Universidade Federal de São Carlos– UFSCar
Centro de Ciências Exatas e de Tecnologia– CCET
Departamento de Computação– DC
Programa de Pós-Graduação em Ciência da Computação– PPGCC

Lucas Candiani Souza

PDARTS: Projected Differentiable Architecture Search for Seismic Inversion

São Carlos
2024

UNIVERSIDADE FEDERAL DE SÃO CARLOS

Centro de Ciências Exatas e de Tecnologia
Programa de Pós-Graduação em Ciência da Computação

Folha de Aprovação

Defesa de Dissertação de Mestrado do candidato Lucas Candiani Souza, realizada em 23/07/2025.

Comissão Julgadora:

Prof. Dr. Murilo Coelho Naldi (UFSCar)

Profa. Dra. Heloisa de Arruda Camargo (UFSCar)

Prof. Dr. Abelardo Gonçalves Pinto (CATI)

O Relatório de Defesa assinado pelos membros da Comissão Julgadora encontra-se arquivado junto ao Programa de Pós-Graduação em Ciência da Computação.

Lucas Candiani Souza

PDARTS: Projected Differentiable Architecture Search for Seismic Inversion

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação do Centro de Ciências Exatas e de Tecnologia da Universidade Federal de São Carlos, como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação.

Área de concentração: Aprendizado de Máquina

Orientador: Murilo Naldi

São Carlos
2024

Agradecimentos

Aos meus pais, Sebastião e Lígia, e minhas irmãs Laís e Lívia, pelo apoio ao longo de toda a minha jornada acadêmica.

À minha namorada, Samara, por dividir a vida comigo e estar ao meu lado nos momentos difíceis.

Aos meus amigos pesquisadores, Lucas Cardoso Silva e Carlos Gomes de Carvalho Jr., pela troca mútua de conhecimento e incentivos.

Ao meu orientador, Prof. Dr. Murilo Naldi, minha sincera gratidão pelos conselhos oferecidos, que com sua experiência e competência me ajudou a me desenvolver como pesquisador.

Ao corpo docente do Departamento de Computação da Universidade Federal de São Carlos, pelos conhecimentos compartilhados em aula.

À Financiadora de Estudos e Projetos (FINEP), e ao Prof. Dr. Hermes Senger, agradeço a oportunidade de participar do projeto de pesquisa como bolsista.

À Universidade Federal de São Carlos, meu agradecimento pela excelência de seu programa de pós-graduação.

Abstract

Seismic inversion is an inverse problem that minimizes both amplitude and phase differences between simulated signals and real observed signals through an optimization problem. Due to its high computational cost and the industry demand for higher resolution, researchers often explore ways to accelerate the process and improve its accuracy. In the last decade, deep learning emerged as a promising alternative for seismic inversion; however, still demanding laborious trial and error processes, integration of domain knowledge, and hyperparameter tuning. To improve model architectures for this task, this paper introduces PDARTS (Projected Differentiable Architecture Search), a method inspired by DARTS which aims to leverage the generalization capability of neural blocks such as Fourier and U-Fourier blocks for the seismic inversion task. Because Fourier-based blocks rely on Fourier transforms, which require square inputs, the asymmetric shape (height, width) of original seismic data necessitates enforcing a square input format to ensure proper operation. To achieve this, PDARTS employs fixed encoder layers connected to a projection layer to reshape spatial dimensions and a convolutional layer to mitigate noise. The decoder, along with the final layers of the encoder, is implemented as a DARTS supernetwork, where neural architecture search (NAS) is conducted to explore the potential of Fourier and U-Fourier neural blocks, aiming to discover new neural networks with better performance. Experiments demonstrated that the best architecture discovered by our method, PDARTSNet, outperforms current state-of-the-art neural networks for seismic inversion. Furthermore, it demonstrates that despite the increased number of network parameters and consequent higher computational costs, PDARTS has proven capable of discovering neural networks with superior performance for seismic inversion.

Keywords: Full-Waveform Inversion, Deep Learning-based Inversion, Neural Architecture Search, Fast-FWI, Fourier Neural Operators.

Resumo

A inversão sísmica é um problema inverso que busca minimizar, via otimização, as diferenças de amplitude e fase entre sinais simulados e sinais reais observados. Em razão de seu alto custo computacional e da demanda por maior resolução na indústria, pesquisadores costumam investigar formas de acelerar o processo e aprimorar sua precisão. Na última década, o aprendizado profundo surgiu como uma alternativa promissora para inversão sísmica; entretanto, ainda depende de extensos testes por tentativa e erro, incorporação de conhecimento de domínio e ajuste de hiperparâmetros. Para melhorar as arquiteturas de modelos nessa tarefa, este artigo apresenta o PDARTS (Projected Differentiable Architecture Search), método inspirado no DARTS que aproveita a capacidade de generalização de blocos neurais, como os de Fourier e U-Fourier, para inversão sísmica. Como esses blocos baseados em transformada de Fourier exigem entradas quadradas, o formato assimétrico dos dados sísmicos originais requer a imposição desse formato. Para isso, o PDARTS emprega camadas fixas de codificação ligadas a uma camada de projeção para ajustar as dimensões espaciais e a uma camada convolucional para atenuar ruídos. O decodificador e as camadas finais do codificador são implementados como uma super-rede DARTS, na qual a busca em arquitetura neural (NAS) explora blocos Fourier e U-Fourier, com o objetivo de descobrir novas redes de melhor desempenho. Os experimentos mostraram que a arquitetura ótima encontrada, PDARTSNet, supera as redes neurais de ponta em inversão sísmica. Além disso, apesar do aumento no número de parâmetros e no custo computacional, o PDARTS provou ser capaz de revelar arquiteturas com desempenho superior para essa aplicação.

Palavras-chave: Inversão de onda completa, Inversão de onda baseada em aprendizado profundo, busca de arquiteturas de redes neurais, aceleração de inversão de onda, operadores neurais de Fourier.

List of Figures

List of Tables

Glossary

CNN Convolutional Neural Network

DARTS Differentiable Architecture Search

DeepONet Deep Operator Networks

DNN Deep Neural Network

FNO Fourier Neural Operators

FWI Full-Waveform Inversion

FDM Finite Difference Method

GAN Generative Neural Networks

HPC High Performance Computing

NAS Neural Architecture Search

NN Neural Network

NO Neural Operators

PDE Partial Different Equations

PDARTS Projected Differentiable Architecture Search

PINN Physics-Informed Neural Networks

ReLU Rectified Linear Unit

Contents

Chapter 1

Introduction

Seismic inversion techniques transform seismic wave signal data into a quantitative earth model of the properties of rocks and other subsurface materials. Such models are widely used for seismologic studies, mining, shallow hazard assessment, CO_2 storage, or hydrocarbon exploration, among other applications (SHEARER, 2019). In the oil and gas industry, these models are critical inputs for many tasks, such as for supporting better drilling decisions, which can avoid environmental hazards and save hundreds of millions of dollars. Moreover, models can be periodically updated and utilized to enhance production efficiency throughout the full operational lifetime of the wells, thus improving industry efficiency. Another example of high societal importance involves capturing CO_2 emissions from industrial activities and storing them in specifically reconditioned subsurface reservoirs.

The Full-Waveform Inversion (FWI) is a data fitting procedure based on full-wavefield modeling to produce material models. FWI utilizes observed seismic signals as input to quantitatively characterize subsurface materials based on their physical properties, such as P-wave velocity, S-wave velocity, density, elasticity, and viscosity, among others (VIRIEUX; OPERTO, 2009). The oil and gas industry widely employs FWI (as illustrated in Figure 1) to reconstruct acoustic velocity-based models of subsurface materials for tasks including the location and characterization of oil and gas reservoirs (JONES, 2019). A velocity model represents the spatial distribution of seismic wave propagation speeds within the subsurface, typically expressed as acoustic or elastic velocities, and serves as a physical proxy for geological structures and material properties present in the surveyed area.

In summary, FWI can be divided into three subproblems (VIRIEUX et al., 2017). The *forward propagation problem* simulates the propagation of acoustic wave signals pro-

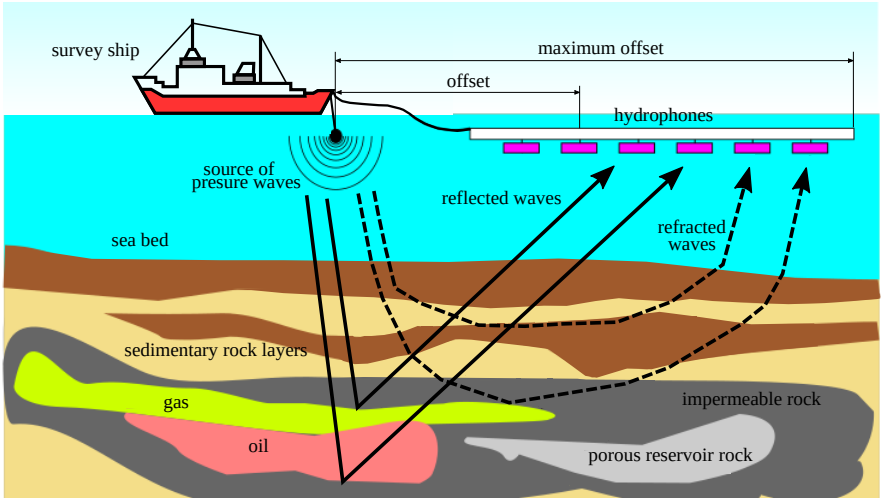


Figure 1 – Full-waveform inversion as a remote sensing tool (SOUZA et al., 2022).

duced by a well-defined source across a spatial domain. The simulation resolves Partial Different Equations (PDE) such as in (SOUZA et al., 2022) considering a velocity model and records the resulting time-dependent signals at some points, called receivers. The *adjoint propagation problem* backpropagates in time the simulated data (e.g., observed by microphones or hydrophones) to calculate the gradient of the objective function with respect to the optimization parameters (usually the velocity model). Then, the *inverse problem* minimizes both amplitude and phase differences between the simulated signals (produced by the forward propagator) and real observed signals at the receivers. The model is incrementally modified, usually employing steepest gradient descent methods, so that the functional that represents the error is sufficiently reduced (VIRIEUX; OPERTO, 2009).

Conventional seismic data impose several practical limitations, given that field recordings are often contaminated by environmental and instrumental noise, which reduces signal-to-noise ratio and requires robust preprocessing and objective-function choices to avoid fitting noise instead of geological signal (VIRIEUX; OPERTO, 2009). The lack of reliable low-frequency content in many acquisitions hinders recovery of the low-wavenumber background model and increases the risk of cycle-skipping, pushing FWI toward local minima unless multiscale or low-frequency reconstruction strategies are used (LI; DEMANET, 2016). Many practical workflows also adopt acoustic or isotropic approximations for simplicity, whereas real rocks exhibit elastic and anisotropic behavior; neglecting anisotropy and elastic effects introduces systematic modeling errors that can be misinterpreted as changes in velocity or other parameters (TARANTOLA, 1984). However, the success and robustness of this workflow depend critically on the quality of the data and on the fidelity of the physical model used for simulation. In practice, conventional field recordings are frequently contaminated by environmental and instrumental noise, which reduces the signal-to-noise ratio and requires careful preprocessing and appropriate objective function choices to avoid fitting noise instead of geological signal (VIRIEUX; OPERTO, 2009).

Likewise, the common absence of reliable low-frequency content hampers recovery of the low-wavenumber background model and increases the risk of cycle-skipping, tending to trap FWI in local minima unless multiscale or low-frequency reconstruction strategies are applied (LI; DEMANET, 2016). Finally, many practical workflows adopt acoustic or isotropic approximations for simplicity, while real rocks often behave elastically and anisotropically; neglecting elasticity and anisotropy introduces systematic modeling errors that can be misinterpreted as variations in velocity or other subsurface parameters (TARANTOLA, 1984).

Over the past decade, the substantial increase in seismic data availability, combined with the remarkable success of Deep Neural Network (DNN) across various scientific domains, has propelled the emergence of data-driven seismic inversion (illustrated in Figure 2), an approach that employs DNNs to enhance FWI solutions (ADLER; ARAYA-POLO; POGGIO, 2021). However, the design of these DNNs still relies a lot on expert domain knowledge and a laborious trial-and-error process.

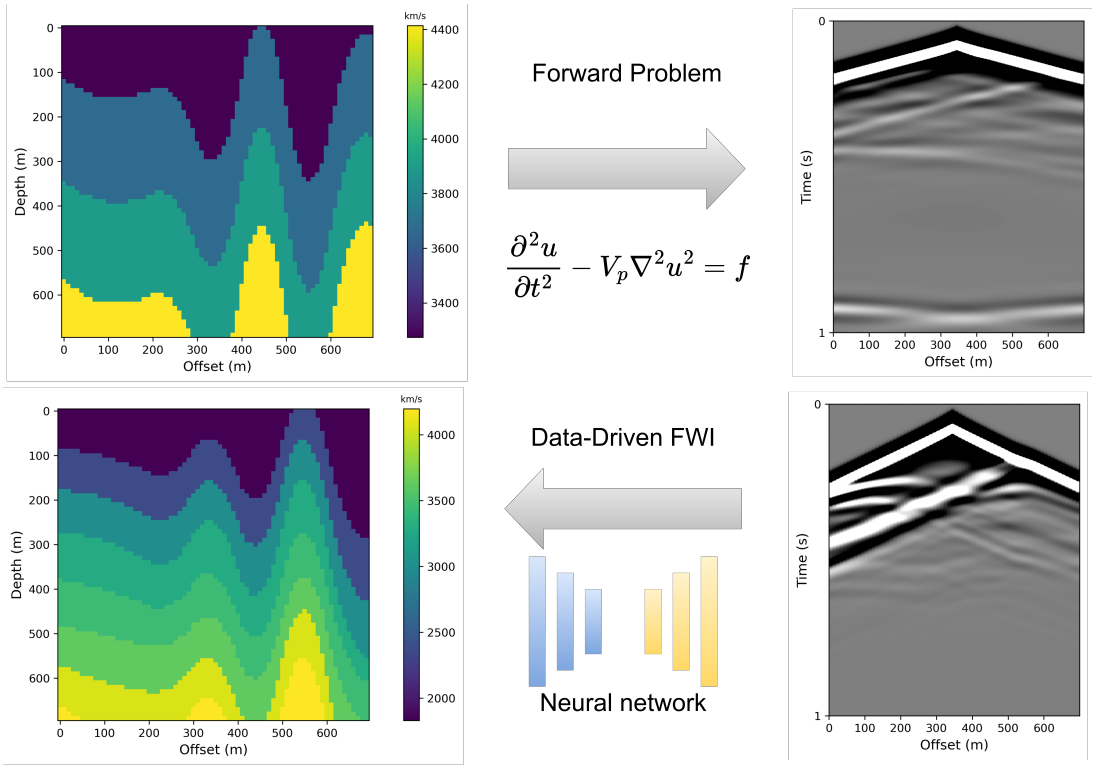


Figure 2 – Forward modeling and Data-driven FWI.

