UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
CENTRO DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO

Deep Learning Applications in Geophysics: Data Matching and Inversion

Katerine de Jesus Rincon Perez

Supervisor: Prof. Dr. Samuel Xavier de Souza

Doctoral Thesis Proposal presented to the pos-graduate Program in Electrical and Computer Engineering at UFRN. (area of concentration: Computer Engineering) as part of the requirements for obtaining the title of Doctor of Science.

Número de ordem PPgEEC: D390
Natal, RN, August 26, 2025

Universidade Federal do Rio Grande do Norte - UFRN
Sistema de Bibliotecas - SISB
Catalogação de Publicação na Fonte. UFRN - Biblioteca Central Zila Mamede

Perez, Katerine de Jesus Rincon.
Deep Learning Applications in Geophysics: Data Matching and Inversion / Katerine de Jesus Rincon Perez. – 2025.
115 f.: il.

Thesis – (doctorate) – Universidade Federal do Rio Grande do Norte, Centro de Tecnologia, Programa de Pós-Graduação em Engenharia Elétrica e de Computação, Natal, 2025.
Orientação: Prof. Dr. Samuel Xavier de Souza.

1. Time lapse seismic – Thesis. 2. OBN – Thesis. 3. Deep Learning – Thesis. 4. Non-repeatability effects – Thesis. 5. Inversion – Thesis. 6. FWI – Thesis. I. Souza, Samuel Xavier de. II. Título.

RN/UF/BCZM CDU 004.8:550.3

Elaborado por Jackeline dos Santos Pinheiro da Silva Maia Cavalcanti – CRB-15/317

Deep Learning Applications in Geophysics: Data Matching and Inversion

Katerine de Jesus Rincon Perez

Thesis approved on August 26, 2025 by the examining board composed of the following members:

Prof. Dr. Samuel Xavier de Souza (Supervisor) . . . . . DCA/UFRN

Prof. Dr. Bruno Souza Carmo . . . . . USP

Prof. Dr. Hermes Senger . . . . . UFSCAR

Prof. Dr. Gilberto Corso . . . . . DBF/UFRN

Prof. Dr. João Medeiros de Araujo . . . . . DFTE/UFRN

Prof. Dr. Luis Affonso Henderson Guedes de Oliveira . . . . . DCA/UFRN

*To my beloved children, **Ángel** and **Joaquín**, you are my light, my inspiration, and the heartbeat behind my every effort. Your laughter fuels my joy, your curiosity sparks my creativity, and your resilience reminds me daily of the strength found in persistence. This journey is as much for me as it is because of you.*

Acknowledgments

To give thanks is to pause for a moment amid everything and recognize that nothing is achieved in solitude. Every step taken is imbued with the presence, support, and generosity of many. My gratitude is what will remain in memory and in my heart, recalling with tenderness those who sustained me, inspired me, and walked with me through both luminous days and shadowed ones.

I begin by thanking God for the light that guided my steps, for the silent strength that sustained me, and for lifting me up whenever I felt lost along the way. To my parents in Colombia, Alcira and Jairo, who, even from afar, were always present—steadfast and attentive—ready to listen whenever the longing grew heavy. To my brother, Juan, for always being proud of me.

To my beloved husband, Jaime, who was my emotional and academic support, my comfort, and my companion at every stage of this journey. No one ever said it would be easy, but it was your love, patience, and constant presence that made this path possible. I also thank my husband's family for their constant prayers and unwavering support.

To Prof. Samuel, for the opportunity to join this program and for believing in my potential from the very beginning.

To Prof. João, for his firm guidance, patience, constant scientific support, and dedication at every stage of this work. To him I owe special gratitude for accompanying my growth as a researcher so closely, for challenging me, and for pushing me beyond my own limits.

To my laboratory colleagues—Edwin, Danyelle, Daniel, and Ramon—and to the research group, for the exchange of ideas and the many hours of collaborative work.

To my friends outside academia, Laura Castro and Alberto Ruiz, for reminding me that life also requires balance, joy, and moments of rest.

To Shell Brazil, for sponsoring the project *New Computationally Scalable Methodologies for Target-Oriented 4D Seismic in Pre-Salt Reservoirs* at the Federal University of Rio Grande do Norte, which made this research possible.

