Shifting Paradigms: Data-Centric Approach for Marine Statics Correction using Symmetric Autoencoding

by

Brindha Kanniah

Submitted to the Department of Earth, Atmospheric and Planetary Sciences
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2024

Authored by: .....

Brindha Kanniah
Department of Earth, Atmospheric and Planetary Sciences
August 20, 2024

Certified by .....

Laurent Demanet
Director of Earth Resources Laboratory (ERL) Professor of Applied Mathematics
Thesis Supervisor

Accepted by .....

Robert D. van der Hilst
Schlumberger Professor of Earth and Planetary Sciences
Department Head

Shifting Paradigms: Data-Centric Approach for Marine Statics Correction using Symmetric Autoencoding

by

Brindha Kanniah
Submitted to the Department of Earth, Atmospheric, and Planetary Sciences
on August 20, 2024, in Partial Fulfillment of the Requirements for
the Degree of Doctor of Philosophy

Abstract

Deep learning has demonstrated remarkable performance in a wide variety of domains and is often leveraged for making high-stakes decisions. Parallel to its growing and beneficial presence in other domains, deep learning is gaining a notable reputation for solving challenging problems in geophysics. A key problem - given the escalating energy and geosequestration demands in present times - is marine statics correction. The traditional workflow for correcting marine statics has been based on a model-centric paradigm. This paradigm involves a series of transformations between non-commensurate spaces: first, inversion from seismic data space to velocity model space and second, forward modeling from velocity model space to seismic data space. Statics correction within this paradigm has severe drawbacks, mainly the high compute, time and labor cost, and inaccuracies stemming from errors in velocity model inversion or from unmet assumptions about subsurface structure. Overcoming these drawbacks was thus, the prime motivation for our study - where we chose to leverage deep learning as the core algorithmic tool to understand the limits of the model-centric paradigm and explore the performance horizons of a different, data-centric, paradigm to statics correction. The main feature of the data-centric paradigm is the direct mapping between commensurate data spaces, eliminating the need for intermediary transformations to and from velocity model space. Initial benchmark tests on the model-centric approach revealed the impact of inaccuracies in velocity model inversion as substantial non-zero timeshifts - exceeding 0.01s, and reaching values as large as 0.04s - for most arrivals in seismic data. These arrival time precision levels are unacceptable for good seismic imaging and time-lapse analysis; underscoring the need for an improved approach to marine statics correction. Consequently, we began our investigations into the data-centric paradigm. With the focus of disentangling the effects of varying seawater velocity from coherent subsurface geology in seismic records, we implemented an autoencoder algorithm, named SymAE. Notably, SymAE leverages the permutation symmetry of coherent subsurface information to perform the separation of information from nuisance variations. Once trained, SymAE is able to redatum selected subsurface and water velocity information in its latent space to produce statics-corrected seismic records. Our results show that for training datasets of increasing subsurface complexity, SymAE strongly converges all dynamic timeshifts to zero, aligning perturbed traces to reference traces. Crucially, SymAE delivers the required timeshift precision of 0.01 seconds for all arrivals - an achievement that the model-centric approach falls short of. This notable precision improvement using SymAE highlights how a streamlined data-centric paradigm outperforms the traditional model-centric paradigm of marine statics correction. This finding is pivotal as it is the foundation that lays the groundwork and opens the path towards the real-world deployment of SymAE for statics correction in challenging deepwater environments.

Acknowledgements

I would not be here without the kindness, generosity, and support of so many people, whom I'd like to thank. First, I am deeply grateful to my advisor, Professor Laurent Demanet. Thank you for believing in me and truly encouraging me to pursue my passion for research and science. I truly appreciate your kind and patient mentorship. I have especially enjoyed the freedom of having stimulating and insightful conversations with you on all topics,including geophysics and neural networks of course. I will always be grateful that you took me under your wing in my first year of the PhD program - and have since then, guided me towards a deep and fruitful experiential understanding of what true scientific research is like.

I am grateful to my committee members, Professor Julien de Wit, Professor George Barbastathis, and Professor Robert van der Hilst, for all your encouragement, advice and support throughout grad school. I am delighted that I had the chance to take classes taught by both Professor Julien de Wit and Professor Robert van der Hilst, who are both brilliant stars in their respective fields of earth and planetary sciences. I fondly remember writing my essay, The Meaning of Life and Being Alive Through a Cosmological Lens, which explored relationships between scales of consciousness, life and space in Professor de Wit's signature class, The Space Odyssey. Till date, it is one of my favourite classes at MIT, with fascinating speakers and topics brought together in a spring semester. I will always admire and hope to emulate the freedom for exploration and independent thinking that Professor de Wit encourages his students to possess.

I am also indebted to Professor Pawan Bharadwaj at the Indian Institute of Science, who spent a generous amount of time, remotely guiding me on the works and implementation of the deep learning model, SymAE. His insight and expertise was invaluable for successfully conducting the experiments in this thesis.

I would like to thank my fellow PhD mates, without whom this journey would not have been as joyful, fun, and meaningful as it has been. First to my office-mate Hillary Chang, thank you for being a ray of sunshine that I get to meet in the office (almost) every day. I would also like to thank Caroline and Mathilde from the 4th floor office for being my

best friends, and for all the fun adventures we had together, both in and out of the office. I will cherish our great and simple moments of being silly and dancing together.

I want to thank my family that live across the world in Malaysia for all their love and support throughout my life. Appa, Amma and Geetha, thank you for being by my side always and reminding me, that you are only a flight ticket away.

Lastly, I want to thank MIT, for being my intellectual home for the last ten years. It was my dream to come to MIT as a young girl from Malaysia; and now, as I complete my PhD, I will take all the lessons, experiences and memories from MIT, and will treasure them for a lifetime. For me, the best part of MIT has always been the people - and I have been blessed to have been surrounded by the most multi-faceted, sparkling and scintillating group of people I could have ever imagined meeting in my lifetime. I hope to do MIT proud in the future as I step into the world, that is beyond our little paradise.

Contents

Acknowledgements **4**

1 Introduction **17**

1.1 Motivation 17

1.2 A Summary on Time-Lapse Seismic Imaging 20

1.3 Paradigm Shift from a Model-Centric to a Data-Centric Approach 22

1.4 Precisions Relating Velocity Model to Timeshifts 27

1.5 Review of Core Concepts 31

1.5.1 Static Correction 31

1.5.2 Forward Model 37

1.5.3 Inverse Model 40

1.5.4 Deep Learning 41

2 Model-Centric Benchmark Test **46**

2.1 Deep Learning Approach 46

2.2 Data Preparation and Implementation 48

2.3 Results 50

2.4 Discussion 55

2.5 Conclusion 57

3 Single Horizontal Reflector **61**

3.1 Introduction 61
3.1.1 SymAE and Marine Seismic Survey Non-Repeatability 62
3.2 Data and Methods 64
3.2.1 Data 64
3.2.2 Method 68
3.2.3 Training SymAE 70
3.3 Results 72
3.4 Discussion 75
3.5 Conclusion 78

4 Interface Model **83**

4.1 Data 83
4.2 Hyperparameter Tuning 88
4.3 Results 91
4.4 Discussion 97
4.5 Conclusion 102
4.6 Future Directions 102

5 Conclusions and Future Directions **108**

5.1 Future Directions 110

6 Appendix **113**

6.1 Seismic Phantom Generator 113

List of Figures

1 Average global surface temperature change from 1880 to 2024. The bars indicate the deviation from the 20th century average temperature, taken between 1901 to 2000. Blue bars show cooler than average temperature, and red bard indicate warmer than average temperatures. The data shows a an accelerated warming trend in the 20th centure, with the year 2023 being the warmest year since records began in 1850 at 1.18°C above the 20th century average of 13.9°C. Image from [Administration and Atmospheric, 2023]. 18

2 Summary of paradigm changes in geophysics research, with details pertaining to our focus in marine statics correction. The arrows between the headers indicate the direction for the shifting paradigms, from classical non deep-learning (DL), to model-centric deep learning and lastly, to data-centric deep learning. 23

3 Standard static correction till the 21st century: most near-surface timeshifts are removed by applying datum static correction. The datum in this case is the horizontal plane, which functions as a reference plane, on which a pseudo source and receiver are positioned vertically below the original source and receiver located on the surface of the weathered layer. Image from [Cox, 2000]. 33

4 Two-layer velocity model used to show the variations in distance traveled by the ray propagating from a source and arriving at a receiver, with offset distance x. Consider the case where the near-surface layer has a depth, Z_1, of 500m, and a reflector depth, $Z_1 + Z_2$ of 1500m. The travel path length in the near surface layer is labeled as d. Image from [Cox, 1999]. . 33

5 Different travel path lengths as a function of source-receiver offset, x, and velocity ratio between the deeper layer V_1 and near-surface V_0, $\frac{V_1}{V_0}$, whereby the near-surface is assumed to be a water layer with velocity of 1500m/s. Notice that the travel path length d increases with increasing offset x, from approximately 95m to 205m, in correspondence to velocity ratios ranging between 3.00 to 1.25. Larger velocity ratios lead to smaller angles of incidence at the interface between near-surface and deeper layer, hence a smaller distance traveled in the near-surface layer. In conclusion, deviations in ray path distance and in velocity structure of the traveled region exist in practical seismic recordings, which ultimately result in travel time perturbations that are different from the vertical raypath assumption of datum statics correction. Thus, statics correction is a dynamic process that is offset and depth dependent. Image from [Cox, 1999]. 34

6 A diagrammatic representation of the Multilayer Perceptron (MLP) network. This network possesses an input layer with nine nodes, three hidden layers each containing 13 nodes, and an output layer with five nodes. The arrows between the nodes indicate the weights connecting nodes from one layer to the next layer. The transparency of these arrow signifies the strength of the weight. 42

7 Architectural overview of the autoencoder algorithm. X represents the input data of the input layer, Z represents the extracted codes in the latent space, and X represents the reconstructed output data of the output layer. The encoder performs the non-linear downsampling of data from X to Z, and the decoder performs the nonlinear upsampling of latent codes from Z to X'. Image edited from [Berahmand et al., 2024] 45

8 A simplified overview of the neural network architecture designed to infer the water velocity profile from the seismic record of a wave propagating through the upper water layer and geological subsurface beneath. This architecture is an amalgamation of various deep learning algorithms. These algorithms - namely U-NET, MLP, GRU and CNN - have previously shown their capability in successfully performing the inversion from seismic data space to velocity model space [Araya-polo et al., 2018; Adler et al., 2019; Gelboim et al., 2023]. 48

9 Inversion results using our specially designed neural network, described in Section 2.1 The network inverts for water velocity profiles of the 2000m deep ocean layer, utilizing shot records from Ocean Bottom Nodes located on the seafloor. The annotated percentage value indicate the relative amplitude modulations (perturbations) to a Reference Hood's water velocity profile, which is detailed further in Section 3.2.1. The network successfully captures the curvature of various perturbed profiles in the top 1000m of the ocean, and the overlapping monotonically increasing profiles with depth,in the lower 1000m. Overall, the network predictions have a RMSE of 0.40894 compared to ground truth velocity profiles. 51

10 NMO-based forward model results for seismic records with distinct velocity models, consisting of the water layer and underlying subsurface geology. To evaluate the water velocity predictions, we NMO-processed the predicted profiles to seismic records, and calculated the difference between the predicted and ground truth records. **Column 1:** Ground truth and predicted depth-dependent water velocity profiles for three different velocity perturbations, with their respective RMS velocity values indicated by dashed lines. **Column 2:** Ground truth seismic records, from which the predicted water velocity profiles in Column 1 are inverted from. **Column 3:** NMO-based forward modeled seismic records. **Column 4:** Predicted minus ground truth seismic records (their difference). 59

11 Timeshifts measured on ground truth vs predicted traces of the 5.987% perturbed water velocity sample from the test dataset. The traces shown are for the zero offset and far offset (3.25km) cases. Although the zero offset traces show better alignment than the far offset traces, most timeshifts for the trace pairs are non-zero and substantial, reaching values as large as 0.04s. This highlights the impact of inversion inaccuracies in the velocity model, which create traveltime inaccuracies in the forward modeled seismic data. 60

12 Separation of time-scales and associated information. Coherent information (marker shapes) tied to the physical state varies on a much slower scale compared to the nuisance information (marker distance from the baseline) which corrupts measurements of the physical state. Each measurement taken is represented as a marker in the diagram, and its disentanglement of coherent and nuisance information is the aim of SymAE. Image from [Bharadwaj et al., 2022]. 63

13 Visualization of a model used to create synthetic shot gather data. V_p here refers to the acoustic wave propagation velocity through the domain. The source, marked with a red x, is positioned in the water column at a depth of 10m. An array of equally-spaced receivers, marked with blue triangles, are positioned on the seafloor at a depth of 2000m. The seawater column appears to be vertically homogenous because the depth-dependent water velocity profile does not vary drastically enough for a visible change in the plot. However, as shown in Section 3.2.1, the modeled water velocity does indeed change with depth through the domain. 65

14 **Top-left:** Acoustic velocity profile of water column in the Gulf Of Mexico derived from remotely operated vehicle (ROV) temperature-pressure-conductivity measurements - from September 2011 to January 2012. Image from Wang et al. [2012]. **Top-right:** Hood's velocity (reference) profile and limits of realistic perturbations. The minimum and maximum velocities corresponds to p values (see Equation 17) of -2% and $+2\%$ respectively. **Bottom panel:** Supplementary perturbation ranges of the Hood's velocity profile used for testing of SymAE: $-6\% \leq p \leq +6\%$ and $-25\% \leq p \leq +50\%$. 67

15 Macro-view of the SymAE learning algorithm network architecture. Total information content of instances of a datapoint are represented as solid coloured arrows. The summation function in the Symmetric Encoder extracts coherent information (solid black arrow) and the Dropout Masks applied to the output of the Nuisance Encoder (NEnc) enforces the extraction of *remaining* nuisance information (dotted colored arrows). The disentangled information is combined and decoded (via Fuse) to reproduce the input datapoint. Image from [Bharadwaj et al., 2022]. 69

16 Auxiliary shot gather resulting from a subsurface model with the Hood's water velocity profile (reference velocity). Its nuisance code is a low-dimensional representation of the Hood's profile and therefore, is used by SymAE to redatum shot gathers with a velocity profile that is perturbed from the Hood's profile. 73

17 Redatuming results for different water velocity perturbation instances. The topmost row of the second column shows the reference shot gather; this shot gather arises from the same subsurface geology as the perturbed instances, however its water column follows the Hood's velocity profile. Thus, the aim of redatuming is to map perturbed instances to reference instance. SymAE performs well on the $+1.914\%$ and -5.5% perturbations; but its performance declines at a much higher perturbation of $+50.00\%$, where it displays comprised ability in capturing the 2 and 4 arrivals. **Column 1:** Perturbed instances; top-to-bottom row sampled from the $[-2, +2]$, $[-6, +6]$, $[-25, +50]$ datasets respectively. **Column 2:** Corresponding redatumed instances from perturbed instances of Column 1.**Column 3:** Residuals between redatumed and reference instance. **Column 4:** Residuals between redatumed and perturbed instances. . . . 80

18 Results of redatuming the $+6\%$ perturbed instance of test datapoint X_{901}. The timeshifts compared to reference trace, before and after redatuming, for zero offset and far offset are shown. All timeshifts *reach* zero. 81

19 Results of redatuming the $+6\%$ perturbed instance of another test datapoint, X_{906}, with different subsurface velocities than X_{901}. The timeshifts compared to reference trace, before and after redatuming, for zero offset and far offset are shown. All timeshifts *converge* to zero. 81

20 Average gain achieved upon redatuming test data. Higher values imply higher redatuming impact when correcting timeshifts between perturbed and reference data. 82

21 Average normalized residual norm achieved upon redatuming test data. Lower values characterize better generalization. 82

22 Visualization of a model used to create synthetic shot gather data. Vp here refers to the acoustic wave propagation velocity through the domain. The source, marked with a red x, is positioned in the water column at a depth of 10m. An array of equally-spaced receivers, marked with blue triangles, are positioned on the seafloor at a depth of 2000m. The seawater column appears to be vertically homogenous because the depth-dependent water velocity profile does not vary drastically enough for a visible change in the plot. However, as shown in Section 3.2.1, the modeled water velocity does indeed change with depth through the domain. For each datapoint, one instance within kk=1,...,11 is modeled as a reference instance, ref, that follows a reference velocity profile (described in Section 3.2.1). The remaining 10 instances are variations from the reference velocity profile. . 85

23 SymAE hyperparameter tuning results for dropout rates (applied to the output of the nuisance encoder) ranging from 0.2 to 0.8. The dropout rate of 0.5 yields the lowest normalized residual norm between unseen reference and redatumed seismic records, and the highest gain in timeshift reduction before and after redatuming of the perturbed seismic records. 89

24 SymAE hyperparameter tuning results for latent dimensions of the coherent encoder ranging from 60 to 120. The coherent latent dimension of 70 yields the lowest normalized residual norm between unseen reference and redatumed seismic records, and highest gain in timeshift reduction before and after redatuming of the perturbed seismic records. 90

25 SymAE hyperparameter tuning results for latent dimensions of the nuisance encoder ranging from 20 to 80. The nuisance latent dimension of 70 yields the lowest normalized residual norm between unseen reference and redatumed seismic records, and highest gain in timeshift reduction before and after redatuming of the perturbed seismic records. 90

26 Dynamic static correction results for four velocity models, taken from the unseen test dataset. Each row represents one velocity model, with its characteristic subsurface structure depicted in the left-most column. The second column displays the perturbed shot record annotated with its specific water perturbation, and the third column displays the reference shot record arising from the Hood's water velocity profile. The final two columns show the residual of redatumed - reference record, and instance (perturbed) - reference record. The contrast between pre- and post-redatumed residuals clearly demonstrates that SymAE has aligned perturbed traces with reference traces, significantly diminishing the timeshifts between arrivals in the perturbed and reference traces. 92

27 Two randomly selected models from the interface model test dataset, Model 919 and Model 926, featuring a distinct subsurface velocity profile. Each subsurface model has ten replicated instances with different velocity perturbations. These models were used for the subsequent computation of timeshifts between reference and perturbed traces, and between reference and redatumed traces; done with the purpose of further evaluation of SymAE's statics correction performance. The results of these computations are presented in Figures 28 and 29 respectively. 93

28 Results of redatuming the +4.453% perturbed instance of test datapoint, Model 919. The timeshifts compared to reference trace, before and after redatuming, for zero offset and far offset are shown. All timeshifts of the large-amplitude arrivals are reduced to zero. 95

29 Results of redatuming the −5.813% perturbed instance of another test datapoint, Model 926, with different subsurface velocities than Model 919. The timeshifts compared to reference trace, before and after redatuming, for zero offset and far offset are shown. All timeshifts of the large-amplitude arrivals are reduced to zero except for the Reflector 3 multiple in the far offset case which has a timeshift of 0.01122s 96

30 Average normalized residual norm achieved upon redatuming 100 unseen datapoints from the *horizontal reflector* and *interface model* datasets. Lower values indicate better generalization. The value highlighted in green is the result of this experiment, and the values highlighted in blue are best results from previous experiments, serving as a benchmark. 97

31 Average gain achieved upon redatuming 100 unseen datapoints from the *horizontal reflector* and *interface model* datasets. Higher values indicate better timeshift reduction before and after redatuming. The value highlighted in green is the result of this experiment, and the values highlighted in blue are best results from previous experiments, serving as a benchmark. 98

32 Function used to generate velocity models with user-controlled parameters. 114

33 GeoPhyInv code used to generate synthetic phantom dataset consisting of a collection of datapoints. This code block includes forward modeling on synthetically generated velocity models to produce their corresponding shot gathers. 115

1 Introduction

Deep learning has demonstrated remarkable performance in a wide variety of domains and is often leveraged for making high-stakes decisions. Some of the domains deep learning has excelled in include computer vision, natural language processing, autonomous systems, drug discovery - and more recently, these domains have been stretched to include unknown and evolving classes, a concept referred to as open-world learning [Zhu et al., 2024]. Similar to its beneficial presence in other domains, deep learning is gaining a notable reputation for solving challenges and optimizing processes in geophysics [Dramsch, 2020; Adler et al., 2021; Khosro Anjom et al., 2023]. With rapid advances in technology - acquisition equipment and compute power - the amount of real and simulated geophysical data is growing tremendously. This makes deep learning an especially suited tool for working with geophysical data as it is a data-driven method, relying on an abundance of data for training and predicting [Bergen et al., 2019; Yu and Ma, 2020, 2021]. Thus, my thesis studies the application of deep learning to a challenging problem in geophysics, the marine statics correction problem. The outline of this thesis consists of several parts: Chapter 1, an overview of key concepts in geophysics and deep learning pertaining to our study; Chapter 2, a benchmark test that tests the predominant model-centric paradigm for marine statics correction using deep learning; Chapter 3, a study that explores an alternative data-centric paradigm for marine statics correction; Chapter 4, a study extending the results from Chapter 3 for a more intricate geological scenario.

1.1 Motivation

Marine seismic data is used in a myriad of applications, such as the mapping of seafloor faults and tectonic structure in offshore subduction zones, analyzing shelf structures in hurricane-affected regions [Survey, 2023], oil and gas exploration, reservoir monitoring and subsea carbon dioxide (CO_2) capture and storage [International Environmental Law, 2023; Luo et al., 2023b; Obobi Ume Onwuka and Akinsola Adu, 2024], and measuring sea level rise [Demirbağ et al., 1999; Foundation, 2012]. Carbon dioxide capture and storage

(CCS) holds a particularly vital position in this array of applications as climate change continues to reach unprecedented heights. Figure 1 shows the changes in global temperature over the last 140 years, highlighting the accelerated rise of surface temperatures over the last three decades. Understanding the severity of climate change and its potentially catastrophic implications, various agreements and net-zero emission initiatives have been set in place to restrain global warming to manageable levels - notably, the 2015 Paris Agreement, which has garnered a commitment from 196 countries in the pursuit of limiting "the (global average) temperature increase to 1.5°C above pre-industrial levels" [Nations, 2015].

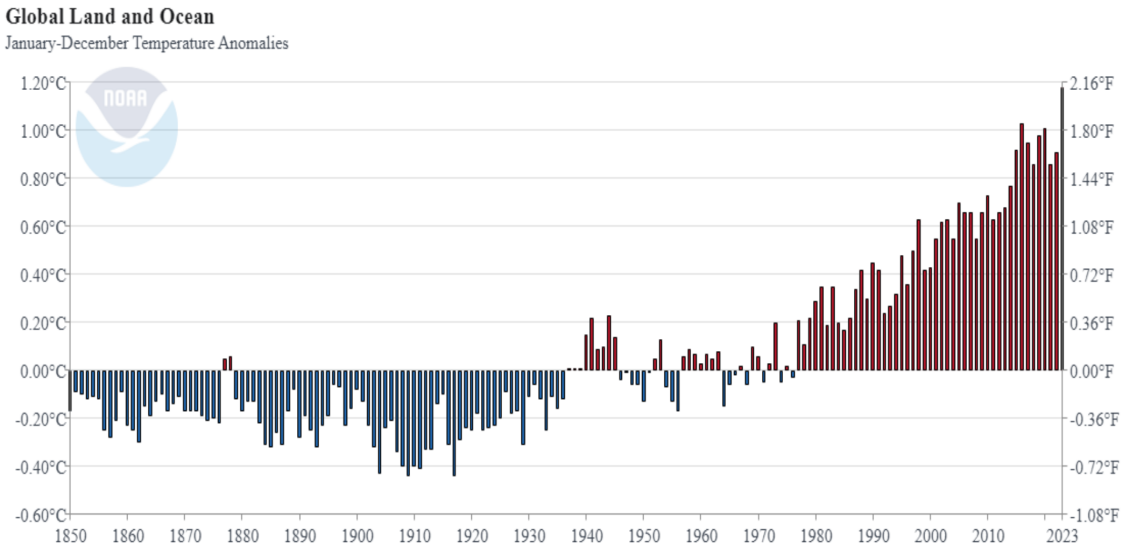

Figure 1: Average global surface temperature change from 1880 to 2024. The bars indicate the deviation from the 20th century average temperature, taken between 1901 to 2000. Blue bars show cooler than average temperature, and red bard indicate warmer than average temperatures. The data shows a an accelerated warming trend in the 20th centure, with the year 2023 being the warmest year since records began in 1850 at 1.18°C above the 20th century average of 13.9°C. Image from [Administration and Atmospheric, 2023].

Hence, parallel to and in support of these commitments, carbon capture and sequestration (CCS or geosequestration) has emerged as a key solution. As of 2023, offshore (marine) CCS is being advanced at a remarkable scale; companies and governments have announced plans to construct more than fifty new offshore CCS projects throughout the world. The fruition of these plans would result in a 200-fold increase in the annual amount of CO_2 injected beneath the seabed [International Environmental Law,

2023]. The expansion of CCS at this scale will create new processes in energy and sequestration pipelines. One such process that is extremely relevant to our study, is the time-lapse monitoring of undersea CO_2 storage sites, which aids in understanding fluid-flow dynamics within geological formations. This knowledge is essential for detecting and preventing CO_2 leakage from these sites. Specifically, in the offshore context, subsurface storage sites are situated beneath the seabed and use porous and permeable geological strata, such as depleted oil and gas reservoirs, or deep saline aquifers [Luo et al., 2023a]. The suitability of these geological formations for carbon storage depends on factors such as permeability, caprock integrity, and containment capacity [Obobi Ume Onwuka and Akinsola Adu, 2024]. Therefore, thorough mapping and assessment of subsea subsurface geology structure is vital to identify formations that possess true carbon sequestration potential for safe long-term storage.

Marine seismic acquisition is the process used to explore and create maps of the subsurface structure beneath the ocean. The acquisition happens by sending seismic energy from a source through the upper ocean layer into the subsurface, and recording the waves that reflected back from the subsurface, at receivers that are usually placed on the seafloor or towed on streamer cables attached to boats. The survey is typically conducted along a series of predetermined paths known as sail-lines, which are strategically followed by the survey vessel to cover the area of interest. The entire survey is usually performed over an extend period of time, which can range between days to months apart.

In deepwater settings, these surveys are affected by changes in the acoustic velocity of water layer, primarily due to tidal, salinity and temperature variations. The changing water layer creates lateral discontinuities (jitters) on cross-line timestamp sections of the 3D seismic stack; these lateral discontinuities are known as statics in the geophysics community. The statics ultimately cause deterioration in the seismic stack and subsequent 3D imaging. Furthermore, these effects propagate into 4D processing, where timeshifts between surveys lead to low repeatability in time-lapse seismic data[Lacombe et al., 2006]. Thus, it is pivotal to remove the statics through a static correction procedure, before subsequent processing steps of stacking, imaging and time-lapse analysis.

The conventional workflow for correcting deepwater statics is a labor-intensive and costly process, often handled manually and susceptible to inaccuracies. This is the prime mo-

tivation for our study: we seek to find a deep learning algorithm that can circumvent the challenges tied to the conventional workflow for more effective statics correction. Specifically, we are interested in using the symmetric autoencoder (SymAE) algorithm to separate the effects of varying water velocity and coherent subsurface geology in the trace data of a shot gather [Bharadwaj et al., 2020, 2022]. This entails performing offset and depth dependent timeshifts on each trace of a seismic records. SymAE is a data-driven algorithm, and therefore learns from a training dataset, how to disentangle and encode the unseen latent variables of water velocity and subsurface geology, without reliance on knowing the physics of the model. Through training, SymAE learns to separate symmetric from non-symmetric information, each representing information about subsurface geology and water velocity, respectively; this ultimately creates meaningful and useful latent spaces within the network that can be leveraged to produce new seismic records free of statics.

1.2 A Summary on Time-Lapse Seismic Imaging

Statics correction is an essential prerequisite for accurate imaging of Earth's subsurface, because in the simplest form, an image is formed when the imaging operator $F*$ acts on the seismic data d, which represents the displacement $u(x,t)$ of particles in the solid, due to the propagating elastic wave. Imaging is indeed an inverse problem, that seeks to solve for the velocity model m in the forward map $d = \mathcal{F}(m)$ [Demanet, 2021]. Refer to Section 1.5.3 for further details on the least-squares framework for finding m as the solution of the minimization problem. In this framework, the imaging operator F^* emerges in the gradient of the least squares cost function J (again described in further detail in Section 1.5.3), as $\frac{dJ}{dm}[m] = F^*(\mathcal{F}[m] - d)$. Mathematically, F^* is the adjoint of the linearized forward modelling operator $F = \frac{d\mathcal{F}}{dm}$. In practical terms, F^* is called the imaging operator because its application on seismic data d as $F^*(d - \mathcal{F}[m_o])$ produces a good image of scatterers within the subsurface model, given that m_o is a smooth background model reasonably close to the true solution m. Because the data d is crucial to the imaging condition, any corruption from statics will result in distortions in the final image. This often manifests as the incorrect positioning of subsurface reflectors. For more information, the sensitivities of changes in reflector position due to timeshift

statics in d are discussed in detail in Section 1.4.[1]

Acquisition of the data d happens in a seismic survey; this survey is considered a three-dimensional (3D) survey capable of producing 3D images - showing the length, width and depth of geological features - if the source and receiver positions follow orthogonal geometric configurations [Vermeer, 2003].Repeats of the 3D seismic survey, taken over time, forms a fourth-dimensional (4D) time-lapse dataset. In recent times, the predominant application of 4D analysis is in reservoir monitoring and management; allowing the volumetric imaging of dynamic reservoir processes such as fluid movement, pressure build-up, and heat flow. Information relayed by the 3D image however are the spatially contrasting features in the subsurface that give rise to seismic reflections. These features, taken at a snapshot in time, are a coupling of time-invariant static geology properties (i.e.,lithology, porosity, and shale content of reservoir rock) and time-varying dynamic fluid-flow properties (i.e., fluid saturation, pore pressure and temperature changes) [Lumley and Behrens, 1998]. This is where 4D seismic shines - in its ability to decouple the contributions of static geology and dynamic fluid-flow processes to the 3D image. By taking the difference between 3D images, the static geology contributions cancel out, leaving a new image of the time-varying changes caused by reservoir fluid-flow.

Understanding these dynamic reservoir processes are beneficial for a myriad of applications: mapping bypassed oil, monitoring injection fronts and contacts of fluids such as water, steam and carbon dioxide, identifying pressure compartmentalization, and characterizing the fluid-flow properties faults [Lumley and Behrens, 1998]. As described in Section 1.1, with accelerated carbon capture and sequestration efforts in coming years to curb global temperature rise, with the long-term storage of industrial carbon dioxide in deep geological reservoirs - there will be a major demand for enduring geophysical monitoring and verification of CO_2 injection through the life of the CO_2 storage project. Specifically, as CO_2 is injected into subsurface geological formations — such as depleted oil and gas fields, deep saline aquifers, and unmineable coal beds — it will be necessary to image and track the evolving CO_2 plume. This will allow engineers to ensure that the storage reservoir is filled efficiently as expected, that the CO_2 does not migrate to

[1] In time-lapse seismic reservoir monitoring, which aims to detect subtle changes in fluid flow like variations in reservoir pore pressure and fluid saturation, a spatial resolution of 15 meters is typically required to accurately position reflectors [Lumley and Behrens, 1998]. This level of spatial resolution corresponds to a time shift resolution of 0.01 seconds.

other subsurface resources (such as groundwater and hydrocarbons), that it does not flow toward high-risk areas such as major faults with uncertain seismic and flow properties, and that the CO_2 remains sealed within the storage reservoir over time [Lumley, 2019]. In summary, it is crucial to develop accurate and efficient methods to remove the statics corruption of seismic data, because inaccuracies at this foundational data level will cascade into further processing steps of 3D imaging and 4D analysis. Both of which are vital techniques for mapping the subsurface and its dynamic properties, which are indispensable for critical applications of reservoir monitoring and management, to address the needs of mankind and environmental concerns of the planet.

1.3 Paradigm Shift from a Model-Centric to a Data-Centric Approach

The conventional method for statics correction is a model-centric approach, meaning that it requires velocity model information; specifically, pertaining to changes in its velocity profile that would engender timeshifts. This approach can be decomposed into two stages: first, inverting from seismic data space to velocity model space and second, forward modeling from velocity model space to seismic data space. The inversion from seismic data space to velocity model space enables the retrieval of *measured* velocity profiles.[2] The subsequent forward modeling from velocity model space to seismic data space, allows for the replacement of *measured* velocity profile with a *reference* velocity profile that would homogenize the effects of velocity variations between shot records. For a more comprehensive description of the statics correction methodology, see Section 1.5.1.

Given this framework, the logical and natural response towards finding a deep learning algorithm that replaces the functionality of the conventional method, would be to emulate its two-stage process - inversion and forward modeling - that transform between model space and data space domains. Figure 2 provides a visual representation of this concept, outlined under the heading Model-Centric Deep Learning. In the context of marine

[2]In this thesis, the *measured* velocity profile refers to the *real* velocity profile, as used in Section 1.5.1. Hence both terms are used interchangeably and denote the initially unknown velocity model of the perturbed geological area of interest.

acquisition, it is the changes in water velocity that create statics. Hence, this is the *measured* velocity model we intend to replace with the *reference* model. As the observed seismic records holds information from both water (ocean) and subsurface layers, the inversion must differentiate the seismic contributions of these two velocity models.