To address the increasing complexity of designing models for seismic inversion, this work explores the use of Neural Architecture Search (NAS) to optimize model architectures. NAS is an emerging research field that aims to automate the process of neural network design (BAYMURZINA; GOLIKOV; BURTSEV, 2022). Its goal is to shift the paradigm of neural network design from an expert-driven process to an almost entirely automated one. Although traditional NAS demonstrates impressive performance, it is computationally intensive because evaluating each candidate architecture requires

training and validation, resulting in poor scalability with increasing network complexity. Differentiable Architecture Search (DARTS) was proposed to address this scalability issue. Instead of searching within a discrete set of candidate architectures, DARTS relaxes the search space to be continuous. This change allows the optimization of the architecture performance on the validation set using gradient descent (LIU; SIMONYAN; YANG, 2018).

1.1 Main Objectives

This work aims to evaluate and optimize neural network architectures for the 2D seismic inversion problem using the OpenFWI dataset. In particular, the following main objectives are pursued:

- ❏ Evaluate the performance of existing neural network architectures on the 2D seismic inversion task using the OpenFWI dataset (DENG et al., 2022), a benchmark that includes large-scale, multi-structural data relevant to full waveform inversion (FWI).

- ❏ Investigate the integration of Fourier-based and U-Fourier neural blocks into deep learning models for seismic inversion, given their demonstrated effectiveness in capturing global and multi-scale features.

- ❏ Propose architectural solutions to enable the use of Fourier-based layers on input data with arbitrary spatial dimensions. This includes the design of a preprocessing pipeline based on fixed encoder layers and a spatial projection module that reshapes arbitrary input sizes into square tensors suitable for FFT operations.

- ❏ Introduce Projected Differentiable Architecture Search (PDARTS), a method based on Differentiable Architecture Search (DARTS), tailored for seismic inversion. PDARTS performs neural architecture search over a supernetwork that includes Fourier and U-Fourier blocks, expanding the design space and enabling the automatic discovery of effective architectures for FWI.

- ❏ Assess the effectiveness of PDARTS in generating high-performing architectures and compare its results with state-of-the-art models on the OpenFWI benchmark.

1.2 Secondary Objectives

In addition to the primary goals listed above, this work pursues the following secondary objectives:

- ❏ Make the PDARTS framework publicly available as an open-source tool for neural architecture search in seismic inversion, including documentation and usage examples.

- ❏ Validate the generalization capability of the best architectures discovered by PDARTS through additional benchmarks and challenges (e.g., Kaggle competitions), assessing performance on datasets beyond OpenFWI.

- ❏ Asses the impact of integrating Fourier and U-Fourier blocks on model complexity, by measuring changes in parameter count, computational cost (FLOPs), and training time.

- ❏ Document best practices and lessons learned from applying NAS to full waveform inversion, aiming to inform future research at the intersection of geophysics and automated machine learning.

1.3 Dissertation Outline

The remainder of this paper is organized as follows:

- ❏ Section 2 presents the comprehensive theoretical background necessary foundational to this work, covering the physics of full waveform inversion (acoustic wave equations and numerical solution methods), the fundamentals of convolutional neural networks, U-Nets, GANs, DeepONets, Fourier Neural Operators (including U-Fourier blocks), and the principles of Neural Architecture Search.

- ❏ Section 3 introduces PDARTS (Projected Differentiable Architecture Search), detailing how a fixed convolutional encoder and a non-parametric projection layer reshape asymmetric seismogram inputs into square tensors, followed by a DARTS supernetwork that jointly searches over convolutional, FNO/UFNO and skip-connection operations. It also describes the discretization process for selecting the dominant operation per layer; and presents PDARTSNet, the resulting architecture, alongside two manual baselines (FNONet and UFNONet) to benchmark the benefits of automated NAS.

- ❏ Section 4 surveys the landscape of deep learning methods for full waveform inversion, from early convolutional encoder–decoder models (e.g., InversionNet and U-Net variants) and GAN-based approaches, to operator-based frameworks such as FNOs and DeepONets. large-scale adaptations like BigFWI, and recent neural architecture search efforts in seismic inversion, highlighting how our PDARTS approach extends this body of work by integrating Fourier and U-Fourier blocks into the NAS search space.

- Section 5 describes the complete experimental setup: introduces the OpenFWI dataset and its ten subsets, highlights the four most challenging cases, and specifies the quantitative metrics used to evaluate performance. It then presents the comparison models and outlines the search protocol with an inner train/validation split and a held-out test set. A publicly available code repository is provided to ensure full reproducibility of the NAS experiments.

- Section 6 presents a comprehensive quantitative and qualitative evaluation: it reports MAE, RMSE, and SSIM for all models across the four challenging subsets for both L1 and L2 losses; summarizes parameter counts and average improvements; confirms statistical significance with Friedman and Nemenyi tests; and provides detailed inversion examples and error-map visualizations to illustrate model performance under increasing geological complexity.

- Section 7 summarizes the PDARTS contribution, discusses limitations related to search-time complexity and absence of model-size penalties, and outlines future work on validating arbitrary-shape input handling and integrating explicit complexity constraints into the search.

Chapter 2

Theoretical Background

This section presents the theoretical principles behind the techniques used in this work. It provides the essential framework for the methods and experiments presented.

2.1 Full Waveform Inversion

Acoustic waves are a means of energy propagation through a medium in space. These waves travel with a characteristic velocity and exhibit phenomena like diffraction, reflection, and interference as they interact with the medium. The propagation of acoustic waves can be described by pressure variation, particle velocity, particle displacement, and/or acoustic intensity. The propagation of acoustic waves is often used as a remote sensing tool to probe domains that are otherwise difficult to observe physically. Depending on the properties of the medium and the application, the simulation of acoustic waves may or may not consider variations in material density. For example, the acoustic wave equation with a constant density approximation is frequently used in seismic inversion workflows to estimate the P-wave velocity in the ground, which is later used to help locate raw material deposits such as oil and gas (VIRIEUX; OPERTO, 2009; SHEARER, 2019). In medical imaging, similar methods consider variations in the material density or elasticity to study and diagnose tumors and other lesions in the human body (GUASCH et al., 2020; XIA et al., 2017; MARIAPPAN et al., 2016). Acoustic tomography also plays an essential role in understanding and monitoring ocean processes such as the global tides and internal waves (MUNK; WUNSCH, 1982; DUSHAW et al., 1997) and atmospheric turbulence (WILSON; THOMSON, 1994). Finally, the acoustic wave can be used in structural modeling to identify failures in complex structures such as bridges and buildings (SHIOTANI et al., 2015; LI; GAN; QIN, 2017).

The simulation of acoustic waves (the forward problem in the context of FWI) uses differential equations to simulate acoustic wave propagation. There are several acoustic wave equations. Equation 1 presents the second-order equation with constant density:

$$\frac{1}{v_p^2}\frac{\partial^2 p(x,t)}{\partial t^2} - \nabla^2 p(x,t) = s(x,t) \tag{1}$$

where v_p is the velocity of the pressure wave field, and $p(x,t)$ can be expanded to three dimensions as $p(x,y,z,t)$, as well as the source $s(x,t) = s(x,y,z,t)$. Another widely used equation is the first-order variable density acoustic wave (Equation 2):

$$\begin{aligned} \frac{1}{\rho v_p^2}\frac{\partial p(x,t)}{\partial t} - \nabla . v(x,t) = s, \\ \rho\frac{\partial v(x,t)}{\partial t} - \nabla p(x,t) = 0, \end{aligned} \tag{2}$$

where p is the pressure wave field and v is a wave field vector for particle velocities (derived from displacements) along the different coordinate axes. Several other acoustic wave equations are of interest and have been implemented in previous works (VIRIEUX; OPERTO, 2009; SOUZA et al., 2022).

There are several methods for the numerical solution of differential equations in FWI, among them the finite difference (LOUBOUTIN et al., 2019) and the finite elements (ROBERTS et al., 2022). However, the Finite Difference Method (FDM) with a regular grid is the most frequently used in the industry for FWI.

The forward problem consists of simulating the seismic wavefield using PDE. Using matrix notation, we define the discrete forward problem in the time domain as (VIRIEUX; OPERTO, 2009):

$$M(x)\frac{d^2 u(x,t)}{dt^2} = A(x)u(x,t) + s(x,t), \tag{3}$$

where M and A are the mass and stiffness matrices, respectively. The term $\mathbf{s}$ represents the source, and $\mathbf{u}$ denotes the seismic wavefield. In acoustic modeling, $\mathbf{u}$ usually represents pressure, while in the elastic case, $\mathbf{u}$ usually represents horizontal and vertical particle velocities. Time is indicated by t, and spatial coordinates are denoted by $\mathbf{x}$. The complete FWI solution involves a second part which consists of an optimization problem of the objective function with the misfit of the norm l_2:

$$\min_m f(m) = \sum_{i=1}^{n_s} \frac{1}{2}\left\| d_i^{pred}(m,q_i) - d_i^{obs} \right\|_2^2, \tag{4}$$

where $f(m)$ is the objective function as a function of the discretized parameters of the model (slowness squared, where slowness = velocity^{-1}) collected in the vector $\mathbf{m}$. The index i runs over the total number of receivers n_s. The predicted data d_i^{pred} is represented by solving the following wave equation:

$$\begin{aligned} d_i^{pred}(m,q_i) &= P_r u(m) \\ u(m) &= A(m)^{-1} P_s^{\intercal} q_i. \end{aligned} \tag{5}$$

where the matrix A(m) represents the discretized wave equation parameterized by the vector of the unknown velocity model m to be determined. The vector q_i is the time-dependent distribution of sources for the i^{th} trigger record. This source function is injected into the discretized grid by the adjoint (denoted by $\intercal$) of the constraint operator P_s. This last operator restricts the wave field to the place of origin. After applying the inverse of the wave operator, we simulate the observed data by restricting the wave field to the receiver locations using the constraint operator P_r.

The term $d_i^{pred} = F(m, q_i)$ in Equation (4) simulates a given model m_i for the velocities of wave propagation and a set of signal sources q_i. This term must be calculated for each receiver (usually in the order of hundreds) and repeated an enormous amount of times in a process that seeks to minimize the error between the captured real data and the simulated data. In other words, the calculation of d_i^{pred} (the forward problem) is the most expensive part of the FWI process.

The cost for a single execution of the forward problem may involve the calculation of $\mathcal{O}(10^9)$ to $\mathcal{O}(10^{14})$ grid points. As shown in Equation (4), the FWI problem demands one execution of the forward problem for each source s (while s is typically $\mathcal{O}(10^4)$), and all the sources should be simulated around 100 to 150 iterations of the optimization loop. Thus, an industrial FWI workflow can typically require the execution time ranging from $\mathcal{O}(10^9)$ to $\mathcal{O}(10^{14})$ grid point calculations. It is a very large-scale application, which in practical terms, can occupy hundreds of machines in an High Performance Computing (HPC) cluster for weeks to months, depending on the configuration of the problem.

2.2 Convolutional Neural Networks

Convolutional Neural Network (CNN) are a class of deep neural networks that have proven to be highly effective in tasks involving image and video recognition and various other kinds of data with a grid-like topology.

Unlike fully connected layers, where each neuron is connected to every input neuron, each neuron in a convolutional layer is connected only to a small input region through filters called kernels. These kernels typically have much smaller height and width than the input but span across its entire depth (number of channels) (O'SHEA; NASH, 2015). When receiving an input, the convolutional layer slides its kernels across it, performing convolution and generating a single value for each operation, as shown in Figure 3. After the convolution operation, an element-wise non-linear activation function is applied, usually Rectified Linear Unit (ReLU), resulting in the activation map. This nonlinearity is beneficial as it assists the network in discerning non-linear features (GU et al., 2018).

The number of data points the kernel slides through between each convolution is called stride, and its default value is 1 (thus applying convolution to each possible position for the kernel in the input frame). This topology allows concepts learned by the model from one

region of the image to be applied across other regions, reducing the number of parameters compared to a traditional Neural Network (NN) and enhancing generalization capability.

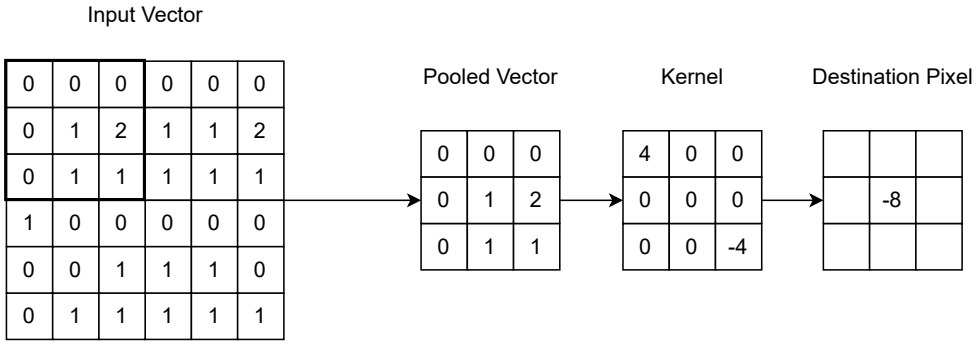


Figure 3 – A visual representation of a convolution operation. Source: (O'SHEA; NASH, 2015)

As shown above, a convolutional filter will produce an activation map whose spatial dimensions will be lower than the input due to the convolution operation. Some applications prefer to avoid this dimension reduction, and thus, they use zero padding. Zero padding involves adding a border of zeros around the activation map, ensuring that the map retains the desired dimensional size. This technique helps maintain spatial dimensions across layers. It can be crucial in specific architectures where preserving spatial information is essential for the task, such as in segmentation tasks or when working with images where spatial relationships are critical. The spatial dimensionality of the output can be defined as shown in Equation 6:

$$D_{\text{out}} = \frac{(D_{\text{in}} - K + 2P)}{S} + 1 \tag{6}$$

where D is the input spatial dimension, K is the kernel spatial dimension, P is the 0 padding, and S is the layer stride. The number of channels - or depth - will equal the number of kernels in the layer.