Abstract

This work applies *Deep Learning (DL)* methodologies to solve geophysical problems, with a focus on Time lapse seismic, an essential technique for monitoring changes in reservoirs and supporting strategic management, using synthetic pre-salt seismic data acquired with *Ocean Bottom Nodes (OBN)*. The first approach focused on mitigating one of the main limitations of TL seismic: data non-repeatability between surveys, caused by various environmental factors and equipment positioning variations, which can mask real reservoir changes. To address this challenge, we proposed *Conditional Generative Adversarial Networks (cGANs)*. This methodology was developed to correct non-repeatability effects, improving data quality before applying *Double Difference Full Waveform Inversion (DDFWI)*.

As a second application, designed to estimate velocity variations and serve as a technique parallel to inversion, we constructed a synthetic dataset representing scenarios with different velocity anomalies. A *Convolutional Neural Network (CNN)* was trained using synthetic baseline and monitor seismograms generated from a pre-salt velocity model. The network received the time-lapse difference between baseline and monitor data as input, and produced the corresponding velocity model as output. To benchmark the approach, conventional Double-Difference Full Waveform Inversion (DDFWI) was also performed, providing a reference for computational cost, efficiency, and inversion quality. Results demonstrate that the CNN achieved accurate inversion performance while delivering significantly higher computational efficiency compared to the traditional DDFWI workflow.

Both scenarios were trained with 80% of the data and tested with the remaining 20%. The predictions were evaluated using quantitative metrics such as the mean squared error (MSE), the normalized root mean squared error (NRMS), and the structural similarity index (SSIM), indicating a good inference capacity of the model.

Keywords: Seismic time-lapse, OBN, Deep Learning, non-repeatability effects, inversion, FWI.

Resumo

A pesquisa apresentada nesta tese insere-se no crescente interesse da geofísica em aproveitar metodologias de *Deep Learning (DL)* em tarefas de exploração e monitoramento de reservatórios. Em particular, o trabalho concentra-se em dados sintéticos representativos de cenários do pré-sal brasileiro, adquiridos com a técnica *Ocean Bottom Nodes (OBN)* em sísmica *Time-lapse (TL)*, cuja finalidade é comparar aquisições sísmicas realizadas em diferentes momentos da vida produtiva de um campo, com o propósito de identificar mudanças relacionadas à evolução do reservatorio.

Esse enfoque permite inferir variações de pressão e saturação de hidrocarbonetos, fornecendo informação crítica para o gerenciamento da produção, a tomada de decisões econômicas e a mitigação de riscos em projetos. Em ambientes complexos como o pré-sal brasileiro, onde a heterogeneidade das camadas salinas introduz desafios significativos na aquisição e no processamento, dispor de ferramentas capazes de melhorar a qualidade dos dados e acelerar a inversão constitui uma necessidade tanto científica quanto industrial.

Um dos principais obstáculos enfrentados pela sísmica TL é a não repetibilidade das aquisições. Embora em teoria o monitoramento devesse registrar exclusivamente mudanças no reservatório, na prática surgem diferenças artificiais causadas por variações inevitáveis como: a posição de fontes, flutuações ambientais e ou ruído instrumental. Essas diferenças introduzem o chamado ruído 4D, que muitas vezes é da mesma ordem de magnitude que o sinal associado ao reservatório. Métodos tradicionais como o binning 4D, a equalização cruzada ou o processamento simultâneo pre-stack mitigan esse problema, mas com resultados parciais e frequentemente custosos em termos de tempo de computação e esforço de parametrização. Nesse contexto, o DL oferece uma alternativa inovadora, capaz de aprender diretamente dos dados e propor soluções adaptativas mais rápidas e, potencialmente, mais robustas.

