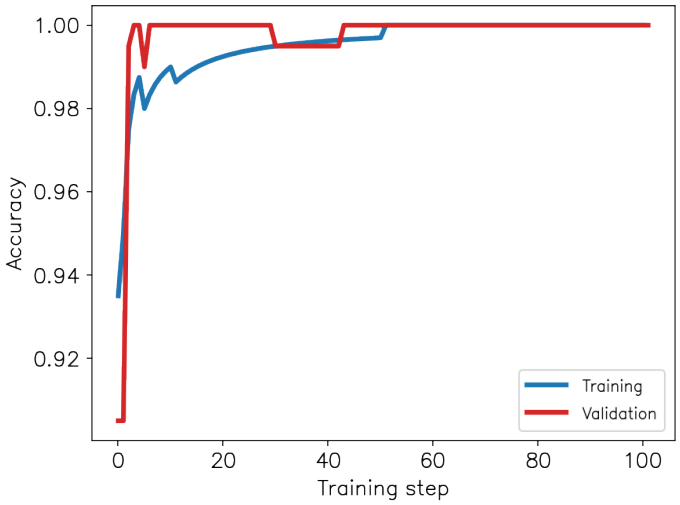


Figure 2.16: Accuracy learning curve obtained from training the CNN on 1000 focused and unfocused seismic image patches created via wave equation migration. [**CR**]

With the training dataset of focused and unfocused image patches created from wave-equation migration, I then performed a second stage of training of the CNN where I used the estimated weights and biases from the first stage of training as the initial model. This allowed the CNN to generalize to the validation dataset with a relatively small set of training images and few epochs. For this stage of training I also used the ADAM optimization algorithm with the batch size and learning rate as I used for the first stage of training but trained for only two epochs. Figure 2.16 shows the computed training and validation accuracies during all 100 training steps (1000 total examples and a batch size of 20 resulted in the 100 training steps). Note that the initial validation accuracy was only 90% but then increased to 100% after the 100 training steps.

CNN prediction as a focusing measure

With the CNN trained with both the data from convolutional modeling and the data from wave-equation migration, it was ready to be used to perform focusing analysis on the residual migration images shown in Figure 2.6. The first step in performing the CNN-based focusing analysis was to window each residually migrated image within the region of interest (shown in the yellow box in Figure 2.3) and then form image

patches of size 64×64 samples within this region of interest. Given that each image provided 119 patches, this resulted in a total of 119×161 ρ patches = 19,159 total patches for focusing analysis. Then using the fault locations from the velocity model building procedure, I assign a value of $\rho = 1$ to all patches that do not contain a significant portion of a fault (determined by a threshold on the number of pixels within the patch.) Note that for a real seismic image, this requires the fault locations which are not known *a priori*. For the real seismic image applications in Chapters 3 and 4 of this thesis, I perform fault segmentation on the seismic image to obtain the fault locations which can then be used for patch selection.

To then use the CNN for image-focusing analysis on these patches, I collect the image patches that correspond to the same spatial location, but vary with ρ into spatial patch groups. Then, for each patch within a spatial patch group, I perform a prediction using the CNN and save the computed focusing scores for all ρ values. Figure 2.17 shows the computed focusing scores reconstructed from all patches and smoothed with a triangular smoother of length 0.5 km in the x direction and 0.15 km in the z direction. In order to provide spatial context, the focusing scores are superimposed on the residually migrated images. When the faults become better focused and the power of the stack larger, the focusing score becomes larger. This is apparent for the bottom portion of the leftmost faults in the image (below $z = 2$ km). For the $\rho = 0.98$ image (Figure 2.17(a)) this portion of these faults is overmigrated and hence has received a low focusing score. However, for the image corresponding to $\rho = 0.995$ (Figure 2.17(d)), there is less overmigrated energy and therefore the faults have received a higher focusing score. With the computed focusing scores for all ρ values, I then selected the ρ value that provided the highest focusing score. Mathematically this can be expressed as

$$\rho_{ij} = \underset{\rho}{\text{argmax}}\ s(\mathbf{I}(\rho)_{ij}; \mathbf{w}, \mathbf{b}, \boldsymbol{\theta}, \theta_0), \tag{2.29}$$

where $\mathbf{I}(\rho)_{ij}$ is the collection of residually migrated image patches within the spatial patch group positioned at the ith patch along the z-axis of the patch grid and the jth patch along the x-axis of the patch grid. Performing this operation for all spatial

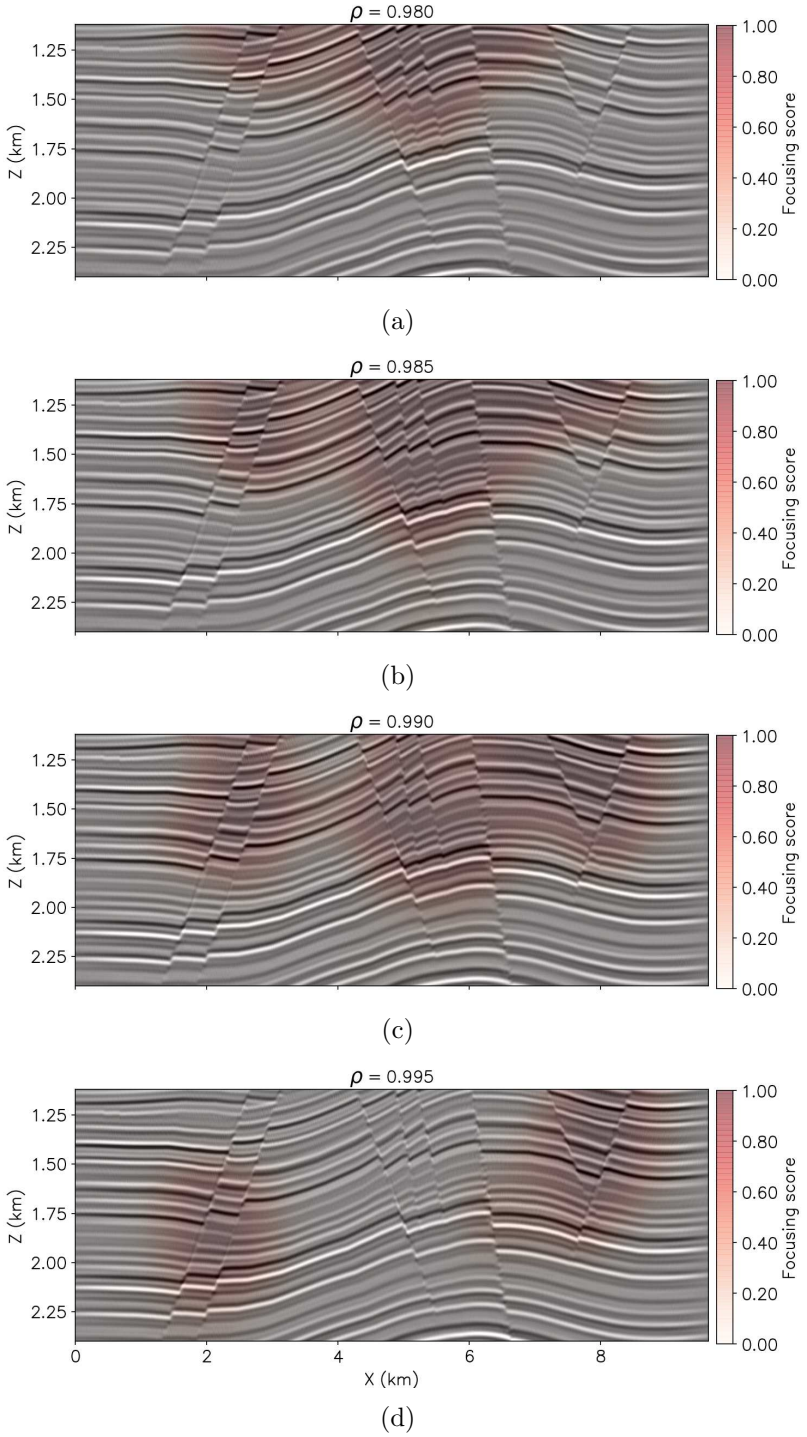


Figure 2.17: Focusing scores computed as a result of CNN-based focusing analysis superimposed on a selection of residual migration images. The larger focusing score values within certain ρ images correspond to spatial regions where the CNN determined that the image was better focused. [**CR**]

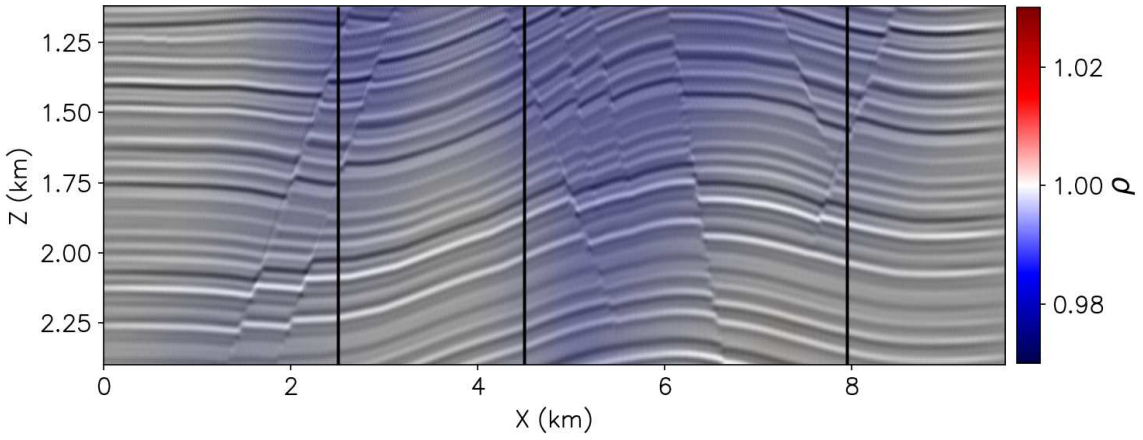


Figure 2.18: Estimated $\rho(z, x)$ from CNN-based focusing analysis superimposed on the unfocused image shown in Figure 2.5. The three vertical bars indicate the spatial locations used for comparing the $\rho(z, x)$ from CNN-based focusing analysis and the $\rho(z)$ picked from semblance panels. [**CR**]

patch groups then results in ρ values for all patches within the patch grid. To obtain a smooth estimate of ρ for all points in the image, I assign the estimated ρ_{ij} to all pixels within the patch and then reconstruct all patches which yields a $\rho(z, x)$ with constant valued patches. Note that in the spatial positions where the patches overlap, average all computed values from each overlapping patch that contributed to that spatial location. With the reconstructed $\rho(z, x)$ I then smooth it with a triangular smoother of length 0.5 km in the x direction and 0.15 in the z direction. Figure 2.18 shows the estimated $\rho(z, x)$ superimposed on the unfocused image. It is readily apparent the CNN has determined that ρ values between 0.98 and 0.99 will provide the best focusing of the faults. These values of ρ are consistent with what was visually observed in the residual migration images shown in Figure 2.6. It is also apparent from the estimated $\rho(z, x)$ that larger velocity errors are present towards the top of the region of interest, therefore requiring lower values of ρ to improve the focusing of those image points. Note that in between $x = 4$ and 6 km, the CNN has also determined that lower values of ρ are needed to improve the focusing of the image.

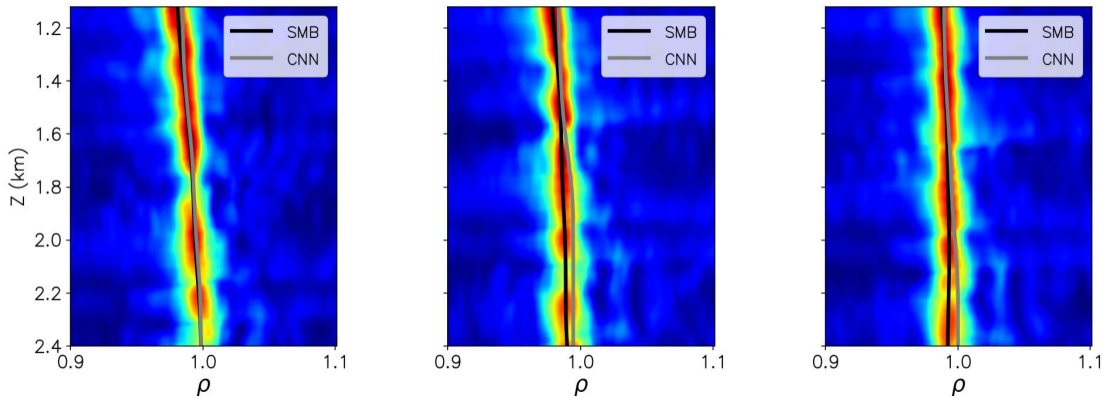


Figure 2.19: Comparison of estimated ρ_{CNN} (gray curve) and ρ_{SMB} (black curve) at (a) $x = 2.5$ km, (b) $x = 4.5$ km and (c) $x = 7.95$ km. For all panels the estimated ρ_{CNN} is in good agreement with the picked ρ_{SMB} at the fault positions at each of the three spatial locations (denoted by the vertical bars in Figure 2.18). Away from the fault locations, ρ_{CNN} diverges slightly from the picked ρ_{SMB}. [**CR**]

To further verify the estimated $\rho(z, x)$ from the CNN-based focusing analysis, I can compare it with the picked $\rho(z)$ from semblance panels at various spatial locations. To avoid confusion, from this point in the chapter I will refer to the $\rho(z, x)$ estimated from the CNN as $\rho_{\text{CNN}}(z, x)$ and the $\rho(z)$ picked from semblance panels as $\rho_{\text{SMB}}(z)$. Figure 2.19 shows three semblance panels computed at $x = 2.5$ km, $x = 4.5$ km and $x = 7.95$ km as indicated by the vertical bars shown in Figure 2.18. Each panel shows the $\rho_{\text{SMB}}(z)$ that was picked directly from the computed semblance panel and additionally, the $\rho_{\text{CNN}}(z)$ extracted at the corresponding spatial location. Note that in general for each of the three panels, there is good agreement between the picked $\rho_{\text{SMB}}(z)$ and the estimated $\rho_{\text{CNN}}(z)$ despite that $\rho_{\text{CNN}}(z, x)$ is unaware of the semblance panel. As the CNN-based focusing analysis was performed only at the fault locations, it is apparent that for each of the three panels, the two $\rho(z)$ best agree at the depths that correspond to fault locations within the image. This also explains why $\rho_{\text{SMB}}(z)$ and $\rho_{\text{CNN}}(z)$ have the best agreement in Figure 2.19 as for all depths, nearly each patch contains a portion of the fault at $x = 2.5$ km. It is important to keep in mind that while for this example, the agreement between ρ_{SMB} and ρ_{CNN}

helped to validate the estimated $\rho_{\text{CNN}}(z, x)$ from the CNN-based focusing analysis, ρ_{CNN} need not (and in some cases should not) agree with the picked ρ_{SMB}. This is due to the limitations of residual focusing analysis with semblance that were discussed previously in this chapter and that will be further demonstrated with synthetic and real data examples in Chapters 3 and 4 of this thesis. Also within those chapters, I will provide additional methods that can be used to better compare the accuracy of the focusing analysis performed by a semblance-based focusing analysis and a CNN-based focusing analysis.

UNDERSTANDING THE FAULT-FOCUSING CNN

While in the above sections I presented the theory of the fault-focusing CNN and also demonstrated that it can extract focusing information from seismic images, I did not provide information regarding the inner-workings of the CNN and up to this point, have essentially presented the CNN as a "black box". While the entirely data-driven approach to the CNN is advantageous in that it does not rely on specific rules for assessing the focusing of seismic images, it can be difficult to understand how the CNN can successfully distinguish focused images from unfocused images. Moreover, an understanding of how the CNN works can aid in debugging and troubleshooting cases in which it fails.

In the final section of this chapter, I attempt to unpack the fault-focusing CNN with the hopes to provide the reader with an improved understanding of its inner-workings. I will first perform the most straightforward analysis of examining the computed feature maps from a single seismic image patch. Then, I will apply three additional techniques that attempt to reconstruct images that maximize the output and inner activations of the CNN. Through the interpretation of the estimated features and images from these approaches, I will show that the CNN attempts to extract discontinuities (e.g., faults) in order to classify an image as focused or unfocused.

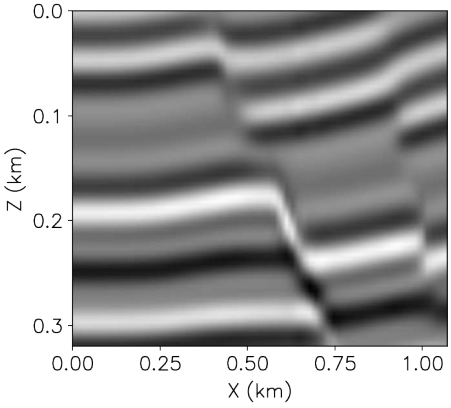


Figure 2.20: The focused image patch used for analyzing computed feature maps in Figures 2.21, 2.22, 2.23 and also for the saliency optimization shown in Figure 2.25 [**ER**]

Direct of feature maps

As the primary goal of CNNs is to extract features from images, the most straightforward way to understand how the fault-focusing CNN can distinguish between focused and unfocused images is to directly visualize the intermediate feature maps computed at the different stages of the CNN. Recall from Figure 2.10 that the fault-focusing CNN is composed of three convolutional blocks where each block consists of a convolution operation, an element wise ReLU activation function and then a max-pooling operation The convolution operation, actually numerically implemented as a cross-correlation, performs a cross correlation of the CNN weights with the input image or intermediate features. Mathematically, for the first block in the CNN this can be expressed as

$$\mathbf{h}_j^{(1)} = \mathbf{b}_j^{(1)} + \mathbf{w}_j^{(1)} \star \mathbf{I}, \tag{2.30}$$

where $\mathbf{I}$ is the input image patch, $\mathbf{h}_j^{(1)}$ is output feature map, $\mathbf{w}_j^{(1)}$ are the weights of the $\mathbf{b}_j^{(1)}$ are the biases for the jth output channel and $\star$ indicates the cross correlation operation. The superscript indicates that these are the intermediate features, weights and biases for the first block. After the computed cross-correlation operation, the feature maps undergo a max pooling operation which operates on a 2×2 region of the image and reduces it to a single value by computing the maximum over the four samples in the region. This acts to downsample the image by half along each dimension and enables the extraction of longer-wavelength features within the image.

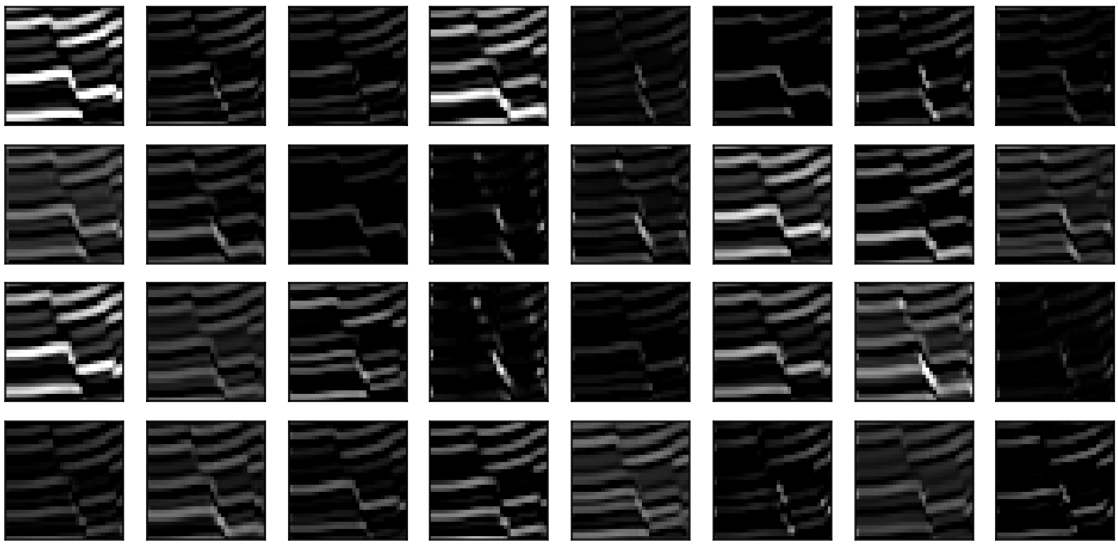


Figure 2.21: Computed feature maps from the first convolutional block (channels of $\mathbf{a}_j$). Note that in a number of the channels, the horizons have been eliminated leaving primarily the fault segment. [**CR**]

I denote the max-pooled feature maps as $\widetilde{\mathbf{h}}_j^{(1)}$. Finally, the output of the max-pooled feature map is provided as input to a ReLU activation function which can be expressed as:

$$\mathbf{a}_j^{(1)} = \max(0, \widetilde{\mathbf{h}}_j^{(1)}), \tag{2.31}$$

where the max function is applied element-wise to $\widetilde{\mathbf{h}^1}$.

As is shown in Figure 2.10, for the first convolutional block, I let the number of output channels to be 32. Figure 2.21 shows the result of the application of the first convolutional block to the image patch shown in Figure 2.20. Observing the different channels shown in Figure 2.21, it is apparent that in many of the images, the horizons have largely been eliminated and the fault segment has been accentuated. Also note that due to application of the ReLU activation function, all negative amplitudes within the images have been clipped and set to 0.0.

For the two additional convolutional blocks, the cross-correlation operation changes as the input to the cross-correlation no longer is a single channel image, but contains

many channels. To perform the cross-correlation across all input channels Equation 2.30 changes to:

$$\mathbf{h}_j^{(k)} = \mathbf{b}_j^{(k)} + \sum_{i=1}^{N_c} \mathbf{w}_{ji}^{(k)} \star \mathbf{a}_i^{(k-1)} \tag{2.32}$$

where N_c is the number of input channels (channels of $\mathbf{a}_i^{k-1}$). Using Equation 2.32, and performing the same max-pooling and ReLU operations as were used in the first convolutional block, the CNN computes both $\mathbf{a}^{(2)}$ and subsequently $\mathbf{a}^{(3)}$ as the next sets of feature maps. These feature maps are displayed in Figure 2.22 and 2.23 respectively. As was observed with the computed feature maps shown in Figure 2.21, both convolutional blocks two and three continue to accentuate the fault position and remove the horizons within the image. However, this is difficult to precisely assess as the images after each convolutional block are halved in size in both x and z directions resulting in low-dimensional images (8×8 after the third convolutional block).

For the final step of the CNN, the image of dimension $64 \times 8 \times 8$ is then formed into a vector of length 8,192 and is transformed into a vector of length 128 via the fully-connected layer. This can be expressed as the following affine transformation:

$$\mathbf{h}^{(4)} = \mathbf{W}\mathbf{a}^{(3)} + \mathbf{b}^{(4)}, \tag{2.33}$$

where $\mathbf{W}$ contains the weights of the fully-connected layer (each row of $\mathbf{W}$ corresponds to a neuron within the fully-connected layer). This final intermediate feature map is then provided to a ReLU activation function. The output of this ReLU activation function is the $\mathbf{x}_f$ vector as described above in the CNN architecture section of this chapter. Unfortunately, with the flattening of the feature maps and the introduction of the fully-connected layer, the spatial structure of the images is lost and it is considerably more difficult to interpret the computed feature maps.

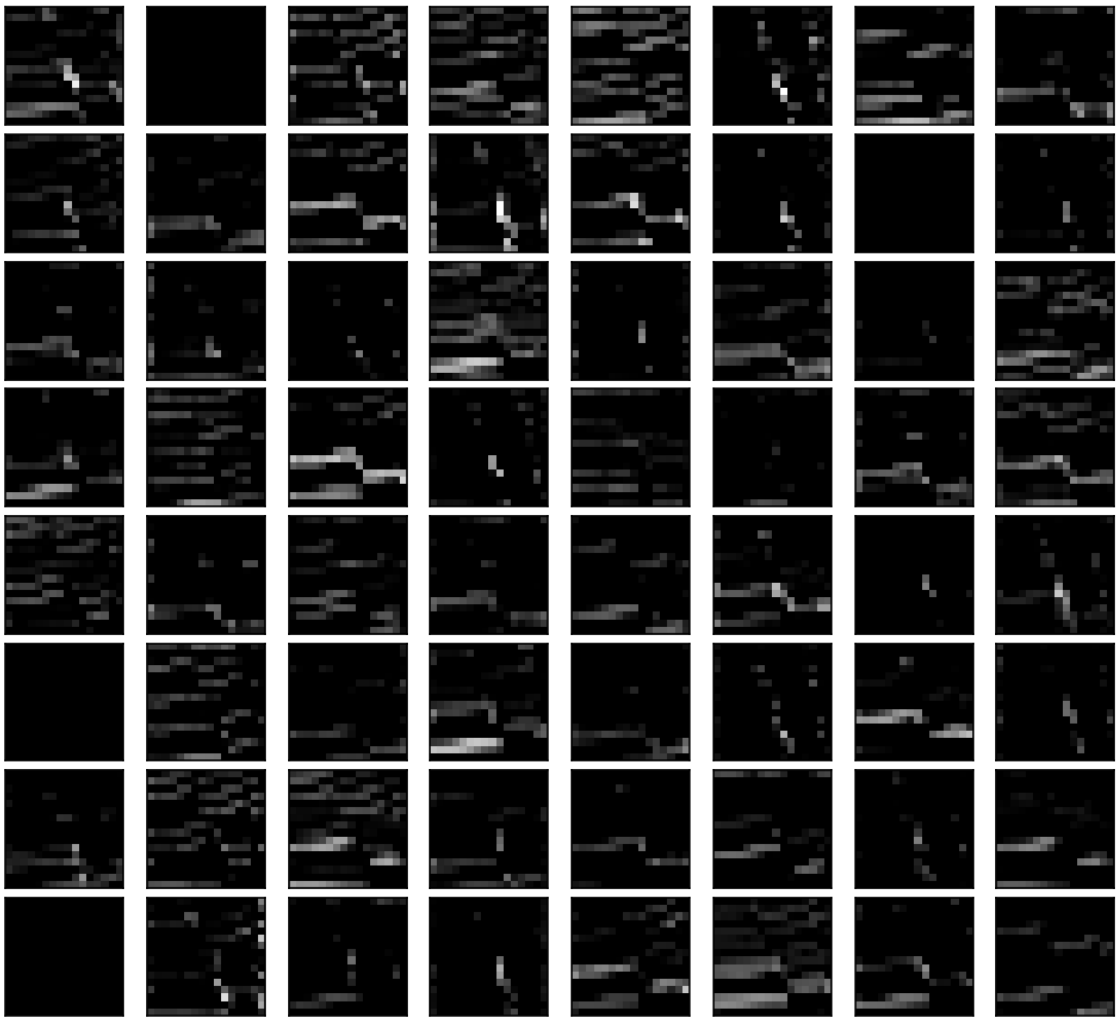


Figure 2.22: Feature maps computed from the second convolutional block. A total of 64 channels and each image is of size 16×16. As for the feature maps from the first block shown in Figure 2.21 note that horizons have largely been removed from most channels and the pixels related to the fault positions have the highest relative amplitudes (all images are displayed with the same dynamic range). [**CR**]

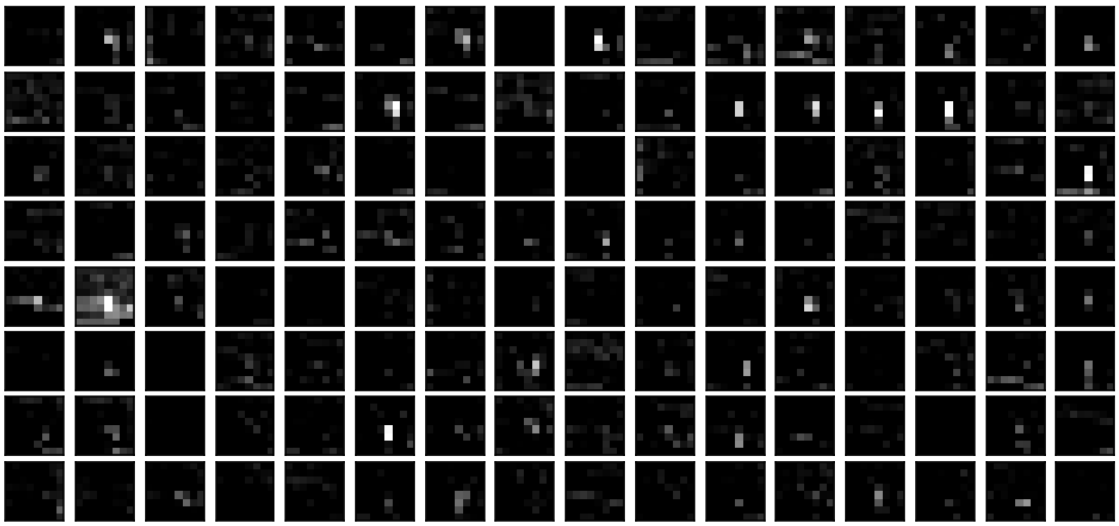


Figure 2.23: Feature maps computed from the third convolutional block. A total of 128 channels and each image is of size 8×8. Due to the small image size, it is difficult to assess precisely what information is contained within these images, but qualitatively, it appears that large amplitude pixels within many of the channels are due to the strong amplitudes from the fault positions accentuated from the previous convolutional blocks. [**CR**]

Reconstruction-based approaches for CNN understanding

While in the previous section I directly visualized the intermediate feature maps computed within the different stages of the fault-focusing CNN, the interpretation was qualitative and also can be difficult to properly assess for the lower-resolution feature maps computed at the later stages of the CNN. In this section, I will compute additional attributes from the CNN that enable a more precise interpretation of what features the fault-focusing CNN extracts from the image to assess the focusing of the image.

One quantitative approach that can aid in improved understanding of the feature extraction within CNNs is to estimate an image that maximizes the output of the trained CNN (Simonyan et al., 2014). As an image resulting from such a procedure would maximize the CNN output, it would also contain the features that CNN uses in order to distinguish focused and unfocused images. This image estimation problem

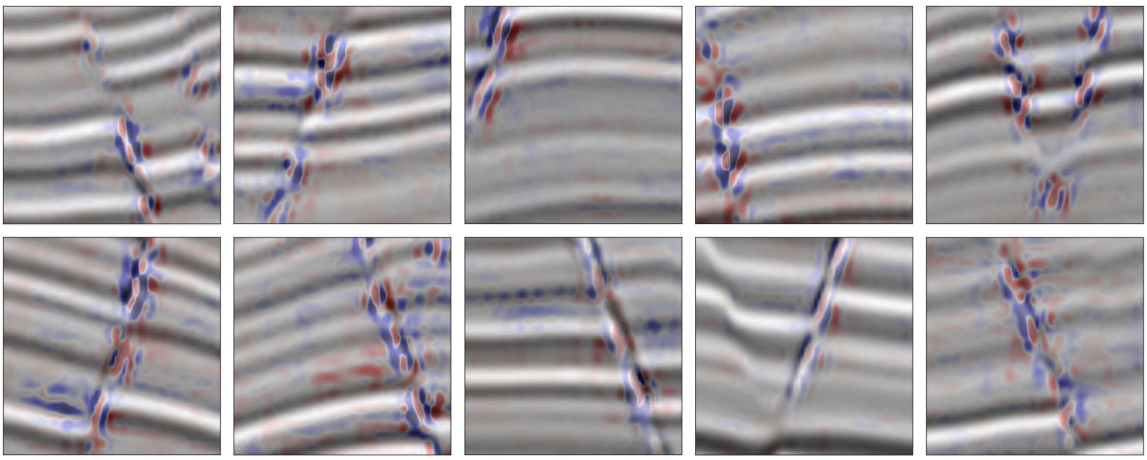


Figure 2.24: Image-specific saliency maps computed for the fault-focusing CNN. The saliency maps (shown in red and blue) are superimposed on the seismic image patch for which it was computed. Note that saliency maps correlate well with the position of the fault within the image patches indicating that pixels corresponding to the fault locations contribute most to the assessment of the focusing of the patch. [**CR**]

can be implemented by maximizing the following objective function:

$$J(\mathbf{I}) = y(\mathbf{I}) - \lambda ||\mathbf{RI}||_2^2, \tag{2.34}$$

where $\mathbf{I}$ is the image patch, y is the CNN output before the sigmoid activation function, $y = \theta_0 + \boldsymbol{\theta}^T \mathbf{x}_f(\mathbf{I})$, and $\mathbf{R}$ is a regularization operator. Minimizing Equation 2.34 using a gradient-based method requires computing the gradient of Equation 2.34 which can be expressed as follows:

$$\nabla J = \frac{\partial y}{\partial \mathbf{I}} - \lambda \mathbf{R}^T \mathbf{RI}. \tag{2.35}$$

Neglecting the regularization term in Equation 2.35 and examining the derivative of the CNN output with respect to the image, we can recognize that this term can be interpreted as a sensitivity of the output focusing score to changes in the input image. Therefore, visualizing this term for a particular image will indicate which pixels of the image contribute most to the output focusing score. Within the deep learning

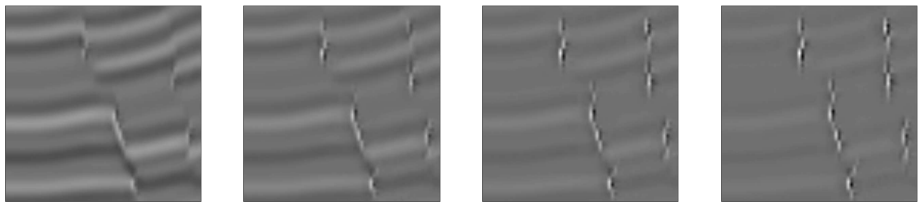
and computer vision community, this computed sensitivity attribute is referred to as an image class saliency map (Simonyan et al., 2014). Figure 2.24 shows the computed saliency maps for 10 input stacked seismic image patches. For each image, the saliency map is superimposed on the input seismic image patch. Note that for each image patch shown in Figure 2.24 the saliency map correlates well with the fault position indicating the pixels associated with the fault segment in the image provide the most influence to the output focusing score. Note that the CNN was not provided with any information regarding the position of the fault.

Using the gradient in Equation 2.35 and negating the objective function, I performed 20 iterations of gradient descent with a fixed learning rate of 0.1 in order to maximize Equation 2.34. For the initial image, I used the image shown in Figure 2.20 and I used the identity operator for the regularization operator and used $\lambda = 0.001$. Figure 2.25 shows the estimated image after 5, 10, 15 and 20 iterations of the steepest descent inversion. Examining the images estimated at the different iterations of the inversion, it is apparent that the optimization procedure is eliminating the horizons within the image and is promoting the discontinuities associated with the fault. This further confirms what was shown in both the visualization of the intermediate feature maps shown in Figures 2.21 - 2.23 as well as the saliency maps shown in Figure 2.24.

In addition to estimating images that maximize the output focusing score, Equation 2.34 can be modified in order to estimate images that maximize the intermediate activations within the fault-focusing CNN:

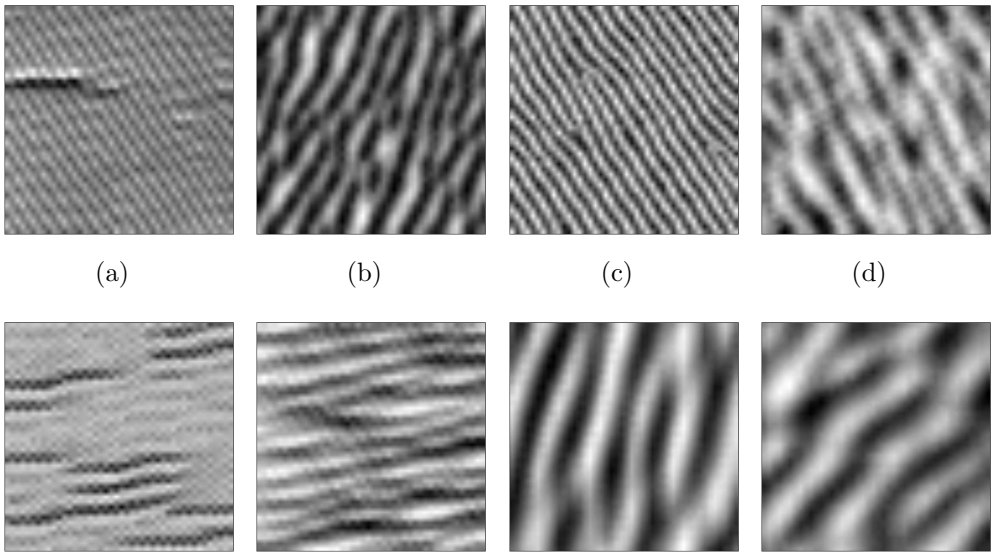
$$J(\mathbf{I}) = \mathbf{h}_j^{(k)}(\mathbf{I}) - \lambda||\mathbf{RI}||_2^2, \tag{2.36}$$

where $\mathbf{h}_j^{(k)}$ is the output of the convolution from the jth channel of the kth layer of the fault-focusing CNN. Maximizing Equation 2.36 to visualize features learned by the CNN is referred to as activation maximization (Erhan et al., 2009). To maximize this objective, I used 200 iterations with a gradient-descent algorithm with a learning rate of 0.1. I started from a random initialization and used a Laplacian operator as a regularization operator with $\lambda = 0.01$.



(a) (b) (c) (d)

Figure 2.25: Estimated image resulting from optimizing Equation 2.34 after (a) 5 iterations, (b) 10 iterations, (c) 15 iterations and (d) 20 iterations. It is apparent that the CNN is promoting discontinuities associated with the fault in the input image to maximize the output score. [**CR**]



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 2.26: The result of activation maximization for a subset of intermediate feature maps for $\mathbf{h}^{(2)}$. These patterns within these images indicate edges of different orientations that maximize the different activations within the fault-focusing CNN. [**CR**]

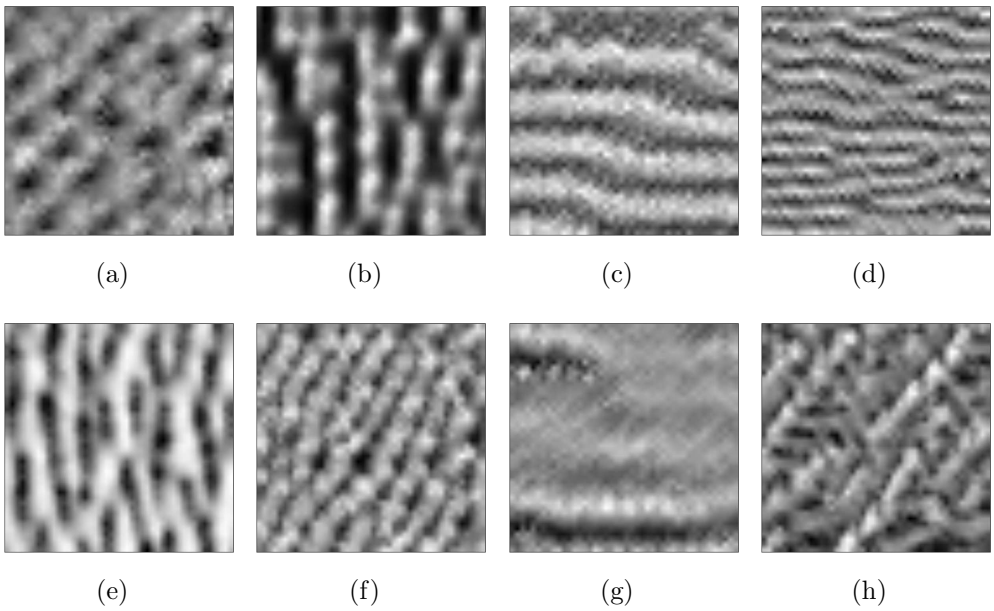


Figure 2.27: The result of activation maximization for a subset of intermediate feature maps for $\mathbf{h}^{(3)}$. When compared to the images in Figure 2.26, these estimated sinusoidal patterns appear to be of longer wavelength. [**CR**]

The results of performing activation maximization for a subset of channels for the feature maps computed at the second and third layers ($\mathbf{h}^{(2)}$ and $\mathbf{h}^{(3)}$) are shown in Figures 2.26 and 2.27. I selected these estimated images as they appeared to be the most interpretable and could be most easily understood (it is not uncommon for activation maximization to result in noisy uninterpretable images (Nguyen et al., 2019)). In general, the images appear to contain plane-wave like patterns of varying dips and frequencies. Within the computer vision community, these patterns have been interpreted as edges of different orientations that the CNN searches for within the image (Mahendran and Vedaldi, 2016). Note that the images estimated from the activations within the third convolutional layer (Figure 2.27), tend to appear slightly more complex and in general to contain longer wavelength patterns. This likely occurs due to the lower resolution features extracted at this stage of the network and that in general, CNNs tend to extract more complex features at the layers closer to the output.

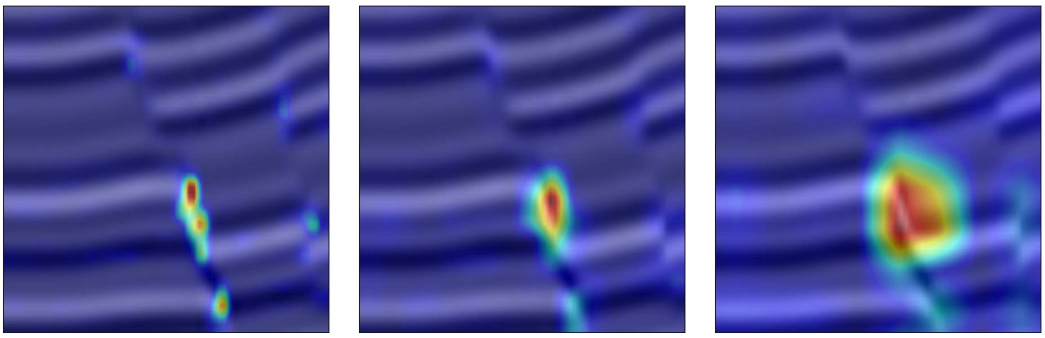
The final visualization approach I used for visualizing the features extracted from the CNN is known as class activation mapping (CAM) (Zhou et al., 2016). CAM approaches are useful for CNN feature visualization as similar to saliency, they indicate the regions of an input image that are used by the CNN for classifying the image. The first step in implementing CAM is to backproject the focusing score computed at the output layer for a particular image, onto the intermediate feature maps within the CNN. This can be done by computing the gradient of the output score, y, with respect to a feature map (Selvaraju et al., 2017). For a particular image, the computed gradients will be larger for certain feature maps and smaller for others. Therefore, the magnitudes of the gradients can be used as weights indicating the importance of the computed feature maps for a particular image. To compute this magnitude, a global average pooling computation is performed on the gradient which can be expressed as follows:

$$\eta_j^{(k)} = \frac{1}{N} \sum_z \sum_x \left(\frac{\partial y(\mathbf{I})}{\partial \mathbf{h}_j^{(k)}} \right)_{z\,x}, \tag{2.37}$$

where z and x are the sample indices in the z and x directions of the gradient and N is the total number of samples within the gradient. With these magnitudes computed for all feature maps within a layer, they can then be used to weight each of the feature maps computed from a particular image within a layer. This weighted sum can be expressed as follows:

$$\mathbf{c}^{(k)} = \max \left(0,\ \sum_j \eta_j^{(k)} \mathbf{h}_j^{(k)} \right), \tag{2.38}$$

where $\mathbf{c}^{(k)}$ is the class activation map for the kth layer. The ReLU function is applied to the output as only the positive elements of the activation functions contribute to the focusing score for a focused image. As $\eta_j^{(k)}$ will vary in magnitude depending on the image provided to the CNN, the computed $\mathbf{c}^{(k)}$ will show the most relevant information extracted at the kth layer in order to classify the input image. This is demonstrated in Figure 2.28 which shows the computed class activation maps for each layer of the fault-focusing CNN for the input image patch shown in Figure 2.20. Like the saliency maps shown in Figure 2.24, the class activation maps correlate well with



(a) (b) (c)

Figure 2.28: Class activation maps computed for (a) the first, (b) the second and (c) the third convolutional layers within the fault-focusing CNN for the image shown in Figure 2.20. Note that longer wavelength information is more relevant deeper in the network. [**CR**]

the position of the fault indicating that this portion of the image is used for classifying the input image as focused. Also note that the class activation maps become broader and smoother for the deeper layers within the CNN. This is due to the lower resolution feature maps extracted at the deeper stages of the CNN.

SUMMARY AND CONCLUSIONS

In this chapter I introduced the central workflow of this thesis for assessing velocity errors in seismic images using prestack Stolt residual depth migration and a fault-focusing CNN. I showed that by training a CNN on synthetic focused and unfocused seismic image patches containing geologic faults, the output of the CNN can be used as a focusing measure for seismic images. For a synthetic seismic image containing several unfocused faults, I demonstrated that the CNN is able to accurately assess the focusing errors along the faults. I also provided an additional analysis that helps to better understand what features the CNN extracts from the images in order to provide a focusing measure. By directly visualizing the extracted features from a seismic image, computing saliency and class activation maps, I showed that without having any prior knowledge of the fault location, the CNN localizes the fault and uses

the strength of the fault discontinuity within the image in order to assess the focusing of the image patch.

In the subsequent chapters, I will demonstrate the effectiveness of the workflow presented in this chapter on a subsalt synthetic and two field data examples. For each example, I will provide a comparison of the estimated $\rho(z, \mathbf{x})$ between the CNN-based focusing analysis and the traditional semblance-based focusing analysis and will show that the CNN-based focusing analysis is more robust to a lack of illumination that limits the signal to noise ratio within the angle gathers. Additionally, I will introduce two methods that I use to QC the estimated $\rho(z, \mathbf{x})$: image refocusing and fault segmentation. Using both of these techniques, I further show that the CNN-based approach leads to better focusing of the faults and consequently, improved automatic fault interpretation.