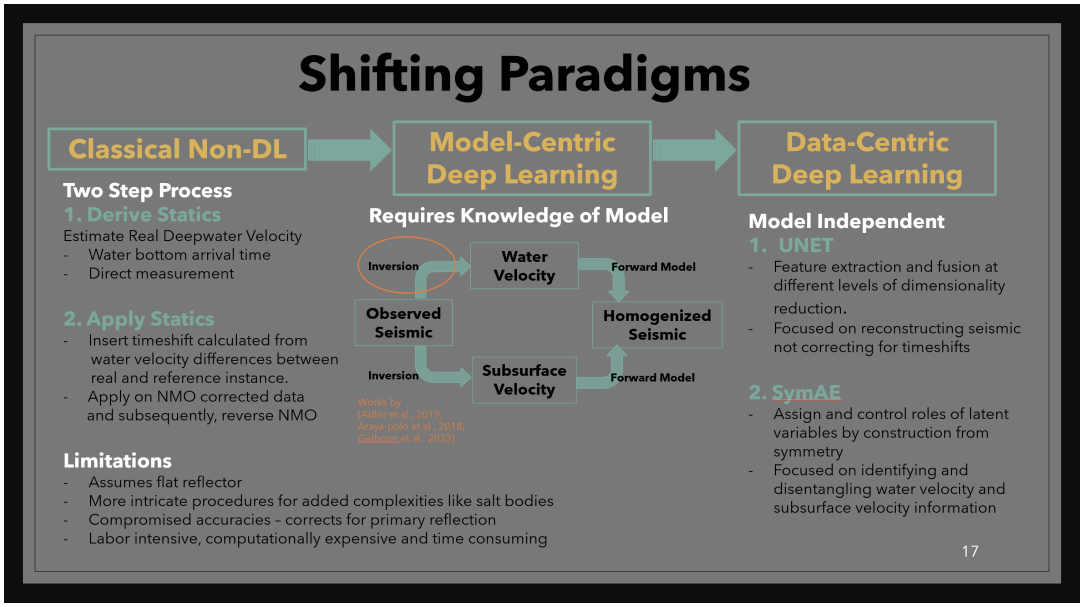

Figure 2: Summary of paradigm changes in geophysics research, with details pertaining to our focus in marine statics correction. The arrows between the headers indicate the direction for the shifting paradigms, from classical non deep-learning (DL), to model-centric deep learning and lastly, to data-centric deep learning.

Previous studies have been done on inverting for entire subsurface models from seismic data [Adler et al., 2019; Araya-polo et al., 2018; Gelboim et al., 2023], but to the best of our knowledge, no research has specifically addressed inverting for a section of the velocity model from seismic data. This will become the focus of Chapter 2 in this thesis.

Collectively, Figure 2 highlights the changing climate in geophysics research between three paradigms: Classical (or Conventional) Non-Deep Learning, Model-Centric Deep Learning and Data-Centric Deep Learning. Here, we briefly overview the three different paradigms in the context of statics correction:

1. **Classical Non-Deep Learning** methodology consists of a two-step process of deriving and applying statics. This two-step process is described in more details in Section 1.5.1 and 3.1. Notably, this approach is premised on first defining *measured* and *reference* subsurface velocity models, before deriving and applying statics. Due

to additional steps of having to first define geological models, this approach incurs added time, compute and labor cost. It is also prone to inaccuracies, which may arise from inaccurate derivation of the *measured* model or unmet assumptions applied onto subsurface models.

2. **Model-Centric Deep Learning** closely resembles the structure of Classical Non-Deep Learning, where the statics correction process is centered around inverting for the *measured* velocity model and establishing the *reference* velocity model. The main difference lies in switching conventional inversion algorithms, such as tomography and full-waveform inversion, for deep learning algorithms. This shift is typically aimed at reducing the computational load of having to iteratively calculate numerical solutions of the wave equation for the inferred model, till the simulated data aligns reasonably well with measured data. Upon establishing the velocity models, the final seismic record is generated by forward modeling with the wave equation.

3. **Data-Centric Deep Learning** is a more streamlined approach for statics correction compared to the previous two paradigms, directly mapping the *measured* seismic record, that is statics-corrupted, to the *reference* seismic record, that is statics-free. This approach is guided by the seismic data - forming the basis of the features and their interconnections that the network learns. Therefore, this approach operates independently of velocity model information, and hence achieves model-independence when performing on in-distribution (ID) data.[3] This model-independence marks a substantial advancement over the previous two paradigms.

The Evolution from U-Net to SymAE

In the Data-Centric Deep Learning paradigm illustrated in Figure 2, two different deep learning algorithms - U-Net and SymAE - are listed with their corresponding functionality. While our primary focus is on SymAE, U-Net is also mentioned due to its previous application in our research on marine statics correction. This interest began during my internship at Shell [Mateeva, 2021], where the aim was to overcome the expensive and error-prone Classical Non-Deep Learning workflow, currently used on their marine seis-

[3]For additional information on the requirements for creating an ID dataset and the network's robustness to out-of-distribution data, refer to Section 4.6.

mic acquisitions, with a new deep learning solution. The initial (and natural) idea was to use the U-Net algorithm, which has shown exceptional performance in image processing tasks [Ronneberger et al., 2021].

The 2D seismic record, before and after statics correction, can also be viewed as just a 2D image. With this perspective, the U-Net's inputs were seismic records arising from the real (*measured*) deepwater environment, and the U-Net's outputs were seismic records derived from the *reference* velocity model. We had anticipated, that by training the U-Net on such a dataset, the algorithm would learn the map from inputs to outputs, and produce corresponding *reference* seismic records for unseen *measured* records. However, the results of our experiments using U-Net revealed that despite being trained with the presence of timeshifts (induced by seawater velocity variations) in the traces of the reference output records as compared to the *measured* input records, when the network made predictions, it was aligned to the foundational task of *reconstructing* seismic records from extracted lower-resolution feature maps.

It is important to note that the timeshifts caused by changing velocities of the ocean layer, are typically only a fraction of the dominant period of the seismic records,[4] which is determined by the peak frequency (most energetic component in the frequency spectrum) of the source wavelet. A direct way to calculate the timeshift is by taking the difference, $\frac{\mathrm{d}}{v_{\mathrm{new}}} - \frac{\mathrm{d}}{v_{\mathrm{old}}} = \Delta t$, where d is the distance traveled by the wave in the medium, v_{new} and v_{old} represent the new and old velocities of the changing medium, and Δt, the resulting timeshift.

This highlights that the reconstruction of *reference* seismic records is a compounded task which involves indentifying features across various temporal-scales: the traveltime gaps between all arrivals (direct arrival, primary reflections and multiples), the dominant period which influences the shape of each arrival's waveform, and the subtle timeshifts.[5] Given the complexity of the task, it is not surprising that the U-Net was only able to capture the first two temporal scales - arrival traveltime gaps and dominant period -

[4]The dominant period represents the difference in time between successive peaks or troughs of the predominant wave in the seismic records. It is computed as $\tau_o = \frac{1}{f_o}$, where f_o is the peak frequency of the source wavelet and τ_o is the dominant period [Duarte et al., 2020].

[5]The subtle timeshifts which are typically a fraction of the dominant period, also changes the arrival time of waves in either positive or negative directions based on the sign difference of $v_{\mathrm{new}} - v_{\mathrm{old}}$, or analogously, $v_{reference} - v_{measured}$

which constitute the *foundational* temporal features needed to reconstruct the seismic records. Thus, the U-Net reached its limits here, and was unable to transcend the larger temporal scales to produce shifted traces at the subtlest temporal scale of changing water-velocity induced timeshifts.

This insight catalyzed the search for a deep learning solution that would emphasize the subtle statics timeshifts, without compromising the extraction of features from larger temporal scales (which represent the foundational building blocks of the seismic record). To this end, we recognized that information conveyed by the larger temporal scales related to the large-scale geological profile of the medium, whereas information encoded in the smaller scale timeshifts corresponded to small geological perturbations (the water velocity variations). Another thing we noted, was that by nature of marine seismic acquistions, which is taken repeatedly in the same area over a long duration of time (between days and months), the large-scale geological profile can be reasonably assumed to be unchanging, whereas the smaller ocean-layer profile perturbations vary within this time period. This results in seismic records that feature *subsurface geology coherence* but *water velocity variations*.

This distinction between the coherency of larger-scale temporal features and the complementary dis-coherency, or variation, of smaller-scale temporal features (timeshifts) that occurs during the extended time period of a marine seismic survey, thus became the insight guiding our search for an alternative deep learning algorithm post U-Net experiments. Understanding that coherent information would be symmetric across all seismic records collected in the particular survey on the geological region of interest, we could directly embed this property of symmetry into the physical structure of the neural network [Bharadwaj et al., 2020]. For this reason, we chose to focus our study on SymAE, a symmetry-based autoencoder that separates symmetric and non-symmetric information from data - allowing the separation (and emphasis) of both larger-scale temporal features and subtle timeshifts - for the even more precise reconstruction of statics-free, *reference* seismic records.

1.4 Precisions Relating Velocity Model to Timeshifts

As described in Section 1.3, the timeshifts due to changes in the medium's velocities are computed as $|\frac{\mathrm{d}}{v_{\text{new}}} - \frac{\mathrm{d}}{v_{\text{old}}}| = \Delta t_{\text{perturbed}}$. Considering the midway case (see Section 3.2.1) of water layer velocity perturbations, p, in the $[-6, +6]$ range from the Hood's water velocity profile, we assume a uniform background water velocity layer of 1550m/s (approximate water velocity at 0.4km depth of $p = +6\%$ profile) for the perturbed profile and 1500m/s (approximate water velocity at 0.4km depth of the Hood's profile) for the reference profile.[6] Using statics correction, our aim is to reduce $\Delta t_{\text{perturbed}}$ to the smallest value possible, $\Delta t_{\text{corrected}}$, which ideally approaches zero.

To achieve a precision in timeshifts of Δt, the level of precision required of the background velocity Δv and reflector location Δz in the wave equation model is computed by,

$$|\Delta v| = z\Big|\Delta\Big(\frac{1}{t}\Big)\Big| = \frac{z}{t^2}|\Delta t| \tag{1}$$

$$|\Delta z| = v|\Delta t| \tag{2}$$

where v is the background velocity of the medium, z is the distance travelled by the wave and t is the arrival time of the wave. The parameters v, z and t depend on the chosen arrival, because the wave's propagation path through the medium varies for different arrivals; this means it may experience different velocity profiles v, travel through different distances z and have varying traveltimes t from source to receiver.

We examine the precisions of four different wave arrivals: the direct arrival, where the wave travels from the source at the sea surface through the water layer to the receivers on the seafloor; the triplet and quintuplet which are multiples of the direct arrival through the water layer; and lastly, the primary reflection where the wave similarly propagates from the source at the sea surface, through the water layer, continues into the subsurface, reflects of the first interface/reflector in the subsurface, and then reaches the receivers on the seafloor. For more details and an illustration of a similar medium, refer to Section 3.2.1.

[6] Refer to Figure 14 to visually estimate the individual water velocities at 0.4km depth for the perturbed and reference profiles.

1. **Direct Arrival**

   The zero-offset direct arrival's $\Delta t_{\text{perturbed}}$ is 0.04s $\approx \left|\frac{2000\text{m}}{1500\text{m/s}} - \frac{2000\text{m}}{1550\text{m/s}}\right|$, where 2000m is the distance from the sea surface to the seafloor. A 75% reduction in $\Delta t_{\text{perturbed}} =$ 0.04s yields in $\Delta t_{\text{corrected}} = 0.01$s. The direct arrival's parameters are $z = 2000$m, $v = 1500$m/s and $t = \frac{2000\text{m}}{1500\text{m/s}} = 1.33$s. Substituting these values along with $\Delta t_{\text{corrected}} = 0.01$s into Equations 1 and 2, yields $\Delta v = 11.31$m/s and $\Delta z =$ 15.00m. Relative to the background velocity, Δv represents a sensitivity of $0.75\% = \frac{11.31\text{m/s}}{1500\text{m/s}} \times 100$. Relative to the water layer depth (total distance travelled by the wave), Δz represents a sensitivity of $0.75\% = \frac{15\text{m}}{2000\text{m}} \times 100$. Thus, correcting the statics of the direct arrival to 25% of its original value produces precision on the velocity field of 11.31m/s, which corresponds to a sensitivity to velocity differentials of 0.75%; and a precision on the interface position of 15.00m, corresponding to a sensitivity to depth differentials of 0.75%.

2. **Seafloor Triplet**

   The zero-offset seafloor triplet $\Delta t_{\text{perturbed}}$ is 0.13s $\approx \left|\frac{6000\text{m}}{1500\text{m/s}} - \frac{6000\text{m}}{1550\text{m/s}}\right|$, where 2000m is the distance from the sea surface to the seafloor. A 92.25% reduction in $\Delta t_{\text{perturbed}} = 0.13$s yields in $\Delta t_{\text{corrected}} = 0.01$s. The triplet's parameters are $z = 6000$m, $v = 1500$m/s and $t = \frac{6000\text{m}}{1500\text{m/s}} = 4.00$s. Substituting these values along with $\Delta t_{\text{corrected}} = 0.01$s into Equations 1 and 2, yields $\Delta v = 3.75$m/s and $\Delta z = 15.00$m. Relative to the background velocity, Δv represents a sensitivity of $0.25\% = \frac{3.75\text{m/s}}{1500\text{m/s}} \times 100$. Relative to the total distance travelled by the wave, Δz represents a sensitivity of $0.25\% = \frac{15\text{m}}{6000\text{m}} \times 100$. Thus, correcting the statics of the triplet to 7.75% of its original value produces precision on the velocity field of 3.75m/s, which corresponds to a sensitivity to velocity differentials of 0.25%; and a precision on the interface position of 15.00m, corresponding to a sensitivity to depth differentials of 0.25%.

3. **Seafloor Quintuplet**

   The zero-offset quintuplet's $\Delta t_{\text{perturbed}}$ is 0.22s $\approx \left|\frac{10000\text{m}}{1500\text{m/s}} - \frac{10000\text{m}}{1550\text{m/s}}\right|$, where 2000m is the distance from the sea surface to the seafloor. A 93.35% reduction in $\Delta t_{\text{perturbed}} =$ 0.22s yields in $\Delta t_{\text{corrected}} = 0.01$s. The direct arrival's parameters are $z = 10000$m, $v = 1500$m/s and $t = \frac{10000\text{m}}{1500\text{m/s}} = 6.67$s. Substituting these values along with $\Delta t_{\text{corrected}} = 0.01$s into Equations 1 and 2, yields $\Delta v = 2.25$m/s and $\Delta z =$

15.00m. Relative to the background velocity, Δv represents a sensitivity of $0.15\% = \frac{11.31\text{m/s}}{1500\text{m/s}} \times 100$. Relative to the total distance travelled by the wave, Δz represents a sensitivity of $0.15\% = \frac{15\text{m}}{10000\text{m}} \times 100$. Thus, correcting the statics of the quintuplet to 4.65% of its original value produces precision on the velocity field of 2.25m/s, which corresponds to a sensitivity to velocity differentials of 0.15%; and a precision on the interface position of 15.00m, corresponding to a sensitivity to depth differentials of 0.15%.

4. **Primary Reflection**

   For the zero-offset primary reflection, we select a reflector located 4000m below the sea surface with a subsurface velocity of 2325m/s (midpoint velocity for a linearly increasing velocity profile of 1800m/s at the 2000m seafloor depth to 2850m/s at the 4000m reflector depth). The zero-offset direct arrival's $\Delta t_{\text{perturbed}}$ is 0.04s $\approx \left|\left(\frac{2000\text{m}}{1500\text{m/s}} + \frac{4000\text{m}}{2325\text{m/s}}\right) - \left(\frac{2000\text{m}}{1550\text{m/s}} + \frac{4000\text{m}}{2325\text{m/s}}\right)\right|$, where 2000m is the distance from the sea surface to the seafloor and 4000m is the two-way reflection distance between the seafloor and subsurface reflector. A 75% reduction in $\Delta t_{\text{perturbed}} = 0.04\text{s}$ yields $\Delta t_{\text{corrected}} = 0.01\text{s}$. The primary reflection's parameters are $z = 6000\text{m}$, $v_{\text{rms}} = 2007\text{m/s}$[7] and $t = \frac{2000\text{m}}{1500\text{m/s}} + \frac{4000\text{m}}{2325\text{m/s}} = 3.05\text{s}$. Substituting these values and $\Delta t = 0.01\text{s}$ into Equations 1 and 2, yields $\Delta v = 6.45\text{m/s}$ and $\Delta z = 20.07\text{m}$. Relative to the background rms velocity, Δv represents a sensitivity of $0.32\% = \frac{6.45\text{m/s}}{2007\text{m/s}} \times 100$. Relative to the total distance travelled by the wave, Δz represents a sensitivity of $0.33\% = \frac{20.07\text{m}}{6000\text{m}} \times 100$. Thus, correcting the statics of the primary reflection to 25% of its original value results in a precision on the velocity field of 6.45m/s, corresponding to a sensitivity to velocity differentials of 0.32%, and a precision on the interface position of 20.07m, corresponding to a sensitivity to depth differentials of 0.33%.

For clarity, we base our conclusions on the precisions relating velocity model to timeshifts for the rest of our study, using the direct arrival wave, which involves only a single velocity layer without any reflections from an interface. In this context, the term "precision" can

[7]For consistency with the other parameters z and t of the primary reflection that encompass the entire wave propagation path downwards from sea surface source to subsurface reflector and upwards to the seafloor reflector, we use the v_{rms} value for the parameter v as it is a weighted average of the velocity layers (water layer and subsurface layer) of the propagation path, with the weights being the traveltimes through each layer. For a more detailed description on v_{rms}, see Section 2.3.

be used interchangeably with "resolution," a term commonly found in 3D and 4D seismic literature. By using the direct arrival as our reference, to achieve a timeshift resolution of 0.01 seconds - representing 25% of the original timeshifts between perturbed and reference seismic data - the required resolutions in the wave equation model are 15.00 meters for the interface/reflector position and 11.31m/s for the background velocity. This corresponds to a sensitivity of 0.75% for both depth and velocity differentials.

Notably, because $|\Delta z| = v|\Delta t|$, the 15m depth resolution corresponding to 0.01s timeshift resolution of the direct arrival, is also applicable to all seafloor multiples which propagate through the same water velocity layer. For waves that travel through the subsurface layer though (such as the reflector primary) and consequently through higher velocity regions, the required timeshift resolution increases, to a value less than 0.01s. However, to standardize these resolutions towards a baseline quantity, which will serve as a useful threshold value in assessing the performance of experiments in this thesis, we approximate that all arrivals require a minimum timeshift precision of 0.01s.

Required Statics Correction Precision for Imaging

Acquired seismic data is either considered to be correct in the physical, unknown medium; or incorrect in a hypothesized medium. Statics correction aims for the data to be a better match with wave propagation in the reference medium. See Section 1.5.1 for a more comprehensive description about the nature of the statics correction process.

To achieve a good seismic image in the reference medium, a spatial resolution of 15m is required [Lumley and Behrens, 1998]. As previously explained, this spatial precision require a time precision of at least 0.01 seconds for all wave arrivals. Uncorrected records from velocity perturbations in the seawater layer generally exhibit traveltime inaccuracies ranging from 0.02s to 0.22s, which is inadequate.[8]. Thus, the goal of marine statics correction, as investigated in our thesis, is to provide the 0.01s level of time precision for every wave arrival in the seismic record.

[8]This range of traveltime inaccuracies, from 0.02s to 0.22s, applies to seismic records that capture arrivals up to the quintuplet wave (as demonstrated by the computations in this section) and for water velocity perturbations within the [-6,+6] range (see Section 3.2.1 for more details on this range).

1.5 Review of Core Concepts

Before embarking on the following chapters that describe our experiments in marine statics correction, here we compile a companion of concepts that grounds the reader with context and background information for smoothly understanding the ensuing experiments. These core concepts are divided into three main domains: statics correction, geophysics and deep learning. Within each domain, we explore more specialized topics that pertain to our experiments.

1.5.1 Static Correction

An essential step in reflection seismic processing sequences is the correction of statics: timeshifts that arise in pre-stack seismic traces due to velocity or topographic heterogeneities in the wave propagation path. Uncorrected for, these timeshifts induce lateral discontinuities (jitters) in (3D) common-midpoint (CMP) gathers, which create stack deterioration and imaging artifacts [Lacombe and Shelf, 2009]. Furthermore, these errors propagate into time-lapse (4D) interpretation methods - which are based on timeshifts between baseline and monitor (a repeated seismic survey on the baseline area) stacks to evaluate changes in the geological area of interest. Thus, accurate static correction is a vital prerequisite for improved stack coherency, meaningful imaging and post-stack 4D analysis.

Statics can arise in different seismic acquisition environments. In onshore seismic surveys, precipitation-related changes in soil moisture is a key factor for altering the near-surface velocity profile [Bergmann et al., 2014]; whereas in offshore (marine) seismic surveys, changes in tides, weather and currents induce variations in sea surface elevation levels; and changes in pressure, salinity and temperature of the water layer create variations in the water velocity profile. Additionally, production-related and injection-related processes within reservoirs create large geomechanical changes to the reservoirs, which consequently create considerable velocity changes to the overburden and traveltime timeshifts. This effect is especially observed when monitoring hydrocarbon producing or carbon dioxide storage reservoirs and their surroundings [Bergmann et al., 2014; Yan

et al., 2023].

It is important to note that static correction is a dynamic process, which is a function of reflector depth and source-to-receiver offset. Up till the 21st century, statics correction was based on the assumption of vertical raypaths - which is an especially poor approximation when the near-surface possesses complex geology with depth-dependent velocities - and was known as *datum* static correction. The vertical raypath assumption for statics corrections (known as datum statics correction) leads to adjustments in reflection times that simulate data recording vertically beneath or above, each surface location (Figure 3). In reality however, ray paths between sources and receivers may exist an an angle to the vertical; this angle affects the distance traveled by rays through each velocity layer in a heterogeneous medium. Hence, the non-vertical angles of ray propagation govern reflection arrival times - and these angles are determined by the offset distances between source and receiver sets, from which the rays emanate from and arrive at respectively. Additionally, the movement of source and receiver positions to a datum (the reference plane) changes the ray-path significantly, especially if the ray-path is not vertical - adding further inaccuracies to the vertical ray-path assumption. Hence, the simplification of vertical ray-paths is chiefly inaccurate: the drawbacks of which are thoroughly discussed in *Statics Corrections for Seismic Reflection Surveys* by Cox [1999], and the remedial case for dynamic statics correction presented in it as well. Figures 5 illustrates the limitations of datum statics correction, by showing how the travel path lengths vary for different source and receiver offsets, in a two-layer near-surface velocity model, as shown in Figure 4.

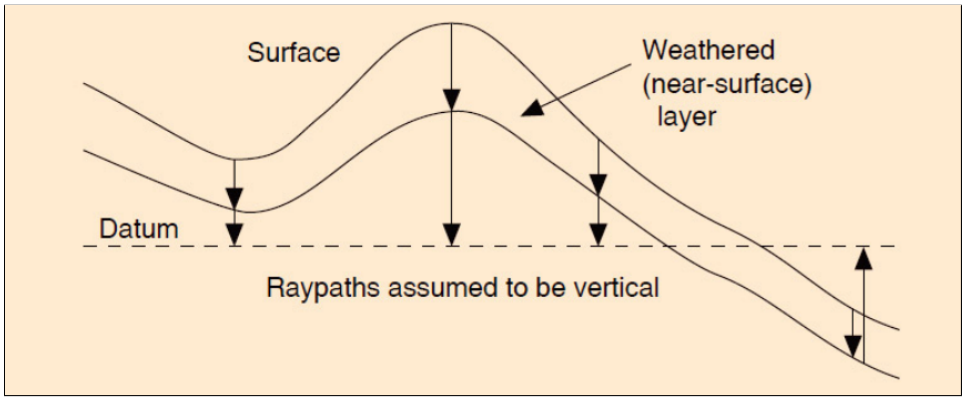

Figure 3: Standard static correction till the 21st century: most near-surface timeshifts are removed by applying datum static correction. The datum in this case is the horizontal plane, which functions as a reference plane, on which a pseudo source and receiver are positioned vertically below the original source and receiver located on the surface of the weathered layer. Image from [Cox, 2000].

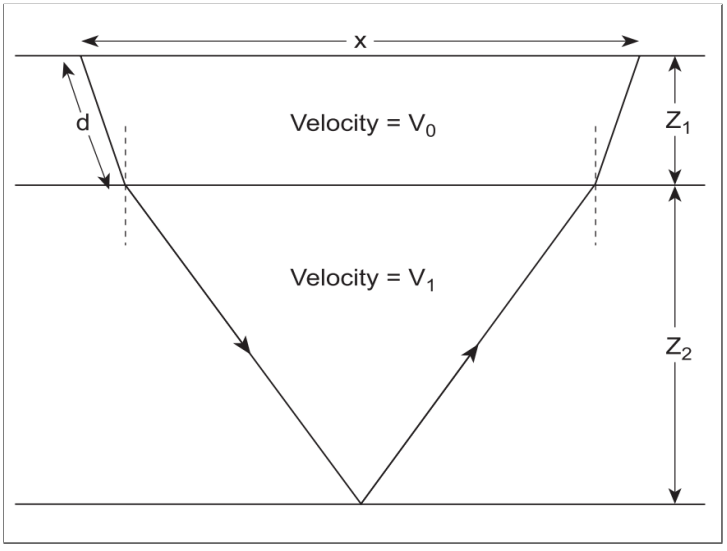

Figure 4: Two-layer velocity model used to show the variations in distance traveled by the ray propagating from a source and arriving at a receiver, with offset distance x. Consider the case where the near-surface layer has a depth, Z_1, of 500m, and a reflector depth, $Z_1 + Z_2$ of 1500m. The travel path length in the near surface layer is labeled as d. Image from [Cox, 1999].

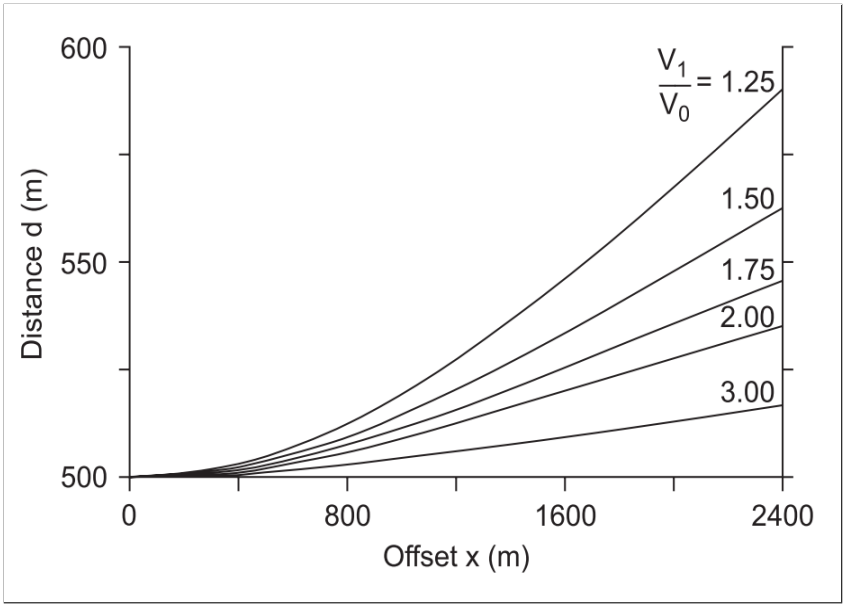

Figure 5: Different travel path lengths as a function of source-receiver offset, x, and velocity ratio between the deeper layer V_1 and near-surface V_0, $\frac{V_1}{V_0}$, whereby the near-surface is assumed to be a water layer with velocity of 1500m/s. Notice that the travel path length d increases with increasing offset x, from approximately 95m to 205m, in correspondence to velocity ratios ranging between 3.00 to 1.25. Larger velocity ratios lead to smaller angles of incidence at the interface between near-surface and deeper layer, hence a smaller distance traveled in the near-surface layer. In conclusion, deviations in ray path distance and in velocity structure of the traveled region exist in practical seismic recordings, which ultimately result in travel time perturbations that are different from the vertical raypath assumption of datum statics correction. Thus, statics correction is a dynamic process that is offset and depth dependent. Image from [Cox, 1999].

The computation of statics correction consists of a two-step workflow of: First, deriving statics. Second, applying static corrections to the seismic measurements. When deriving statics, it is important to understand that statics arise primarily due to elevation or velocity heterogeneities in the ray propagation path between source and receiver. Thus, the total static correction $T_{x,t}$ applied to a trace at offset x and arrival time t can be decomposed as $T_{x,t} = T^V_{x,t} + T^E_{x,t}$, where $T^V_{x,t}$ is the velocity correction and $T^E_{x,t}$ is the elevation correction. In order these statics correction values, parameters for the changing velocity layer and/or elevation need to be acquired. These parameters are usually obtained by direct measurement made during seismic surveys. For example, it is typical for surface elevation is obtained from survey information of the shot line, and layer velocity measurements obtained from uphole surveys at discrete locations along the line. Ultimately, the goal is to construct - from measurements - a geological model of the perturbed region (whether due to velocity and/or elevation heterogeneities).

In contrast to the *real* geological model of the perturbed region, is the *datum* model of the region. Traditionally, the datum was defined by Sheriff (1991) as "An arbitrary reference surface, reduction to which minimizes local topographic and near-surface effects. Seismic times and velocity determinations are referred to the datum plane as if sources and geophones had been located on the datum plane and as if no low-velocity layer existed." In a broader sense, the usage of the term datum is for a *reference* model - which includes adjustments to the velocity or elevation heterogeneities of the *real* model - to which measurements are corrected. Thus the datum model possesses replacement velocity and elevation profiles, that are known and chosen.

Once the reference datum model has been chose, the next steps for statics correction is the derivation and application of dynamic timeshifts (offset and depth dependent) to the seismic measurements. The timeshifts are the traveltime differences of raypaths, between source and receiver pairs, within the velocity structure of the real model and reference model. Calculating these timeshifts through the ray-tracing method composes the derivation step of statics correction. Once the timeshifts are computed, the application step of statics correction consists of applying the timeshifts to arrivals in the pre-stack seismic traces. In the specific case of horizontal and flat reflectors bounding homogenous velocity layers, the normal move-out (NMO) correction can be utilized. In

this case, leveraging the knowledge of the hyperbolic relationship for two-way reflection times, the measured prestack traces of a CMP seismic record (from the *real* model) can be NMO corrected to transform the reflection hyperbola into a straight horizontal line; essentially, transforming the entire source-receiver offset range into a zero-offset setting. Thus, for primary reflections, the timeshifts between NMO corrected real and NMO corrected reference models (with different velocities but same elevation levels) is simply $\left|T^{V}_{x,t'}\right| = \left|2d(\frac{1}{v_0} - \frac{1}{v_1})\right|$, where t' is the arrival time of the primary reflection of the real model, v_o the reference model, and v_1 the real model. The application of these timeshifts to the NMO corrected real model data moves the primary reflections such that they had arisen by wave propagation in the velocity structure of the reference model. Subsequently, this new timeshifted record (*redatumed* record) can be reverse NMO corrected using the reference model velocity to produce the *corrected* seismic record, with primary reflections aligned to a new and known reference velocity. This NMO-based static correction was utilized by Celine Lacombe when correcting for water column - tide level and velocity - variations in 4D United Kingdom Continental Sheld (UKCS) marine seismic data. The result being, a reduction in timeshifts and a narrowing of the timeshift distribution between baseline and monitor stack volumes [Lacombe and Shelf, 2009].

When the assumption of horizontal and flat reflectors - bounding homogenous velocity layers - are not met. and when performing statics correction on multiples; the NMO-based static correction method described above breaks down. Instead, the fundamental concept of raypaths should be returned to - specifically, the (arrival) raypaths' angles, which encode the velocity structure of the model, and distances, which encode the elevation information of the model [C. Henley, 2009]. Thus, the computation of these ray-traces require knowledge of the real and reference geological models, their velocity profiles and topographies. Once both models have been defined, certain techniques may be employed to ease the computation of the ray-traced timeshifts: notably, transforming the data into the horizontal ray parameter p and intercept time τ (contribution to the travel time from the vertical component of propagation) domain so that dynamic corrections become simple time and offset shifts [Cox, 1999]; or transforming the data into the radial trace domain (in which, the near-surface raypath angle is the principle parameter), sorting the radial traces into common-raypath-parameter gathers, and implementing statics correction on these gathers using interferometric methods [Deere,

2009; C. Henley, 2009; Henley, 2014].

Overall, statics correction has been a model-centric process, which relies on first, defining real and reference geological models (with velocity and elevation profiles) before deriving and applying statics. Due to the additional steps of defining models, this process incurs added compute and time cost. Moreover, if the real model is inaccurately derived from measurements (typically, geological regions can be undersampled, and interpretative decisions are taken to interpolate values between observational points) or the field data itself is of compromised quality (for instance, low signal-to-noise ratio), this leads to further errors in the derivation of statics, and to the final velocity derivation of the region [Cox, 1999]. In an effort to overcome the high cost and potential inaccuracies and subjectivity ubiquitous to the model-centric appoach, data-centric approaches for statics correction have been more recently explored.

One such data-centric approach was implemented by Bergmann et al. [2014] on the 4D data from the Ketzin carbon dioxide storage site in Germany. First, trace-to-trace timeshifts of the data from different seismic records were estimated by crosscorrelations. Then, the timeshifts were decomposed in a surface-consistent manner into its components: source, receiver, and the stress-induced reservoir velocity 4D static differences. By doing so, the spatial distribution of decomposed statics in the geographical region was realized, and was applied on further monitor post-stacks without needing to invert for velocities from from each monitor seismic record (a requirement for the model-centric statics correction). They found the time-lapse difference (TLD) method enhanced signal to noise ratio in the 4D difference stacks, in addition to significantly reducing the time and labor intensity required to perform the correction, when compared to model-centric method.

1.5.2 Forward Model

Wave Equations

Elastic waves are propagating disturbances in solids. This disturbance causes the particles in the solid to experience displacement $u(x,t)$, a time-dependent vector field; and velocity $v = \frac{du}{dt}$, a time-dependent vector field. The linear elastic wave equations in an

isotropic medium is:

$$\rho\frac{d^2u}{dt^2} = (\lambda + 2\mu)\nabla(\nabla \cdot u) - \mu\nabla \times \nabla \times u \tag{3}$$

where λ, is the compression stiffness of the sample when it is compressed in one axis and constrained in other two axes [Lowe, 2001]; and μ, shear modulus, are the material-dependent Lamé parameters; and ρ is the density of the medium.