Reducing input dimensionality can benefit other applications. This is achieved using pooling, which computes a single value for each NxN region (typically 2x2). The most common types are MaxPooling, which returns the maximum value from each region, and AveragePooling, which returns the average. For example, fully connected layers are stacked on top after convolutional layers when the task involves classification. However, if the goal is to produce another image, deconvolutional layers are used. The section below will cover this topic.

2.3 U-Nets

For some tasks, a single label for the image (such as the result obtained using a CNN with fully connected layers on top) is not sufficient; information about the position

of the label in the image is necessary. To address this problem, (RONNEBERGER; FISCHER; BROX, 2015) proposed the U-Net, an architecture based on convolutional and deconvolutional layers that allows classifying each input image pixel.

This architecture does not have any fully connected layers, only convolutional layers, upsampling (increases the resolution of the input), and downsampling layers (decreases the resolution of the input), as seen in Figure 4. The main idea is to supplement the convolutional layers followed by pooling — where the convolution slightly reduces the spatial dimensions, while the pooling usually halves the height and width — with convolutional layers followed by upsampling operators, resulting in higher resolution output. Features from the contracting portion are combined with the output of upsampling layers to aid in localization through so-called "skip connections."

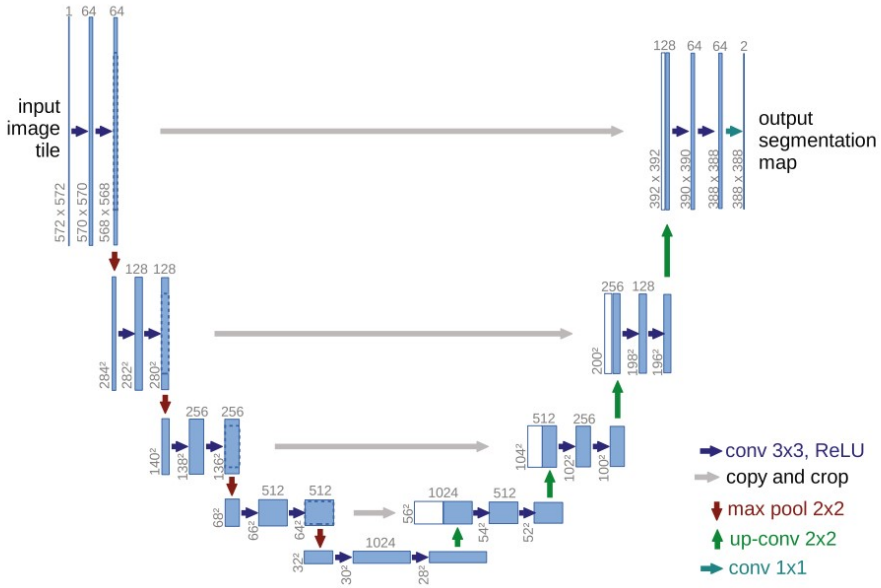


Figure 4 – A visual representation of U-net architecture. Source: (RONNEBERGER; FISCHER; BROX, 2015)

Finally, in the last deconvolution block, the feature map (which contains a feature vector for each pixel) will undergo a final convolution, 1x1, producing an output with the same spatial dimension as the feature map with a designated class for each pixel. This way, segmentation and image generation can be performed, with each class mapped to a color.

In the original U-Net architecture, the output dimensions are smaller than the input dimensions. This reduction occurs because the convolutions are performed without padding. As a result, the output represents the segmentation of a cropped version of the original image. However, it is possible to incorporate padding in each convolution to prevent the spatial dimensions from decreasing, thereby maintaining the original size throughout the convolutional blocks.

2.4 Generative Adversarial Networks

Unlike traditional machine learning algorithms based on optimization, Generative Neural Networks (GAN)s are based on game theory (GOODFELLOW et al., 2020). A GAN has two main components: the generator and the discriminator. The generator produces an image based on random Gaussian noise, and the discriminator attempts to distinguish whether the presented images are real or created by the generator.

In generative models like this, examples from the training set are treated as samples from a distribution $p_{data}(x)$. The goal of the generator is to learn a distribution $p_{model}(x)$ such that $p_{model}(x)$ is as close as possible to $p_{data}(x)$. Figure 5 shows a visual representation of the structure of a GAN, including the generator and discriminator networks, as well as the data flow.

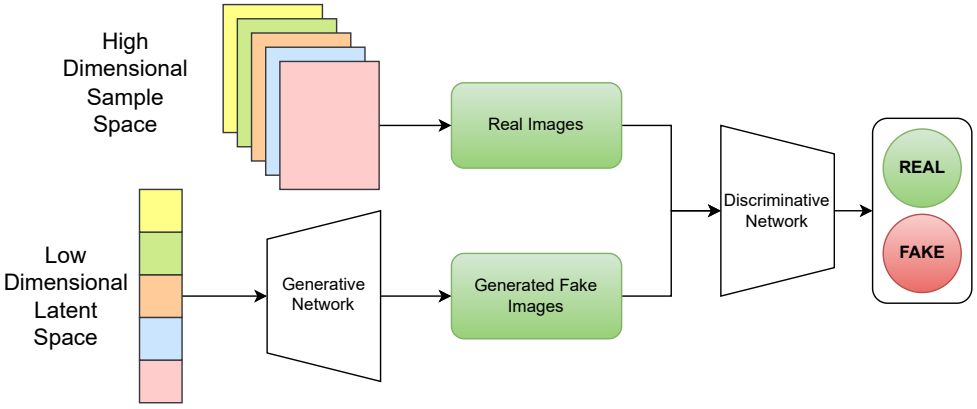


Figure 5 – Visual representation of GAN's structure

The training process is a min-max game between the two components: the discriminator is fixed, and the generator is trained to make the generated images resemble those from $p_{data}(x)$. Then, the generator is fixed, and the discriminator is trained to better distinguish the real examples from the generated ones. This alternating process continues until the discriminator can no longer distinguish real data from fake data. The objective function of the discriminator (D) can be seen in Equation 8, and the objective function of the generator (G) in Equation 7.

$$\min_G \mathbb{E}_{\mathbf{z}\sim p_{\mathbf{z}}(\mathbf{z})}[\log(1 - D(G(\mathbf{z})))] \tag{7}$$

$$\max_D \mathbb{E}_{\mathbf{x}\sim p_{\text{data}}(\mathbf{x})}[\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z}\sim p_{\mathbf{z}}(\mathbf{z})}[\log(1 - D(G(\mathbf{z})))] \tag{8}$$

where x represents real samples, z represents random Gaussian noise vectors, G(z) are images generated from the noise z, and D(x) and D(G(z)) represent the probabilities assigned by the discriminator to a given real data sample being real and a generated sample being real, respectively.

2.5 DeepONets

Deep Operator Networks (DeepONet) is a neural network architecture designed to approximate nonlinear operators between functional spaces. They differ from traditional deep learning methods, which typically focus on finite-dimensional data such as images or time series. A DeepONet consists of two main sub-networks: a "trunk" network and a "branch" network. The branch network processes an input function discretized at predetermined locations, while the trunk network inputs the coordinates of the desired output function (ZHU et al., 2023). The general layout of a DeepONet can be seen in Figure 6, where z are input parameters for the trunk net, Ψ is the discretized input function (the branch net input), $\mathcal{L}$ represents a given NN layer that will be applied after the information for each part is combined - in this case, a Fourier Neural Operators (FNO).

The output of the DeepONet is then calculated as a weighted combination of the outputs of the branch and trunk networks. This process can be understood as the branch network capturing the characteristics of the input function and the trunk network encoding information about how these characteristics should be mapped to the output function. The weighted combination allows the network to learn complex, nonlinear relationships between the input and output functions (MOLINARO et al., 2023).

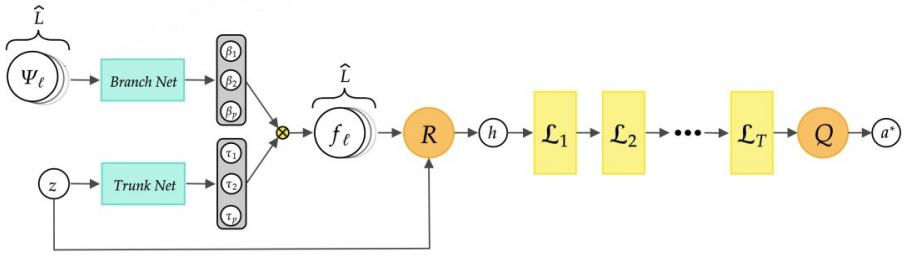


Figure 6 – Architecture of a Deep Operator Network. Source: (MOLINARO et al., 2023)

One of the main benefits of the operator regression approach offered by DeepONets is their ability to simulate complex nonlinear systems without retraining the neural network. Once a DeepONet is trained, it can quickly generate predictions for new input functions than traditional numerical methods. This ability to generalize to unseen input functions makes this architecture a powerful tool for modeling complex physical systems and solving partial differential equations.

Additionally, in the FWI application, it is possible to use the parameters employed for the geological survey that generated the seismogram, such as the source frequency and the positions of the geophones, as inputs to the trunk network (ZHU et al., 2023). This allows the network to learn how parameter changes affect the solution, resulting in a model invariant to the data collection conditions.

2.6 Fourier Neural Operators

Neural Operators (NO) refer to a class of models where mathematical operators - such as convolutions or transforms - are implemented using neural networks with the goal of enhancing the flexibility and machine learning capability in tasks involving complex mathematical operations, such as signal processing, image analysis, and solving PDEs (MOLINARO et al., 2023). This method uses an integral kernel with nonlinear activation functions, allowing the NOs to learn highly non-linear operators by composing successive layers analogous to standard neural networks. The main advantage of this approach is that it needs to be trained only once on the equations set, and then obtaining a solution for a new pair of functions only requires a forward pass of the network. However, it comes with a high computational cost, with a processing and memory complexity of $O(n^2)$ (LI et al., 2020a).

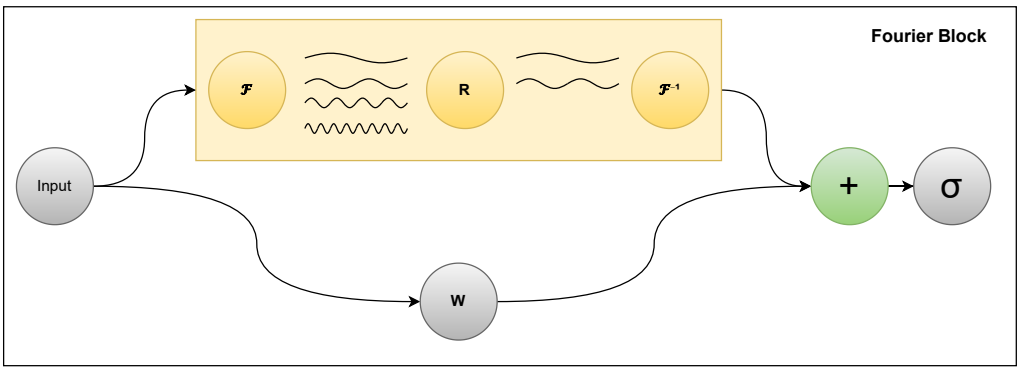


Figure 7 – Architecture of a Fourier Neural Block.

FNOs were proposed as a method to solve PDEs by parameterizing the integral kernel directly in the Fourier space (frequency domain) (LI et al., 2020b). This approach retains the advantages of standard NOs while being much faster to compute. The inner product multiplication by the weight vector has a complexity of $O(k_{\max})$, where $k_{\max}$ represents the number of frequencies extracted from the Fourier transform of the input. Therefore, most of the computational cost comes from computing the Fourier transform and its inverse. Typically, the Fourier transform has a complexity of $O(n^2)$. However, the method truncates the frequency vector obtained by the transform, limiting the complexity to $O(n*k)$, where k represents the number of frequencies after truncation. Additionally, it is possible to use the Fast Fourier Transform (FFT), which has a complexity of $O(n \log n)$, but this requires uniform data discretization; in other words, the data must be arranged in a square grid. Figure 7 shows the architecture of a Fourier block, which serves as a building block for constructing the FNO. In this diagram, $\mathcal{F}$ and $\mathcal{F}^{-1}$ are Fourier transforms and Fourier inverse transforms, respectively. R represents the truncated weight vector, W denotes a local linear transform, and σ corresponds to an activation function applied to the combined output. These components also appear in the mathematical formulation of

the FNO, shown in Equation 9, which extends the representation by including additional variables: $u(x)$ denotes an output computed from an input x; $v(x)$ denotes the intermediate representation of x at the t^{th} layer; and k is the frequency or modal parameter that indexes the spectral components of $v(x)$ obtained through the Fourier transform.

$$u(x) = \sigma\left(W^t v^t(x) + \mathcal{F}^{-1}\left(R^t \cdot \mathcal{F}(v^t)(k)\right)(x)\right) \tag{9}$$

2.6.1 U-Fourier Blocks

U-Fourier blocks were proposed for inversion problems (ZHU et al., 2023). They represent a variation of the traditional Fourier block, incorporating an internal U-Net encoder-decoder (RONNEBERGER; FISCHER; BROX, 2015), referred to as U in Figure 8. This architecture allows for a more accurate solution, as operations in the Fourier space capture low-frequency information, complemented by the U-Net capturing high-frequency information. Although the overall asymptotic complexity remains $O(n \log n)$, the extra U-Net component introduces a larger constant overhead, which increases the total number of operations and, consequently, the practical computational cost.

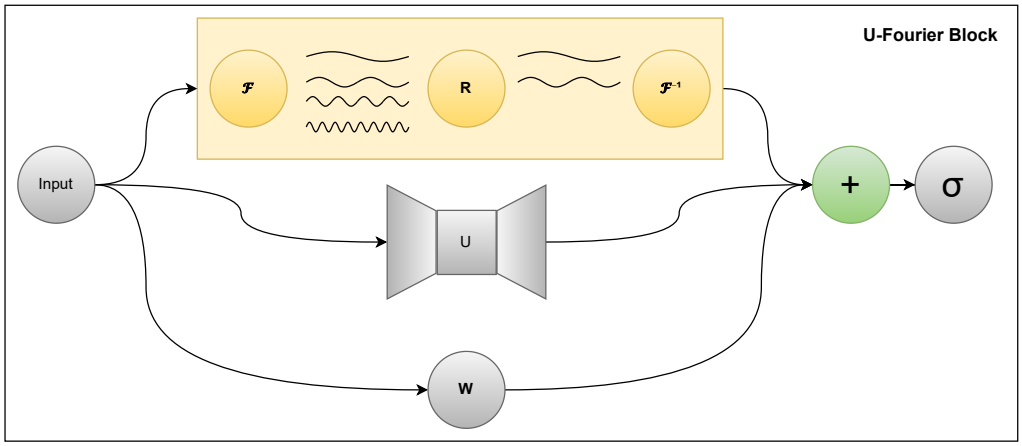


Figure 8 – Architecture of a U-Fourier Neural Block.