Chapter 3

2D synthetic subsalt and field data example

In Chapter 2, I demonstrated how a CNN can be constructed and trained to measure the focusing of a seismic image and I demonstrated its ability to assess the focusing of seismic images on a simple 2D seismic image. In this chapter, I will apply the CNN-based image focusing measure to a 2D subsalt synthetic dataset as well as a 2D real data example. The subsalt example demonstrates how the CNN-based focusing measure performs in areas where angle-range is highly limited due to a lack of illumination, a common occurrence in geologic regions such as the Gulf of Mexico that contain numerous salt bodies. The field data example is from the Gulf of Mexico and when migrated, the image contains several normal listric faults that exhibit fault plane reflections. The field data example demonstrates how the CNN-based focusing measure generalizes to field data examples, a challenging task for many supervised learning-based processing techniques. For both examples, I show that the estimated $\rho(z, x)$ from the CNN-based approach results in improved fault and image focusing over a traditional semblance-based approach.

IMAGE REFOCUSING AS A QC FOR FOCUSING ANALYSIS

Before I describe the two examples on which I tested the CNN-based focusing measure, I will introduce the QC measure that I use to assess the quality of the estimated $\rho(z, \mathbf{x})$ for the examples shown in this thesis. This QC measure is especially useful as the values of the estimated $\rho(z, \mathbf{x})$ can be assessed directly only for the case of constant velocity errors.

While the immediate goal of a focusing analysis is to estimate a focusing parameter (in the case of this thesis, the residual migration parameter ρ), the final goals are to improve the focusing of the seismic image and to position reflectors at their correct depths. To achieve both of these goals, the depth velocity model must be updated and the image re-migrated with the updated depth model. The most common approach to updating the depth velocity model is via tomography. Audebert et al. (1997) showed that after picking the focusing parameter from a CRP scan, they could compute a dip-independent RMS velocity along the normal ray which could then be used as input to a zero-offset 3D tomographic inversion (Lanfranchi et al., 1996). Additionally, Sava and Biondi (2004a) showed that after estimating $\rho(z, \mathbf{x})$ from a residual focusing analysis, they could first estimate an image perturbation via the application of a linearized residual migration operator and a scaling of the image with $\Delta\rho = 1 - \rho(z, \mathbf{x})$ and then use this image perturbation as the input to a linearized wave-equation migration velocity analysis in order to estimate a slowness perturbation in depth and update the migration velocity model.

As the purposes of focusing analysis in this thesis are directed towards improving the interpretation of geologic faults, I instead perform an interpretation-driven QC to assess the quality of the estimated $\rho(z, \mathbf{x})$. This QC consists of performing a "refocusing" (i.e., correction) to the unfocused image by directly using the estimated $\rho(z, \mathbf{x})$ (MacKay and Abma, 1989). Mathematically, this refocusing operation can be expressed as:

$$I_{\text{ref}}(z, \mathbf{x}) = I(z, \mathbf{x}, \rho(z, \mathbf{x})), \tag{3.1}$$

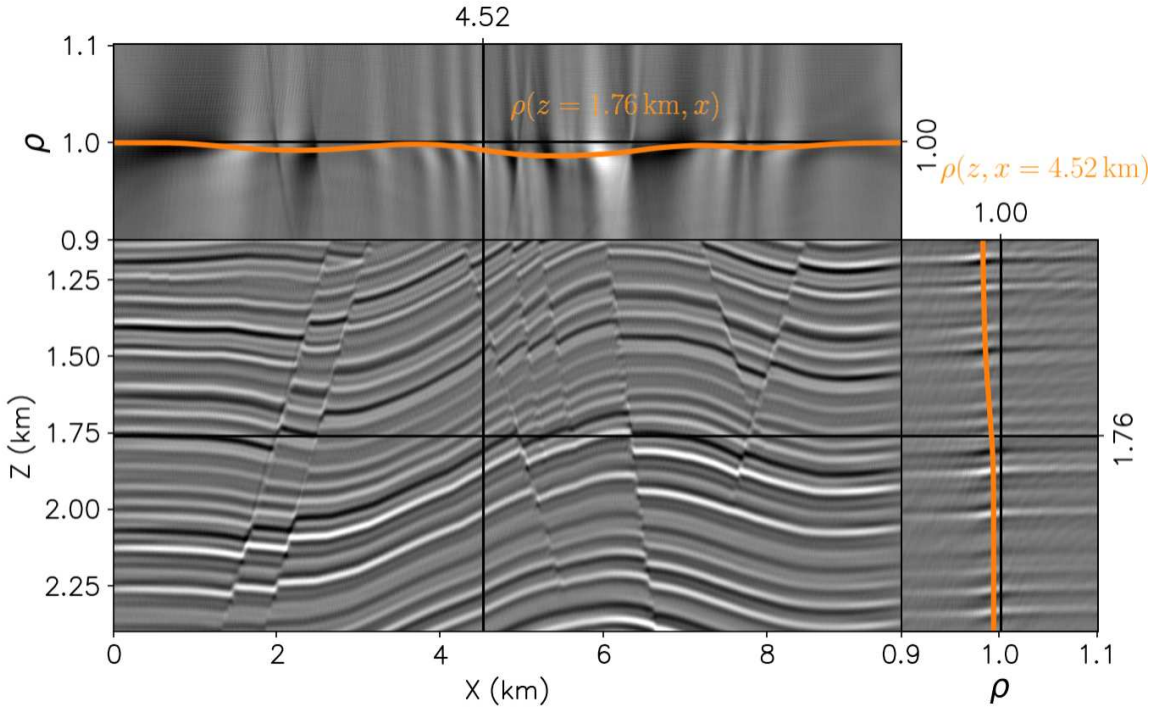


Figure 3.1: Slices of the estimated $\rho(z, x)$ (orange curves) from the example in Chapter 2 plotted on the stacked residual migration images $I(z, x, \rho)$. The refocused image is computed by extracting the samples from $I(z, x, \rho)$ along the surface given by $\rho(z, x)$. [**CR**]

where I_{ref} is the refocused stacked image, $I(z, \mathbf{x}, \rho)$ is a collection of stacked residually migrated images and $\rho(z, \mathbf{x})$ is the estimated spatially-varying residual migration parameter (i.e., focusing map). Typically, Equation 3.1 is numerically implemented via a high-order spline interpolation. Looking at Equation 3.1, we can understand that this step essentially amounts to selecting the optimally focused regions of the image as determined by residual focusing analysis. If the estimated $\rho(z, \mathbf{x})$ is correct, then after this step is performed, reflections should appear more coherent, and the focusing of faults (e.g., fault plane reflections and diffractions) should be improved. Figure 3.1 visually demonstrates the mechanics of Equation 3.1 by presenting the collection of stacked residually migrated images from the example shown in Chapter 2. The orange lines plotted on the $\rho - x$ and $\rho - z$ planes show slices of the $\rho(z, x)$

surface used to select the best focused regions of the residually migrated images. The refocused image, I_{ref}, obtained after numerically performing the interpolation described in Equation 3.1 using the $\rho(z, x)$ and $I(z, \mathbf{x}, \rho)$ from Chapter 2, is shown in Figure 3.2(b). In order to compare the improvement in the refocused image, Figure 3.2 also shows the unfocused image from Chapter 2 in panel (a) and the ground truth focused image (migrated with the correct velocity) in panel (c). When comparing each of the three images, it is apparent that the diffractions have been focused and the focusing of the faults has been improved in the refocused image making it comparable to the ground-truth focused image.

2D SYNTHETIC SUBSALT EXAMPLE

The first example of this chapter to which I apply the CNN-based image-focusing analysis is a 2D synthetic subsalt example. Due to the high velocity contrast of salt compared with the surrounding sediments as well as its complex shapes, salt bodies are often the cause of many problems when imaging seismic data. One of these problems is that salt dramatically limits the illumination of imaging targets that reside beneath it. This then often results in little to no signal along the aperture-angle axis making traditional semblance-based focusing analysis difficult. Figure 3.3(a) shows an example of a stacked seismic image that contains a large salt overhang under which are positioned several faults. To create the image, I first performed linearized modeling and then migration using wave-equation depth migration. The acquisition geometry followed closely of that used on the Sigsbee data examples, in which 500 streamer shots were modeled with a maximum offset of approximately 8 km (Paffenholz et al., 2002). The maximum frequency used for modeling and imaging was 50 Hz. A negative velocity anomaly extending from 5 - 17 km in the lateral direction and from 0.5 - 1.5 km in depth was introduced during imaging in order to create an unfocused image. Comparing the faults not under the salt, with those under the salt, we observe significantly more diffracted energy on the faults that are not positioned under the salt. Nevertheless, all of the faults exhibit signs of defocusing, making them crucial in this scenario for accurate and robust focusing analysis. Figure

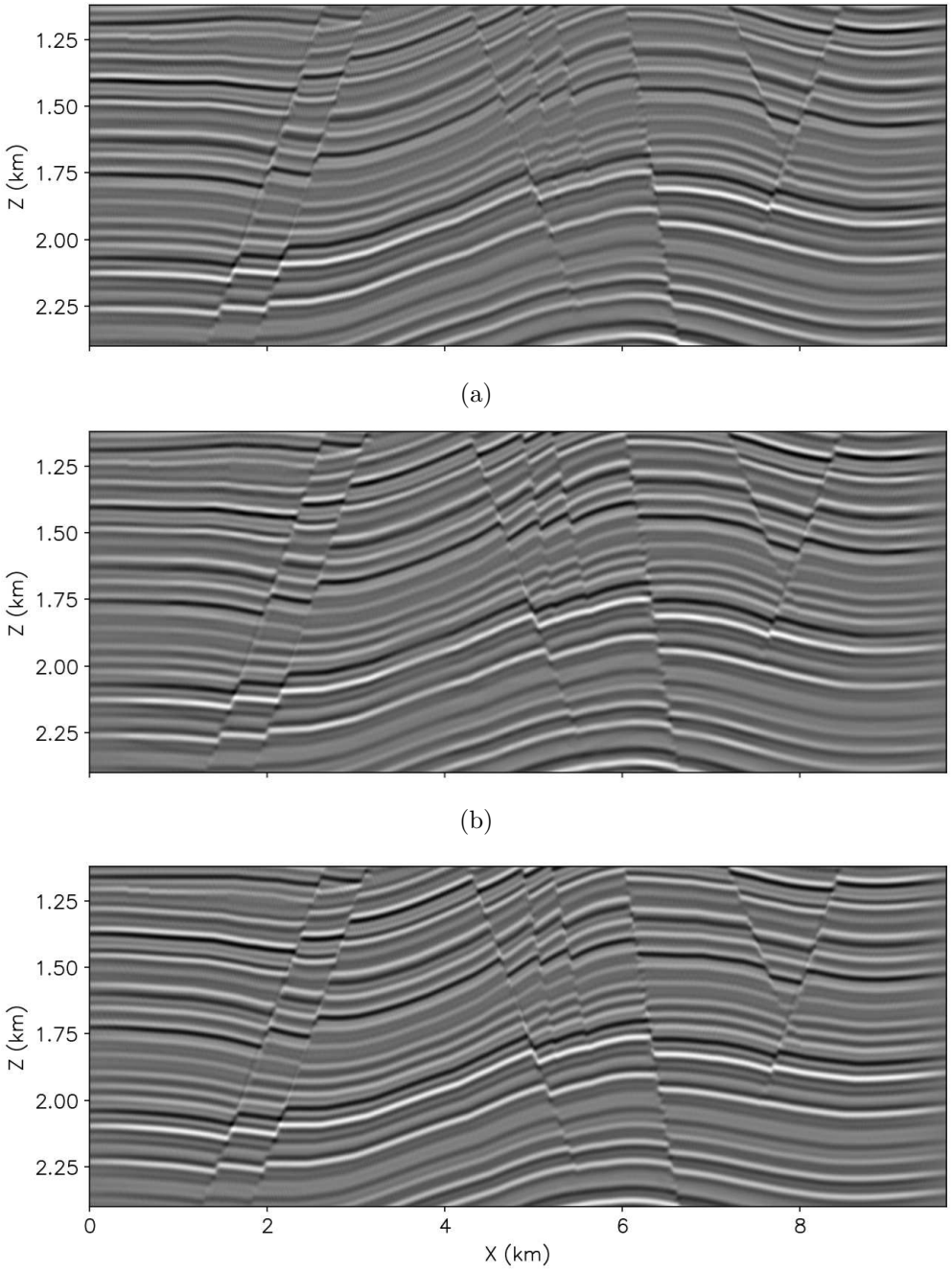
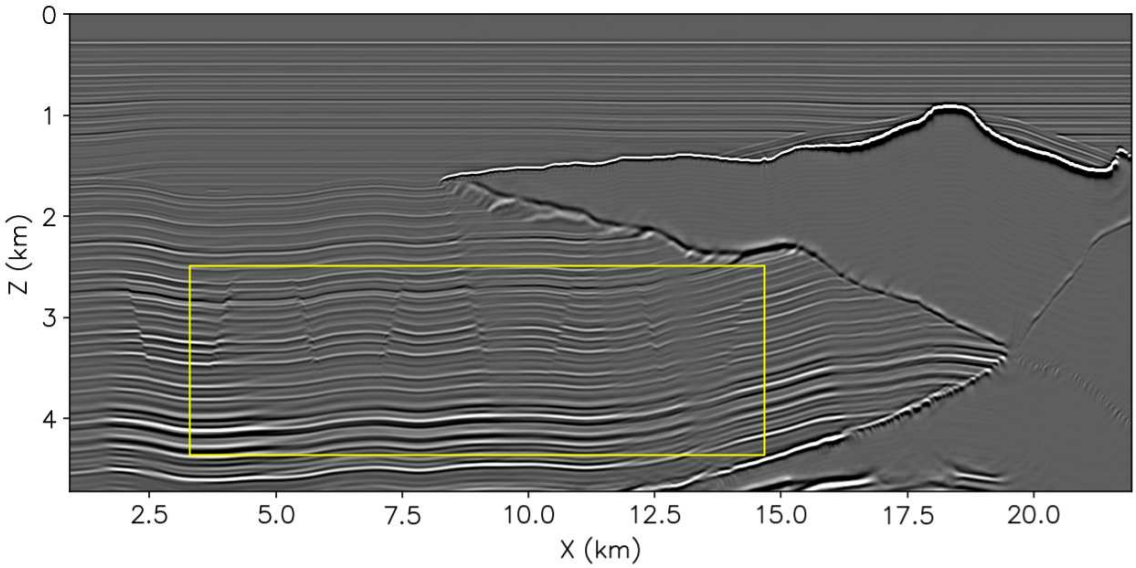


Figure 3.2: Comparison of (a) the unfocused image from Chapter 2, with (b) the image refocused with the estimated $\rho(z, x)$ and (c) the image migrated with the correct velocity (ground truth focused image). Comparing the faults within the different images, it is apparent that faults within the refocused image have minimal diffracted energy and are qualitatively similar in appearance to those within the ground truth image. [**CR**]

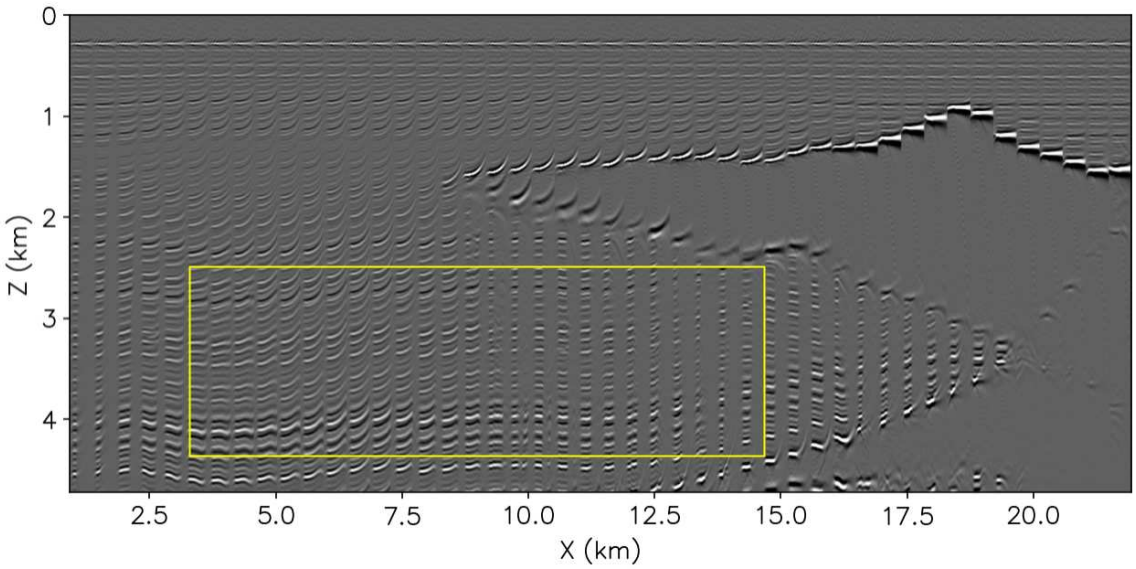
3.3(b) shows the angle gathers plotted at approximately 0.5 km intervals. We can very clearly observe a significant reduction in energy along the aperture angles for the image points beneath the salt. The region of interest for this test is contained within the highlighted box shown in Figure 3.3(a) and is plotted in Figure 3.4. The angle gather shown next to the stack is extracted at the position of the black line (approximately 9.7 km). The goal of this example is to first investigate the limitations of semblance for reflections that originate from beneath the salt that contain little aperture-angle information. To test this, I first performed prestack Stolt residual depth migration for a range of ρ values to create many residually focused images. I then estimated ρ using only the aperture-angles by means of semblance. Then, I applied my CNN-based focusing analysis from the previous chapter with the goal to robustly extract accurate focusing information using only the diffractions from the faults as well as the coherence from reflections available within stacked images. To compare and QC the results I used the refocusing technique and as described in the first section of this chapter.

2D Prestack Stolt residual depth migration

In order to capture the velocity errors present within the image, I applied prestack residual depth migration for ρ values ranging from 0.9 to 1.1 with an interval of 0.00125. This resulted in 161 total prestack residually migrated images, from which I could perform image-focusing analysis. Figure 3.5 shows five of these images ranging from $\rho = 0.94$ to $\rho = 1.06$. Comparing the images shown in these figures, we can first observe that visually, it is difficult to see moveout differences along the aperture-angle axis. As I will show in the following section, this will greatly limit the resolution of semblance. However, we can qualitatively observe that the power of the stack as well as the diffractions are best focused for the image corresponding to $\rho = 0.97$.



(a)



(b)

Figure 3.3: 2D synthetic image where faults are positioned beneath the salt overhang. Panel (a) shows the stacked unfocused image and panel (b) shows the angle gathers plotted at approximately 0.5 km intervals. Note the dramatic reduction in aperture-angle information of the image points beneath the salt overhang. The highlighted box indicates the region of interest for this test. [**CR**]

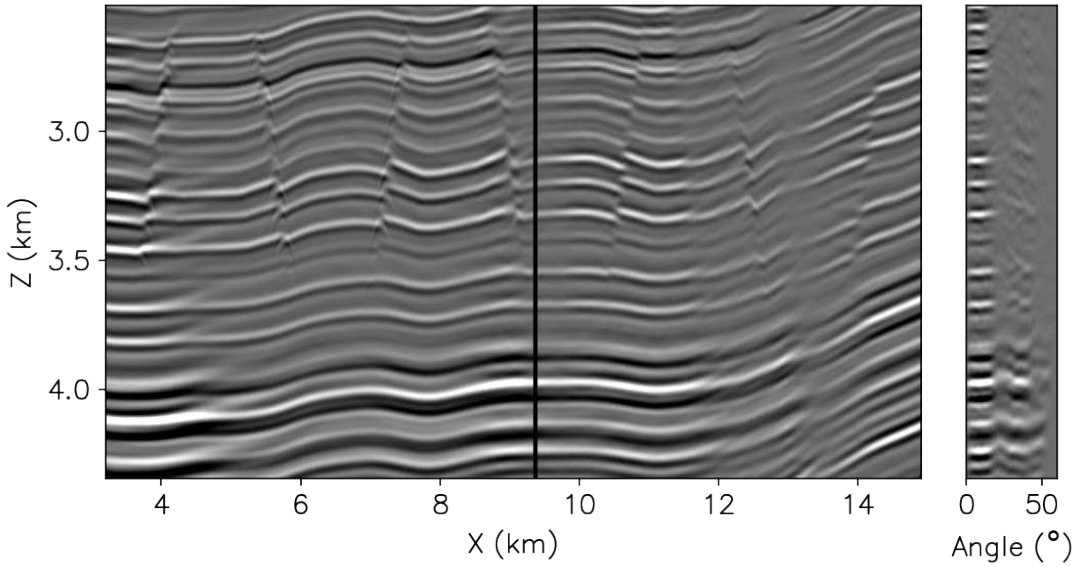


Figure 3.4: The stacked image windowed within the region of interest shown in Figure 3.3. The black line plotted at $x \approx 9.7$ km shows the lateral position from which the angle gather was extracted. Note that due to the presence of the salt, the angle range as been limited to a maximum of angle of approximately 20°. [**CR**]

Semblance-based focusing analysis

I then performed semblance-based focusing analysis for each image point. For regions unaffected by the salt (e.g., $x = 5$ km), semblance-based focusing analysis performed well and resulted in a high-resolution semblance map which was simple to pick. Figure 3.6(a) shows an example of a high-resolution semblance map extracted at x = 5 km. In contrast, semblance-based focusing analysis performs rather poorly for image locations beneath the salt. This is clearly shown in Figure 3.6(b) where little to no coherent energy is present above 3.5 km. Without a clear trend available in the semblance map, picking the correct $\rho(z)$ for this image point is a very difficult task and typically will result in erroneous picks.

Repeating this process for all image points and then smoothing the picks with a triangular smoother of length 0.3 km in depth and a 0.46 km length smoother in the lateral direction resulted in an estimate for $\rho(z, x)$. Figure 3.7 shows the

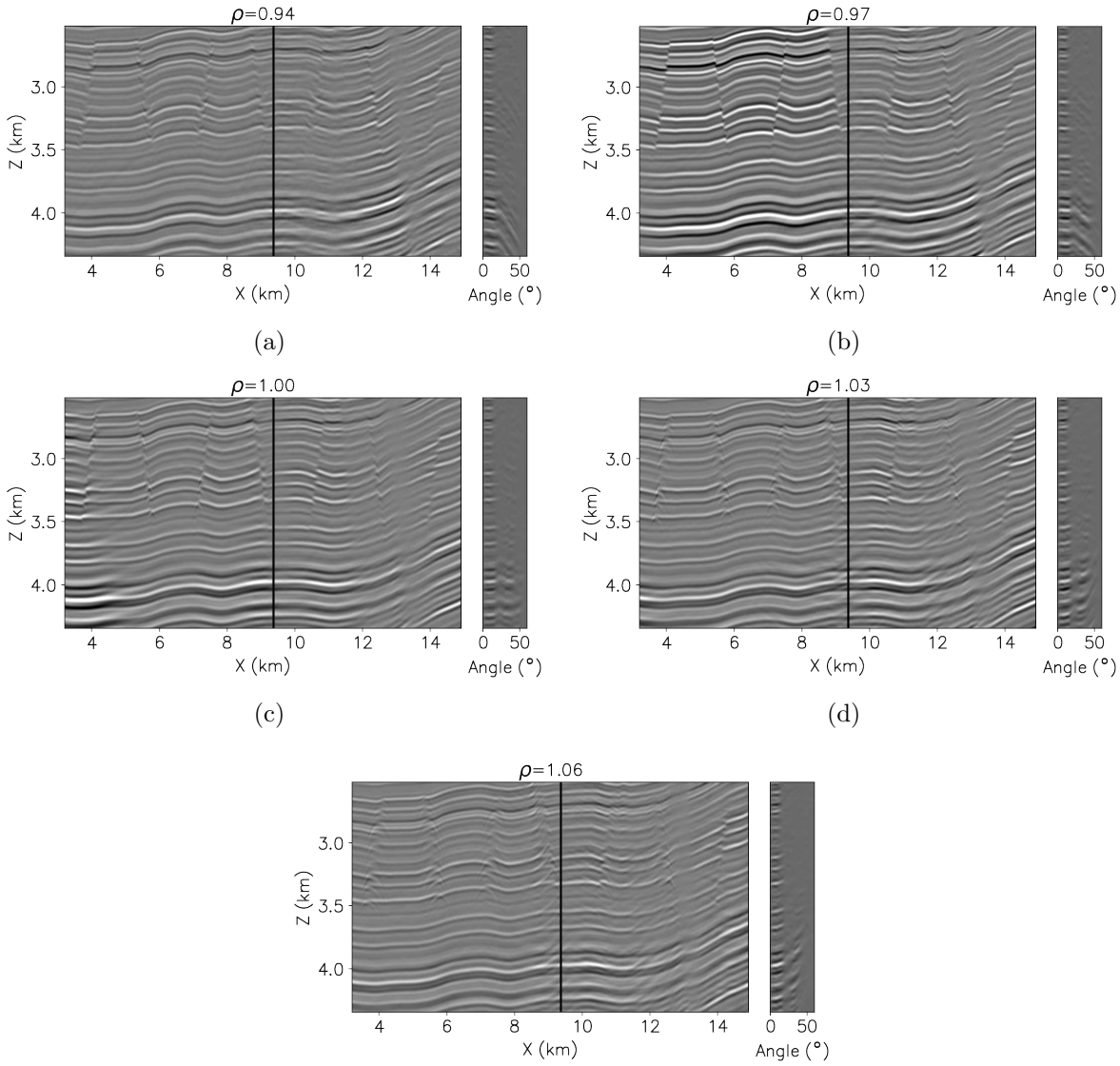
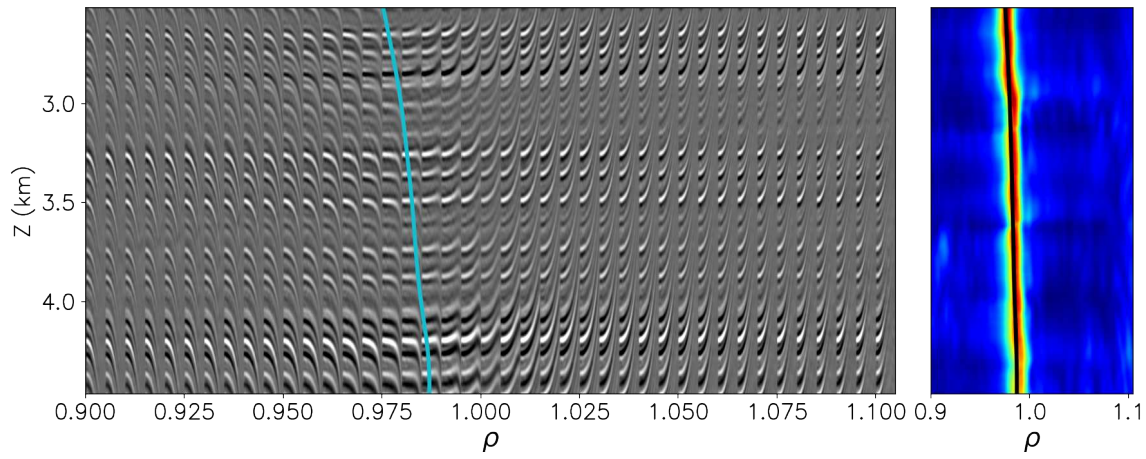
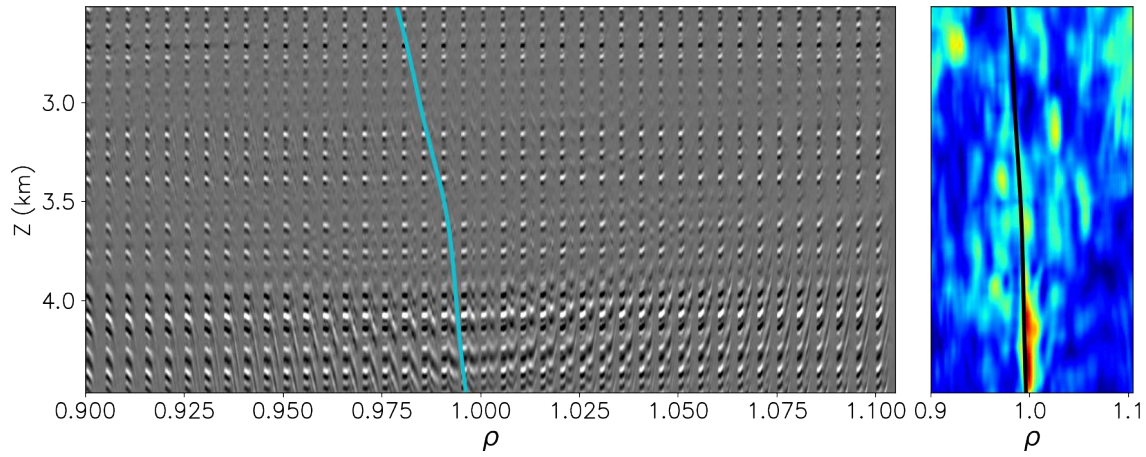


Figure 3.5: A selection of residual migration images taken from the application of prestack Stolt residual depth migration to the unfocused image. While it is clear that a single ρ value will not correct for all of the velocity error, we observe that the best focusing of the faults and largest stack power for $\rho = 0.97$ (panel (b)). [**CR**]



(a)



(b)

Figure 3.6: Semblance-based image-focusing analysis for a (a) well-illuminated image point (extracted at x = 5km and (b) a poorly-illuminated image point (extracted at x=9.5km). [**CR**]

estimated $\rho(z, x)$ superimposed on the unfocused image. Examining the values of $\rho(z, x)$, we first can observe that semblance does well in assessing the focusing of the image within the well-illuminated portion of the region of interest. For image points between 4 and 8 km, the estimated $\rho(z, x)$ is generally consistent with the focusing of the faults observed in the residual migration panels shown in Figure 3.5. However, we can observe several areas within the region of interest in which semblance performed suboptimally. The first of these is in the lower leftmost portion of the image (beneath 3.5 km and extending to 4 km laterally). We observe that semblance incorrectly has assessed that no focusing correction is needed in this region. Observing the shape of the angle gathers in this region as shown in Figure 3.3(b), it is clear that the gathers appear to slightly undulate at the large angles and have deviated from the hyperbolic curvature that can be corrected by prestack Stolt residual migration. It is likely then that prestack residual migration has not fully flattened the gathers and therefore picking the semblance maxima has led to an incorrect estimate of ρ. While the purpose of this example is to consider the effects of illumination and not complex velocity errors, as discussed in Chapter 2, this is another limitation of semblance-based focusing analysis and residual prestack migration that must be considered.

The second area in which semblance-based focusing analysis performed suboptimally is within the subsalt region extending from 12.5 - 14.1 km. This region is located well under the salt overhang and therefore greatly suffers from a lack of illumination and is the most challenging within this region of interest. At approximately 4 km depth and 14 km in the lateral direction, we can observe a sharp change from $\rho \approx 0.96$ to $\rho \approx 1.03$ which is most likely due to an illumination shadow zone that occurs within that region of the image. As I will later show when comparing the refocused images from the semblance-based focusing analysis and the CNN-based focusing analysis, this rapid change in $\rho(z, x)$ will create significant artifacts in the refocused image.

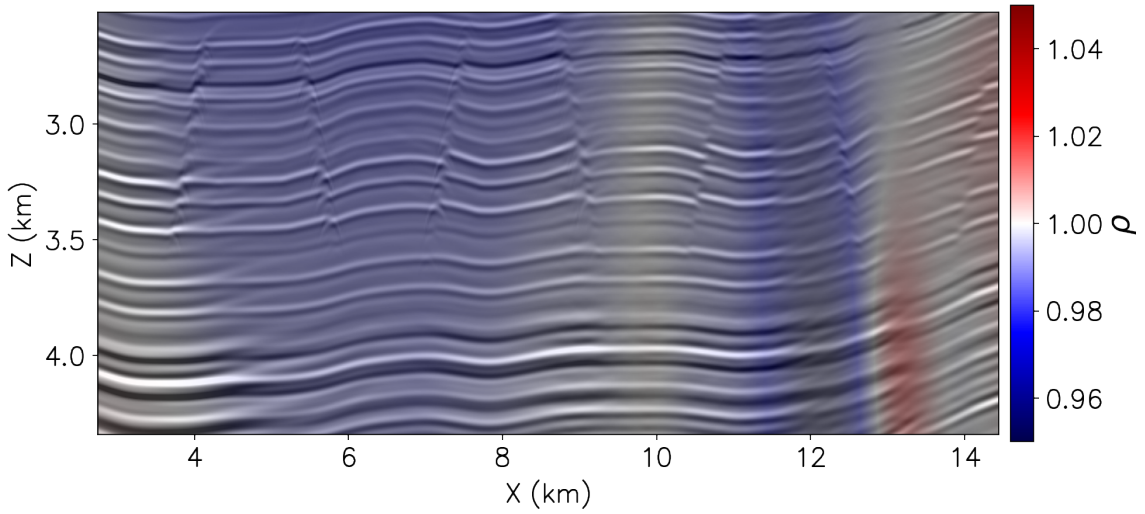


Figure 3.7: The estimated $\rho(z, x)$ from semblance-based focusing analysis superimposed on the unfocused image. [**CR**]

CNN-based focusing analysis

The application of semblance-based focusing analysis to the subsalt image showed that in the poorly illuminated zone of the region of interest, the use of semblance led to a rapidly varying $\rho(z, x)$ beneath the salt that was inconsistent with the focusing of the residual migration images shown in Figure 3.5. In the following section I will demonstrate that my CNN-based approach provides an automatic way of robustly assessing image focusing of images subsalt by using the diffractions as well as reflector coherence from only the stacked image. As I did in Chapter 2, I first create training data for the CNN using prestack residual migration on synthetic images obtained with the convolution of a wavelet with a synthetic reflectivity model. I then trained the CNN and estimated $\rho(z, x)$ using CNN-based focusing analysis. I will show that the estimated $\rho(z, x)$ agrees well with the focusing of the diffractions as shown in Figure 3.5 and results in a refocused image with fewer artifacts than the image refocused from the semblance-based approach.

Training data creation

As I have already described the design of the CNN in Chapter 2, the first step in performing the CNN-based focusing analysis for this subsalt example was to create training data to train the CNN. Similar to the example shown in Chapter 2, I created training images by first creating a synthetic velocity model, after which I computed the reflectivity via a derivative along the depth axis. The synthetic models I created had a geologic structure similar to that of the synthetic image shown in Figure 3.3(a), where I varied the undulation of the sediments as well as the depth and dip of the faults. I then convolved a 30 Hz Ricker wavelet with the reflectivity model to create a focused synthetic zero-offset seismic image. I created prestack seismic images via padding as I did in Chapter 2 and performed prestack Stolt residual migration for ρ values between 0.96875 - 1.03135 (apart from $\rho = 1$). To create stacked seismic images, I then converted both the focused and unfocused synthetic images from subsurface offset to angle and stacked over angle.

While this approach to creating focused and unfocused training image patches was computationally efficient and quickly resulted in many training samples, it could incorporate the effects of wave propagation which were non-negligible for this subsalt example. The largest of these effects is the limited illumination that, as I showed in Figure 3.3(b), severely limits the range of angles on the aperture-angle axis. To overcome this limitation, I first constructed an aperture-angle mute to simulate the effects of the limited aperture within the images created via convolution and residual migration. I computed this mute by smoothing the absolute value of the amplitudes of the seismic image shown in Figure 3.3. This essentially provided an envelope of the prestack image and an estimate of the energy within the image that I could then use as a mute for the synthetic images. The application of this mute to the prestack synthetic seismic images limited the angle range to what was observed in the migrated images and created a lateral variation in the amplitudes of the stacked images beneath the salt.

While the application of this mask helped to create more realistic focused and

unfocused images created from convolution and Stolt residual migration, they could not fully replace the need for incorporating focused and unfocused image patches created from wave-equation migration to help the CNN to generalize to focused and unfocused migrated images. Therefore, as I did in Chapter 2, I created an additional 100 synthetic images from linearized modeling and migration. To create the synthetic velocity models I used the same procedure as I used for the synthetic convolution images. Using the same patch-forming procedure from Chapter 2, I created a total of 5,120 focused patches and 5,120 unfocused patches from the convolution-based images and 500 focused and 500 unfocused patches from the wave-equation migration-based images. Figure 3.8 shows four examples of unfocused patches and four examples of focused patches created using this approach. The leftmost patches were created from the convolution-based approach and the rightmost patches from the wave-equation migration-based approach.

CNN training

With a prepared training set, I then trained the CNN using the unfocused and focused stacked image patches. I followed the same procedure for training as was described with the simple example in Chapter 2: I use the binary cross entropy loss function for the loss function and the ADAM optimization algorithm with a fixed learning rate of 1×10^{-4}. Of the 10,240 convolution-based training samples, I used 8,192 for training and 2,048 for validation during training. After training for 20 epochs, the CNN was able to achieve 99% accuracy on both training and validation convolution-based datasets. Following the training on the convolution-based images, I performed a secondary stage of training on the wave-equation migration-based patches. For this secondary training stage, I used the weights estimated from the first training stage as the starting point of the CNN weights and trained for a total of 10 epochs. After the 10 epochs of training, the CNN achieved 99% accuracy on the training and validation data.

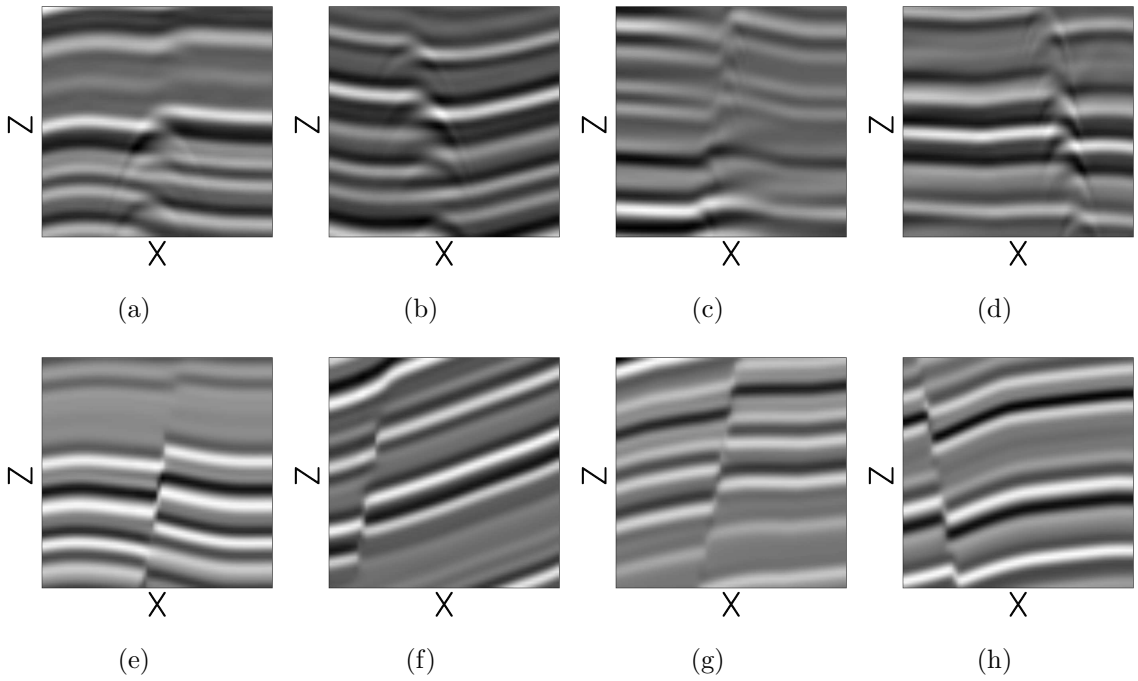


Figure 3.8: Synthetic training patches used for training the fault-focusing CNN. Patches in panels (a) - (d) show unfocused patches and patches (e) - (h) show focused patches. Note also that the leftmost patches of each row (panels (a)-(b) and (e)-(f)) show patches from the convolution-based training images while the rightmost patches (panels (c)-(d) and (g)-(h) show patches extracted from the wave-equation migration-based training images. [**CR**]

Focusing analysis with the trained CNN

With the CNN trained on the synthetic training images, it could then be used for performing image-focusing analysis on the residually migrated images generated from the prestack Stolt residual depth migration. Following the workflow outlined in Chapter 2, I first formed patches from all residually migrated stacked images within the region of interest. This resulted in a total of 16,905 patches. Note that unlike the example shown in Chapter 2 which used only patches that contained faults, for this example I predicted on all patches within the region of interest. I did this primarily to investigate the CNNs ability to use reflector coherence for assessing the focusing of an image patch. While the CNN was not explicitly trained to use reflector coherence,

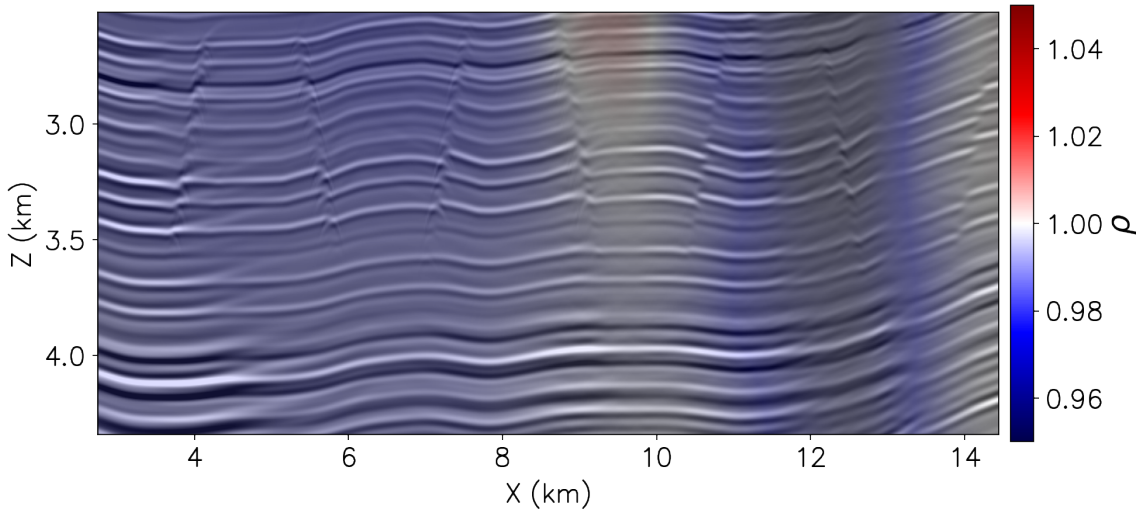


Figure 3.9: The estimated $\rho(z, x)$ from CNN-based focusing analysis superimposed on the unfocused image. [**CR**]

implicitly it uses the coherence of the imaged horizons in order to extract the discontinuities associated with the faults as was demonstrated in the CNN interpretation section in Chapter 2.

After predicting focusing scores for all patches and forming the scores into spatial groups, I used Equation 2.29 to estimate ρ for each spatial patch. After smoothing all estimated ρ patches with a triangular smoother of length 0.23 km in depth and 0.7 km in the lateral direction, I obtained the estimated $\rho(z, x)$ shown in Figure 3.9 superimposed on the unfocused image. We can first observe that in general, the estimated $\rho(z, x)$ is consistent with the focusing of the diffractions as observed in the $\rho = 0.97$ image shown in Figure 3.16(b). We can also observe that overall, the estimated $\rho(z, x)$ is more uniform than when compared to the $\rho(z, x)$ estimated from semblance. However, within the region above 3 km and centered at approximately 10 km laterally, we can observe an incorrect assessment of the image focusing where the CNN has overestimated the focusing parameter with a value of $\rho \approx 1.02$.

QC and comparison of results

With the estimated $\rho(z, x)$ from both the semblance-based focusing analysis ($\rho_{\text{SMB}}(z, x)$) and the CNN-based focusing analysis ($\rho_{\text{CNN}}(z, x)$), refocused images could be obtained with the application of Equation 3.1 to the cube of residually migrated stacked images. Figure 3.10 shows refocused images from using $\rho_{\text{CNN}}(z, x)$ and $\rho_{\text{SMB}}(z, x)$ in Equation 3.1. For comparison purposes I have also included the unfocused image and the image migrated with the correct velocity (the ground-truth focused image). Comparing the two refocused images in Figures 3.10(b) and 3.10(c), we can observe that the refocusing from both semblance and the CNN improved the focusing of the faults as well as some of the horizons within the image. This is especially the case for the heavily diffracted faults located in the well-illuminated zone (within 3 - 8 km laterally). However, we do observe some key differences as are highlighted within the yellow boxes that have been placed on the images.

Comparing the focusing and sharpness of the faults, the largest difference between the two images can be observed in box A where in the image refocused using ρ_{SMB}, we can still observe diffracted energy from the faults. Additionally, we observe that the fault discontinuity on the bottommost reflector is almost non-existent. In contrast, in the image refocused with ρ_{CNN} we can observe a considerably sharper fault with minimal diffracted energy. While this region is largely unaffected due to the lack of illumination introduced from the presence of the salt, we nevertheless observe here that due to the limitations of prestack Stolt residual depth migration, maximizing the power of the stack will not always result in the best focusing within the physical axes of the image. Another location within the region of interest where a difference in fault focusing can be observed is within box B. Assessing the focusing of the fault here is especially challenging given that the location of this fault coincides directly with the location of an illumination shadow zone. It is due to this reason that in fact, we observe the refocused fault from the CNN-based focusing analysis is slightly undermigrated due to the overstimation of ρ within this region. Comparing the two images within this box, we can observe that in this case, the image refocused with ρ_{SMB} provides more coherent reflectors and fewer artifacts. This overestimation of ρ from

the CNN-based analysis could likely be improved by training it on a larger amount of focused and unfocused image patches created from wave-equation migration.