Taking the Helmholtz decomposition of the general vector field u as the sum of independent parts of the longitudinal scalar field ϕ and transverse vector field ψ, $u = \nabla\phi + \nabla \times \psi$, and noting the vector identities that the curl of a gradient is zero, and the divergence of a curl is zero,

$$\nabla \times u = \nabla \times \nabla \times \psi = -\Delta\psi \tag{4}$$

$$\nabla \cdot u = \Delta\phi \tag{5}$$

Inserting these equivalences into the elastic wave equation, we get

$$\nabla\left[\rho\frac{\delta^2\phi}{\delta t^2} - (\lambda + 2\mu)\Delta\phi\right] + \nabla \times \left[\rho\frac{\delta^2\psi}{\delta t^2} - \mu\Delta\psi\right] = 0 \tag{6}$$

Taking the divergence and curl of Equation 3, we respectively obtain each field solving their own wave equation in isolation:

$$\rho\frac{\delta^2\phi}{\delta t^2} - (\lambda + 2\mu)\Delta\phi = 0 \tag{7}$$

$$\rho\frac{\delta^2\psi}{\delta t^2} - \mu\Delta\psi = 0 \tag{8}$$

As such, ϕ are the longitudinal, pressure waves (P-waves) and ψ are the transverse, shear

waves (S-waves). From Equations 7, we derive the P wave speed as $\sqrt{\frac{\lambda+2\mu}{\rho}}$ and S wave speed as $\sqrt{\frac{\mu}{\rho}}$.

Fluid-Solid Interface Mode Conversion

As discussed in the previous section, a key difference between P-waves and S-waves is that S-waves propagate with a velocity that is dependent on the material having a non-zero shear modulus. A water layer has a shear modulus of zero and no resistance to shear stress, thus only allowing for the propagation of P-waves. In the marine acquisition case, an incident P-wave that has traveled through the water layer to reach the seafloor (water bottom) is reflected, transmitted or converted to S-wave in the underlying solid. Specifically, the conversion to S-wave happen if the P-wave strikes the seafloor at an oblique incidence. This is known as a mode-converted wave [El Allouche et al., 2008].

The degree of mode conversion is influenced by the angle of incidence, the P-wave velocity, S-wave velocity, and the densities of the mediums. Smaller angles of incidence cause the reflected and transmitted P-wave to carry larger amounts of energy, whereas larger angles of incidence cause the mode conversion to S-wave to become more efficient [Nanda, 2016]. The conversion coefficients at the fluid-solid interface are premised on the continuity of traction $\tau = \sigma \cdot \hat{n}$, where $\hat{n}$ denotes the unit normal to the interface; and the continuity of the normal component of velocity $\dot{u} \cdot \hat{n}$ [Komatitsch et al., 2000]. The conversion transmission coefficient T_{PS}, the amplitude ratio of the transmitted S-wave and incident P-wave is,

$$T_{PS} = \frac{2\cos\theta_1}{\left[\cos\theta_1\left(A_1 - A_2\frac{\rho_2\alpha_2}{\rho_1\alpha_1}\cos(2\varphi_2)\right) - A_2\cos(2\theta_2) + sin(\varphi_2)\right]} \tag{9}$$

with

$$A_1 = \frac{\rho_2\alpha_2/(\rho_1\alpha_1)}{\alpha_2/\beta_2}\sin(2\varphi_2) \text{ and } A_2 = \frac{\alpha_2\cos(2\varphi_2)}{\beta_2\sin(2\theta_2)} \tag{10}$$

where α, β and ρ denote the P-wave velocity, the S-wave velocity and medium density; θ is the angle between the normal of the fluid-solid interface and the transmitted P-wave, and φ is the angle between the normal of the fluid-solid interface and the transmitted S-wave [El Allouche et al., 2008].

1.5.3 Inverse Model

Seismic velocity inversion computes the velocity model of a certain target area, from the recorded seismic data d_r and can be summarized as :

$$\hat{m} = F^{-1}(d_r) \tag{11}$$

where F^{-1} is the inversion operator. The prevalent framework of solving for m is least-squares, whereby m is the solution of the minimization framework:

$$\min_m J[m], \text{ where } J[m] = \frac{1}{2}||d - F[m]||^2 \tag{12}$$

where $||d||_2^2 = \sum_{r,s} \int_0^T |d_{r,s}(t)|^2$ is the L^2 norm squared of the recorded seismic data indexed by receiver r, source s and time t; and J is the objective function we seek to minimize.

The minimization is typically performed using iterative methods based on variations of J at a base point m_0. The variations of J is primarily expressed through its functional gradient $\frac{\delta J}{\delta m}[m_0]$ and functional Hessian $\frac{\delta^2 J}{\delta m^2}[m_0]$. These variations appear in the two main methods used for the minimization process: gradient descent and the Gauss-Newton Iteration.

In gradient descent, the model update $m^{(k+1)}$ is computed as $m^{(k+1)} = m^{(k)} - \alpha \frac{\delta J}{\delta m}[m_k]$, where α is a user-chosen parameter affecting the speed and stability of convergence. In contrast, in the Gauss-Newton method, the model update is computed as $m^{(k+1)} =$

$m^{(k)} - \left(\frac{\delta^2 J}{\delta m^2}[m_k]\right)^{-1} \frac{\delta J}{\delta m}[m_k]$, where $\left(\frac{\delta^2 J}{\delta m^2}[m_k]\right)^{-1}$ is the inverse of the functional Hessian. Both methods have pros and cons, namely gradient descend typically converges slowly whereas the the Gauss-Newton method involves the Hessian of J which is a big matrix that incurs high inversion and storage cost.

In general, seismic inversion is an ill-posed problem. This is chiefly caused by: (a) the existence of too many local minima in the landscape of $J[m]$ hindering the search for the global minimum, and (b) directions at m_1 where $J[m]$ has zero (or near-zero) curvature in the vicinity of the solution m*; in this case, the solution is unstable that small noise variations of the data cause big movements of the global minimum in mistaken directions [Demanet, 2021; Gelboim et al., 2023].

1.5.4 Deep Learning

Representation learning is a range of methods that allows a machine to be fed with raw data and automatically discover the representations needed for classification or detection. In our study, we focus on deep learning, a form of representation learning implemented through deep neural networks (DNNs). These networks consist of simple yet non-linear modules that transform data representations across multiple layers, achieving higher levels of abstraction. Through this process, the network identifies significant features within the dataset, which can then be used for a myriad of tasks, including prediction, classification, regression, and generative modeling. These applications span a variety of domains such as image recognition, natural language understanding, and biological system predictions.

The foundational form of a deep neural network is the multilayer node (MLP) network, which consists of an input layer, several hidden layers, and an output layer [Goodfellow et al., 2016]. Figure 6 illustrates an example of an MLP network. This network comprises discrete node units (nodes) arranged vertically in layers and connected by weights (directed edges) between nodes of different layers. The input layer, on the left, receives the input data. The three middle layers are the hidden layers, and the output layer, on the right, contains the final computed values.

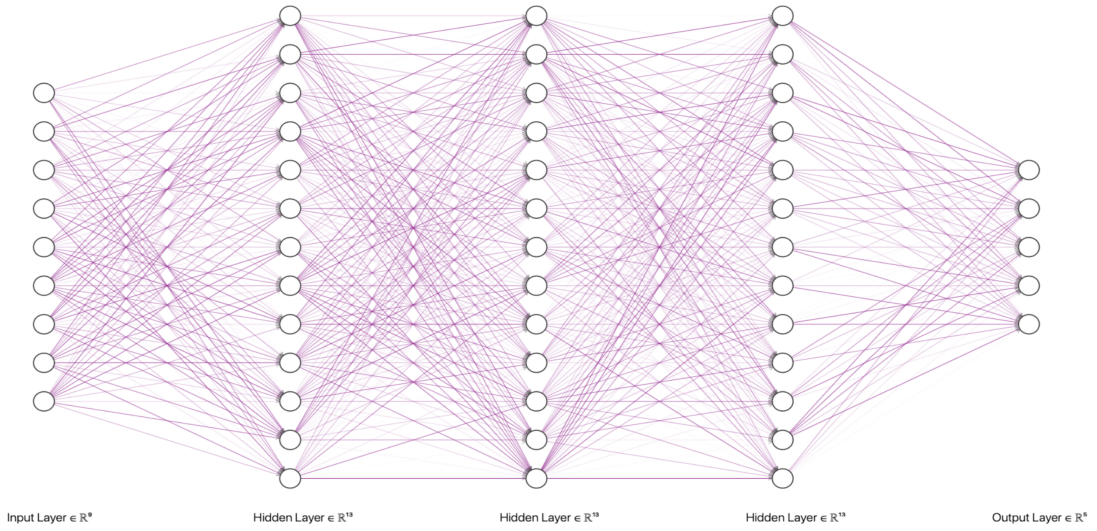

Figure 6: A diagrammatic representation of the Multilayer Perceptron (MLP) network. This network possesses an input layer with nine nodes, three hidden layers each containing 13 nodes, and an output layer with five nodes. The arrows between the nodes indicate the weights connecting nodes from one layer to the next layer. The transparency of these arrow signifies the strength of the weight.

To understand how the model learns representations, each node in the network can be thought of representing an element within a set of features unique to that layer. Expanding on this, each layer organizes nodes into hierarchical levels of abstraction. The input layer contains the raw features. As data progresses through each layer, the feature set from the previous layer is transformed into a more abstract feature set in the current layer through a series of nonlinear transformations: weighted connections and activation functions. Therefore, the feature set in the current layer is composed of elemental building blocks from the preceding layer; forming a more abstract depiction of the data. This representation captures intricate patterns and relationships within the dataset. By extracting significant features from the input dataset, the model learns the hierarchical mapping from input data to target outputs [Bengio et al., 2014; Lecun et al., 2015].

Processes

The explanation in this section is primarily based on the works of Lecun et al. [2015], who has clearly delineated several key deep learning concepts. The first concept to understand is the purpose of the nodes within neural networks. The node can be viewed

as a processor that generates a real-valued activation output. Note that the superscripts l and m refer to the nodal units of the network. Each calculates a weighted sum, net^l, from the activations of the previous layer, y^m, and applies a differentiable activation function, f, to this sum. Typically, an additional term, a scalar bias b, is added to net^l before applying f, giving $y^l = f(net^l + b^l)$. This bias shifts the input of the activation function vertically from the origin. However, for clarity in understanding the learning algorithm, we will ignore the bias term for the remainder of this study.

The activation function f is usually a non-linear real valued function such as *tanh*, *sigmoid* ot *elu*, which compresses the weighted sum into a smaller range and has a non-zero derivative within this range. This property is crucial for computing gradients during the backward pass of the learning algorithm.

Now that we understand the node computes the activation function applied to a weighted sum of previous activations, we can move on towards understanding the objective function. The objective function measures the error between the network's output predictions and the actual desired outputs. To minimize this error, the network adjusts its weights (and biases), each by a unique amount and direction. This adjustment is guided by the gradient vector, which indicates how much the objective error would increase or decrease if each weight were slightly altered. By using this gradient vector, the weights are updated in the opposite direction to the gradient, thereby reducing the objective error in subsequent computations.

The objective error function can be conceptualized as a surface in a high dimensional space of, whereby the vertical axis is the error value and the horizontal axes, the weight values. Intuitively, this forms a hilly landscape; with regions of hills (high error) indicating poor network performance and regions of valleys (low error) indicating good performance. This landscape can consists of several local minima, and a global minimum where the set of weights yield the lowest error and optimal performance. The landscape can also have other features like saddle points, sharp ridges and plateaus, which may pose as challenges when finding the lowest possible error.

The process of finding the lowest error entails training the network. During training, the network uses an iterative optimization algorithm to find the local minimum of the error

landscape. This can be visualized as a ball rolling down the hills and valleys, moving in the direction of steepest descent, as determined by the gradient of the error function with respect to the weights. Stochastic gradient descent (SGD) is the most commonly used optimization algorithm used in practice. It calculates the objective error on a few samples (a batch) of training data, computes the average gradient vector for these samples, and adjusts the weights accordingly. This process is iterated with different batches till the average objective error stabilizes and stops decreasing. The stochasticity in this method arises because each batch of randomly-chosen samples, gives a noisy estimate of the average gradient over all samples. Despite the navigation path being noisy, it is effective in its convergence, often reaching good weight configurations faster than more elaborate optimization techniques. Once the network has found its point of lowest error in the landscape, its performance is evaluated by applying it to an unseen dataset, known as the test dataset. This measures the generalization ability of the network; how well it is able to produce accurate outputs for new, previously unseen inputs.

Autoencoder

The autoencoder is an unsupervised deep learning algorithm that specializes in reducing input data from a high-dimensional feature space to a lower-dimensional representation, know as the latent space; and subsequent reconstruction of the the input data from the low-dimensional latent space [Wang et al., 2016; Bank et al., 2021]. Thus, the latent codes in the latent space is a compressed representations of non-linear features extracted from the input data. This compression is ensured by constraining the number of latent nodes to be lower than the number of nodes defining the input layer - thus, achieving the desired low dimensional representation of the input data. The architectural backbone of the autoencoder consists of three components: the encoder, the decoder, and the latent space. Figure 7 illustrates how the three components are connected to each other: the encoder downsamples input data X to the latent space, and the decoder upsamples the latent codes to reconstructed input (output layer) X'.

The objective error function of the autoencoder, which guides the adjustment of weights during training via the gradient vector of the function, is simply the reconstruction error between input X and output X' [Berahmand et al., 2024], defined as:

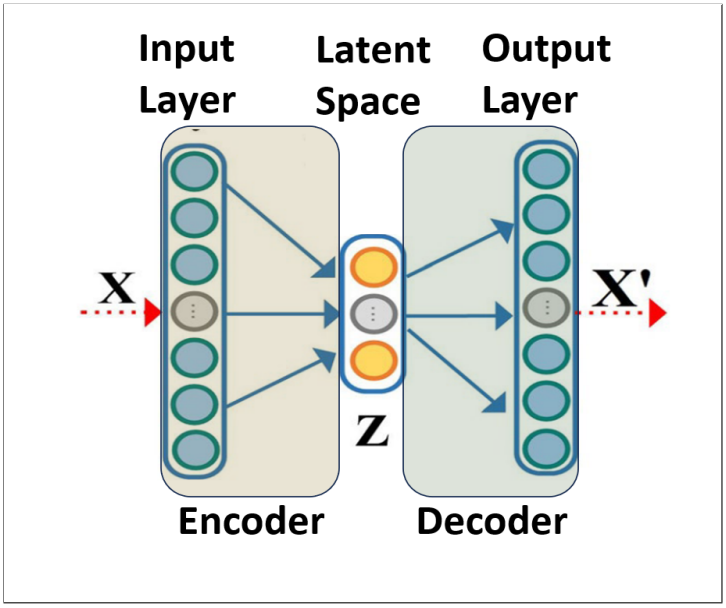

Figure 7: Architectural overview of the autoencoder algorithm. X represents the input data of the input layer, Z represents the extracted codes in the latent space, and X represents the reconstructed output data of the output layer. The encoder performs the non-linear downsampling of data from X to Z, and the decoder performs the nonlinear upsampling of latent codes from Z to X'. Image edited from [Berahmand et al., 2024]

$$L(X, X') = \sum_{i=1}^{n} ||X_i - X_i^{'||^2} \tag{13}$$

Thus, during training, the network minimizes the reconstruction error L by adjusting the weights of the encoder and the decoder, to obtain the final latent code. The latent code is consequentially, a meaningful low-dimensional feature representation of the data. We highlight the fundamentals of the autoencoder algorithm here because this will form the basis for the innovative design of SymAE, that seeks to attain a disentangled latent representations of symmetric and non-symmetric information from the input data - using two encoders, instead of one. For a more elaborate explanation on SymAE, see section 3.2.2 and [Bharadwaj et al., 2022].

2 Model-Centric Benchmark Test

In this chapter we investigate the limits of the model-centric deep learning paradigm for marine statics correction. As depicted in Figure 2, this approach is primarily based on replacing classic seismic inversion methods, such as full-waveform inversion (FWI) and traveltime tomography, with deep learning algorithms. When embedded in the marine statics correction workflow, this replacement will significantly reduce the high compute time and cost required by the iterative nature of classic inversion methods that seek to fit the inferred model to the observed seismic data. Instead of having to generate repeated seismic simulations from model updates, deep learning will allow for the direct inversion of the *measured* velocity model - the real velocity profile of the geological area of interest. As mentioned in Section 1.5.1, it is vital to first establish this *measured* model before performing subsequent computations of deriving and applying statics.

Our study begins with designing a unique and novel deep learning algorithm that maps seismic records to their corresponding water velocity models. We then analyse the model predictions of the algorithm to determine their viability for usage in the (model-centric) marine statics correction workflow. The numerical results of the tests in this chapter thus becomes the benchmark quantities that we aim to supersede in the following chapters; which will focus on a more direct, data-centric approach, for marine statics correction.

2.1 Deep Learning Approach

We design a neural network to perform the task of inverting for the water velocity model from a seismic record. The deep learning algorithm is generally perceived as a hierarchical composition of non-linear functions (that are assembled in parallel as a layer). Our network extends the notion of hierarchical compositions to the deep learning algorithms themselves, each with their own hierarchical composition of non-linear functions. Thus, our network comprises various components from different deep learning algorithms and layers, all interconnected to create a unified network.

The deep learning algorithms we employed were inspired from previous research in seis-

mic inversion using deep learning; specifically, the works of Mauricio Araya-Polo in Deep Learning Tomography [2018], Deep Recurrent Architectures for Seismic Tomography [2019] and Encoder-Decoder Architecture for 3D Seismic Inversion [2023]. In Deep Learning Tomography, it was shown that a multilayer feedfoward network was capable of mapping from the semblance space of a stack of shot records to the velocity model space. Here, the input to the neural network was the semblance cube, a pre-engineered feature of the seismic data, encompassing patterns that correlate with reflector position and velocity. In Deep Recurrent Architectures for Seismic Tomography, it was demonstrated that recurrent neural networks, which are adept at processing sequential data by retaining previous states (a memory-like capability), achieved greater accuracy in velocity model inversion from seismic records compared to convolutional networks. This superiority arises from the fact that seismic records represent spatially sampled time series of propagating waves, exhibiting both short-term and long-term dependencies. In Encoder-Decoder Architecture for 3D Seismic Inversion, it was shown that the U-Net style architecture (with skip-connections that replicated and conveyed encoder feature maps to the decoder) was able to effectively reconstruct 3D velocity models from seismic records, even in the presence of white noise or field noise seismic contamination. Keeping these algorithms in consideration for potential integration, we developed a novel neural network designed to infer a partial velocity model from a seismic record registering a wave propagating throughout the complete velocity model.

The reason we aim to infer a partial velocity model from the complete model is because we want to invert for just the water velocity profile from the total model, which encapsulates an upper water column layer and lower geological subsurface layer. The water velocity profile is depth-dependent and isotropic - forming a horizontally layered medium in the 2D model. With the depth of the water column and granularity of its velocity profile (across the depth-axis) kept constant across different models, the water velocity profile can be represented as a 1D series of velocity values changing with depth. In contrast, the seismic record capturing wave propagation through the entire model (water column and geological subsurface) is 2D, with time and receiver location axes. Therefore, the mapping from seismic record to water profile involves transitioning from 2D data space to 1D model space; where the 2D seismic record serves as the input to the neural network, while the 1D profile is the output.

Our network's complete architecture is illustrated in Figure 8, and its specifics are outlined in Table 1. The algorithms mentioned earlier were integrated into our network; where the U-Net segment handles feature extraction and retention; and the Flattening and Multilayer Feedforward segment converts 2D tensors into 1D vectors, fusing and further modifying the extracted feature maps. The Gated Recurrent Unit (GRU) segment specializes in learning long-term and short-term sequential dependencies within the data, and the Smoothing segment employs convolution by sliding weighted kernels across the entire 1D vector.

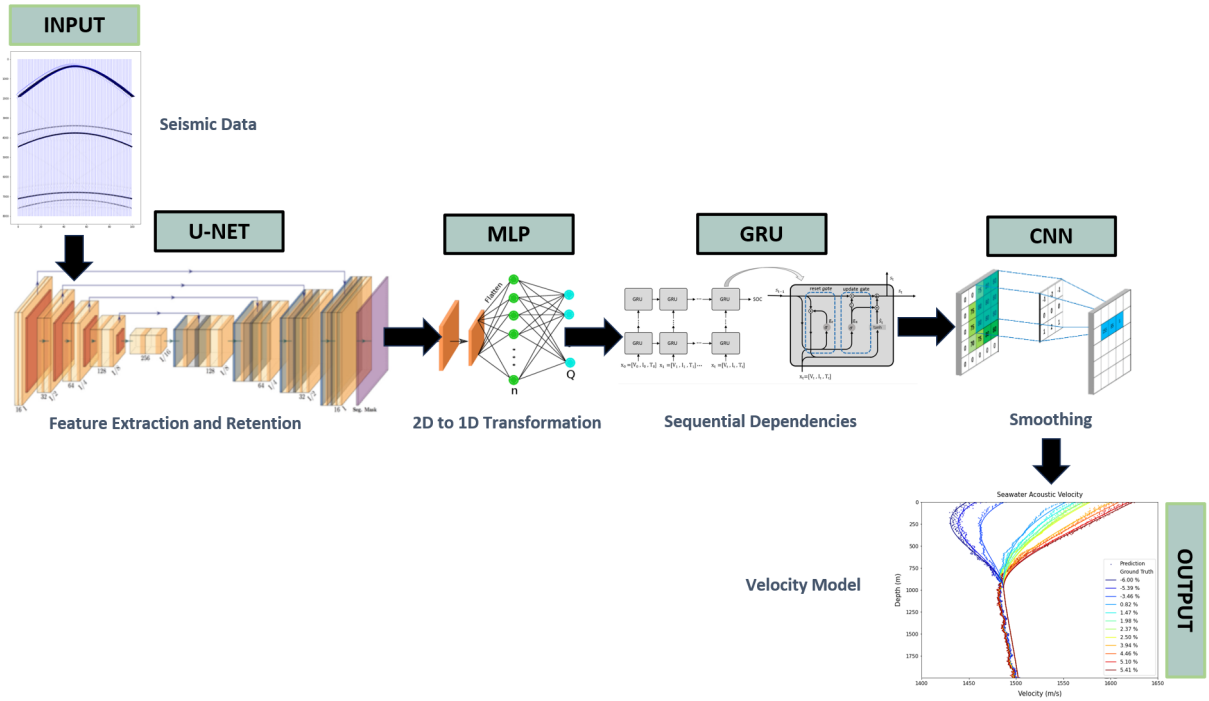

Figure 8: A simplified overview of the neural network architecture designed to infer the water velocity profile from the seismic record of a wave propagating through the upper water layer and geological subsurface beneath. This architecture is an amalgamation of various deep learning algorithms. These algorithms - namely U-NET, MLP, GRU and CNN - have previously shown their capability in successfully performing the inversion from seismic data space to velocity model space [Araya-polo et al., 2018; Adler et al., 2019; Gelboim et al., 2023].

2.2 Data Preparation and Implementation

We generated 1000 velocity models using the GeoPhyInv software, which produces 2D geologically feasible models with a realistic mix of features. Each model had dimensions of 6.5km width and 6km depth, represented by a 2D tensor grid of 1301x1201 points.

Block	Layer	Unit
Input	0	Seismic Record (100 x 544 x 1 grid points)
Enc1, U-NET	1	Conv2D (64, (3 x 3), elu)
	2	Conv2D (64, (3 x 3), elu)
	3	MaxPool
Enc2, U-NET	4-6	Enc1 (128)
Enc3, U-NET	7-9	Enc1 (256)
Enc4, U-NET	10	Conv2D (512, (3 x 3), elu)
	11	Conv2D (512, (3 x 3), elu)
Intermediary Layers	12	BatchNormalization
	13	Dropout (0.5)
Dec1, U-NET	14	UpSampling2D
	15	Conv2D (256, (2 x 2), relu)
	16	Concatenate
	17	Conv2D (256, (3 x 3), relu)
	18	Conv2D (256, (3 x 3), relu)
Dec2, U-NET	19-23	Dec1 (128)
Dec3, U-NET	24-28	Dec1 (64)
Intermediary Layers	29	Conv2D (2, (3 x 3), relu)
	31	Conv2D (1, (3 x 3), relu)
	32	Reshape (100 x 544)
	33	Permute (2,1)
	34	BatchNormalization
	35	Dropout (0.5)
	36	Flatten
MLP	37	Dense (800, relu)
	38	Dense (400, linear)
Intermediary Layers	39	Reshape (400 x 1)
	40	BatchNormalization
GRU	41	GRU (400, linear)
Interewediary Layer	42	Reshape (400 x 1)
CNN, Smoothing	43	Conv1D (2, (5), linear)
	44	Conv1D (1, (5), linear)
	45	BiasLayer
Output	46	Velocity Model (400 x 1 gridpoints)

Table 1: Detailed neural network architecture desgined to infer the water velocity model from a seismic shot record.

The water layer in each model was 2km deep, spanning the sea surface to seafloor; and the underlying subsurface extended a 4km depth from the seafloor to the base of the medium. As this dataset is identical to the [-6,6] perturbation horizontal reflector dataset of Chapter 3, we refer the reader to Section 3.2.1 for a detailed description of the subsurface geology and water velocity profile of the 1000 velocity models. Additionally, to minimize reflections from boundaries, we applied perfectly matched layer boundaries on the right, left, and bottom edges of the medium.

To simulate seismic data, we used forward modeling via an acoustic wave equation solver in GeoPhyInv, using a 6.78Hz peak frequency Ricker wavelet as a source. Each velocity model had a source placed at the top of the sea surface, halfway across the model's width at 3.25km, and 100 Ocean Bottom Node (OBN) receivers evenly spaced along the seafloor at 2km depth.

The 1000 velocity models were divided into separate training, testing, and validation sets using a 7:2:1 ratio respectively. Each 2D shot record, which served as the network input, was paired with its corresponding 1D water velocity profile, the network output/target. The neural network was designed using TensorFlow and trained for 400 epochs with a mini-batch size of 100. Training was performed using the Adam optimizer and included early stopping regularization, to minimize the MSE loss function and prevent over-fitting.

2.3 Results

We trained our network, detailed in Section 2.1, on OBN-acquired seismic records as input and their corresponding water velocity profiles as output. After training, the network was tested on unseen seismic records from the held-out test dataset, and its water velocity predictions analyzed. Figure 9 presents the predictions generated by our neural network for a subset of 12 unseen shot records, each arising from a distinct velocity profile of the water layer and geological subsurface beneath it. The predicted water velocity profile of each seismic record was compared to its ground truth profile using the root mean square error (RMSE) metric. The RMSE was 0.40894 on 100 unseen velocity models, indicating a close similarity between the predicted water velocity profiles and their corresponding ground truths profiles.

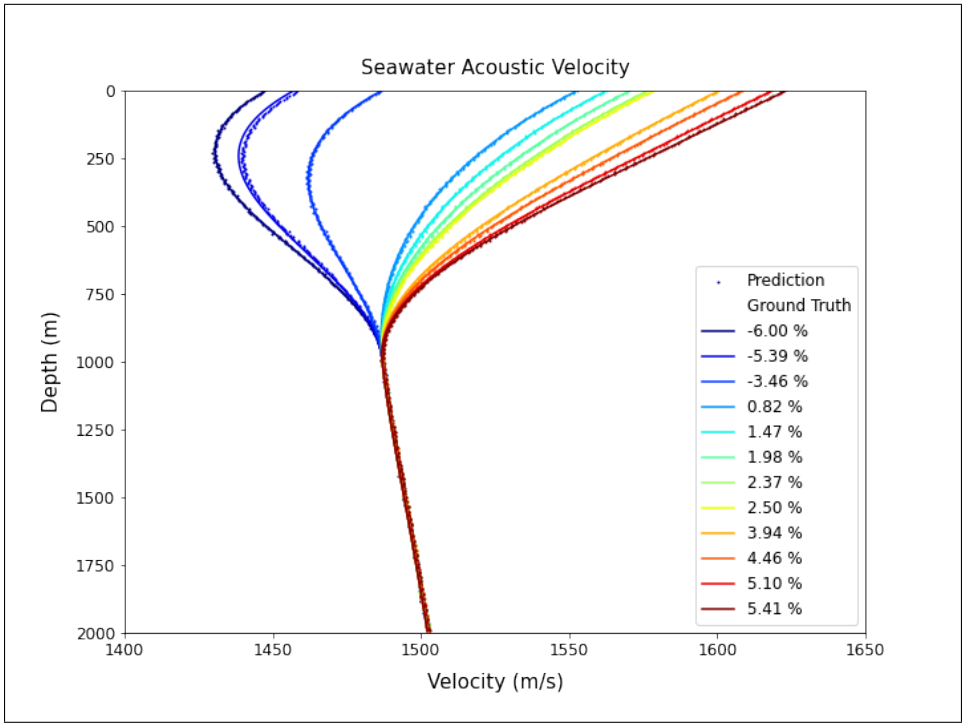

Figure 9: Inversion results using our specially designed neural network, described in Section 2.1 The network inverts for water velocity profiles of the 2000m deep ocean layer, utilizing shot records from Ocean Bottom Nodes located on the seafloor. The annotated percentage value indicate the relative amplitude modulations (perturbations) to a Reference Hood's water velocity profile, which is detailed further in Section 3.2.1. The network successfully captures the curvature of various perturbed profiles in the top 1000m of the ocean, and the overlapping monotonically increasing profiles with depth,in the lower 1000m. Overall, the network predictions have a RMSE of 0.40894 compared to ground truth velocity profiles.

In Figure 9, from depths of 1000m to 2000m (seafloor), the predictions for all models are superimposed on one another and are aligned well with their ground truth. From depths of 0m to 1000m, where the water velocity profile exhibits a more curved structure, most predicted points closely follow the curves of the ground truth profile lines. However, there are a few points that deviate from their respective ground truth lines; notably, the upper-most points (between 0m to 250m) of the -6.00% perturbation profile and the upper to mid-length points (between 0m to 700m) of the -5.39% perturbation profile. Despite these deviations, the overall results show that the neural network is capable of generating satisfactory water velocity profiles from seismic shot records.

To evaluate the performance of the inferred water velocity profiles in seismic data space, we aim to forward model these profiles to their corresponding seismic records, with the acquisition parameters outlined in Section 2.2. Since the inversion process we focused

on maps seismic records to water velocity models (which comprise our dataset), we do not have access to the true subsurface geology information pertaining to each seismic record. Instead, we recognized that the subsurface geology information essential for the forward model is encoded within the features of a seismic record itself : the reflection pattern - which includes the amplitude and arrival times of the propagating waves - that is influenced by the jump in acoustic impedance between different subsurface layers [Demanet, 2021]. With this understanding, we proceed to forward model the water velocity profiles, despite it being the partial velocity model of the complete model (upper water layer and underlying geological subsurface), using the processing method described below.

To obtain the seismic record of the inferred water velocity profiles without explicit knowledge of their subsurface velocity profiles, we refer to the conventional statics correction methodology, which replaces the *measured* water velocity profiles with a constant water velocity profile known as the *reference* velocity, to perform the traveltime shifts on the seismic data. In our study, the *measured* velocity represents the ground truth profile, and the *reference* velocity represents inferred (neural network predicted) profile. This statics correction methodology is based on the Normal Moveout (NMO) correction procedure, which is fitting for our study as our subsurface velocity models feature flat and horizontal, seafloors and reflectors, bounding homogeneous velocity layers - assumed conditions for implementing NMO-based correction.

Fundamentally, the NMO correction uses the hyperbolic assumption to compute the velocity required to flatten the hyperbolic curve observed in seismic reflection data. The assumption is grounded in the knowledge that the travel time of seismic wave reflecting off a flat, horizontal interface follows a hyperbolic trajectory as a function of offset distance from the source [Castle, 1994],

$$t(x)^2 = t_o^2 + \frac{x^2}{V_{rms}^2} \tag{14}$$

where x is the source-receiver offset; t_0, the vertical (zero-offset) two-way travel time;

$t(x)$, the non-zero offset two-way travel time; and V_{rms}, the rms medium velocity,

$$V_{\text{rms}} = \sqrt{\frac{\sum_{k=1}^{N} \Delta\tau_k V_k^2}{\sum_{k=1}^{N} \Delta\tau_k}} \tag{15}$$

where V_k is the interval velocity of the kth layer and $\Delta\tau_k$ is the vertical travel time in the kth layer. By applying this correction and adjusting for the hyperbolic shape, the velocity is estimated. By shifting each trace forward in time by $t(x) - t_o$, the reflection is flattened from its original hyperbolic shape, such that its arrival time is t_o for all offsets [Stein and Wysession, 2003].

To forward model our inferred water velocity profiles, using the input dataset of corresponding seismic records of ground truth velocity (complete geological model consisting of water and subsurface layers), we perform an NMO-based procedure, which consists of the following steps [9]:

1. **NMO Correction with Estimated Velocity Profiles:** Perform NMO correction on the seismic records with their measured (or estimated) water velocity profiles to flatten the hyperbolic reflection curve.

2. **Static Timeshift Application:** Apply a static timeshift across the entire shot record. This timeshift is based on the zero-offset arrival time difference between the measured (ground truth, in our case) and reference (predicted, in our case) rms water velocity profiles.

3. **Reverse NMO Correction:** Conduct a reverse NMO correction on the seismic records using the rms reference velocity to account for the dynamic nature of timeshifts with offset, and thereby recreate the hyperbolic-shaped reflection curve once more.

Figure 10 shows the result of applying the NMO-based forward model procedure to three randomly selected seismic samples from our test dataset. In the first column, the ground truth and predicted velocity profiles are displayed, with their respective rms velocity

[9] This NMO-based forward model process is analogous to that detailed in Lacombe et al. [2006] for the specific function of deriving and applying statics on seismic records in deepwater statics correction.

values indicated as dashed lines. The second column shows the ground truth seismic record, which is input into the neural network to infer the predicted velocity profile. The third column presents the NMO-based forward modeled seismic record; created by passing the ground truth seismic record from the second column through the three step process described above, replacing its *measured* velocity profile V_{rms} with its *reference* velocity profile $\hat{V}_{\text{rms}}$. Finally, the fourth column depicts the difference of the seismic records: predicted - ground truth.

Overall, these samples show a discrepancy between the predicted and ground truth seismic records. This discrepancy is measured by the normalized residual norm of the two records (for more details on this metric, see Section 4.4), annotated at the top of the *difference* records in Figure 10. The normalized residual norms for the three samples range from 0.251 to 0.723, increasing as the difference between $\hat{V}_{\text{rms}}$ and V_{rms} becomes larger. By nature of the NMO-based forward modeling procedure, the predicted inferred water velocity profiles, which are noisily scattered about the smooth ground truth profiles, are averaged to a single scalar value $\hat{V}_{\text{rms}}$ which is only slightly different from V_{rms}. Therefore, although the water velocity predictions are noisy, this noise is does not appear in the predicted seismic records, as it translates only the scalar difference, $\hat{V}_{\text{rms}} - V_{\text{rms}}$. Consequentially, the reflection pattern in the predicted seismic records preserves the arrival features of the ground truth records, but experiences slight timeshifts, resulting in amplitude differences - at corresponding timestamps and traces - between the two records.