2.7 Neural Architecture Search

NAS is an automated process designed to find the optimal neural network architecture for a specific task, given predefined constraints. Traditional methods involve defining a search space, which represents the set of possible architectures considered during the process. This search space is often depicted as directed acyclic graphs (DAGs), where each node in the graph represents a latent representation, such as a feature map in convolutional networks. To explore this search space, common strategies include grid search, random search, and evolutionary algorithms (PHAM et al., 2018; LIASHCHYNSKYI;

LIASHCHYNSKYI, 2019; ELSKEN; METZEN; HUTTER, 2019; YING et al., 2019; ASSUNÇÃO et al., 2019). The search strategy defines how performance evaluation will be used to explore more promising architectures (ELSKEN; METZEN; HUTTER, 2019). Although traditional NAS demonstrates impressive performance, it is computationally intensive because evaluating each candidate architecture requires training and validation, resulting in poor scalability with increasing network complexity.

DARTS was proposed to address the scalability issue. Instead of searching within a discrete set of candidate architectures, DARTS relaxes the search space to be continuous. This change allows for optimization of architecture performance on the validation set using gradient descent (LIU; SIMONYAN; YANG, 2018). In this method, architectures are also represented as DAGs, where each directed edge connecting two nodes corresponds to an operation that transforms the latent representation of the first node. Learning the cell boils down to learning the operations on its edges. To make the search space continuous, the choice of a specific operation is relaxed to a softmax over all possible operations. The architecture search task is then reduced to learning a set of continuous variables.

Furthermore, DARTS uses supernets as an efficient way to represent a large search space of possible network architectures (BAYMURZINA; GOLIKOV; BURTSEV, 2022). A supernet is a comprehensive neural network that incorporates various candidate architectures as sub-networks. The advantage of using supernets lies in the weight sharing between these subnetworks, which allows for a faster and more efficient search process compared to training each candidate architecture from scratch. Essentially, methods that use supernets train the weights of the supernet once and then search for the optimal architecture by selectively activating or deactivating connections within the supernet, leveraging the knowledge embedded in the shared weights. Figure 9 depicts a super network with two subnetworks, represented in green and blue, which share the same blocks and weights depicted in red.

The objective is to find an architecture that minimizes the validation loss, where the weights associated with this architecture are obtained by minimizing the training loss. At the end of the search, a discrete architecture can be obtained by replacing each mixed operation with the most probable operation; that is, by creating a network in which each layer is defined by the operation (a candidate from the search space) that has the highest weight in the supernetwork.

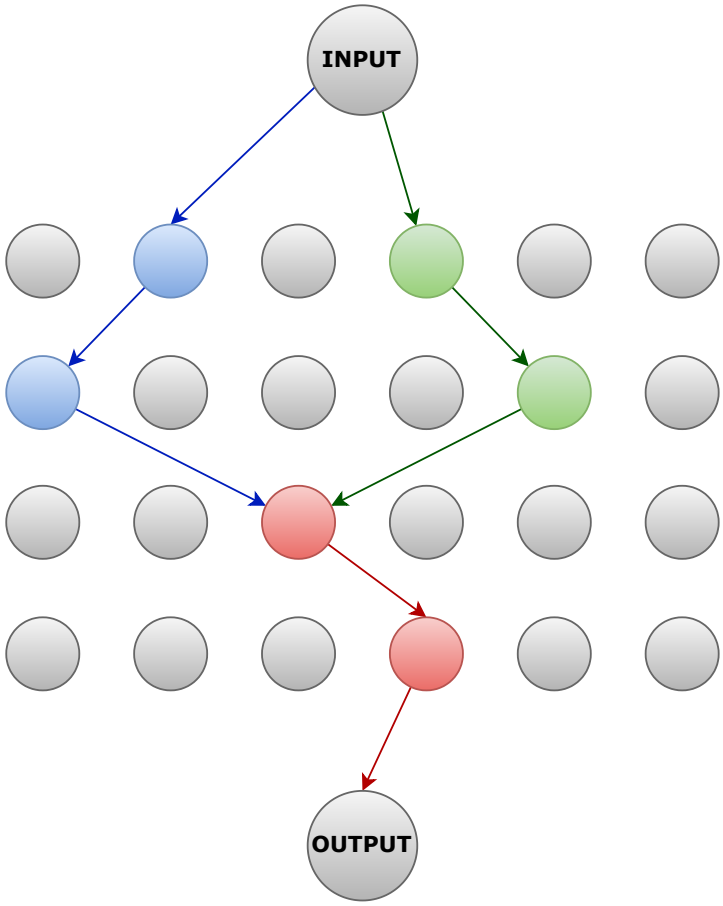


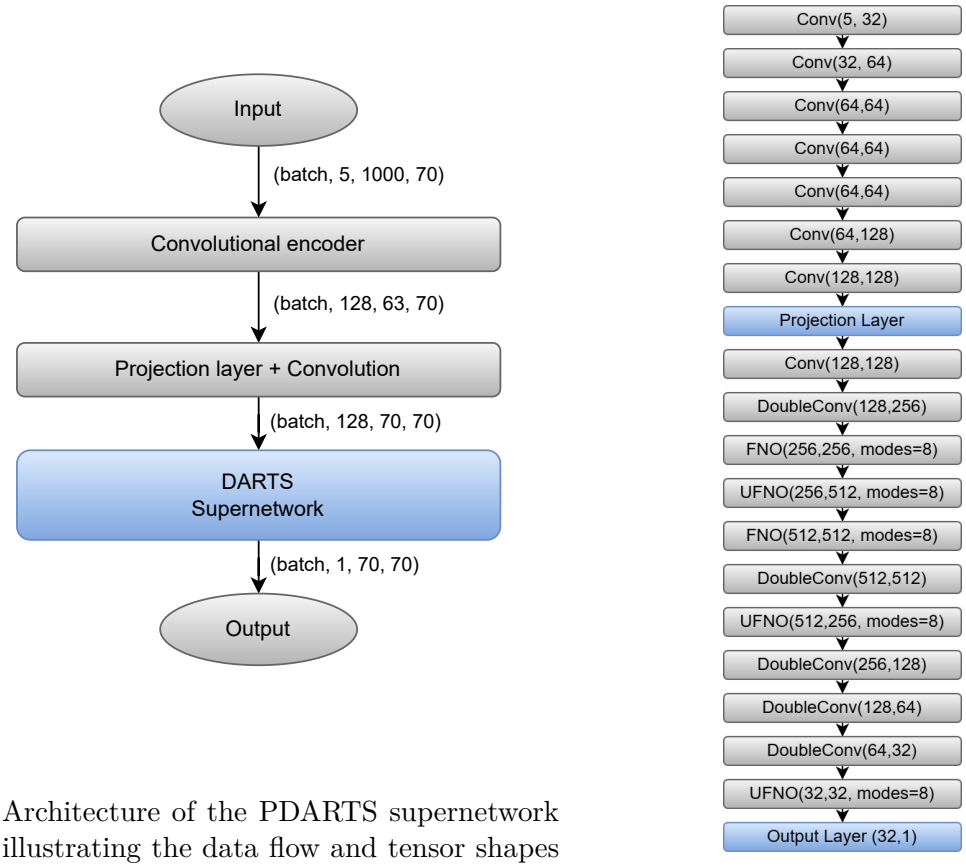
Figure 9 – Representation of a Super Network with two subnetworks.

Chapter 3

Proposal: PDARTS

FWI datasets are defined by highly asymmetric inputs and symmetric outputs, which challenges traditional U-Net architectures that preserve input spatial dimensions. To address this, OpenFWI (DENG et al., 2022) introduces convolutions with asymmetric kernels and strides to gradually reshape the input toward a square, followed by the standard U-Net stages: a contracting path, a bottleneck, and transposed convolutions in the expanding path. Notably, in this design the input does not become perfectly square during the contracting path, only achieving this form in the decoder. Applying DARTS in this context introduces additional complexity: Fourier and U-Fourier blocks require square inputs for efficient 2D Fourier transforms, while the DARTS supernetwork enforces that all candidate operations produce outputs of identical shape for a given input. This interplay necessitates explicit mechanisms to maintain shape consistency across the search space.

To minimize the number of non-parametric reshape operations, we propose a solution called PDARTS (Projected Differentiable Architecture Search). In the initial layers, a fixed convolutional encoder reshapes the input into a more square-like form with spatial dimensions of (63, 70), as illustrated in Figure 10a. This encoder follows the design of OpenFWI's InversionNet, where height contraction is combined with an increase in the number of channels through the alternating use of asymmetric kernels and strides. This strategy preserves essential information while yielding a representation more suitable for subsequent spatial and spectral operations. A single non-parametric operation, referred to as a projection, is then applied to reshape the input into a square. This is followed by a convolution for mitigating potential artifacts introduced by the reshape. The resulting output is subsequently used as the input for a DARTS supernetwork, where each layer contains a mixed-operations block encompassing all operations in the search space (see Figure 10a). These operations include convolution, double convolution, FNO, UFNO,



(a) Architecture of the PDARTS supernetwork illustrating the data flow and tensor shapes across its blocks when applied to the OpenFWI dataset.

(b) PDARTSNet, architecture discovered through our proposed method.

Figure 10 – PDARTS general structure and best model found architecture

and skip connections. During the architecture search, the importance of each candidate is learned through gradient descent applied to continuous weights. Within each supernetwork cell, the designer specifies a *steps* parameter that determines the number of intermediate nodes (and thus the effective depth) of the cell. Each step adds another layer in which every candidate primitive (standard convolution, double convolution, FNO, UFNO, and skip connection) is instantiated on each edge. Increasing the steps value yields deeper, potentially more expressive architectures but also dramatically raises the total parameter count, especially when high-parameter blocks like FNO and UFNO are included in the search space, and can exacerbate vanishing-gradient issues during training. Nevertheless, a larger search space enables the discovery of more complex architectures with greater generalization power. Conversely, reducing steps shrinks the search space and improves memory use and gradient stability at the cost of limiting architectural complexity. In our experiments, we set the number of steps to 2 and stacked 7 mixed-operation blocks before the output layer. This configuration results in a maximum of 14 layers, assuming no skip connections are selected.

Throughout the search, continuous weights assigned to each operation are optimized via gradient descent. Once the search is complete, a softmax is applied over each edge to

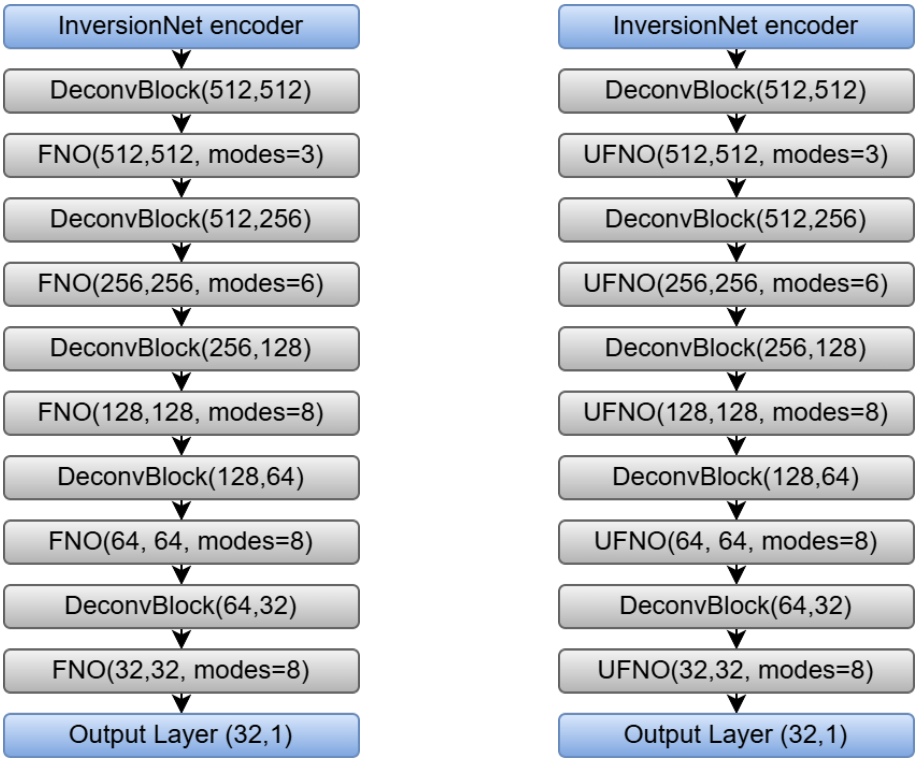
select the top-N operations, a process known as discretization, which converts the over-parameterized supernetwork into the final PDARTSNet architecture. In our experiments, discretization was performed with $N = 1$, meaning that each layer retains only a single candidate block (see Figure 10b). However, it is also common to use $N = 2$, which increases architectural complexity by allowing parallel paths within the same layer. In Figure 10b, each layer is annotated with its input and output channel dimensions, and you can clearly observe the interleaving of spatial and spectral blocks. It's important to note that the fixed encoder's convolutional kernels are intentionally asymmetric, always of size $(N, 1)$ in the $(time, space)$ dimensions ($N = 7$ for first layer, and $N = 3$ for the other fixed encoder layers), and assymetric strides of $(2, 1)$ in layers 1,2,4 and 6, to accommodate the seismogram's original shape. After the projection layer, all kernels become symmetric, ensuring compatibility with Fourier-based operations. This design prevents any shape mismatches between the outputs of different candidate operations. As a result, the PDARTS search method enables the exploration of hybrid architectures that combine spatial and spectral operations within the same network. The discretized model obtained through this process, PDARTSNet, illustrates in Figure 10b that alternating spatial convolutions, which are effective at capturing local structures and sharp interfaces, with spectral transforms, which provide global receptive fields for efficiently modeling low-frequency wave phenomena, yields clear benefits in terms of inversion accuracy and generalization. This hybrid design leverages the complementary strengths of both domains to achieve superior inversion accuracy and robust generalization. During the architecture search, the optimization is guided by the L1 loss (MAE), while the final model is later trained and evaluated using both L1 and L2 (MSE) losses, following the same protocol adopted in the OpenFWI benchmark.