A primeira linha metodológica explorada nesta tese focou em reduzir os efeitos da não repetibilidade. Para isso, implementou-se uma *Conditional Generative Adversarial Network (cGAN)*, inspirada na arquitetura pix2pix, treinada com pares de disparos *Monitor-Baseline* na parte rasa, onde somente os efeitos de não repetibilidade são importantes e não existe sinal vindo do reservatório. O objetivo da rede foi receber dados afetados pela não repetibilidade e devolver versões ajustadas que preservassem as estruturas geológicas fundamentais. O desenho do pré-processamento foi decisivo: em vez de normalizar globalmente todos os sismogramas, optou-se por dividir cada disparo em janelas sobrepostas e aplicar normalização local. Essa escolha permitiu preservar as dinâmicas relativas de amplitude em cada janela e favorecer o aprendizado de padrões coerentes na escala espacial de interesse. Essa estratégia, mostrou vantagens claras na estabilidade do treinamento e

na capacidade de generalização do modelo. Com a aplicação de validação cruzada *k-fold*, confirmou-se que o treinamento era consistente e que as perdas nos testes se reduziam em diferentes partições do conjunto de dados, demonstrando a generalização da rede na solução do problema.

Os resultados dessa primeira parte foram satisfactorios. A cGAN conseguiu melhorar a repetibilidade dos dados, gerando versões mais limpas e coerentes para aplicações posteriores de inversão. Embora tenha sido detectada uma tendência a superestimar amplitudes, o comportamento geral da rede foi satisfatório no que diz respeito à preservação das estruturas. Métricas como o *Normalized Root Mean Square error (NRMS)* mostraram reduções significativas nos desajustes entre dados baseline e monitor, e o índice *Structural Similarity Index (SSIM)* evidenciou que as saídas mantinham a coerência espacial dos eventos sísmicos. Isso demonstra que a aplicação de cGAN pode constituir um passo prévio de grande valor em fluxos de trabalho de sísmica TL, fornecendo dados corrigidos para posterior análise com métodos físicos de inversão.

A segunda linha metodológica abordou um desafio ainda maior: a inversão de modelos de velocidade diretamente a partir de diferenças sísmicas. Construiu-se um conjunto sintético representando diferentes cenários de anomalias de velocidade no reservatório, com modelos baseline e monitor projetados para simular condições de produção. A partir deles foram gerados sismogramas que, posteriormente, foram processados para obter a diferença TL. A arquitetura de rede implentada é uma *Convolutional Neural Network (CNN)* desenhada para transformar diferenças sísmicas em modelos de velocidade. O processo começa com a entrada de seções sísmicas (diferenças entre baseline e monitor), que passam por sucessivas camadas convolucionais e de pooling, responsáveis por extrair padrões locais e reduzir a dimensionalidade dos dados. Em seguida, a informação é achatada (flatten) e processada por camadas densas, que aprendem combinações globais dos atributos sísmicos. Por fim, a saída é reestruturada em um grid bidimensional, representando o modelo de velocidade reconstruído.

Para avaliar o desempenho dessa rede, estabeleceu-se um paralelo com a inversao fisica *Double-Difference Full Waveform Inversion (DDFWI)*, metodologia de referência que aplica ajustes iterativos baseados na física da propagação de ondas. Os resultados mostraram que a CNN foi capaz de reconstruir perfis de velocidade que preservavam as principais características na zona do reservatório. A rede capturou com notável fidelidade as variações relativas. Métricas quantitativas como o RMS e o índice SSIM confirmaram a qualidade das predições, enquanto os mapas de diferença destacaram que os erros mais significativos se concentravam em áreas de ruído residual, e não nas regiões críticas do reservatório.

Um aspecto particularmente relevante foi a comparação em termos de custo computacional. Enquanto a DDFWI requeria entre quatro e cinco horas de processamento, mesmo com paralelização em threads, a rede convolucional produzia predições em questão de minutos, uma redução de tempo importante. Esse resultado é notável, pois em contextos de monitoramento TL a rapidez na interpretação pode marcar a diferença entre decisões oportunas e atrasos custosos. Assim, a CNN apresenta-se como uma alternativa ideal para

cenários em que se priorizam resultados aproximados mas rápidos, ou em que já exista conhecimento prévio do reservatório que permita contextualizar os modelos gerados pela rede.