Comparing the images within the remaining boxes (C - E), we observe a similar focusing of the faults in both refocused images. This is in large part due to the fact that these faults are poorly illuminated as they are located beneath the salt overhang. Nevertheless, we can observe improved reflector coherence within the image refocused from ρ_{CNN}. This is true in box C where we can observe a more consistent amplitude across the reflectors that cross the shadow zone towards the top of the box. This is also true for both boxes D and E where we observe significantly fewer artifacts present in the image refocused from ρ_{CNN}. These improvements can be most likely attributed to the fact that the CNN-based approach not only will provide a large focusing score for images containing better focused faults, but also for more coherent reflectors. Additionally, recall that the semblance-based approach is performed separately for each image point and then smoothed across all image points. While the resulting ρ will be smooth, because of the lack of structural information, the resulting refocused image can contain spatially incoherent artifacts as are apparent within box E of Figure 3.10(b). The CNN-based approach is performed on a patch-wise basis and therefore a structurally consistent ρ is enforced across the patch.

As a final point, it is important to emphasize that the CNN-based analysis was performed entirely on stacked images. Not only is this more and aligned with what is done for real subsalt image focusing analyses performed in industry (Wang et al., 2006, 2009), using stacked images also provides several orders of magnitude of savings in terms of memory and data storage. This is especially the case for 3D images where a lack of sufficient memory and large amounts of data I/O become the computational bottleneck for performing focusing analysis using 3D prestack residual Stolt migration. Therefore, using the CNN-based approach on only stacked images can serve as an alternative to a semblance-based approach when data storage and large memory resources are limited.

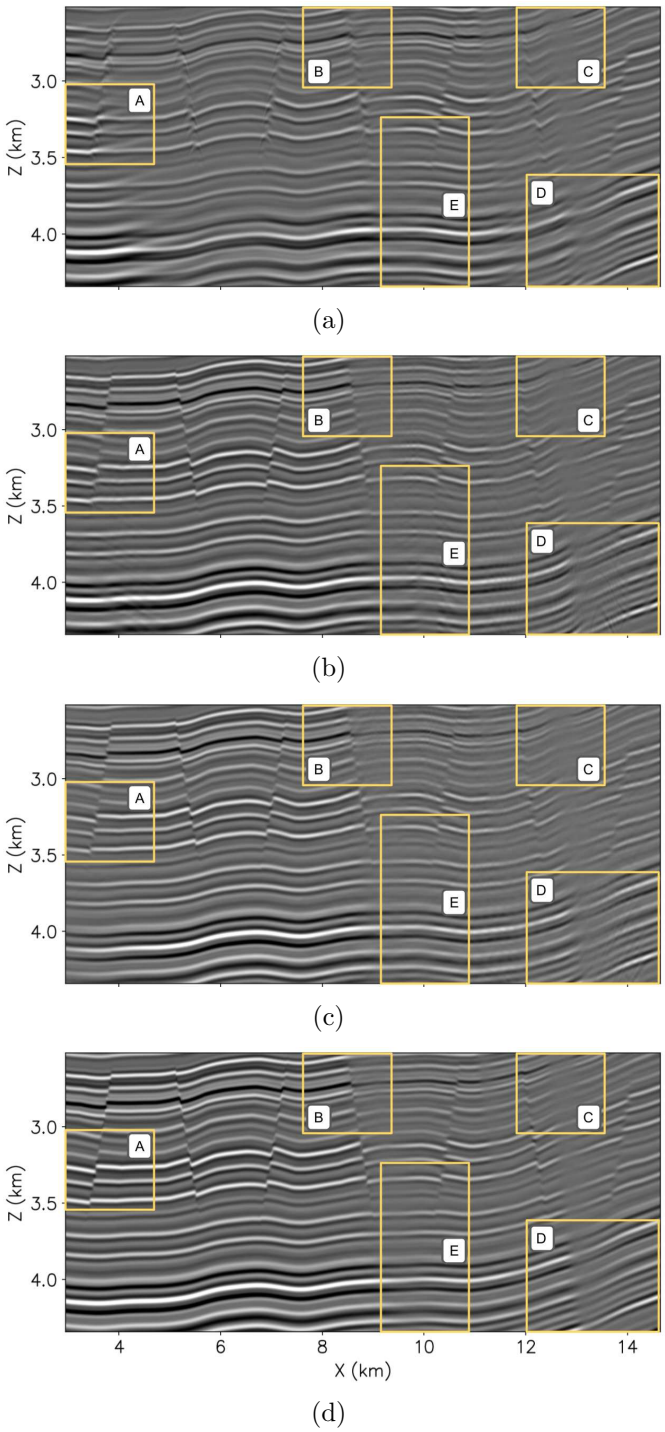


Figure 3.10: Comparison of the refocused images obtained from (b) ρ_{SMB} and (c) ρ_{CNN}. Panels (a) and (d) contain the unfocused and ground-truth focused images respectively and are included for reference. The yellow annotated boxes highlight regions where the two refocused images differ in fault focusing or reflector coherence. [**CR**]

FAULT SEGMENTATION AS A QC FOR FAULT FOCUSING

In this section I introduce automatic fault segmentation as a QC to be used in addition to the refocusing QC I introduced at the beginning of the chapter. As I will show for both the 2D and 3D field data examples in this thesis, segmenting the faults on focused/unfocused images helps to provide a more precise qualitative analysis on the focusing of the faults. Additionally, as I use a fault confidence attribute to segment the faults, this attribute can be used as a quantitative measure for assessing the focusing and interpretation of faults.

While there are many algorithms for fault segmentation, CNNs have recently shown remarkable success in segmenting faults in a wide variety of geologic scenarios (Wu et al., 2019; Di et al., 2019). I also adopt a CNN-based approach for fault segmentation. My approach resembles closely to that of the approach described by Wu et al. (2019). For the network, I use a 2D U-net CNN architecture for 2D images and a 3D U-net CNN architecture for 3D images. Using the same model-building approach described in the previous chapter, I build 2D and 3D synthetic images with faults. For the label, I extract the computed trajectory of the fault: the pixels on the fault trajectory receive a label of one, and the pixels away from the fault trajectory receive a label of zero. Figure 3.11(a) shows several examples of fault labels plotted on faults within a synthetic seismic image. The red pixels indicate the pixels within the seismic image that pertain to the faults. The training procedure for training a CNN to perform fault segmentation is similar to the training procedure for the fault-focusing CNN. First, overlapping patches are formed for each image as well as for the labels. Following the approach of Wu et al. (2019), I also chose a patch size of 128 $\times$ 128 pixels, and as I did for the fault-focusing CNN, I create overlapping patches from four different patch grids. This created a total of 4,000 patches for training. For the CNN training, I used a weighted binary-cross entropy loss function as opposed to a standard binary-cross entropy loss function as there exists a large class imbalance in the labels (i.e., the ratio of non-fault pixels to fault pixels is large) and I used the ADAM optimization algorithm with a learning rate of 1×10^{-4} to minimize the

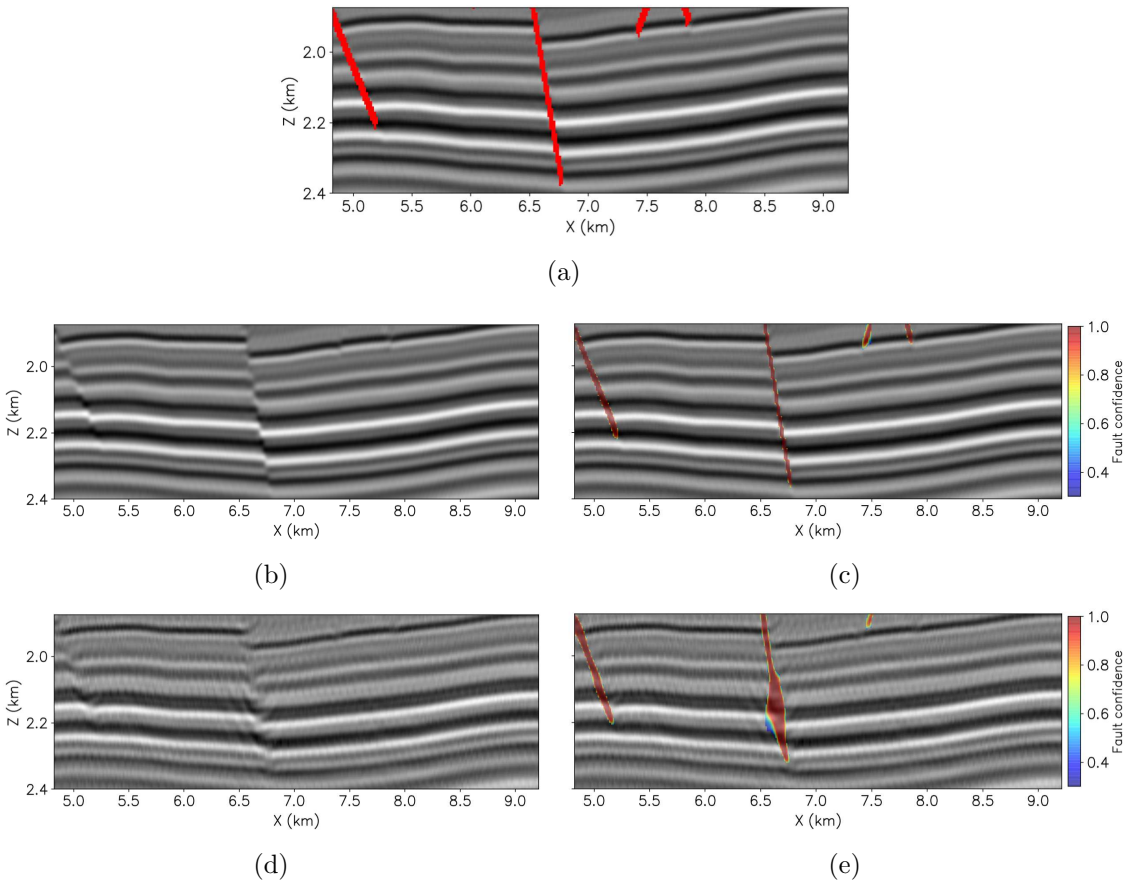


Figure 3.11: Fault segmentation used as a metric for fault focusing. (a) An unfocused fault within a synthetic image, (b) the computed fault confidence for (a), (c) the same synthetic image but with the fault now focused and (d) the computed fault confidence of (c). Panel (e) shows the ground truth label for the fault position superimposed on the fault. [**CR**]

loss. (For a more detailed explanation of the CNN architecture and training please see Appendix A).

Figure 3.11(c) shows the result of applying the 2D fault segmentation CNN to a focused synthetic seismic image. It is clear that the CNN is able to identify the discontinuities in the horizons and correctly assign those pixels as belonging to the faults. To demonstrate how the CNN predictions are affected by the lack of focusing of the fault due to velocity error, I applied a prestack Stolt residual migration to the image for $\rho = 0.96$. This unfocused image is shown in Figure 3.11(d). The artifacts in the image due to the overmigration are readily apparent on each of the faults present within the image. Figure 3.11(e) shows the result of fault segmentation applied to the unfocused image. Comparing the CNN prediction on the focused image to the image on the unfocused image, it is clear that the fault confidence on the central fault is spatially much more spread out and less geological in appearance than the fault confidence computed from the focused image.

In addition to the qualitative analysis on the focusing of the fault, if the position of the true fault location is known, I can use a quantitative metric in order to better quantify the focusing or lack of focusing on the fault. While for real seismic images, the true fault location is never known, a manual interpretation of the faults can be used or as I show for the 2D field data example, a proxy can be used for the true fault locations. For the quantitative metric, I use the metric known as the Jaccard similarity coefficient or intersection over union (IOU) (Jaccard, 1912). This metric is commonly used for assessing the quality of predictions from computer vision tasks, as it provides a measure of overlap between a predicted set of pixels and a target set of pixels. Mathematically, it can be computed as follows:

$$\text{IOU} = \frac{N_{11}}{N_{11} + N_{01} + N_{10}}, \tag{3.2}$$

where N_{11} is the number of predicted pixels that were true positives, N_{01} is the number of predicted pixels that were false negatives and N_{10} is the number of predicted pixels that were false positives. Using the fault label as the target, I compute the IOU of

the fault segmentation for the unfocused image as 0.438 and for the refocused image as 0.755. Comparing these results between both predictions, we observe a significant improvement in the IOU for the refocused image, which indicates an improved fault segmentation.

2D GOM FIELD DATA EXAMPLE

In the first numerical example of this chapter, I demonstrated on a synthetic subsalt image that the CNN-based focusing analysis could refocus a poorly-focused subsalt image from only analyzing the diffractions and reflections within the stacked image. In this next example, I also perform a focusing analysis but on a 2D field dataset from the Gulf of Mexico (GOM). The focusing analysis in this example will be performed on prestack image patches and as was shown in the previous example, will be used to refocus poorly focused normal faults. I will also offer a comparison between the CNN-based approach and a semblance-based approach and quantitatively demonstrate the CNN-based approach provides a better focusing of the faults than a semblance-based approach.

Data processing, velocity model building and depth migration

The data for this example were taken from a survey line acquired off the Texas coast of the Gulf of Mexico in the early 1980s (Claerbout, 1985). The data consist of 148 shots acquired with a streamer acquisition with a maximum offset of 3.4 km and are shown sorted in midpoint-offset coordinates in Figure 3.12. In order to create a depth velocity model for wave-equation depth migration, I needed to process and perform velocity analysis on each common-midpoint gather. As the source spacing and receiver spacing were the same for this dataset, the even and odd midpoints contained a different number of offsets (Claerbout, 2010). To overcome this issue, I split the data into even and odd midpoints and processed them separately. To process the midpoints, I first muted the direct arrival and the refracted energy, and then

performed a normal moveout (NMO) velocity analysis which resulted in semblance panels for each midpoint. I then picked the RMS velocity on each semblance panel which resulted in an estimate for the RMS velocity for all midpoints. I then performed a regularized Dix inversion for each RMS velocity trace to convert the RMS velocity to interval velocity (Fomel, 2007). Finally, I stretched the velocity from time to depth, providing an interval velocity in depth which I could use for wave-equation depth migration. The final estimated interval velocity used for migration is shown in Figure 3.13. As the original estimated migration velocity did not result in obvious defocusing of the image, I introduced a slow velocity anomaly in the overburden (between depths 0.5 - 1 km and laterally extending between 10 and 14 km).

To migrate the data, I sorted the midpoint-offset cube into common shot gathers and imaged each shot with a wave equation depth migration with a maximum frequency of 50 Hz and 41 subsurface offsets. The resulting image is shown in Figure 3.14. Examining the amplitudes near the faults, we observe an unfocused fault plane reflection on the central fault as well as weak diffracted energy from the fault positioned at approximately 10 km. In order to investigate the effects of a limited-aperture during image-focusing analysis, I applied a mute on the gathers in order to simulate data that had been acquired with only 1.0 km offset. Figure 3.15 shows the region of interest of the image used for this study as well as an extracted muted angle gather. Diffracted events on the fault near $x = 10$ km and the unfocused fault plane reflection on the central fault provide signs of undermigration. Also note that little to no curvature information is available along the aperture-angle axis due to the application of the mute.

2D Prestack Stolt residual depth migration

As I did for the synthetic example, my first step was to perform 2D prestack Stolt residual depth migration. I also chose to create residual migration images ranging from $\rho = 0.9 - 1.1$ with an interval of 0.00125. Also, as before, I applied a depth correction to each image in order to correct for depth shifts between the different

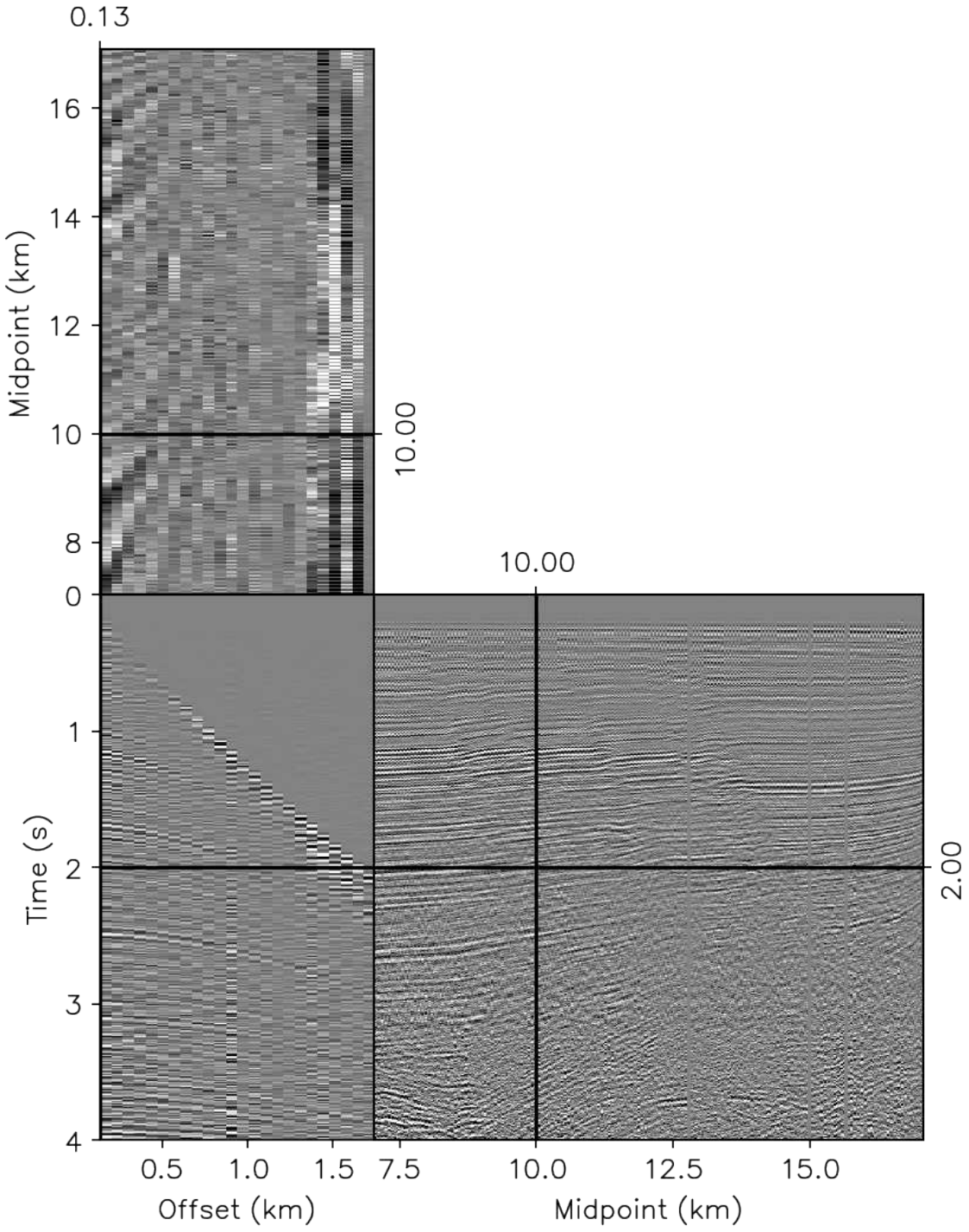


Figure 3.12: 2D GOM dataset used for the field data example in this chapter. [**ER**]

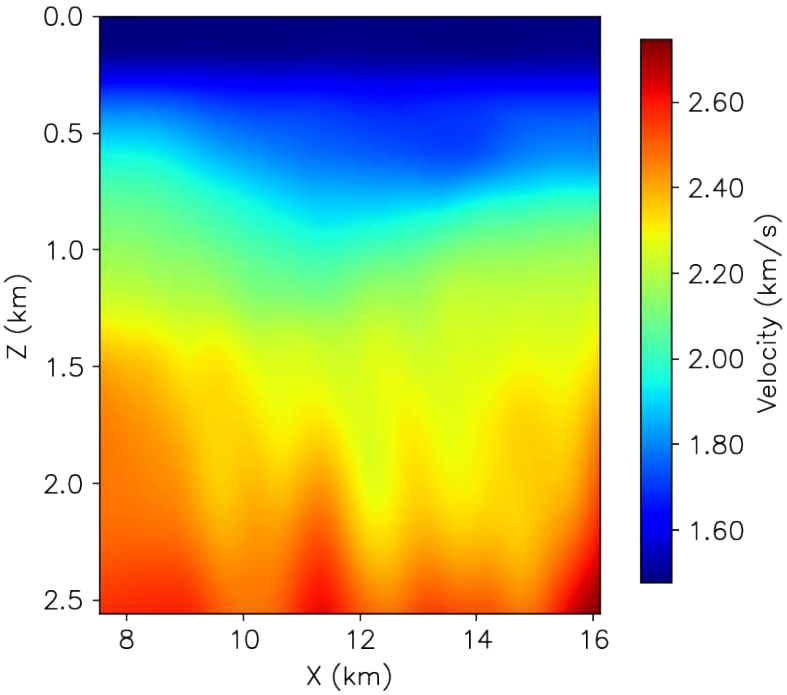


Figure 3.13: Estimated interval velocity used for wave equation depth migration. A slow velocity anomaly was introduced in the overburden to create defocusing within the image. [**ER**]

migrated images. Again, this resulted in a total of 161 residually migrated images that I could use for prestack image-focusing analysis. Figure 3.16 shows the region of interest for five residual migration images ranging from 0.94 - 1.06. Examining the different images in Figure 3.16 we see that while none of the images have been fully corrected for the velocity error, the image corresponding to $\rho = 0.97$ (Figure 3.16(b)) provides the best focusing of the fault plane reflections as well as the largest stack power.

Semblance-based focusing analysis

In order to quantitatively assess the focusing of the image, I computed semblance for an angle gather within the unfocused image. Figure 3.17(a) shows the result of computing semblance for $\rho = 0.9 - 1.1$ for an angle gather extracted at 11.3 km within the unfocused image. Again, I picked the maxima using the automatic picking

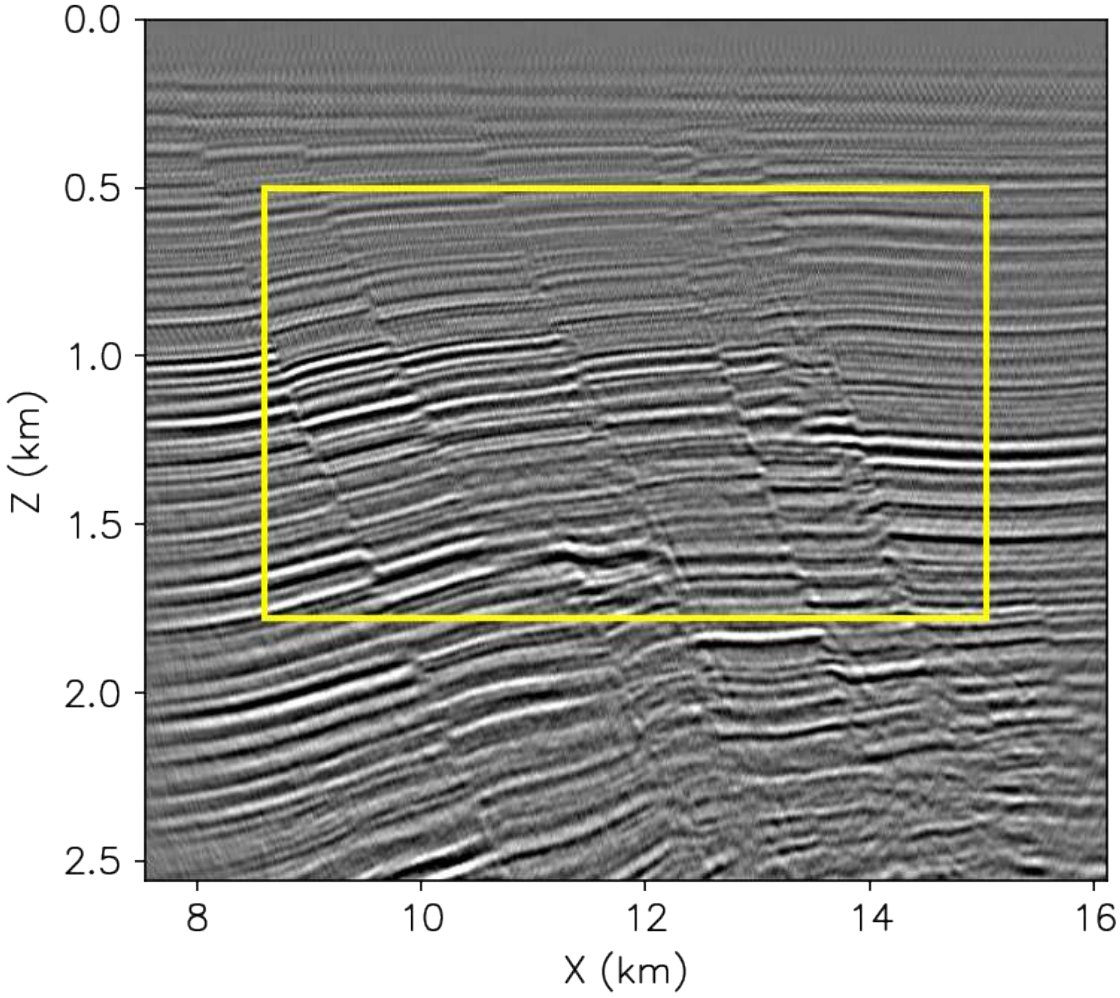


Figure 3.14: Unfocused depth-migrated image to be used in this study with the region of interest shown within the highlighted box. [**CR**]

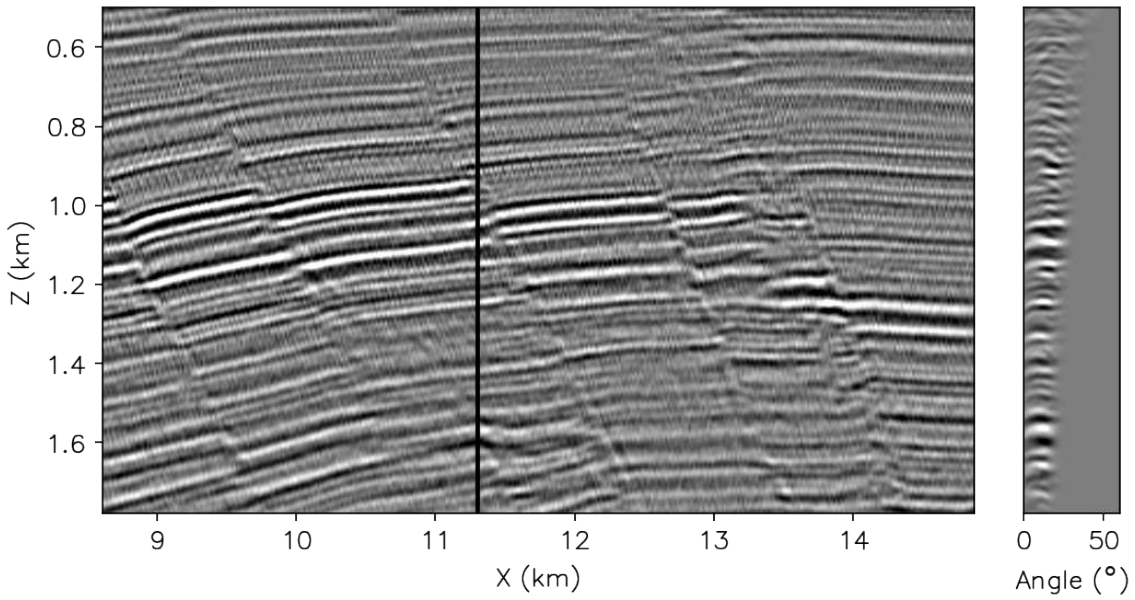


Figure 3.15: The region of interest shown in Figure 3.14 and a muted angle gather extracted at 11.3km. The mute applied on the aperture-angle axis has left little curvature information on the angle gathers. [**CR**]

algorithm described in Fomel (2009) and the picks are displayed in the cyan and black curves superimposed on the angle gather and semblance panels shown in Figure 3.17(a). While we can observe a clear trend in the semblance panel, the large width of the trend due to the limited-aperture angle information introduces considerable uncertainty in the pick. To better illustrate this, I performed a reference focusing analysis on the full aperture image. The semblance computed from the angle gather extracted from the same spatial location within the full aperture image is shown in Figure 3.17(b). While we can observe a tight and clear trend in the semblance panel in Figure 3.17(b), we can see a very broad smeared trend in the semblance panel in Figure 3.17(a). The picks of each of the semblance panels are also shown in Figure 3.17(b). There is a significant difference between the picks from the full and limited-aperture data, which indicates that an inaccurate pick resulted from the focusing analysis on the limited-aperture data.

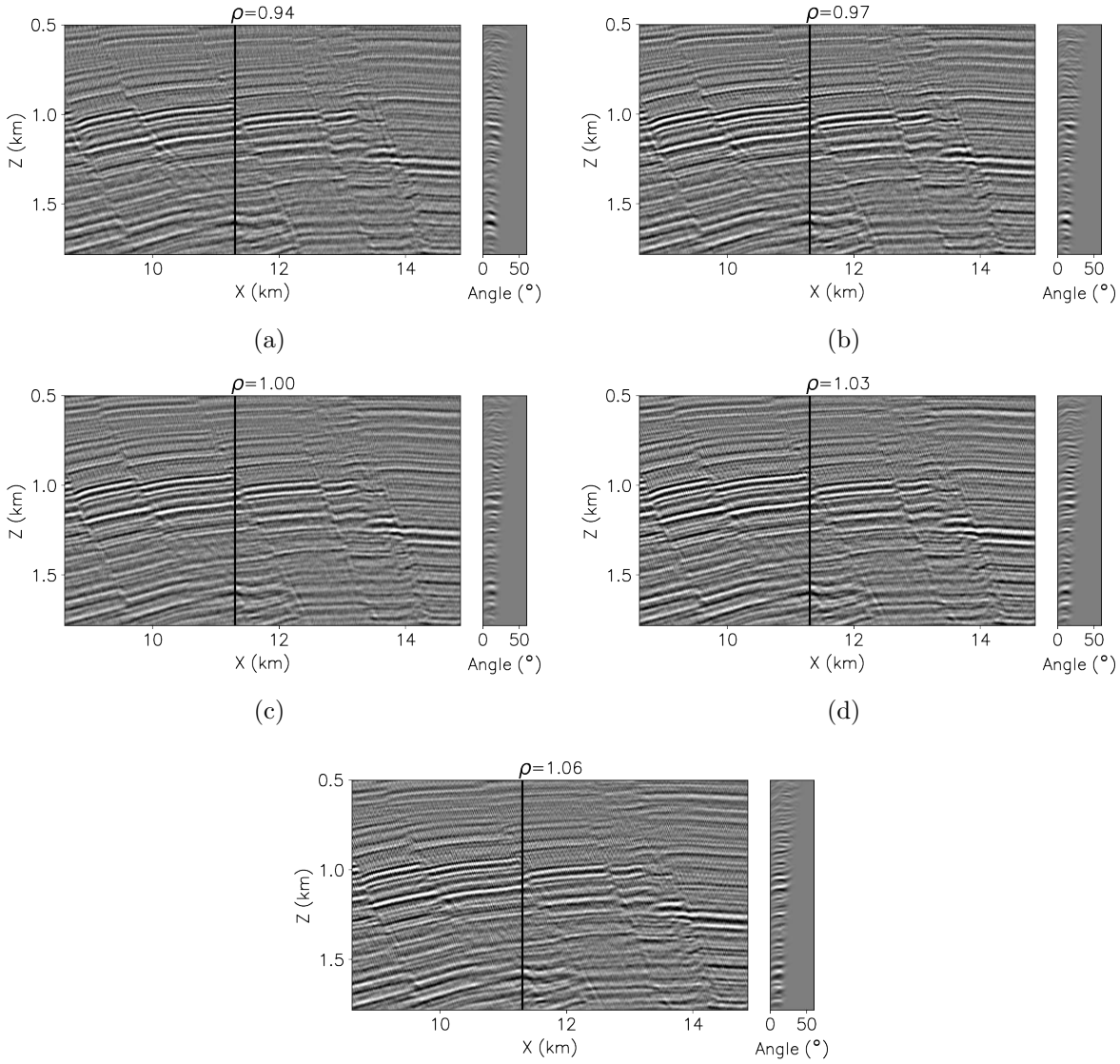
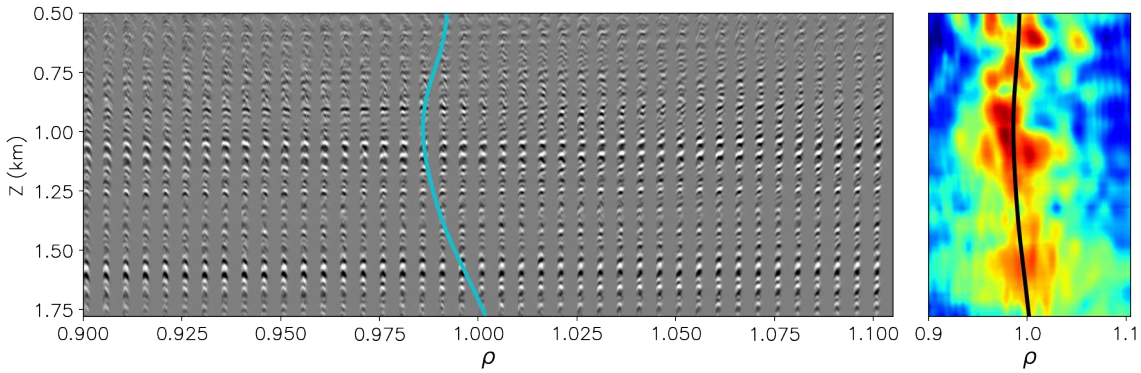
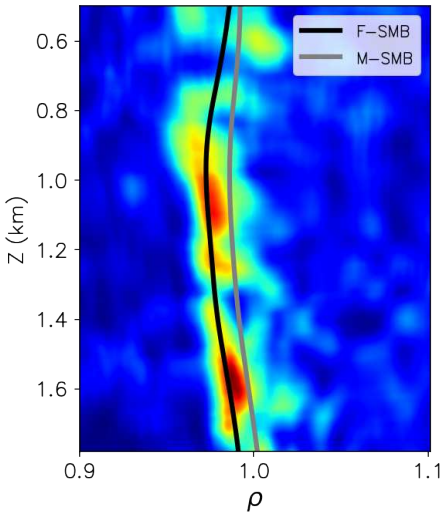


Figure 3.16: A selection of residual migration images taken from the application of prestack Stolt residual depth migration to the unfocused image. While it is clear that a single ρ value will not correct for all of the velocity error, we observe that the best focusing of the faults and largest stack power for $\rho = 0.97$ (panel (b)). [**CR**]



(a)



(b)

Figure 3.17: (a) Computed ρ semblance from an angle gather extracted at 11.3 km. The left panel shows the angle gather residually migrated for ρ values between 0.9 and 1.1. The right panel shows the computed semblance. The black and cyan curves show the pick of the maxima of the semblance panel. (b) Comparison of picked maxima from focusing analyses performed on full and limited-aperture images. The black curve shows the pick from the full aperture focusing analysis and the gray curve the pick from the limited-aperture focusing analysis (panel (a)). [**CR**]

CNN-based focusing analysis

The CNN-based focusing analysis for this field dataset closely resembles that described in Chapter 2 and shown in the previous subsalt example. I first created a training dataset, then I trained the CNN and finally used the CNN for image-focusing analysis. A key difference of this example is that the synthetic training images do not constitute the entire training set, but rather serve as a pre-training set that allows the CNN to be trained on relatively few field training examples. Additionally, a key component of the focusing analysis performed for this example is a workflow that I will describe for labeling real examples and then incorporating them into the training.

Training data creation

My primary strategy for training data creation was to create as many training examples as possible from the focused and unfocused images shown in Figure 3.16. To do this, I first formed patches from the limited-aperture unfocused image. As I did for the subsalt example, I chose a patch size of 64 depth samples $\times$ 64 lateral samples and selected all available angles (32 in total). As I did for the subsalt example, I chose to have 50% overlap in both z and x directions when forming patches which resulted in 225 spatial patches. After forming patches on all 161 residually migrated limited-aperture images, I obtained a total of $225 \times 161 = 36{,}225$ patches.

As these patches resulted from real seismic images, they needed to be labeled. To label the patches, I grouped all residually migrated patches that belonged to the same spatial location and labeled the "best focused" patch of this group as a focused patch (receiving a label of one) and then all other patches within the spatial patch group as unfocused (receiving a label of zero). For patches without faults and near the surface, which were largely unaffected by the aperture-angle mute, I could use the power of the stack as well as visual reflector coherence within the patch as a metric for selecting the best focused patch. For all other patches, I first performed automatic fault segmentation on the patch pertaining to the $\rho = 1$ image. If an image patch with a segmented fault contained enough fault pixels indicating that it contained at

least one fault, then I performed fault segmentation for all patches within the patch group. Then for each patch within the group, I visually labeled the patch that had the most coherent and accurate fault segmentation as focused and all other patches as unfocused. Figure 3.18 shows the fault segmentation for a patch for different ρ values. Qualitatively, the fault segmentation for the $\rho = 1$ image shown in Figure 3.18(b) appears to have the sharpest and most precise fault segmentation and therefore, this image patch received the focused label for this patch group.

While this approach for creating a labeled training set is advantageous in that it greatly facilitates model generalization and minimizes the data-domain gap, the labeling procedure can be time-consuming and the resulting labeled images are dependent on the individual performing the labeling. Additionally, this approach will generate far fewer focused training images than unfocused training images as only one out of 161 is labeled focused for each patch group. While this can still result in a relatively large number of training patches, it will lead to a largely unbalanced dataset that will likely bias the model to predicting unfocused patches. For this reason, I only selected one focused and one unfocused patch for each patch group. This will lead to a perfectly balanced dataset but has the downside of providing significantly fewer patches. Due to this reduction in the number of training patches, as well as my desire to not spend too much time on manually labeling patches, I only labeled 56 real data patches. Figure 3.19(a) shows examples of unfocused and focused real data training patches. Note that for the purposes of this example and to investigate generalization, all labeled field patches used for training were selected from outside of the region of interest.

To overcome this challenge of limited labeled real data training patches, I supplemented the field training set with a synthetic pre-training set. The addition of a synthetic pre-training set also has the advantage of helping to reduce biases and uncertainties that arise due to a manual labeling procedure. To create synthetic training image patches, I followed nearly the same procedure as I did for the synthetic subsalt example: I create a synthetic reflectivity model with undulating reflectors as well as normal faults. I then convolve this reflectivity model with a 20 Hz Ricker wavelet

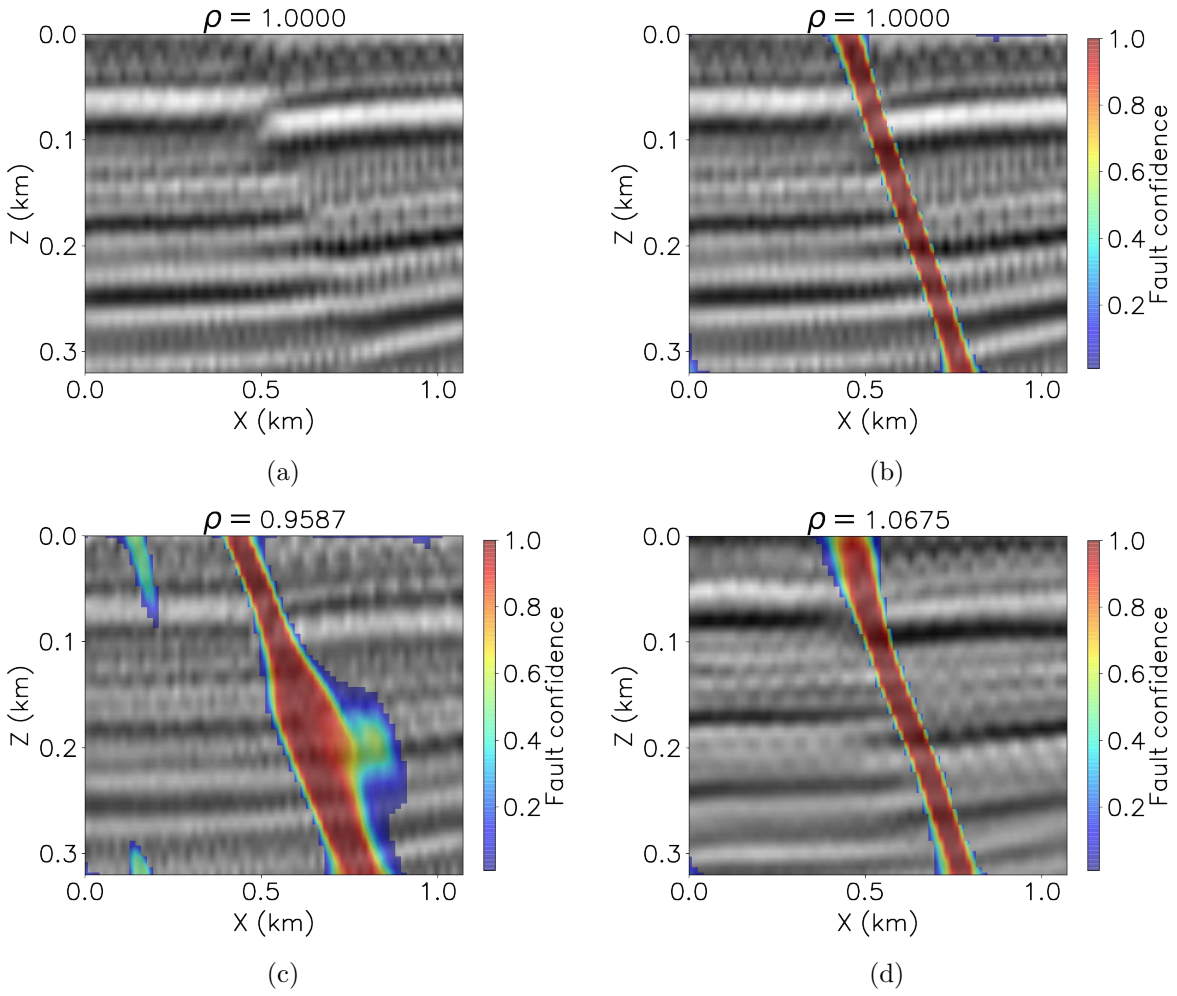


Figure 3.18: Fault segmentation on a training patch for different ρ values. (a) The image for $\rho = 1$ and the estimated fault confidence superimposed on the (b) $\rho = 1$ image, (c) $\rho = 0.9587$ image and the (d) $\rho = 1.0675$ image. Qualitatively, comparing the fault segmentations for the different residually migrated images, we observe that the $\rho = 1$ image has the most precise fault segmentation. [**CR**]

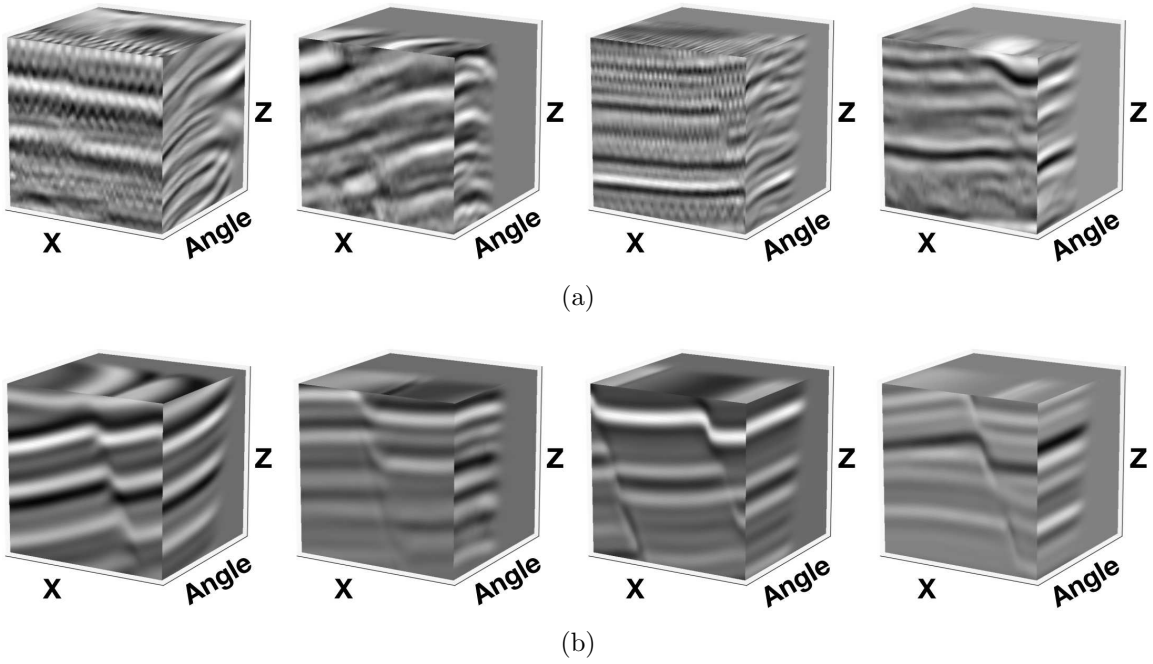


Figure 3.19: A selection of prestack training image patches used to train the fault-focusing CNN. (a) Labeled patches taken from the limited-aperture image and (b) labeled patches created from the synthetic training image procedure. For both panels (a) and (b), the two leftmost images are unfocused training patches and the rightmost images are focused training patches. [**CR**]

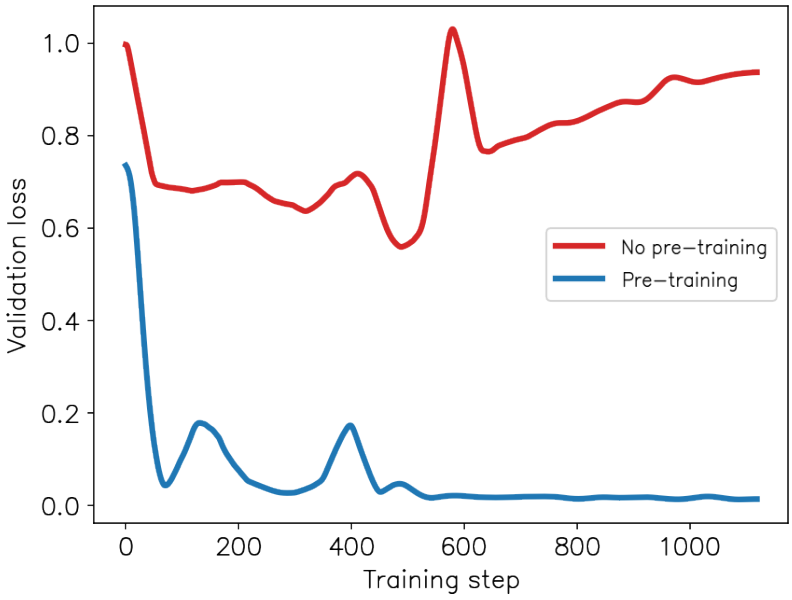
to create a band-limited synthetic seismic image. To minimize the domain gap between the synthetic and real images, I attempt to make the geological structure and frequency content of the synthetic seismic image similar to that of the real seismic image. I then simulate a subsurface-offset focused image and apply Stolt residual migration to create an unfocused image Finally, I convert both images to angle and apply a mute on the gathers to mimic the aperture range on the real seismic image. For each synthetic seismic image, I then created patches from each image using the same patch grid I used for the real seismic image. Performing this procedure for 1,000 models created 8,192 unfocused and focused synthetic pre-training patches Similar to Figure 3.19(a), Figure 3.19(b) shows examples of unfocused and focused synthetic pre-training patches.