To assess whether the architecture obtained through PDARTS is truly superior, two manually designed models were also considered: FNONet and UFNONet. Both networks use the same encoder as InversionNet, but differ in the blocks employed in the decoder. FNONet uses FNO blocks, while UFNONet incorporates UFNO blocks, as illustrated in Figure 11. The purpose of this comparison is to determine whether the architecture discovered through search outperforms a manually constructed network using the same building blocks.

Unlike the encoder used in PDARTSNet, the InversionNet encoder in these models is applied all the way to the bottleneck rather than stopping once a square shape is achieved. The transposed convolutional blocks in the decoder convert the linear bottleneck vector into progressively larger square forms with sides of 5, 10, 20, 40, and 80, and the output layer crops the edges to produce a final output of size $(70, 70)$. The number of frequency modes in each Fourier block is determined according to Equation 10, where N represents the spatial dimension of the square input, reflecting the limitation imposed by the input size on the maximum number of modes obtainable via FFT. For memory

considerations, 8 was chosen as the maximum number of modes. This choice is consistent with PDARTSNet, which also uses 8 modes in all Fourier layers, as the spatial dimensions of its inputs allow for this number.

$$\text{max modes} = max(8, \frac{N}{2} + 1) \tag{10}$$



(a) FNONet structure. (b) UFNONet structure.

Figure 11 – Comparison of the architecture of the two manually built models.

Chapter 4

Related Work

CNNs have been successfully employed to solve FWI problems (ARAYA-POLO et al., 2018; WANG; YANG; MA, 2018; MAO et al., 2019; WANG; MA, 2020). An encoder-decoder CNN, known as InversionNet, was introduced for accurate and efficient velocity model prediction (WU; LIN, 2018). The U-Net architecture has also been explored in the context of FWI (YANG; MA, 2019; TARGINO et al., 2020). Despite the high computational cost associated with training, significant reductions in inversion time can be achieved once a well-generalized network is obtained (YANG et al., 2016). In further work, the InvMixer architecture was proposed as a more flexible alternative that achieves twice the throughput of U-Net while using fewer parameters (ZHANG; ARAYA-POLO; SHRIVASTAVA, 2023). Generative Adversarial Networks were also proposed, such as the VelocityGAN (ZHANG; LIN, 2020), which treats waveform inversion as an image-to-image translation problem. More recently, Physics-Informed Neural Networks (PINN) have gained attention as a prominent method for solving both forward and inverse problems. The performance of PINNs was compared against classical adjoint optimization, and a hybrid method was proposed to leverage both the efficient gradient computation of the continuous adjoint method and the neural network-based representation of the unknown material field (HERRMANN et al., 2023).

The FNO was employed primarily as a fast and efficient solver for seismic wave equations in variable velocity models (LI et al., 2023). The focus was placed on learning mappings between velocity models and their corresponding wavefields in the frequency domain, particularly by solving the Helmholtz equation. The authors also proposed the Paralleled Fourier Neural Operator (PFNO) to handle multiple source locations and frequencies simultaneously. Although the primary focus is on forward modeling, the authors observed that the FNO could be an efficient method for FWI.

A Fourier DeepONet was introduced by (ZHU et al., 2023), using OpenFWI-derived data sets to simulate wave propagation at varying frequencies through velocity models, as well as data collection of receivers positioned at different locations. As a result, qualitative results exclusively for the original OpenFWI examples are not available. Analyzing the performance graphs, it is evident that the Fourier DeepONet achieves results comparable to those of InversionNet and VelocityGAN for the original frequency and receiver positions. However, the proposed network demonstrates superior performance in scenarios where these parameters vary. In contrast, our research focuses on whether NAS methods can discover architectures with improved performance by incorporating Fourier and U-Fourier neural blocks into the search space.

The BigFWI model (JIN et al., 2023) uses the original InversionNet architecture but is trained on a combined dataset that includes training examples from all subsets of OpenFWI. The model is then validated separately on each individual subset. On average, the performance improved compared to training on each subset individually, although for some subsets the results remained unchanged or even worsened.

Deep learning models for seismic inversion were improved by applying NAS (ZHAN et al., 2022). Unlike the present work, which incorporates Fourier-based blocks into the search space, their approach focused on variations of the U-Net architecture, including replacing the encoder with a pre-trained ResNet34. Using DARTS for efficient architecture search, the authors proposed NasUnet, a model that achieved competitive performance in terms of SSIM compared to U-Net and ResUnet, while significantly reducing the number of parameters. The study also explored data generation strategies using both physics-based simulations and GANs to address the limited availability of real seismic data. It is important to note that their experiments were conducted on a different dataset, generated synthetically from the SEG C3 NA model (VAILLANT, 2002).

While prior work in full waveform inversion has explored both handcrafted CNN/U-Net architectures and operator-based models such as FNOs and DeepONets, these approaches either rely on fixed design choices or focus exclusively on forward modeling. Recent NAS efforts in seismic inversion have been restricted to variations of convolutional encoder–decoder cells and ResNet-based encoders on synthetic datasets. In contrast, PDARTS represents the first differentiable search framework that incorporates Fourier and U-Fourier blocks directly into its supernetwork, enabling automatic discovery of hybrid spectral–spatial architectures. Moreover, by introducing a lightweight projection module and denoising convolution before the NAS stage, PDARTS can handle the asymmetric inputs of seismic data—something neither standard DARTS implementations nor previous spectral methods have addressed.

Chapter 5

Methods

This section outlines the experimental setup used to evaluate the impact of NAS techniques on seismic inversion tasks using the OpenFWI dataset. We describe the dataset and its subsets, the evaluation metrics employed to assess model performance, and the baseline models used for comparison. Special attention is given to the more challenging data subsets, where improvements from NAS are expected to be most pronounced. Additionally, we make available[1] the code with the details of the architecture and training procedure of the proposed models, including the NAS-based approach, to ensure reproducibility and enable a fair comparison with prior work.

5.1 Datasets

We used the OpenFWI dataset (DENG et al., 2022), which is organized into seismograms as inputs and corresponding two-dimensional velocity models as outputs. This dataset was selected because it enables reproducibility and provides a standardized benchmark for evaluating inversion methods. It is further subdivided into groups based on the geological structures present and the difficulty of the examples. The Flat and Curve subsets represent flat and curved structures, respectively, while Vel and Fault indicate the absence or presence of geological faults in the velocity models. In 2D velocity models, geological faults appear as lines that abruptly disrupt the continuity of the layers, indicating where they have shifted relative to each other. The Style subset consists of style-transferred images derived from COCO (LIN et al., 2014). Additionally, each subset is divided into variations A and B, which reflect differences in the complexity of the velocity models. In the context of seismic inversion, this complexity can be characterized by

[1] <https://github.com/HPCSys-Lab/DL-NAS-Inversion>

the structural features of the models. Simpler cases tend to exhibit smooth and continuous layers, whereas more complex ones include structural variations such as curvature, heterogeneous regions, or abrupt discontinuities like faults. The seismograms (inputs) are represented as tensors with a shape of (5, 1000, 70), corresponding to (channels, height, width), while the velocity models (outputs) are organized as (1, 70, 70). Figure 12 shows the five seismograms (one per channel) corresponding to a single velocity model. The seismograms were recorded over the same area with different acoustic source locations. Table 1 provides the number of examples available in each subset. To ensure a fair comparison with the literature, the proposed models were trained on the exact same training and validation splits.

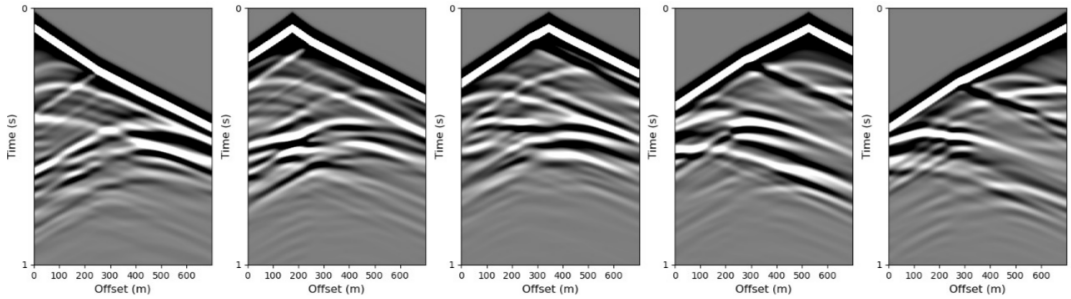


Figure 12 – Depiction of an example of the five input seismogram channels.

Given that the authors of OpenFWI achieved good results for the easier subsets, the focus of this work will be to evaluate the impact of NAS on improving the results for the more challenging subsets: CurveVel-A (CVA), CurveVel-B (CVB), FlatFault-B (FFB), and CurveFault-B (CFB). The selection of these subsets is based on their relatively lower performance in the original OpenFWI study. Investigating these more challenging cases allows for a more rigorous assessment of the potential of NAS to improve results in complex scenarios.

Table 1 – Number of examples for each OpenFWI subset.

Subset	**Train Examples**	**Validation Examples**
FlatVel-A	24000	6000
FlatVel-B	24000	6000
CurveVel-A	24000	6000
CurveVel-B	24000	6000
FlatFault-A	48000	6000
FlatFault-B	48000	6000
CurveFault-A	48000	6000
CurveFault-B	48000	6000
Style-A	60000	7000
Style-B	60000	7000

5.2 Evaluation Measures

To evaluate the results of the models, the same metrics used in the original OpenFWI study (DENG et al., 2022) will be applied: Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Structural Similarity Index Measure (SSIM), which can be respectively defined according to Equations 11, 12 and 13. In the equations, x and y represent the true and predicted images, μ_x e μ_y the mean of the pixel values in each image, σ_x^2 and σ_y^2 the variances for each image, σ_{xy} the covariance between both images and C_1 and C_2 are small constants to avoid division by 0.

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|x_i - y_i| \tag{11}$$

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - y_i)^2} \tag{12}$$

$$\text{SSIM} = \frac{1}{N}\sum_{i=1}^{N}\frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{13}$$

It is important to note that the values of μ_x, μ_y, σ_x^2, σ_y^2, and σ_{xy} in the SSIM calculation are not computed over the entire image but rather within a sliding window that moves across the image. The SSIM is computed locally for each window, and the final SSIM value is obtained by averaging the SSIM values for all windows. This allows SSIM to capture structural similarities in different regions of the image, providing a more accurate comparison than a global metric (DOSSELMANN; YANG, 2011).

5.3 Baseline Models

The baseline models used for comparison were those provided in the original OpenFWI study (DENG et al., 2022), specifically InversionNet and VelocityGAN. These models were chosen because OpenFWI offers a standardized benchmarking framework that ensures experiment reproducibility and fair comparison across different approaches. Additionally, FNONet and UFNONet were used in the comparison to evaluate how effective these Fourier transform-based blocks are for inversion tasks compared to traditional convolutional neural networks. PDARTSNet was included as the model derived from a neural architecture search using the proposed PDARTS method. The architecture search was conducted using the CurveFault-B subset, which is considered the most complex subset in the dataset. This choice is based on the assumption that a model capable of generalizing well to this subset is expected to perform adequately on simpler subsets. To avoid data leakage during the search process, the original validation set was not used. Instead, the original training set was split into new training and validation subsets, where the new validation set had the same number of examples as the original validation set. The original

validation set was reserved exclusively as a holdout test set, used only to evaluate the performance of candidate architectures. After the search phase, the final discretized model was trained on the original CurveFault-B training set and evaluated using the original validation set.

Chapter 6

Results

Tables 2, 3, and 4 present the performances of the models, including InversionNet and VelocityGAN from the original article (DENG et al., 2022), along with the proposed models, for MAE, RMSE, and SSIM, respectively. The results are shown for each evaluated subset: CurveVel-A (CVA), CurveVel-B (CVB), FlatFault-B (FFB) and CurveFault-B (CFB). Performances are reported for models trained using both the loss functions L1 and L2, where L1 corresponds to the Mean Absolute Error (MAE) and L2 to the Mean Squared Error (MSE). The best results are highlighted in bold face.

Table 2 – MAE of Models for Each Evaluated OpenFWI Subset.

	L1 Loss				L2 Loss			
Model	CVA	CVB	FFB	CFB	CVA	CVB	FFB	CFB
InversionNet	0.0685	0.1497	0.1055	0.1646	0.0690	0.1624	0.1106	0.1669
VelocityGAN	0.0482	0.1268	0.0925	0.1571	0.0510	0.1428	0.0946	0.1583
FNONet	0.0470	0.1240	0.0962	0.1506	0.0535	0.1393	0.1013	0.1555
UFNONet	0.0425	0.1093	0.0826	0.1438	0.0481	0.1220	0.0882	0.1454
PDARTSNet	**0.0287**	**0.0863**	**0.0522**	**0.0993**	**0.0342**	**0.1075**	**0.0550**	**0.1068**

By analyzing the evaluation metrics, it is evident that InversionNet produced the poorest results, performing worse than all other models across every subset. VelocityGAN and FNONet both achieved slightly better performances than InversionNet, as shown in Table 6, with increases in the number of parameters of 4.9% and 49.6%, respectively, as reported in Table 5. UFNONet achieved a more notable performance improvement but required a 225.0% increase in the number of parameters compared to InversionNet. These results suggest that including Fourier blocks only in the decoder, as done in both FNONet and UFNONet, provides limited benefit while significantly increasing model complexity. How-

Table 3 – RMSE of Models for Each Evaluated OpenFWI Subset.

	L1 Loss				L2 Loss			
Model	CVA	CVB	FFB	CFB	CVA	CVB	FFB	CFB
InversionNet	0.1273	0.2891	0.1741	0.2477	0.1202	0.2801	0.1723	0.2412
VelocityGAN	0.1034	0.2618	0.1600	0.2427	0.0976	0.2611	0.1553	0.2336
FNONet	0.1033	0.2559	0.1609	0.2305	0.1011	0.2471	0.1568	0.2235
UFNONet	0.0993	0.2460	0.1501	0.2236	0.0922	0.2335	0.1460	0.2137
PDARTSNet	**0.0048**	**0.2030**	**0.1110**	**0.1761**	**0.0633**	**0.2114**	**0.1055**	**0.1749**

Table 4 – SSIM of Models for Each Evaluated OpenFWI Subset.