To understand the difference between $\hat{V}_{\text{rms}}$ and V_{rms}, we take its Mean Absolute Percentage Error (MAPE) to measure the average proportion of error across 100 test data points and the R^2 score to evaluate the goodness of fit of $\hat{V}_{\text{rms}}$ to V_{rms}. The MAPE was 0.26% and the R^2 was 0.91954, indicating a a high degree of accuracy and strong fit between $\hat{V}_{\text{rms}}$ and V_{rms}. Additionally, we calculated the average normalized residual norm (refer to Section 4.4 for more details on this metric) for the 100 test seismic records (from which V_{rms} was derived) and the corresponding predictions (forward modeled records using $\hat{V}_{\text{rms}}$): this value was 0.25750. This indicates that on average, the residuals (differences between ground truth seismic and predicted seismic values) are 25.7% of the range of the ground truth seismic values.

To gain a deeper understanding on the discrepancy between predicted and ground truth seismic records, we closely examined their traces to calculate their arrival timeshifts. The timeshifts were computed using the Moving Window Cross-Correlation method, which is described in detail in Section 3.3. Figure 11 below shows the timeshifts for a 5.987% water velocity perturbation sample from the test dataset for the zero offset and far offset case, at 3.25km. The predicted trace of the zero offset case achieves better accuracy than the far offset case with its direct arrival reaching at the same time as its ground truth counterpart ($\Delta t_1 = 0$s); and the timeshifts of the other arrivals (Δt_2 and Δt_3) ranging from 0.0225s to 0.0449s. In contrast, the direct arrivals of the far offset case are misaligned with a timeshift Δt_1 of 0.0225s, and the other arrivals have timeshifts (Δt_2, Δt_3 and Δt_4) ranging from 0.0337s to 0.0449s. Overall, most timeshifts for the traces are non-zero and larger than the precision requirement of 0.01s described in Section 1.4, highlighting the impact of inversion inaccuracies in the velocity model towards forward modeled seismic data.[10]

2.4 Discussion

Our study was focused on inverting for water velocity profiles from marine seismic records using a uniquely designed deep learning network. Accurate inversion results would indicate that this model-centric approach to statics correction via deep learning is potentially viable. We obtained inversion results of our water velocity profiles, which closely resembled the ground truth curves - although the predicted values were slightly scattered about the ground truth, yielding an RMSE of 0.40894.

The model-centric framework for statics correction requires accurate inversion for the water velocity profile, which constitutes the *measured* profile, for replacement with a selected *reference* velocity profile that would homogenize the timeshifts due to water velocity perturbations. Thus, the predicted and ground truth velocity profiles both

[10]Further data-centric experiments in this thesis involve similar timeshift computations between redatumed (predicted) and reference (ground truth) seismic traces. To compare the timeshifts from our model-centric benchmark test with subsequent experiments, refer to Figures18, 19, 28, and 29. Our benchmark test indicates that, except for specific cases (such as the direct arrival of the zero offset trace) where a timeshift precision of 0.01 seconds is achieved, most arrivals exhibit larger timeshifts, reaching up to 0.04 seconds. In contrast, our subsequent data-centric SymAE experiments show that the 0.01s level of time precision can be consistently achieved for every wave arrival.

represent the *measured* velocity profiles. To assess the impact of the deviations between the predicted water velocity and ground truth profiles, we forward modeled the predicted water velocity profiles to their corresponding seismic records. This was done following the procedure outlined for NMO-based static correction as we did not have explicit access to their *true* subsurface velocity profiles.

The nature of the NMO-based static correction method requires that the 1D vector of predicted and ground truth depth-dependent velocity profiles first be averaged to their corresponding scalar rms velocity, before subsequent processing. The rms velocity influences the shape of the hyperbolic curvature and timeshifts applied to all arrivals of the seismic record, making it a vital quantity. The rms velocity MAPE of 0.26% and R^2 score of 0.91954 indicated a high accuracy between predicted and ground truth rms velocity values. Despite this high accuracy, the forward-modeled seismic records from predicted water velocity values were moderately accurate, with an average normalized residual norm of 0.25750, when compared to the ground truth seismic records from ground truth water velocity values.

Thus, the error in the predicted seismic record values is about a quarter of the range of values seen in the ground truth records. Given that our dataset's seismic records featured relatively simple subsurface geology profiles—single horizontal and flat reflectors located between two homogeneous velocity layers—we expect the seismic error would be exacerbated when applied to a dataset with more complex subsurface geology profiles. Furthermore, as described in Section 1.5.1, when the assumption of horizontal and flat reflectors are not met, the fundamental concept of raypaths - their angles and distances traversed through the medium - should be revisited to compute the ray-traced timeshifts for static correction. As such, in addition to the increased water velocity inversion error arising from seismic records of complex subsurface geology profiles, a raypath-based static correction method (that is not NMO-based) will be less forgiving towards observed noise in the inverted profiles, as it explicitly computes the traveltime of the ray through each value in the water velocity profile, representing a layer of a certain depth (dependent on the granularity of the profile) in the ocean layer.

The numerical results of the tests in this chapter which becomes the benchmark quantities we aim to supersede in following chapters begin with the predicted seismic records

showing an error margin of about 25% compared to the ground truth records. Closer examination of the records, as illustrated in Figure 11, reveals that obtaining a 0.01s level of timeshift precision happens in very limited cases such as the zero offset , direct arrival. For other arrivals, the resulting timeshifts exceed 0.01s, reaching as large as 0.04s. Based on precision computations detailed in Section 1.4, a timeshift of 0.02s corresponds to depth resolution of 30m and a timeshift of 0.04s corresponds to a depth resolution of 60m. As time-lapse reservoir seismic monitoring requires a spatial resolution of 15m to image dynamic fluid flow processes within the reservoir [Lumley and Behrens, 1998], timeshifts larger than 0.01s are not precise enough for good imaging as they could result in spatial inaccuracies between 30m to 60m in migrated images. Thus, we conclude that inaccuracies in the velocity model inversion process in traditional statics correction lead to reduced time resolution in seismic data, potentially causing severe reflector positioning errors; in our case, these errors could be as large as 60m. This conclusion therefore indicates the need for further improvement of the current inversion process, based in the traditional model-centric paradigm for statics correction.

2.5 Conclusion

Traditionally, statics correction is a model-centric process that relies on the accurate inversion of geological velocity models as a prerequisite for its subsequent steps, of deriving and applying statics. For marine seismic data, this corresponds to the accurate inversion of the depth-dependent water velocity profile from the seismic record. Generally, statics correction for marine data is then implemented by NMO processing the data using the traveltime difference between the physical, unknown medium and hypothesized medium.

For a deeper understanding of the limitations of this model-centric paradigm for statics correction, we performed benchmark testing using deep learning for inversion from data space to model space. Specifically, we developed an innovative neural network by combining in sequence, various deep learning algorithms to transform 2D marine seismic records into 1D depth-dependent water velocity profiles of the ocean layer. The neural network was successful in producing good water velocity predictions that aligned well with ground truth profiles. However, the predictions were not entirely free from inaccu-

racies. Accordingly, when the predicted profiles were NMO-processed to seismic records, they yielded sub-optimal results when compared to ground truth seismic, only achieving the 0.01s precision level only in very limited cases (such as the zero offset, direct arrival), while other arrivals exhibited timeshifts as large as 0.04s, which is unacceptable for good imaging. Thus, there is a compelling need for further improvement in the accuracy of the predicted water velocity profiles.

This realization paves the way towards two approaches: first, enhancing the inversion algorithm for improved velocity model predictions making them viable for better time resolution in statics-corrected records; and second, investigating alternatives to the predominant model-centric framework for statics correction. We chose to pursue the second approach, and the remainder of this thesis focuses on examining a different paradigm of statics correction, which is based on a data-centric framework instead.

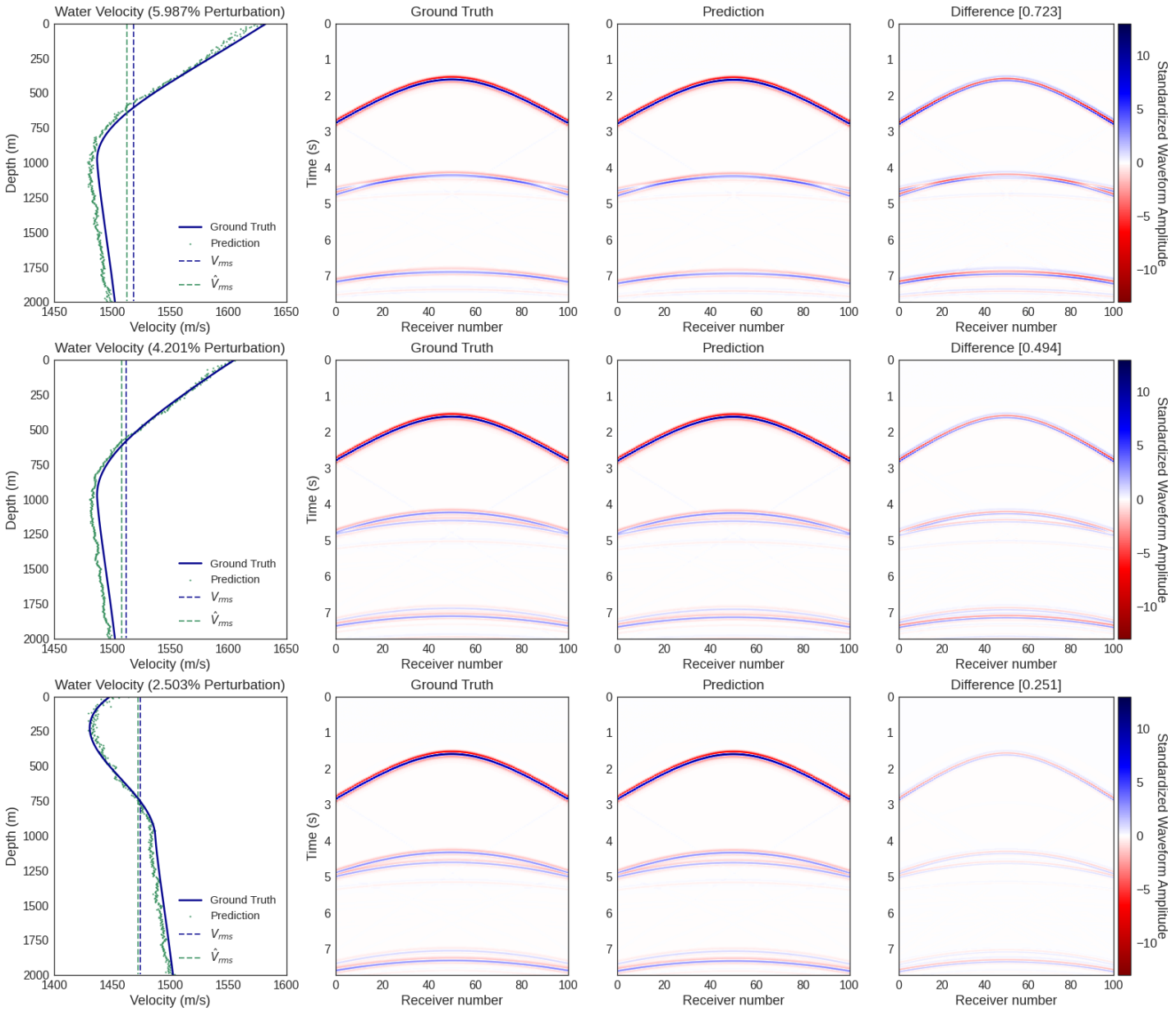

Figure 10: NMO-based forward model results for seismic records with distinct velocity models, consisting of the water layer and underlying subsurface geology. To evaluate the water velocity predictions, we NMO-processed the predicted profiles to seismic records, and calculated the difference between the predicted and ground truth records. **Column 1:** Ground truth and predicted depth-dependent water velocity profiles for three different velocity perturbations, with their respective RMS velocity values indicated by dashed lines. **Column 2:** Ground truth seismic records, from which the predicted water velocity profiles in Column 1 are inverted from. **Column 3:** NMO-based forward modeled seismic records. **Column 4:** Predicted minus ground truth seismic records (their difference).

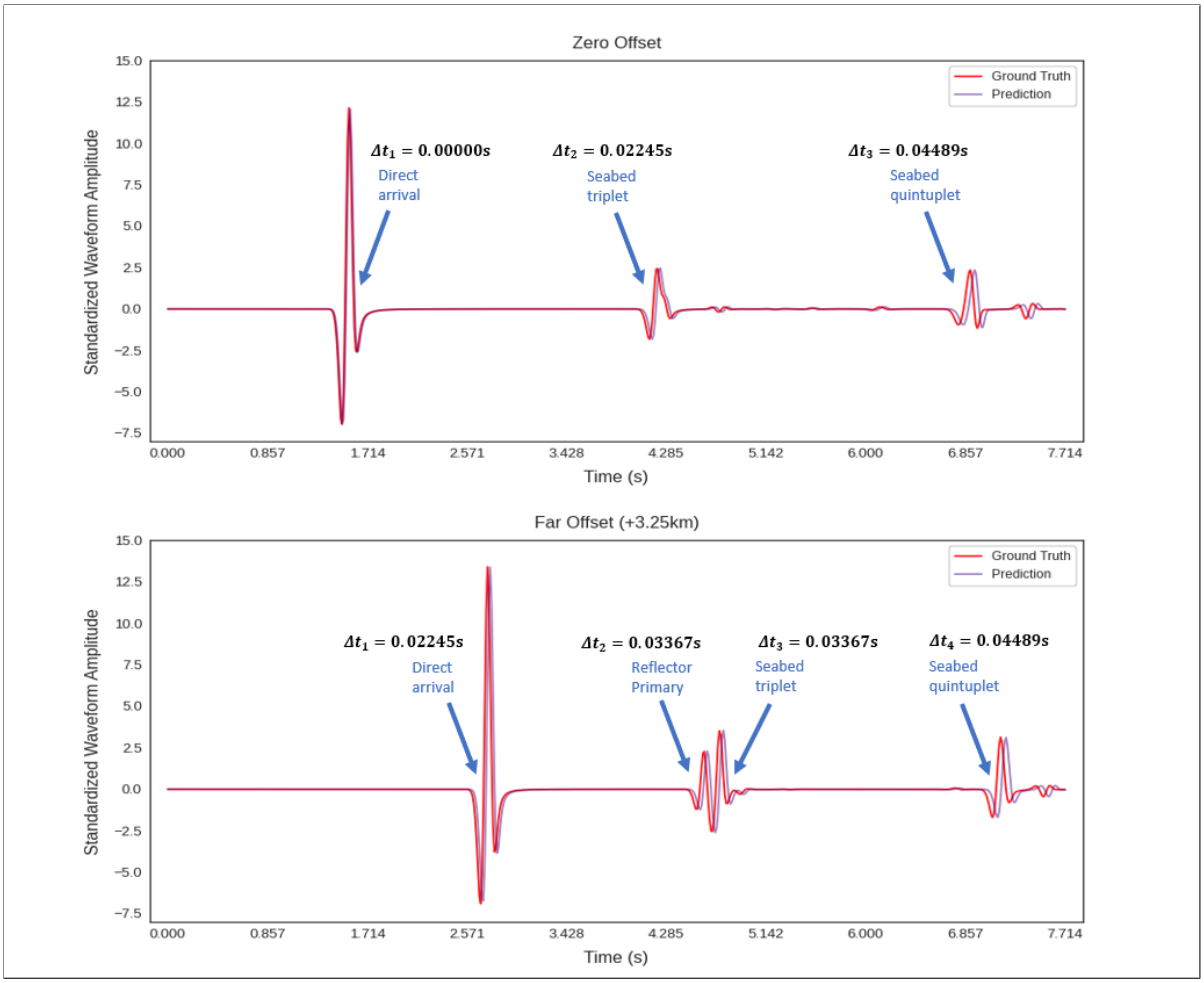

Figure 11: Timeshifts measured on ground truth vs predicted traces of the 5.987% perturbed water velocity sample from the test dataset. The traces shown are for the zero offset and far offset (3.25km) cases. Although the zero offset traces show better alignment than the far offset traces, most timeshifts for the trace pairs are non-zero and substantial, reaching values as large as 0.04s. This highlights the impact of inversion inaccuracies in the velocity model, which create traveltime inaccuracies in the forward modeled seismic data.

3 Single Horizontal Reflector

3.1 Introduction

Deepwater marine seismic measurements undergo statics that arise from tidal and seawater velocity variations that are spatio-temporal in nature. These statics corrupt the seismic data acquired from such surveys, leading to inaccuracies in 3D imaging and non-repeatability in 4D time-lapse monitoring of the subsurface. As outlined briefly in Section 1.3, the classical method for statics correction consists of a two-step workflow: First, deriving statics. Second, applying static corrections to the seismic measurements.

The problem with this workflow is that there are complications inherent to both steps of deriving and applying statics. The derivation of statics require an estimation of the time-variant acoustic (pressure) velocity though the seawater column (hereafter referred to as water velocity for simplification) which is obtained from either NMO-based evaluation of the direct arrival's traveltime curve [Lacombe et al., 2006], or direct measurement with a specific instrument such as the Pressure Inverted Echo Sounder (PIES) [Wang et al., 2012]. The application of offset-dependent (dynamic) statics correction requires applying a timeshift calculated from the difference between measured velocity water bottom arrival time and a known reference velocity water bottom arrival time on NMO corrected data (using the real water velocity), and subsequent reverse NMO correction (using the reference water velocity) [Lacombe et al., 2006]. This method works for simple subsurface geology that assumes a flat reflector or a known geological target, but require more intricate procedures such as a TauP inversion when complexities, such as a salt body, exist in the overburden [Huang et al., 2016]. These methods however, are not full-proof (significantly, dynamic statics correction and TauP inversion are only theoretically accurate when correcting for the primary reflection that has traveled through the water column once); thus, the derived and applied statics often have compromised accuracies. Furthermore, the workflow is computationally expensive and time consuming - especially because water velocities are constantly changing and require a separate static correction process for each velocity variation in a seismic stack.

The purpose of this chapter is to determine if the classic two-step workflow for deep-water statics correction can be replaced by a data-centric approach that bypasses the need to transform between the non-commensurate spaces of seismic data and velocity model space. Specifically, we implement the symmetric autoencoder (SymAE) algorithm to disentangle the effects of varying water velocity and coherent subsurface geology in seismic records - and subsequently, create statics-corrected seismic records, which are standardized with a reference water velocity profile of choice [Bharadwaj et al., 2020, 2022].

3.1.1 SymAE and Marine Seismic Survey Non-Repeatability

SymAE was designed to solve poorly controlled scientific experiments where acquired measurements suffer from corruption by unmodelled and uncontrollable nuisance variations [Bharadwaj et al., 2022]. A sub-class of such experiments involve variations in measurements, that occur across two different time scales: a slower scale and a faster scale. On the slower scale, the coherent information intrinsically tied to the physical state, varies. On the faster scale, the nuisance information that corrupts measurements of the physical state, varies. The separation of scales and associated information is expressed in Figure 12. For SymAE to effectively disentangle coherent and nuisance information within this experimental context, a certain time period is fixed in which the coherent information is assumed to be unvarying, and only the nuisance information varies on the fast time scale. Additionally, during this time period, multiple noisy (nuisance-corrupted) measurements of the coherent physical state can be acquired - these are called *instances* of a datapoint.

In practice, this contrast in scales is seen during deepwater marine seismic acquisition. Typically, a complete marine seismic survey using towed streamers containing sources and/or receivers takes within the order of weeks or months to complete [Mateeva, 2021]. This time-scale aligns with the emerging trend over the last decade and a half of frequent 4D reservoir monitoring,[11] with surveys repeated on-demand over weeks to months. This

[11]Prior to the last decade and half, 4D offshore monitoring focused on longer timescales, with surveys conducted several years apart, typically every 3 to 5 years. These surveys were mainly aimed at optimizing field development strategies and ensuring the operational licensing of exploration companies [Carpenter, 2014; Mateeva et al., 2015].

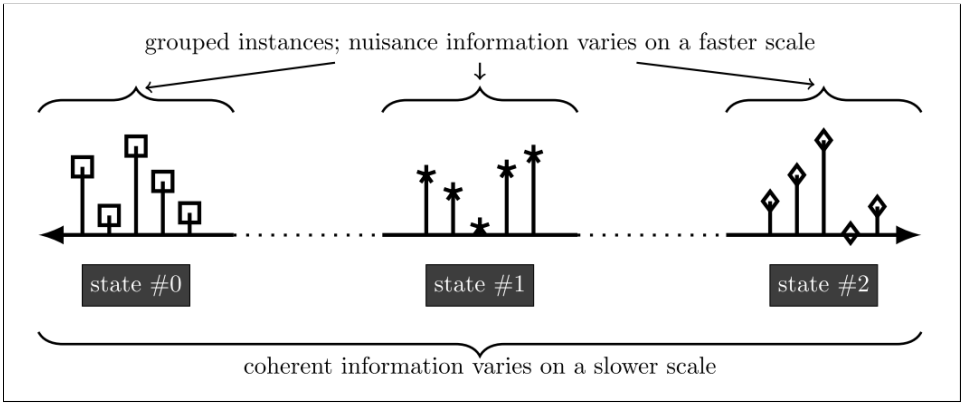

Figure 12: Separation of time-scales and associated information. Coherent information (marker shapes) tied to the physical state varies on a much slower scale compared to the nuisance information (marker distance from the baseline) which corrupts measurements of the physical state. Each measurement taken is represented as a marker in the diagram, and its disentanglement of coherent and nuisance information is the aim of SymAE. Image from [Bharadwaj et al., 2022].

frequency matches shorter reservoir dynamic timescales, providing critical information for optimizing reservoir injection and production processes, and ensuring the operational safety of these activities [Carpenter, 2014; Mateeva et al., 2015]. In contrast to the slower time-scale of reservoir changes, faster time-scale acoustic velocity variations - in the order of hours to days - are experienced by the water layer through which the seismic wave travels downwards from the sea surface (where the seismic source is located) towards the offshore reservoir. These variations are driven by waves and tides, which are influenced by the temperature, pressure, and salinity fields of the sea [Lynch et al., 1996] (See Section 3.2.1 for more information on the water velocity profile and the variations it undergoes).

If we break the survey into smaller time windows of about twelve hours, a source can be fired from the same position multiple times for the acquisition of multiple seismic measurements. Within this time period, it is reasonable to assume that the subsurface geology is unvarying whereas the overhead water velocity is varying. Thus, subsurface is represented as coherent information, and water velocity as nuisance information. The removal of water velocity nuisance information from the seismic measurements will significantly reduce the non-repeatability issues that propagate into time-lapse (4D) analysis from corrupted three-dimensional (3D) imaging. This is because water velocity variations introduce timeshifts into seismic traces, that if uncorrected, invert as inaccurate

velocity structures in the 3D image.

In order to correct for water velocity variation, we intend to map all water velocity profiles in the seismic measurements to a single water velocity profile, known as a reference profile. This will homogenize the effects of water velocity across instances. Consequently, differences in seismic measurements, acquired within a time window, is solely attributed to differences in the physical state of the subsurface geology.

3.2 Data and Methods

3.2.1 Data

The dataset used in the training and testing of SymAE consists of synthetic data generated by the GeoPhyInv toolbox, an acoustic and elastic wave equation solver for high-performance computations. A two-dimensional model of rectangular shape was used as our physical domain. It has 1301 gridpoints on the x-axis spanning a length of 6500m, and 1201 gridpoints on the x-axis spanning a depth of 6000m. The sea surface is positioned at a depth 0m, which is the top of the physical domain. The model is divided into an upper layer consisting of a water column and a bottom layer of bedrock; the subsurface we are interested in imaging.

The dataset is composed of seismic measurements from 1000 different subsurface models, which constitute the datapoints $\{X_i\}_{i=1,\ldots,1000}$. Each datapoint has 11 different water velocity models, which constitute the instances $\{\tau_k\}_{k=1,\ldots,11}$.[12] This results in a total of 11,000 models in the entire seismic dataset. Figure 13 shows an example of a model generated. The following sections describe the specifics of the subsurface bedrock and water column structure.

Subsurface structure

The subsurface extends from the seafloor, at a depth of 2000m, to the bottom of the

[12]For each datapoint, one instance within $\{\tau_k\}_{k=1,\ldots,11}$ is modeled as a reference instance, τ_{ref}, that follows a reference velocity profile (described in Section 3.2.1). The remaining 10 instances are variations from the reference velocity profile.

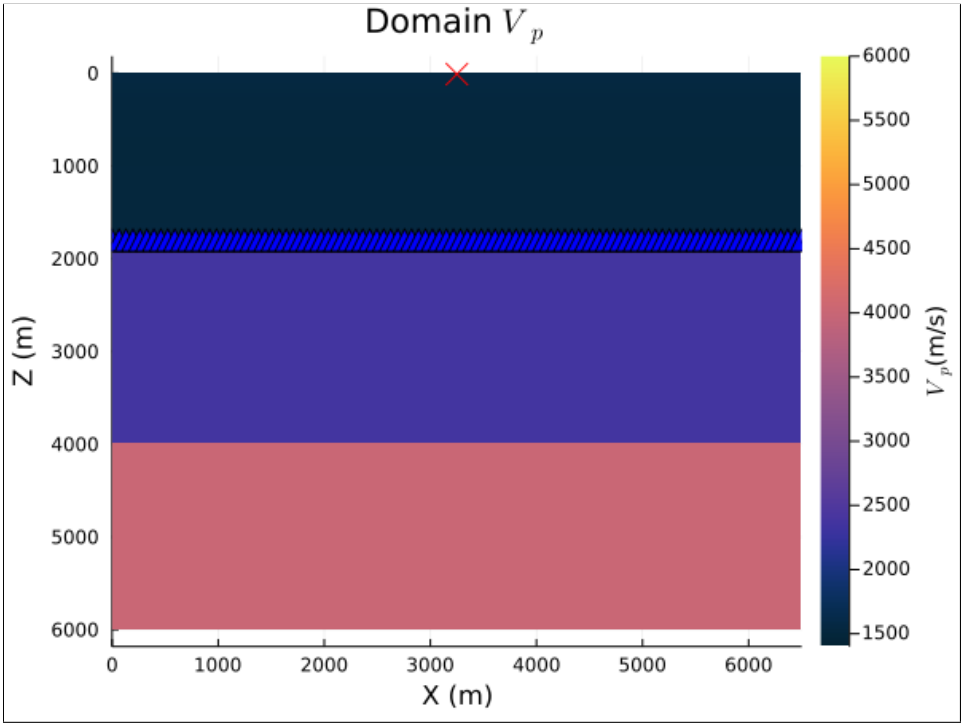

Figure 13: Visualization of a model used to create synthetic shot gather data. V_p here refers to the acoustic wave propagation velocity through the domain. The source, marked with a red x, is positioned in the water column at a depth of 10m. An array of equally-spaced receivers, marked with blue triangles, are positioned on the seafloor at a depth of 2000m. The seawater column appears to be vertically homogenous because the depth-dependent water velocity profile does not vary drastically enough for a visible change in the plot. However, as shown in Section 3.2.1, the modeled water velocity does indeed change with depth through the domain.

physical domain, at 6000m. Around the halfway point of total subsurface depth, a horizontal reflector is placed through the length of the domain. A slower seismic velocity is assigned for the region above the reflector (upper-layer), and a larger seismic velocity is assigned for the region below the reflector (lower-layer). Across datapoints, the position of the horizontal reflector is uniformly distributed between 3300m to 4900m; the values for the upper-layer velocity and lower-layer velocity is uniformly distributed between 1800ms^{-1} to 2850ms^{-1}; and 2850ms^{-1} to 5700ms^{-1} respectively.

Water column structure

The water column has a depth-dependent velocity function, that consists of two states: *reference* and *perturbed*. The reference water velocity follows the Hood's model, a 6th-order polynomial expressing velocity as a function of water depth, for temperate and

tropical ocean basins:

$$\begin{aligned} V_H(z) =& 1541.30 - 0.18026z + 2.12895 * 10^{-4}z^2 - 1.15430 * 10^{-7}z^3 \\ &+ 3.28150 * 10^{-11}z^4 - 4.62212 * 10^{-15}z^5 + 2.52598 * 10^{-19}z^6, \end{aligned} \tag{16}$$

where $V_H(z)$ is the Hood's water velocity as a function of depth, z. The Hood's model was derived by Advocate and Hood [1993] by combining the results of 14 velocimeter surveys across different locations in the northwest Gulf of Mexico, during the years 1988 to 1990. The velocimeter surveys provided measurements of the acoustic velocity profile in that location. Notably, the resulting Hood's water velocity profile varies non-linearly with depth. This profile is strongly dependent on the interaction of temperature, salinity and pressure (depth) [Del Grosso, 1974]. Besides these variables, the profile is also influenced by the presence of impurities (i.e., gas bubbles), and organic and inorganic matter [Advocate and Hood, 1993].

Additionally, water velocity is subject to other spatio-temporally varying factors such as tide, season, location and ocean current [Han et al., 2012]. To account for these variations, we create a perturbed water velocity profile, that takes the form of a cosine square modulation on the amplitude of the Hood's model (reference profile) for depths of 0m to 1000m:

$$V_m(z) = \cos\left(\frac{z}{1000} * \frac{\pi}{2}\right)^2 \frac{p}{100} V_H(z) \tag{17}$$

$$V_p(z) = V_H(z) + V_m(z), \tag{18}$$

$V_m(z)$ represents the modulation of the reference velocity $V_H(z)$, p is the percentage of cosine squared modulation applied, and $V_p(z)$ is the resulting perturbed water velocity from the sum of modulated and Hood's velocities. Based on discussions with scientists from Shell [Mateeva, 2021], p realistically ranges from -2% to $+2\%$. These limits are shown in Figure 14, with the Hood's velocity profile for comparison purpose. Perturbations to the Hood's velocity cease to exist below 1000m as this region has nearly constant temperature and pressure profiles, and is thus largely unaffected by seasonal variations. This stratum of the oceanic body is known as the *deep isothermal layer*, where water

velocity increases with depth primarily due to increasing pressure [Advocate and Hood, 1993].

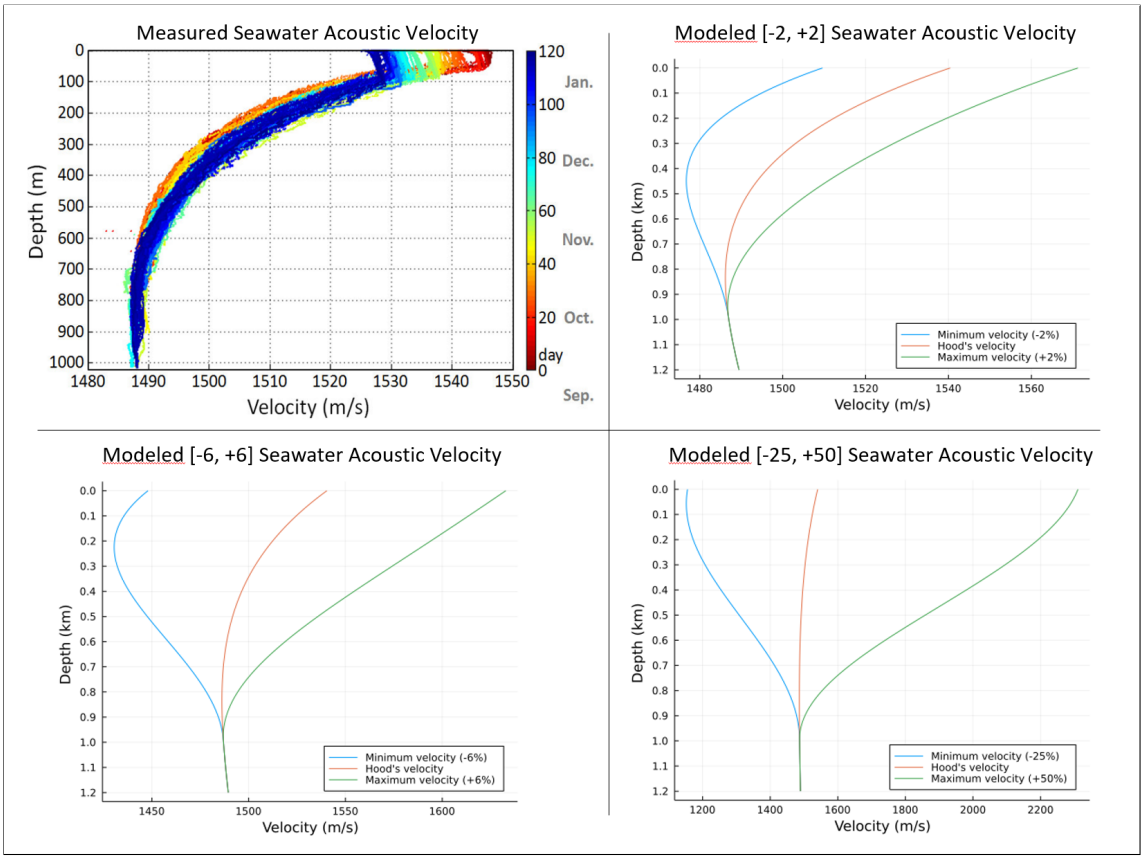

Figure 14: **Top-left:** Acoustic velocity profile of water column in the Gulf Of Mexico derived from remotely operated vehicle (ROV) temperature-pressure-conductivity measurements - from September 2011 to January 2012. Image from Wang et al. [2012]. **Top-right:** Hood's velocity (reference) profile and limits of realistic perturbations. The minimum and maximum velocities corresponds to p values (see Equation 17) of -2% and $+2\%$ respectively. **Bottom panel:** Supplementary perturbation ranges of the Hood's velocity profile used for testing of SymAE: $-6\% \leq p \leq +6\%$ and $-25\% \leq p \leq +50\%$.