Em conjunto, os resultados da tese permitem extrair conclusões relevantes. Em primeiro lugar, as cGANs constituem uma ferramenta eficaz para melhorar a repetibilidade dos dados sísmicos TL, fornecendo um pré-processamento que aumenta a confiabilidade das análises posteriores. Em segundo lugar, as CNNs demonstram um desempenho ideal para a inversão sísmica, combinando precisão com uma eficiência computacional em comparação com os métodos tradicionais. Em terceiro lugar, a investigação ressalta a importância de conceber as metodologias de DL não como substitutos excludentes da física computacional, mas como complementos que, integrados em fluxos híbridos, podem maximizar as fortalezas de cada abordagem.

Dessa maneira, o trabalho estabelece uma base sólida para o uso mais amplo do DL na exploração e no monitoramento sísmico. Para além dos experimentos sintéticos apresentados, o desafio futuro consiste em aplicar essas metodologias a dados reais do pré-sal, onde a complexidade geológica e as limitações operacionais colocarão à prova a robustez dos metodos.

Palavras-chave: Sísmica time-lapse, OBN, Deep Learning, efeitos de não repetibilidade, inversao, FWI.

BIBLIOGRAPHIC PRODUCTION

Articles published in scientific journals

During the course of this Ph.D., I contributed to several scientific publications. The first work presented here is directly related to the research developed in this thesis, particularly in the application of deep learning methods for seismic inversion. Additionally, I participated as a co-author in other collaborative studies, primarily contributing to the execution of the experimental components.

1. Rincon, K., Araujo, R.C.F., Galvão, M.M., Xavier-de-Souza, S., de Araujo, J.M., Barros, T. and Corso, G. (2024) 'Single-shot time-lapse target-oriented velocity inversion using machine learning', *Applied Sciences*, 14(21), 10047.

2. Collazos Gonzalez, J.A., Rincon Perez, K. de J., Pinheiro, D., da Costa, C.A.N., Gebre, M.G., Corso, G.F., Barros, T., de Araujo, J.M. and Wang, Y. (2024) 'Seismic trace interpolation based on the principle of reciprocity using a conditional generative adversarial network (cGAN)', *Journal of Geophysics and Engineering*.

3. Collazos Gonzalez, J.A., Rincon Perez, K. de J., Barros, T., Corso, G.F. and de Araujo, J.M. (2024) 'Comparison of U-Net with conditional generative adversarial networks and cycle-consistent adversarial networks for real seismic data interpolation: Tupi Field', *Research, Society and Development*, 13(7), e1613746226. doi:10.33448/rsd-v13i7.46226.

Abstracts published in conferences

In addition to the work presented in this thesis, I contributed to other research efforts involving deep learning applications in geophysics. These contributions were presented in the form of extended abstracts or short papers at international conferences. Although the studies are not directly included in the main body of this thesis, they reflect my broader involvement in deep learning research and its integration into seismic data analysis.

1. Ferreira, P., Rincon, K., Araujo, R.C.F., Barros, T., Corso, G. and Lopez, J. (2025) 'AI-assisted 4D seismic inversion using very sparse OBN geometries', in *Proceedings of the First EAGE/SBGf Workshop on Marine Seismic Acquisition*.

2. Collazos, J., Rincon, K., Fagua, E., Medeiros, J., Kiyashchenko, D. and Lopez, J. (2025) 'Denoise 4D fast-track image using a conditional GAN: Brazilian Pre-salt example', in *SEG Image 2025 Conference Proceedings*. Society of Exploration Geophysicists.

3. Collazos Gonzalez, J.A., Rincon Perez, K. de J., da Costa, C.A.N., Pinheiro, D. and de Araujo, J.M. (2023) 'Using convolutional neural networks type GAN to build seismic 4D processing tools', paper presented at the *International Conference on Geoscience, High Performance and Cloud Computing (ICGHPC)*.

4. Collazos Gonzalez, J.A., da Costa, C.A.N., Pinheiro, D., Rincon Perez, K. de J., Gebre, M.G., de Araujo, J.M. and Lopez, J. (2023) 'Seismic data interpolation with an iterative workflow and generative adversarial networks', in *Proceedings of the 84th EAGE Annual Conference and Exhibition*. European Association of Geoscientists and Engineers.