Training the fault-focusing CNN

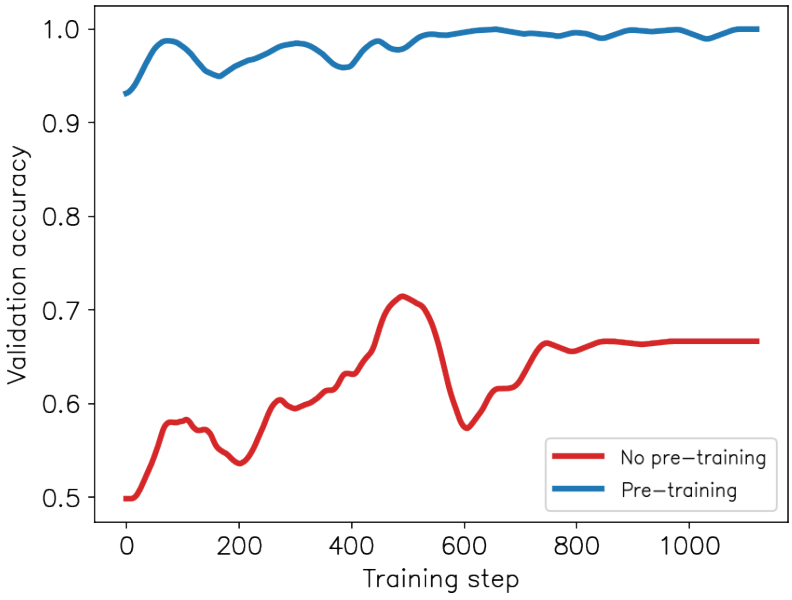
Now with two training datasets, one composed of synthetic and the other real seismic images, I split the training into stages as I did for the convolution-based and migration-based patches in the previous examples. For the pre-training stage, I specified a batch size of 20 and trained for 20 epochs and used 80% of the 8192 patches for training and 20% for validation. For the synthetic validation training set the CNN achieved nearly 100% accuracy in classification. In the secondary stage of training I trained on the 56 real data training patches and used an additional 20 for validation. I also trained for 20 epochs but used a batch size of one as I found that smaller batch sizes provided the best validation accuracy. I found that with the addition of the pre-training set, I was able to achieve very good classification accuracy on the validation data as opposed to the accuracy achieved by training with only the real data training patches. Figure 3.20 shows the comparison of the learning curves of the secondary stage of training with and without the pre-training step. We can observe that by first pre-training on the synthetic seismic images, the network is able to avoid overfitting the real data and to achieve nearly perfect accuracy.

Focusing analysis with the trained CNN

To test the trained fault-focusing CNN, I made predictions on image patches within the region of interest. Forming the region of interest into residually migrated patch groups, I then computed the fault-focusing score for each patch in the patch group and chose the ρ value corresponding to the patch that had the maximum focusing score (Equation 2.29). I assigned that selected ρ value over the whole patch and performed the same for all patches within the target region. I then smoothed the selected ρ values with a 0.5 km triangular smoothing filter along the lateral direction and a 0.15 km smoothing filter in depth. Figure 3.21(a) shows the predicted $\rho(z, x)$ superimposed on the unfocused image. We observe that the predicted $\rho(z, x)$ smoothly varies in the unfocused region between values of 0.96 and 0.98 in the region of most severe defocusing which is consistent with the image focusing observed



(a)



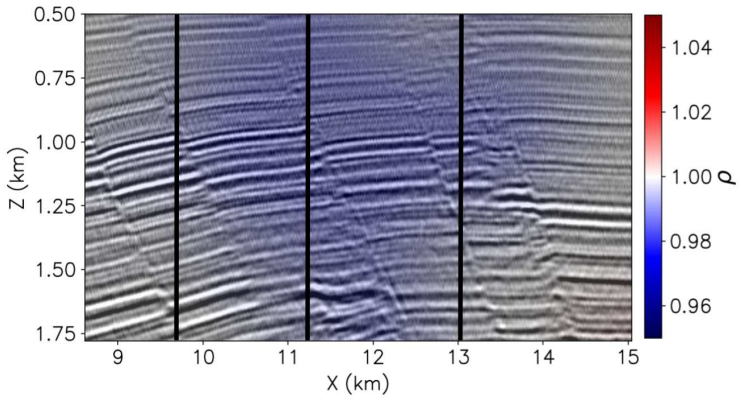
(b)

Figure 3.20: Learning curves with and without the pre-training stage. Panel (a) shows the comparison between the validation loss and panel (b) the comparison of the validation accuracy of the model with and without pre-training. Note to better display the trend of the curves, each of the curves was smoothed with a 50-point triangular smoothing filter. [**CR**]

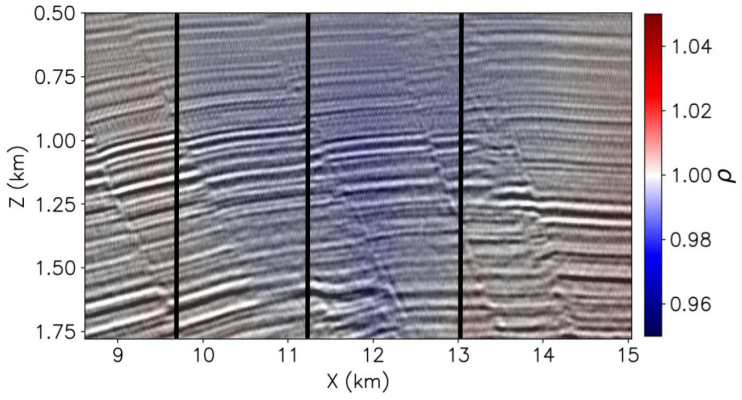
in the residual migration images (Figure 3.16). Figure 3.21(b) shows the estimated $\rho(z, x)$ from the semblance-based approach. While the two predicted $\rho(z, x)$ generally agree spatially, there are significant differences in magnitude in the predicted residual migration parameters. Additionally, we can observe that the estimated $\rho(z, x)$ from the semblance-based approach oscillates above and below $\rho = 1$. In order to better compare the estimated $\rho(z, x)$, I performed a semblance-based focusing analysis for all spatial locations within the full-aperture image. The results of this focusing analysis are shown in Figure 3.21(c). Comparing all three estimated $\rho(z, x)$, we observe both spatially and in terms of magnitude that the CNN-based approach agrees quite well with the full aperture semblance-based focusing analysis. Figure 3.22 shows a comparison of the estimated $\rho(z, x)$ for all three focusing analyses, taken at $x = 9.6$, 11.3 and 13 km. For both panels (b) and (c) of Figure 3.22, there is near perfect agreement between the full aperture analysis and CNN analysis. We can also observe good agreement in panel (a), but we observe that the $\rho(z)$ from the CNN-based approach diverges from the full-aperture semblance approach below a depth of about 1.2 km.

QC and comparison of results

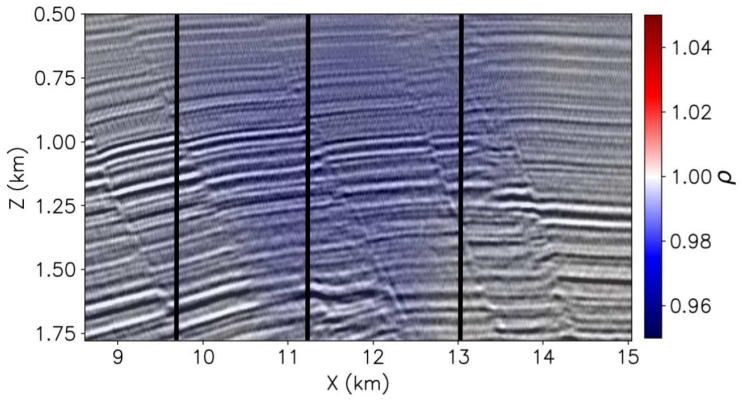
Again, to perform a QC of the estimated $\rho(z, x)$ from the CNN and semblance-based approaches, I obtained refocused images via the application of Equation 3.1 to the residually migrated cube of stacked images. Panels (b)-(d) of the left column of Figure 3.23 show the results of performing this correction with the ρ estimated from the CNN-based approach, the ρ estimated using a semblance-based approach on the limited-aperture data and the ρ estimated from the reference full-aperture focusing analysis. Comparing these panels with the original unfocused image in the left column of panel (a), we observe significant improvement in the focusing of the faults for each of the corrected images. However, we can observe some residual defocusing in the muted semblance image (panel (c)). This is more readily apparent in the fault segmentation of each image shown in the right column of Figure 3.23. Qualitatively, comparing the central fault of the fault segmented images in panels (b)-(d), we observe



(a)

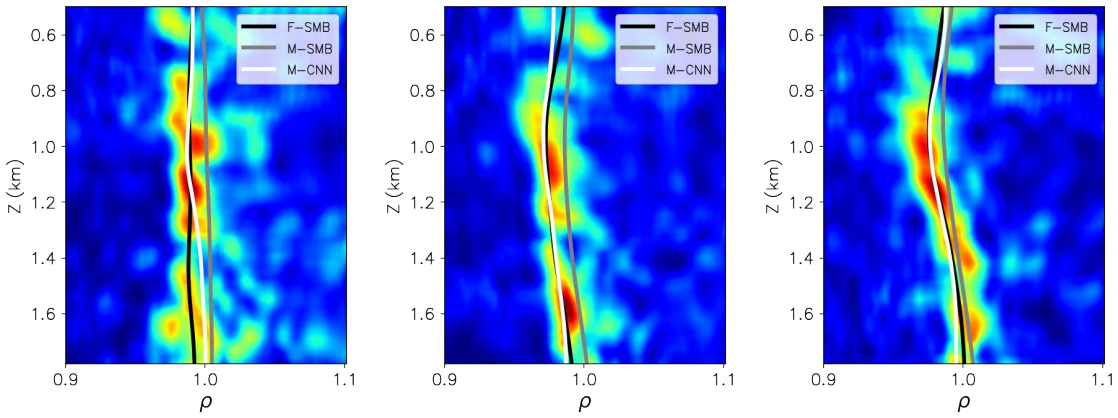


(b)



(c)

Figure 3.21: Comparison of estimated $\rho(z, x)$ for (a) my CNN-based approach, (b) the aperture-angle based approach and (c) the aperture-angle based approach computed on the fully illuminated dataset included for reference. The black vertical bars indicate the spatial locations at which the $\rho(z, x)$ values were extracted and compared in Figure 3.22. [**CR**]



(a) (b) (c)

Figure 3.22: Comparison of $\rho(z)$ extracted at (a) x=9.6km, (b) x=11.3km and (c) x=13km. Each semblance panel was computed from the full-aperture data focusing analysis. The gray curve shows the result of performing semblance-based focusing analysis and the white curve the result from the CNN (both on the limited-aperture data). The black curve shows the pick of the semblance panels for reference. [**CR**]

at approximately 0.8 km depth that the segmentation of panels (b) and (d) is better resolved than that of panel (c).

To provide a quantitative comparison of the observed improvement in the fault segmentation, I computed the IOU for assessing the quality of the predicted faults in the images shown in Figures 3.23(a) - 3.23(d). To compute the IOU for each of these images, I first applied a threshold to the fault predictions in each of the images where any pixel containing less than 0.5 fault confidence was set to zero and all others received a value of one. I then specified the segmented faults from the reference focusing analysis as the ground truth fault locations which allowed me to compute N_{11}, N_{01} and N_{10} in Equation 3.2. Table 3.1 shows the computed IOU values from the images shown in Figures 3.23(a)-3.23(c). Note that there is a significant improvement in the IOU of the faults segmented within the CNN corrected image compared to both the original unfocused and semblance corrected image.

	Unfocused	**Semblance**	**CNN**
IOU	0.58	0.72	0.83

Table 3.1: Computed IOU metrics for the faults segmented within the original unfocused image (Figure 3.23(a)), the image corrected after semblance-based focusing analysis (Figure 3.23(c)) and the image corrected after my CNN-based focusing analysis (Figure 3.23(b)).

DISCUSSION

The results of the focusing analysis shown in the previous section indicate that the CNN can extract focusing information from the faults within the image when there lacks sufficient information along the aperture angles. I demonstrated that I could achieve nearly the same results of a full aperture focusing analysis, with a very limited-aperture. For the spatial locations where the results differ, I believe that this difference could be attributed to a few different reasons. One reason could be due to the lack of geological focusing information within the image. For example, comparing the different faults in Figure 3.23, I observe that although the velocity is incorrect, the left-most fault has been largely unaffected due to the velocity error, while the central fault is very much unfocused. The fault segmentation of the different images readily shows the difference in focusing of the faults. In scenarios such as these, it will be more challenging for the CNN to perform an accurate focusing analysis. Another possible source of uncertainty that must be considered when training an image-focusing CNN is the labeling of field data examples. While manual/interpretative focusing analysis has shown to be quite successful for migration scanning methods, manual interpretation and labeling will most likely contain errors in labeling images as focused or unfocused. While this is true for all supervised learning techniques, my approach is more robust to these errors in that at relatively little cost, I can provide a large amount of pre-training images to the CNN that are exactly focused or unfocused. Moreover, the exact classification of an image patch as unfocused or focused is not of the utmost importance. Rather, I require that the CNN provide a higher relative focusing score to patches that are better focused. This relaxes the accuracy requirement for the CNN for classification and instead allows the CNN to act as a

data-driven focusing measure of seismic images with faults. Lastly, there is always the issue with providing enough training data for the network to generalize to many images. I suspect that when applying the fault-focusing CNN to different seismic images, a small amount of training image patches from each new image may need to be provided to the CNN for training. This will be especially true in cases in which acquisition parameters or the geology varies greatly between surveys.

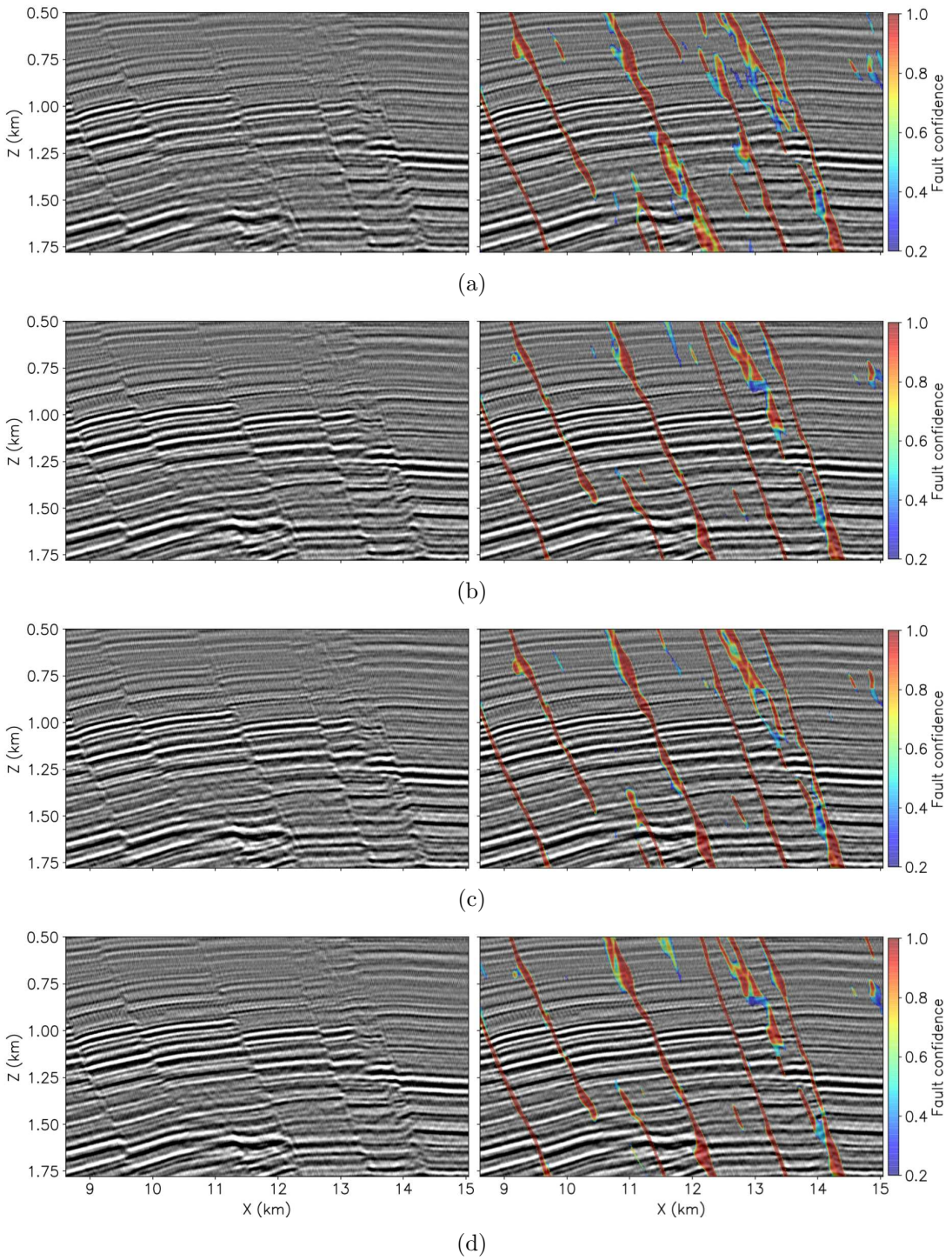


Figure 3.23: Comparison of refocused images and fault segmentation. The left column shows the image and the right column is the image with the fault segmentation of (a) the original unfocused image, (b) image refocused with CNN, (c) image refocused with semblance and (d) reference image computed from full-aperture focusing analysis. [**CR**]

Chapter 4

Application to 3D Netherlands F3 data

In this chapter, I describe the extension of my CNN-based focusing analysis to 3D and show the application of 3D CNN-based focusing analysis to a real 3D seismic dataset. The field data example is from the F3 block of the Dutch sector of the North Sea and contains several large faults oriented in the crossline direction that I use for interpretative image-focusing analysis. I begin by providing an overview of the dataset and then describing the processing, velocity model building and depth imaging applied to the data to create a prestack image cube as input for a 3D focusing analysis. I then perform common-azimuth Stolt residual depth migration, which creates 3D residually focused prestack images. With these images, I then perform a semblance-based focusing analysis that provides semblance panels that are automatically picked. This yields an estimate of $\rho(z, x, y)$. Then, I describe the extension of the 2D focusing analysis described in Chapters 2 and 3, to 3D. This requires constructing a CNN that makes use of 4D cross-correlations on an input prestack image patch. After training the 4D CNN on a mixture of synthetic and field training images, I use it to perform focusing analysis on the prestack image cube and estimate $\rho(z, x, y)$. Finally, I compare the refocused images that result from the estimated $\rho(z, x, y)$ from both the semblance and CNN-based approaches and show that in general, the CNN-based approach provides better focusing of the faults than the semblance-based approach.

DATASET OVERVIEW

Geologic setting

The 3D dataset to which I applied semblance-based and CNN-based focusing analyses were acquired in 1989 by Nederlandse Aardolie Maatschappij (NAM) in the Netherlands F3 block of the North Sea which is located within the Dutch Central Graben (Duin et al., 2006). The data were provided to SEP by the Geologic Survey of the Netherlands (TNO) in October 2020. Figure 4.1 shows the location of these data within the Dutch sector of the North Sea. These data were originally processed and imaged by Western Geophysical (Tachon et al., 1989) and the time-migrated cube that resulted from their processing and imaging was made publicly available via the Dutch Mining Act (Zima, 2003) and further publicized by dGB Geosciences with the release of their open-source seismic interpretation software known as OpendTect (de Groot and Bril, 2005).

In addition to the public availability of the migrated volume as well as 26 well logs, this dataset has been attractive to researchers due to the considerable amount of geologic variety that can be interpreted within the migrated cube. Figure 4.2 shows a 3D display of three slices of the time migrated cube produced by Western Geophysical. From 0 - 1 s we can observe very clear sigmoidal bedding in the stratigraphic layers that belong to the Upper North Sea group (Song et al., 2017). Continuing down after 1 s, there exists a heavily faulted shale layer with a chaotic appearance. This chaotic appearance can be attributed to polygonal and conical faulting that has occurred within this layer (Kuhlmann and Wong, 2008; Hale and Groshong Jr, 2014). Continuing into the subsurface at approximately 1.5 s in two-way time, there exists a chalk layer that exhibits a particularly high seismic amplitude relative to the surrounding sediments. Finally, beneath the chalk layer we can observe that from 0 - 12.5 km, large crossline faults (faults with a strike oriented primarily in the crossline direction) extend over 10 km in length and from 12.5 - 25 km a large salt diapir that was formed during the Zechstein period (Remmelts, 1996). In addition to these features observed within these three slices of the cube, the image contains a number

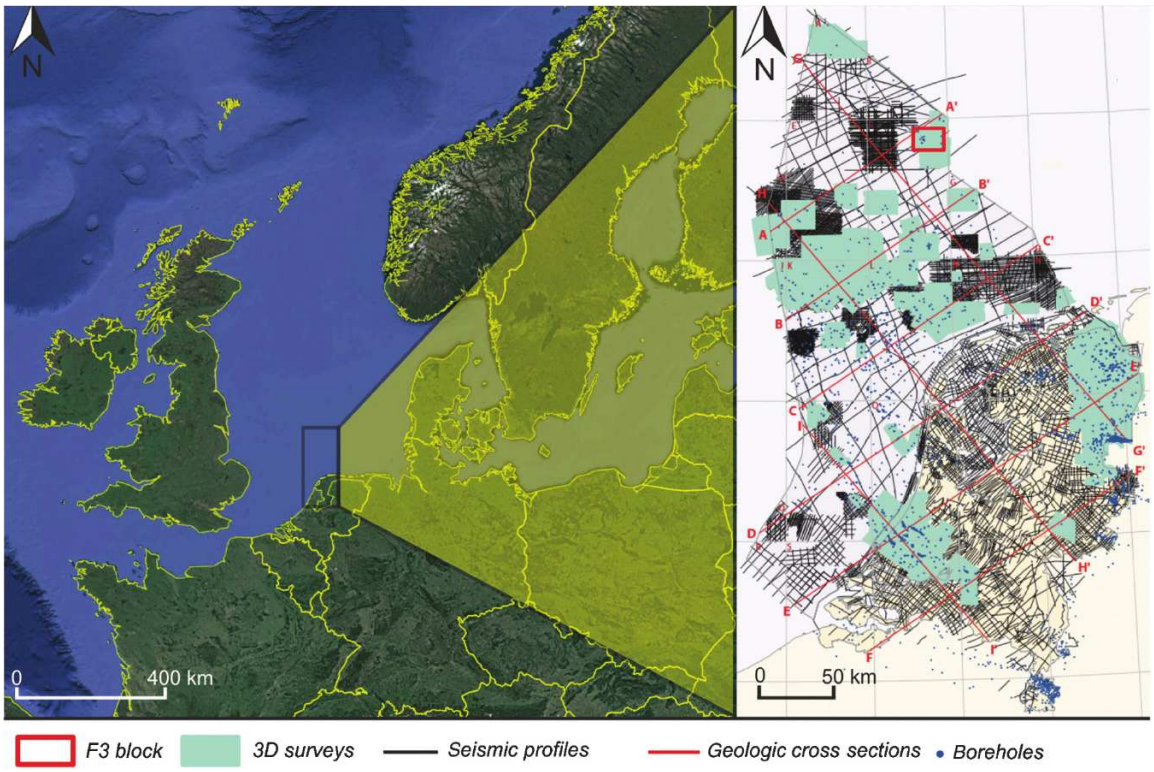


Figure 4.1: Location of F3 block within the Dutch sector of the North Sea. Figure courtesy of Alaudah et al. (2019). [**NR**]

of areas with shallow gas as well as bright spots (Schroot and Schüttenhelm, 2003; Guo et al., 2014).

The geologic variety just described has made this migrated cube the subject and test case of many seismic interpretation algorithms over the past decade. In fact, multiple authors have annotated and labeled this dataset so that it may be used for supervised learning and a potential benchmark for machine learning models (Silva et al., 2019; Alaudah et al., 2019; Campos Trinidad et al., 2021). Of the geologic features present within the image, I will be concentrating on the focusing and interpretation of the large crossline faults shown within the region of interest in Figure 4.2. I will create my own image starting from the raw seismic traces and will perform focusing analysis on this image. To my knowledge, this thesis is the first published

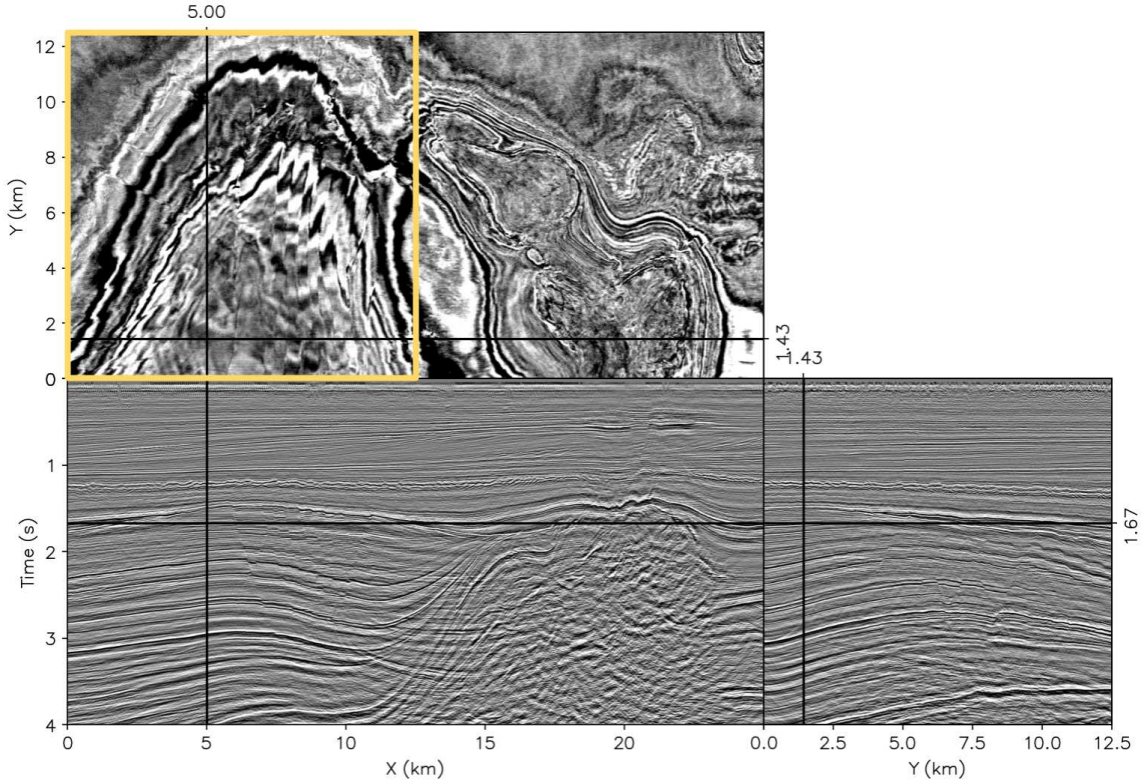


Figure 4.2: Time migrated cube created by Western Geophysical and provided to SEP by TNO. The yellow box on the depth slice indicates the lateral region of interest over which I performed depth imaging and focusing analysis. [**ER**]

work that shows the results of depth imaging on the raw prestack data from the F3 block.

Acquisition geometry

The data were acquired using a legacy acquisition geometry known as quad/quad (Vermeer, 2012). The quad/quad method of acquiring seismic data employs two source vessels each towing two streamers and each with two airguns. This setup was popular in the earlier days of 3D seismic acquisition as it potentially allows for 16 midpoint/image lines to be acquired for every sail line, greatly reducing the

total time of acquisition. It was later discovered that in spite of this added efficiency, quad/quad surveys typically result in low-fold data (Vermeer, 2012). Figure 4.3 shows a crossline schematic of the quad/quad acquisition used to acquire the F3 data. The spacing between streamers was 100 m and the spacing between sources was 50 m. Additionally the distance between the sources and the nearest streamer was 25 m. This setup resulted in a total of 12 midpoint lines where the four innermost lines received twice the fold of those on the edges.

Each of the four streamers consisted of 120 channels with a group interval of 25 m. With a near-source offset of approximately 100 m, the maximum inline offset was approximately 3 km. Each streamer was positioned at a depth of 7 m and the source depth was 5 m. Each of the vessels used a flip-flop airgun firing scheme where the vessels would alternate by first firing their starboard airguns (lead vessel, then follower vessel) and then their port airguns. Each gun would then fire every 75 m resulting in an inline shot spacing of 18.75 m. With this acquisition configuration, a total of 101,219 shots were acquired. Figure 4.4(a) shows every other source coordinate plotted on a time slice extracted at 1.68 s and every 1000th receiver coordinate is shown plotted in Figure 4.4(b). From 0.0 km to approximately 7.4 km, the vessels traversed in the negative x direction, and beyond 7.4 km, the vessels traversed in the positive x direction. Figure 4.5 shows the geometry of a single shot positioned at $x = 6.4$ km and $y = 1.13$ km. The red star shows the source position and the green triangles show the positions of the receivers on the four streamers. At the time of the shot, the vessel was traversing in the negative x-direction.

DATA PROCESSING

My first step in processing the data was to window the shots to create a sub-dataset for creating an image within my region of interest. As Figure 4.2 shows, the region of interest extended from 0.0-12.5 km in the y-direction and 0.0-12.5 km in the x-direction. Then, in order to make imaging this region more computationally feasible, I created a spatial grid of 100 $\times$ 100 m and for each grid point in the grid, selected

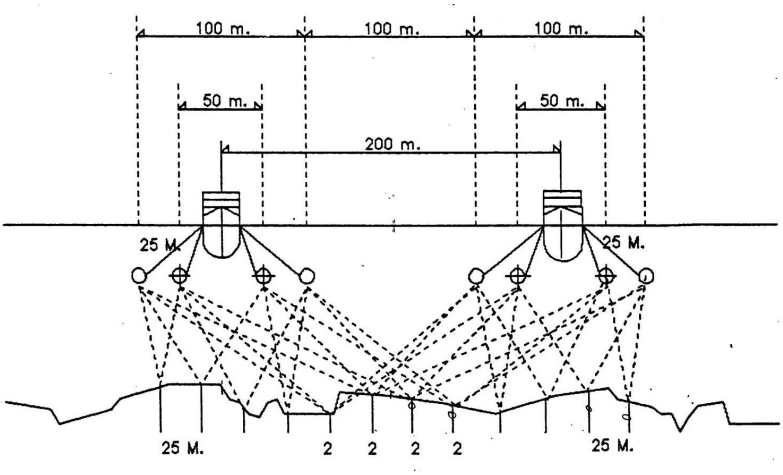
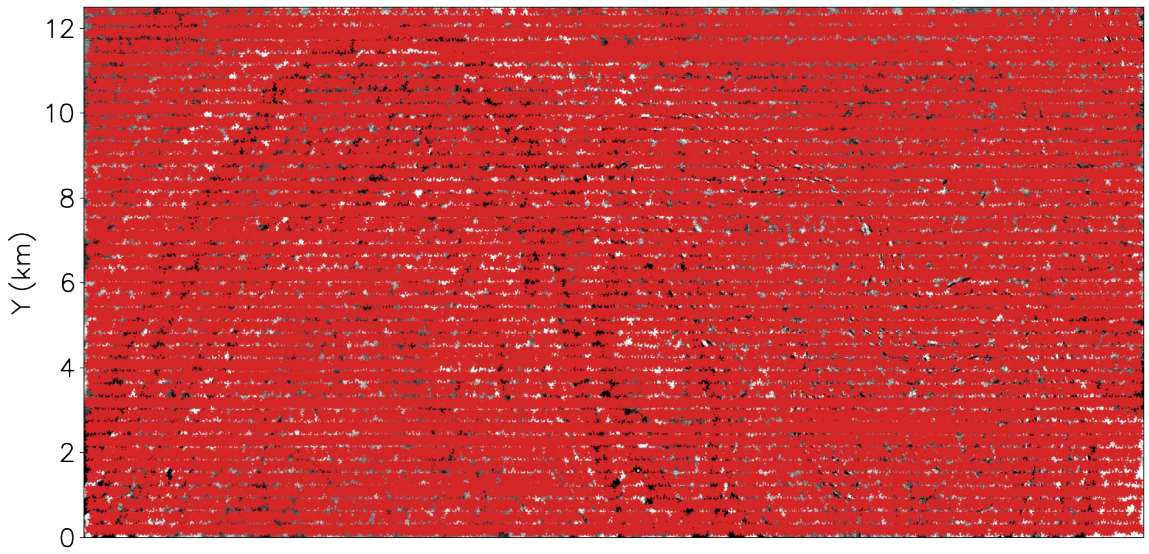


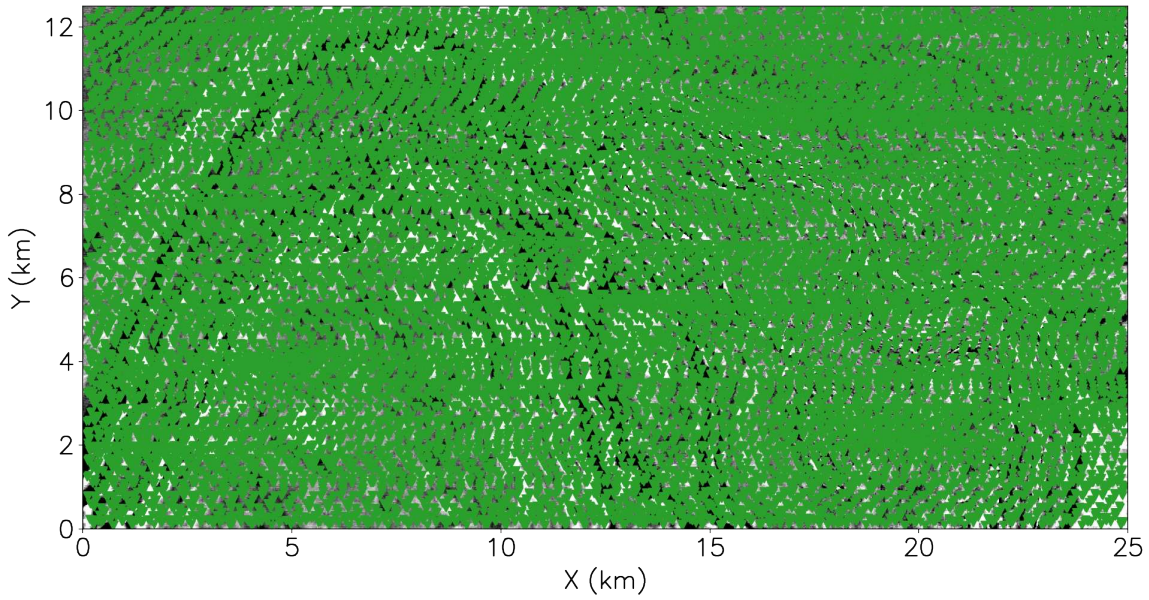
Figure 4.3: A schematic of the quad/quad acquisition geometry used to acquire the F3 data. Image courtesy of Smith et al. (1989). [**NR**]

the nearest shot to the grid point. This resulted in a total of 15,625 shots for imaging. Figure 4.6 shows each of these shots plotted on a time slice windowed to the region of interest. With the data windowed, I needed to apply processing before they could be imaged. My first step in processing was to QC each of the shots and remove any shots that contained any errors in the acquisition geometry or were labeled in the trace headers as bad traces. Then, I resampled the data from 2.0 to 4.0 ms sampling. Then, on the resampled data, I muted the refracted energy and the direct arrival from each streamer within each shot. To design a mute for a streamer, I computed the travel time of the near offset trace using a water velocity of 1.5 km/s and then for all subsequent traces, used a hyperbolic travel time equation assuming a receiver spacing of 25 m and the same water velocity. With the travel time of the direct arrival computed, I then constructed a mute function as a sine-tapered weighting function for each trace, which I then applied to the trace. (Claerbout, 2010).

After muting, I removed the airgun bubble from the data using gapped deconvolution (Yilmaz, 2001; Claerbout, 2014). To perform this debubble step, I first needed to select a trace from the data from which I could estimate a debubble filter. I selected the 20th trace within the shot with position $x = 0.064$ km, $y = 0.143$ km. While the exact choice of the trace for estimating the debubble filter is not crucial, I chose this trace as it appeared to have a strong and clean bubble signal. To estimate the debubble filter, I specified the filter to have 30 coefficients and a gap of 10 samples.



(a)



(b)

Figure 4.4: (a) Every other source coordinate for the F3 dataset and (b) every 1000th receiver coordinate plotted on a time slice extracted at 1.68 s. [**CR**]

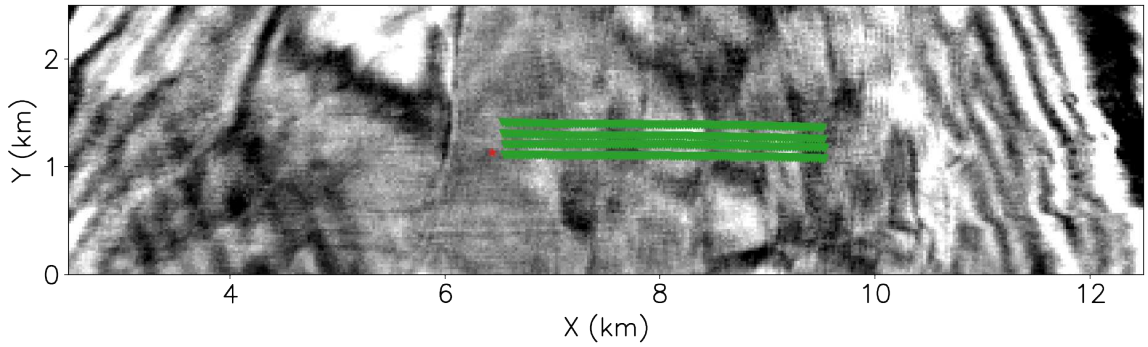


Figure 4.5: The acquisition geometry of a single shot positioned at $x = 6.4$ km and $y = 1.13$ km. The red star indicates the position of the source and the green triangles indicate the position of the receivers. [**CR**]

After 300 iterations of conjugate gradient inversion, the residual appeared to have little to no bubble signal, which indicated the estimated coefficients were suitable for predicting the bubble in the data. After debubbling, I then needed to apply a gun-and-cable static correction in order to bring both the source and the receivers up to the same sea level datum. This static correction can be easily computed by computing the sum of the airgun and streamer depths and dividing by the water velocity. Using the 5 m depth of the source and the 7 m depth of the streamer and 1.5 km/s water velocity, this resulted in a +8.0 ms correction to apply to the data. As the final step for processing these data, I performed a trace-by-trace denoising, in which I computed the energy of each trace, and if this quantity exceeded a specific threshold, I removed the trace and interpolated it with nearest neighbor interpolation. Figure 4.7(a) shows an example of a raw gather, and Figure 4.7(b) shows an example of a fully processed gather used as input for depth migration.