	L1 Loss				L2 Loss			
Model	CVA	CVB	FFB	CFB	CVA	CVB	FFB	CFB
InversionNet	0.8074	0.6727	0.7208	0.6163	0.8223	0.6661	0.7186	0.6053
VelocityGAN	0.8624	0.7111	0.7476	0.5996	0.8758	0.6962	0.7552	0.6033
FNONet	0.8587	0.7127	0.7365	0.6259	0.8566	0.7007	0.7375	0.6248
UFNONet	0.8734	0.7453	0.7651	0.6431	0.8860	0.7538	0.7753	0.6442
PDARTSNet	**0.9303**	**0.8084**	**0.8763**	**0.7487**	**0.9404**	**0.7864**	**0.8779**	**0.7447**

ever, the internal U-Nets within the UFNO blocks appear to contribute more effectively to performance, which may explain the difference between FNONet and UFNONet.

In the case of PDARTSNet, despite a 383.6% increase in the number of parameters compared to InversionNet and a 48.8% increase compared to UFNONet, the model achieved a significant improvement in all metrics. This gain can be attributed to a more effective combination of Fourier and convolutional blocks, enabled by the neural architecture search process. Additionally, the results indicate that Fourier blocks may yield greater performance improvements when placed in the encoder, where the input still represents a time-series signal.

Table 5 – Number of Parameters for each model.

Model	Parameter Count
InversionNet	2.44×10^7
VelocityGAN	2.56×10^7
FNONet	3.65×10^7
UFNONet	7.93×10^7
PDARTSNet	1.18×10^8

A visual analysis of the results shows that, for simpler models with continuous and smooth layers, there is little difference in the inversion outputs produced by each method, as can be seen in Figure 13 for the CurveVel-B subset. However, as the complexity of the model increases - featuring more heterogeneous regions and geological faults - the differences become more evident. The velocity models generated by PDARTSNet

Table 6 – Average improvement of each model over InversionNet for all metrics.

Model	MAE improvement	RMSE improvement	SSIM improvement
VelocityGAN	12.63%	8.26%	3.94%
FNONet	13.02%	10.47%	3.98%
UFNONet	21.59%	14.99%	8.11%
PDARTSNet	42.84%	36.44%	19.25%

reveal subsurface structures with greater clarity, better separation between regions, and significantly reduced blur. This improvement is evident even when compared to both other models that incorporate Fourier blocks, reinforcing the notion that using them exclusively in the decoder provides limited benefit.

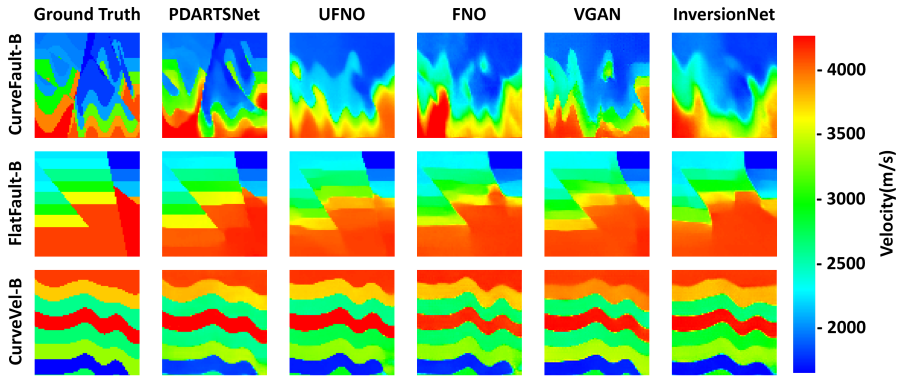


Figure 13 – Inversion examples for CFB, FFB and CVB subsets using the InversionNet (pre-trained), VelocityGAN (pre-trained), FNO, UFNO, and PDARTSNet models.

For a more detailed visual assessment, additional inversion examples are presented in Section 6.1. These include side-by-side comparisons with the true models as well as plots of the prediction error (i.e., the difference between the predicted and true velocity models). This allows for a more thorough visual analysis of the regions where each model fails to accurately capture subsurface structures.

A statistical analysis was performed to assess whether the proposed models significantly outperform the baseline methods. First, a Friedman test was conducted for each of the metrics reported in Tables 2, 3, and 4, yielding p-values of 0.00001, 0.00002, and 0.00001, respectively, indicating significant differences among the models. To identify which models differ significantly, a Nemenyi post-hoc test was applied. This test compares all pairs of models based on their average ranks across datasets. A pair is considered significantly different if the difference in their average ranks exceeds the critical distance (CD), calculated as presented in Equation 14. In this equation, q_α is the critical value from the Studentized range distribution for a given significance level, k is the number of models, and N is the number of datasets.

$$\mathrm{CD} = q_\alpha \cdot \sqrt{\frac{k(k+1)}{6N}}, \tag{14}$$

The critical distance diagrams for MAE, RMSE, and SSIM are shown in Figure 14, where models not connected by a horizontal line present statistically significant difference. The critical difference diagrams show that PDARTSNet statistically outperformed the baseline models across all evaluated subsets, confirming that the improvements observed qualitatively in the inversion examples are also supported by quantitative evidence.

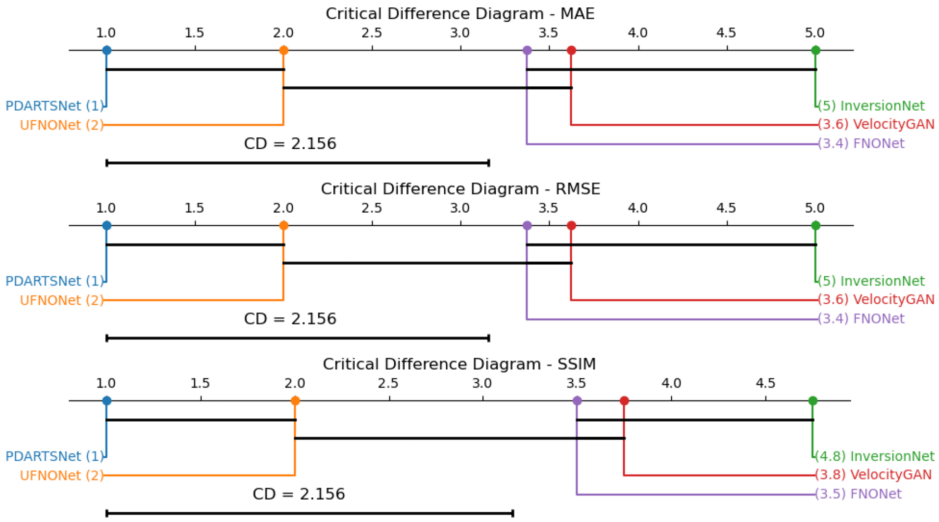


Figure 14 – Critical difference diagrams for MAE, RMSE, and SSIM, comparing the performance of the evaluated models.

6.1 Inversion Examples with Difference Plots

This section presents inversion examples in more detail, including visual comparisons between the predicted and true velocity models, as well as the corresponding error maps obtained by computing the absolute difference between them. The examples cover two subsets of the OpenFWI dataset, FlatFault-B (FFB) and CurveFault-B (CFB), chosen to represent distinct levels of structural complexity and challenge. The results are shown for all five models evaluated: InversionNet, VelocityGAN, FNONet, UFNONet, and PDARTSNet. This allows for a comprehensive qualitative comparison of their performance, particularly in capturing fault structures and sharp velocity contrasts. The error maps provide additional information on the spatial distribution and magnitude of the inversion errors, highlighting the strengths and limitations of each model. Figure 15 shows the inversion results of each model for a sample from the FFB dataset. PDARTSNet achieves the most accurate prediction, with errors confined to smaller regions of the velocity model. Although all models capture the general structure inferred from the seismogram, discrepancies become more evident in cases with increased layering and fault complexity, as illustrated in Figure 16, which presents another sample from the same dataset. In this example, PDARTSNet exhibits significantly reduced error areas and sharper layer

boundaries, while the other models produce blurrier results to varying degrees. Figures 17, 18, 19, and 20 display inversion results for samples from the CFB subset. In addition to the geological faults found in the FFB subset, the CFB models present more complex and irregular layer boundaries. Unlike the FFB subset, where boundaries are consistently straight lines, CFB includes curved, wavy, and arbitrarily shaped interfaces. These characteristics lead to more complex seismograms, making the inversion task more challenging. This effect is evident when comparing the velocity models and error maps from the FFB subset (Figures 15 and 16) with those from the CFB subset (Figures 17 to 20). In the more complex CFB scenarios, neural network predictions are generally blurrier and often fail to capture key structural features, especially in deeper layers. This is reflected in the error maps, which show larger affected regions. As confirmed by the structural similarity scores for the CFB subset in Table 4, PDARTSNet consistently delivered the most accurate inversions among the evaluated models. Nevertheless, some predictions from the proposed model better captured specific features of the ground truth, as seen in Figures 17 and 18, though other significant structures appear blurred. The lower-quality inversions often occur when the ground truth velocity model includes low-velocity regions underlying higher-velocity layers, as seen in Figures 19 and 20.

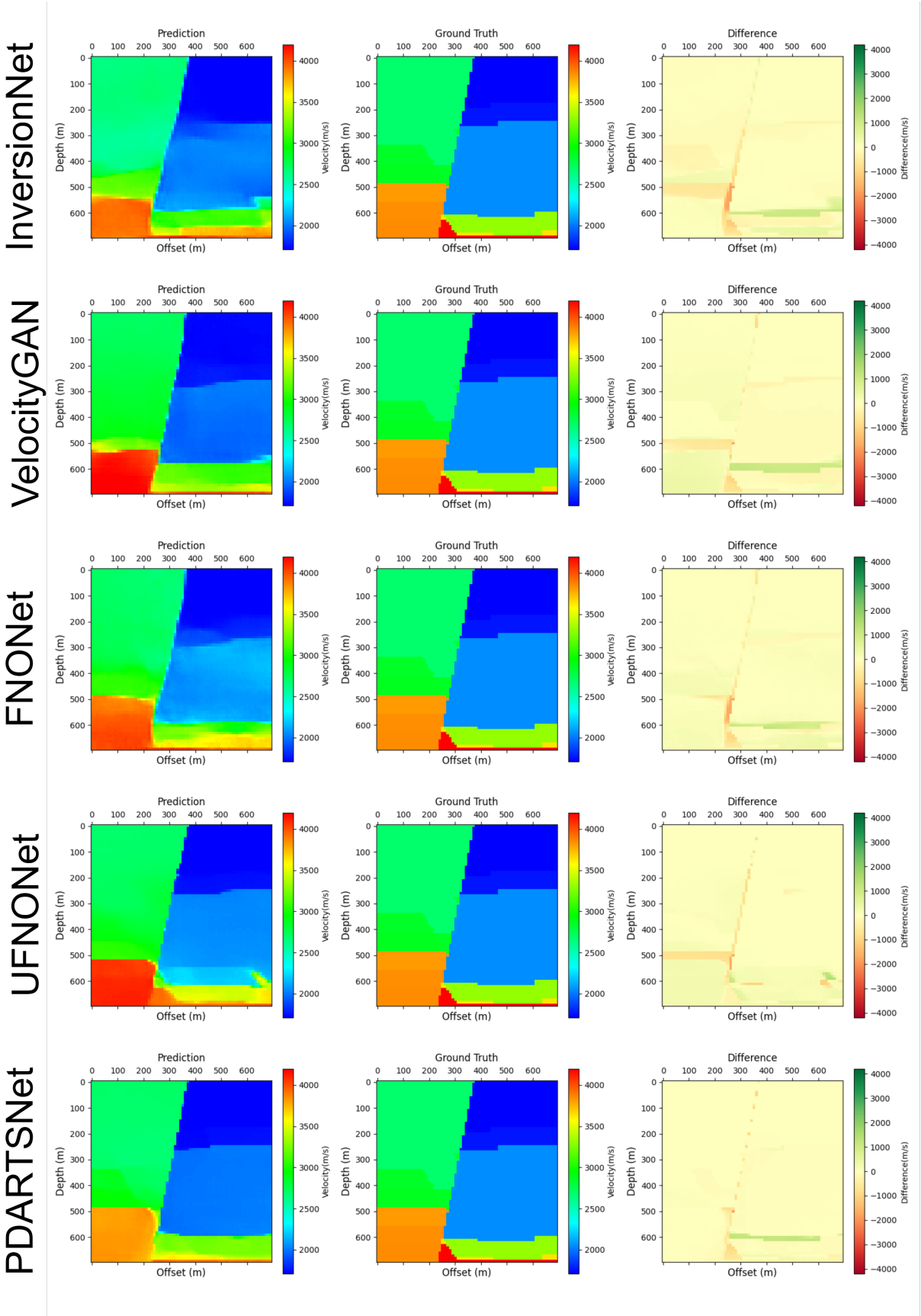


Figure 15 – Example 1 of seismic inversion on FFB subset for all evaluated models.

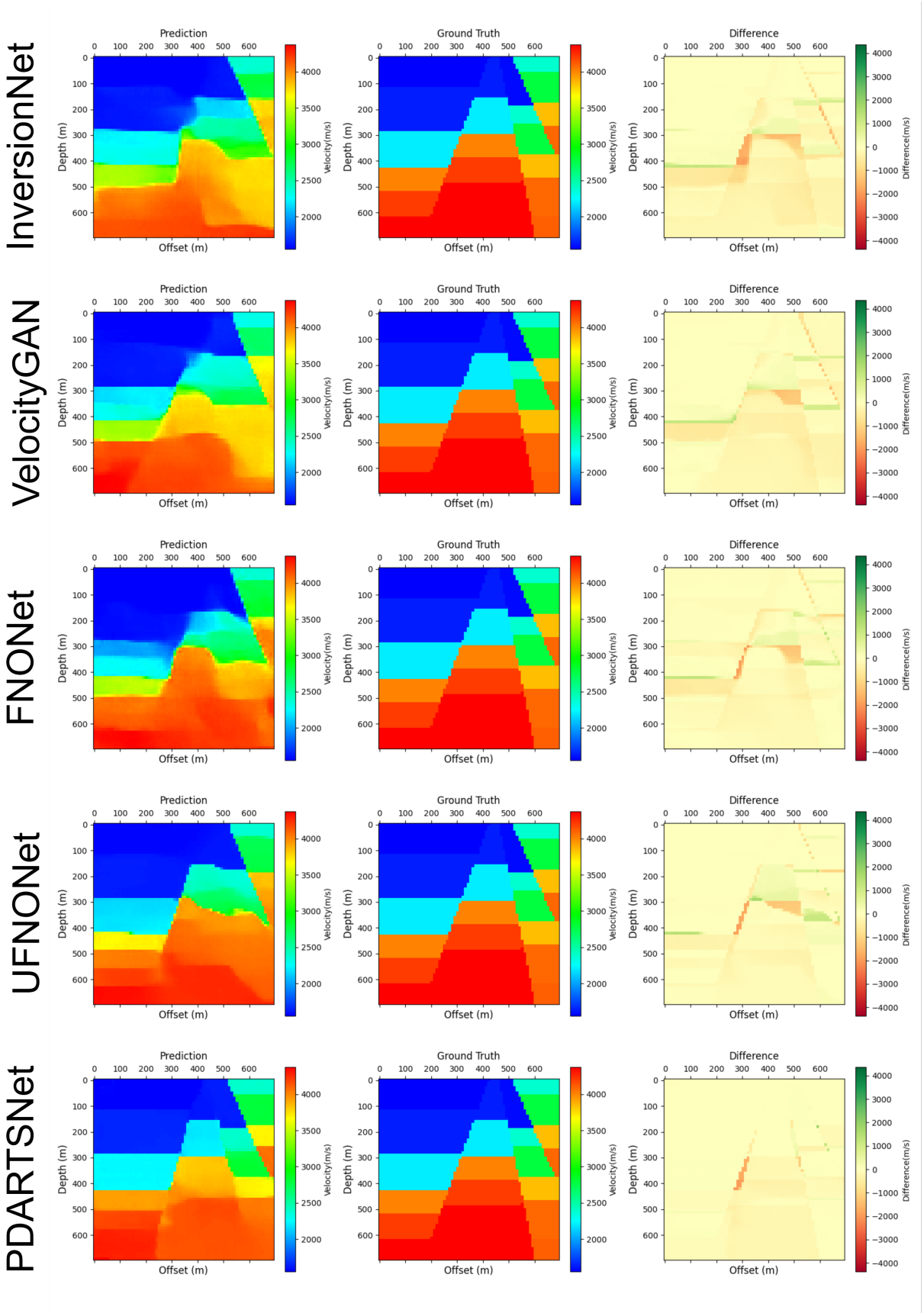


Figure 16 – Example 2 of seismic inversion on FFB subset for all evaluated models.