Although the velocity perturbation p is realistically limited to $[-2, +2]$, to test the performance of SymAE on larger timeshifts, the perturbation range was expanded by creating additional datasets with p of $[-6, +6]$ and $[-25, +50]$ (Figure 14).

Shot gather simulations. From the physical models described, shot gather data is generated with the following specifications:

Finite difference simulations. The 2D acoustic wave equation solved with a second order in space and second order in time finite-difference modeling. Perfectly

matched layers (PML) used at all model boundaries, except the z= 0 free surface, which has a Dirchlet boundary condition of 0. Simulations ran for a time range of 8.8456s, with 0.00062s time steps; with grid-spacing of 5m in x and z directions.

Acquisition geometry. Single source at a depth of 10m from the top of the domain (within the water column). 100 receivers, equally spaced, positioned at seafloor depth of 2000m.

Source signature. Ricker wavelet with peak frequency of 6.78Hz and maximum frequency of 20.23Hz.

3.2.2 Method

As described in detail in Section 1.5.4, the autoencoder can be viewed as a dimensionality-reduction method as it maps high-feature data to a low-feature latent space. This latent space serves as a bottleneck, offering a compressed embedding of the datapoints [Wang et al., 2016; Spinner et al.]. The SymAE deep learning algorithm is based on the autoencoder but has the unique functionality of disentangling nuisance variations and coherent information, from a group of measurements, in the latent space. This separation is achieved by passing the input data into two encoders: the first encoder (CEnc), decodes coherent information; the second encoder (NEnc), decodes nuisance information. CEnc leverages the permutation symmetry of the coherent information across all instances in a vector (datapoint), by summing across the transformed data on the instance dimension - thereby constraining the encoder to *only* extract coherent information. NEnc however, does not sum across the transformed data, but extracts and retains instance-specific nuisance information. The ability of the nuisance encoder to exclude any coherent information is enforced by adding noise to the output of the nuisance encoder by Gaussian dropout.[13] In addition to being an integral part of SymAE's design, Gaussian dropout also acts as a regularizer that reduces the generalization error of SymAE and prevents over-fitting. Figure 15 illustrates the complete SymAE model with the two encoders described.

[13]The Gaussian dropout method multiplies a random variable, $r_q \sim N(1, \frac{q}{1-q})$, with the output of the activation function, a_i of a hidden node i, with probability (rate) q. The resulting activation, $r_q a_i$ is thus perturbed from a_i [Srivastava et al., 2014].

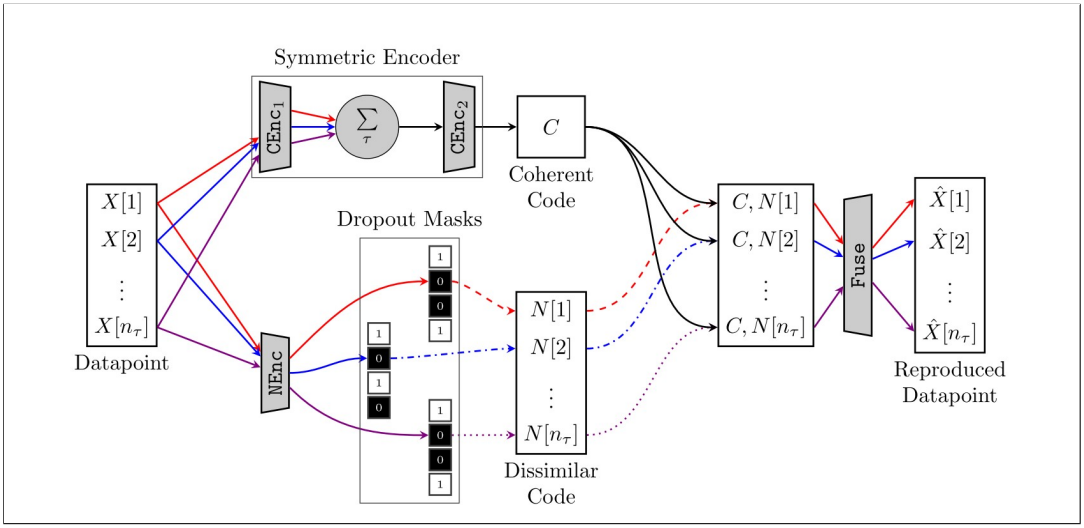

Figure 15: Macro-view of the SymAE learning algorithm network architecture. Total information content of instances of a datapoint are represented as solid coloured arrows. The summation function in the Symmetric Encoder extracts coherent information (solid black arrow) and the Dropout Masks applied to the output of the Nuisance Encoder (NEnc) enforces the extraction of *remaining* nuisance information (dotted colored arrows). The disentangled information is combined and decoded (via Fuse) to reproduce the input datapoint. Image from [Bharadwaj et al., 2022].

Remember that each datapoint X_i, SymAE is trained on, is a vector of instances $\{\tau_k\}$. A datapoint exists in a high-feature space, and is represented by low-feature coherent information that is symmetric across all instances, and low-feature nuisance information that is varying across instances. Once the latent codes are produced, the decoder fuses the instance-specific nuisance code with the vector-extensive coherent code, and non-linearly upsamples the fused latent code to reconstruct the original datapoint. SymAE is trained to minimize a loss function of the form

$$L(\theta_{\text{Enc}}, \theta_{\text{Dec}}) = \frac{1}{n}\sum_{i=1}^{n}\left(X_i - \text{Dec}[\text{Enc}(X_i;\, \theta_{\text{Enc}});\, \theta_{\text{Dec}}]\right)^2, \tag{19}$$

where θ_{Enc} and θ_{Dec} denote the parameters (weights) of SymAE encoders and decoder; and n, the number of datapoints. This loss function is equivalent to the Mean Squared Error (MSE) function.

Redatuming

SymAE's ability to disentangle information in the latent space can be leveraged to generate hybrid datapoints from chosen coherent and nuisance codes. Once SymAE has been

trained on a dataset to extract coherent and nuisance codes, the coherent code from any datapoint can be combined with the nuisance code from any instance (of any datapoint) of a similar dataset. Redatuming is pivotal for our purpose of fusing the coherent code of the subsurface geology, with the nuisance code of the reference water velocity to produce corrected shot gathers. Subsequently, applying SymAE's decoder (Fuse) to this hybrid latent code will yield a hybrid instance - a shot gather of a wavefield that has traveled through the new medium with the chosen subsurface geology and water velocity profile - as expressed in equation (20).

$$X_{i\to j}[\tau] = \text{Fuse}[\text{Cenc}(X_j); \text{Nenc}(X_i[\tau])] \tag{20}$$

X_i and X_j are two different datapoints. $\text{Cenc}(X_j)$ represents the coherent code from X_j, which is symmetric across instances. $\text{Nenc}(X_i)[\tau]$ represents the nuisance code from $X_j[\tau]$, which is instance-specific. The fusion and decoding of these codes, generate the redatumed datapoint, $X_{i\to j}[\tau]$.

3.2.3 Training SymAE

To summarize, we created three datasets with different p, percentage perturbation, ranges: [-2,+2], [-6,+6], [-25,+50]. Each dataset consists of 1000 datapoints X_i; each datapoint consists of ten perturbed instances and one reference water velocity instance, τ_i. From each dataset, 700 datapoints were allocated for training, 200 datapoints for validating and 100 datapoints for testing SymAE.

SymAE was trained on datapoints that only contained perturbed instances. Thus, the inputs and outputs to SymAE consisted of X_i for $i = 1, ..., 700$ and τ_k for $k = 1, ..., 10$. Additionally, training was performed with a batch size of 2 datapoints (primarily to fit GPU memory requirements) and an Adam optimizer with an initial learning rate of 0.001. During validation, the expressivity of encoders was observed to have the most impact on results. Hence, this was the hyperparameter we focused on for tuning.

Controlling and balancing expressivity

Expressivity is understood as how the architectural properties of a neural network affect

the functions it is able to compute, and ensuing performance [Raghu et al., 2017]. The expressivity of the encoders is largely influenced by the dimensionality of their latent spaces. Additionally, for the nuisance encoder, the Gaussian dropout applied to its latent output also significantly influences its expressivity. One way to gauge expressivity is to examine the training and validation curves for signs of overfitting or underfitting. Naturally, a network that is more expressive will tend to overfit the data; as more parameters (activation units) enable the computation of more complex functions [Raghu et al., 2017].

For SymAE, tuning expressivity comprises not only in finding the right latent space dimensionality of the individual coherent and nuisance encoder, but also in balancing the expressivity of coherent and nuisance encoders. If the nuisance encoder were to have a much larger expressivity compared to the coherent encoder, this would drown the ability of the coherent encoder to capture any information. As a result, SymAE would behave more like an autoencoder as its ability to capture and disentangle symmetric and coherent information is compromised, and there would effectively only be one latent space.

There is no formulaic method to tune expressivity (or any other hyperparamter), and often a tuning strategy is found by trial and error. The tuning strategy that worked for us was:

Step 1: Finding a latent space dimensionality for the coherent encoder, that is balanced, with respect to the nuisance encoder. This can be ensured by checking that a redatumed instance has the right coherent information from one datapoint, and nuisance information from an instance of another datapoint.

Step 2: Fixing the dimensionality of the coherent encoder.

Step 3: Fine-tuning the expressivity of the nuisance encoder by adjusting its latent dimension and dropout rate, in tandem. More latent dimensions increase expressivity, whereas fewer latent dimensions reduce expressivity. We inspected the training curve to ensure it does not overfit or underfit the data, whilst producing the least MSE loss.

This tuning strategy had to be applied individually for each training dataset of different p ranges. The final hyperparameters chosen are compiled in Table 2. As alluded to before, large p ranges require more expressivity, and thus more nuisance latent dimensions.

Dataset Range	NEnc Dimension	Dropout Rate
[-2, +2]	15	0.5
[-6, +6]	22	0.65
[-25, +50]	100	0.5

Table 2: When tuning hyperparameters of SymAE, we focused on fine-tuning the nuisance encoder (NEnc) latent dimension and the Gaussian dropout rate. The chosen values, which yield best validation and test results, are shown here.

3.3 Results

Having trained and tuned SymAE on three different datasets with water velocity perturbation ranges of $[-2, +2]$, $[-6, +6]$ and $[-25, +50]$, the ability of SymAE to correct for the perturbed velocity of an unseen instance (from datapoint X_{901}) such that its water velocity profile matches that of the reference Hood's profile is tested.[14] Another unseen instance (from datapoint X_{987}) arising from a different subsurface geology than our perturbed instance, but with the desired Hood's water velocity profile, is provided for redatuming - this instance is called the auxiliary record. Equation (21) expresses redatuming of this test case, to produce the corrected datapoint, $X_{901\to987}[\tau_{\text{ref}}]$. Figure 16 shows the auxiliary record, $X_{987}[\tau_{\text{ref}}]$), and Figure 17 shows the results of redatuming an instance of datapoint X_{901} across all three datasets.

$$X_{901\to987}[\tau_{\text{ref}}] = \text{Fuse}[\text{Cenc}(X_{901}; \text{Nenc}(X_{987}[\tau_{\text{ref}}])] \tag{21}$$

For a more detailed inspection of the correction performed by SymAE on traces, the Moving Window Cross-Correlation method was used to measure the timeshifts between perturbed and reference, and redatumed and reference traces.[15] We measured the timeshifts

[14] All instances used in this testing stage belong to the set of (testing) datapoints that SymAE has not seen during previous training and validating stages.

[15] The Moving Window Cross-Correlation is a commonly used technique to measure arrival time

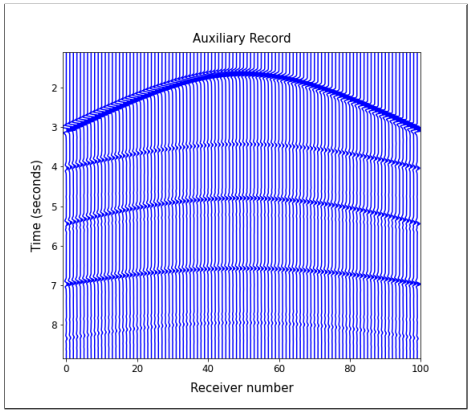

Figure 16: Auxiliary shot gather resulting from a subsurface model with the Hood's water velocity profile (reference velocity). Its nuisance code is a low-dimensional representation of the Hood's profile and therefore, is used by SymAE to redatum shot gathers with a velocity profile that is perturbed from the Hood's profile.

on the $[-6, +6]$ dataset, specifically on the $+6\%$ perturbation instances of datapoint X_{901}. This dataset range and perturbation value were chosen as SymAE performs well on this dataset (shown in Figure 17), and the timeshifts are larger and more discernable compared to those observed in the $[-2, +2]$ dataset. Figure 18 shows the timeshifts before and after redatuming, for both the 0km (zero) offset and 3.25km (far) offset traces.[16] For each trace, the timeshifts between reference and perturbed traces vary with arrival times of waveforms and distance from the source (offset). $\Delta t_1, \Delta t_2, \Delta t_3, \Delta t_4, \Delta t_5$ are the timeshifts of the direct arrival, seabed triplet (multiple), seabed quintuplet (multiple), reflector primary and reflector multiple, respectively. From the figure, for both offsets, the timeshifts prior to redatuming have non-zero values; after redatuming, these timeshifts are zero. Thus, the perturbed traces were aligned to the reference traces - redatuming has corrected for the water velocity variation induced timeshifts.

To further understand the performance of SymAE, timeshifts on a new test case (datapoint X_{906}), with a different subsurface than X_{901}, were measured (Figure 19). Unlike the previous test, here waves from the reflector arrive after the seabed multiples. SymAE

differences in seismic traces [Mikesell et al., 2015], [Liu et al., 2010]. In the continuous domain, consider f and g to be functions of t. The windowed-signals are $f(t; y) = g(t)w(t - y)$ and $g(t; y) = g(t)w(t - y)$, where $t - y$ specifies the location of the windowed function on the signal. The cross-correlation of these windowed signals is $(f \star g)(\tau) = \int_{-\infty}^{\infty} f(t; y)g(t + \tau; y)dt$. For our purpose, discretizing this equation and taking the lag of maximum cross-correlation of windowed signals yields the timeshift between $f(t)$ and $g(t)$ within that window (local timeshift), whereby we set the reference trace as $f(t)$ and the perturbed trace as $g(t)$.

[16]The 3.25km far offset trace lies at the edge of the physical domain, and was collected by the 100 receiver.

is able identify this inversion of arrival times, and perform the corresponding timeshift corrections. However, in the far-offset redatumed trace, there exists a non-zero timeshift of 0.01382s, in the reflector primary; this was corrected from the original timeshift of 0.02764s between reference and perturbed traces. Collectively, these results reveal that the timeshifts strongly converge to zero, however it is not certain *all* timeshifts reach absolute zero. Notably, this finding demonstrates the superior data-centric performance of SymAE compared to our model-centric benchmark test. In the benchmark test, most arrivals fail to achieve the required time precision of 0.01s, whereas SymAE achieves the 0.01s level of precision for all wave arrivals.

To quantify the performance of SymAE, we compute the L^2 norm of residuals between perturbed and reference, and redatumed and reference instances across all test datapoints. These norms are used to compute the gain of each instance: $\frac{||\text{reference-perturbed}||_2}{||\text{reference-redatumed}||_2}$. The gains are averaged over all instances to obtain the mean gain achieved by SymAE. Additionally, to probe (and compare with) the performance of SymAE on datasets outside of its training range, we cross-test SymAE trained on one dataset range and tested on a different dataset range. The results are compiled in Figure 20. SymAE achieves a gain > 1, meaning overall reduction in timeshifts, for all testing ranges. Significantly, SymAE performs best (maximal gain) when trained and tested on datasets of equivalent range.

Since gain represents the relative improvement in timeshift alignment between redatumed and perturbed states; to assess absolute performance independent from the influence of perturbed timeshifts, we compute and average the normalized residual norm (test error), $\frac{||\text{reference-redatumed}||_2}{||\text{reference}||_2}$, between redatumed and reference instances, across all test datapoints (Figure 21). The residual norm is simply a measure of how well the trained network generalizes to new data [Kawaguchi et al., 2020]. Generalization is highest for the $[-2, +2]$ SymAE tested on $[-2, +2]$ data, and this value is close to the generalization of the $[-6, +6]$ SymAE tested on $[-6, +6]$ data. However, compared to the aforementioned cases, the generalization ability decreases by approximately a factor of two for the $[-25, +50]$ SymAE tested on $[-25, +50]$ data.

3.4 Discussion

Generalization can be understood as the formulation of general concepts based on essential features common to specific examples [Mitchell et al., 1986].[17] From SymAE's training dataset which highlights timeshift variation due to underlying variation in seawater velocity, the main general concepts to be learned consists of:

1. Identification and disentanglement of coherent information depicting subsurface geology from nuisance information depicting water velocity variation.

2. Sufficiently-expressive encoding of coherent and nuisance information in the latent space.

3. Appropriate functional complexity (from the composition of non-linear activation functions) required to reconstruct waveforms in shot gathers, based on encoded coherent and nuisance information.

If SymAE learns (finds) these general concepts from the space of all possible concepts via training, then the learned SymAE generalizes well; the generalization gap[18] is minimized and SymAE is able to make good predictions on test data.

We evaluate the predictions on test data by performing redatuming - which necessitates the learning of all three general concepts - on perturbed instances. Good predictions entail the reconstruction of all arrival waves and strong convergence of timeshifts (with respect to reference trace) to zero in redatumed instances. The results show that SymAE generalizes well when trained and tested on $[-2, +2]$ and $[-6, +6]$ datasets, but its performance declines on the $[-25, +50]$ dataset.

The weaker performance on the $[-25, +50]$ dataset could be attributed to several reasons. First, the training dataset could be providing compromised knowledge of features relevant to general concepts being learned, specifically the concept of distinguishing the

[17] This understanding elucidates how generalization is actually an inductive process, where general laws are induced from particular examples [Mingers, 2012].

[18] The generalization gap can be understood as the difference between the model's performance on training data and its performance on unseen data drawn from the same distribution. More accurately, the generalization gap is the difference between expected and empirical risk of a learning algorithm on a distribution of data [Kawaguchi et al., 2020].

right coherent information. Higher water velocity perturbations cause larger changes in the seismic isochrone shape (due to the large bending of ray-paths) [Han et al., 2012], which may cause the seismic energy to penetrate weakly or not at all in some areas of the subsurface [Kazemi et al., 2020], leading to non-uniform illumination of the horizontal reflector. Thus, the horizontal reflector is classified as nuisance information rather than coherent information. Second, we observe that increased water velocity perturbation require increased latent dimensions in the nuisance encoder, suggesting the need for higher expressivity in the learning algorithm. Besides increasing the nuisance latent dimension, this can be achieved by increasing the width and depth (or changing layer types) of SymAE's current architecture, to increase the complexity of the computed function, $f_{A(S)}$.[19] Third, the trainability of this network could be compromised. Finding a good minimizer of the loss function depends on the behavior of the loss-landscape (chaotic, or having more convex-like structure) and minimization trajectory (determined by network initialization and optimizer) in the landscape. It is possible that the $[-25, +50]$ SymAE has a more chaotic landscape, whereby changes to network architecture (i.e., introducing skip-connections) and size (i.e., increasing width) could flatten-out the chaos - allowing for improved loss minimization [Choromanska et al., 2015; Li et al., 2018].

Nevertheless, as illustrated in Figure 14, it is unlikely that real seawater velocity perturbations will reach the limits of the $[-25, +50]$ (or even the $[-6, +6]$ range). Thus for 1D real-life applications, the architectural design and optimization decisions used in the current version of SymAE should suffice. This is with the caveat that the real subsurface structure and variation of structure between datapoints is similar to those that we modeled in this study. It is probable that the real subsurface has multiple reflectors with different depths and dips, and inhomogeneities within layer velocities. In this case, the training dataset S should be modified to account for the subsurface differences (either by real data or more-realistic simulations), so that a new $f_{A(S)}$ (learned-SymAE), which still learns the general concepts listed, can be found.

From the results of this study, we induce the following conclusions:

1. SymAE is able to disentangle the coherent and nuisance information identified in

[19] Expressivity is analagous to the complexity of the function $f_{A(S)}$ computed by the learning algorithm (neural network architecture) A when trained on dataset S [Kawaguchi et al., 2020]

the context of deepwater seismic acquisition.

2. In the latent space, SymAE is able to combine selected information points, to produce a hybrid latent point which has the selected subsurface and water velocity configurations.

3. Through SymAE's process of reconstruction from hybrid latent codes (by the learned-decoder), the timeshifts of perturbed instances are corrected for, leading to a strong convergence of arrival times to those of the reference instance.

Thus we have determined that SymAE is a learning algorithm capable of performing offset and depth dependent timeshifts in seismic measurements. This highlights SymAE's ability in signal processing for the non-linear classification of different arrivals: direct arrival, primary reflections and multiples. This is done by using different convolutional kernels that slide on the time-axis of the traces of the 2D seismic record to apply statics (timeshifts) pertaining to the type of arrival the sliding window encounters. SymAE's capability in identifying and correcting specific timeshifts related to different arrivals is learned directly from the seismic data (with shot records serving as inputs of SymAE) and does not depend on an understanding of the underlying physics of wave propagation and multiple scattering in the velocity model.

This foundational understanding and confirmation is needed before using SymAE to tackle real environments, which introduce additional complexities to the problem. These complexities include increased heterogeneity in subsurface velocity structure, lateral (x-direction) water velocity variation and the evolution (subsidence /uplift) of seafloor depth.[20] Addressing these challenges is beyond the scope of this study, but it is a worthwhile pursuit for practical applications of this algorithm in real deepwater acquisition.

[20]The changes in seafloor depth directly influence and co-evolve with the position of OBN receivers, which are embedded within the seafloor [Laurson, 2024]. In a 2007 OBN survey conducted by Fairfield Industries for Shell in the Gulf of Mexico, it was observed that the average positional change of 16 nodes over a 60-day marine acquisition period was 5.3 meters. The resulting depth-migrated 2D image difference sections (generated from the acquired OBN seismic data) had an NRMS of approximately 10% for P-wave signals, and approximately 50% for S-wave signals. This noise from changing receiver positions manifested as substantial depth-misalignment of the reflectors in the resulting 2D images. Thus, it is crucial to remove the effects of OBN position variation to achieve accurate subsurface imaging and ensure good 4D repeatability [Hays et al., 2008].

3.5 Conclusion

Conventional deepwater static correction consists of a two-step workflow of deriving and applying water velocity static corrections to the seismic dataset. Due to challenging complications in this workflow, the aim of our study is to bypass the conventional workflow with a deep learning algorithm that can perform the necessary offset and depth dependent timeshifts. We created a seismic dataset that highlights timeshift variation due to the underlying variation in seawater velocity, and used this to train SymAE; an autoencoder-based learning algorithm that has the capability of disentangling symmetric (coherent) and nuisance (incoherent) information from a dataset, and encoding them in two separate latent spaces.

During training, SymAE learns to encode varying seawater velocity as nuisance information and unchanging subsurface geology as coherent information. With the learned-SymAE, nuisance information encoding a reference seawater velocity profile from one seismic datapoint, and coherent information encoding a subsurface velocity profile from another seismic datapoint, can be combined in its latent space, to construct a *redatumed* datapoint. Thus, we redatumed an auxilliary datapoint containing a reference seawater velocity profile (Hood's model), with test datapoints containing instances of *perturbed* seawater velocity profiles. Each test datapoint also has an associated reference instance, with the Hood's velocity profile, that is not passed into SymAE but used to evaluate the redatumed predictions of SymAE.

Comparisons of the redatumed and reference instances yield a strong convergence of the arrival times of the redatumed waves to those of the reference waves; the non-zero timeshifts between perturbed and reference instances strongly converge to zero valued timeshifts between redatumed and reference instances, satisfying the 0.01s resolution in time for accurate 3D imaging. The perturbed timeshifts are hence, corrected for. Therefore, we conclude that is SymAE is a learning algorithm capable of performing offset and depth dependent timeshifts in seismic measurements due to depth-dependent seawater velocity variations.

This conclusion is the foundational understanding and confirmation that enables further

research into using SymAE for practical applications. A host of other complexities are introduced by the real environment during deepwater seismic acquisition. These complexities can be layered on one at a time into the training dataset, and the performance of SymAE investigated with each modification to the dataset.

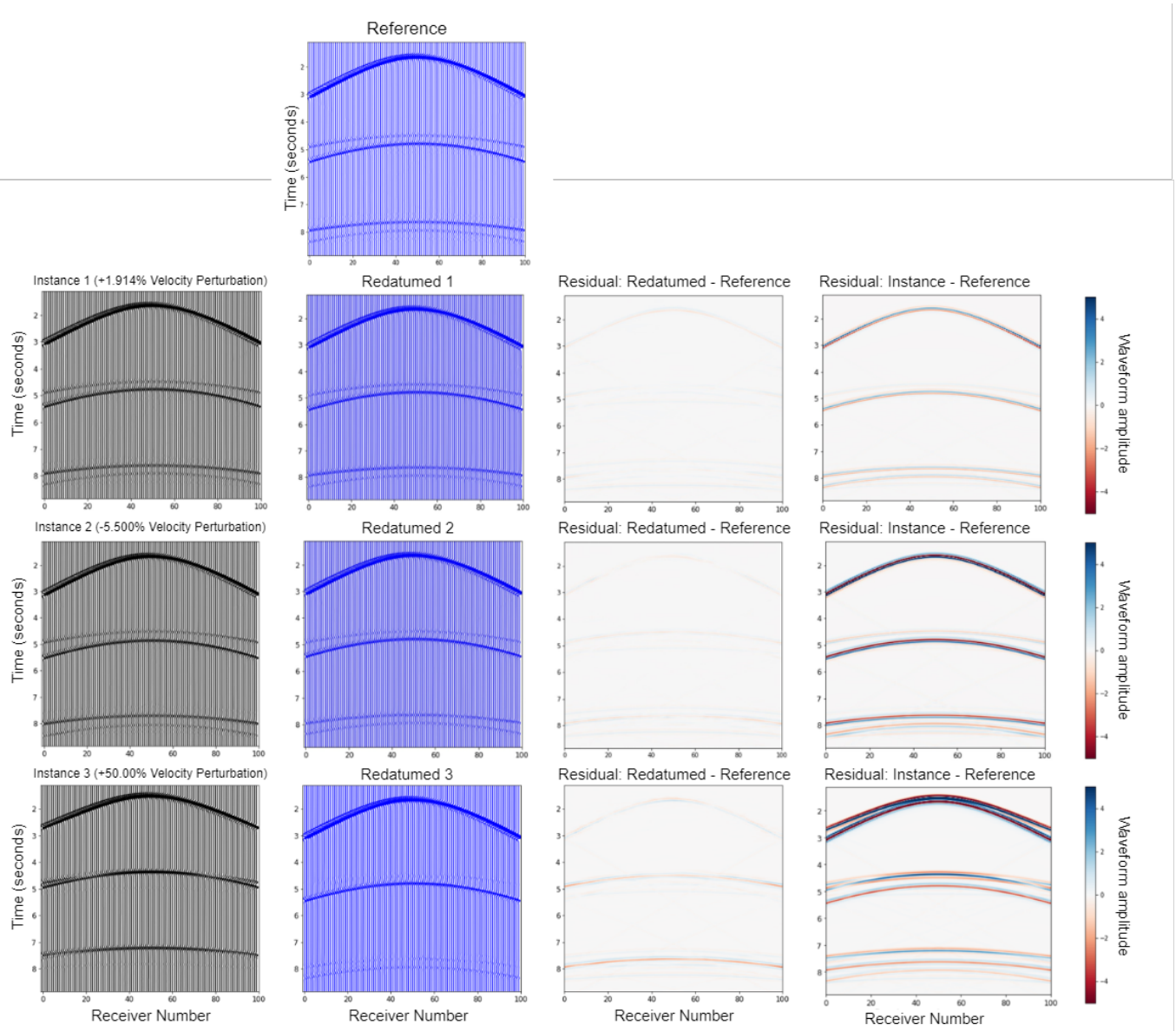

Figure 17: Redatuming results for different water velocity perturbation instances. The topmost row of the second column shows the reference shot gather; this shot gather arises from the same subsurface geology as the perturbed instances, however its water column follows the Hood's velocity profile. Thus, the aim of redatuming is to map perturbed instances to reference instance. SymAE performs well on the $+1.914\%$ and -5.5% perturbations; but its performance declines at a much higher perturbation of $+50.00\%$, where it displays comprised ability in capturing the 2 and 4 arrivals. **Column 1:** Perturbed instances; top-to-bottom row sampled from the $[-2, +2]$, $[-6, +6]$, $[-25, +50]$ datasets respectively. **Column 2:** Corresponding redatumed instances from perturbed instances of Column 1.**Column 3:** Residuals between redatumed and reference instance. **Column 4:** Residuals between redatumed and perturbed instances.

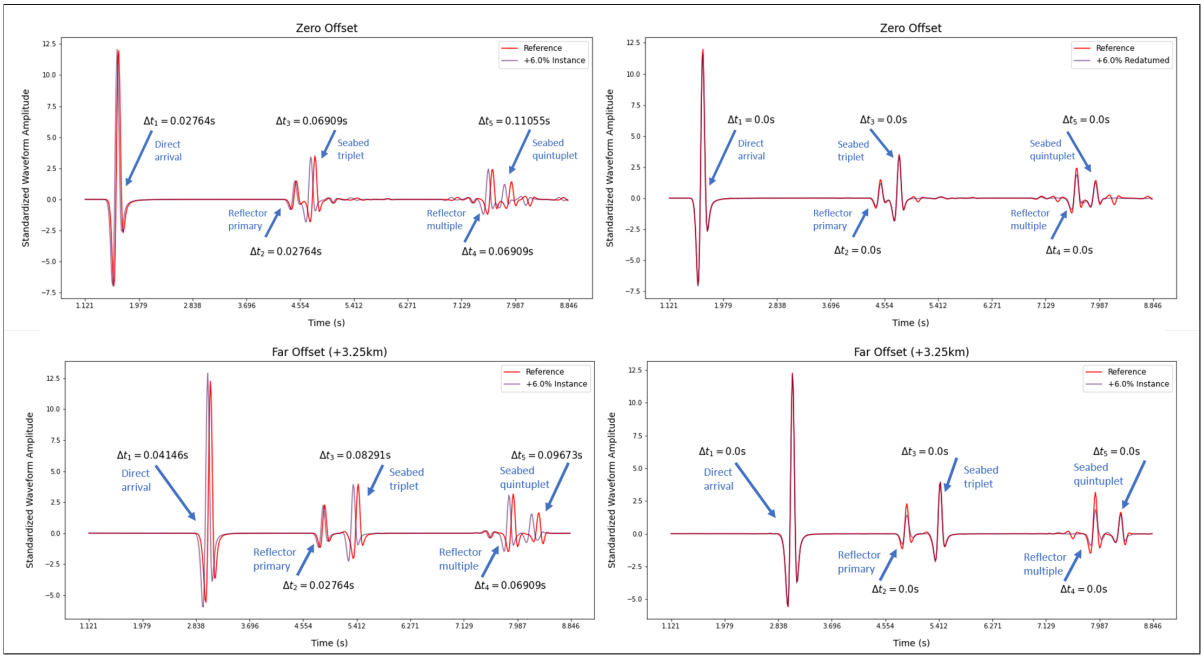

Figure 18: Results of redatuming the +6% perturbed instance of test datapoint X_{901}. The timeshifts compared to reference trace, before and after redatuming, for zero offset and far offset are shown. All timeshifts *reach* zero.

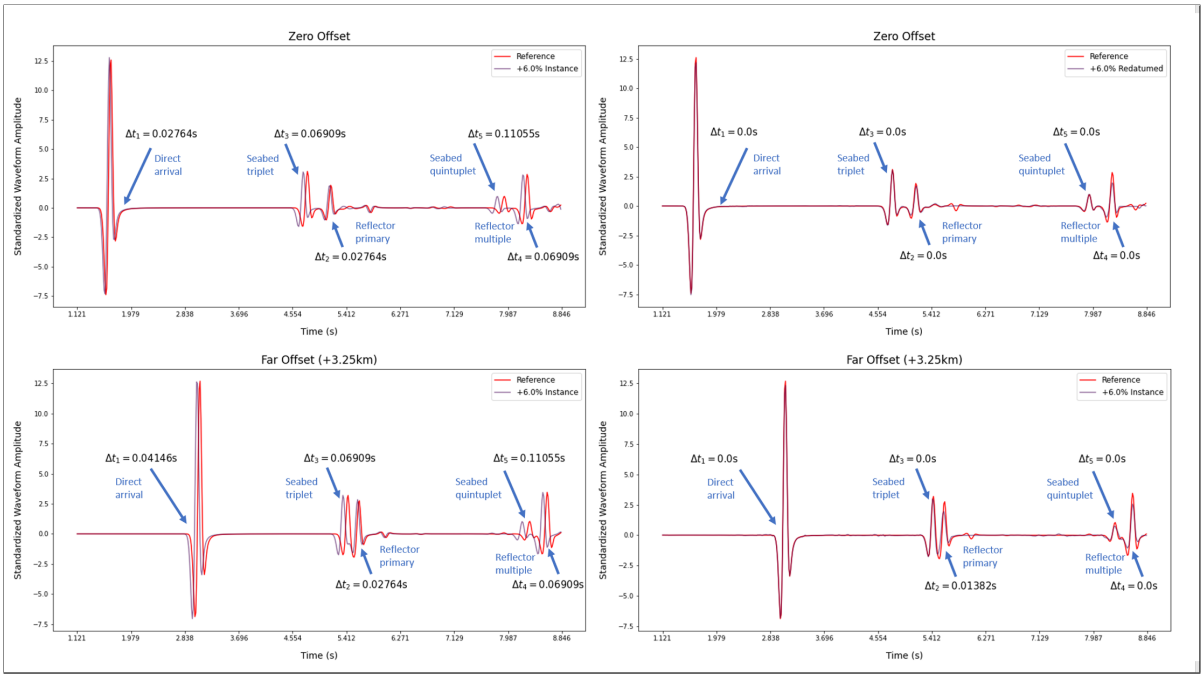

Figure 19: Results of redatuming the +6% perturbed instance of another test datapoint, X_{906}, with different subsurface velocities than X_{901}. The timeshifts compared to reference trace, before and after redatuming, for zero offset and far offset are shown. All timeshifts *converge* to zero.