5. Collazos Gonzalez, J.A., Rincon Perez, K. de J., Fagua, E., de Araujo, J.M. and Lopez, J. (2023) 'Predicting full-track from fast-track processed seismic data with GAN: A real data example', in *SEG Image 2023 Conference Proceedings*. Society of Exploration Geophysicists.

6. Araujo, R., Rincon, K., dos Santos, P., Corso, G., de Araujo, J.M., Xavier-de-Souza, S. and Barros, T. (2023) 'Data matching of time-lapse ocean bottom node seismic using convolutional neural networks for improved repeatability', in *Proceedings of the Brazilian Geophysical Society (SBGF) Conference*.

Submitted Articles

At the time of writing this thesis, I have also contributed to research articles that have been recently submitted for publication. These works reflect ongoing developments related to the themes explored in this dissertation, particularly in the application of deep learning to seismic data processing and interpretation. Although they are not included in the main chapters, they represent a continuation of the research trajectory initiated during this Ph.D.

1. Collazos Gonzalez, J.A., Rincon Perez, K. de J., Fagua, E., de Araujo, J.M. and Lopez, J. (submitted) 'Full-track from fast-track seismic processing with a Pix2Pix model in a Brazilian Pre-Salt field', *Geophysical Prospecting*.

Contents

List of Figures

List of Tables

List Abreviations

- **AI**: Artificial Intelligent.
- **ANN**: Artificial Neural Networks.
- **CDGANs** :Cyclic Discriminative Generative Adversarial Networks for Image-to-Image Transformation
- **cGAN**: Conditional Generative Adversarial Network.
- **CNN**: Convolutional Neural Network.
- **CycleGAN**: Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks.
- **D**: Discriminator.
- **DDFWI**: Double difference Full Waveform Inversion.
- **DL**: Deep Learning.
- **DLG**: Discriminator Learning Rate.
- **FWI**: Full Waveform Inversion.
- **G**: Generator.
- **GAN**: Generative Adversarial Network.
- **DLG**: Generator Learning Rate.
- **L1**: L1 Norm.
- **L2**: L2 Norm.
- **ML**: Machine Learning.
- **MLOps**: Machine Learning Optimization
- **MSE**: Mean Square Error.
- **OBN**: Ocean Botton Node.
- **pix2pix**: Image-to-Image Translation with Conditional Adversarial Network.
- **ReLU**: Rectified Linear Unit.
- **RNN**: Recurrent Neural Network.
- **RTM**: Reverse Time Migration.
- **SNR**: Signal-to-Noise Ratio.
- **Tanh**: Hyperbolic Tangent.
- **TL**: Time Lapse, 4D Seismic.
- **WGANs**: Wasserstein GANs

Glossary

To facilitate understanding of some of the concepts used in this document, a glossary was created emphasizing terms related to deep learning. These terms are presented alphabetically and explained theoretically or in the context of their application, depending on their relevance and use.

- **Adam**:
  It is an optimization algorithm that adjusts the learning rate of each weight individually, using moving averages of first and second-order gradients to correct the direction and magnitude of weight updates. Adam combines the advantages of Stochastic Gradient Descent (SGD) with the method of moments.(Kingma & Ba 2014)
- **Activation function**:
  The output of a layer is transformed to introduce non-linearity between successive layers. This procedure is necessary because each weighted sum of the inputs can result in values greater than 1, and the output must be expressed in probabilistic terms. Consequently, the activation function normalizes the output, restricting it to the range between 0 and 1. In some cases, certain activation functions normalize values between -1 and 1.
- **Batch** :
  The dataset (N-patches) is used for an update step during network training.
- **Binary Cross entropy**:
  The loss function measures the difference between a neural network's predictions and the expected results in binary classification problems. This function evaluates the model's performance, penalizing predictions that differ from actual values. During the training phase, binary cross-entropy must be minimized to increase the accuracy of model predictions. See equation 1

$$\text{Binary Cross Entropy} = -\frac{1}{N}\sum_{i=1}^{N} y_i \log(\hat{y}) + (1-y_i)\log(1-\hat{y}) \tag{1}$$

- **Channels / Features**:
  These are the characteristic properties or phenomena of the input data that are extracted in a layer. Channels and features refer to the same dimension in the data (for example, a geophysical image will be composed of 1 channel that contains amplitude information)
- **Convolution**:
  Convolution is a mathematical operator that transforms two functions, a and b, in

a third function, c. This operator is widely used in convolutional neural networks (CNNs) to extract relevant features from input data, such as images. The convolution process involves applying a filter (or kernel) to the input and generating a feature map.