VELOCITY MODEL BUILDING

In order to perform depth imaging on the processed data, I needed to construct a 3D velocity model in depth. Fortunately, TNO was able to provide SEP with the time migration velocity picks from a DMO velocity analysis which provided me with a good

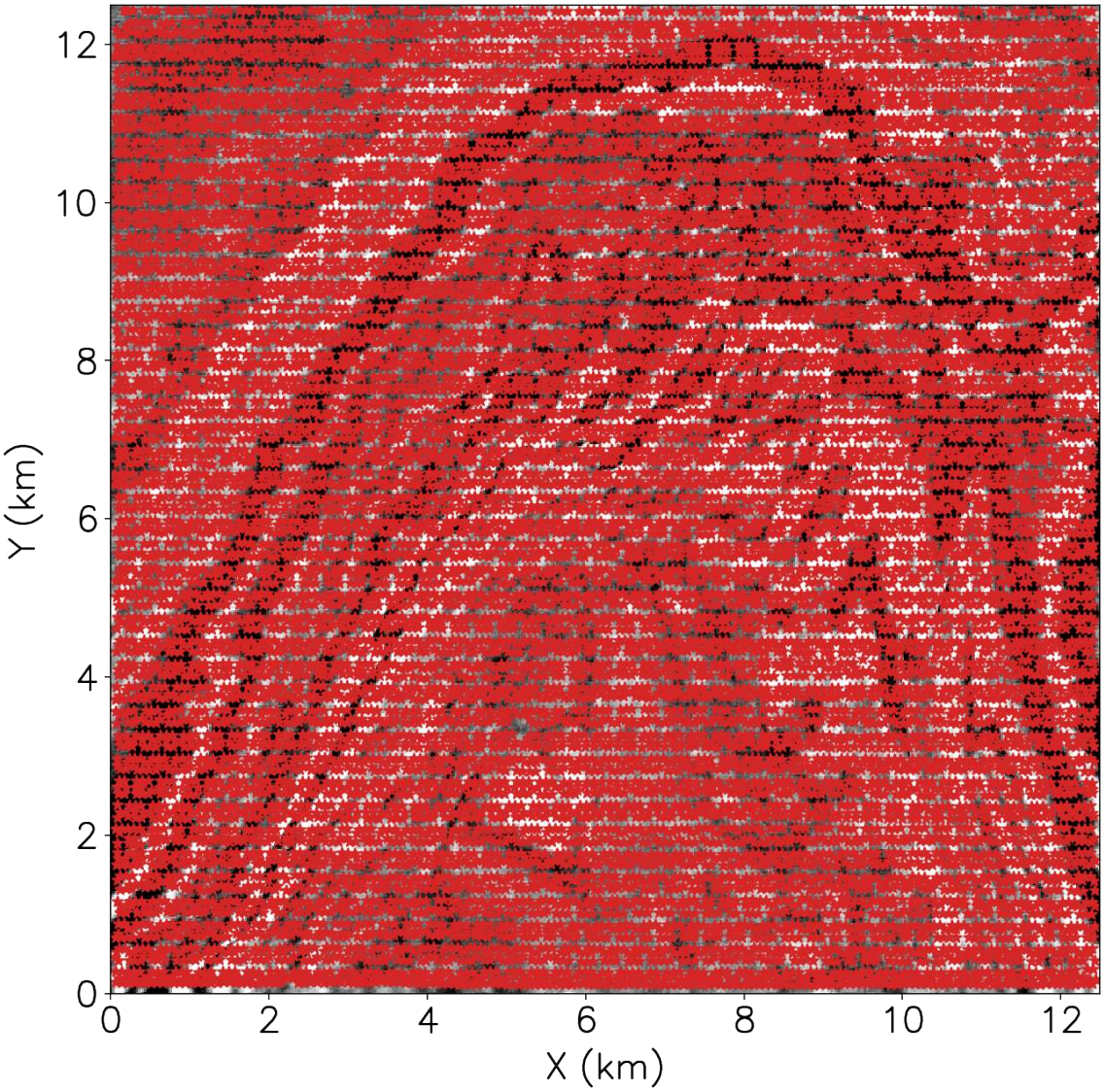
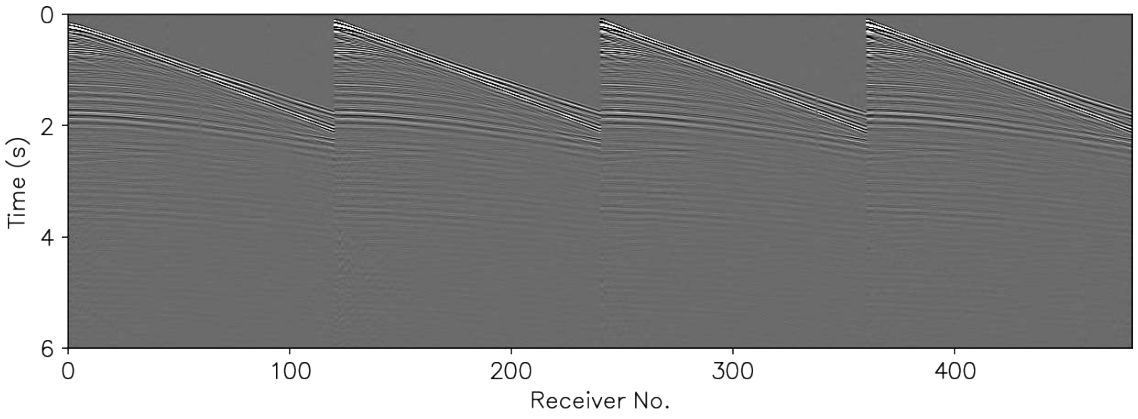
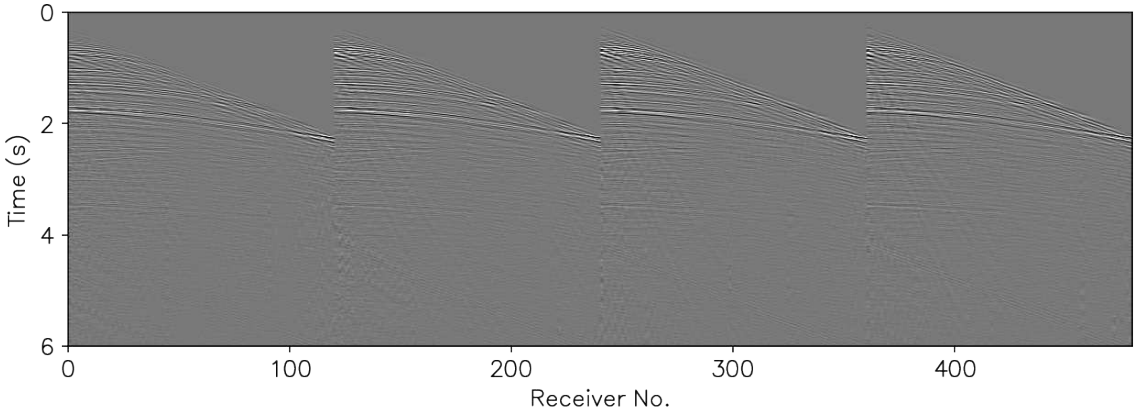


Figure 4.6: The source coordinates used for imaging the data within the region of interest plotted on a time slice extracted at 1.68 s. [**CR**]



(a)



(b)

Figure 4.7: (a) An example of a raw shot extracted from the SEGY and (b) the same shot after muting, debubbling, gun-and-cable static correction and trace denoising. [**CR**]

starting point for building an interval velocity in depth. To begin, I first parsed the time migration velocity files, which contained the picks for particular image points sampled at every 100 ms. Figure 4.8 shows the picks extracted from the pick files and plotted on a time slice extracted at 1.6 s. With these picks, I then performed a linear interpolation in time and a bilinear interpolation in space for each interpolated time slice, which resulted in a regularly sampled velocity model of dimensions 1500 time samples $\times$ 1000 inline (x) samples $\times$ 500 crossline (y) samples and sampling intervals of 4 ms in time and 25 m in space for both inline and cross line dimensions. I then smoothed this gridded velocity model with a 3D triangular smoother of 1.5 km in length in the inline and cross line dimensions and 0.12 s in time. Then, to create an interval velocity model in depth, I performed a regularized Dix inversion for each velocity trace (the same method that I described in Chapter 3) and then stretched the velocity from time to depth with a depth sampling interval of 5 m. Finally, I windowed the velocity model to the region of interest, thereby creating a velocity model ready to perform depth imaging. The final migration interval velocity model is shown in Figure 4.9

DEPTH IMAGING

With the data fully processed and the estimated interval velocity in depth, I then imaged all 15,625 shots with a 3D extended shot-profile wave equation migration that used a maximum frequency of 60 Hz and 41 subsurface offsets. Additionally, as the original estimated migration velocity did not result in any obvious defocusing, I introduced a constant velocity error by scaling the migration velocity by 0.98. The resulting depth-migrated cube is shown in Figure 4.10. The crossline faults of interest are clearly apparent in the depth slice at $z = 1.43$ km. Also apparent are two graben blocks present at approximately 6 and 8 km in the crossline slice. While the faults are visible in both inline and the crossline directions, there are also a number of migration smiles present within the crossline direction, which create a considerable amount of noise in the crossline image as well as the depth slice. This coherent noise had the potential of causing issues for automatic interpretation as well as focusing analysis. In

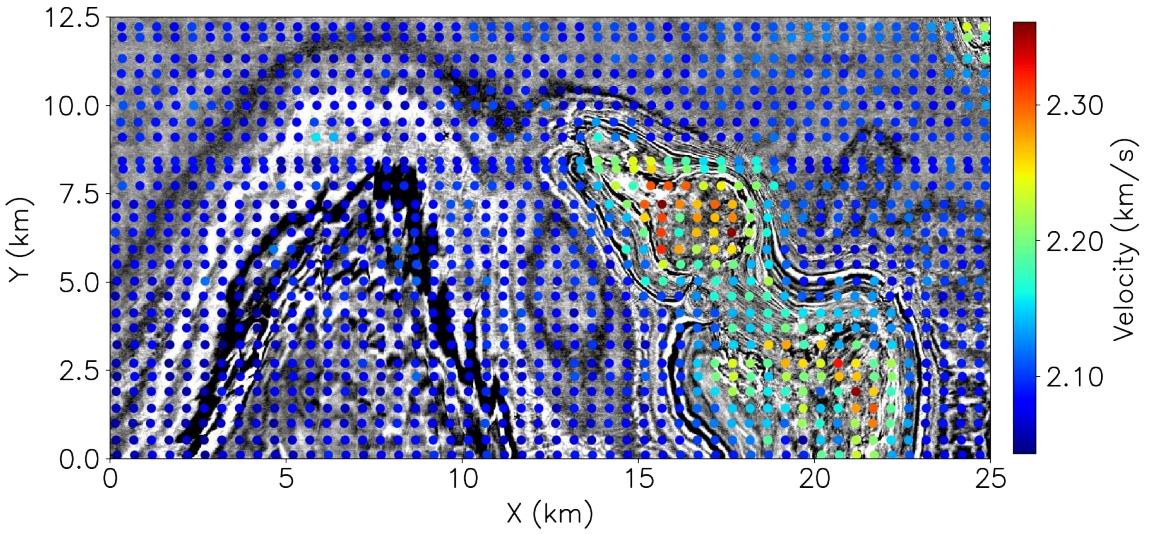


Figure 4.8: A time slice at t = 1.6 s of the RMS velocity picks provided by TNO plotted at their respective image points. Note that the increased velocities at the position of the salt body indicate that the velocities agree with the subsurface geology. [**CR**]

order to overcome these issues, I applied a dip filter in the crossline direction for each subsurface offset image where the maximum dip in the frequency domain allowed was 6 km^{-1}. In addition to the dip filter, I also applied a structure-oriented smoothing filter with a filter half-width of 80 samples in order to further remove incoherent and coherent noise that was not aligned with the dips present within the image (Fehmers and Höcker, 2003; Hale, 2009). While the application of this filter helped to remove much of the noise present within the image and therefore improve the image quality, any smoothing applied to an image will reduce the overall sharpness of the image to a certain degree. While this reduction in sharpness is not ideal for analyzing the focusing of a seismic image, I found that it was not overly detrimental and I was able to find a tradeoff between image smoothness and noise reduction that allowed for an accurate analysis of the focusing of the images. Panel (a) of Figure 4.11 shows a zoom-in on the processed image where the depth range has been limited to 1.25 - 2.5 km. For reference, in panel (b) of Figure 4.11 I have also included the original migrated image obtained without the constant scale factor applied to the migration velocity (I

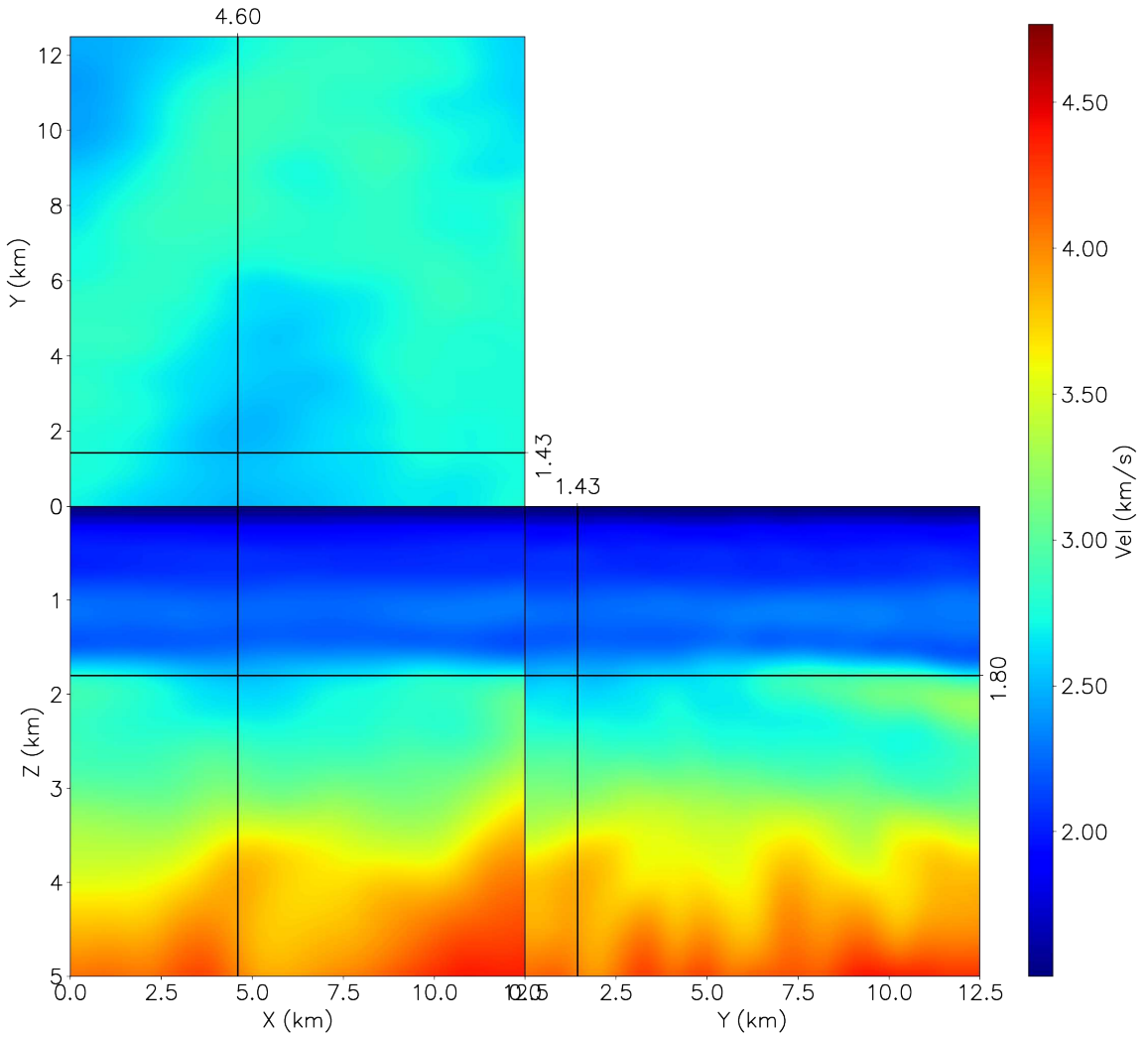


Figure 4.9: The estimated interval velocity windowed to the region of interest. This velocity cube scaled by a value of 0.98 was used for the depth migrations shown in this chapter. [**CR**]

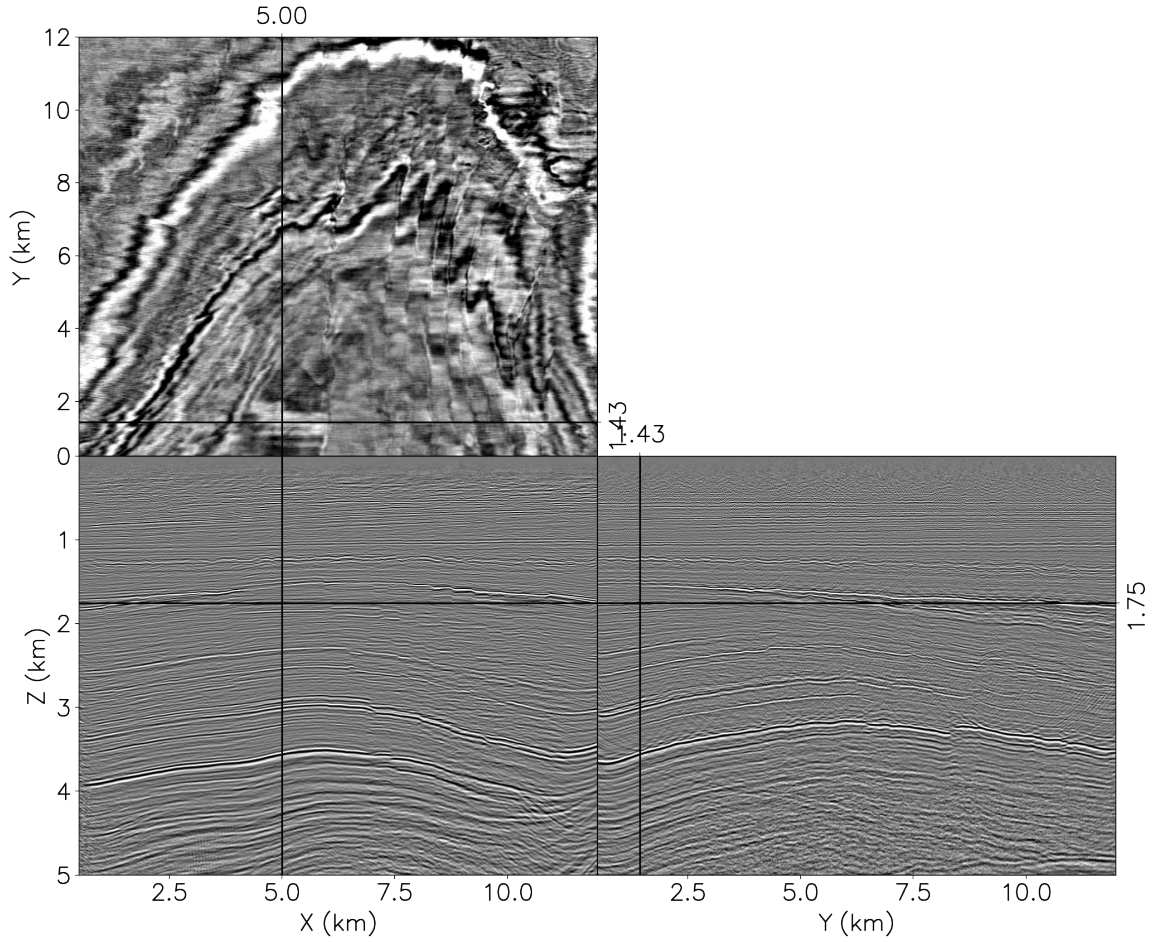


Figure 4.10: The depth migrated image obtained after 3D shot-profile wave equation migration with the scaled estimated depth velocity model. [**CR**]

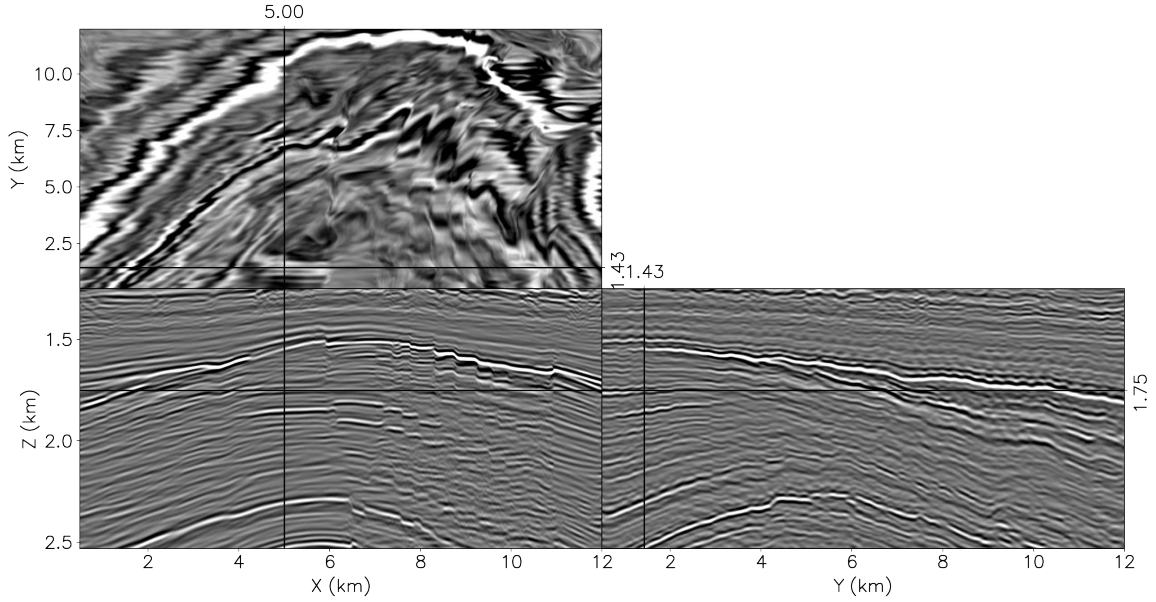
refer to this image as the focused image). Comparing the inline panels of of the 3D displays of these two images, it is clear that many faults exist and that the reflectors appear to be somewhat incoherent in the image obtained from scaling the migration velocity, resulting in a poor image of the faults in this image (I refer to this image as the unfocused image). In addition to examining the coherence of the reflectors and the focusing of the faults to assess the velocity error, I also converted the subsurface-offset domain images to the angle domain (Sava and Fomel, 2003). I converted the 41 subsurface offsets to 64 angles for each image point within the 3D volume. Panel (a)

of Figure 4.12 shows the gathers plotted at every 0.5 km for the $y = 1.43$ km slice of the unfocused image cube and again for reference I have included the angle-domain focused image in panel (b) of Figure 4.12. Comparing the two angle-domain images it is apparent that from depths of 0.75 to 1.75 km the gathers show subtle signs of undermigration for most lateral image points in the unfocused image. Additionally, the horizon at approximately 2.6 km depth also shows signs of undermigration. In spite of these clearly observable undermigrated events, the gathers in both images are also clearly contaminated with a significant amount of noise as is apparent in the gathers laterally positioned from 1 - 4 km as well as from 8 - 12 km. Within the depth range of 2 to approximately 3.5 km, we can observe many grossly overmigrated events that overlap stronger amplitude events. These overmigrated events are clearly some form of coherent noise that have most likely been caused by multiples present within the data. As I will show in the next section of this chapter, this coherent noise that overlaps the signal from the reflections will be problematic for a pure semblance-based focusing analysis.

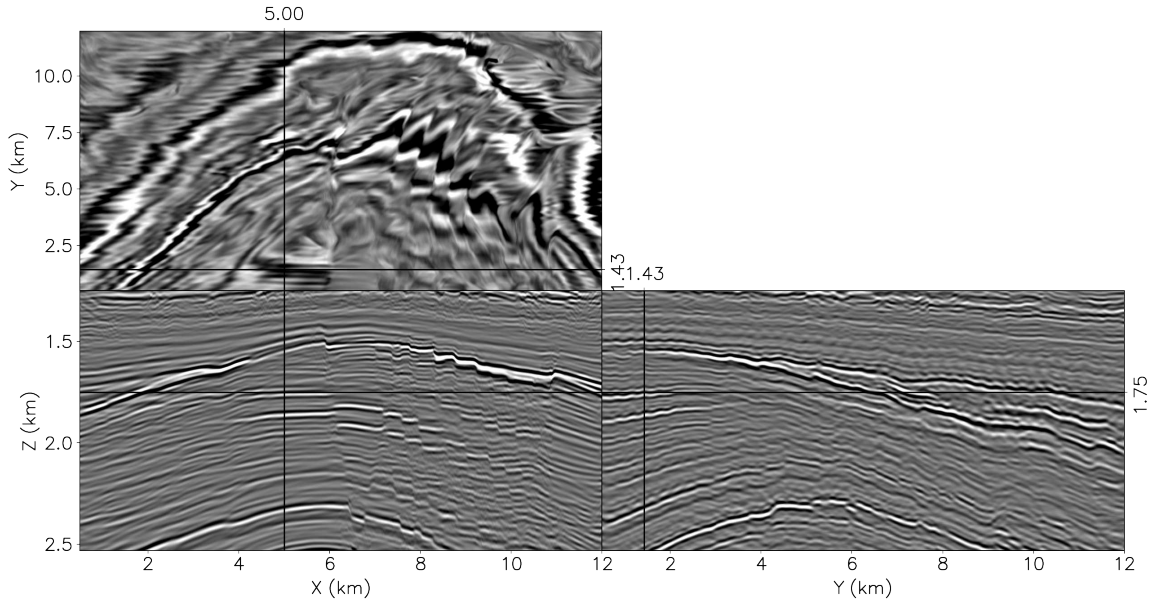
COMMON AZIMUTH RESIDUAL MIGRATION

Using the 3D prestack depth migrated image, I then proceeded to create residually migrated images that can be used for focusing analysis. As I only computed subsurface offsets along the inline direction, I used a common-azimuth Stolt residual migration. As I did in Chapter 2 for 2D and 3D full prestack Stolt residual depth migration, I will derive the expressions for common-azimuth residual migration as presented in Sava (2003) as they are key to the focusing analysis shown in this chapter. To derive the common-azimuth prestack Stolt residual migration, operator I start from the common-azimuth Stolt migration operator which can be written as a cascade of two operators:

$$\begin{aligned} k_{zx} &= \sqrt{\frac{\omega^2}{v^2} - \frac{1}{4}(k_{m_x} + k_{h_x})^2} + \sqrt{\frac{\omega^2}{v^2} - \frac{1}{4}(k_{m_x} - k_{h_x})^2}, \\ k_z &= \sqrt{k_{zx}^2 - k_{m_y}^2}, \end{aligned} \tag{4.1}$$

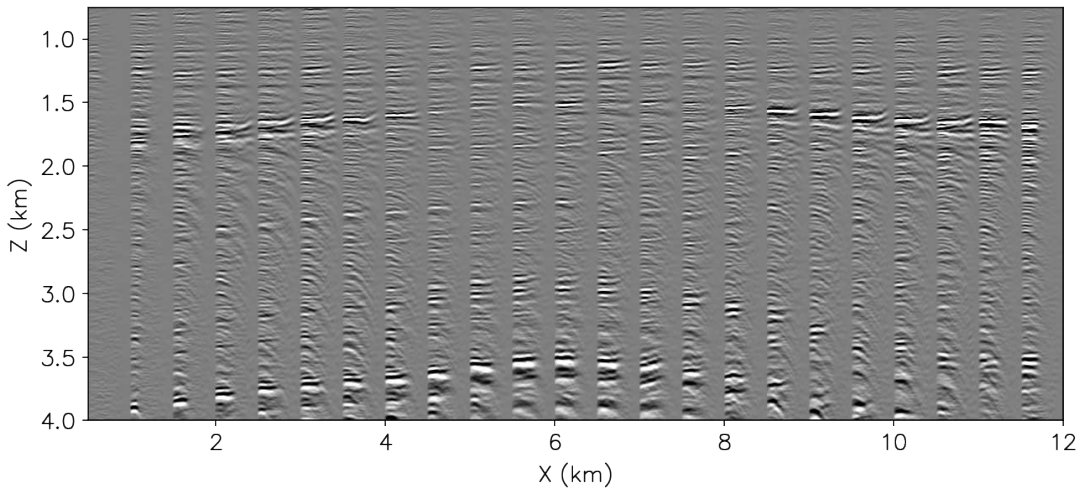


(a)

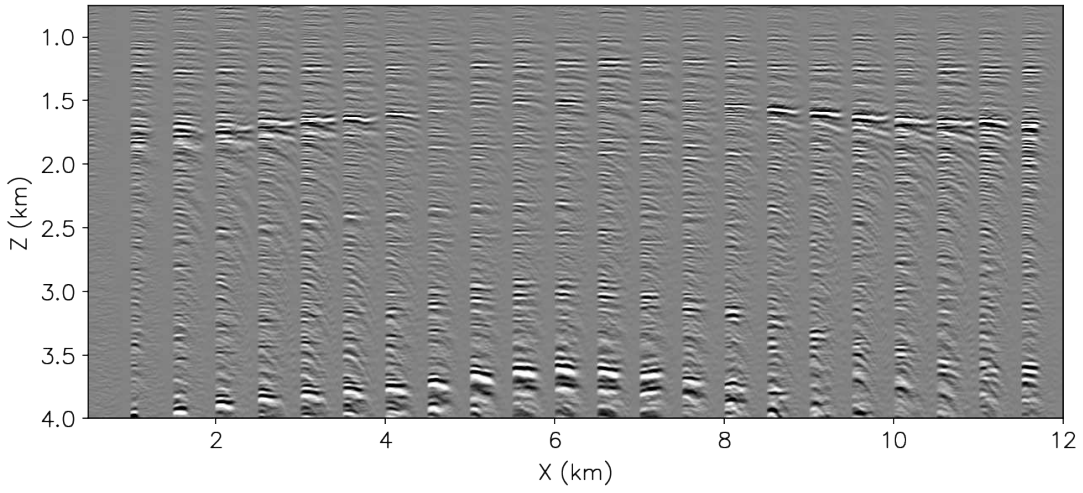


(b)

Figure 4.11: The dip filtered and smoothed depth migrated image windowed to the depth region of interest. Note in the inline panel from 8 - 11 km in the x direction the crossline-oriented faults are unfocused due to the velocity error. [**CR**]



(a)



(b)

Figure 4.12: Angle gathers extracted at 0.5 km intervals plotted for $y = 1.43$ km. At $z = 1.25$ km there is clear moveout on the gathers indicating signs of undermigration. The multiples present within the image appear on the angle gathers as grossly overmigrated events. These are especially apparent from 0 - 4 km and from 8 - 12 km. [**CR**]

where k_{m_x} and k_{m_y} are the inline and crossline components of $\mathbf{k}_m$, and k_{h_x} is the inline offset wavenumber. Biondi and Palacharla (1996) derived this operator by first deriving a common-azimuth DSR operator, using a stationary phase approximation on the full prestack DSR operator. Then, by setting the velocity constant as is done for Stolt migration, they realized they could split the more complicated 3D common azimuth DSR operator into this cascade of two operators. The first operator k_{zx} in Equation 4.1 is the 2D prestack DSR operator which can be easily derived by setting all crossline terms to zero in Equation 2.5. As the second operator contains terms from both k_{zx} and the spatial crossline direction, it can be interpreted as a crossline migration operator. As I did for deriving the full prestack Stolt residual migration operator in Equation 2.11, I can derive the common-azimuth Stolt residual depth migration operator by first deriving an expression from Equation 4.1 for ω in terms of k_{z_0} and v_0 and then substitute that expression into Equation 4.1. Again, after some algebra I obtain the following expression for ω^2:

$$\omega^2 = v^2 \frac{\left[k_z^2 + k_{m_y}^2 + k_{h_x}^2\right]\left[k_z^2 + k_{m_y}^2 + k_{m_x}^2\right]}{4(k_z^2 + k_{m_y}^2)}. \tag{4.2}$$

Now expressing ω^2 in terms of k_{z_0} and v_0, and substituting it into k_{z_x} in Equation 4.1, I obtain the following common-azimuth residual Stolt operator:

$$\begin{aligned} k_{z_x} &= \frac{1}{2}\sqrt{\frac{v_0^2}{v^2}\frac{\left[k_{z_0}^2 + k_{m_y}^2 + k_{h_x}^2\right]\left[k_{z_0}^2 + k_{m_y}^2 + k_{m_x}^2\right]}{k_z^2 + k_{m_y}^2} - (k_{m_x} + k_{h_x})^2} \\ &\quad + \frac{1}{2}\sqrt{\frac{v_0^2}{v^2}\frac{\left[k_z^2 + k_{m_y}^2 + k_{h_x}^2\right]\left[k_{z_0}^2 + k_{m_y}^2 + k_{m_x}^2\right]}{k_z^2 + k_{m_y}^2} - (k_{m_x} - k_{h_x})^2}, \\ k_z &= \sqrt{k_{zx}^2 - k_{m_y}^2}. \end{aligned} \tag{4.3}$$

Using this operator, I then could create 3D prestack residually migrated images. For the residual migration, I chose to create residual migration images for ρ ranging from 0.95 to 1.05 with an interval of 0.00125. This created a total of 81 residually migrated images for focusing analysis. Figures 4.13 and 4.14 show a selection of residually

migrated images plotted at an interval of $d\rho = 0.025$ for an inline slice extracted at $y = 1.25$ km and a depth slice at $z = 1.75$ km. Comparing the different residual migration images in Figure 4.13, it is apparent in Figure 4.13(b) that the crossline faults are best focused within the image corresponding to ρ value of 0.975. Consistent with the constant velocity error introduced in the image of 0.98, I found that the crossline faults were best focused for ρ values between 0.975 and 0.98. Additionally, in the depth slices shown in Figure 4.14, we can observe that the faults are in general sharper for ρ values 0.95 and 0.975 shown in Figures 4.14(a) and 4.14(b). This is especially apparent for the portions of the fault between 4 - 8 km in the crossline direction.

SEMBLANCE-BASED FOCUSING ANALYSIS

With the 81 3D prestack residual migration images, I then performed a semblance-based focusing analysis. As I only computed the extended images along the h_x direction, I could compute ρ semblance for each image point in 3D in the same manner as I did for the 2D images in the previous chapters. For each image point within the 3D volume, I first extracted the angle gather for all depths. I then computed semblance over the residual migration images as I described in Equation 2.13 in Chapter 2 and as I did for the two 2D examples in Chapter 3. Finally, using the same automatic picking approach that I used in Chapter 3, I picked the maxima of the semblance scans for each image point. The left panel of Figure of 4.15 shows the residually migrated angle gather and the right panel shows the computed semblance for an angle gather extracted at $x = 7.5$km and $y = 1.43$ km. The semblance panel shows strong coherent events down to approximately 1.5 km and also below 2.5 km in depth. Between these depths however, there appears to be relatively little coherent energy in the angle gathers. In addition to the residually migrated gather and the semblance panel, Figure 4.15 also shows the smoothed pick that resulted from the application of the automatic picker. While the pick is consistent with the coherent energy in the semblance panel, the automatic picker appears to have picked ρ values between 0.981 and 1.0 which is slightly higher than what I expected for the focusing of the faults in

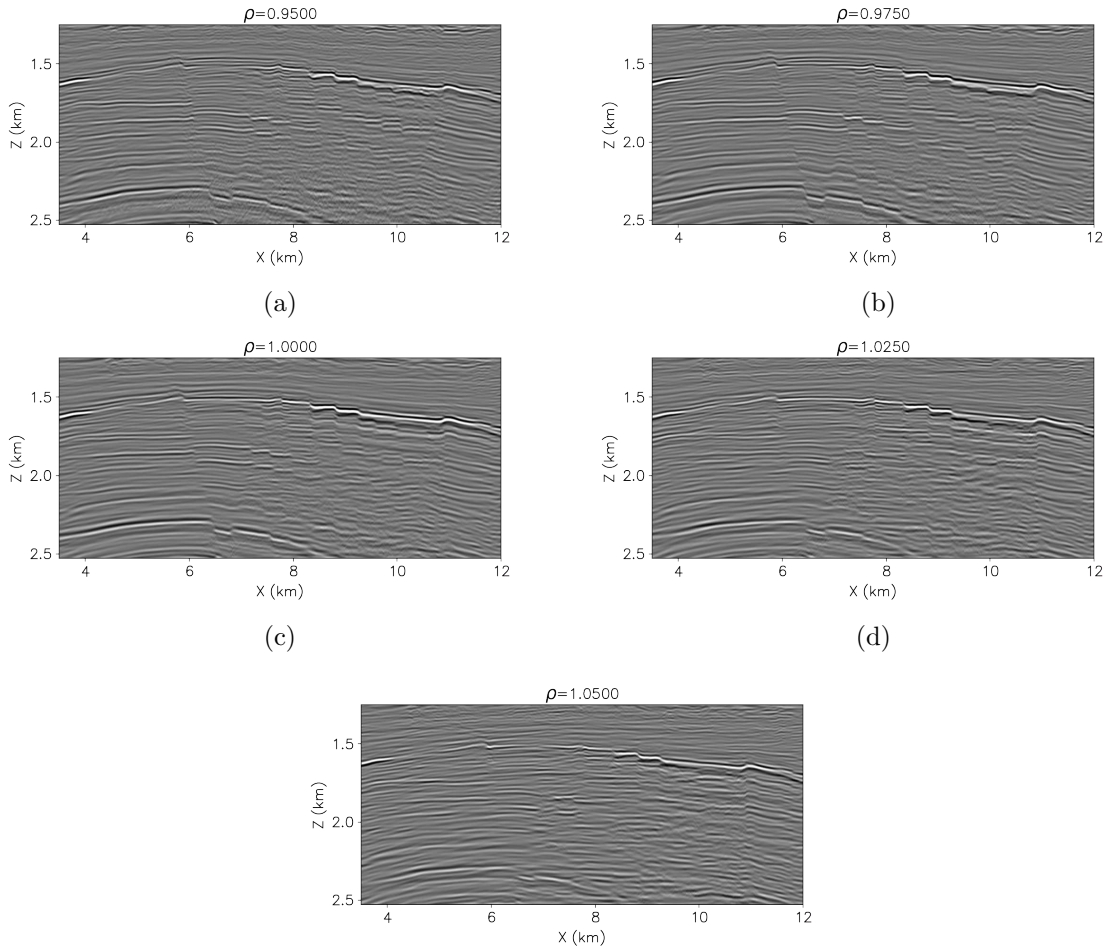
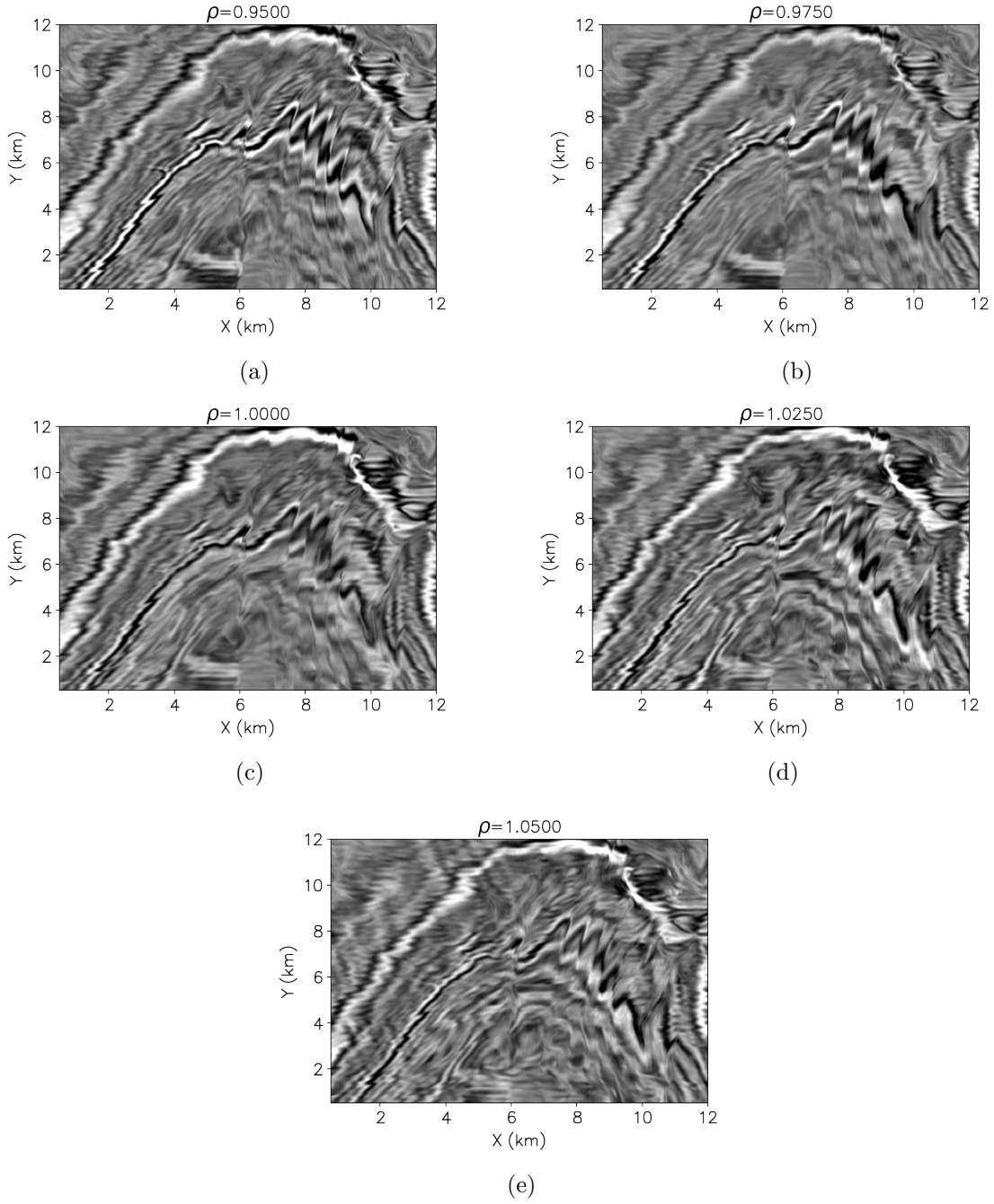


Figure 4.13: A selection of residual migration images taken from the application of 3D prestack common azimuth Stolt residual migration to the migrated volume plotted at an interval of $d\rho = 0.025$. The inline shown was extracted at $y = 1.25$ km. The reflectors from 6 - 10 km appear most coherent and the faults best focused for ρ values between 0.975 and 0.980 as is evident in the $\rho = 0.975$ image. [**CR**]



(a) (b) (c) (d) (e)

Figure 4.14: A selection of residual migration images taken from the application of 3D prestack common azimuth Stolt residual migration to the migrated volume. The depth slice shown was extracted at $z = 1.75$ km. While it is more difficult to assess the quality of the focusing of the faults in this view than the inline view, the faults shown within the $\rho = 0.95$ and $\rho = 0.975$ images appear to be sharpest. This is especially apparent for the portions of the faults between 4 - 8 km in the crossline direction. [**CR**]

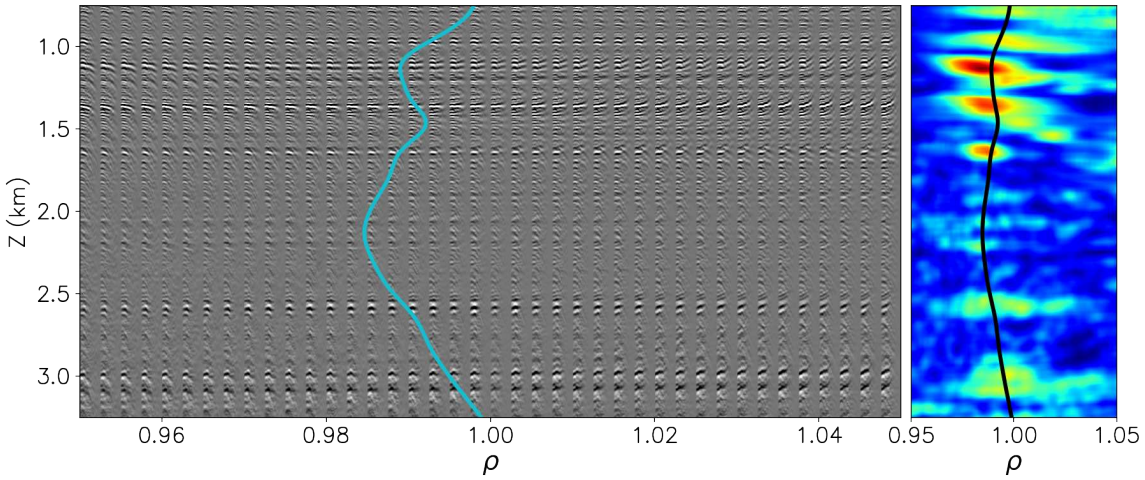


Figure 4.15: Computed ρ semblance from an angle gather extracted at $x = 7.5$ km and $y = 1.43$ km. The left panel shows the angle gather residually migrated for ρ values between 0.95 and 1.05 and the right panel shows the computed semblance from each residual migration. The black and cyan curves show the smoothed $\rho(z)$ pick resulting from the automatic picker. [**CR**]

that region as is shown in Figure 4.13(b).

Repeating this semblance computation and automatic picking procedure for each image point resulted in $\rho(z)$ picks for all points in space. To create the $\rho(z, x, y)$ volume, collected all picks and smoothed them in 3D with a smooth-by-division procedure (Fomel, 2007). I used a 3D triangular smoother as the smoothing operator, where the smoother was 0.2 km in depth and 0.5 km in both the inline and crossline directions. Figure 4.16 shows the resulting estimate of $\rho_{\text{SMB}}(z, x, y)$ superimposed on the unfocused image, after the $\rho(z)$ for all image points are smoothed. On average, from the three slices shown, it is clear that the estimated $\rho(z, x, y)$ is below $\rho = 1.0$, which is consistent with the focusing of the images shown in Figures 4.13 and 4.14. However, the values are larger than $\rho = 0.975$ which, as shown in Figures 4.13 and 4.14, will not provide the best focusing of the faults. Additionally, ρ values larger than 1.0 are also readily apparent in the image. This is due to the poor illumination present within these areas, which is especially prevalent at the edges.

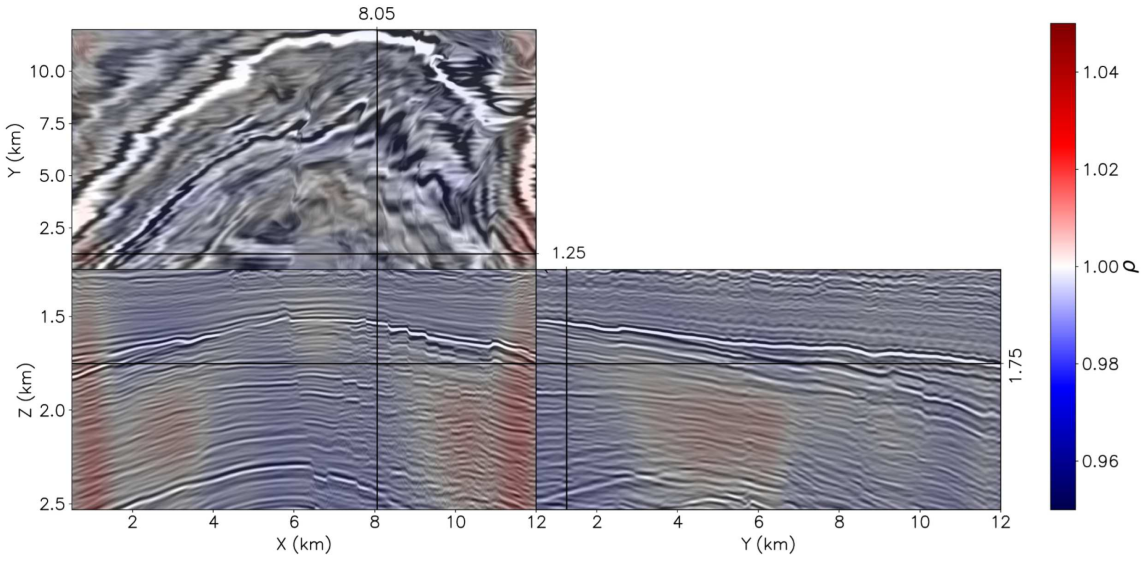


Figure 4.16: The estimated $\rho_{\text{SMB}}(z, x, y)$ superimposed on the unfocused F3 image. On average the estimated $\rho_{\text{SMB}}(z, x, y)$ is higher than $\rho = 0.9775$, which as Figures 4.13 and 4.14, will not provide the best focusing of the crossline-oriented faults within the image. The ρ values estimated above 1.0 are largely due to poor illumination. [**CR**]

3D CNN-BASED FOCUSING ANALYSIS

The application of CNN-based focusing analysis to 3D prestack images follows the same workflow as was outlined in Chapter 2 and applied to 2D images in Chapter 3. However, some significant changes needed to be made in order to extend the method to 3D prestack images. The first of these was that the CNN itself needed to be extended to take in 4D prestack image patches. This required implementing custom convolutional operators since 4D images are non-standard in computer vision problems and therefore do not come readily available in deep learning frameworks. Additionally, this introduced significantly more parameters into the CNN and required a modification of the architecture in order to reduce the total number of trainable parameters. The other needed change results from the enormous amount of data generated by 3D prestack common azimuth residual migration. This large size of the dataset allows for the creation of a large field training set that can help to remove

the data-domain gap that occurs when training only on synthetics. However, this does create the difficult task of accurately labeling a large number of patches so that they may be used for training. In the following sections, I will first describe the creation of both synthetic pre-training and field training datasets used to train the CNN. I then describe the extension of the fault-focusing CNN from 2D to 3D prestack images using 4D cross-correlations. Finally, I show the results of training the 3D fault-focusing CNN and the application of the CNN to the unfocused prestack image.

Training data creation

As I described in Chapter 3, while ideally the CNN could be trained entirely from pre-stack patches extracted from real seismic images, this dataset was not available within SEP during the time this research was conducted. Therefore, I constructed a dataset from focused and unfocused synthetic seismic image patches as well as focused and unfocused patches from the focusing analysis that I performed on the F3 dataset and described above in this chapter.

Similar to what I had done for the 2D training creation described in Chapter 3, the procedure for creating 3D synthetic focused and unfocused images also makes use of the synthetic model building method outlined in Chapter 2. To make synthetic models, I took the following steps: I first constructed a layered velocity model in which the velocity varied linearly between 1.5 and 5 km/s. Then, once the layered model was 90% deposited in depth, I folded the layered model to introduce undulation in the layering that matched the dips observed in the inline direction observed in the 3D F3 image. After the folding, I continued to deposit flat layers until I reached the desired depth of 5 km with the last layer being a 1.5 km/s water layer. Finally, I introduced 3D faulting into the model. Within the depths of approximately 1.5 - 2.25 km, the faulting within the F3 dataset occurs primarily in the crossline direction, with small faults that occasionally intersect these large faults. Beneath 2.25 km, the faulting changes and is largely dominated by faults intersecting an anticlinal structure

at an azimuth of 55° measured counterclockwise from the y-axis. As the crossline-oriented faults are better imaged within my 3D volume, I decided to create a faulting structure that resembled those faults. Therefore, for each 3D synthetic image, I introduced undulating crossline-oriented faults with an azimuth of 180° and dips and throws that resembled those observed in the F3 inline images.