Gain across Different Ranges of Data			
Test Data	[-2,2] SymAE	[-6,6] SymAE	[-25,50] SymAE
[-2,2]	2.819	2.267	1.172
[-6,6]	3.453	6.704	3.243
[-25,50]	1.911	4.276	6.894

Figure 20: Average gain achieved upon redatuming test data. Higher values imply higher redatuming impact when correcting timeshifts between perturbed and reference data.

Normalized Residual Norm across Different Ranges of Data			
Test Data	[-2,2] SymAE	[-6,6] SymAE	[-25,50] SymAE
[-2,2]	0.084	0.104	0.201
[-6,6]	0.190	0.098	0.202
[-25,50]	0.721	0.322	0.200

Figure 21: Average normalized residual norm achieved upon redatuming test data. Lower values characterize better generalization.

4 Interface Model

With the previous experiments in Chapter 3, we have come to understand SymAE's capacity for marine statics correction in a qualitatively pristine subsurface model that creates reflected arrivals with a single flat and horizontal interface. Here, we build upon the experiments in Chapter 3, by adding a layer of complexity to the subsurface velocity profiles - with the presence of multiple flat interfaces of distinct depths and dips. Hence, the seismic dataset used in experiments of this chapter are altered from the previously used horizontal reflector dataset, and referred to as the interface model dataset.

The experimental workflow follows that of Chapter 2: first, the training of SymAE on the interface model dataset - and parallel hyperparameter tuning of the network during this process, using supportive generalization information from testing with the validation dataset; second, the redatuming of subsurface geology and water column profiles of unseen seismic records from the test dataset; third, the comparative analysis of redatumed seismic records to reference seismic records.

Our overaching aim is to determine if SymAE is able to handle the heightened complexities and features introduced by this more intricate geological case into the seismic data. As outlined in the Conclusion of the preceding chapter (Section 3.5), our future research direction for using SymAE in practical applications involves gradually integrating realistic complexities - especially those pertaining to the marine geological environment and seismic acquisition - into the training dataset and evaluating SymAE's performance with each dataset modification

4.1 Data

The new interface model dataset used in the training and testing of SymAE consists of synthetic data generated by the GeoPhyInv toolbox, an acoustic and elastic wave equation solver for high-performance computations. To ensure congruency between datasets, similar to the previous horizontal reflector dataset, the size and resolution of the interface model's physical domain, the number of datapoints generated, and the seismic acquisi-

tion parameters remain constant. The difference between the interface model dataset and the previous horizontal reflector dataset lies in its subsurface geology, whereby three flat reflectors are introduced into the subsurface structure unlike the single horizontal reflector present in the previous subsurface structure. Additionally, the three reflectors are not constrained to lie horizontally, but instead lie at an angle to the horizontal plane. Thus, the geometric attribute of *dip* is established in our reflectors. Besides that, the water column structure is unchanged from the previous dataset, and maintain the same depth-dependent acoustic velocity profiles.

A two-dimensional model of rectangular shape was used as our physical domain. It has 1301 gridpoints on the x-axis spanning a length of 6500m, and 1201 gridpoints on the z-axis spanning a depth of 6000m. The sea surface is positioned at a depth 0m, which is the top of the physical domain. The model is divided into an upper layer consisting of a water column and bottom layer of bedrock; the subsurface we are interested in imaging. The dataset is composed of seismic measurements from 1000 different subsurface models, which constitute the datapoints Xii=1,...,1000. Each datapoint has 11 different water velocity models, which constitute the instances kk=1,...,11.3 This results in a total of 11,000 models in the entires seismic dataset. Figure 22 shows an example of a model generated. The following sections describe the specifics of the subsurface bedrock and water column structure.

Subsurface structure

The subsurface extends from the seafloor, at a depth of 2000m, to the bottom of the physical domain, at 6000m. Three flat reflectors are placed at 3030m ± 30m, 4060m ± 130m and 5090m ± 30m. This depth placement of reflectors roughly divides the subsurface into four layers. Each flat reflector is then rotated about its midpoint on the x-axis (3250m), from the horizontal plane to create dip. For reflectors with midpoints below 3750m, the dip angle is between -5 to 0, reflectors with with midpoints between 3750m and 4750m have dip angles between 0 to $+8$, and reflectors with midpoints above 4750m have dip angles between -5 to 0. These angle ranges were chosen so that flat reflectors lie cleanly on the subsurface without intersecting each other.

A different and individual seismic velocity is assigned for each layer in the subsurface,

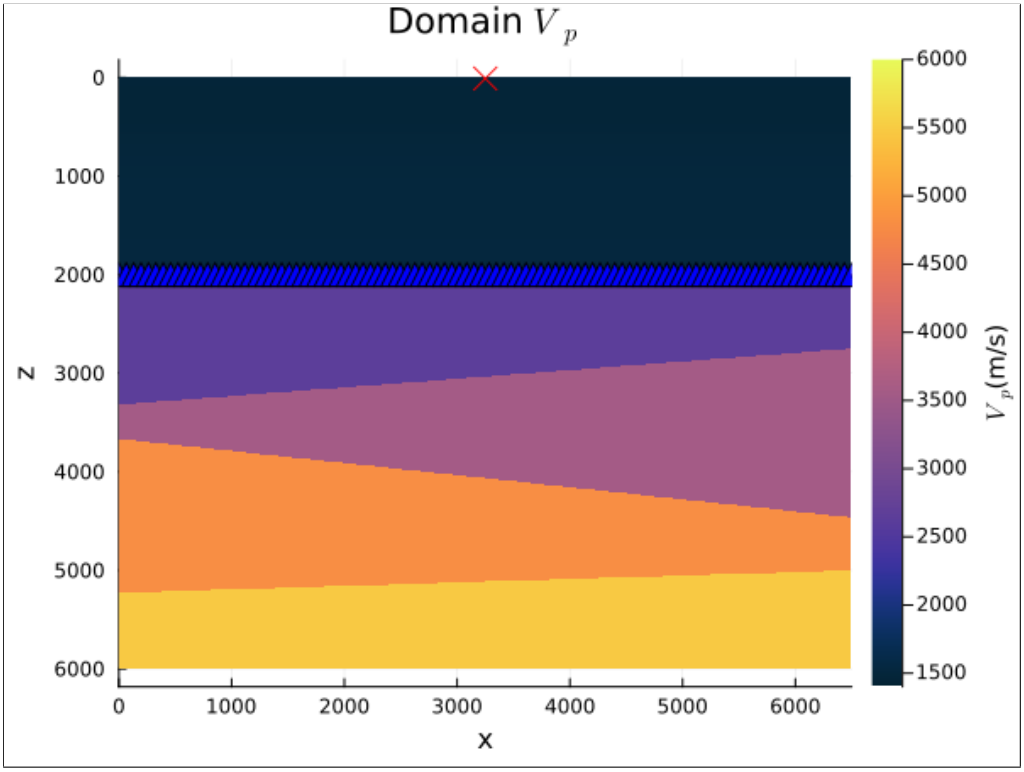

Figure 22: Visualization of a model used to create synthetic shot gather data. Vp here refers to the acoustic wave propagation velocity through the domain. The source, marked with a red x, is positioned in the water column at a depth of 10m. An array of equally-spaced receivers, marked with blue triangles, are positioned on the seafloor at a depth of 2000m. The seawater column appears to be vertically homogenous because the depth-dependent water velocity profile does not vary drastically enough for a visible change in the plot. However, as shown in Section 3.2.1, the modeled water velocity does indeed change with depth through the domain. For each datapoint, one instance within kk=1,...,11 is modeled as a reference instance, ref, that follows a reference velocity profile (described in Section 3.2.1). The remaining 10 instances are variations from the reference velocity profile.

which are bounded by the dipping reflectors. The velocities range between 1800m/s to 6000m/s for the entire subsurface. The selection of each layer's velocity consists of a three-step process: First, the velocity range bounds of 1800m/s to 6000m/s is linearly mapped to the subsurface depth range of 2000m to 6000m. Hence, the map $f(z) = y$, for z being depth and y being velocity is created. Second, a velocity value is assigned to each dipping reflector by applying the previously computed linear map onto the midpoint depth (depth at x-axis midpoint of 3250m) of the reflector to output its corresponding velocity value. Accordingly, z is the midpoint depth; f, the linear map; and y, the assigned reflector velocity value. Third, a uniformly-distributed random value between the velocity bounds of the layer (which are the assigned velocities of reflectors bounding

the layer, the 1800m/s value for an upper seafloor boundary, or 6000m/s for the physical domain bottom boundary) is selected for each layer. In a nutshell, the seismic velocity of each layer is different and increases with subsurface depth; and falls between the 1800m/s to 6000m/s range.

To summarize, across datapoints, the depth midpoint and dip angles of the three dipping reflectors are uniformly distributed between their specified ranges mentioned earlier in this section, and the velocities of the subsurface layers bounded by these reflectors are also uniformly distributed between the linearly mapped velocity bounds of each reflector at the reflector's midpoint depth.

Water column structure

The water column structure of our velocity models remains unchanged from the previous dataset, following a depth-dependent velocity function that consists of two states: reference and perturbed. The reference water velocity adheres to Hood's model, while the perturbed water velocity is derived from Hood's model with a cosine-squared modulation applied to the top 0m to 1000m of the water layer. Each datapoint generated includes one reference instance, $\tau_{k=1}$, with the Hood's water velocity model, and 10 perturbed instances, $\tau_{kk=2,...,11}$, with the aforementioned modulations to the Hood's water velocity model. This water velocity format is identical to the previous horizontal reflector dataset, as we aim to evaluate SymAE's performance for static correction in more complex subsurfaces. Therefore, there is no alteration to the water velocity profiles in this new dataset. For more details about deriving the water velocity profile, including the equation describing Hood's model, refer to Section 3.2.1. The velocity perturbation p for this dataset falls within the [-6,+6] range. This range was selected based on findings from the previous horizontal reflector experiments, which indicated that the [-6,+6] range is sufficiently challenging for testing SymAE's static correction capabilities due to the larger timeshifts caused by larger ray bending propagation paths. These experiments also showed that SymAE performs better with the [-2,+2] perturbation range, which is typically observed in the real ocean, than with the [-6,+6] range. Consequently, we can infer that SymAE would perform better in realistic deepwater environments where the [-2,+2] perturbation range is observed. Thus, the [-6,+6] dataset was chosen as it

represents a challenging edge case for testing SymAE's capabilities for static correction.

Shot gather simulations. From the physical models described, shot gather data was generated with the following specifications:

Finite difference simulations. The 2D acoustic wave equation solved with a second order in space and second order in time finite-difference modeling. Perfectly matched layers (PML) used at all model boundaries, except the z= 0 free surface, which has a Dirchlet boundary condition of 0. Simulations ran for a time range of 8.8456s, with 0.00062s time steps; with grid-spacing of 5m in x and z directions.

Acquisition geometry. Single source at a depth of 10m from the top of the domain (within the water column). 100 receivers, equally spaced, positioned at seafloor depth of 2000m.

Source signature. Ricker wavelet with peak frequency of 6.78Hz and maximum frequency of 20.23Hz.

Data Normalization

As the generated seismic records have a wide range of amplitudes - mostly pertaining to the direct arrival and primary reflections exhibiting larger amplitudes and multiples showing lower amplitudes - we normalized the seismic records. This was done to reduce the large contrast in magnitudes amongst various arrivals; enhancing the visibility of subtler amplitudes and diminishing the influence of extreme amplitude values. This would aid SymAE in noticing and learning a broader range of features from the seismic record [Huang et al., 2023]

In our study, we applied arctan normalization to the seismic records to calibrate their dynamic amplitude ranges; compressing high-amplitude values and expanding low-amplitude values for a more uniform amplitude distribution. The normalization function used was,

$$A(r,t)_{\text{normalized}} = \arctan\left(\frac{A(r,t)}{\alpha}\right) \tag{22}$$

where $A(r,t)$ is the amplitude of the seismic signal at time t for trace r, and α is a scaling parameter that influences the slope of the arctan function with respect to $A(r,t)$. A larger α results in a flatter curve at the origin and gentler transition towards the asymptotes at $\pm\frac{\pi}{2}$, whereas a smaller α results in a steeper curve at the origin and a faster transition towards the asymptotes at $\pm\frac{\pi}{2}$. Via experimentation, we decided on $\alpha = 0.1$ which amplified smaller amplitude values near the origin for increased visibility in the seismic records. After applying the arctan transformation, the resulting amplitudes were in a more normalized range, between $\frac{-\pi}{2}$ and $\frac{\pi}{2}$.

4.2 Hyperparameter Tuning

From our previous experiment with tuning SymAE's hyperparameters for the horizontal reflector dataset, we learned that the expressivity of SymAE's encoders strongly influenced its performance. Thus, the elements which control expressivity - latent dimensions of the coherent and nuisance encoder, and the Gaussian dropout applied to the output of the nuisance encoder - were the hyperparameters we focused on for tuning.

The interface model dataset, which includes additional reflectors of varying depths and dips in the subsurface geology, differs from the previous dataset with a single horizontal reflector. The increased complexity of the new subsurface geology necessitates higher expressivity, suggesting the need for increased latent dimensions in the coherent encoder; which captures the symmetric subsurface geological information for all instances in each datapoint.

We conducted three experiments, each focusing on one of the high-impact hyperparameters mentioned above: Gaussian dropout rate, coherent encoder latent dimensions, and nuisance encoder latent dimensions. In each experiment, we trained SymAE on the interface model dataset, whilst varying the values of the focused hyperparameter being optimized for. The training process involved minimizing the reconstruction MSE loss, using the Adam optimizer with a learning rate of 0.001.

Figures 23, 24 and 25 illustrate the outcomes of our experiments. First, keeping the coherent and nuisance latent dimensions fixed, we tested dropout rates from 0.2 to 0.8, and

observed that the dropout rate of 0.5 provided the lowest normalized residual norm between unseen reference and redatumed seismic records, and the highest gain in timeshift reduction before and after redatuming of the perturbed seismic records (Figure 23). Next, setting the dropout rate to 0.5 and keeping the nuisance latent dimension fixed to its value during dropout rate tuning, we tested coherent latent dimensions from 60 to 120. We observed that the coherent latent dimension of 70 provided the lowest normalized residual norm and highest gain on the unseen test dataset. Lastly, setting the coherent latent dimension to 70 and maintaining the dropout rate of 0.5, we tested nuisance latent dimensions from 20 to 80. We observed that the nuisance latent dimension of 60 provided the lowest normalized residual norm and highest gain on the unseen test dataset. Thus, the optimal network tailored for our new interface model dataset has the following configuration: **Gaussian dropout rate of 0.5, coherent encoder latent dimension of 70 and a nuisance encoder latent dimension of 60.** In the Results section, we will explore the predictions made by this optimized network in greater detail.

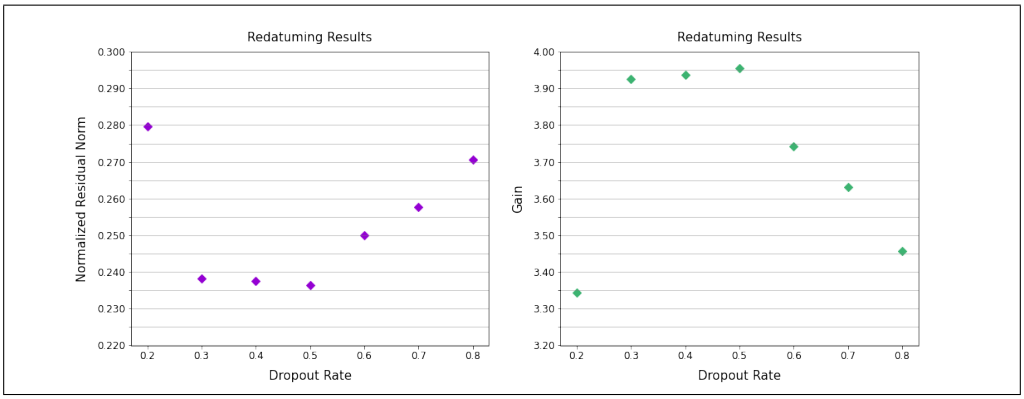

Figure 23: SymAE hyperparameter tuning results for dropout rates (applied to the output of the nuisance encoder) ranging from 0.2 to 0.8. The dropout rate of 0.5 yields the lowest normalized residual norm between unseen reference and redatumed seismic records, and the highest gain in timeshift reduction before and after redatuming of the perturbed seismic records.

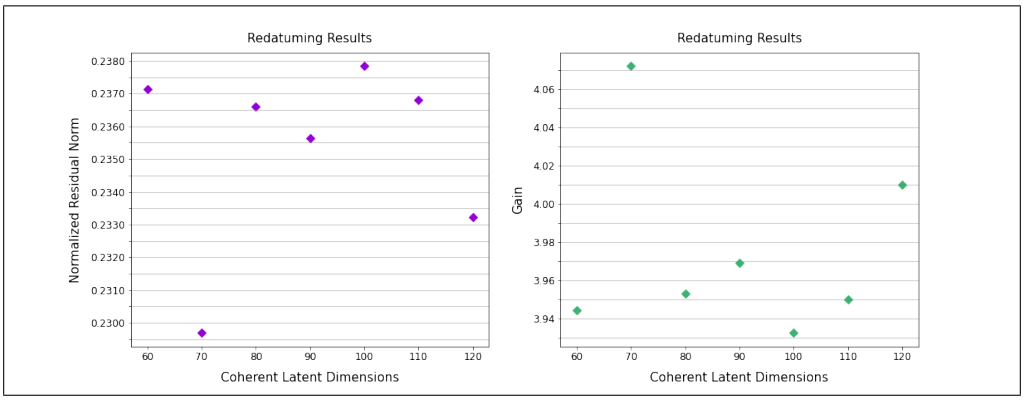

Figure 24: SymAE hyperparameter tuning results for latent dimensions of the coherent encoder ranging from 60 to 120. The coherent latent dimension of 70 yields the lowest normalized residual norm between unseen reference and redatumed seismic records, and highest gain in timeshift reduction before and after redatuming of the perturbed seismic records.

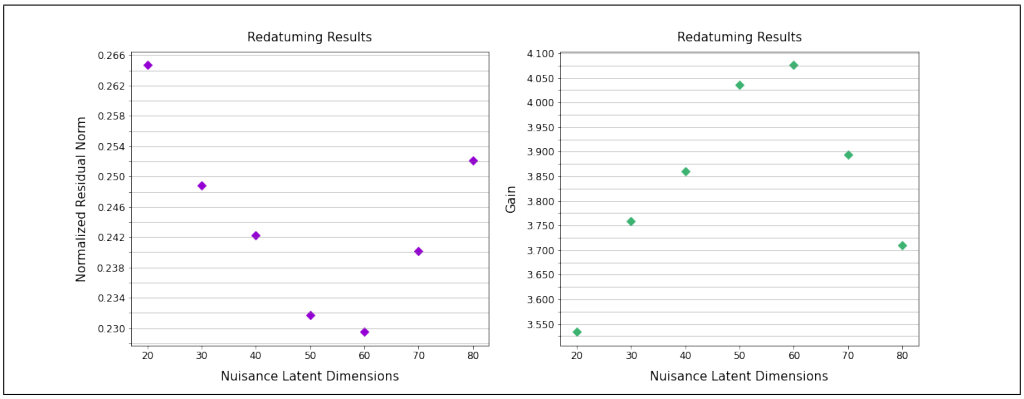

Figure 25: SymAE hyperparameter tuning results for latent dimensions of the nuisance encoder ranging from 20 to 80. The nuisance latent dimension of 70 yields the lowest normalized residual norm between unseen reference and redatumed seismic records, and highest gain in timeshift reduction before and after redatuming of the perturbed seismic records.

4.3 Results

With the optimal network configuration established in Section 4.2, we trained SymAE on the interface model dataset with [-6, +6] perturbation range. Upon completion of training, SymAE's ability to redatum unseen seismic records from the test dataset to produce statics-corrected records was evaluated. Figure 26 presets the pre- and post-corrected records for three different velocity models, each characterized by distinct subsurface and water velocity profiles. The amplitude differences expressed in the colorbar of the last-two columns, are percentage differences calcuated as $\frac{A(r,t)-B(r,t)}{A(r,t)_{\max}} \times 100\%$, where $A(r,t)$ is the amplitude of the reference trace at time t and receiver r, $B(r,t)$ is the amplitude of either the redatumed or reference trace at time t and receiver r, and $A(r,t)_{\max}$ is the maximum amplitude value of the reference record for all times and traces. Before redatuming, all three instance - reference residuals exhibit notably strong amplitude differences, mostly residing between the $\pm 25\%$ to $\pm 100\%$ range. After redatuming, all three instance - reference residuals exhibit significantly weaker amplitude differences, primarily within the 0% to $\pm 20\%$ range. These results clearly demonstrate that SymAE's redatuming process leads to the alignment of perturbed traces to reference traces, thereby minimizing the timeshifts between arrivals in the perturbed and reference traces.

In Figure 26, as we descend down the rows, the magnitude of the model's water perturbation increases from 2.000%, 4.215%, to 6.000%. Based on the results of the horizontal reflector case that demonstrated the increase in the test dataset's average normalized residual norms with increasing water perturbation ranges (see Figure 21), we would expect the gradual degradation in statics correction with increasing water perturbation. However, it is observed that there are stronger residual amplitude differences in the redatumed - reference records for the first model (top row of Figure 26) than in the second model (middle row of Figure 26), despite the second model experiencing a larger water perturbation. This contrast in amplitudes is especially visible in the later arrivals of the far offset receivers, where certain redatumed-reference amplitudes of the first model show differences larger than $\pm 25\%$. This finding suggests that there could be other factors at play besides water perturbations that affect the performance of SymAE; which may even supersede the influence of water perturbation on the accuracy of the redatumed records. What is clear in Figure 26 is that the reflectors in the first model's subsurface exhibits

greater dipping angles, than those of the second model's subsurface. This indicates that increased subsurface complexity is an additional factor influencing SymAE's performance on the shot record. We will explore this topic further in the discussion section of this chapter.

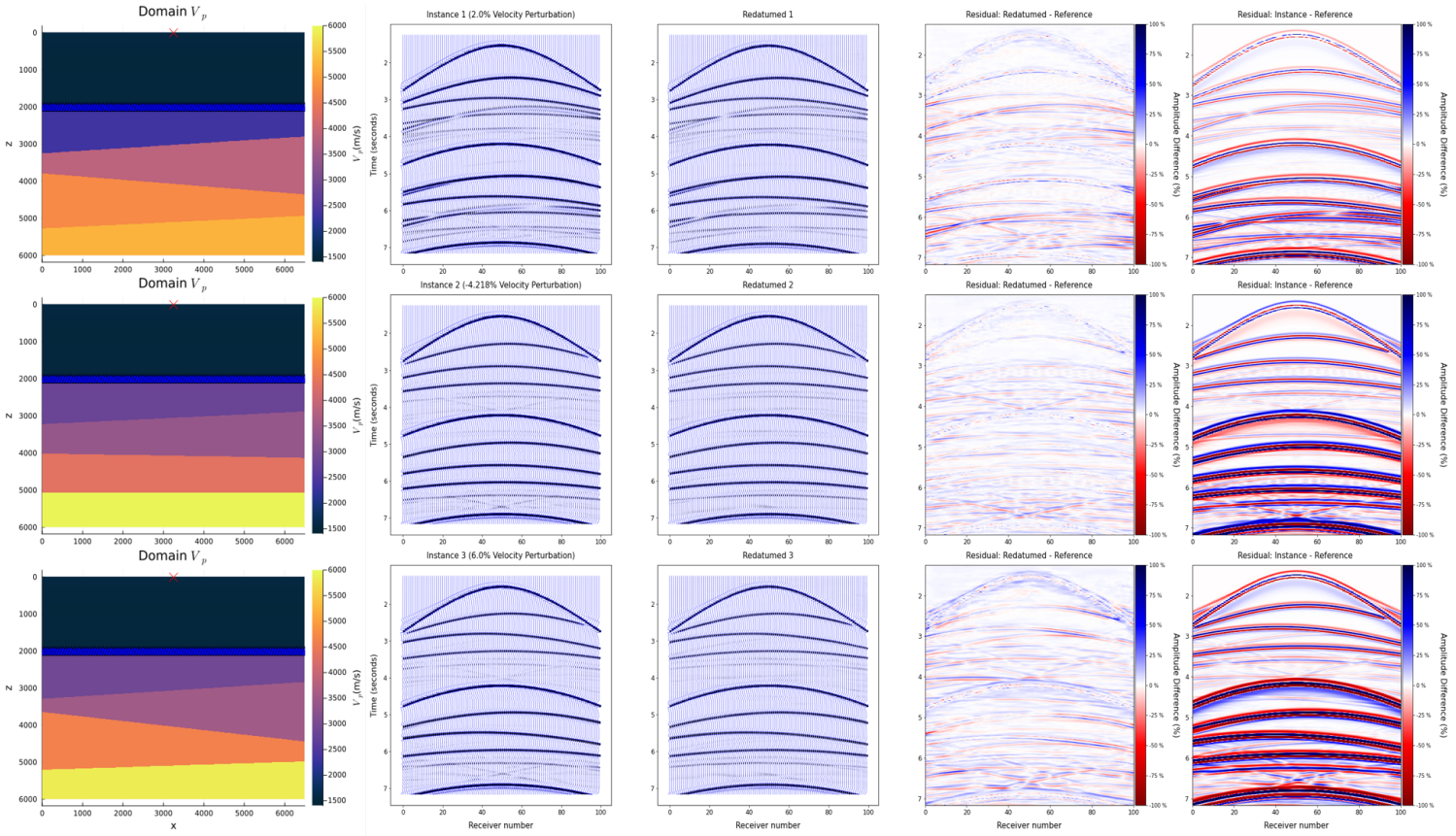

Figure 26: Dynamic static correction results for four velocity models, taken from the unseen test dataset. Each row represents one velocity model, with its characteristic subsurface structure depicted in the left-most column. The second column displays the perturbed shot record annotated with its specific water perturbation, and the third column displays the reference shot record arising from the Hood's water velocity profile. The final two columns show the residual of redatumed - reference record, and instance (perturbed) - reference record. The contrast between pre- and post-redatumed residuals clearly demonstrates that SymAE has aligned perturbed traces with reference traces, significantly diminishing the timeshifts between arrivals in the perturbed and reference traces.

To analyze the correction performed by SymAE on the traces more thoroughly, we computed the timeshifts between reference and perturbed traces, and between reference and redatumed traces using the Moving Window Cross-Correlation method (see Section 3.3) for details of the method).[21] Two unseen velocity models in our test dataset were randomly selected for this analysis: Model 919 and 926, as shown in Figure 27. Each model

[21] By nature of the Moving Window Cross-Correlation method, the fine-grained accuracy or precision of

features a distinct subsurface velocity profile, and encompasses instances with different velocity perturbations. Specifically, we focused on the +4.453% perturbation for Model 919 and the −5.813% perturbation for Model 926.

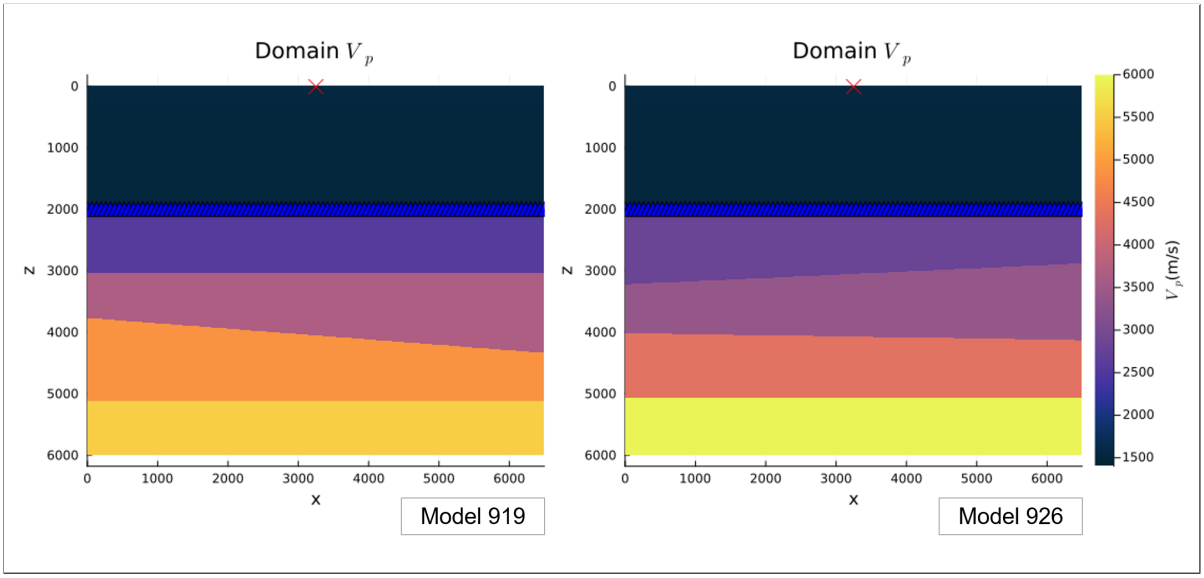

Figure 27: Two randomly selected models from the interface model test dataset, Model 919 and Model 926, featuring a distinct subsurface velocity profile. Each subsurface model has ten replicated instances with different velocity perturbations. These models were used for the subsequent computation of timeshifts between reference and perturbed traces, and between reference and redatumed traces; done with the purpose of further evaluation of SymAE's statics correction performance. The results of these computations are presented in Figures 28 and 29 respectively.

For Model 919, Figure 28 illustrates the timeshifts before and after redatuming for both the 0km (zero) offset and -3.25km (far) offset traces; and for Model 926, Figure 29 shows the timeshifts before and after redatuming for both the 0km (zero) offset and +3.25km (far) offset traces. For both models, the timeshifts between reference and perturbed traces are different for both zero offset and far offset cases, and vary with the depth traversed by the seismic ray, which is indicated by its arrival time on the trace. This underscores the dynamic nature of static timeshifts, which depend on both offset and depth. Δt_1 represents the timeshifts of the direct arrival from the seismic source at the ocean surface to the seabed receiver; multiples of this ray propagation path are labeled as seabed triplet and seabed quintuplet in the figures. Additionally, the primary and multiple reflections from the three different subsurface reflectors are labeled as Reflector 1 primary, Reflector 2 primary, Reflector 3 primary, Reflector 1 multiple, Reflector 2

the timeshifts is limited by the step size dt of the seismic traces, which is 0.01122 seconds. Consequently, the smallest possible timeshift is 0.01122 seconds, and each timeshift is a multiple of this step size.

multiple, and Reflector 3 multiple, respectively. In the far offset case for Model 919, we see in Figure 28 that there is no Reflector 1 primary labeled. This is because the Reflector 1 primary superposes with the direct arrival wave; both waves reach the far offset receiver at the same arrival time, and the resultant amplitude is the sum of displacements of both individual waves.

After redatuming, all timeshifts in Model 919 are reduced to zero except for the Reflector 3 multiple in the far offset case which has a timeshift of 0.01122s (the minimum possible timeshift based on the granularity of dt, as explained in the previous Footnote). In contrast, in Model 926, all timeshifts are reduced to zero after redatuming. Collectively, these findings demonstrate that redatuming and subsequent reconstruction of the seismic record with SymAE's decoder, aligns perturbed traces to the reference traces effectively, across different subsurface models. This outcome is consistent with results from Chapter 3, which show that the timeshifts for all offsets and depths strongly converge to zero, achieving the 0.01s level of precision for each wave arrival. These results further confirm the superior accuracy of the data-centric SymAE compared to the model-centric benchmark test on marine statics correction. In the benchmark test, timeshifts showed weak convergence to zero, with most arrival timeshifts exceeding 0.01s and reaching up to 0.04s. Besides this, the velocity profile of the upper water layer and underlying subsurface medium not only affects the arrival times of direct arrivals and various reflections but also influences how these waves interact and combine with each other, resulting in complex waveforms when the waves do not meet in perfect alignment. This variability in waveforms and arrival times is evident in the traces of Models 919 and 926. This shows that even in the presence of arrivals that may have undergone superposition, SymAE is able to align such perturbed traces to their respective reference, thereby strongly converging all timeshifts to zero.