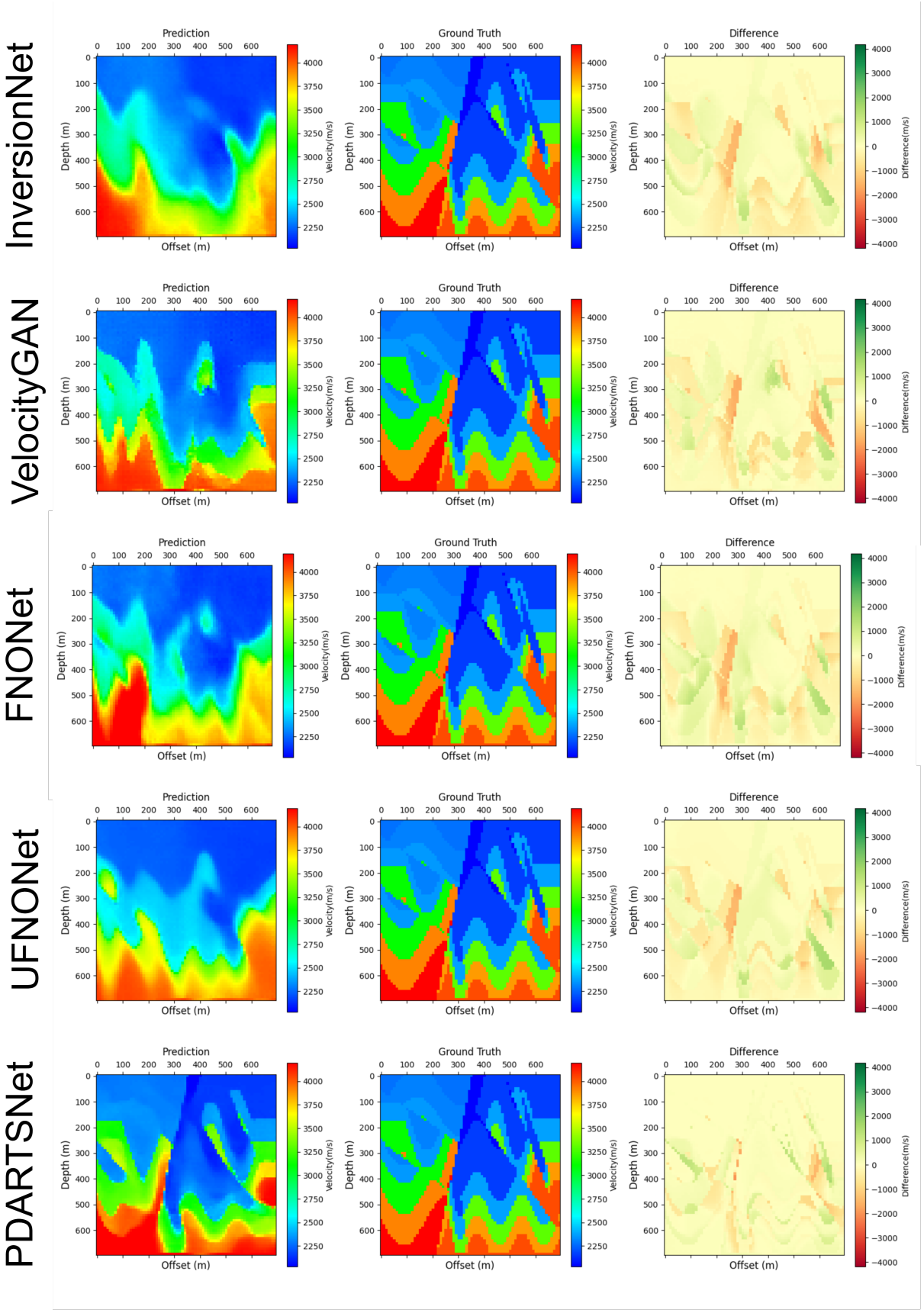


Figure 17 – Example 1 of seismic inversion on CFB subset for all evaluated models.

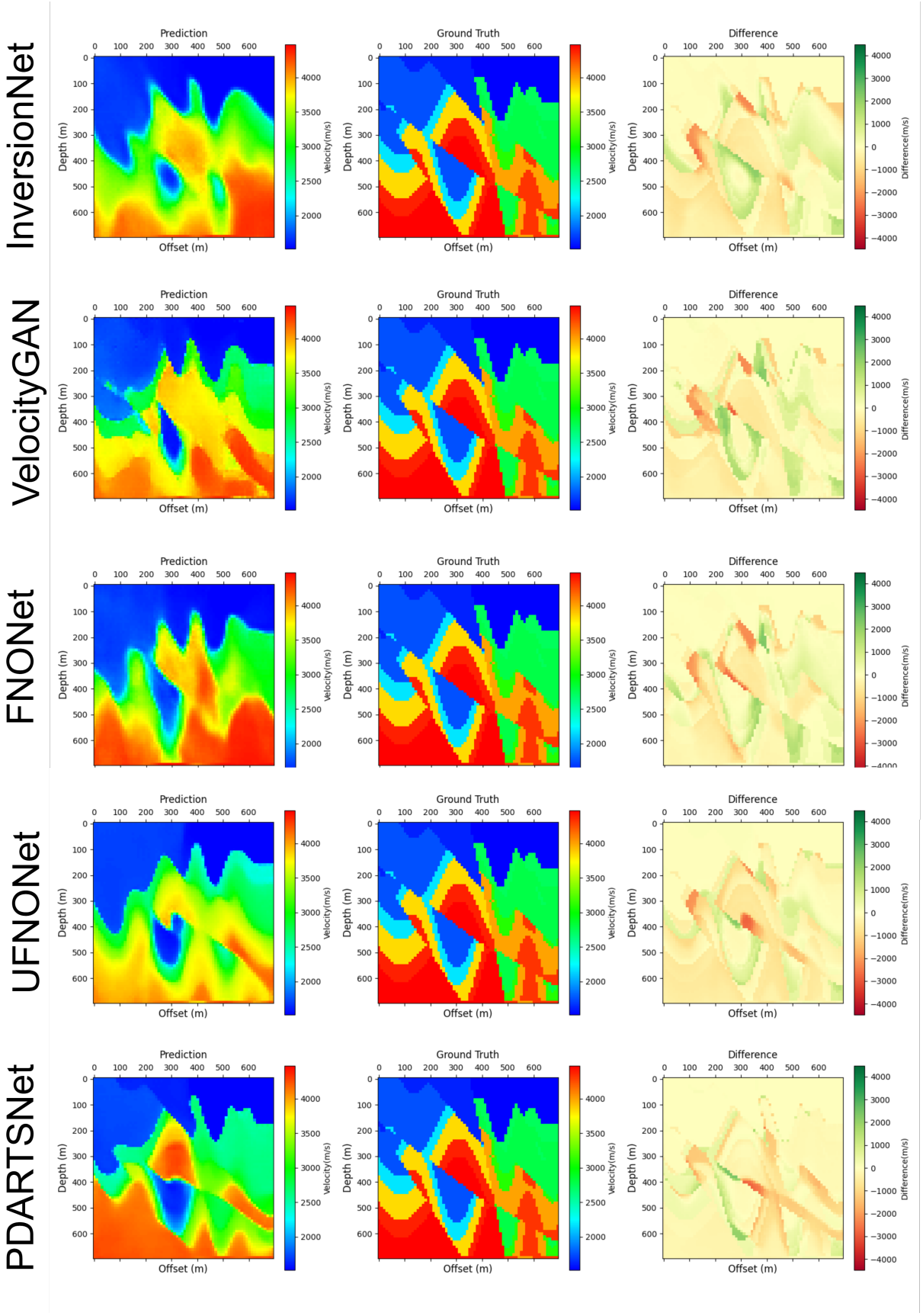


Figure 18 – Example 2 of seismic inversion on CFB subset for all evaluated models.

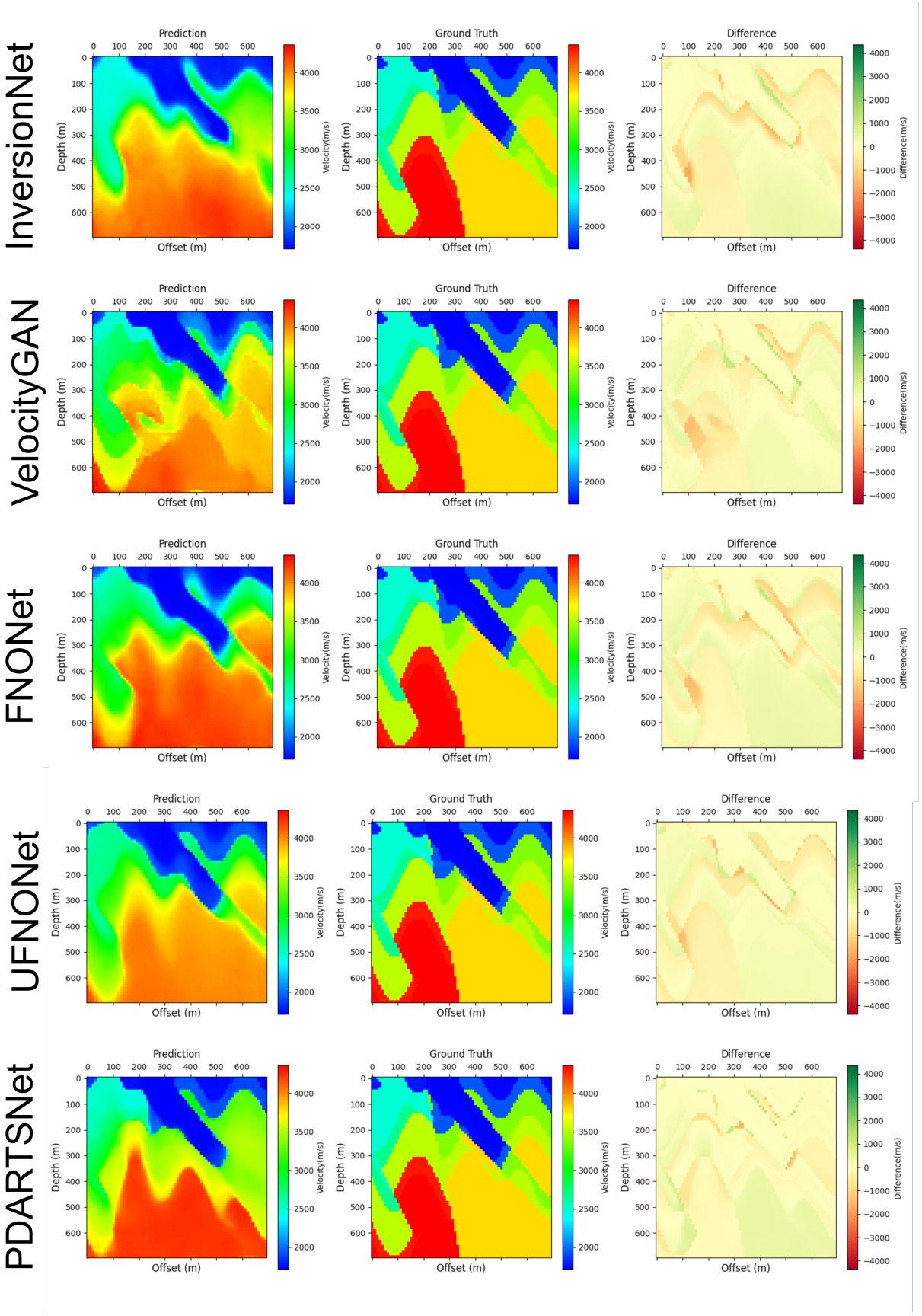


Figure 19 – Example 3 of seismic inversion on CFB subset for all evaluated models.