Using the synthetic velocity model created, I could then create focused and unfocused synthetic images. First, I computed the vertical derivative of the velocity, thereby providing a reflectivity model that I subsequently convolved with a 20 Hz Ricker wavelet. Then, I padded the 3D image with 20 negative and 20 positive inline subsurface offsets (h_x) and converted these offsets to angles, thereby obtaining a 3D prestack focused image. For the unfocused image, after creating the extended subsurface offset image, I performed a 3D common-azimuth Stolt residual migration. This then created diffractions around faults and moved energy away from the zero subsurface offset. Lastly, I converted the subsurface offsets to angles, thereby providing an unfocused 3D angle-domain image. While I could provide only the stacked image to the CNN for focusing analysis, providing all axes of the image cube is beneficial especially in areas where geologic heterogeneity is large and has potentially not been seen by the CNN. Therefore, I needed to create 4D patches from the focused and unfocused synthetic volumes. To do this, I used the same patch extraction procedure as I described in Chapters 2 and 3, but applied a 3D patching to each image along the extended axis. After patching the image within the faulted region with a spatial patch size of $(64 \times 64 \times 64)$ with a stride of 32 samples in each spatial direction and keeping all available angles (32 in total), I created approximately 850 patches per 3D image. As all of these patches resulted from the same model I found them to be highly spatially correlated and therefore I would keep only 10% of the patches from each model, which resulted in 85 patches per model. I then repeated this for over 40 models to create 4,608 4D patches for my training set. Figure 4.17 shows examples of unfocused and focused 4D patches used for training and validation of the 3D fault-focusing CNN.

In addition to creating a training dataset composed of synthetic unfocused and

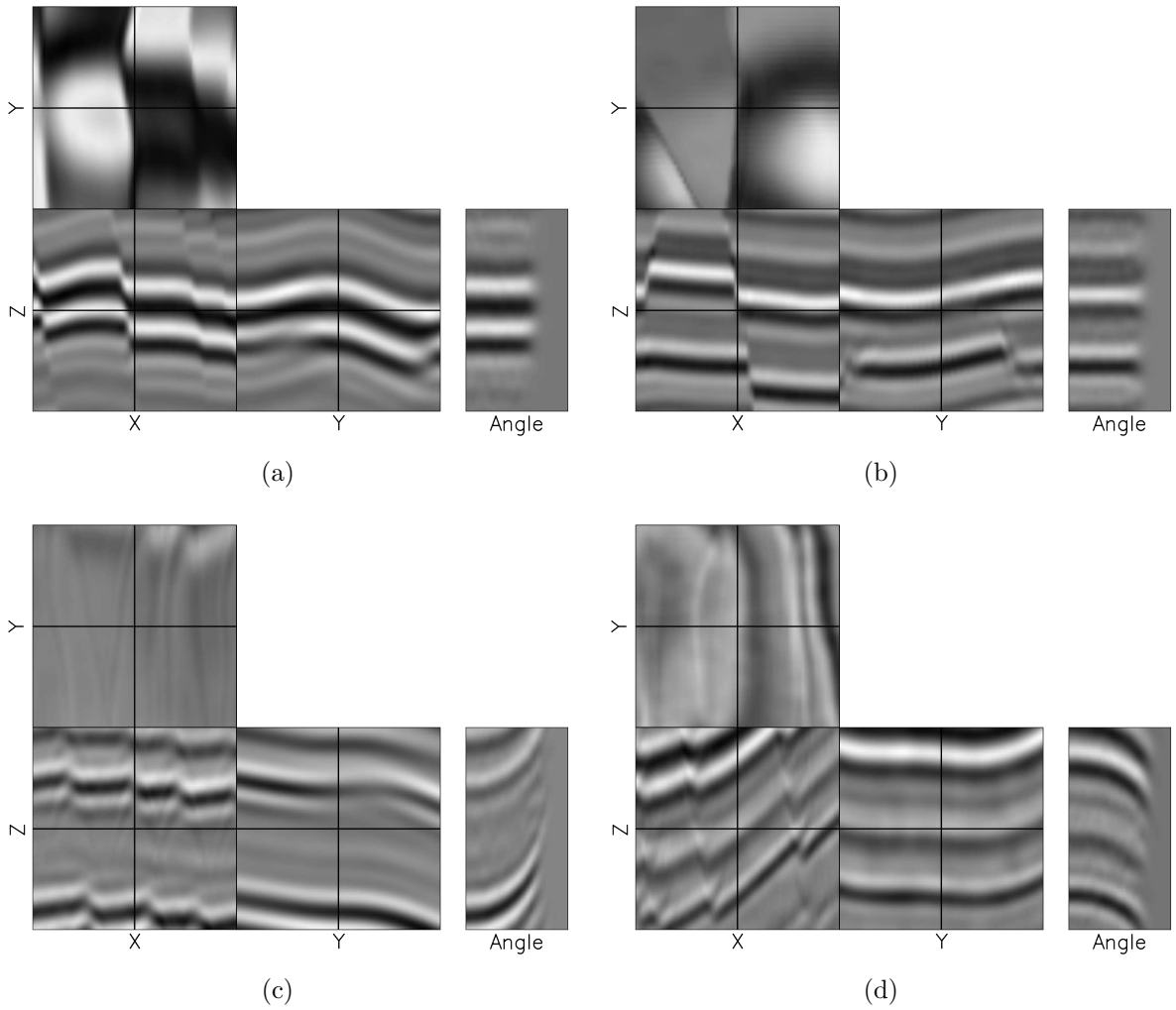


Figure 4.17: Synthetic 3D prestack focused and unfocused training patches used for training and validation of the 3D prestack fault-focusing CNN. The position of the cross hairs in the depth slices indicate the location from which the angle gather has been extracted. Panels (a) and (b) show focused training patches and (c) and (d) show unfocused training patches. [**CR**]

focused 3D prestack image patches, I performed a manual focusing analysis on the stacked residual migration images. I first inspected the volume primarily in the inline direction and found that for inlines between 0.5 - 2 km, the faults appeared to be best focused for faults for ρ values near 0.975. I continued to visually track the horizons from inline to inline and found that as expected with the introduction of a constant velocity error, the faults along the crossline direction appeared to be best focused for similar values of ρ. As my manual interpretative analysis was highly qualitative, I also performed a 3D fault segmentation on each residually migrated stack. The fault segmentation was performed with a 3D CNN that was designed with a U-Net architecture. The training images consisted of 3D synthetic reflectivity models that were created in the same way as the 3D synthetic models used for the fault-focusing training images. For the labels, I assigned a value of one to the pixels along the fault trajectory and zero away from the trajectory. I trained for 25 epochs and the CNN achieved 98% accuracy on the validation patches. Figure 4.18 shows the segmented faults on the residually migrated images for the same inline slice shown in Figure 4.13. Consistent with my qualitative interpretation, the faults in the images corresponding to ρ values between 0.975 and 0.98 (as is apparent in the $\rho = 0.975$ in Figure 4.18(b)) have the best segmentation. When comparing the faults in Figure 4.18(b) with the remaining segmentations in Figure 4.18, we can observe that the predicted fault positions are the most coherent and are consistent with the fault positions observed in Figure 4.13(b). Examining the segmentation of the faults across all inlines as is shown in the depth slices in Figure 4.19 also helps to assess the change in focusing of the faults with a change in ρ. Comparing the segmented faults from the different residually migrated images shown in Figure 4.19, it is clear that for large ρ values the predicted fault confidences are more spread out across the faults as the faults become increasingly undermigrated. Figures 4.19(a) and 4.19(b) have the sharpest predictions, which are also most consistent with the faults apparent in the images shown in Figure 4.14.

With my fault confidence-guided interpretative analysis of the focusing of the faults within the image, I then created training patches from the residually migrated prestack images of the F3 data. To do this, I first windowed the migrated cube within

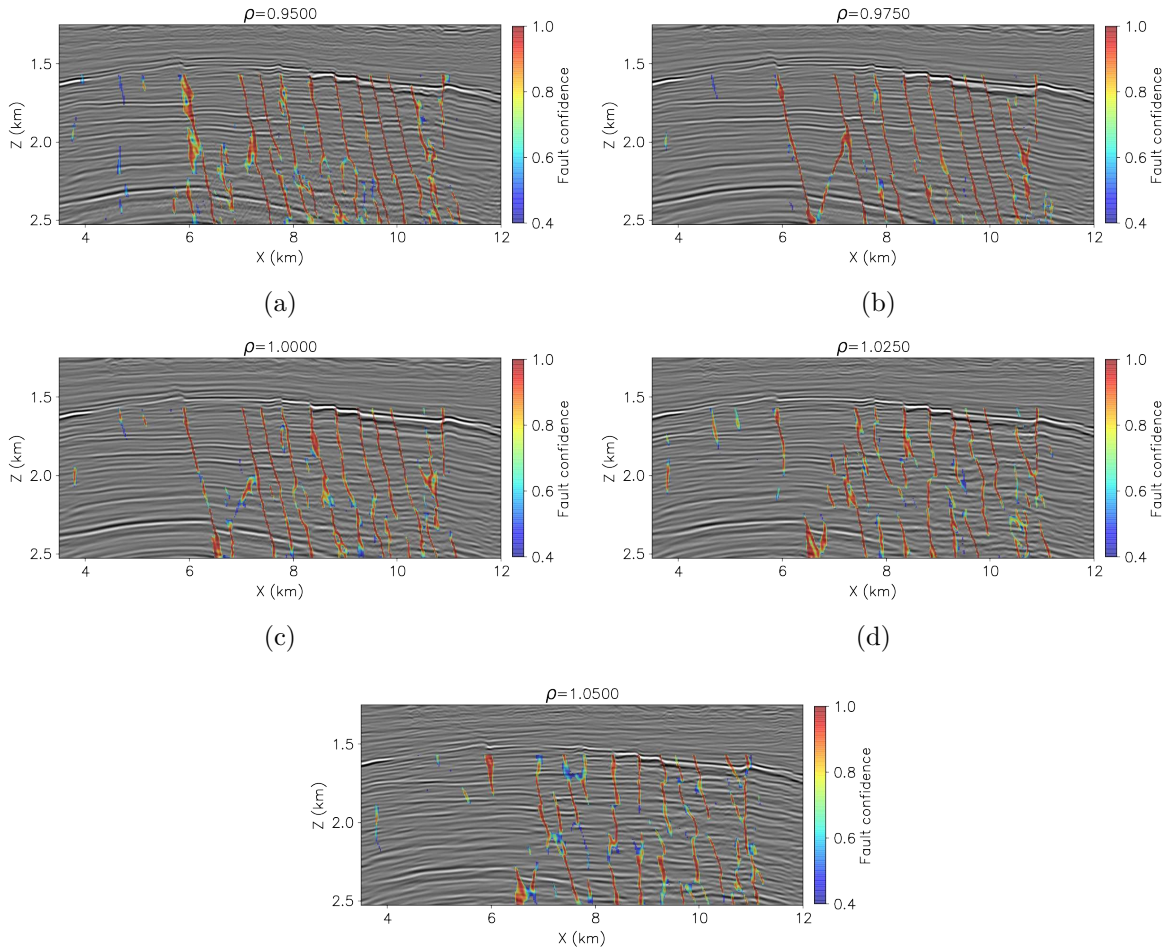


Figure 4.18: 3D fault segmentation of the residually migrated images shown for the inline slice in Figure 4.13. Consistent with my qualitative observation of the faults, the predicted fault confidences are most continuous and consistent with the fault positions for the $\rho = 0.9750$ image. [**CR**]

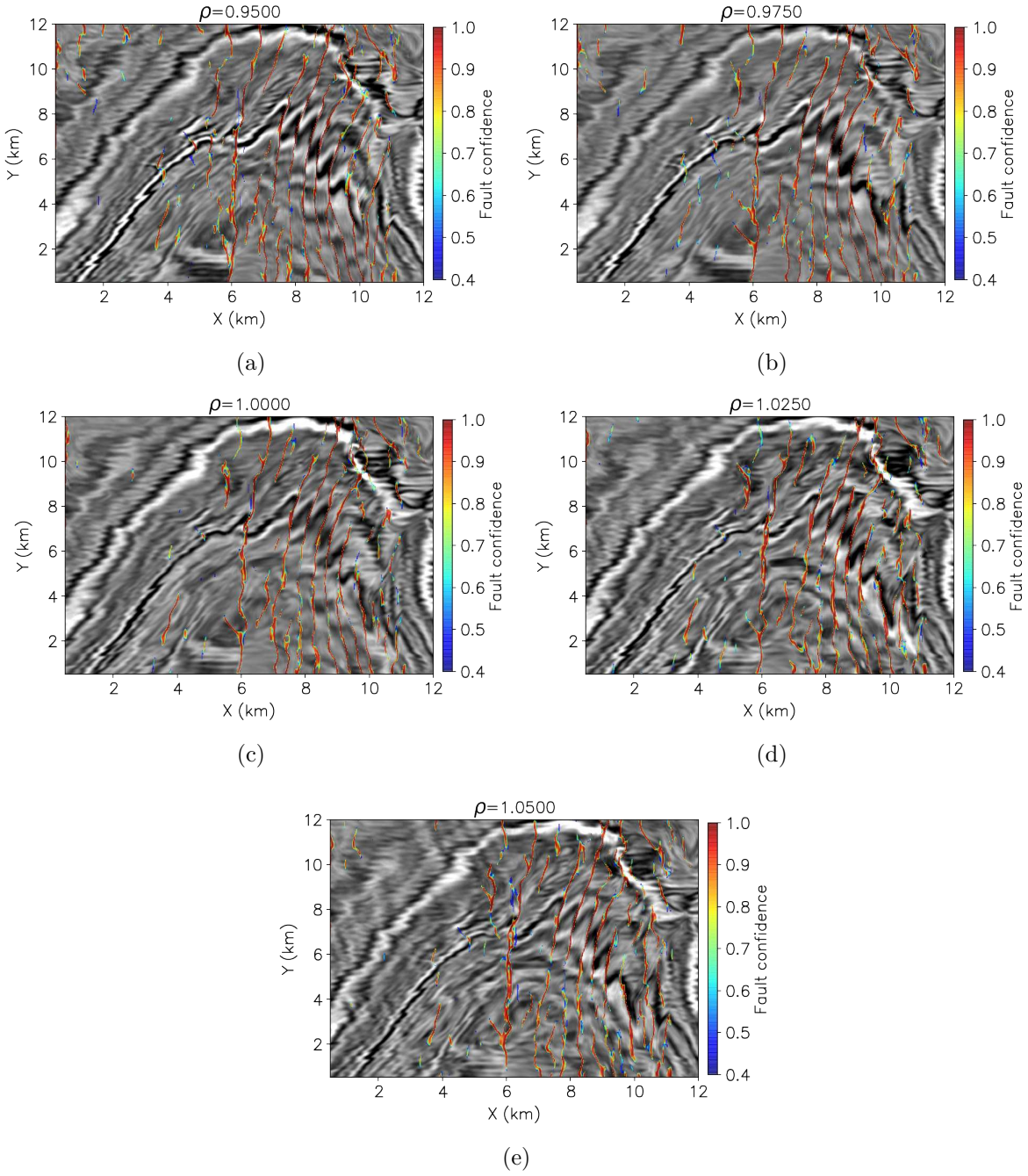


Figure 4.19: 3D fault segmentation of the residually migrated images shown for the depth slice in Figure 4.14. The images corresponding to ρ values of 0.950 and 0.9750 have the most continuous and sharpest predicted fault confidences for the faults between 4-8 km in the crossline direction, which is consistent with my qualitative interpretation of the residually migrated depth slices shown in Figure 4.14. [**CR**]

a depth range of 1 - 3.6 km and used all inlines and crosslines within this depth range. Then, using the same patching parameters as I used for the synthetic training images, I formed patches of all 81 3D prestack residually migrated cubes. This resulted in a total of 3,375 $\times$ 81 = 528,525 patches where each patch had the same dimensions of the synthetic training patches (64 samples in each spatial dimension and 32 angles). To label the patches as focused and unfocused, I first compared the images between values of $\rho = 0.975$ and $\rho = 0.98$. As the images within this range did not dramatically vary in focusing and I observed consistent focusing for the $\rho = 0.975$ image I labeled this image as the focused patch within the spatial patch group. While the $\rho = 0.98$ in theory should correct for the velocity error present within the image, it is important to keep in mind that ρ is an RMS measure of velocity error rather than an interval measure and cannot be used directly to assess the velocity errors within the image. Additionally, the original migration velocity was not exact (as is apparent in the angle-domain image shown in Figure 4.12(b)) and therefore, the best focusing may occur for ρ values different than 0.98. With the focused image selected for each spatial patch group, I chose the unfocused patch by randomly selecting a patch at least 0.02 ρ values away from $\rho = 0.975$. Performing this for all spatial patches resulted in a total of 3,375 focused and 3,375 unfocused patches. Figure 4.20 shows four examples of focused and unfocused patches extracted from the residual migration images.

Architecture of the 3D prestack fault-focusing CNN

With both synthetic and field training sets prepared, I then implemented a 3D extension of the 2D fault-focusing CNN presented in Chapter 2. Recall that for 2D prestack images, the CNN employed 3D cross-correlations over the two spatial axes as well as over the angle axis. Naturally, this then required that 4D cross-correlations be used for the CNN to take a 3D prestack image as input. While 1D, 2D and 3D cross-correlations are provided with deep learning frameworks such as TensorFlow and PyTorch (Abadi et al., 2015; Paszke et al., 2019), 4D convolutions are non-standard in computer vision problems and therefore are not included as part of these libraries.

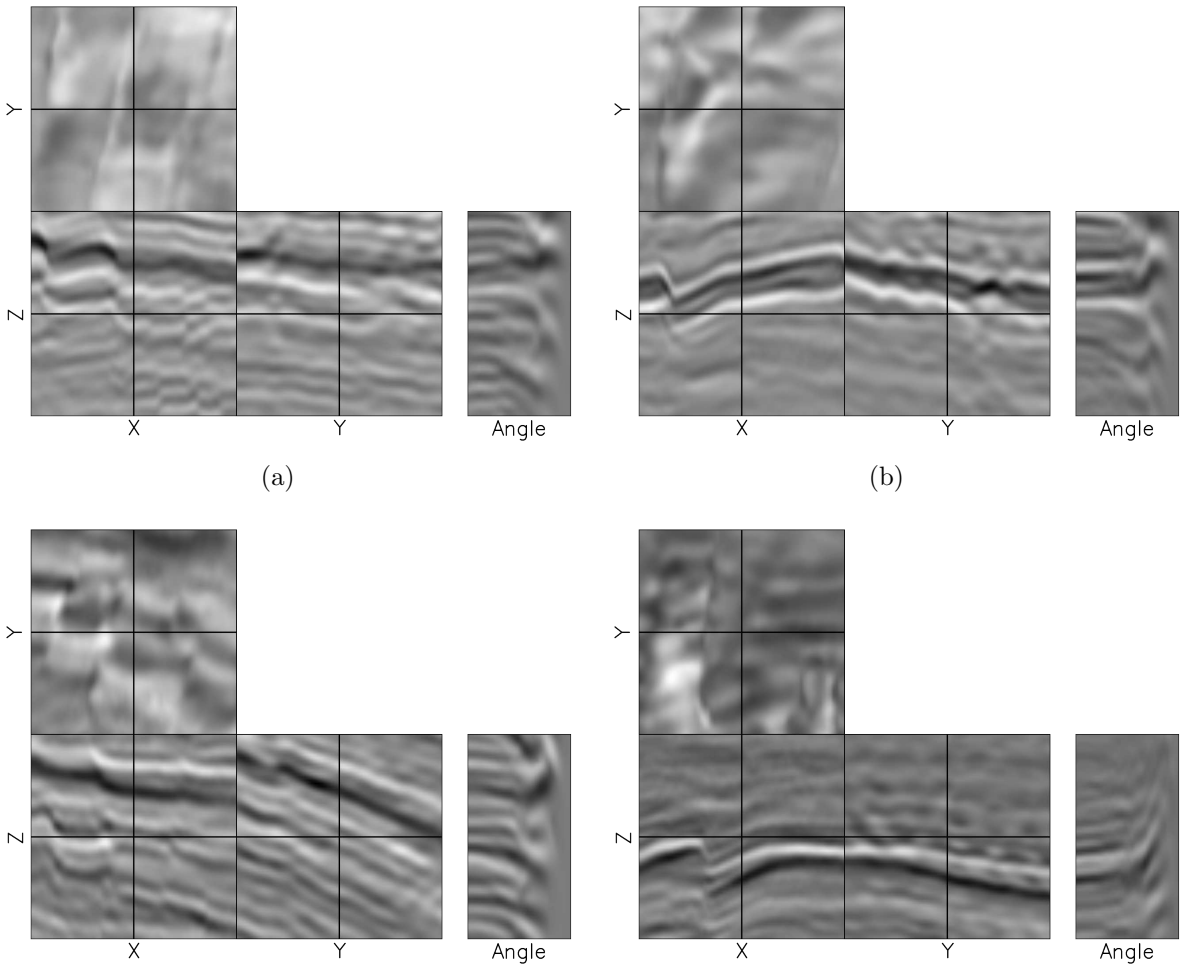


Figure 4.20: 3D prestack focused and unfocused training patches extracted from the residual migration images of the F3 data. The position of the cross hairs indicate the location from where the angle gather has been extracted. Panels (a) and (b) show focused training patches and (c) and (d) show unfocused training patches. [**CR**]

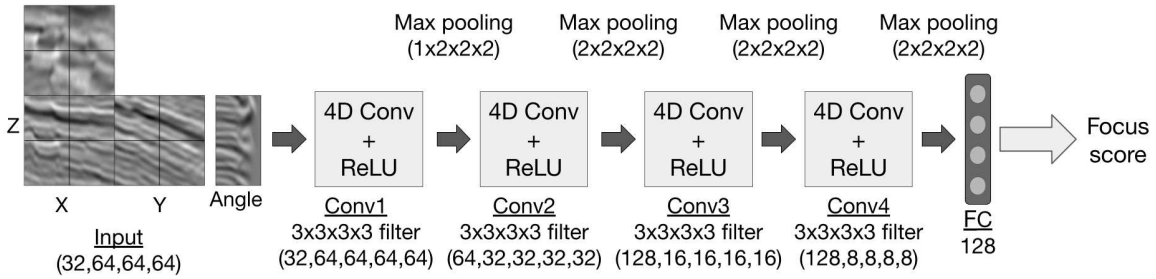


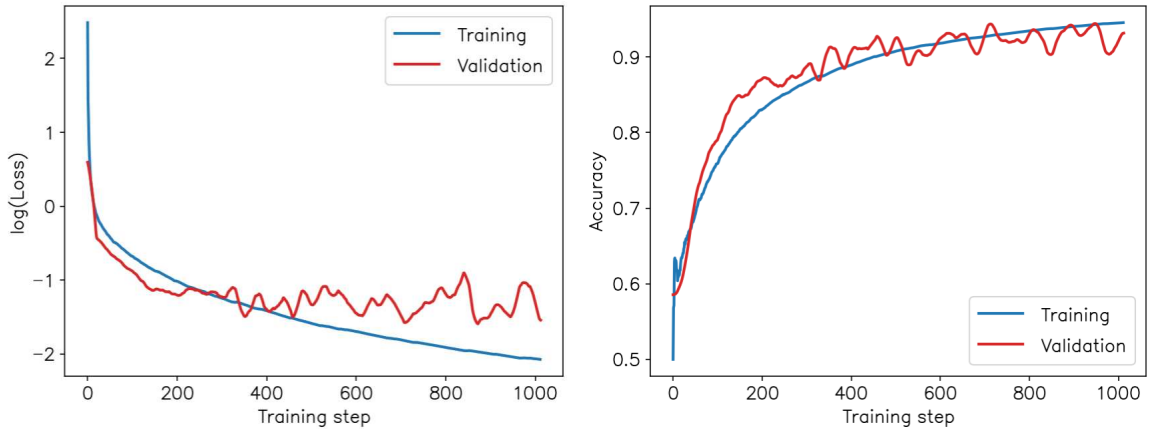
Figure 4.21: Schematic of the 3D fault-focusing CNN. The input to the CNN is a prestack image patch with 64 samples in each spatial direction and 32 angles. The patch passes through four 4D convolutional layers with ReLU activation functions and in between each layer I apply a pooling layer. The feature maps are then flattened and are provided to a fully-connected layer which outputs a feature vector of length 128. Finally, this feature vector is used as the input to the output classification layer. [**NR**]

This then required my own implementation of a forward and backward 4D cross-correlation operator within the PyTorch framework. While the most straightforward way to implement this would be via a 4D cross-correlation kernel, I instead chose to split the 4D kernel into a summation of 3D kernels (see Appendix D for more details). This allowed for the use of purely 3D cross-correlation operators for which there exist native PyTorch implementations. Figure 4.21 shows the 3D prestack image focusing CNN with 4D convolutional layers. Note that apart from the 4D convolutional layers as well as the introduction of an additional convolutional layer, this network closely resembles the 2D network shown in Figure 2.10. As with the 2D network, I wanted to extract all information available within the prestack image patch, while keeping the network relatively small in order to reduce the total number of trainable parameters. The figure shows that the CNN takes as input a 3D prestack image patch and that the patch is processed by four 4D convolutional layers with a ReLU output activation function. I used four convolutional layers as opposed to three (used for 2D images) in order to reduce the significant increase in the number of parameters that are created due to the introduction of the additional axis. After the output of the first convolutional layer, the feature maps are passed into a 3D spatial max pooling layer and after subsequent layers a 4D max pooling layer is used. (As PyTorch also does not

have 4D max pooling layers, this required a custom implementation but is rather straightforward as the max pooling operation is separable.) Finally, the feature maps are reshaped and provided to a fully-connected layer, which outputs a feature vector of 128 elements. As was done for 2D prestack images, the output feature vector is provided to a final classification layer that computes the focusing score. With this design, the feature extraction component consisted of 6,353,792 parameters. Including the 129 parameters from the classification layer, the CNN contained a total of 6,353,921 trainable parameters.

Training the fault-focusing CNN

Similar to the training for the 2D fault-focusing CNN, I also split the training of the 3D fault-focusing CNN into two stages. The first stage was a pre-training stage in which I trained the CNN using only the synthetic data. Of the 4,608 synthetic training samples I used 4,096 for training and 512 for validation. I used a batch size of 4 samples and a fixed learning rate of 5×10^{-5}. After training for 1,024 training steps (one epoch), the CNN achieved 99% accuracy on the validation data. Now, with the pre-trained CNN as the initial weights, I trained the CNN on the patches extracted from the F3 residual migration images. With the same learning rate and batch size as I used for the pre-training stage, I trained the CNN for a total of 10 epochs on 2,024 focused and 2,024 unfocused patches taken from spatial patch groups located inlines between $y = 5.6 - 12$ km of the image. As compared to the 2D image described in Chapter 3, I decided to use more field data patches when training as I wanted to take advantage of the fact that more samples were generated after the 3D common-azimuth residual migration. However these 4,048 samples used for training constituted only 0.8% of the total number of patches generated during residual migration and that the CNN would be used for automating focusing analysis. After training for a total of 10 epochs, the CNN was able to achieve a maximum of 94.4% accuracy on the validation set which consisted of 450 patches that were selected randomly between $y = 0 - 5.6$ km. Figure 4.22 shows the learning curves computed from both the training and validation sets and Table 4.1 provides a complete list of metrics computed on the



(a) (b)

Figure 4.22: Learning curves for CNN training of field training set plotted for every 10th training step. (a) The logarithm of the training and validation loss and (b) the training and validation accuracy. Note that the validation loss and accuracies were averaged over 20 samples to better display the overall trend during training. [**CR**]

validation data.

Focusing analysis with the trained CNN

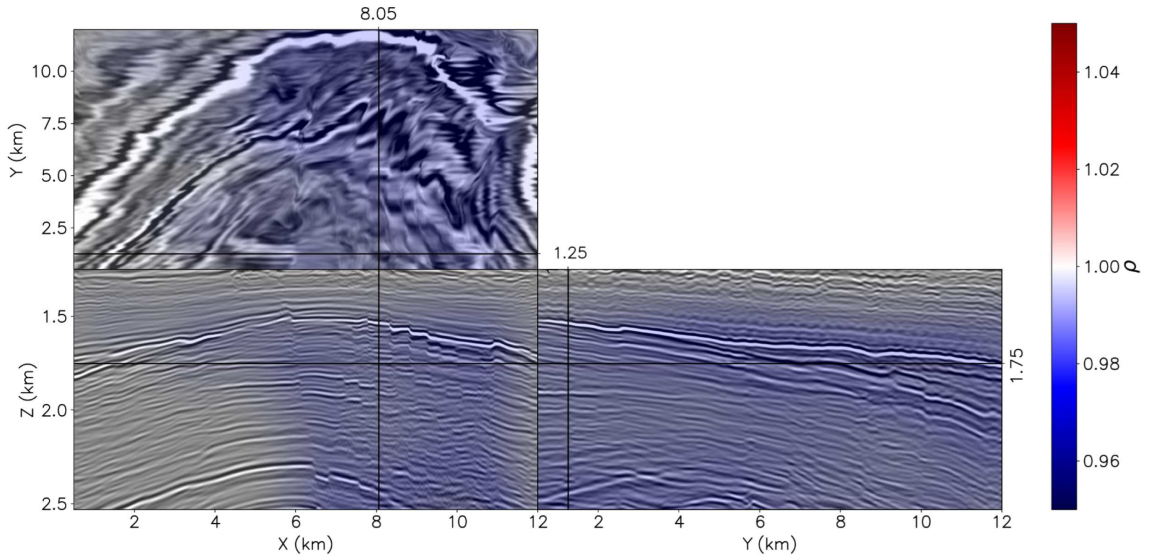
With the trained CNN parameters, I then performed focusing analysis for all residually migrated images. Using the predicted fault confidence cube, I first identified all prestack image patches that contained detected faults. Then, for these residually migrated spatial patch groups, I used the CNN to obtain the focusing score for each prestack image patch. With the predicted focusing scores for a residually migrated patch group, I then selected the ρ that provided the highest focusing score among these patches. There were approximately 1,575 patches that contained detected faults and therefore the CNN performed focusing analysis on 81 ρ values $\times$ 1,575 = 127,575 patches. For all other patch groups, I assigned a value of $\rho = 1.0$. It is important to note that the spatial patch groups between $y = 5.6-12$ km have been used for training and therefore the network has been provided a label for those spatial patch groups. However, as the CNN is not used directly as a classifier and rather the maximum

Accuracy	**Precision**	**Recall**	**F1**
0.944	0.903	0.995	0.947

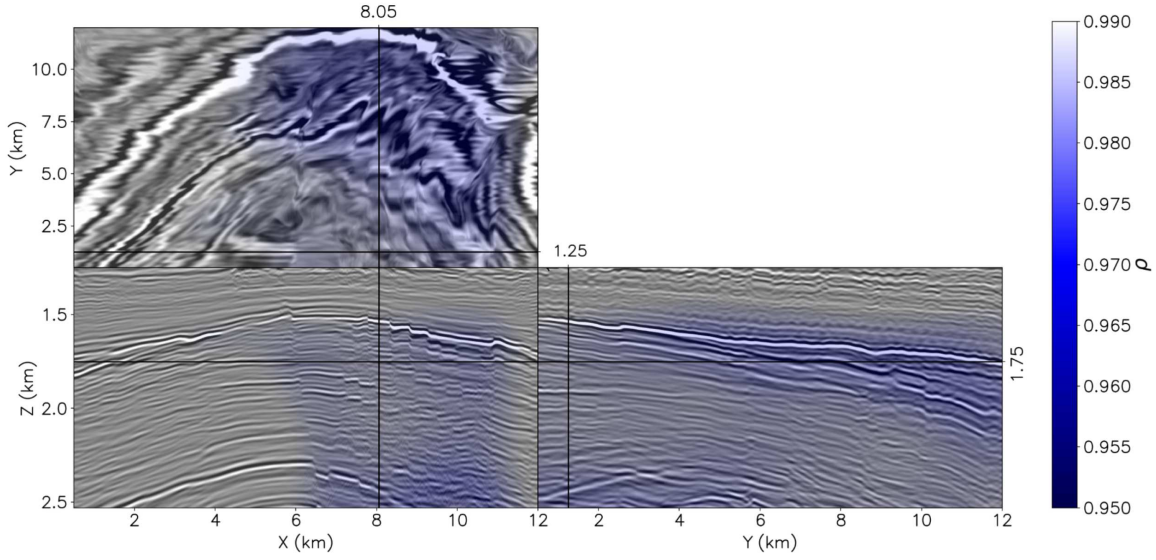
Table 4.1: Metrics computed on the F3 validation data with the trained fault-focusing CNN. The validation patches were taken between $y = 0 - 5.6$ km in the image.

focusing score computed across all patches is used to select the best focused patch, then as long as the training loss does not reach exactly zero when training (a perfect fit to the training data), other patches apart from the labeled patch can provide larger focusing scores and therefore can be picked in order to estimate the ρ value for that spatial patch. For the remaining patch groups located within $y = 0 - 5.6$ km, the CNN did not encounter these patches during training and therefore the estimated ρ from these patches will provide a more robust indication of the generalization of the network to patches outside of the training set.

With the ρ values estimated from each spatial group, I assigned the estimated ρ value to all pixels within the cube, and then from all constant ρ patches, formed the full ρ cube. I then smoothed the ρ cube with a triangular smoother of length 0.15 km in the z-direction and 0.75 km in both inline and crossline directions. The resulting $\rho_{\text{CNN}}(z, x, y)$ is shown superimposed on the unfocused image in Figure 4.23. As expected, the estimated $\rho_{\text{CNN}}(z, x, y)$ appears on average to be between 0.96 and 0.98. In order to provide a proper comparison with $\rho_{\text{SMB}}(z, x, y)$ shown in Figure 4.16, the min and max ρ values in Figure 4.23(a) are 0.95 and 1.05 respectively. However, because the values of ρ in ρ_{CNN} fall within a narrow range, it is difficult to see how $\rho(z, x, y)$ changes spatially. To provide a better display of $\rho_{\text{CNN}}(z, x, y)$, Figure 4.23(b) shows the same display as Figure 4.23(a) but with a min ρ value of 0.95 and a max ρ value of 0.99. It is apparent in this figure that the CNN has predicted lower ρ values along the chalk horizon. Additionally, it appears to have predicted lower ρ values for inlines between 4 and 12 km.



(a)



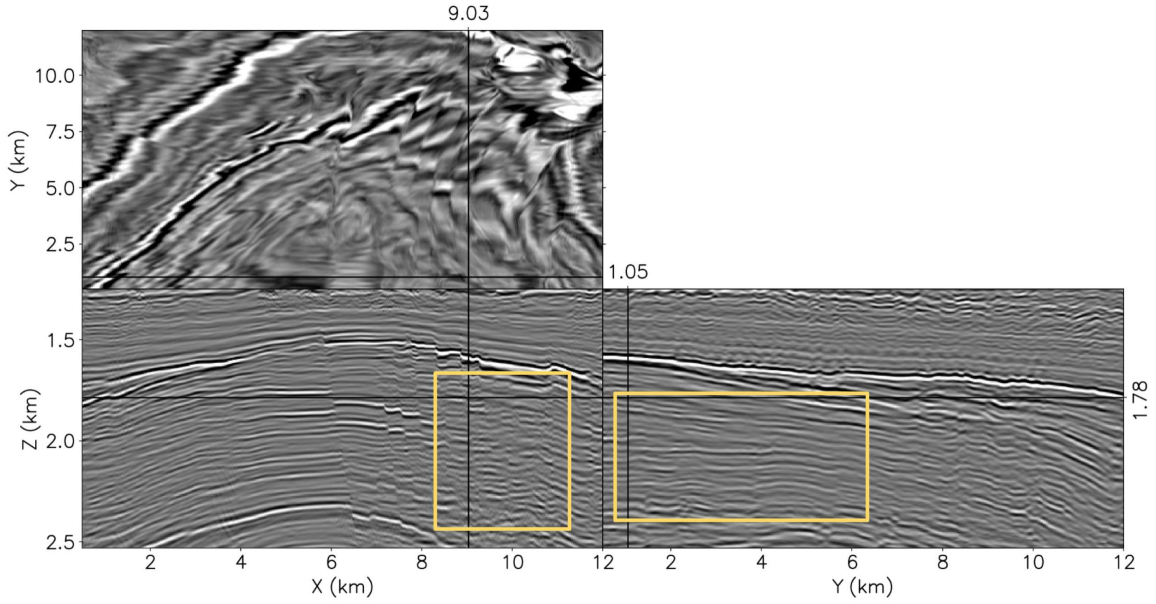
(b)

Figure 4.23: The estimated $\rho(z, x, y)$ as a result of performing CNN-based focusing analysis. Panel (a) shows ρ_{CNN} superimposed over the unfocused image with the same range of ρ as Figure 4.16. Panel (b) also shows ρ_{CNN} superimposed on the unfocused image, but with a max value of $\rho = 0.99$. [**CR**]

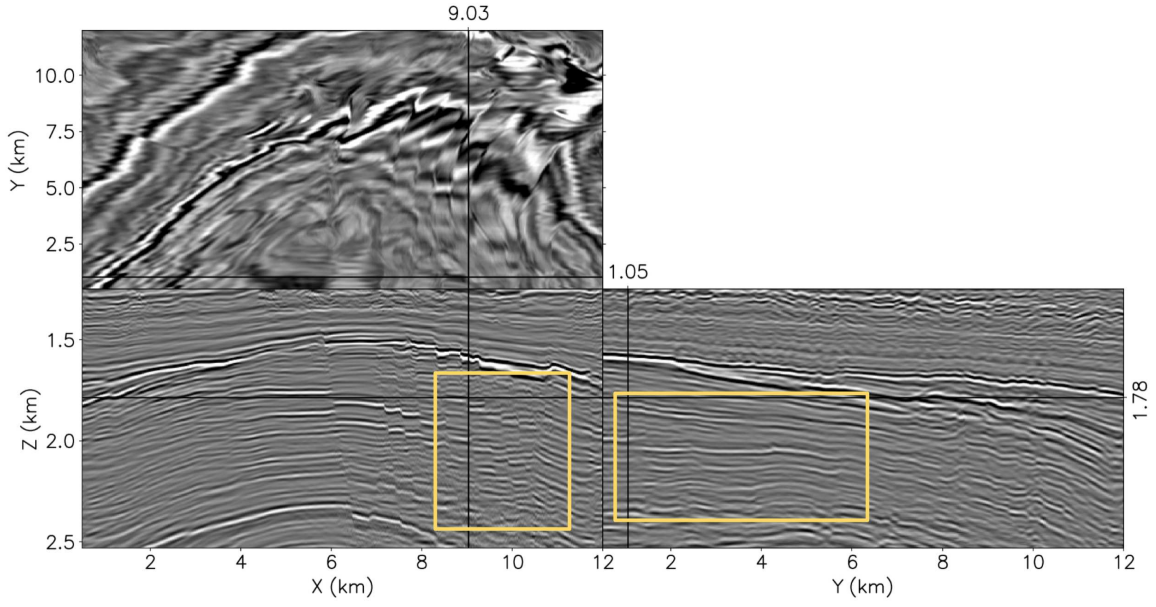
QC AND COMPARISON OF THE RESULTS

As I did in Chapter 3, I used the "refocusing" procedure described in Equation 3.1 in order to QC and better compare ρ_{SMB} and ρ_{CNN}. Figure 4.24 shows the comparison of the image refocused from ρ_{SMB} and the image refocused from ρ_{CNN}. The inline slice is extracted at $y = 1.05$ km, the crossline slice is extracted at $x = 9.03$ km and the depth slice at $z = 1.79$ km. As ρ_{CNN} was different from 1.0 only at the positions of the faults, any differences of interest between the two images will be present in the faulted region of the image. The most obvious difference between the two images is present within the crossline slice and is highlighted by the yellow box. Comparing the two images within that region, it is apparent that two horizons positioned within 2.0 - 2.2 km in depth are significantly more coherent within the image refocused with ρ_{CNN}. Additionally, a small fault that displaces these two horizons and positioned at approximately $y = 5.5$ km is readily apparent in the image refocused with ρ_{CNN} and hardly visible in the ρ_{SMB} refocused image. Comparing the inline slices of the two images, we can also observe that the faults within the highlighted are better imaged within the ρ_{CNN} refocused image. The heavily faulted horizons appear more coherent in the ρ_{CNN} image therefore making the faults much easier to distinguish.

To better compare the faults between the original unfocused image and the refocused images, I also performed 3D fault segmentation on each of the images. The procedure for fault segmentation was the same as I used for segmenting the residual migration images shown in Figures 4.18 and 4.19. Figure 4.25 shows the comparison of an inline slice extracted at $y = 1.05$ km for the unfocused image, the image refocused with ρ_{SMB} and the image refocused with ρ_{CNN}. The left column of the figure shows the inline image and the right column shows the predicted fault confidences superimposed on the images. Comparing the faults as well as the predicted fault confidences between the different images, it is apparent that in general, the image refocused with ρ_{CNN} provides the best image of the faults and the most geologically feasible fault confidences. This is first most apparent on the two faults that make up the graben positioned at approximately 6.5 km. Comparing these faults beneath a depth of 2.3 km, it is clear that the faults are better defined in both refocused images. Examining



(a)



(b)

Figure 4.24: Comparison of 3D F3 images refocused with (a) ρ_{SMB} (b) ρ_{CNN}. The highlighted boxes indicate regions of improvement of the ρ_{CNN} image over the ρ_{SMB} image. [**CR**]

the remaining portions of these two faults, it is also clear that the predicted fault confidences are more continuous in both refocused images. Comparing the faults positioned between 8.5 - 12 km in both images, it is apparent that they are better imaged in the image refocused with ρ_{CNN}. In both the unfocused and image refocused with ρ_{SMB}, the faulted horizons are incoherent and hardly visible (this is especially the case for between 1.7 - 1.9 km in depth and 10 - 11 km in the inline direction). However, in the image refocused with ρ_{CNN}, we can observe clear discontinuities in these horizons indicating the positions of the fault planes. This improvement in the image has also led to more geologically realistic as well as continuous predicted fault confidences in the image refocused with ρ_{CNN}.

In addition to comparing an inline slice, examining the faults along a depth slice will provide a useful comparison between the three images. Figure 4.26 shows a comparison of a depth slice extracted at $z = 1.785$ km from each of the three images. Again, the images are shown in the left column and the right column shows the fault confidences overlain on the images. Note that the depth slice has been windowed in x from 8 - 12 km and in y to a maximum of 8 km. Comparing the three images in the left column, it is apparent that the region of the faults between 8 - 9 km in x and 6 - 8 km in y appear to be the best defined in the image refocused with ρ_{CNN}. This also is represented in the predicted fault confidences which also appear to be sharpest within this region for the image refocused with ρ_{CNN}. Additionally, we can observe clear improvement in the predicted fault confidence for the fault whose inline trajectory is centered on average near $x = 10.5$ km. In the unfocused image, we observe that this fault is not very well delineated by the fault confidences in the unfocused image. In the image refocused with ρ_{SMB}, the fault is delineated, but somewhat incoherent and the prediction appears noisy. Observing this fault in the image refocused with ρ_{CNN}, the fault is clearly delineated by the fault confidence and the fault confidence is coherent and appears to be consistent with the fault observed in the depth slice in the left column of Figure 4.26(c).

While we can observe an improvement in the coherence of the fault positioned at approximately $x = 10.5$ km within the depth image shown in Figure 4.26(c), it is

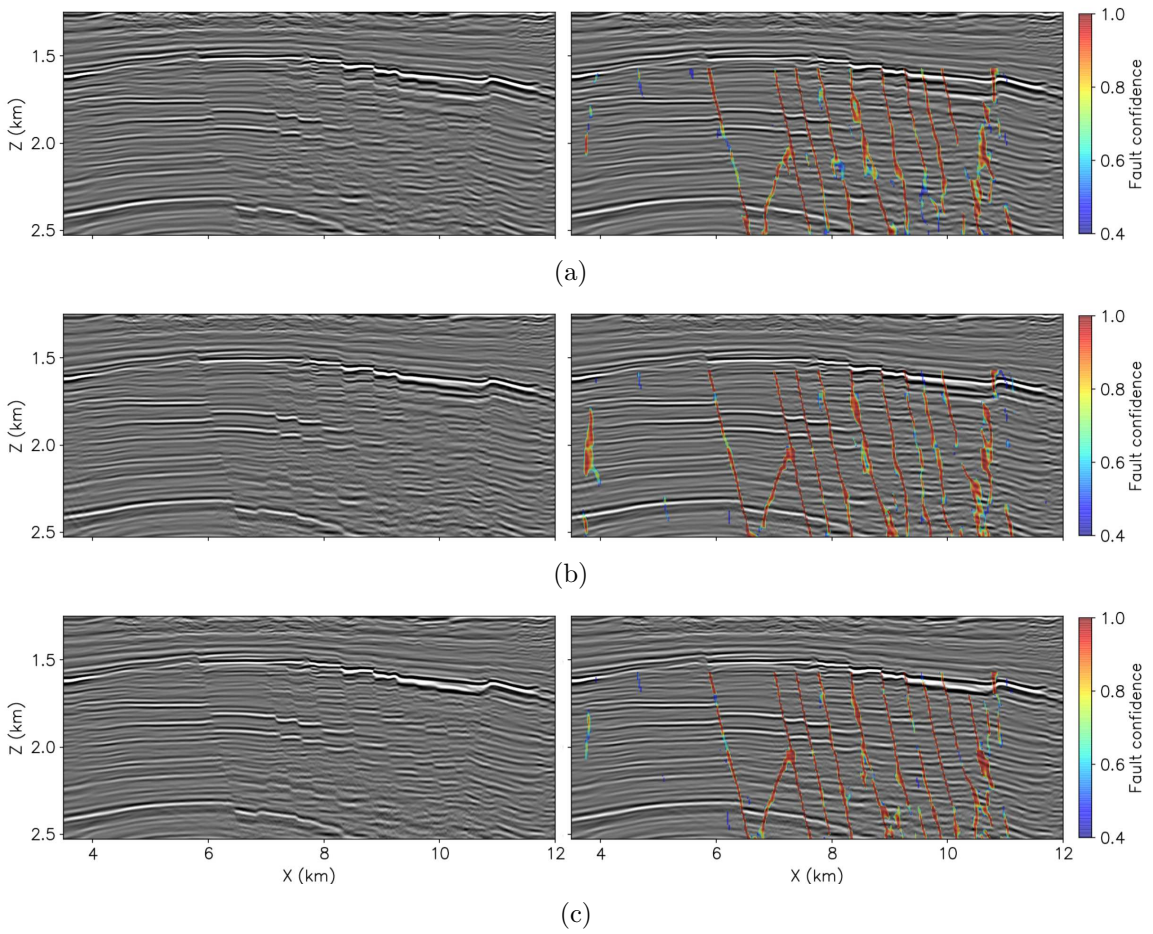


Figure 4.25: Comparison of (a) the original unfocused image, (b) the image refocused with ρ_{SMB} and (c) the image refocused with ρ_{CNN} for an inline slice extracted at $y = 1.05$ km. The left column shows the images and the right column shows the images with the fault confidence superimposed. [**CR**]

possible that the CNN is merely picking coherent noise that it has identified as a fault and is important to verify this prediction within an inline slice. Figure 4.27 shows the comparison of the three images for an inline slice extracted at $y = 4.175$ km. First comparing this inline with the inline shown in Figure 4.25, we can observe that for all three images, the faults are better delineated in the inline shown Figure 4.25. This is true for all inlines moving away from the origin of the crossline axis. The exact cause of this is currently unknown and would require further investigation of the velocity model and migrated image as well as a possible anisotropic velocity analysis. In spite of the fact that image of the faults is poorer for this inline, the crossline faults are still apparent within the images and again, appear to be best imaged within the image refocused with ρ_{CNN}.