- **Epoch**:
  The time the neural network needs to process all the training data once. During an epoch, the network adjusts its weights and biases based on errors calculated from predictions compared to actual values. This process is repeated multiple times over multiple epochs so that the network can learn effectively and improve its accuracy in the prediction task.
- **Kernel**
  A small matrix of weights, typically of size $k \times k$, where k is smaller than the input dimensions.
- **Learning rate**:
  The parameter that controls the step size in stochastic gradient descent determines how much the weights are adjusted relative to the loss gradient. An adequate learning rate is essential to ensure efficient model convergence, so a learning rate that is too high can cause the model to fail to converge or oscillate. At the same time, a learning rate that is too low can result in an excessively slow training process.
- **Loss / Cost function**
  The metric that evaluates the difference between the neural network's predictions and the expected results is known as. During the training phase, the goal is to minimize this loss function to improve the model's accuracy. The loss function quantifies the prediction error, and minimizing this error is essential for the network to learn to make more accurate predictions.
- **Mean Absolute Error (MAE)**:
  This metric calculates the average of the absolute discrepancies between predicted and actual values in the shallow part of the data. It provides a direct and intuitive way to assess the typical magnitude of errors in the data set (Willmott & Matsuura 2005). See equation 2

$$MAE = \frac{1}{N} \sum_{i=1}^{n} |y_i - \hat{y}_i| \tag{2}$$

- **Mean square error (MSE)**:
  This measure calculates the mean of the squared differences between predicted and actual values. By giving more weight to larger errors, MSE helps identify significant deviations between generated and actual pixels (Chai & Draxler 2014). See equation 3.

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{3}$$

- **Optimizer**
  The algorithm updates the weights and/or learning rate to reduce loss during the training phase.

- **Overfitting**
  When an algorithm overfits training data, the model tends to memorize specific correspondences between inputs and outputs rather than learning generalizable patterns. Consequently, the model exhibits poor generalization ability when exposed to new datasets not seen during the inference phase. This phenomenon, known as overfitting, compromises the model's effectiveness when applied to out-of-sample data.
- **Padding**
  Process of adding null pixels in or around the image to compensate for the size loss of the output image after convolution.
- **Peak Signal-to-Noise Ratio(PSNR)**:
  This metric evaluates the quality of pixel reconstruction by comparing generated pixels with real pixels, considering the maximum possible pixel value. A higher PSNR indicates greater similarity between the generated and real images.(Horé & Ziou 2010).See equation 4

$$PSNR = 10\log_{10}\left(\frac{MAX^2}{MSE}\right) \tag{4}$$

- **Pooling**
  The Pooling operation is similar to the convolution operation; however, the distinction lies in how the dot product is changed through an operation that reduces the feature map. Pooling applies a function, such as the average (Average Pooling) or the maximum value (MaxPooling), within a window K. A Pooling operation widely used to reduce the size of features and highlight the most representative values is MaxPooling with a $2x2$ kernel, which selects the maximum value within the kernel. This technique effectively reduces the dimensionality of feature maps, preserving the most important information and reducing the computational load in subsequent neural network layers.
- **Rectified Linear Unit (ReLU)**
  It is an activation function that activates neurons only for output values greater than 0 5. ReLU is widely used in deep learning because it helps mitigate the problem of gradient disappearance during the training phase. This problem occurs when gradients become extremely small, making it difficult to update the neural network's weights effectively. In turn, ReLU allows for more efficient gradient propagation, accelerating the convergence process and improving model performance.See equation 5.

$$ReLU(x) = max(x, 0) \tag{5}$$

- **Signal Noise Ratio (SNR)**:
  This metric evaluates the relationship between the desired signal's strength and the background noise's strength. A higher SNR suggests that the generated pixels more closely resemble the real pixels, with less noise interference (Huynh-Thu & Ghanbari 2008). See equation 6.