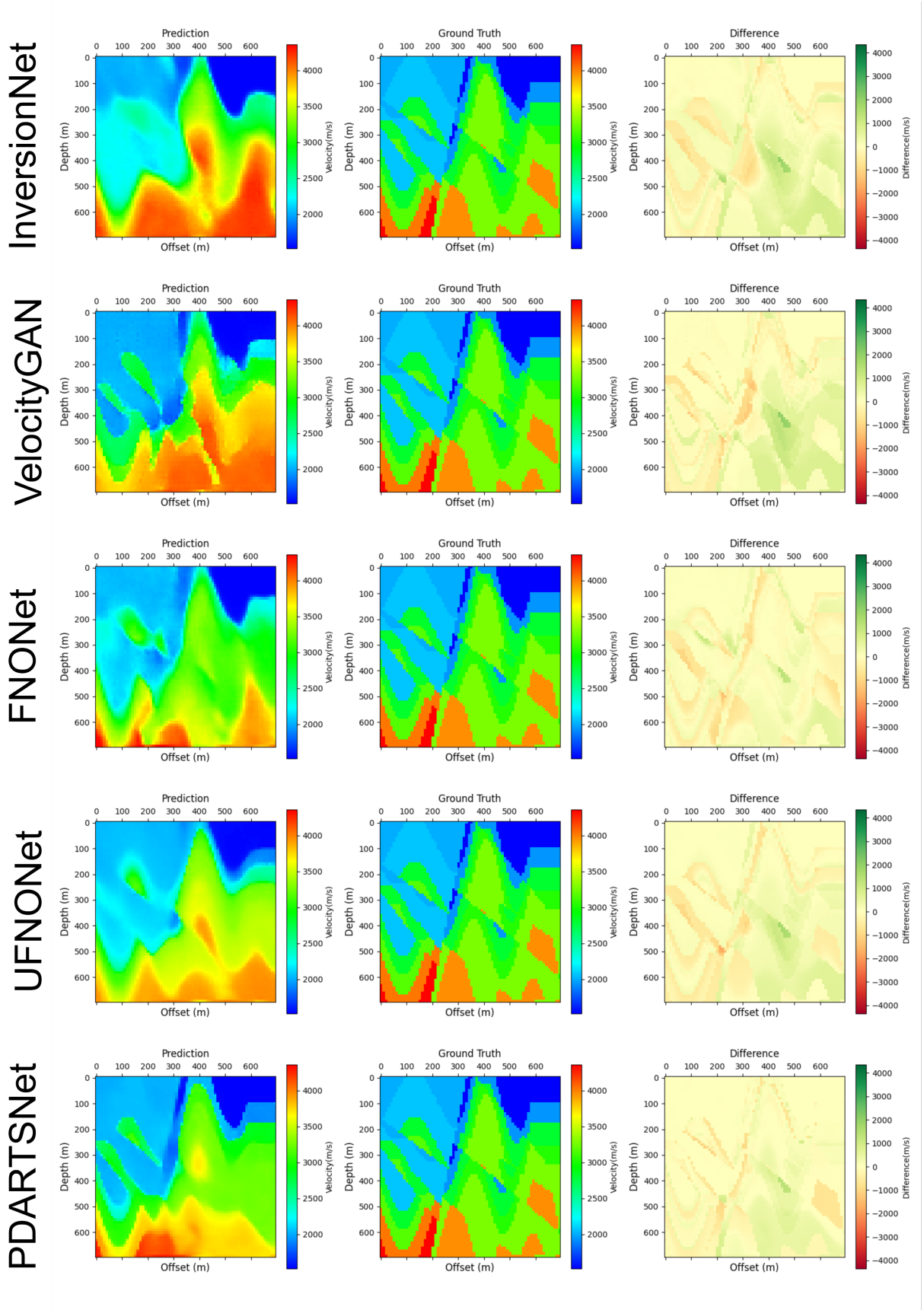


Figure 20 – Example 4 of seismic inversion on CFB subset for all evaluated models.

Chapter 7

Conclusion and Future Works

This work proposed a novel approach to adapt DARTS to the particular characteristics of seismic inversion datasets, which typically present highly asymmetric inputs and square-shaped outputs. To address this mismatch while enabling the inclusion of Fourier-based operations in the search space, we introduced PDARTS, a method that employs a fixed encoder and a single projection layer to reshape the input before feeding it into the DARTS supernetwork. The best model found through the search, referred to as PDARTSNet, was evaluated against both baselines and manually designed architectures. Across all quantitative metrics (MAE, RMSE, and SSIM) as well as qualitative inversion results, PDARTSNet consistently outperformed the compared models, demonstrating the effectiveness of the proposed strategy.

It should be noted that the search process did not include any explicit penalty for model complexity, leaving open the possibility that similar performance could be achieved with fewer parameters. Additionally, the supernetwork used during the search is significantly more demanding in terms of computational resources than the final discretized model. This becomes more critical when considering the extension to three-dimensional seismic inversion problems, where data dimensionality naturally leads to larger models. In this context, conducting an architecture search on two-dimensional datasets provides a computationally feasible way to identify architectural patterns that are likely to generalize effectively to three-dimensional problems.

As future work, we plan to evaluate the PDARTS-derived network's ability to process inputs of arbitrary shapes, leveraging the flexibility provided by the projection layer—although this has not yet been experimentally validated due to the fixed input shape of the current dataset. Another promising direction is to introduce a complexity penalty during the search to obtain more compact and scalable architectures.

References

ADLER, A.; ARAYA-POLO, M.; POGGIO, T. Deep learning for seismic inverse problems: Toward the acceleration of geophysical analysis workflows. **IEEE Signal Processing Magazine**, IEEE, v. 38, n. 2, p. 89–119, 2021.

ARAYA-POLO, M. et al. Deep-learning tomography. **The Leading Edge**, Society of Exploration Geophysicists, v. 37, n. 1, p. 58–66, 2018.

ASSUNÇÃO, F. et al. Fast denser: Efficient deep neuroevolution. In: SEKANINA, L. et al. (Ed.). **Genetic Programming**. Cham: Springer International Publishing, 2019. p. 197–212. ISBN 978-3-030-16670-0.

BAYMURZINA, D.; GOLIKOV, E.; BURTSEV, M. A review of neural architecture search. **Neurocomputing**, Elsevier, v. 474, p. 82–93, 2022.

DENG, C. et al. Openfwi: Large-scale multi-structural benchmark datasets for full waveform inversion. **Advances in Neural Information Processing Systems**, v. 35, p. 6007–6020, 2022.

DOSSELMANN, R.; YANG, X. D. A comprehensive assessment of the structural similarity index. **Signal, Image and Video Processing**, Springer, v. 5, p. 81–91, 2011.

DUSHAW, B. D. et al. A topex/poseidon global tidal model (tpxo. 2) and barotropic tidal currents determined from long-range acoustic transmissions. **Progress in Oceanography**, Elsevier, v. 40, n. 1-4, p. 337–367, 1997.

ELSKEN, T.; METZEN, J. H.; HUTTER, F. Neural architecture search: A survey. **Journal of Machine Learning Research**, v. 20, n. 55, p. 1–21, 2019.

GOODFELLOW, I. et al. Generative adversarial networks. **Communications of the ACM**, ACM New York, NY, USA, v. 63, n. 11, p. 139–144, 2020.

GU, J. et al. Recent advances in convolutional neural networks. **Pattern recognition**, Elsevier, v. 77, p. 354–377, 2018.

GUASCH, L. et al. Full-waveform inversion imaging of the human brain. **npj Digital Medicine**, v. 3, n. 1, p. 28, 2020. ISSN 2398-6352.

HERRMANN, L. et al. On the use of neural networks for full waveform inversion. **Computer Methods in Applied Mechanics and Engineering**, Elsevier, v. 415, p. 116278, 2023.

JIN, P. et al. Does full waveform inversion benefit from big data? **arXiv preprint arXiv:2307.15388**, 2023.

JONES, I. F. Tutorial: the mechanics of waveform inversion. **First Break**, European Association of Geoscientists & Engineers, v. 37, n. 5, p. 31–43, 2019.

LI, B. et al. Solving seismic wave equations on variable velocity models with fourier neural operator. **IEEE Transactions on Geoscience and Remote Sensing**, IEEE, v. 61, p. 1–18, 2023.

LI, J.; GAN, L.; QIN, H. Acoustic velocity tomography for damage detection in concrete. In: IEEE. **2017 29th Chinese Control and Decision Conference (CCDC)**. [S.l.], 2017. p. 146–149.

LI, Y. E.; DEMANET, L. Full waveform inversion with extrapolated low frequency data. **arXiv preprint**, 2016. ArXiv:1601.05232. Disponível em: <https://arxiv.org/abs/1601.05232>.

LI, Z. et al. Neural operator: Graph kernel network for partial differential equations. **arXiv preprint arXiv:2003.03485**, 2020.

_____. Fourier neural operator for parametric partial differential equations. **arXiv preprint arXiv:2010.08895**, 2020.

LIASHCHYNSKYI, P.; LIASHCHYNSKYI, P. Grid search, random search, genetic algorithm: a big comparison for NAS. **arXiv preprint arXiv:1912.06059**, 2019.

LIN, T.-Y. et al. Microsoft coco: Common objects in context. In: SPRINGER. **Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13**. [S.l.], 2014. p. 740–755.

LIU, H.; SIMONYAN, K.; YANG, Y. Darts: Differentiable architecture search. **arXiv preprint arXiv:1806.09055**, 2018.

LOUBOUTIN, M. et al. Devito (v3.1.0): an embedded domain-specific language for finite differences and geophysical exploration. **Geoscientific Model Development**, v. 12, n. 3, p. 1165–1187, 2019.

MAO, B. et al. Subsurface velocity inversion from deep learning-based data assimilation. **Journal of Applied Geophysics**, Elsevier, v. 167, p. 172–179, 2019.

MARIAPPAN, L. et al. Magneto acoustic tomography with short pulsed magnetic field for in-vivo imaging of magnetic iron oxide nanoparticles. **Nanomedicine: Nanotechnology, Biology and Medicine**, Elsevier, v. 12, n. 3, p. 689–699, 2016.

MOLINARO, R. et al. Neural inverse operators for solving pde inverse problems. **arXiv preprint arXiv:2301.11167**, 2023.

MUNK, W. H.; WUNSCH, C. I. Observing the ocean in the 1990s. **Philosophical Transactions of the Royal Society of London. Series A, Mathematical and Physical Sciences**, The Royal Society London, v. 307, n. 1499, p. 439–464, 1982.

O'SHEA, K.; NASH, R. An introduction to convolutional neural networks. **arXiv preprint arXiv:1511.08458**, 2015.

PHAM, H. et al. Efficient neural architecture search via parameter sharing. In: PMLR. **Proceedings of the 35th International Conference on Machine Learning**. [S.l.], 2018. v. 80, p. 4095–4104.

ROBERTS, K. J. et al. spyro: a firedrake-based wave propagation and full-waveform-inversion finite-element solver. **Geoscientific Model Development**, Copernicus GmbH, v. 15, n. 23, p. 8639–8667, nov. 2022. ISSN 1991-9603.

RONNEBERGER, O.; FISCHER, P.; BROX, T. U-net: Convolutional networks for biomedical image segmentation. In: SPRINGER. **Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18**. [S.l.], 2015. p. 234–241.

SHEARER, P. M. **Introduction to Seismology**. 3nd. ed. Cambridge, UK: Cambridge University Press, 2019. ISBN 9781316635742.

SHIOTANI, T. et al. Visualization of damage in rc bridge deck for bullet trains with ae tomography. In: SHEN, G.; WU, Z.; ZHANG, J. (Ed.). **Advances in Acoustic Emission Technology**. New York, NY: Springer New York, 2015. p. 357–368. ISBN 978-1-4939-1239-1.

SOUZA, J. F. de et al. **Simwave - A Finite Difference Simulator for Acoustic Waves Propagation**. 2022. Disponível em: <https://arxiv.org/abs/2201.05278>.

TARANTOLA, A. Inversion of seismic reflection data in the acoustic approximation. **Geophysics**, v. 49, n. 8, p. 1259–1266, 1984.

TARGINO, J. et al. A deep-learning inversion method for seismic velocity model building. In: EUROPEAN ASSOCIATION OF GEOSCIENTISTS & ENGINEERS. **First EAGE Conference on Machine Learning in Americas**. [S.l.], 2020. v. 2020, n. 1, p. 1–5.

VAILLANT, L. **SEG-EAGE salt data**. [S.l.], 2002.

VIRIEUX, J. et al. An introduction to full waveform inversion. In: _____. **Encyclopedia of Exploration Geophysics**. [S.l.: s.n.], 2017. p. R1–1–R1–40.

VIRIEUX, J.; OPERTO, S. An overview of full-waveform inversion in exploration geophysics. **Geophysics**, Society of Exploration Geophysicists, v. 74, n. 6, p. WCC1–WCC26, 2009.

WANG, W.; MA, J. Velocity model building in a crosswell acquisition geometry with image-trained artificial neural networks. **Geophysics**, Society of Exploration Geophysicists, v. 85, n. 2, p. U31–U46, 2020.

WANG, W.; YANG, F.; MA, J. Velocity model building with a modified fully convolutional network. In: SEG. **SEG International Exposition and Annual Meeting**. [S.l.], 2018. p. SEG–2018.

WILSON, D. K.; THOMSON, D. W. Acoustic tomographic monitoring of the atmospheric surface layer. **Journal of Atmospheric and Oceanic Technology**, v. 11, n. 3, p. 751–769, 1994.

WU, Y.; LIN, Y. Inversionnet: A real-time and accurate full waveform inversion with cnns and continuous crfs. **arXiv preprint arXiv:1811.07875**, 2018.

XIA, H. et al. The forward and inverse problem based on magneto-acoustic tomography with current injection. **Journal of Biomedical Science and Engineering**, Scientific Research Publishing, v. 10, n. 5, p. 97–105, 2017.

YANG, F.; MA, J. Deep-learning inversion: A next-generation seismic velocity model building method. **Geophysics**, Society of Exploration Geophysicists, v. 84, n. 4, p. R583–R599, 2019.

YANG, P. et al. A review on the systematic formulation of 3-d multiparameter full waveform inversion in viscoelastic medium. **Geophysical Journal International**, Oxford University Press, v. 207, n. 1, p. 129–149, 2016.

YING, C. et al. NAS-bench-101: Towards reproducible neural architecture search. In: CHAUDHURI, K.; SALAKHUTDINOV, R. (Ed.). **Proceedings of the 36th International Conference on Machine Learning**. [S.l.], 2019. (Proceedings of Machine Learning Research, v. 97), p. 7105 – 7114.

ZHAN, C. et al. Neural architecture search for inversion. **arXiv preprint arXiv:2201.01772**, 2022.

ZHANG, T.; ARAYA-POLO, M.; SHRIVASTAVA, A. Invmixer—an efficient deep neural network for seismic inversion. In: SEG. **SEG International Exposition and Annual Meeting**. [S.l.], 2023. p. SEG–2023.

ZHANG, Z.; LIN, Y. Data-driven seismic waveform inversion: A study on the robustness and generalization. **IEEE Transactions on Geoscience and Remote sensing**, IEEE, v. 58, n. 10, p. 6900–6913, 2020.

ZHU, M. et al. Fourier-deeponet: Fourier-enhanced deep operator networks for full waveform inversion with improved accuracy, generalizability, and robustness. **Computer Methods in Applied Mechanics and Engineering**, Elsevier, v. 416, p. 116300, 2023.