Examining the faults within these inline images at approximately 1.8 km in depth and at $x = 10.5$ km, we can find the fault that corresponds to the fault of interest in the depth slice (the tip of the rightmost fault in each of images). First, comparing this fault within the images shown in the left column of Figure 4.27, we can observe that the fault appears to be undermigrated in both the original image as well as the image refocused with semblance. However, it appears to be properly focused within the image refocused with ρ_{CNN}. Observing the fault confidences within each of these images, it is clear that in the original unfocused image, the CNN detected the discontinuity to the right of this fault, but failed to detect the exact position of this fault, hence leading to the absence of this fault in the depth slice. In the image refocused with ρ_{SMB}, we can observe that the while the fault is detected, the fault confidence is spread out over the fault with a low fault confidence. This also explains the incoherent and noisy prediction observed of this fault in the depth slice shown in the right column of Figure 4.26(b). Finally, in the image refocused with ρ_{CNN}, we can observe that this fault has been detected by the CNN and the predicted fault confidence is coherent and coincides with the fault seen in the inline image. Again, this is consistent with the coherent predicted fault confidence seen in the depth slice of the image refocused with ρ_{CNN}.

While the image refocused with ρ_{CNN} does in general provide an improved image

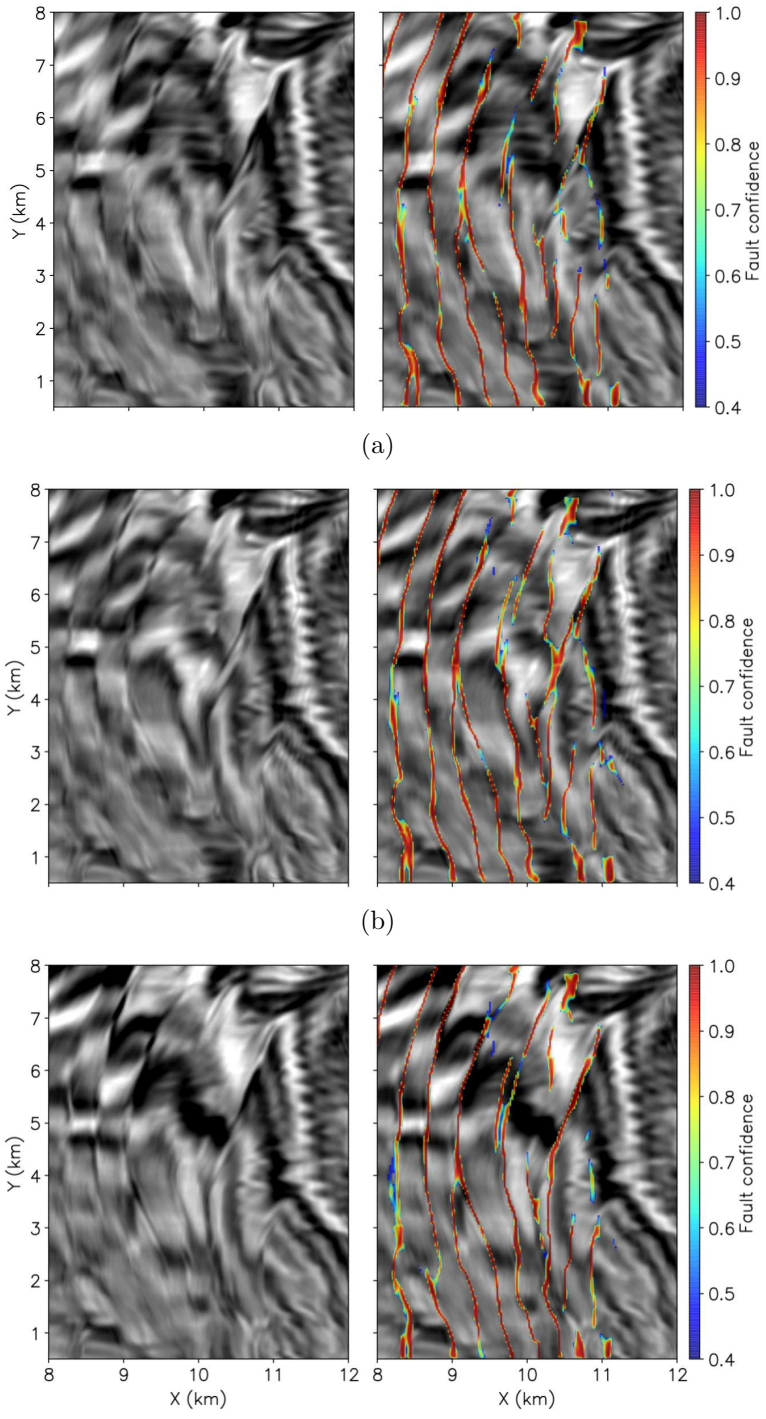


Figure 4.26: Comparison of a depth slice extracted at $z = 1.785$ km from (a) the unfocused image, (b) the image refocused with ρ_{SMB} and (c) the image refocused with ρ_{CNN}. The fault with a trajectory centered at approximately $x = 10.5$ km appears to be most coherent within the image refocused with ρ_{CNN}. [**CR**]

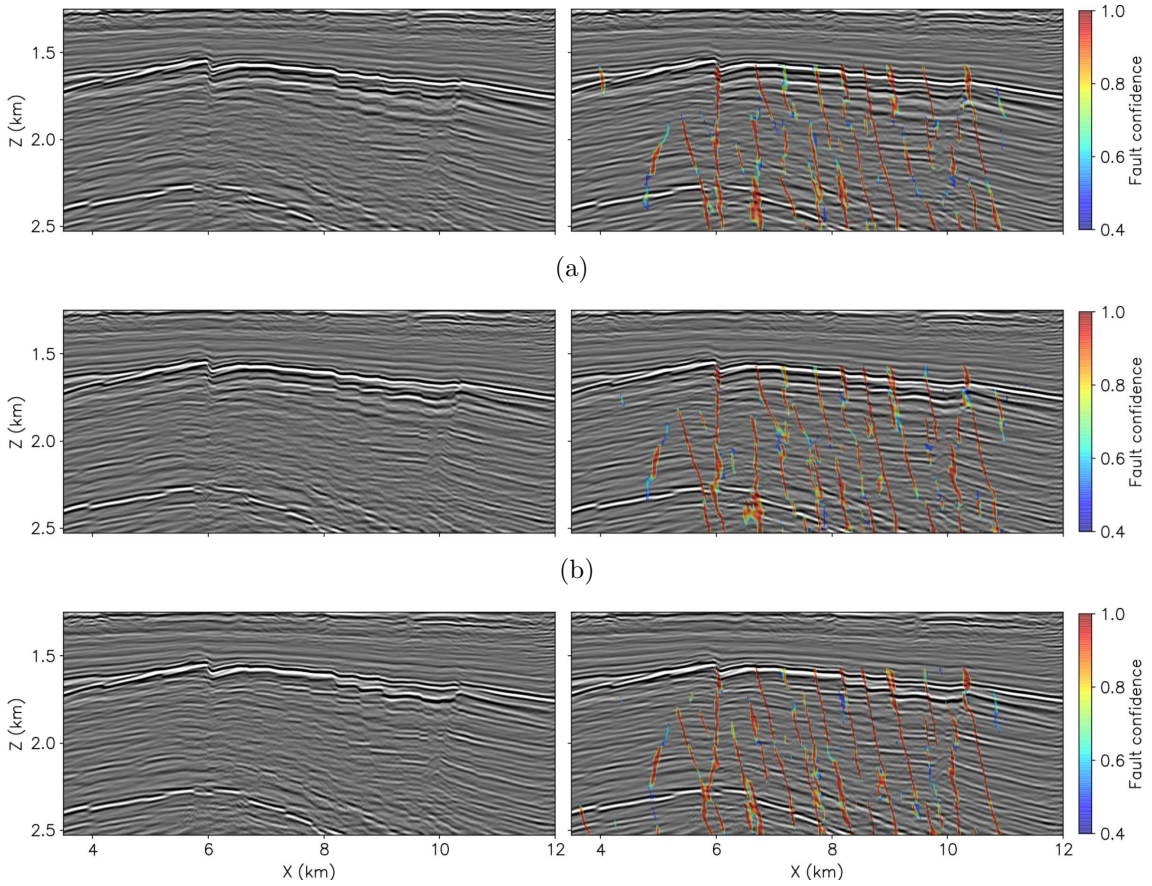


Figure 4.27: Comparison of an inline slice extracted at $y = 4.175$ km from (a) the unfocused image, (b) the image refocused with ρ_{SMB} and (c) the image refocused with ρ_{CNN}. [**CR**]

of these faults as well as the predicted fault confidence, the improvement is not global throughout the image and in certain locations within the image, the image refocused with ρ_{CNN} does not show any improvement over the unfocused image. An example of this is shown in Figure 4.27(c), where the predicted fault confidence associated with the fault positioned at approximately $x = 9$ km has degraded at depths near 2.25 km when compared with both the original as well as the image refocused with ρ_{SMB}. Closely examining this fault, we can observe that this is due to a less apparent discontinuity in the horizon at a depth of approximately 2.25 km. This horizon in the ρ_{CNN} appears to cross the fault plane at this depth whereas it terminates in the original image and the image refocused with ρ_{SMB}.

As a final QC I compare the refocused images with the original "focused" image that was obtained from the original velocity model (no constant scaling factor). Panels (a), (b) and (c) of Figure 4.28 show the inline slice extracted at $y = 1.05$ km, the inline slice extracted at $y = 4.175$ km and the depth slice extracted at $z = 1.785$ km within the focused image respectively. Note that in order to account for the depth shift between the images, I shifted the focused image in depth in order to approximately align the two images for comparison purposes. Comparing the inline slice extracted at $y = 1.05$ km of each of the images in Figure 4.25 with the inline of the focused image, it is clear that the faults within the image refocused with ρ_{CNN} are most similar with the faults in the focused image shown in Figure 4.28(a). This is also apparent when comparing the depth slices in Figure 4.26 with the depth slice from the focused image. However, when comparing the depth slice from the image refocused with ρ_{CNN} shown in Figure 4.26(c), with the depth slice from the original image, we can see an improvement in the automatic detection of the fault with the trajectory centered at approximately $x = 10.5$ km. The fault confidences appear sharper and more geologically consistent for this particular fault shown within the image refocused with ρ_{CNN}. This is likely due to the fact that the original migration velocity did not result in a perfectly focused image and that the CNN was able to identify improved focusing in the residually focused images thus resulting in an improved image of the fault.

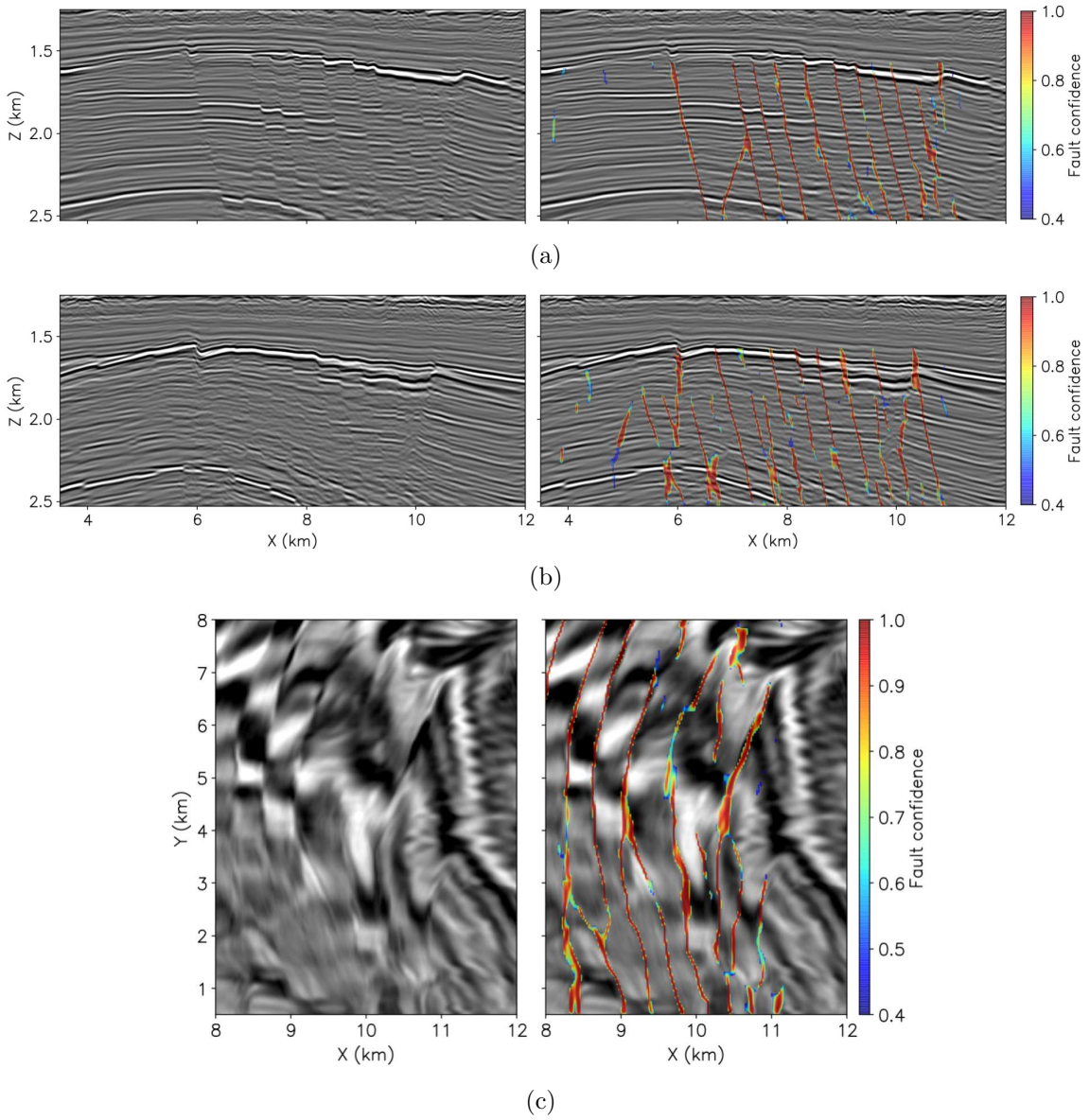


Figure 4.28: Three different slices of the focused migrated image (without the velocity error) included for reference. (a) An inline slice extracted at $y = 1.05$ km and to be compared with Figure 4.25, (b) an inline slice extracted at $y = 4.175$ km and to be compared with Figure 4.27 and (c) a depth slice extracted at $z = 1.785$ km and to be compared with Figure 4.26. Note that the focused image was shifted in depth in order to best align the images for comparison with the unfocused and refocused images in Figures 4.25, 4.27 and 4.26. [**CR**]

Comparing the inline extracted at $y = 4.175$ km, it is also clear that the image refocused with ρ_{CNN} (Figure 4.27(c)) is most similar to the original migrated image shown in Figure 4.28(b). However, in comparing these images, we can observe several differences. Similar to as I pointed out when comparing this inline refocused with ρ_{SMB} and ρ_{CNN}, it is clear that the fault positioned at approximately $x = 9$km has a more geologically consistent fault confidence within the original migrated image shown in Figure 4.28(b) than in Figure 4.27(c). Additionally, a feature of interest within this inline is the strong reflector positioned between $x = 9 - 10.1$ km and at $z = 1.75$km in depth. It is apparent in both the original migrated image and the image refocused with ρ_{SMB} that this reflector has a negative polarity, however in the image refocused with ρ_{CNN}, it has a positive polarity. While just from analyzing these images it is difficult to assess the correctness of the imaged reflector, as this reflector is positioned near the depth slice shown in Figure 4.26 it is possibly a case in which the CNN-based approach was able to detect improved focusing within a residually focused image in that region.

DISCUSSION

In the previous section, I showed from two inline slices as well in a depth slice that in general the refocused image from ρ_{CNN} provided a better image of the faults than did the original unfocused image as well as the image refocused with ρ_{SMB}. However, it is important to compare the application of the two methods to this 3D dataset and discuss their relative advantages and disadvantages. While Chapter 3 does contain a discussion between both the semblance-based and CNN-based approaches, the application of both methods to a different and 3D field data example has brought to light some additional key advantages and disadvantages of each method.

Comparison of computational cost

The first and most obvious point of comparison that arises with a 3D data application is the computational cost associated with each method. Regardless of the method used for focusing analysis, the main bottleneck in performing focusing analysis on a collection of 3D prestack residually migrated images is maneuvering the explosion of data that results from a 3D prestack residual migration. For the comparatively small windowed F3 dataset used in this chapter, the input prestack data were only approximately 40 GB. Performing extended imaging and prestack residual migration generated an approximate total of 4 TB of focused and unfocused images. (Recall also that the residual migration images shown in this chapter were generated with common-azimuth residual migration. A full prestack residual migration would easily generate over 100 TB just for this small example.) The goal of any focusing analysis is to then generate a $\rho(z, x, y)$ for the entire volume using this large ensemble of focused and unfocused images. The semblance-based approach is attractive in processing this large amount of data as each spatial image point can be treated independently and therefore this approach is highly parallelizable and also little memory is required for semblance-based focusing analysis of one spatial image point. For the semblance-based focusing analysis described in this chapter, I used a total of 640 CPU cores and 400 GB of RAM distributed across 20 compute nodes, which completed the

semblance-based focusing analysis in approximately 2 hours.

In comparison to the semblance-based approach, the CNN-based approach requires both significantly more memory as well as compute resources. Since the CNN-based approach is considerably more involved than the semblance-based approach, it is easiest to analyze the cost of the method by separately examining the cost of the training and the prediction stages of the workflow. The training stage can further be split into two sub-stages which are first to create the training data and then to train the CNN on the created training data. For the synthetic pre-training data, I used 640 cores and approximately 5 TB of RAM distributed across 20 nodes to create a total of 4,608 focused and unfocused patches in approximately 8 hours. For the field training data, I first formed patches from all prestack residual migration images. This required pre-processing of the data (e.g., structure-oriented smoothing on each inline) as well as sufficient memory in order to hold the patches in memory before writing to disk. Using 752 cores and approximately 4 TB of memory distributed across 61 nodes, I created a total of 528,525 patches in 9 hours. Note that by introducing 50% overlap in each spatial direction, the size of the dataset further increases from 4 TB to a total of 16 TB.

For the training stage, the training was performed on four V100 NVIDIA GPUs with 16 GB of RAM and approximately 300 GB of CPU (host) RAM. This large amount of CPU RAM was required in order to hold all training samples in CPU memory before individually transferring a patch to the GPU memory during training. The combined time of both pre-training and field data training stages was approximately 3.25 hours. Note that this time could be significantly reduced by distributing the training across multiple GPUs and also using optimized 4D cross-correlation operators (Choy et al., 2019). However, it is important to note that the dataset described in this chapter is relatively small and a much larger training dataset would be necessary for the network to be able to generalize across different surveys. Therefore, even with distributed training and optimized cross-correlation kernels, the computational cost associated with training is non-negligible.

For the prediction stage, I distributed the CNN focusing analysis of 127,575

patches across 8 V100 16 GB GPUs. As all patches reside on disk, this operation is heavily input-output (I/O) bound and therefore this operation finished in approximately 8 hours. While I used a naive approach of organizing the patches on disk and a more sophisticated approach could be employed, these approaches cannot overcome the increase in the size of the data that is introduced when forming overlapping patches. The heavy cost of this prediction step is unfortunate as it does not follow the slow-training, fast-prediction model as promised and demonstrated by other deep learning applications (NVIDIA Corporation, 2015; Araya-Polo et al., 2018). These applied deep learning examples showed that while they have high upfront training costs, the prediction costs after the model has been trained are minimal. Additionally, it should be noted that even with very large training datasets, it is unlikely that the model will ever be "fully-trained" in that it will perform a robust focusing analysis on any image that it may be provided. Rather, after initial training, additional smaller datasets will need to be provided in order for the CNN to generalize to new images. This additional cost must be also considered when performing a CNN-based focusing analysis.

While the computational cost of the CNN-based focusing analysis is significantly higher than the semblance-based approach, a significant reduction in cost can be achieved by using only the stacked images. Recall also that as I discussed in Chapter 1, the most recent focusing analyses performed within industry operate only on stacked images (DeLaughter et al., 2014). Therefore, if only the stacked image is needed as input to the focusing analysis, then a prestack residual Stolt migration can be applied to the input image, but only the stacked residual migration images would need to be saved to disk. This reduces the size of the dataset to be used for focusing analysis by an order of magnitude and greatly reduces the data I/O costs that constitute the biggest computational bottleneck for both semblance-based and CNN-based focusing analysis. Additionally, optimized 3D cross-correlation operators are readily available within cuDNN (Chetlur et al., 2014) and therefore both training and prediction times will be reduced significantly. However, with the removal of the additional information provided along the offset/angle axis, a larger, highly-curated training dataset will likely be needed to account for large geologic variability between patches.

CNN generalization

As the CNN-based approach is a supervised learning-based approach, the generalization of the model to a wide variety of data is a key concern. As discussed in Chapter 3 with the 2D GOM field data example, the CNN trained for performing focusing analysis on the F3 migrated image was trained on synthetic data tailored for this example as well as on many examples extracted directly from residually migrated images computed from the original F3 image. Therefore, unless the reflectivity and geology of a new unseen image resemble the reflectivity and geology of the samples used in this chapter and also were created using the same imaging and residual imaging parameters, it is unlikely that the CNN will perform equally well on a new image as it did on the F3 image. However, the results in this chapter do show that the CNN can generalize over samples within a survey, which is a first step towards training a CNN that can generalize to samples from different geological regions. In order for the CNN to generalize to new images from a wide range of geological regions, a large training dataset of focused and unfocused seismic image patches extracted from images that have undergone focusing analysis would be required. Such a dataset would likely result from an industrial effort in which geophysicists label focused and unfocused portions of seismic images from which patches can be formed (Liu et al., 2021). However, as I also discussed in Chapter 3, because the 4D CNN is not used directly as a classifier, but rather the predicted focusing score is used as a focusing measure and only the highest focusing score within a spatial patch group is used to estimate ρ, the precise classification of particular patch is less crucial. The fact that the unthresholded output of the CNN is used as a focusing measure should ease the labor-intensive task of developing a generalized model for image focusing analysis.

ACKNOWLEDGEMENTS

I would like to thank TNO for providing the F3 data used in this chapter and for Johannes Ravestein at TNO and Shuki Ronen for their help in processing the data. Also, the majority of the computing for this chapter was performed on the Sherlock

cluster. I would like to thank Stanford University and the Stanford Research Computing Center for providing computational resources and support that contributed to these research results.

Chapter 5

Conclusions and future directions

THESIS SUMMARY

In this thesis, I presented a novel workflow for performing automatic interpretative seismic image-focusing analysis. The key component of the workflow is the use of a fault-focusing CNN that assesses the focusing of a seismic image patch over both the physical and angle/offset axes. Through the use of all axes of the seismic image patch, and by training on datasets consisting of synthetic and real focused and unfocused seismic image patches containing geological faults, the CNN is able to robustly assess the focusing of a seismic image patch. Additionally, I demonstrated that the CNN extracts the focusing information provided by the geological faults present within the image patches in order to perform the assessment of the focusing of the seismic image patch.

To assess the effectiveness of the CNN-based workflow, I applied it to a subsalt synthetic and two real data examples and compared the results with a traditional semblance-based focusing approach. Through the comparison of the faults refocused with focusing maps estimated from the semblance and CNN-based approaches, I demonstrate that the CNN-based approach provided a more robust estimate of the focusing errors present within the unfocused images. Additionally, for the field data examples, I performed fault segmentation on the images refocused from both the

CNN-based and semblance-based approaches. In both cases, the interpreted faults surfaces from the CNN-based approach were more continuous and consistent with the geology observed within the images.

FUTURE DIRECTIONS

Creating large and robust training datasets from real seismic images

While the results presented in this thesis demonstrate that the CNN-based fault-focusing analysis is able to robustly assess focusing errors from faults within prestack seismic images, the results shown here are specific to the datasets used in this thesis and are by no means comprehensive. While in general it can be difficult for any newly developed method to generalize to multiple datasets, it is especially difficult for supervised learning-based approaches as they are entirely data-driven and are designed to fit a set of training data. Therefore, independent of the design of the CNN that is to be trained, it is crucial that the dataset used for training the CNN contain many samples and be comprehensive and representative of the samples not found within the training dataset in order for the CNN to have any hopes of accurately classifying samples that do not pertain to the training dataset. Unfortunately, as I have stated previously, at the time of writing this thesis, I do not have such a dataset and therefore, the models I have trained for CNN-based will likely not generalize well to new prestack images. While I do have the capability of creating many synthetic training examples and also, because I do not use the CNN strictly as a classifier, the CNN-based workflow is less vulnerable to the domain shift, in the event that more data become available, the generalization of the CNN would greatly improve from training on additional data.

In spite of the fact that the models trained as part of the work shown in this thesis are in large part limited to the images from which the training datasets were created,

creating a large training set of focused and unfocused seismic images is not an impossible task. While focusing analysis of seismic images is no longer routinely performed, before the shift of the focus in industry to the use of full-waveform inversion for estimating subsalt velocities, focusing analysis was a key component of subsalt imaging and model-building workflows (Wang et al., 2006, 2008; Epili et al., 2011; Ma et al., 2011; Xie et al., 2012; DeLaughter et al., 2014). Therefore, it is entirely possible to create 3D image patches from the focused and unfocused images that resulted from the focusing analyses of subsalt images. As I demonstrated in Chapter 4 with the F3 prestack image patch, many thousands of 3D seismic image patches can be created from the focusing analysis performed on a single image. However, to create a training set for supervised learning, each of these patches needs to be labeled. This labeling procedure could possibly be done in an interpretative manner (e.g., along horizons extracted from the image or in a faulted portion of the image) and therefore, hundreds of patches could be labeled at a time by an experienced seismic processing geophysicist. Additionally, since automatic approaches have been developed for performing image-focusing analysis, these approaches can be used in order to help accelerate the labeling of focused and unfocused image patches (Négron et al., 2000; Whiteside et al., 2011).

Other imaging methods for seismic image-focusing analysis

For all examples of focusing analysis presented in this thesis, I used the method of prestack Stolt residual depth migration. As I have previously stated, this method is advantageous in that it can create focused and unfocused prestack seismic images in a highly computationally efficient manner. Moreover, it has demonstrated to successfully provide focused and unfocused images suited for migration velocity analysis in complex geological areas such as in the presence of salt bodies (Sava and Biondi, 2004b). However, in spite of these advantages, it has had little adoption for focusing analysis within the seismic imaging community and other methods for focusing analysis are typically preferred.

Over the years, the three most common methods for image-focusing analysis have been common-offset scans via prestack Kirchoff depth migration, wave-equation migration (WEM) scans and RTM delayed imaging time (DIT) scans (Audebert et al., 1996; Wang et al., 2006, 2009). In order for the fault-focusing CNN to best assess the image focusing on image patches generated from these methods, it would need to be trained with training datasets composed of images from each of these methods. Additionally, it is important to note that focusing analysis with WEM and RTM DIT scans produce only stacked images and therefore the patches do not contain any curvature information provided by the angle/offset gathers While I demonstrated that the CNN-based framework is able to accurately assess focusing errors from faults within stacked seismic images, when training on real seismic image patches, the addition of the angle/offset axis will help the CNN to generalize to field training datasets with images originating from different geologic scenarios. Therefore, a highly-curated training dataset of unfocused and focused stacked images will enable the fault-focusing CNN to robustly assess the image focusing within only stacked images.

Additionally, it could be of added benefit to provide the CNN with more than just a single image patch from the set of focused and unfocused images. This is consistent with how experienced seismic processing geophysicists perform focusing analysis on stacked images (Négron et al., 2000; Wang et al., 2009; Whiteside et al., 2011). With multiple patches, the CNN would also be provided with the focusing axis (ρ axis for Stolt residual depth migration) which could allow for a more robust assessment of the focusing within stacked images. This approach would require modification of the architecture of the CNN and also of the CNN output. With multiple input patches, the CNN would then be tasked with classifying the patch that is best focused among the provided patches. This would require that CNN output a multi-class prediction as opposed to a binary prediction.

Incorporating additional geological focusing information

To assess the focusing of images within the physical axes of seismic images with a CNN, I train the CNN with seismic image patches containing focused and unfocused faults. While there are a number of geological features that could be used assess to the focusing of geological images, I chose to use faults as they occur relatively frequently in the subsurface and many faults of interest in the exploration geophysics community can be modeled with relatively simple geometric techniques in both 2D and 3D images (Clapp, 2014; Wu et al., 2020). Additionally, it is important to note that as faults in seismic images are identified as discontinuities along geologic horizons, implicitly, the coherence and continuity of reflections within seismic images are also necessary for the CNN to assess the focusing of a seismic image patch containing a fault.

However, the CNN-based image-focusing analysis workflow presented in this thesis is not limited to only using the focusing of faults and geologic horizons. Two additional geologic features that could be provided to the CNN for training are channels and salt bodies. The edges of channels within seismic images often provide strong enough discontinuities to create diffracted energy when unfocused due to velocity error (Bashir et al., 2020). Additionally, like faults, channels within seismic images can be modeled with relatively simple geometric techniques (Sylvester et al., 2019; Gao et al., 2020) and therefore, as I did with faults, synthetic focused and unfocused channels could be provided to the CNN in order to compensate for a lack of real training images containing focused and unfocused channels. However, while both channels and faults are 3D in nature, the focusing information provided by channels is considerably richer within depth slices than it is along inline and crossline slices. This restricts the focusing analysis of channels primarily to 3D images.

Although channels are relatively simple to model with geometrical techniques, salt bodies, conversely, are not (Zhang et al., 2009, 2013). Therefore, it is difficult to supplement field training datasets with synthetic images consisting of focused and unfocused rugose salt surfaces. However, as focusing analyses within industry has been primarily performed on geologic regions positioned near or below salt bodies,

there exists a plethora of data and expertise regarding the focusing and unfocusing of salt. Furthermore, as I demonstrated with the improved interpretation of faults in this thesis, focusing analysis often improves the interpretation of salt (Sava, 2003; Whiteside et al., 2011). Therefore, the expertise and data resulting from focusing analysis of salt bodies will allow for the potential creation of large training sets of seismic images containing focused and unfocused salt surfaces.

Appendix A

Automatic fault segmentation with a CNN

For all the 2D and 3D fault segmentations shown in Chapters 3 and 4 of this thesis, I closely followed the CNN-based approach presented by Wu et al. (2019). Therefore, while what I present in this appendix is not new, it does play a central role in the work presented in this thesis and therefore can aid the reader in better understanding the results presented in Chapters 3 and 4 of this thesis. The outline for this appendix is as follows: I begin by first introducing the 3D modified U-Net CNN architecture introduced by Wu et al. (2019). Then, I show both the 2D and 3D training samples used to train the 2D and 3D CNNs used in Chapters 3 and 4. Finally, I describe the loss function used for training the CNNs as well as the training process and show the results of fault segmentation on synthetic image patches.

U-Net CNN architecture

The CNN architecture used for the CNN-based fault segmentation in this thesis and introduced by Wu et al. (2019) is a modified version of a U-net architecture (Ronneberger et al., 2015). Figure A.1 shows a schematic of the architecture. The input

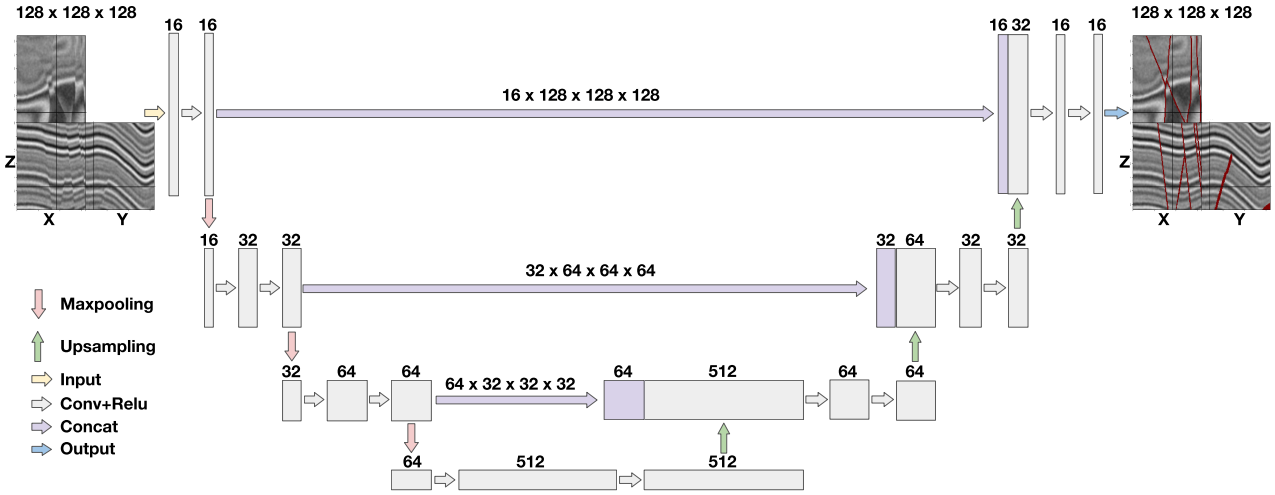


Figure A.1: 3D modified U-net architecture used for 3D fault segmentation. The input to the network is a single-channel 3D image patch and the output is a 3D image containing the predicted fault pixels. Note that the 2D architecture is equivalent but takes as input 2D image patches images and uses 2D operations. Figure modified from Wu et al. (2019). [**NR**]

to the CNN is a 3D stacked image patch of dimensions $128 \times 128 \times 128$ and the output is an image containing the predicted fault pixels and is also of dimensions $128 \times 128 \times 128$. The network itself consists of two primary components: a contracting path that shrinks the input image via a series of convolutions and max-pooling operations and an expanding path that expands the computed features using convolutions and upsampling operations. The 3D filters within each convolution operation are of size $3 \times 3 \times 3$ and the pooling kernels are of size $2 \times 2 \times 2$. Within the contracting portion of the network, the number of channels after each convolution are increased and the dimensions of the features are halved after each max-pooling operation until the feature maps consist of 512 channels that are $32 \times 32 \times 32$ pixels in size. (Note that the number on top of the boxes that represent the feature maps indicates the number of channels at that stage of the CNN). Within the expanding path of the CNN, the number of channels are decreased after each convolution and the feature maps are upsampled which doubles the size of the images along each dimension. These upsampling and convolution operations are repeated until a single-channel image of

dimensions $128 \times 128 \times 128$ has been obtained.

The U-net architecture can be considered as a specific implementation of a general class of CNN architectures known as encoder-decoder architectures (Zhou et al., 2018). The contracting portion of the network can be thought of as an encoder that computes low-dimensional features from the input image and the expanding portion then acts as a decoder that reconstructs the fault locations from the computed features. Apart from the bottleneck portion of the CNN that acts as the main connection between the encoder and decoder portions of the network, additional connections known as skip connections connect the higher dimensional features between the encoder and the decoder. The skip connections help the CNN to retain the high-resolution features extracted from the image at the early stages of the encoder portion of the CNN and thus improve the resolution and accuracy of the output segmentation. Additionally, for the skip connections to be valid, the dimensions of the features on each side of the connection must be the same. This requires then that the contracting and expanding paths of the CNN be symmetric which gives the CNN a "U" shape.

For the 2D fault segmentations shown in Chapter 3 of this thesis, I used a 2D variant of this architecture. The 2D CNN required input images of dimensions 128×128 and the output predictions were also of dimensions 128×128. The architecture of the CNN was identical to that shown in Figure A.1 but the convolution, max-pooling and upsampling operations were all 2D. The kernel sizes for the convolutions and pooling operations were the same for both 3D and 2D CNNs.

CNN training

In order to train the CNN, I first needed to create training samples. As described above, the CNN takes as input 3D image patches with faults and the label is also a 3D image but the pixel values are equal to one when the pixel is located on a fault plane and zero elsewhere. Due to the lack of publicly available 3D interpreted fault structures, I also followed the approach of Wu et al. (2019) and created 3D synthetic seismic images with faults and used the computed fault surface as the label. To

create the synthetic training patches, I used the same approach that I used to create synthetic images to train the fault-focusing CNN described in Chapter 2 of this thesis. Figures A.2 and A.3 show examples of 3D and 2D image patches used to train the fault segmentation CNNs used in Chapters 4 and 3. For the 3D CNN, I created a total of 256 training samples and for the 2D I created a total of 4,000. As the goal of the CNN is to classify whether a pixel contains a fault, a binary cross-entropy loss function could be used as the loss function to be minimized in order to estimate the CNN weights and biases. However, as each image patch contains significantly more non-fault pixels than fault pixels, there exists a large class imbalance within the training data. In order account for this class imbalance, Wu et al. (2019) suggest to use a weighted binary cross-entropy loss function which can be expressed as follows:

$$\mathcal{L}(\mathbf{w}, \mathbf{b}) = -\beta \sum_{i=1}^{M} y_i \log(c_i(\mathbf{w}, \mathbf{b})) - (1-\beta) \sum_{i=1}^{M} (1-y_i) \log(1-c_i(\mathbf{w}, \mathbf{b})), \quad \text{(A.1)}$$

where M is the total number of pixels in the training set, c_i is the predicted fault confidence and y_i is the label for the ith sample pixel of the training set, $\mathbf{w}$ and $\mathbf{b}$ are the weights and biases of the fault segmentation CNN and β is the ratio of non-fault pixels to total pixels in the training dataset. Comparing Equation A.1 to 2.19, we can observe that the two expressions are equivalent apart from the introduction of the β term. The introduction of β will appropriately weight the loss function according to the distribution of the classes found within the training dataset.

Using the synthetic training patches and Equation A.1 as a loss function, I trained both the 3D and the 2D fault-segmentation CNNs using the ADAM optimization algorithm with a learning rate of 1×10^{-4}. For the 2D fault segmentation CNN I specified a batch size of 20 samples and trained for a total of 30 epochs. Similar to the training of the fault-focusing CNN, I specified a batch size of one sample while training the 3D fault segmentation CNN and I trained for a total of 25 epochs. Figures A.4 and A.5 show the training and validation accuracies for the 3D and 2D fault segmentation CNNs respectively. For the validation data, I used a total of 52 samples and 800 samples during the training of the 3D and 2D CNNs respectively.

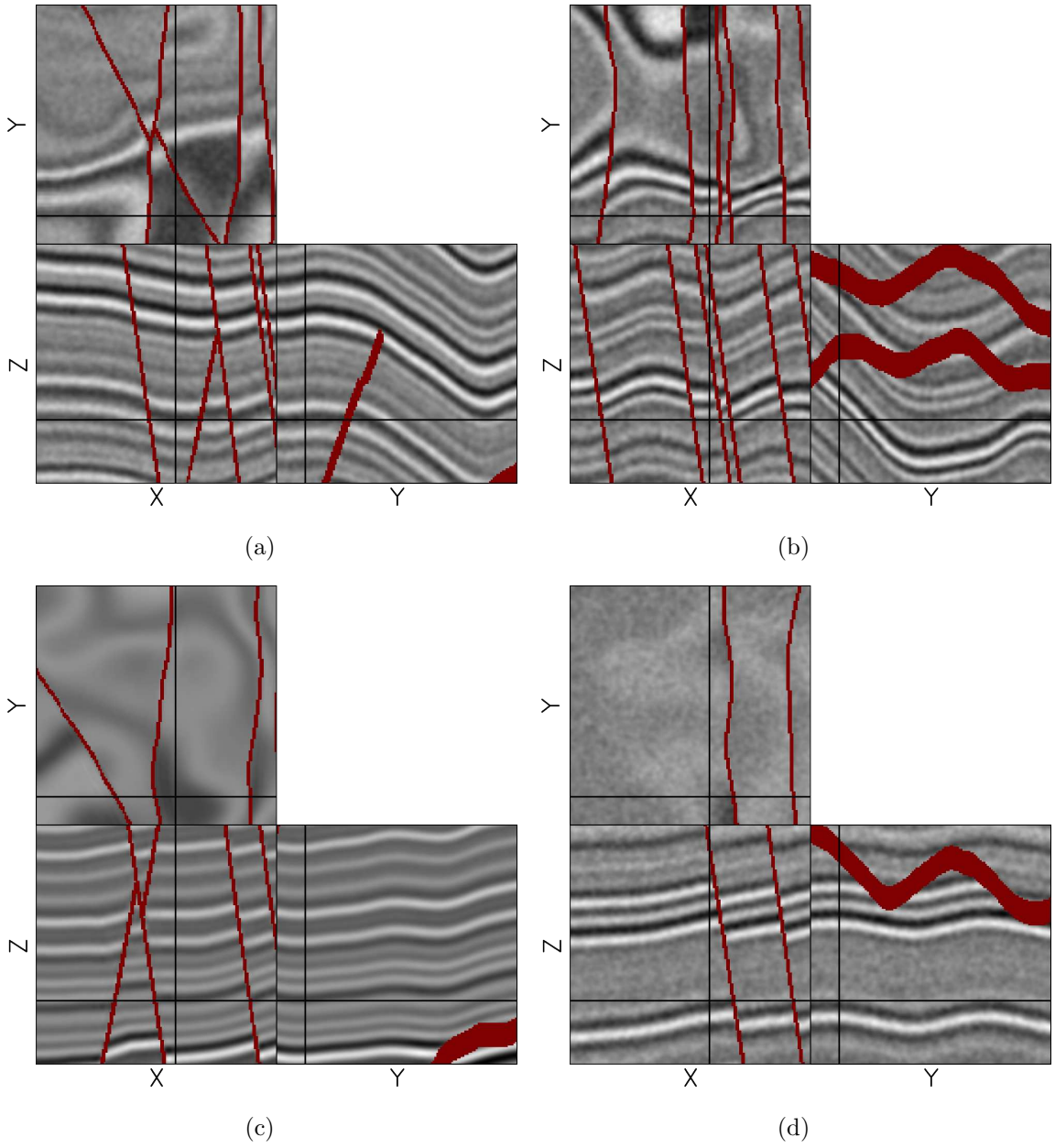
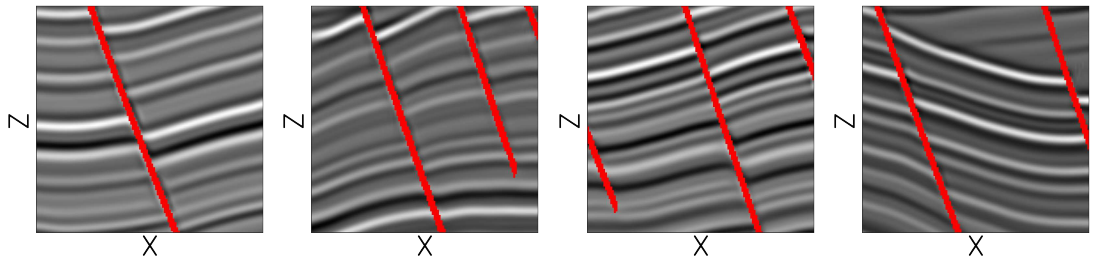
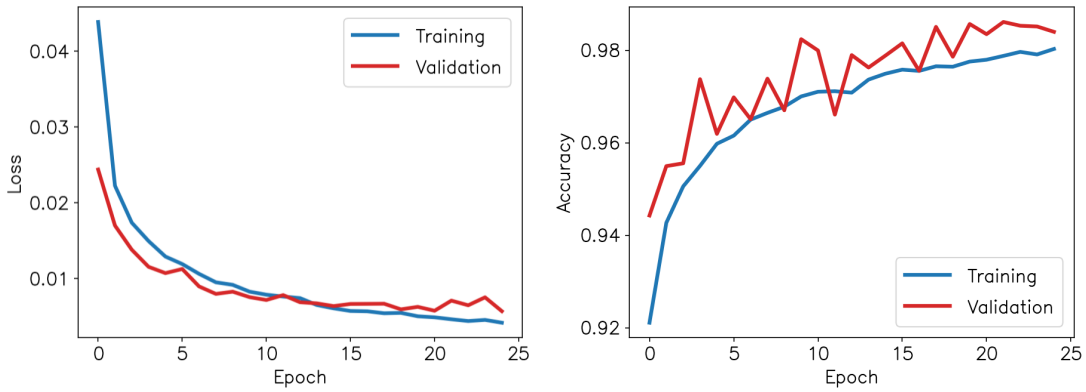


Figure A.2: Examples of 3D training patches used to train the 3D fault segmentation CNN. The red pixels indicate the positions of the faults and were used as the labels during training. [**CR**]



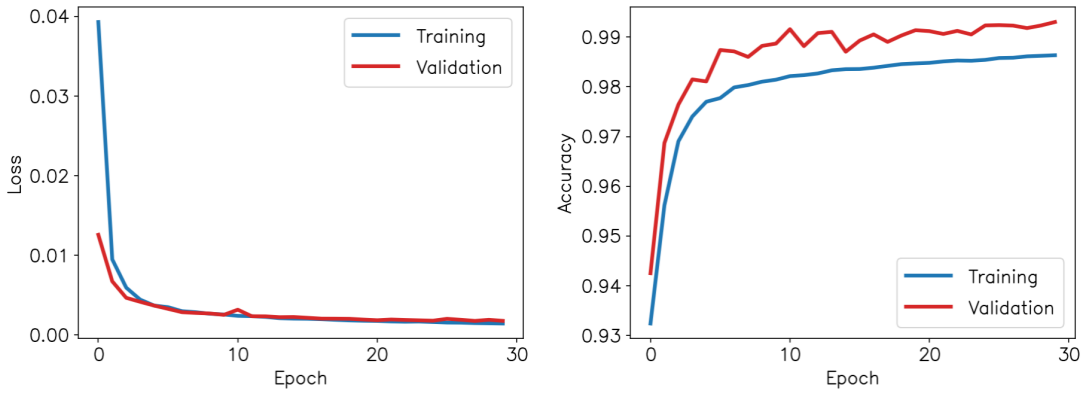
(a) (b) (c) (d)

Figure A.3: Examples of 2D training patches used for training the CNN used for fault segmentation in Chapter 3. Note that all faults had the same dip direction as the real data example in order to improve generalization from the synthetic training data to the real data example. [**CR**]



(a) (b)

Figure A.4: Learning curves obtained from training the 3D fault segmentation CNN. The CNN was trained for a total of 25 epochs on 256 3D synthetic training patches and 52 validation patches. The training and validation losses and accuracies demonstrate that the CNN is able to fit both training and validation data with low error. [**CR**]



(a) (b)

Figure A.5: Learning curves obtained from training the 2D fault segmentation CNN. The CNN was trained for a total of 30 epochs on 4000 2D synthetic training patches and 800 validation patches. [**CR**]

The learning curves in Figures A.4 and A.5 show that the CNN is able fit both the training and validation data with high accuracy.