The error between redatumed and reference seismic records is quantified by taking the normalized L^2 norm of residuals between both records. The L^2 norm of a seismic record is a measure that represents that overall magnitude of the seismic signal, by taking the

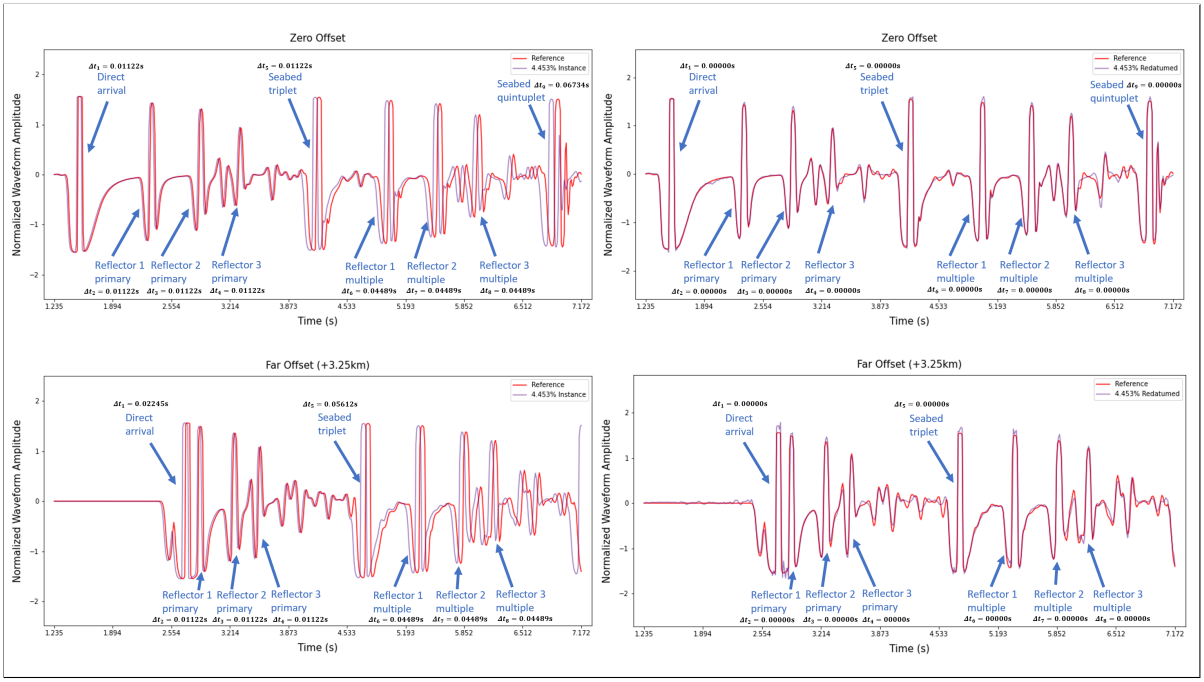

Figure 28: Results of redatuming the +4.453% perturbed instance of test datapoint, Model 919. The timeshifts compared to reference trace, before and after redatuming, for zero offset and far offset are shown. All timeshifts of the large-amplitude arrivals are reduced to zero.

square root of the sum of squares of amplitude values across the seismic record,

$$||A||_2 = \sqrt{\sum_{r=1}^{m}\sum_{t=1}^{n}|A(r,t)|^2} \tag{23}$$

where $A(r,t)$ is the amplitude of the seismic signal at time t for trace r, n is the number of time-points and m the number of traces. The L^2 norm of the residuals is thus,

$$||A-B||_2 = \sqrt{\sum_{r=1}^{m}\sum_{t=1}^{n}|A(r,t)-B(r,t)|^2} \tag{24}$$

where $A(r,t)$ is the amplitude of the seismic signal of record A; and $B(r,t)$, the amplitude of the seismic signal of a different record B. From this equation, we can see that $||A-B||_2$ computes the residual between amplitudes at each timepoint, for each trace; and then sums residuals across all timepoints for each trace, across all traces of the seismic record. When the two records A and B are the reference and perturbed records, $||A-B||_2$ is measure of the accuracy of the prediction, as it encodes the differences between the redatumed and reference record elements (amplitude values at each time and trace/offset

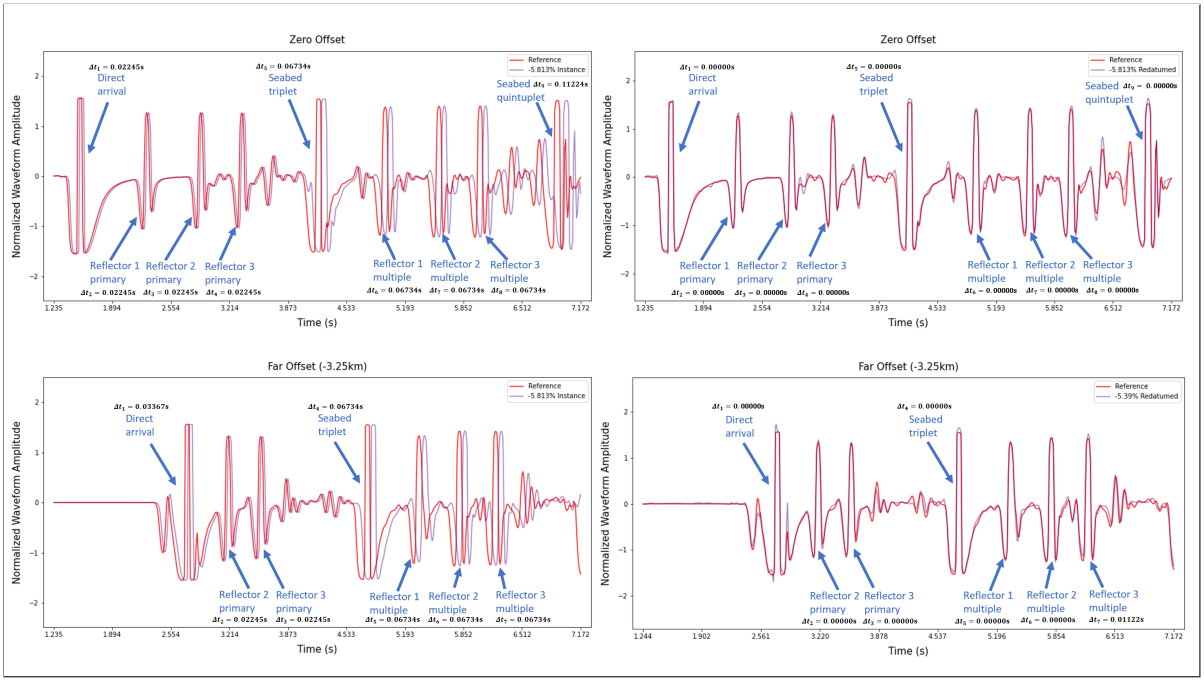

Figure 29: Results of redatuming the -5.813% perturbed instance of another test datapoint, Model 926, with different subsurface velocities than Model 919. The timeshifts compared to reference trace, before and after redatuming, for zero offset and far offset are shown. All timeshifts of the large-amplitude arrivals are reduced to zero except for the Reflector 3 multiple in the far offset case which has a timeshift of 0.01122s

.

point). We then take the normalized L^2 norm of the residuals, which is simply

$$\frac{||A-B||_2}{||A||_2} = \frac{\sqrt{\sum_{r=1}^{m}\sum_{t=1}^{n}|A(r,t)-B(r,t)|^2}}{\sqrt{\sum_{r=1}^{m}\sum_{t=1}^{n}|A(r,t)|^2}} \tag{25}$$

With this definition of the normalized L_2 norm (Equation 25), A would represent the reference seismic record, and B, the perturbed seismic record. This definition enables us to understand how the amplitude residuals between A and B scale in relation to the actual amplitude of waves observed in the reference record A - giving us insight into the significance of the error of the redatumed records.

To evaluate the effectiveness of SymAE in correcting for statics, we computed the normalized L^2 norm of residuals between redatumed and reference instances for all 100 test datapoints. These values were then averaged over all instances (1000 instances = 100 test datapoints $\times$ 10 instances per datapoint), yielding a mean normalized L_2 residual norm of 0.22494. To gauge the L^2 norm of redatumed-reference residuals relative to the original residual between redatumed-perturbed instances, we computed their ratio

of residual L_2 norms, $\frac{||\text{reference-perturbed}||_2}{||\text{reference-redatumed}||_2}$, which is equivalent to the gain achieved via SymAE's static correction method. Averaging the gains over all 1000 instances, the resulting mean gain value totaled 4.66693. The notably low normalized residual norm of 0.229 implies that SymAE's successful correction of timeshifts caused by velocity variations, effectively aligning redatumed data with reference data. Furthermore, the observed 4.7× gain underscores the substantial enhancement in seismic data quality before and after application of SymAE's statics correction methodology.

4.4 Discussion

The results reveal that SymAE performs well on the interface model dataset, which features several horizontal and flat reflectors at different depths and dips in the underlying subsurface. To understand SymAE's performance on this dataset relative to the previous horizontal reflector dataset (which is now our benchmark), we present in Figures 30 and 31 , a summary of the aggregated results, depicting the test dataset's average normalized residual norm and gain.

Normalized Residual Norm across Different Ranges of Data			
Test Data	[-2, 2] SymAE	[-6, 6] SymAE	[-25, 50] SymAE
[-2, 2] Horizontal Reflector	0.084	0.104	0.201
[-6, 6] Horizontal Reflector	0.190	0.098	0.202
[-25, 50] Horizontal Reflector	0.721	0.322	0.200
[-6, 6] Interface Model		0.225	

Figure 30: Average normalized residual norm achieved upon redatuming 100 unseen datapoints from the *horizontal reflector* and *interface model* datasets. Lower values indicate better generalization. The value highlighted in green is the result of this experiment, and the values highlighted in blue are best results from previous experiments, serving as a benchmark.

Figure 30 shows that the [-6,+6] interface model has a normalized residual norm that is slightly higher than the [-25,+50] horizontal reflector dataset; this norm is therefore in the ballpark of results demonstrated by SymAE in previous experiments. Additionally, it is not surprising that the interface model norm is slightly higher than that of the [-25,+50] dataset: its subsurface model encompasses additional complexities, which introduce more scattering events and multiples that are observed in the trace data.

Gain across Different Ranges of Data				
Test Data		[-2, 2] SymAE	[-6, 6] SymAE	[-25, 50] SymAE
[-2, 2]	Horizontal Reflector	2.819	2.267	1.172
[-6, 6]	Horizontal Reflector	3.453	6.704	3.243
[-25, 50]	Horizontal Reflector	1.911	4.276	6.894
[-6, 6]	Interface Model		4.667	

Figure 31: Average gain achieved upon redatuming 100 unseen datapoints from the *horizontal reflector* and *interface model* datasets. Higher values indicate better timeshift reduction before and after redatuming. The value highlighted in green is the result of this experiment, and the values highlighted in blue are best results from previous experiments, serving as a benchmark.

Figure 31 shows that the [-6,+6] interface model has a gain that is between that of the [-2,+2] and [-6,+6] horizontal reflector datasets; again demonstrating that its performance resides in the ballpark of previous SymAE experiments. Its gain value is an expected outcome. Firstly, this is because the [-2,+2] dataset has original timeshifts between perturbed and reference seismic records that are smaller in magnitude than those of the [-6,+6] dataset, which exhibits greater velocity perturbations and thus larger traveltime differences. Secondly, reining back into our previous explanation, the interface model [-6,+6] dataset introduces additional subsurface complexity, and thus yields a lower gain than that of its counterpart in the horizontal reflector dataset.

Overall, the results demonstrate a gradual degradation in the performance of SymAE with increasing subsurface geology complexity. The introduction of additional reflectors of different depths and dips into the subsurface layer (consequentially resulting in a four-layered velocity subsurface profile) creates more scattering events at the reflectors, which generate new branches of the incident wave; the reflected and transmitted waves. These multiples, along with the primary reflections, may interact with each other - superposing constructively, destructively, or partially; depending on the phase-alignment of the waves. This causes multiples of large amplitudes and small amplitudes, and of a variety of shapes, to emerge in the seismic records.

Therefore, if we look closely at the seismic traces, as shown in Figures 28 and 29, we see that the multiples consists of large amplitude reflector multiples[22] which are the labeled

[22]The large amplitude multiples are the first-order multiples that arise when the seismic waves reflect off the interface, travels upwards, reflect off the seabed, travels downwards, reflecting off the interface again before returning upwards to the seabed. Hence, it involves one extra round trip between two

reflector multiples, and small amplitude multiples. In the aforementioned figures, it is evident that post-redatuming, the timeshifts strongly converge to zero; specifically, in the direct arrival, seabed triplet, seabed quintuplet, reflector primaries and large amplitude (first-order) reflector multiples (for the purpose of brevity, we classify these arrivals as large amplitude arrivals, and other arrivals as small amplitude arrivals). The error between reference and redatumed traces primarily arise from the amplitude differences of the two records. Zooming into the large amplitude arrivals, this error is predominantly caused by the redatumed amplitudes not reaching the full extent (magnitude) of the reference amplitudes, although the shape of the waveforms are preserved. In contrast, zooming into the small amplitude arrivals, the error predominantly arises from the difference in shape between redatumed and reference waveforms, whereby the redatumed waveforms are smoother than the reference waveforms: redatumed traces express a reduced intensity and frequency of crest and troughs compared to reference traces.

Veritably, the ability of SymAE to smooth out the crests and troughs of small amplitude arrivals, creating waveforms that are more continuous and less jagged, is due to its capacity for denoising. This is due to SymAE's design, which is based on the architecture of autoencoders, which is adept at denoising unwanted noise from signals [Berahmand et al., 2024]. This happens because the encoder is designed to abstract and extract informative features from the inputs into a lower-dimensional latent space, and the decoder is designed to reconstruct corresponding inputs based on the compressed latent representation. Since perfect training—where the autoencoder perfectly maps inputs to outputs—is rare, the encoder naturally filters out noise during data compression, and the decoder subsequently reconstructs denoised outputs. In addition to SymAE's denoising capacity based on its fundamental architectural properties, the robustness of SymAE to additive noise can be enhanced by using noisy data as inputs and clean (denoised) data as outputs of the network during training. This approach compels the network to capture and retain essential, noise-free features in its latent space - thus, impelling the conversion of noisy inputs into clean outputs.

These differences between redatumed and reference traces can be understood by examining a few fundamental properties of SymAE. Firstly, as the architectural design of

interfaces, the seabed and a subsurface reflector.

SymAE is based on the autoencoder, it is a *reconstruction algorithm* that seeks to construct its input data from meaningful features captured in the latent space. The presence of more subsurface interfaces results in increased waveform arrivals, which in turn necessitates additional waveform reconstruction for each trace in the seismic records. Thus, parallel to the activity of increased waveform reconstruction, is the increased cumulative error between reconstructed and reference waveforms. Secondly, central to the architecture of SymAE (and autoencoders) are the *convolutional layers* that form the building blocks of the encoder and decoder networks. As convolutions entail sliding a (learned) weighted kernel across the height and width of the 2D seismic records, where the weights of the kernel are multiplied element-wise with corresponding image patch values - and the products summed up to a single scalar value; there is a degree of information abstraction that happens via this process. This enables the extraction of essential features from the seismic record, but may also abstract away subdued trace amplitudes that have a smaller influence on the summation of patch-wise elemental products. Lastly, neural networks are *universal function approximators* [Hornik et al., 1989], not perfect maps between input and output data. Hence, it is expected that some error exists between predictions (redatumed) and ground truth (reference) data. In our case, the redatumed seismic record approximates the reference seismic records but does not perfectly match them; highlighting the inherent nature of deep learning, which is to approximate functions. Bearing these concepts in mind, SymAE's function can be perceived as reconstructing from an empty canvas, amplitudes of seismic traces from key values (the features) extracted in its low-dimensional latent spaces via convolutional operations.

When analyzing the architectural properties of SymAE, we recognize that these are expected constraints of the network - no new insights are garnered from these errors. While achieving absolute precision is seldom feasible, SymAE can be further fine-tuned to capture and retain more nuanced details, such as small amplitude arrivals. Enhancements may involve amplifying predicted values such that amplitude of the final traces extend to reach those of the reference traces. This could be achieved by switching the activation function in the final layer to a linear one that does not squash output values to a limited range. Features are extracted through convolutional layers, which distill information - but increasing the number of filters (parallel kernels in each convolutional layer) may broaden the range of feature extraction by generating more feature maps. SymAE's

limitation of approximating functions however, remains, as it is an intrinsic aspect - both an ability and disability (depending on how one perceives it) - of deep learning [Hornik et al., 1989].

Another potential enhancement to SymAE is combining the inclusion of deeper layers with U-Net inspired skip connections [Ronneberger et al., 2021], that directly transmit information from encoders to decoders, in shallower layers. Based on the hierarchical structure of feature extraction with layers, deeper layers will extract even more latent patterns in the seismic records, which are ultimately represented in the latent space of the autoencoder. To combat the increased abstraction of information captured in deeper layers, skip connections could be used in shallower layers to pass higher-resolution information from the encoders to decoders, to facilitate the fine-grained reconstruction of small amplitude arrivals. However, it is essential to note a potential drawback: this approach may inadvertently pass non-disentangled information from the encoders to the decoder; specifically, if skip connections are introduced before the summation across downsampled instances in the coherent encoder. Therefore, a careful analysis of the implications of introducing skip connections in this scenario is necessary before implementation.

In summary, our experiment with the flat interface dataset reveals that SymAE adeptly manages the complexities of a more intricate geological case, pertaining to multiple reflectors of different depths and dips in the subsurface. With a macro view, we observe that SymAE is successful in converging non-zero timeshifts between perturbed and reference traces to zero, thereby aligning the perturbed to reference traces. The error between redatumed and reference records were quantified by its normalized residual norm and gain, both of which fall within the range observed in previous experiments using the horizontal layer dataset (which constitutes a more simplified geological model). Using a micro view, we understand that the gradual degradation from its [-6,+6] counterpart in the previous dataset can be attributed to the emergence of additional multiples of various shapes and sizes due to increased wave scattering at the multiple reflector locations. This presents a new challenge to the present architecture of SymAE, and highlights an area of further study: modification of SymAE's architectural properties to enhance its predictive ability.

4.5 Conclusion

We extended the research in Chapter 4, to encompass more complex subsurface geology. The new geological profiles feature additional reflectors - three to be precise - of various depths and dips. These reflectors created more arrivals in the seismic traces due to increased scattering in the subsurface medium. This challenged SymAE's prowess in statics correction, especially in the detailed reconstruction of small amplitude multiples. However, SymAE's ability to correct for timeshifts, experienced by large amplitude arrivals such as the direct arrival, primary reflections and first-order multiples, remain uncompromised. In both experiments - the simplified horizontal reflector dataset and the more intricate interface model dataset - SymAE succeeds in aligning perturbed traces to reference traces, strongly converging all offset and depth dependent timeshifts to zero - and satisfying the time precision requirement of 0.01s for all arrivals.

The clear and evident path forward is further innovation on the architectural properties of SymAE to improve its capacity for waveform reconstruction, which would result in the superior accuracy of trace amplitudes of redatumed records. A more ambitious path forward which is a step ahead from our current experiments, but an important prerequisite for *applied* statics correction using SymAE in real geological environments, is explained in the section below.

4.6 Future Directions

Towards Uncharted Territory

The experiments conducted in this chapter and the previous chapter, investigated the ability of SymAE to perform dynamic timeshifts on geological subsurfaces of increasing complexity. To focus exclusively on the matter of enhanced heterogeneity in the geological profile, we ensured that the timestamps of primary reflections and first-order multiples were relatively consistent across all instances for the 1000 datapoints in our datasets. This was done by fixing the number of reflectors to be consistent (either one or three, depending on the dataset) across datapoints, and curbing the range of dipping angles and depths of reflectors of the interface model dataset to exclude extreme values;

these limits are detailed in the Data section of this chapter. By doing so, the timestamps of the large-amplitude reflections, were stabilized to exist within smaller margins for all datapoints. Each arrival at each of these timestamp margins thus likely, represents the primary reflection or a same-order multiple from corresponding reflectors, associated to each timestamp margin.[23]

Thus, by understanding the coherency of timestamp margins across datapoints (hereafter referred to as pan-dataset arrival timestamps), the neural network can focus on the minor timeshifts, induced by water velocity perturbations, within the waveforms for each pan-dataset arrival timestamp; these variations in timeshifts would register as nuisances to SymAE. This results in a pan-dataset understanding of coherent and nuisance information from a pure seismic data space: coherent subsurface information across instances, extends to coherent subsurface information across datapoints, by the gentle moderation of subsurface interface reflections to arrive at similar timestamp margins; nuisance information across instances, extends as the only nuisance information across datapoints, by ensuring that sufficient coherence is reflected in seismic records when variations in subsurface properties exist across datapoints.

We enter into unknown territory via the introduction of diversity into the subsurface profiles which disrupts the pan-dataset coherence of arrival timestamps. This could happen when the number of subsurface interfaces (reflectors) varies between datapoints, or when interfaces of different dipping angles intersect with each other - thereby breaking the flat interface pattern of our previous simulations. We may also introduce velocity heterogenities into the layers bounded by the reflectors, to represent different objects embedded in the subsurface such as salt bodies, which may have distinct velocities compared to the surrounding sedimentary rock surrounding it [Teixeira and Lupinacci, 2019]. These diverse conditions are typical in real geological environments and should be tested with SymAE to prepare for practical applications of the method. The key method of incorporating these diverse conditions is to use a training dataset that encapsulates the diversity of subsurface realities,[24] and which crucially, disrupts the pan-dataset coherence

[23]Realistically, there are deviations to this general approximation, particularly in far-offset arrivals. These arrivals experience increased raypath bending which extends their traveltimes, often leading to interaction and superposition with other scattered reflections.

[24]Note that this diversity exists between datapoints, not between instances of datapoints, which have constant subsurface profiles.

of arrival timestamps - thereby, causing the arrival timestamps themselves to appear as pan-dataset nuisance variations. With this adjustment to the training dataset, and important question rises to the surface, does SymAE breakdown in the presence of two-pan dataset nuisance variations, one arising from water velocity perturbations, and another arising from subsurface velocity variations?[25]Answering this question is beyond the scope of our study, but a key step toward using SymAE for marine statics correction in real deepwater environments.

Generalization and Robustness Considerations

First, we mention that the discussion in this section is informed by the work, *Machine Learning Robustness: A Primer*, by Braiek and Khomh [2024]. As alluded to in the previous discussion, Towards Uncharted Territory, the generalization ability of SymAE to perform dynamic statics correction on a diversity of subsurface profiles - that disrupt the pan-dataset coherence of arrival timestamps - needs to be further researched. As deep learning models are based on the principle of Empirical Risk Minimization, which assumes that training and test data are identically and independently distributed, the i.i.d. assumption (also known as a closed-world assumption); the essential prerequisite for evaluating i.i.d. generalization is to ensure that the test and training data come from the same closed distribution. Therefore, to explore SymAE's i.i.d. generalization ability on a diversity of subsurfaces, the training dataset has to also encompass a similar diversity of subsurfaces. The threshold at this point is that SymAE will likely breakdown in the presence of high subsurface diversity during training, because the multitude of subsurface variation disrupts the pan-dataset coherence of arrival timestamps - creating two pan-dataset nuisance variations from subsurface and water velocity changes, instead of just one pan-dataset nuisance variation arising from water velocity changes. If demonstrated to be true through experiments in the presence of two pan-dataset variations, subsurface diversity at the level which disrupts the pan-dataset coherence of arrival timestamps becomes the most important limitation of SymAE.

Therefore, based on our current understanding of SymAE's capabilities and the aforementioned likely limitation, we deduce that training SymAE on *any* possible marine

[25]Keep in mind, that the water velocity variations exist between instances and datapoints, and subsurface velocity variations exist only between datapoints and *not* between instances of the same datapoint.

seismic data will be unsuccessful because the infinite (or realistically, extremely large) subsurface diversity this entails will break the pan-dataset coherence of arrival timestamps required for SymAE to learn to disentangle subsurface coherence and water velocity variation sucessfully. However, it is important to note that SymAE can theoretically *perform on any subsurface profile* so long as the condition that the training dataset is created to encompass variations to that profile that maintain the coherency of timestamp margins, is met. Consequently, for practical applications, it is advised to train a fresh, randomly initialized SymAE for each location where the statics need to be corrected for. The path for testing for sensitivities of SymAE towards data, is to: first, ensure i.i.d. generalizability; and secondly, evaluate robustness.

To address the first stage of ensuring i.i.d. generalizability, we first form an in-distribution (ID) dataset, from which training and test datapoints are selected from. To form the ID dataset, first, a rough estimate of a subsurface velocity model that captures key large-scale features of the location of interest is required; second, a range of variations to the rough subsurface model should be synthesized or collected from field data, to create the total training dataset (of number of datapoints in the order of at least 1000) with the constraint that these variations do not disrupt the pan-dataset coherency of the large-amplitude arrival timestamps. Once this ID dataset has been created, SymAE should be trained and tested on this dataset, and the stability of its predictive performance ensured. At this stage, potential vulnerabilities include underfitting, which occurs when there are too few or biased data points that fail to represent the full diversity of the problem space. Another issue is short-cut learning, where the network depends on simple yet misleading patterns in the training data instead of instead of capturing the underlying, more generalizable relationships. [26]

To address the second goal of evaluating robustness, we first quote from [Braiek and Khomh, 2024] the general definition of robustness: *Deep learning model robustness denotes the capacity of a model to sustain stable predictive performance in the face of*

[26]In our interface model dataset, the energy of the direct arrival, seafloor multiples and primary reflections in the seismic records was significantly stronger than the other multiples. This makes SymAE susceptible to short-cut learning from these selected arrivals, thereby missing the information conveyed by lower energy multiples. Thus, to mitigate the potential for short-cut learning with the interface model dataset, we applied the arctan normalization to the seismic records to moderate its dynamic amplitude range, allowing for a more consistent learning across all arrivals. Refer to Section 4.1 for further details on our normalization scheme.

variations and changes in the input data. Specifically, a deep learning model is deemed robust, when deployed in a production environment, if its predictive performance remains within the acceptable tolerance level despite variations in input data as defined by the domain of potential changes. Hence, we have to specify the data changes against which the model would be tested, and the tolerance level (or the threshold for proper prediction) to evaluate robustness. In real-world scenarios, offshore reservoir/locational data distributions shifts may happen naturally due to natural environmental distortions (i.e., tectonic activity, seafloor subsidence/uplift and sediment deposition/erosion) or human activities such as production/injection operations in reservoirs. These subsurface perturbations form the domain of potential changes to the input data, potentially shifting the original ID data distribution to a different data distribution. The tolerance level for robustness depends on the application context and assurances needed. For the purpose of 4D seismic reservoir monitoring for imaging subtle fluid-flow features (such as reservoir pore pressure and fluid saturation changes), the typical spatial resolution in x, y and z dimensions required is in the order of 50 feet (approximately 15m). Substituting this spatial resolution $|\Delta z| = 15$ and the parameters of our velocity model into Equations 2, this corresponds to a timeshift resolution of 0.01s, which is equivalent to the level of precision achieved by SymAE's corrected-for timeshifts.

To evaluate the stability of SymAE's performance to these input data changes in dynamic environmental settings, we may compute and compare average normalized residual norm for reference vs redatumed seismic records for both in-distribution data and shifted-distribution data. This approach is analogous to the cross-testing experiments we ran in Chapter 3 on out-of-distribution data (see Figure 21), except here we would be cross-testing different perturbation ranges of the subsurface velocity layer instead of the water velocity layer. Additionally, other robustness metrics developed by the scientific community, such as those proposed by [Laugros et al., 2019] and [Taori et al., 2020], could be implemented in our experiments too.

In summary, SymAE can theoretically be trained to perform on a dataset with any subsurface velocity profile, provided that the velocity model perturbations of the datapoints are not so severe as to disrupt the pan-dataset coherence of arrival timestamps. This forms the closed-world domain of inputs for each training scenario at a specific loca-

tion, where SymAE is expected to maintain stable predictive performance, assuming i.i.d. generalization is achieved. Since a rough estimate of the velocity model at the location of interest is needed to create an ID dataset with perturbations that adhere to the pan-dataset arrival timestamp coherence constraint, this raises the question of whether SymAE is moving closer to a model-centric approach vs data-centric approach than expected? As described in Section 1.3, the model-centric approach consists of a two-stage process that transforms between non-commensurate model space and data space domains: the first stage, inversion of an accurate velocity model from the seismic data; the second stage, forward modeling from velocity model to seismic data, whereby a *reference* velocity model replaces the *inverted/measured* velocity model. In contrast, SymAE is a streamlined, one-step static correction method that operates directly between commensurate spaces (from data space to data space). Once trained, SymAE performs static corrections for multiple seismic records simultaneously (the instances of each datapoint) without needing to go through the intermediate stage of inverting for the velocity model of each seismic record. This direct mapping capability is a key feature that places SymAE as a data-centric approach, setting it apart from the model-centric approach.

For 4D applications in real geological environments, before deploying SymAE at an identified location of interest, two main preliminary checks are essential: first, acquiring the rough velocity structure of the area; and second, ensuring that the extremity of expected temporal perturbations to its velocity structure does not disrupt SymAE's pan-dataset arrival timestamp coherence constraint. Once these checks are completed, SymAE can be trained and used to correct statics for a live-acquisition of seismic records in a single predictive step, without needing additional details on the area's velocity perturbations.

5 Conclusions and Future Directions

This thesis is an investigation into using deep learning to solve a challenging problem in geophysics, the marine statics correction problem, which has predominantly been based in a model-centric paradigm. This paradigm involves a series of transformations between non-commensurate spaces: first, inversion from seismic data space to velocity model space and second, forward modeling from velocity model space to seismic data space. Statics correction within this paradigm has severe drawbacks; mainly the high compute, time and labor cost, and inaccuracies stemming from errors in velocity model inversion or from unmet assumptions about subsurface structure. Overcoming these drawbacks was thus, the prime motivation for our study - where we chose to leverage deep learning as the core algorithmic tool to understand the limits of the model-centric paradigm and explore the performance horizons of a different paradigm to statics correction, that which is data-centric.

In **Chapter 2**, we performed benchmark testing on the the model-centric paradigm for marine statics correction where deep learning was used to invert for the water velocity model from seismic records. For this inversion task, we designed a unique and novel neural network by hierarchically composing in sequence, four deep learning algorithms that have shown success in previous seismic inversion studies. The network performed well, producing predicted water velocity models with a MAPE value of 0.26%. This low degree of inaccuracy in the inverted models was however, assessed for downstream impact, by forward modeling to seismic data using an NMO-based procedure that did not require explicit subsurface velocity model. The results were moderately accurate seismic records with most arrival timeshifts, between ground truth and trace pairs, exceeding 0.01s and reaching values as large as 0.04s. This indicates that statics correction using the model-centric paradigm is unable to meet the time precision requirement of 0.01s for arrivals in the seismic records, necessary for achieving a spatial precision of 15m in the seismic image. Hence, there is a compelling need to improve on this approach for marine statics correction.

Subsequently, in **Chapter 3**, we chose to venture into a different paradigm: data-centric marine statics correction. The main feature of this approach is the direct mapping be-

tween commensurate data spaces, eliminating the need for intermediary transformations to and from velocity model space. To disentangle the cause of timeshifts, water velocity perturbations, from the information we want to preserve, the subsurface velocity profile; we implemented an autoencoder algorithm, named SymAE, that would leverage the permutation symmetry of subsurface information that is coherent at time scales at which, water velocity information is varying. Upon building a phantom generator, we created synthetic datasets with a simple subsurface model that creates reflected arrivals with a single flat and horizontal interface. Using these datasets, the trained SymAE successfully redatumed selected subsurface and water velocity information in its latent space to produce statics-corrected seismic records. In these records, the initial timeshifts between perturbed and reference records were strongly reduced to zero, with all wave arrivals achieving the 0.01s level of precision required for good seismic imaging and time-lapse analysis. The experiments in this chapter thus form the foundational understanding of SymAE's capacity in performing offset and depth dependent statics correction.

Building on this foundational understanding, in **Chapter 4**, we added a layer of complexity to the subsurface velocity models - with the presence of multiple flat interfaces of distinct depths and dips - to more accurately reflect the geological possibilities of real-world environments. Hyperparameter tuning was focused on coherent and nuisance encoder latent dimensions and dropout rate, to sufficiently express the increased information content of complex subsurface models. The introduction of multiple reflectors created more arrivals in seismic traces due to increased scattering in the subsurface medium. This challenged SymAE's prowess in statics correction, especially in the detailed reconstruction of small amplitude multiples. However, SymAE's ability to correct for timeshifts of large amplitude arrivals such as the direct arrival, primary reflections and first-order multiples, remain uncompromised - and is consistent with the results from the previous chapter. Overall, SymAE adeptly manages the complexities of this more intricate geological case; strongly converging all offset and depth dependent timeshifts to zero, and successfully achieving the 0.01s time precision criterion for all arrivals in the seismic traces.

In summary, our experiments demonstrate that a data-centric approach using SymAE for dynamic marine statics correction significantly outperforms the traditional model-

centric approach. SymAE delivers the required time shift resolution of 0.01 seconds for all arrivals, which is vital for precise imaging and time-lapse analysis - an achievement that the traditional approach falls short of. The ultimate goal of this study is the real-world deployment of SymAE. This objective will shape future research, as real deepwater environments present a myriad of additional complexities. This complexities should be layered one at a time into the training dataset, the stability of SymAE evaluated at each stage and necessary innovations made to its design to accommodate increasingly complex real-world scenarios.

5.1 Future Directions

The experiments described in this thesis have provided a foundational understanding of the capabilities and limitations of SymAE in performing dynamic statics correction on marine seismic data. Further work remains to be performed, with the end-goal in mind, of deploying SymAE in real marine environments. Here, we summarize several important research directions that will help propel SymAE and the data-centric approach it espouses, towards enhanced performance, generalizability and robustness.

Performance: Hyperparameter tuning strategy

The hyperparameters of SymAE are its parameters which are set before the learning process begins, and remains constant during training. Broadly, SymAE's hyperparameters can be separated into two domains: architectural properties and training configurations. In Chapter 4, we saw the need for innovations on the architectural properties of SymAE to improve its capacity for waveform reconstruction, which would lead to the increased accuracy of the trace amplitudes of redatumed records. In Chapter 3, when handling the edge case of the largest water velocity perturbation range, the need to improve the trainability of the network was suggested. In general, trainability is a function of the curvature and smoothness of its loss-landscape and the efficiency and trajectory of the optimization path, which are in turn, functions of hyperparameters: learning rate, optimizer type, batch size, regularization to the training loss function and the number of training epochs.

Due to being limited in computational memory and power, we manually performed hyperparameter tuning on our experiments using a labor-intensive, trial-and-error approach, where selected hyperparameters were chosen for each training run and the model evaluated for each combination, and subsequently updated based on performance comparisons of different training runs. To enable a broader exploration of hyperparameter values and to improve exploration efficiency in higher-dimensional spaces, more sophisticated hyperparameter tuning strategies should be explored in the future. Among such strategies which have been gaining more momentum to-date include HyperBand, Bayesian Optimization and PriorBand [Mallik et al., 2023]. Further optimizing SymAE's hyperparameters using advanced tuning strategies will improve its performance on various fronts: notably the predictive accuracy of redatumed records, convergence rate of training and generalization capacity on unseen data.

Generalizability: Disrupting the pan-dataset coherence of arrival timestamps

As alluded to in Section 4.6, the next phase of experiments for SymAE consists of more geologically diverse subsurface profiles that, all-together are trained on by SymAE. Creating this ID dataset with increased diversity will expand the generalization ability of SymAE to perform on a larger variety of unseen subsurface models. The frontier of our research at this point, is to create an ID dataset encompassing a multitude of subsurface profiles that break the pan-dataset coherency of arrival timestamps. SymAE has been shown to perform well when tested on diverse subsurface profiles which have been gently moderated such that subsurface interface reflections arrive at similar timestamp margins (keeping its pan-dataset coherency). However, we have yet to thoroughly test SymAE's performance on the case which disrupts the pan-dataset coherence of arrival timestamps.