$$SNR_{dB} = 10\log_{10}\left(\frac{Psignal}{Pnoise}\right) \tag{6}$$

- **Stride)**:
  Stride refers to the number of pixels by which the filter (or kernel) moves across the input image or feature map during the convolution operation. A stride of 1 means the filter shifts one pixel at a time, resulting in high-resolution output with more detailed spatial information. When the stride is greater than 1, the filter skips pixels as it moves, which reduces the output dimensions and computational load but may also lose some finer details.
- **Structural similarity index measure (SSIM)**:
  This measure assesses perceived image quality by considering lighting, contrast, and structure changes. By examining pixel correlations in localized areas, SSIM provides a more robust assessment of the perceptual quality of generated images.(Wang, Bovik, Sheikh & Simoncelli 2004*b*). See equation 7.

$$\mathrm{SSIM(x,y)} = \frac{(2\mu_x\mu_y + C_1) + (2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{7}$$

  Components:

  - μ_x, μ_y: local means of images x and y. They represent the *luminance* (average brightness).
  - σ_x^2, σ_y^2: local variances, capturing the *contrast* of each image.
  - σ_{xy}: covariance between x and y, measuring how they vary together. This captures the shared *structure*.
  - C_1, C_2: small constants introduced for numerical stability, preventing division by values close to zero.

- **Training / Inference**
  The training phase is the phase in which a machine learning algorithm is built; inference uses this trained model to make a prediction.
- **Transposed convolution**
  The inverse convolution operation uses the same algorithm as convolution. Still, it varies the padding and stride parameters to obtain a new image I_{conv} from a matrix I, keeping the same size or larger size than the previous original entry.

Chapter 1

Introduction

Geophysical exploration using time-lapse (TL) seismic data acquisition enables the monitoring of changes in fluid properties within producing reservoirs, a capability that is crucial for business decision-making and risk mitigation, as emphasized in (Lumley 2001*a*). This need has driven increasing interest in computational methodologies based on Deep Learning (DL), which offer promising solutions for addressing complex geophysical challenges. In response, this Thesis explores two distinct DL applications within the context of TL seismic analysis. The first focuses on mitigating non-repeatability effects in TL acquisitions, enabling corrected data to be reliably integrated into workflows such as Full Waveform Inversion (FWI). The second investigates the direct use of DL for seismic inversion, assessing its performance relative to conventional FWI techniques.

The central objective of this research is to develop and evaluate DL methodologies that enhance the quality and interpretability of time-lapse seismic datasets. To this end, a conditional generative adversarial network (cGAN), inspired by the Pix2Pix architecture, is employed to transform non-repeatable seismic data into adjusted versions that preserve key structural features. Simultaneously, a Convolutional Neural Network (CNN) infers variations in velocity models from seismic differences between baseline and monitoring surveys. These approaches aim to provide efficient and accurate alternatives to traditional inversion methods, while also being applied to improve computational efficiency by significantly reducing processing time and resource consumption. The performance of the proposed models is assessed through both quantitative metrics, such as Mean Squared Error (MSE), Structural Similarity Index (SSIM), and qualitative evaluations to examine their generalization and applicability..

The methodological framework is structured into two computational workflows, each applied to carefully preprocessed synthetic Ocean Bottom Node (OBN) datasets. The first workflow employs the cGAN to correct baseline-monitor shot pairs, thereby reducing acquisition non-repeatability effects. The second uses the CNN to estimate subsurface velocity models directly from input seismic data. Both models are trained in a supervised manner with standardized inputs and optimized loss functions. Their effectiveness is evaluated through a comparative analysis with conventional seismic processing and inversion techniques, highlighting their respective advantages, limitations, and potential contributions to reservoir characterization in practical settings.