Appendix B

Seismic image processing for domain shift reduction

As I described throughout this thesis, while using synthetic training data provides exact labels and potentially large training datasets, unless the distribution of the synthetic training samples is similar to the distribution of the real data, there will exist a domain shift between the two datasets and the generalization of the CNN will be poor. While an important step in minimizing this domain shift is to make the synthetic images as realistic as possible, applying additional processing steps to the real images (effectively making them more synthetic) can also help to minimize this domain shift. For the training examples shown in this thesis, I used an anisotropic non-linear filtering algorithm in order to suppress both random and coherent noise in the 2D and 3D real seismic images used in Chapters 3 and 4. Within the seismic imaging community, this filtering technique is commonly used for processing seismic images for interpretation and is known as structure-oriented filtering/smoothing (Fehmers and Höcker, 2003). In this appendix, I describe the structure-oriented smoothing algorithm in more detail and demonstrate how it improves the generalization of a fault-segmentation CNN trained entirely on synthetic training images.

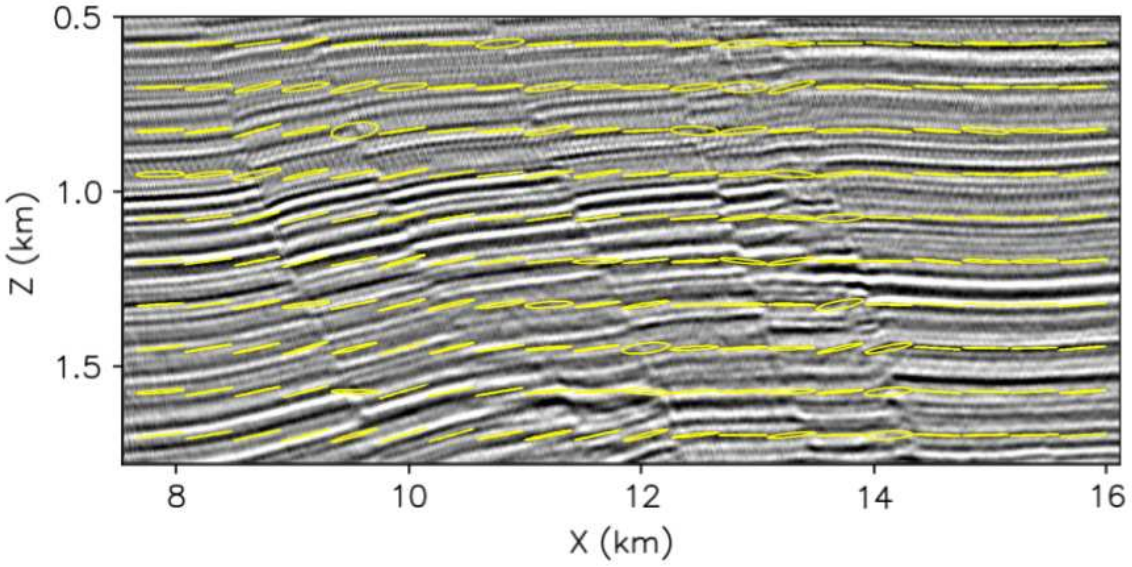


Figure B.1: The eigenvectors of the structure tensors computed for the 2D image used in Chapter 3 of this thesis. The eigenvectors are predominantly aligned with the horizontal layering present within the image. [**NR**]

Structure-oriented smoothing is a type of nonlinear filtering known as anisotropic-diffusion filtering (Weickert, 1998). Anisotropic-diffusion filtering is a partial differential equation-based method for image processing that solves an anisotropic-diffusion equation where the input image is used as an initial condition and the solution to the PDE provides the output smoothed image. I specifically use the algorithm introduced by Hale (2009) that uses an implicit method to solve the PDE making it more computationally-efficient and simpler to implement. He parameterizes the PDE by a structure-tensor field that is computed from smoothed outer products of image gradients and uses the eigenvectors of the structure-tensors computed for every pixel in the image in order to guide the smoothing caused by the anisotropic-diffusion process along the structures within the image. Figure B.1 shows an example of a selection of eigenvectors computed for the 2D image used in Chapter 3 of this thesis. The ellipses were computed by scaling the eigenvectors by a linearity coefficient constructed from the eigenvalues of the structure tensors (Hale, 2009). The strong alignment of the

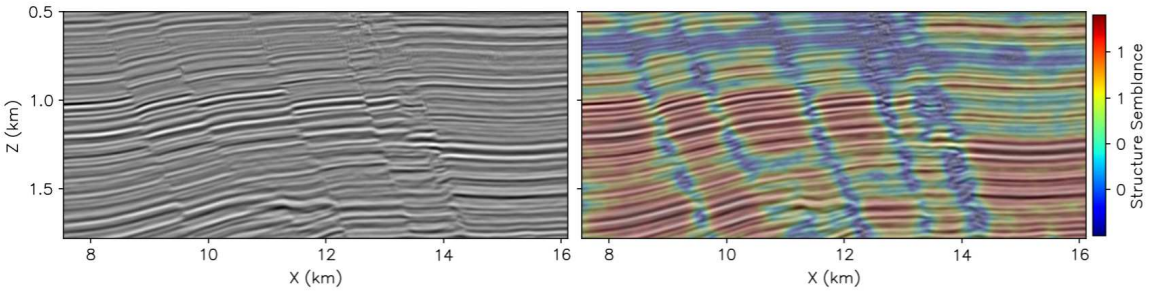


Figure B.2: The image smoothed with structure-oriented smoothing (left) and the computed structure-oriented semblance (right) used to weight the smoothing operator. The low semblance values near the fault positions indicate that the extent of smoothing will be less near the faults. [**CR**]

ellipses with the structure show that the eigenvectors agree with the layered structure present within the image (for an isotropic smoothing filter, the ellipses would be circles at each image point).

An additional step in the algorithm presented by Hale (2009) is the computation of a semblance parameter that weights the smoothing operator in order to reduce the extent of smoothing near discontinuities such as faults. This semblance parameter is computed from the ratio of the squared-smoothed image over the smoothed-squared image. Figure B.2 shows the result of computing the structure-oriented semblance parameter for the image used in Chapter 3. It is apparent from the figure that the lower semblance values align with the positions of the faults. Finally, with the computed structure tensors and semblance parameter, I smoothed the image by solving the anisotropic-diffusion equation presented in Hale (2009). The resulting smoothed image is shown in Figure B.2. It is clear that the smoothing process has reduced a significant amount of noise within the image. However, even with the semblance parameter, it has also slightly smoothed across the faults reducing the strength of the fault discontinuity and partially removing the fault plane reflections.

In addition to smoothing the real image, I also applied structure-oriented smoothing to the synthetic training images. While the fault-building process described in Chapter 2 does result in realistic faults at the exploration scale, the discontinuities

are often unrealistically strong across the fault and are not representative of what is often observed in real seismic images. Applying a structure-oriented smoothing operator helped to remove the excessive sharpness across those faults, while largely preserving the discontinuity.

In order to test if the use of structure-oriented smoothing does indeed reduce the domain shift between the synthetic and real datasets, I created two training datasets, one consisting of smoothed images and the other of unsmoothed images and I trained the CNN on each training set (resulting in two separate CNNs). Additionally, I used each of the trained CNNs for fault segmentation on both the smoothed and unsmoothed versions of the real GOM image used in Chapter 3. The results of the four fault segmentations are shown in Figure B.3 and are superimposed on the original image. Comparing each of the fault segmentations, it is apparent that the fault segmentation resulting from the smoothed real and smoothed synthetic images is the most continuous and best aligns with the faults present within the image. This demonstrates that for this example, introducing the structure-oriented smoothing operator on both the synthetic and real images proved to be best at minimizing the domain shift between the real and synthetic images.

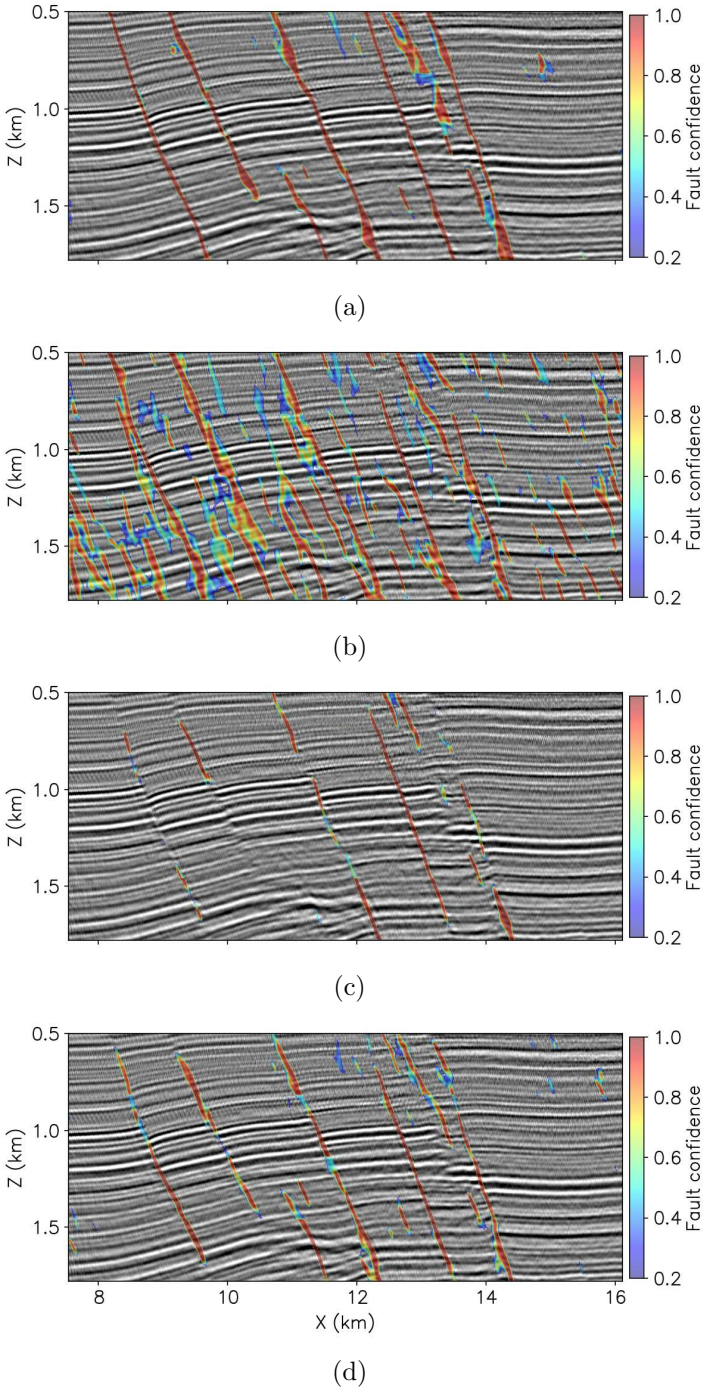


Figure B.3: Comparison of fault segmentations from (a) a CNN trained on smoothed synthetic images and provided a smoothed real image, (b) a CNN trained on smoothed synthetic images and provided an unsmoothed real image, (c) a CNN trained on unsmoothed synthetic images and provided a smoothed real image and (d) a CNN trained on unsmoothed synthetic images and provided an unsmoothed real image. [**CR**]

Appendix C

4D cross-correlations with 3D kernels

As typical deep-learning frameworks such as PyTorch and Tensorflow do not have 4D cross-correlation operators implemented in the standard versions of their libraries, I needed to implement my own operators for the 3D fault-focusing CNN used in Chapter 4 of this thesis and shown in Figure 4.21. While it is entirely possible to write a custom operator that makes use of 4D kernels, I chose to instead express the 4D cross-correlation as a sum of the output of several 3D cross-correlations. This enabled me to use existing optimized kernels available within PyTorch and cuDNN with relatively little wrapper Python code to implement the summation over the kernels. To illustrate this computation of high-dimensional cross-correlations using low-dimensional kernels, I present a simple example where the filter and image are both 2D and have the dimensions of 3×3 and I use a summation of 1D cross-correlations in order to compute the 2D cross-correlation. While this is a simplification of the sum of 3D kernels used in Chapter 4, the goal is to provide the reader with an intuition behind this approach.

$$y[m,n] = x \star w = \sum_{j=-\infty}^{\infty} \sum_{i=-\infty}^{\infty} x[m+i, n+j] \cdot w[i,j] \tag{C.1}$$

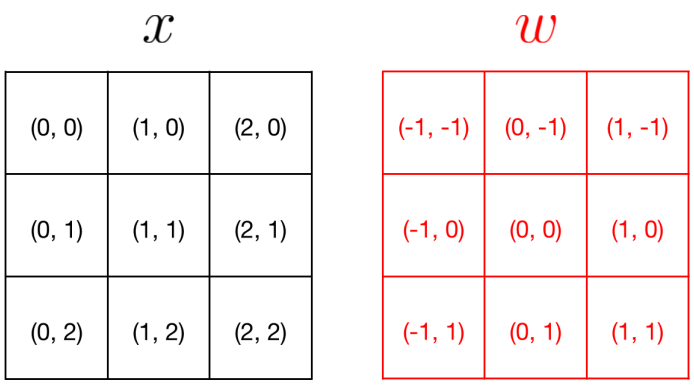


Figure C.1: Indices during 2D cross-correlation of the input image x (left) and the kernel w (right). [**NR**]

For a 2D image x and kernel w, the 2D discrete cross-correlation can be expressed as: where $y[m, n]$ is the output of the cross-correlation at the mth column and the nth row and j and i are summation indices for the rows and columns respectively. Figure C.2 shows the indices of the grids for the image and the kernel. Using Equation C.1 and using assuming a dimension of 3×3 for both the image and the kernel, the first row of the output can be expressed as:

$$\begin{aligned} y[0,0] &= \sum_j \sum_i x[i,j] \cdot w[i,j] \\ &= x[-1,-1] \cdot w[-1,-1] + x[0,-1] \cdot w[0,-1] + x[1,-1] \cdot w[1,-1] \\ &+ x[-1,0] \cdot w[-1,0] + x[0,0] \cdot w[0,0] + x[1,0] \cdot w[1,0] \\ &+ x[-1,1] \cdot w[-1,1] + x[0,1] \cdot w[0,1] + x[1,1] \cdot w[1,1], \end{aligned} \tag{C.2}$$

$$\begin{aligned} y[1,0] &= \sum_j \sum_i x[i+1,j] \cdot w[i,j] \\ &= x[0,-1] \cdot w[-1,-1] + x[1,-1] \cdot w[0,-1] + x[2,-1] \cdot w[1,-1] \\ &+ x[0,0] \cdot w[-1,0] + x[1,0] \cdot w[0,0] + x[2,0] \cdot w[1,0] \\ &+ x[0,1] \cdot w[-1,1] + x[1,1] \cdot w[0,1] + x[2,1] \cdot w[1,1], \end{aligned} \tag{C.3}$$

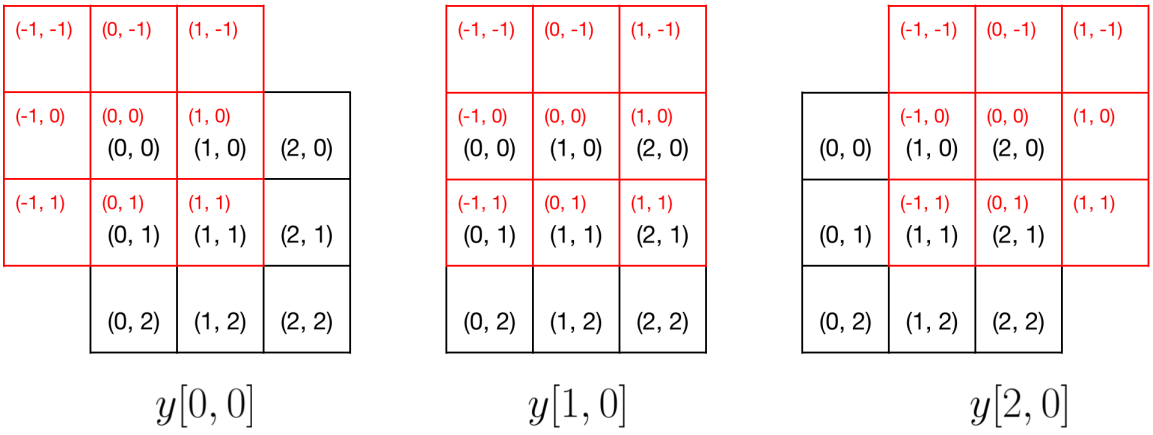


Figure C.2: Visualization of 2D cross correlations for computing the first row (index 0) of the output image. The black grid represents the input image (x) and the red grid indicates the kernel (w). [**NR**]

$$
\begin{aligned}
y[2,0] &= \sum_j \sum_i x[i+2, j] \cdot w[i, j] \\
&= x[1,-1] \cdot w[-1,-1] + x[2,-1] \cdot w[0,-1] + x[3,-1] \cdot w[1,-1] \\
&+ x[1,0] \cdot w[-1,0] + x[2,0] \cdot w[0,0] + x[3,0] \cdot w[1,0] \\
&+ x[1,1] \cdot w[-1,1] + x[2,1] \cdot w[0,1] + x[3,1] \cdot w[1,1]. \qquad \text{(C.4)}
\end{aligned}
$$

Each of these 2D cross-correlations is graphically shown in Figure C.2. Observing the relative position of the kernel and the image, it is apparent that only the first two rows of the image and the last two rows of the kernel contribute to these output values. Additionally, considering each of the rows of both the kernel and the image independently, it is apparent in the figure that the output of the first row can be computed by cross-correlating the rows in a 1D fashion and then computing the sum of these 1D cross-correlations. Mathematically, this can be expressed as follows the result of cross-correlating the first row of the image with the second row of the kernel can be expressed as:

$$
v_1[0] = \sum_i x[i+0, 0] \cdot w[i, 0] = x[-1,0] \cdot w[-1,0] + x[0,0] \cdot w[0,0] + x[1,0] \cdot w[1,0],
$$

$$v_1[1] = \sum_i x[i+1,0] \cdot w[i,0] = x[0,0] \cdot w[-1,0] + x[1,0] \cdot w[0,0] + x[2,0] \cdot w[1,0],$$

$$v_1[2] = \sum_i x[i+2,0] \cdot w[i,0] = x[1,0] \cdot w[-1,0] + x[2,0] \cdot w[0,0] + x[3,0] \cdot w[1,0],$$

where v_1 is a temporary vector to store the output of the 1D cross-correlation. Similarly, the cross-correlation of of the second row of the image with the third row of the kernel can be expressed as

$$v_2[0] = \sum_i x[i+0,1] \cdot w[i,1] = x[-1,1] \cdot w[-1,1] + x[0,1] \cdot w[0,1] + x[1,1] \cdot w[1,1],$$

$$v_2[1] = \sum_i x[i+1,1] \cdot w[i,1] = x[0,1] \cdot w[-1,1] + x[1,1] \cdot w[0,1] + x[2,1] \cdot w[1,1],$$

$$v_2[2] = \sum_i x[i+2,1] \cdot w[i,1] = x[1,1] \cdot w[-1,1] + x[2,1] \cdot w[0,1] + x[3,1] \cdot w[1,1],$$

where v_2, like v_1 is a temporary vector to store the output of the 1D cross-correlation. Now, computing the element-wise addition of v_1 and v_2, results in the following:

$$\begin{aligned} v_1[0] + v_2[0] = {} & x[-1,0] \cdot w[-1,0] + x[0,0] \cdot w[0,0] + x[1,0] \cdot w[1,0] \\ & + x[-1,1] \cdot w[-1,1] + x[0,1] \cdot w[0,1] + x[1,1] \cdot w[1,1], \end{aligned} \tag{C.5}$$

$$\begin{aligned} v_1[1] + v_2[1] = {} & x[0,0] \cdot w[-1,0] + x[1,0] \cdot w[0,0] + x[2,0] \cdot w[1,0] \\ & + x[0,1] \cdot w[-1,1] + x[1,1] \cdot w[0,1] + x[2,1] \cdot w[1,1], \end{aligned} \tag{C.6}$$

$$\begin{aligned} v_1[2] + v_2[2] = {} & x[1,0] \cdot w[-1,0] + x[2,0] \cdot w[0,0] + x[3,0] \cdot w[1,0] \\ & + x[1,1] \cdot w[-1,1] + x[2,1] \cdot w[0,1] + x[3,1] \cdot w[1,1]. \end{aligned} \tag{C.7}$$

Comparing Equations C.2, C.3 and C.4 with C.5, C.6, C.7 it is apparent that apart from the terms associated with -1 row index of the image x, the expressions are equivalent. However, if I assume a zero-value boundary condition for pixels outside of the image (as they are implemented numerically), these values do not contribute to the output cross-correlation and therefore the results of the 2D cross-correlation are indeed equivalent to the result of using the 1D kernels.

While the result above is only a single example of how to split a 2D kernel into 1D kernels and then accumulate the results of the 1D kernels, it can be generalized to higher-dimensional examples and outlines a general approach that can be used for any example. The general idea is to split the high-dimensional kernel along the "slow-axis" (in the case above, along the rows) and then to determine the extent of the samples within the image that are correlated with 1D kernels. This can be determined by realizing that for each 1D correlation between a row extracted at index i from a 3×3 kernel, and a row from an image extracted at index j, the output row index of the cross-correlation will be $k = j - i$. This entire process is illustrated in Figure C.3 where each panel shows the areas of overlap of the image for each row of the kernel. 1D cross-correlations are performed for each of the rows of the kernels over each of the colored regions in the image and the results from each correlation are accumulated at the output row index given by k.

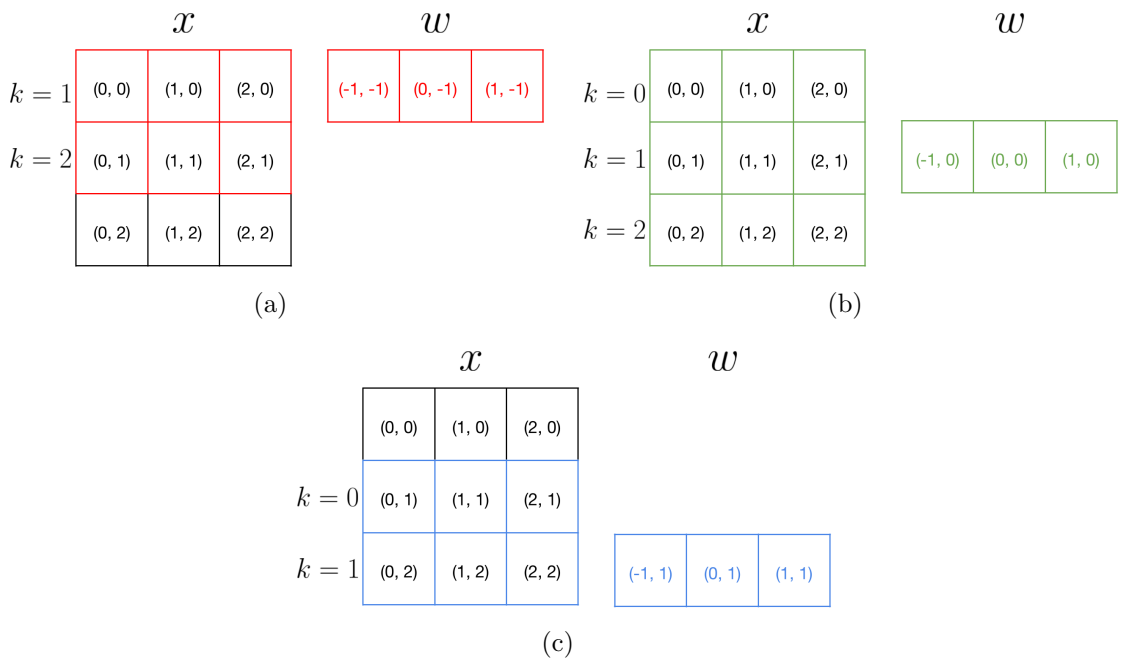


Figure C.3: Regions of overlap of the 2D image for the 1D kernels. (a) The first row of (index of -1) will be correlated with rows zero and one of the image, (b) the middle row of the kernel will correlate with each of the rows zero, one and two of the image and (c), the final row will correlate with the rows one and two of the image. The indices next to the rows of x show the computed output index for the correlation with row the kernel in that panel computed via $k = j - i$, where j is the row index of the image and i is the row index of the kernel. [**NR**]

Bibliography

Abadi, M., A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, 2015, TensorFlow: Large-scale machine learning on heterogeneous systems. (Software available from tensorflow.org).

Alaudah, Y., P. Michałowicz, M. Alfarraj, and G. AlRegib, 2019, A machine-learning benchmark for facies classification: Interpretation, **7**, SE175–SE187.

Araya-Polo, M., J. Jennings, A. Adler, and T. Dahlke, 2018, Deep-learning tomography: The Leading Edge, **37**, 58–66.

Audebert, F., J.-P. Diet, P. Guillaume, I. Jones, and X. Zhang, 1997, Crp-scans: 3d presdm velocity analysis via zero-offset tomographic inversion, *in* SEG Technical Program Expanded Abstracts 1997: Society of Exploration Geophysicists, 1805–1808.

Audebert, F., J.-P. Diet, and X. Zhang, 1996, Crp-scans from 3d pre-stack depth migration: a powerful combination of crp-gathers and velocity scans, *in* SEG Technical Program Expanded Abstracts 1996: Society of Exploration Geophysicists, 515–518.

Bashir, Y., N. Mohd Muztaza, S. Y. Moussavi Alashloo, S. H. Ali, and D. Prasad Ghosh, 2020, Inspiration for seismic diffraction modelling, separation, and velocity in depth imaging: Applied Sciences, **10**, 4391.

Beasley, C. J., W. Lynn, K. Larner, and H. Nguyen, 1988, Cascaded fk migration:

Removing the restrictions on depth-varying velocity: Geophysics, **53**, 881–893.

Biondi, B., 2010, Velocity estimation by image-focusing analysis: Geophysics, **75**, U49–U60.

Biondi, B., and G. Palacharla, 1996, 3-d prestack migration of common-azimuth data: Geophysics, **61**, 1822–1832.

Biondi, B. L., 2006, 3d seismic imaging: Society of Exploration Geophysicists.

Bottou, L., F. E. Curtis, and J. Nocedal, 2018, Optimization methods for large-scale machine learning: Siam Review, **60**, 223–311.

Campos Trinidad, M. J., S. W. Arauco Canchumuni, and M. A. Cavalcanti Pacheco, 2021, Towards a benchmark for sedimentary facies classification: Applied to the netherlands f3 block: Information Management and Big Data, Springer International Publishing, 211–222.

Chetlur, S., C. Woolley, P. Vandermersch, J. Cohen, J. Tran, B. Catanzaro, and E. Shelhamer, 2014, cudnn: Efficient primitives for deep learning: arXiv preprint arXiv:1410.0759.

Choy, C., J. Gwak, and S. Savarese, 2019, 4d spatio-temporal convnets: Minkowski convolutional neural networks: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 3075–3084.

Claerbout, J., 2014, Geophysical image estimation by example: Lulu. com.

Claerbout, J. F., 1985, Imaging the earth's interior: Blackwell scientific publications Oxford.

———, 2010, Basic earth imaging: Cambridge: Free Software Foundation.

Clapp, R. G., 2014, Synthetic model building using a simplified basin modeling approach: SEP-Report, **155**, 145–152.

de Groot, P., and B. Bril, 2005, The open source model in geosciences and opendtect in particular, *in* SEG Technical Program Expanded Abstracts 2005: Society of Exploration Geophysicists, 802–805.

De Vries, D., and A. Berkhout, 1984, Velocity analysis based on minimum entropy: Geophysics, **49**, 2132–2142.

DeLaughter, J., M. Harbin, D. Herrington, A. A. Ortega, S. Yoo, T. Kim, W. Whiteside, and C. Reta-Tang, 2014, Tti dit for dirty salt inversion in the mississippi

canyon area, gulf of mexico, *in* SEG Technical Program Expanded Abstracts 2014: Society of Exploration Geophysicists, 4753–4756.

Di, H., D. Gao, and G. AlRegib, 2019, Developing a seismic texture analysis neural network for machine-aided seismic pattern recognition and classification: Geophysical Journal International, **218**, 1262–1275.

Duin, E., J. Doornenbal, R. Rijkers, J. Verbeek, and T. E. Wong, 2006, Subsurface structure of the netherlands-results of recent onshore and offshore mapping: Netherlands Journal of Geosciences, **85**, 245.

Epili, D., G. Cloudy Jr, J. Cai, Q. Zhang, R. Camp, and S. Lopez-Mora, 2011, Improved subsalt imaging using tti anisotropy and reverse time migration scans, *in* SEG Technical Program Expanded Abstracts 2011: Society of Exploration Geophysicists, 243–247.

Erhan, D., Y. Bengio, A. Courville, and P. Vincent, 2009, Visualizing higher-layer features of a deep network: University of Montreal, **1341**, 1.

Faye, J.-P., and J.-P. Jeannot, 1986, Prestack migration velocities from focusing depth analysis, *in* SEG Technical Program Expanded Abstracts 1986: Society of Exploration Geophysicists, 438–440.

Fehmers, G. C., and C. F. Höcker, 2003, Fast structural interpretation with structure-oriented filtering: Geophysics, **68**, 1286–1293.

Fomel, S., 2007, Shaping regularization in geophysical-estimation problems: Geophysics, **72**, R29–R36.

———, 2009, Velocity analysis using ab semblance: Geophysical Prospecting, **57**, 311–321.

Fomel, S., E. Landa, and M. T. Taner, 2007, Poststack velocity analysis by separation and imaging of seismic diffractions: Geophysics, **72**, U89–U94.

Gao, H., X. Wu, and G. Liu, 2020, Channel simulation and deep learning for channel interpretation in 3d seismic images: Presented at the SEG International Exposition and Annual Meeting, OnePetro.

Gretton, A., A. Smola, J. Huang, M. Schmittfull, K. Borgwardt, and B. Schölkopf, 2009, *in* Covariate shift and local learning by distribution matching: MIT Press, 131–160.

Guo, Q., N. Islam, and W. D. Pennington, 2014, Tuning, avo, and flatspot effects in north sea block f3, *in* SEG Technical Program Expanded Abstracts 2014: Society of Exploration Geophysicists, 538–542.

Hale, D., 2009, Structure-oriented smoothing and semblance: CWP report, **635**.

Hale, D., and R. H. Groshong Jr, 2014, Conical faults apparent in a 3d seismic image: Interpretation, **2**, T1–T11.

Harlan, W. S., J. F. Claerbout, and F. Rocca, 1984, Signal/noise separation and velocity estimation: Geophysics, **49**, 1869–1880.

Hoffmann, D. T., D. Tzionas, M. J. Black, and S. Tang, 2019, Learning to train with synthetic humans: German conference on pattern recognition, Springer, 609–623.

Jaccard, P., 1912, The distribution of the flora in the alpine zone. 1: New phytologist, **11**, 37–50.

Keskar, N. S., D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang, 2016, On large-batch training for deep learning: Generalization gap and sharp minima: arXiv preprint arXiv:1609.04836.

Kessinger, W., 1992, Extended split-step fourier migration, *in* SEG Technical Program Expanded Abstracts 1992: Society of Exploration Geophysicists, 917–920.

Kingma, D. P., and J. Ba, 2014, Adam: A method for stochastic optimization: arXiv preprint arXiv:1412.6980.

Kleinberg, B., Y. Li, and Y. Yuan, 2018, An alternative view: When does sgd escape local minima?: International Conference on Machine Learning, PMLR, 2698–2707.

Krizhevsky, A., I. Sutskever, and G. E. Hinton, 2012, Imagenet classification with deep convolutional neural networks: Advances in neural information processing systems, **25**, 1097–1105.

Kuhlmann, G., and T. E. Wong, 2008, Pliocene paleoenvironment evolution as interpreted from 3d-seismic data in the southern north sea, dutch offshore sector: Marine and Petroleum Geology, **25**, 173–189.

Lanfranchi, P., L. Langlo, A. Aamodt, and P. Guillaume, 1996, Model building using 3d tomographic inversion of multi-arrivals-a north sea case study: 58th EAGE Conference and Exhibition, European Association of Geoscientists & Engineers, cp–48.

Lin, C.-Y., Y.-C. Chiu, H.-F. Ng, T. K. Shih, and K.-H. Lin, 2020, Global-and-local context network for semantic segmentation of street view images: Sensors, **20**, 2907.

Liu, J., P. Devarakota, C. Sutton, S. Ye, and P. Webster, 2021, Automatch: A fully automated adaptive subtraction driven by artificial intelligence: First International Meeting for Applied Geoscience & Energy, Society of Exploration Geophysicists, 1360–1363.

Liu, W., D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, 2016, Ssd: Single shot multibox detector: European conference on computer vision, Springer, 21–37.

Ma, X., B. Wang, C. Reta-Tang, W. Whiteside, and Z. Li, 2011, Enhanced prestack depth imaging of wide-azimuth data from the gulf of mexico: A case history: Geophysics, **76**, WB79–WB86.

MacKay, S., and R. Abma, 1989, Refining prestack depth-migration images without remigration, *in* SEG Technical Program Expanded Abstracts 1989: Society of Exploration Geophysicists, 1258–1261.

———, 1992, Imaging and velocity estimation with depth-focusing analysis: Geophysics, **57**, 1608–1622.

Mahendran, A., and A. Vedaldi, 2016, Visualizing deep convolutional neural networks using natural pre-images: International Journal of Computer Vision, **120**, 233–255.

Montazeri, M., L. O. Boldreel, A. Uldall, and L. Nielsen, 2020, Improved seismic interpretation of a salt diapir by utilization of diffractions, exemplified by 2d reflection seismics, danish sector of the north sea: Interpretation, **8**, T77–T88.

Murez, Z., S. Kolouri, D. Kriegman, R. Ramamoorthi, and K. Kim, 2018, Image to image translation for domain adaptation: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 4500–4509.

Murphy, K. P., 2012, Machine learning: a probabilistic perspective: MIT press.

Négron, J., P. Perfetti, F. Audebert, P. Guillaume, and X. Zhang, 2000, 3d pre-stack velocity estimation: A 4d horizon-consistent method for interpretation of crp-scans to guide tomographic inversion, *in* SEG Technical Program Expanded Abstracts 2000: Society of Exploration Geophysicists, 930–933.

Neidell, N. S., and M. T. Taner, 1971, Semblance and other coherency measures for multichannel data: Geophysics, **36**, 482–497.

Nguyen, A., J. Yosinski, and J. Clune, 2019, Understanding neural networks via feature visualization: A survey, *in* Explainable AI: interpreting, explaining and visualizing deep learning: Springer, 55–76.

NVIDIA Corporation, 2015, Gpu-based deep learning inference: A performance and power analysis: Technical report, NVIDIA Corporation.

Paffenholz, J., B. McLain, J. Zaske, and P. J. Keliher, 2002, Subsalt multiple attenuation and imaging: Observations from the sigsbee2b synthetic dataset, *in* SEG Technical Program Expanded Abstracts 2002: Society of Exploration Geophysicists, 2122–2125.

Paszke, A., S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, 2019, Pytorch: An imperative style, high-performance deep learning library, *in* Advances in Neural Information Processing Systems 32: Curran Associates, Inc., 8024–8035.

Perlin, K., 1985, An image synthesizer: ACM Siggraph Computer Graphics, **19**, 287–296.

Pham, N., S. Fomel, and D. Dunlap, 2019, Automatic channel detection using deep learning: Interpretation, **7**, SE43–SE50.

Remmelts, G., 1996, *in* Salt tectonics in the southern North Sea, the Netherlands: Springer Netherlands, 143–158.

Ronneberger, O., P. Fischer, and T. Brox, 2015, U-net: Convolutional networks for biomedical image segmentation: International Conference on Medical image computing and computer-assisted intervention, Springer, 234–241.

Rothman, D. H., S. A. Levin, and F. Rocca, 1985, Residual migration: Applications and limitations: Geophysics, **50**, 110–126.

Sava, P., and B. Biondi, 2004a, Wave-equation migration velocity analysis. i. theory: Geophysical Prospecting, **52**, 593–606.

———, 2004b, Wave-equation migration velocity analysis. ii. subsalt imaging examples: Geophysical Prospecting, **52**, 607–623.

Sava, P. C., 2003, Prestack residual migration in the frequency domain: Geophysics, **68**, 634–640.

Sava, P. C., B. Biondi, and J. Etgen, 2005, Wave-equation migration velocity analysis by focusing diffractions and reflections: Geophysics, **70**, U19–U27.

Sava, P. C., and S. Fomel, 2003, Angle-domain common-image gathers by wavefield continuation methods: Geophysics, **68**, 1065–1074.

Schroot, B., and R. Schüttenhelm, 2003, Expressions of shallow gas in the netherlands north sea: Netherlands Journal of Geosciences, **82**, 91–105.

Selvaraju, R. R., M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, 2017, Grad-cam: Visual explanations from deep networks via gradient-based localization: Proceedings of the IEEE international conference on computer vision, 618–626.

Shrivastava, A., T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb, 2017, Learning from simulated and unsupervised images through adversarial training: Proceedings of the IEEE conference on computer vision and pattern recognition, 2107–2116.

Silva, R. M., L. Baroni, R. S. Ferreira, D. Civitarese, D. Szwarcman, and E. V. Brazil, 2019, Netherlands dataset: A new public dataset for machine learning in seismic interpretation: arXiv preprint arXiv:1904.00770.

Simonyan, K., A. Vedaldi, and A. Zisserman, 2014, Deep inside convolutional networks: Visualising image classification models and saliency maps, *in* Workshop at International Conference on Learning Representations.

Simonyan, K., and A. Zisserman, 2014, Very deep convolutional networks for large-scale image recognition: arXiv preprint arXiv:1409.1556.

Smith, M., N. Morrison, and R. Knight, 1989, Marine crew supervisory report on behalf of nam holland bv north sea dutch sector block fo3: Technical Report 89/0632, Exploration Consultants Limited.

Song, C., Z. Liu, H. Cai, Y. Wang, X. Li, and G. Hu, 2017, Unsupervised seismic facies analysis with spatial constraints using regularized fuzzy c-means: Journal of Geophysics and Engineering, **14**, 1535–1543.

Stolt, R. H., 1978, Migration by fourier transform: Geophysics, **43**, 23–48.

———, 1996, A prestack residual time migration operator: Geophysics, **61**, 605–607.

Sylvester, Z., P. Durkin, and J. A. Covault, 2019, High curvatures drive river meandering: Geology, **47**, 263–266.

Tachon, R., C. Thomas, T. Davidson, and M. King, 1989, Data processing report for north sea block f-03 3d seismic survey 1989 processed on behalf of nederlande aardolie maatschappij: Technical report, Western Geophysical Company of America.

Taigman, Y., M. Yang, M. Ranzato, and L. Wolf, 2014, Deepface: Closing the gap to human-level performance in face verification: Proceedings of the IEEE conference on computer vision and pattern recognition, 1701–1708.

Tschannen, V., M. Delescluse, N. Ettrich, and J. Keuper, 2020, Extracting horizon surfaces from 3d seismic data using deep learning: Geophysics, **85**, N17–N26.

Vermeer, G. J., 2012, 3d seismic survey design: Society of Exploration Geophysicists.

Waldeland, A. U., A. C. Jensen, L.-J. Gelius, and A. H. S. Solberg, 2018, Convolutional neural networks for automated seismic interpretation: The Leading Edge, **37**, 529–537.

Wang, B., V. Dirks, P. Guillaume, F. Audebert, and D. Epili, 2006, A 3d subsalt tomography based on wave-equation migration-perturbation scans: Geophysics, **71**, E1–E6.

Wang, B., Y. Kim, C. Mason, and X. Zeng, 2008, Advances in velocity model-building technology for subsalt imaging: Geophysics, **73**, VE173–VE181.

Wang, B., C. Mason, M. Guo, K. Yoon, J. Cai, J. Ji, and Z. Li, 2009, Subsalt velocity update and composite imaging using reverse-time-migration based delayed-imaging-time scan: Geophysics, **74**, WCA159–WCA166.

Wang, X., R. Girshick, A. Gupta, and K. He, 2018, Non-local neural networks: Proceedings of the IEEE conference on computer vision and pattern recognition, 7794–7803.

Weickert, J., 1998, Anisotropic diffusion in image processing: Teubner Stuttgart, **1**.

Whiteside, W., Z. Guo, and B. Wang, 2011, Automatic rtm-based dit scan picking for enhanced salt interpretation, *in* SEG Technical Program Expanded Abstracts 2011: Society of Exploration Geophysicists, 3295–3299.

Wrona, T., I. Pan, R. E. Bell, R. L. Gawthorpe, H. Fossen, and S. Brune, 2021, 3d seismic interpretation with deep learning: A brief introduction: The Leading Edge,

40, 524–532.

Wu, X., Z. Geng, Y. Shi, N. Pham, S. Fomel, and G. Caumon, 2020, Building realistic structure models to train convolutional neural networks for seismic structural interpretation: Geophysics, **85**, WA27–WA39.

Wu, X., L. Liang, Y. Shi, and S. Fomel, 2019, Faultseg3d: Using synthetic data sets to train an end-to-end convolutional neural network for 3d seismic fault segmentation: Geophysics, **84**, IM35–IM45.

Xie, J., W. Whiteside, and B. Wang, 2012, Remove rtm artifacts using delayed imaging time gathers: Presented at the 2012 SEG Annual Meeting, OnePetro.

Yilmaz, Ö., 2001, Seismic data analysis: Society of exploration geophysicists Tulsa, **1**.

Yilmaz, O., and R. Chambers, 1984, Migration velocity analysis by wave-field extrapolation: Geophysics, **49**, 1664–1674.

Zhang, Q., I. Chang, and L. Li, 2009, Salt interpretation for depth imaging-where geology is working in the geophysical world, *in* SEG Technical Program Expanded Abstracts 2009: Society of Exploration Geophysicists, 3660–3664.

Zhang, Q., L. Geiger, C. Reta-Tang, S. Hightower, S. Yang, and W. Gao, 2013, New views on salt tectonics and salt modeling using the latest seismic imaging technologies, for subsalt and presalt exploration in the mississippi canyon, gulf of mexico: Gulf Coast of Geological Societies Transactions, **63**, 637–639.

Zhao, T., C. Zhao, A. Kaul, and A. Abubakar, 2021, Automatic salt geometry update using deep learning in iterative fwi-rtm workflows: First International Meeting for Applied Geoscience & Energy, Society of Exploration Geophysicists, 3184–3188.

Zhou, B., A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, 2016, Learning deep features for discriminative localization: Proceedings of the IEEE conference on computer vision and pattern recognition, 2921–2929.

Zhou, Z., M. M. Rahman Siddiquee, N. Tajbakhsh, and J. Liang, 2018, Unet++: A nested u-net architecture for medical image segmentation: Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support, Springer International Publishing, 3–11.

Zima, L., 2003, Upstream joint ventures and the new dutch mining law; upstream joint

ventures en de nieuwe mijnbouwwet: Nederlands Tijdschrift voor Energierecht, **2**.