We suggest gradually disrupting the pan-dataset coherence of arrival timestamps to determine if and where thresholds to SymAE's performance lie in the lens of changing input data; this can be done by taking an incremental approach of widening perturbations to the subsurface profiles in each ID dataset, and evaluating SymAE's performance at each of those datasets before further widening the subsurface perturbation range. A good first step, would be to generate similar synthetics as with our experiments in Chapter 3

and Chapter 4, but with a varying number of reflectors between datapoints - this would create a different number of arrivals between our seismic records, thereby breaking the coherency of the same number of arrivals, each arriving at their corresponding timestamps margins. Furthermore, the range of dipping angles and depths of the reflector can be increased to include more extreme values - thereby generating a myriad of interface patterns and intersections in the subsurface layer. Eventually, SymAE should be applied to benchmark synthetic geological models, such as Marmousi 2 [Martin, 2004], SEG/EAGE Salt Model and BP 2004 Model. Following our suggested incremental testing approach, upon successful performance on these benchmark models, SymAE should then be applied onto real marine seismic data acquired in regions of active offshore monitoring.

Robustness: Real-world deployment

Robustness is a key requirement in fostering trustworthy real-world artificial intelligence systems [Li et al., 2023]. From a technical viewpoint, it is an epistemic concept that presupposes the generalization ability of the model's inductive bias on in-distribution data. It further extends to assess the stability and resilience of this inductive bias in real-world deployment scenarios [Braiek and Khomh, 2024]. In real deepwater environments, a host of additional complexities are introduced to the seismic data: this may include lateral (x-direction) water velocity variation; the evolution (subsidence /uplift) of seafloor depth which moves the position of OBN receivers; and the addition of field noise, from environmental or equipmental sources, for instance. These variations in seismic data define the domain of input data changes, where it is crucial to evaluate the stability of SymAE statics correction performance. An important and useful metric for assessing this stability is the tolerance level of timeshift resolution (time precision for arrivals in the seismic record) needed to achieve the necessary spatial and velocity resolutions of the subsurface model for the intended applications (for computational details that relate timeshifts to velocity model sensitivities, refer to Section 1.4).

6 Appendix

6.1 Seismic Phantom Generator

We created a synthetic seismic data generator (known as a phantom generator in the geophysics community) to obtain large amounts of training data. Using this generator, 1000s of datapoints were created randomly for training our neural networks. We used the GeoPhyInv software, a seismic wave equation solver, to create synthetic earth velocity models and perform forward modeling to obtain our shot gathers for training. All code in the phantom generator was written in the Julia programming language. Below are important snippets (code blocks) of code used in the phantom generator. Figure 32 is the GeoPhyInv code used to generate velocity models and run forward modeling to produce shot gathers, and Figure 33 is the Julia code used to generate a training dataset of random seismic datapoints. Note the use of a Mersenne Twister to generate a sequence of random numbers (which importantly propagate into subsurface geology and water layer parameters), that can be replicated for each run of the same script.

```julia
function new_medium(medium, mgrid; num_subsurface_interface=3, interface_seed=11, gridspace=5., water_depth=2000., water_perturbation=0)
    medium_new=deepcopy(medium)
    vp = medium[:vp]
    rho = medium[:rho]
    nz = length(mgrid[1])
    nx = length(mgrid[2])
    water_vels =  Vector{Float32}(undef, Int(1100/gridspace+1)) #collect water velocities up to 1.2km depth

    water_depth_pt = argmin(abs.(zgrid .- water_depth)) #water depth can be anywhere between 1.2 to 2.4m

    subsurface_velocity_bounds = (1800, 6000)
    z_dom = (minimum(collect(mgrid[1]))/1000, maximum(collect(mgrid[1]))/1000)
    x_dom = (minimum(collect(mgrid[2]))/1000, maximum(collect(mgrid[2]))/1000)
    x_mdpt = (x_dom[2]-x_dom[1])/2
    slope_intersect_list = Vector{Vector{Float64}}()

    rand_interface = MersenneTwister(interface_seed)

    #Initialize variables
    water_intersect = nothing
    subsurface_interval= nothing

   function pt_to_coord(pt)
        #= takes takes gridpoint, and returns its coordinate value in km =#
        coord= (pt-1)*(gridspace/1000)
        return coord
    end

    function pt_to_coord_m(pt)
        #= takes takes gridpoint, and returns its coordinate value in m =#
        coord= (pt-1)*(gridspace)
        return coord
    end

    function coord_to_pt(coord)
        #takes coodinate value in m and returns it gridpoint
        point = coord/gridspace + 1
        return point
    end

    function Hood_function(zcoord)
        #= takes zcoord in m, applies the hoods function and returns the velocity at depth z in m/s =#
        velocity = 1541.30 - 0.18026 * zcoord + 2.12895*(10^-4) * zcoord^2 - 1.15430*(10^-7) * zcoord^3 + 3.28150*(10^-11) * zcoord^4 - 4.62212*(10^-15) * zcoord^5 +  2.52598*(10^-19) * zcoord^6
        return velocity
    end

    function subsurface_depth_to_velocity(depth)
        velocity_change_per_km = (subsurface_velocity_bounds[2]-subsurface_velocity_bounds[1])/(z_dom[2]-water_intersect)
        depth_velocity = (depth-water_intersect)*velocity_change_per_km + subsurface_velocity_bounds[1]
        return depth_velocity
    end

    for i = 1:num_subsurface_interface+1 #=1 to include for seafloor/water interface =#
        if i == 1
            #append water interface
            water_intersect = 2 #seafloor starts at a random value between 1 to 2.2km, and slopes downwards after.- this value is shifted to the x_midpoint of the region.
            subsurface_interval = 5.09-water_intersect #all future interfaces for the subsurface lie before the approximate target depth of 5.09km.
            seafloor_lowest_point = 2
            intersect_velocity = subsurface_depth_to_velocity(water_intersect)
            slope_intersect = [0, water_intersect, intersect_velocity]
        else
            layer_center_point = subsurface_interval/num_subsurface_interface*(i-1)+(water_intersect)
            subsurface_intersect =  rand(rand_interface, round((layer_center_point-0.4)*100):round((layer_center_point+0.2)*100), 1)[1]/100
            if subsurface_intersect ≤ 3.0
                random_angle = deg2rad(rand(rand_interface, -15:15, 1)[1])
                slope = rand(rand_interface, (-1,1), 1)[1]*tan(random_angle)
            elseif 3.0 < subsurface_intersect < 4.0
                random_angle = deg2rad(rand(rand_interface, -45:45, 1)[1])
                slope = rand(rand_interface, (-1,1), 1)[1]*tan(random_angle)
            elseif subsurface_intersect ≥ 4.0
                random_angle = deg2rad(rand(rand_interface, -18:18, 1)[1])
                slope = rand(rand_interface, (-1,1), 1)[1]*tan(random_angle)
            end
            intersect_velocity = subsurface_depth_to_velocity(subsurface_intersect)
            slope_intersect = [slope, subsurface_intersect, intersect_velocity]
        end
        push!(slope_intersect_list, slope_intersect)

        if i == num_subsurface_interface+1
            sort!(slope_intersect_list, by= x -> x[3])  #sort by intersect_velocity in ascending order
        end
    end

    #Hence, slope_intersect_list contains the different interfaces, including the seafloor.

    #Create a list of information associated to each layer
    layer_information = Vector{Vector{Vector{Float64}}}() #this includes the seafloor counted as one layer
    for i = 1:num_subsurface_interface+1
        if i == num_subsurface_interface + 1 #we are looking at the last layer
            layer_bounds_km = [slope_intersect_list[i][2],  (slope_intersect_list[i][2]+ z_dom[2])/2, z_dom[2]]
            layer_bounds_vel = [slope_intersect_list[i][3],  (slope_intersect_list[i][3]+subsurface_depth_to_velocity(z_dom[2]))/2, subsurface_depth_to_velocity(z_dom[2])]
            layer_velocity = [rand(rand_interface, (layer_bounds_vel[1]: layer_bounds_vel[3]),1)[1]]
        else #otherwise use upper and lower bounds from midpoints of above layers
            layer_bounds_km = [slope_intersect_list[i][2],  (slope_intersect_list[i][2]+ slope_intersect_list[i+1][2])/2, slope_intersect_list[i+1][2]]
            layer_bounds_vel = [slope_intersect_list[i][3],  (slope_intersect_list[i][3]+ slope_intersect_list[i+1][3])/2, slope_intersect_list[i+1][3]]
            layer_velocity = [rand(rand_interface, (slope_intersect_list[i][3]: slope_intersect_list[i+1][3]),1)[1]]
        end
        layer_forlist = [layer_bounds_km, layer_bounds_vel, layer_velocity]
        push!(layer_information, layer_forlist)
    end

    #Collect seafloor data
    seafloor = Dict("slope"=> slope_intersect_list[1][1],  "intersect"=> slope_intersect_list[1][2], "xmin"=> x_dom[1], "xmax"=> x_dom[2], "xmidpoint"=> x_mdpt, "nx"=> nx, "nz"=> nz)

    #Append water velocities
    for i in 1:water_depth_pt
        zcoord = pt_to_coord_m(i)
        water_velocity = Hood_function(zcoord)
        if water_perturbation ≠ 0
            if zcoord ≤ 1000
                perturb_C = (cos((zcoord/1000)*(pi/2))^2)*(water_perturbation/100*water_velocity)
                water_velocity = water_velocity + perturb_C
            end
        end
        vp[i, :] .= water_velocity
        rho[i,:] .= 1030 #kg/m^3
    end

    #Save upper bounds (or interfaces) of each layer
    layer_upper_bounds = Vector{Vector{Float64}}()
    for layer = 1:length(layer_information)
        upper_bound = zeros(nx)
        for xpt = 1:nx
            xcoord = pt_to_coord(xpt)
            upper_bound[xpt] = slope_intersect_list[layer][1]*(xcoord-x_mdpt)+slope_intersect_list[layer][2]
        end
        push!(layer_upper_bounds, upper_bound)
    end

    #Append subsurface layer velocities
    for layer = 1:length(layer_information)
        upper_layers = Vector{Vector{Float64}}()
        for j = 1:layer
            push!(upper_layers, layer_upper_bounds[j])
        end
        for xpt = 1:nx
            xcoord = xpt*gridspace
            #upper bound and lower bound refers to z values for each xcoord
```

```
#Generate Velocity Models
#------------------------------------------------------------------------------
#Set flags
generate_models = true
plot_models = false
view_model = false
multiple_source = true
#------------------------------------------------------------------------------
#Set parameters
subsample_factor = 19

num_subsurface_models = 1000 #full scope: 1000
num_water_perturb = 10    #full scope: 10
simulation_number = 124      #ori number: 124

variable_seafloor = false
expand_watervels = true

test1_waterpert = Float32.([-6, -5, -4, -3, -2, 2, 3, 4, 5, 6])
test2_waterpert = Float32.([-6, -5.5, -5, -4.5, -4, 4, 4.5, 5, 5.5, 6])
#test3_waterpert = Float32.([-1, -15, -20, 10, 15, 25, 35, 40, 45, 50])
#------------------------------------------------------------------------------
#Create random datapoints
simulation_generator = MersenneTwister(simulation_number)
#The one number that indirectly controls water perturbation amount, subsurface velocities, reflector depths in the subsurface.

water_number = rand(simulation_generator, 1:10, 1)[1]
water_generator = MersenneTwister(water_number)
models_number = rand(simulation_generator, 11:30, 1)[1]
models_generator = MersenneTwister(models_number)

#setting the seafloor (ocean/subsurface) depth level
if variable_seafloor
    all_seafloor_levels = Int64.(rand(simulation_generator, 1200:2400, num_subsurface_models)) #variable seafloor
else
    all_seafloor_levels = Float32.(fill(2000, num_subsurface_models)) #constant seafloor - pick 2000m following Kiyashchenko Brazil deepwater paper - previous constant seafloor had 1800m as seafloor level.
end

#generate a bunch of  reference models: subsurface layer + depth/dip/vels realizations
random_layers = Int64.(rand(models_generator, 3: 8, num_subsurface_models)) #value goes into number_subsurface interface, gives us 4 to 8 layers, does NOT include the END point
random_depthdips = Int64.(rand(models_generator, 1: num_subsurface_models*2, num_subsurface_models)) #value goes into interface_seed

#generate 10 waterperturbations, for each, of all reference models

if expand_watervels
    #expand water velocity perturbation from ~25% to 50%
    all_water_perturbs = Float32.(rand(water_generator, (-6000:6000)/1000, (num_subsurface_models, num_water_perturb)))
    #adding controlled water perturbations for samples 901-904 on test data
    all_water_perturbs[901,1:10] = test2_waterpert
    all_water_perturbs[902,1:10] = test2_waterpert
    all_water_perturbs[903,1:10] = test2_waterpert
    all_water_perturbs[904,1:10] = test2_waterpert
    all_water_perturbs[905,1:10] = test1_waterpert
    all_water_perturbs[906,1:10] = test1_waterpert
    all_water_perturbs[907,1:10] = test1_waterpert
    all_water_perturbs[908,1:10] = test1_waterpert
else
    all_water_perturbs = Float32.(rand(water_generator, (-2000:2000)/1000, (num_subsurface_models, num_water_perturb)))
end


if generate_models
    for i in 552:1000 #usually use variable 'num_subsurface_models' after :
        #generate 1 reference model and save data
        #Set parameters of model - seafloor, subsurface velocities, and reflector depth.
        water_depth_val = all_seafloor_levels[i]
        layer = random_layers[i]
        depthdip = random_depthdips[i]
        model_string = lpad(i,4,"0")
        #Create new medium and acquisition geometry
        n_medium = new_medium(medium, mgrid; num_subsurface_interface=layer, interface_seed=depthdip, gridspace=5., water_depth=2000., water_perturbation=0)
        ageom = get_ageom(;water_depth=water_depth_val)
        GeoPhyInv.update!(pa, ageom)
        GeoPhyInv.update!(pa, n_medium)
        #Perform forward model
        update!(pa)
        #Collect_velocities
        allvels = n_medium[:vp]
        #Collect Data
        if multiple_source == false
            data = pa[:data][:p]
            if any(isnan, data) == true
                print("nan found in original data")
                break
            end
            data_resampled=resample(data,inv(subsample_factor); dims=1)
            if any(isnan, data) == true
                print("nan found in resampled data")
                break
            end
            d=view(data_resampled,:)
            rescale!(d)
            if any(isnan, data) == true
                print("nan found in rescaled data")
                break
            end

            #Save Data
            fname = "/nfs/wavedata/kbrindha/Phantom Constant Seafloor 8.8s 6.6vel/seismic_" * model_string * ".00.hf"
            datafile = h5open(fname, "w")
            datafile["data"] = data_resampled
            datafile["subsurface velocities"] = allvels
            datafile["water_perturbation"] = Float32(0)
            close(datafile)

        elseif multiple_source == true
            for source in 1:ns
                s_data = pa.c.data[1][source]
                s_data = s_data[:p]
                if any(isnan, s_data) == true
                    print("nan found in original source data")
                    break
                end
                s_data_resampled=resample(s_data,inv(subsample_factor); dims=1)
                if any(isnan, s_data) == true
                    print("nan found in resampled source data")
                    break
                end
                s_d=view(s_data_resampled,:)
                rescale!(s_d)
                if any(isnan, s_data) == true
                    print("nan found in rescaled source data")
                    break
                end

                #Save Data
                source_string = string(source)
                fname = "/nfs/wavedata/kbrindha/Phantom Constant Seafloor 8.8s 6.6vel/seismic_" * model_string * ".00." * source_string * ".hf"
                datafile = h5open(fname, "w")
                datafile["data"] = s_data_resampled
                datafile["subsurface velocities"] = allvels
                datafile["water_perturbation"] = Float32(0)
                close(datafile)
            end
        end

```

References

A. Adler, M. Araya-Polo, and T. Poggio. Deep recurrent architectures for seismic tomography. *81st EAGE Conference and Exhibition 2019*, (June):3–6, 2019. doi: 10.3997/2214-4609.201901512.

A. Adler, M. Araya-Polo, and T. Poggio. Deep Learning for Seismic Inverse Problems: Toward the Acceleration of Geophysical Analysis Workflows. *IEEE Signal Processing Magazine*, 38(2):89–119, 2021. ISSN 15580792. doi: 10.1109/MSP.2020.3037429.

N. O. Administration and Atmospheric. Global temperatures, 2023. URL `https://www.ncei.noaa.gov/access/monitoring/monthly-report/global/202313#:~:text=Updated%20NOAAGlobalTemp%20dataset-,Global%20Temperatures,C%20(57.0%C2%B0F).` Accessed on June 18, 2024.

D. M. Advocate and K. C. Hood. An empirical time-depth model for calculating water depth, northwest Gulf of Mexico. *Geo-Marine Letters*, 13(4):207–211, 1993. ISSN 02760460. doi: 10.1007/BF01207749.

M. Araya-polo, J. Jennings, A. Adler, and T. Dahlke. Deep-learning tomography. *The Leading Edge*, (January 2018):58–66, 2018. doi: 10.1190/tle37010058.1.

D. Bank, N. Koenigstein, and R. Giryes. Autoencoders. *arXiv preprint arXiv:2003.05991v2*, pages 1–22, 2021. URL `https://arxiv.org/abs/2003.05991`.

Y. Bengio, A. Courville, and P. Vincent. Representation Learning : A Review and New Perspectives. *arXiv preprint arXiv:1206.5538v3*, pages 1–30, 2014.

K. Berahmand, F. Daneshfar, E. S. Salehi, Y. Li, and Y. Xu. *Autoencoders and their applications in machine learning: a survey*, volume 57. Springer Netherlands, 2024. ISBN 0123456789. doi: 10.1007/s10462-023-10662-6. URL `https://doi.org/10.1007/s10462-023-10662-6`.

K. J. Bergen, P. A. Johnson, M. V. de Hoop, and G. C. Beroza. Machine learning for data-driven discovery in solid earth geoscience. *Science*, 363(6433), 2019. doi: 10.1126/science.aau0323. URL `https://www.science.org/doi/abs/10.1126/science.aau0323`.

P. Bergmann, A. Kashubin, M. Ivandic, S. Lüth, and C. Juhlin. Time-lapse difference static correction using prestack crosscorrelations: 4D seismic image enhancement case from Ketzin. *Geophysics*, 79(6):B243–B252, 2014. ISSN 19422156. doi: 10.1190/geo2013-0422.1.

P. Bharadwaj, M. Li, and L. Demanet. SymAE: an autoencoder with embedded physical symmetries for passive time-lapse monitoring. In *SEG Technical Program Expanded Abstracts 2020*, Virtual, 2020. Society of Exploration Geophysicists. doi: 10.31223/osf.io/5zh8y. URL `https://library.seg.org/doi/10.1190/segam2020-3423931.1`.

P. Bharadwaj, M. Li, and L. Demanet. Redatuming physical systems using symmetric autoencoders. *arXiv: Computational Physics*, 2022. URL `http://arxiv.org/abs/2108.02537`.

H. B. Braiek and F. Khomh. Machine learning robustness: A primer, 2024. URL `https://arxiv.org/abs/2404.00897`.

D. C. Henley. Raypath interferometry: Statics in difficult places. *The Leading Edge*, (February):202–205, 2009.

C. Carpenter. Instantaneous 4D Seismic Used. *The Journal of Petroleum Technology*, (March):148–151, 2014.

R. J. Castle. A theory of normal moveout. *Geophysics*, 59(6):983–999, 1994. ISSN 00168033. doi: 10.1190/1.1443658.

A. Choromanska, M. Henaff, M. Mathieu, G. B. Arous, and Y. LeCun. The loss surfaces of multilayer networks. *Journal of Machine Learning Research*, 38:192–204, 2015. ISSN 15337928.

M. Cox. *Static Corrections for Seismic Reflection Surveys*. Society of Exploration Geophysicists, Tulsa, Oklahoma, 1999. ISBN 9781560801818. doi: 10.1190/1.9781560801818.

M. Cox. Static corrections for the 21st Century. *CSEG RECORDER*, 25(4):3–4, 2000.

J. Deere. Introduction to this special section—Statics. *The Leading Edge*, 28(2):190–191, 2009.

V. A. Del Grosso. New equation for the speed of sound in natural waters (with comparisons to other equations). *Journal of the Acoustical Society of America*, 56(4): 1084–1091, 1974. ISSN NA. doi: 10.1121/1.1903388.

L. Demanet. MIT Mathematics 18.325, Lecture Notes: Waves and Imaging, 2021. URL: `https://math.mit.edu/icg/resources/notes325.pdf`. Last visited on 31/7/2024.

E. Demirbağ, E. Gökaşan, F. Y. Oktay, M. Şimşek, and H. Yüce. The last sea level changes in the black sea: evidence from the seismic data. *Marine Geology*, 157(3):249–265, 1999. ISSN 0025-3227. doi: https://doi.org/10.1016/S0025-3227(98)00158-3. URL `https://www.sciencedirect.com/science/article/pii/S0025322798001583`.

J. S. Dramsch. Chapter one - 70 years of machine learning in geoscience in review. In B. Moseley and L. Krischer, editors, *Machine Learning in Geosciences*, volume 61 of *Advances in Geophysics*, pages 1–55. Elsevier, 2020. doi: https://doi.org/10.1016/bs.agph.2020.08.002. URL `https://www.sciencedirect.com/science/article/pii/S0065268720300054`.

E. F. Duarte, C. A. Da Costa, J. M. De Araújo, Y. Wang, and Y. Rao. Seismic shot-encoding schemes for waveform inversion. *Journal of Geophysics and Engineering*, 17 (5):906–913, 2020. ISSN 17422140. doi: 10.1093/jge/gxaa051.

N. El Allouche, G. G. Drijkoningen, W. Versteeg, and D. G. Simons. Studying converted waves in shallow marine environment. *Proceedings - European Conference on Noise Control*, pages 3649–3654, 2008. ISSN 22265147. doi: 10.1121/1.2934756.

N. S. Foundation. Marine seismic imaging: Illuminating earth's structure, climate, oceans and hazards, 2012. URL `https://www.nsf.gov/geo/oce/pubs/holbrook_brochure_langseth-jan2012.pdf`. Accessed on June 18, 2024.

M. Gelboim, A. Adler, Y. Sun, and M. Araya-Polo. Encoder–Decoder Architecture for 3D Seismic Inversion. *Sensors*, 23(1):1–12, 2023. ISSN 14248220. doi: 10.3390/s23010061.

I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. The MIT Press, 2016. ISBN 0262035618, 9780262035613.

F. X. Han, J. G. Sun, and K. Wang. The influence of sea water velocity variation on seismic traveltimes, ray paths, and amplitude. *Applied Geophysics*, 9(3):319–325, 2012. ISSN 16727975. doi: 10.1007/s11770-012-0344-2.

D. Hays, P. Docherty, K. Craft, and F. Smit. An Ocean Bottom Seismic Node Repeatability Study. SEG International Exposition and Annual Meeting, pages SEG–2008–0055, 2008.

D. Henley. Static corrections via raypath interferometry: recent field experience. In *Geoconvention 2014: FOCUS*, pages 1–4, 2014. URL `http://www.crewes.org/ForOurSponsors/ConferenceAbstracts/2014/CSEG/Henley_CSEG_2014.pdf`.

K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989. ISSN 08936080. doi: 10.1016/0893-6080(89)90020-8.

L. Huang, J. Qin, Y. Zhou, F. Zhu, L. Liu, and L. Shao. Normalization Techniques in Training DNNs: Methodology, Analysis and Application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(8):10173–10196, 2023. ISSN 19393539. doi: 10.1109/TPAMI.2023.3250241.

R. Huang, P. Wang, K. Nimsaila, and M. Vu. Angle-dependent water column statics correction through sparse TauP inversion. In *78th EAGE Conference and Exhibition 2016: Efficient Use of Technology - Unlocking Potential*, Vienna, 2016. ISBN 9789462821859. doi: 10.3997/2214-4609.201600581.

C. f. International Environmental Law. Deep Trouble The Risks of Offshore Carbon Capture and Storage. pages 1–61, 2023.

K. Kawaguchi, L. P. Kaelbling, and Y. Bengio. Generalization in Deep Learning. *Mathematical Aspects of Deep Learning, Cambridge University Press*, 2020. URL `http://arxiv.org/abs/1710.05468`.

N. Kazemi, J. Wong, H. Zhang, K. Bertram, K. Innanen, and R. Shor. Seismic illumination analysis through physical modeling. In *Geoconvention 2020*, pages 1–5, 2020.

F. Khosro Anjom, F. Vaccarino, and L. V. Socco. Machine learning for seismic exploration: Where are we and how far are we from the holy grail? *Geophysics*,

89(1):157–178, 11 2023. ISSN 0016-8033. doi: 10.1190/geo2023-0129.1. URL https://doi.org/10.1190/geo2023-0129.1.

D. Komatitsch, C. Barnes, and J. Tromp. Wave propagation near a fluid-solid interface: A spectral-element approach. *Geophysics*, 65(2):623–631, 2000. ISSN 00168033. doi: 10.1190/1.1444758.

C. Lacombe and C. Shelf. Correcting for water-column variations. *The Leading Edge*, pages 198–201, 2009. doi: 10.1190/1.3086058.

C. Lacombe, J. Schultzen, S. Butt, and D. Lecerf. Correction for water velocity variations and tidal statics. *SEG Technical Program Expanded Abstracts*, pages 2881–2885, 2006. doi: 10.1190/1.2370124.

A. Laugros, A. Caplier, and M. Ospici. Are adversarial robustness and common perturbation robustness independant attributes? *CoRR*, abs/1909.02436, 2019. URL http://arxiv.org/abs/1909.02436.

W. Laurson. Autonomous survey technology: Cutting the umbilical. https://www.marinetechnologynews.com/news/autonomous-survey-technology-cutting-637782, 2024. Accessed: 2024-07-31.

Y. Lecun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015. ISSN 14764687. doi: 10.1038/nature14539.

B. Li, P. Qi, B. Liu, S. Di, J. Liu, J. Pei, J. Yi, and B. Zhou. Trustworthy ai: From principles to practices. *ACM Computing Surveys*, 55(9), January 2023. ISSN 0360-0300. doi: 10.1145/3555803. URL https://doi.org/10.1145/3555803.

H. Li, Z. Xu, G. Taylor, C. Studer, and T. Goldstein. Visualizing the loss landscape of neural nets. *Advances in Neural Information Processing Systems*, (NeurIPS 2018): 6389–6399, 2018. ISSN 10495258. doi: 10.48550/arXiv.1712.09913.

Z. Liu, J. Huang, and J. Li. Comparison of four techniques for estimating temporal change of seismic velocity with passive image interferometry. *Earthquake Science*, 23 (5):511–518, 2010. ISSN 16744519. doi: 10.1007/s11589-010-0749-z.

M. Lowe. Waves in an Unbounded Medium. *Encyclopedia of Vibration*, pages 1565–1570, 2001. doi: 10.1006/rwvb.2001.0140.

D. Lumley. The role of geophysics in carbon capture and storage. In *Geophysics and Geosequestration*, chapter 2, pages 12–53. Cambridge University Press, 2019. ISBN 9781316480724. doi: 10.1017/9781316480724.003. URL https://doi.org/10.1017/9781316480724.003.

D. E. Lumley and R. A. Behrens. Practical Issues of 4D Seismic Reservoir Monitoring: What an Engineer Needs to Know. *SPE Reservoir Evaluation & Engineering*, 1(6): 528–537, 1998. ISSN 08859248. doi: 10.2118/53004-pa.

J. Luo, Y. Xie, M. Z. Hou, Y. Xiong, X. Wu, C. T. Lüddeke, and L. Huang. Advances in subsea carbon dioxide utilization and storage. *Energy Reviews*, 2(1):100016, 2023a. ISSN 27729702. doi: 10.1016/j.enrev.2023.100016. URL https://doi.org/10.1016/j.enrev.2023.100016.

J. Luo, Y. Xie, M. Z. Hou, Y. Xiong, X. Wu, C. T. Lüddeke, and L. Huang. Advances in subsea carbon dioxide utilization and storage. *Energy Reviews*, 2(1):100016, 2023b. ISSN 27729702. doi: 10.1016/j.enrev.2023.100016. URL https://doi.org/10.1016/j.enrev.2023.100016.

J. F. Lynch, G. Jin, R. Pawlowicz, D. Ray, A. J. Plueddemann, C.-S. Chiu, J. H. Miller, R. H. Bourke, A. R. Parsons, and R. Muench. Acoustic travel-time perturbations due to shallow-water internal waves and internal tides in the Barents Sea Polar Front: Theory and experiment. *The Journal of the Acoustical Society of America*, 99(2): 803–821, 1996. ISSN 0001-4966. doi: 10.1121/1.414657.

N. Mallik, E. Bergman, C. Hvarfner, D. Stoll, M. Janowski, M. Lindauer, L. Nardi, and F. Hutter. PriorBand: Practical Hyperparameter Optimization in the Age of Deep Learning. *Advances in Neural Information Processing Systems*, 36:1–15, 2023. ISSN 10495258.

G. Martin. The Marmousi2 Model, Elastic Synthetic Data, and an Analysis of Imaging and AVO in a Structurally Complex Environment. *Allied Geophysics Laboratory - University of Houston Repository*, page 211, 2004.

A. Mateeva. Personal conversation at Shell, June 2021.

A. Mateeva, J. C. Hornman, P. Hatchell, H. Potters, and J. Lopez. Frequent seismic monitoring for pro-active reservoir management. *SEG Technical Program Expanded Abstracts*, 34:4817–4821, 2015. ISSN 19494645. doi: 10.1190/segam2015-5899850.1.

T. D. Mikesell, A. E. Malcolm, D. Yang, and M. M. Haney. A comparison of methods to estimate seismic phase delays: Numerical examples for coda wave interferometry. *Geophysical Journal International*, 202(1):347–360, 2015. ISSN 1365246X. doi: 10.1093/gji/ggv138.

J. Mingers. Abduction: The missing link between deduction and induction. A comment on Ormerod's rational inference: Deductive, inductive and probabilistic thinking. *Journal of the Operational Research Society*, 63(6):860–861, 2012. ISSN 01605682. doi: 10.1057/jors.2011.85.

T. M. Mitchell, R. M. Keller, and S. T. Kedar-Cabelli. Explanation-Based Generalization: A Unifying View. *Machine Learning*, 1(1):47–80, 1986. ISSN 15730565. doi: 10.1023/A:1022691120807.

N. C. Nanda. Seismic Interpretation Methods. In *Seismic Data Interpretation and Evaluation for Hydrocarbon Exploration and Production*, pages 37–72. Spinger, 2016. ISBN 9783319264899. doi: 10.1007/978-3-319-26491-2.

U. Nations. Paris Agreement. In *Framework Convention on Climate Change*, Paris, 2015. doi: 10.17580/gzh.2021.07.02.

Obobi Ume Onwuka and Akinsola Adu. Subsurface Carbon Sequestration Potential in Offshore Environments: a Geoscientific Perspective. *Engineering Science & Technology Journal*, 5(4):1173–1183, 2024. ISSN 2708-8944. doi: 10.51594/estj.v5i4.994.

M. Raghu, B. Poole, J. Kleinberg, S. Ganguli, and J. S. Dickstein. On the expressive power of deep neural networks. *34th International Conference on Machine Learning, ICML 2017*, 6:4351–4374, 2017.

O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. *IEEE Access*, 9:16591–16603, 2021. ISSN 21693536. doi: 10.1109/ACCESS.2021.3053408.

T. Spinner, J. Körner, J. Görtler, and O. Deussen. Towards an interpretable latent space. https://thilospinner.com/towards-an-interpretable-latent-space/, year = 2024, note = Accessed: 2024-07-31.

N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and T. Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014. ISSN 03702693. doi: 10.1016/0370-2693(93)90272-J.

S. Stein and M. Wysession. *An Introduction to Seismology, Earthquakes and Earth Structure*. Blackwell Publishing, Oxford, UK, 2003.

U. S. G. Survey. Marine seismic imaging, 2023. URL https://www.usgs.gov/programs/coastal-and-marine-hazards-and-resources-program/science/marine-seismic-imaging. Accessed on June 18, 2024.

R. Taori, A. Dave, V. Shankar, N. Carlini, B. Recht, and L. Schmidt. Measuring robustness to natural distribution shifts in image classification. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 18583–18599. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/d8330f857a17c53d217014ee776bfd50-Paper.pdf.

L. Teixeira and W. M. Lupinacci. Elastic properties of rock salt in the santos basin: Relations and spatial predictions. *Journal of Petroleum Science and Engineering*, 180:215–230, 2019. ISSN 0920-4105. doi: https://doi.org/10.1016/j.petrol.2019.05.024. URL https://www.sciencedirect.com/science/article/pii/S0920410519304784.

G. J. Vermeer. Acquisition/processing - 3D seismic survey design optimization. *The Leading Edge*, 22(10):934–941, 2003. ISSN 1070485X. doi: 10.1190/1.1623633.

K. Wang, P. Hatchell, C. Udengaard, K. Craft, and S. Dunn. Direct measurement of water velocity and tidal variations in marine seismic acquisition. In *2012 SEG Annual Meeting*, pages 1–5, Las Vegas, 2012. ISBN 9781622769452. doi: 10.1190/segam2012-0806.1.

Y. Wang, H. Yao, and S. Zhao. Auto-encoder based dimensionality reduction. *Neurocomputing*, 184:232–242, 2016. ISSN 18728286. doi: 10.1016/j.neucom.2015.08.104. URL http://dx.doi.org/10.1016/j.neucom.2015.08.104.

H. Yan, A. Bakk, M. Duda, R. M. Holt, and S. Lozovyi. Overburden 4D seismic analysis: Influence of stress and pore-pressure changes accounting for elastic contrast between a reservoir and its anisotropic surrounding rocks. *Geophysics*, 88(4):MR171–MR184, 2023. ISSN 19422156. doi: 10.1190/GEO2022-0033.1.

S. Yu and J. Ma. Data-driven geophysics: from dictionary learning to deep learning, 2020. URL https://arxiv.org/abs/2007.06183.

S. Yu and J. Ma. Deep Learning for Geophysics: Current and Future Trends. *Reviews of Geophysics*, 59(3):1–36, 2021. ISSN 19449208. doi: 10.1029/2021RG000742.

F. Zhu, S. Ma, Z. Cheng, X.-Y. Zhang, Z. Zhang, and C.-L. Liu. Open-world machine learning: A review and new outlooks, 2024. URL https://arxiv.org/abs/2403.01759.