In recent years, various publications on seismic data processing and interpretation

have proposed different methodologies that incorporate Machine Learning (ML) into their workflows. However, most ML implementations in seismic data processing primarily aim to enhance computational efficiency and apply established ML techniques (Anjom, Vaccarino & Socco 2023). In specialized areas such as TL seismic and TL data processing, the workflow typically includes sequential steps such as 4D binning and simultaneous 4D pre-stack processing. The 4D binning (Lecerf & Besselievre 2018) step is designed to improve data repeatability. In contrast, the simultaneous 4D pre-stack processing step focuses on preserving amplitude variations critical for reservoir monitoring, used when the positions of the source and receiver between traces are minimal. The 4D binning step improves repeatability, while simultaneous 4D pre-stack processing uses cross-equalization to apply shared processing operators (Nguyen, Nam & Park 2015*a*); within the context of applying ML to TL seismic data analysis, studies like Waage, Bünz, Landrø, Plaza-Faverola & Waghorn (2019) analyze the effects of non-repeatability to increase resolution in 3D seismic data. Additionally, Jun & Cho (2021) focuses on creating training datasets to enhance TL seismic data processing repeatability using ML techniques. Other approaches aim to implement methodologies that adjust or eliminate non-repeatability effects; for Example,(Yuan, Zhang, Jia & Zhang 2019*a*) uses 4D cross-equalization with Temporal Convolutional networks (TCNs) to adjust non-repeatability, while Alali, Kazei, Sun & Alkhalifah (2022*a*) employs a Recurrent Neural Network (RNN) trained on data excluding the reservoir area to infer directly related reservoir data. Studies like (Alali, Kazei, Altaf, Zhang & Alkhalifah 2020) use Deep Learning DL to equalize synthetic data, improving repeatability (Lee, Won & Jun 2024)(Alali, Kazei, Sun, Smith, Nivlet, Bakulin & Alkhalifah 2020), and various methodologies are used to reduce 4D noise and enhance accuracy across different seismic surveys. Furthermore, studies like (Yuan, Zhang, Jia & Zhang 2019*b*) implement Fully Convolutional neural networks FCN to reduce computational costs and directly produce accurate velocity changes from 4D data.

Given the success of DL in addressing data matching challenges, particularly in mitigating non-repeatable acquisition effects and aligning time-lapse seismic data, it is natural to extend these methodologies to more complex tasks such as seismic inversion. In recent years, seismic inversion has advanced significantly through the integration of DL techniques, which have enabled the modeling of complex relationships between seismic data and subsurface properties. Unlike traditional approaches that rely heavily on physical models and computationally expensive iterative schemes like Full Waveform Inversion (FWI), DL based methods can learn nonlinear mappings directly from large datasets, whether simulated or real. This process leads to faster estimation of parameters such as velocity or acoustic impedance. In particular, convolutional neural networks (CNNs) have shown strong performance in inversion tasks, effectively capturing spatial and temporal features from seismic data (Wu, McMechan & Wang 2022) (Yang & Ma 2019). Recent developments have introduced architectures like U-Net and residual networks, which further enhance the resolution and robustness of inversion results (Plotnitskii, Ovcharenko, Kazei, Peter & Alkhalifah 2022). These techniques support the creation of direct inversion models that, once trained, can generalize to new scenarios and rapidly predict velocity distributions, even under challenging conditions such as noise and non-repeatable effects in 4D seismic acquisitions (Araya-Polo, Jennings, Adler & Dahlke 2018)

1.1 Thesis Structure

The document is structured into several main sections to address the proposed topic systematically. Firstly, Chapter 1 corresponds to the Introduction, where the general context and motivation of the Thesis are introduced. Chapter 2, Theoretical Fundamentals of Time-lapse Seismic, discusses some topics for DL, including convolutional neural networks (CNNs) and generative adversarial networks (GANs), as applied to geophysical problems. Chapter 3 describes the datamatching process, Chapter 4 includes the published paper related to the inversion approach developed during the Thesis, Chapter 5 describes the inversion process, applied in different scenarios and benchmarked against DDFWI. Finally, chapter 6 presents the main conclusions and broader discussions that follow from the results.

Chapter 2

Background Concepts

TL Seismic, FWI, and Deep Learning

This chapter presents the theoretical background related to geophysics and deep learning (DL), which forms the foundation of our work. We begin in Section 2.1 with a brief overview of the fundamental concepts of TL seismic, OBN acquisition, and the non-repeatability approach 2.1.1, and Full Waveform Inversion (FWI) 2.1.2. Section 2.2 introduces machine learning (ML), providing the conceptual basis for the methods applied later in this study. In Section 2.3, we present a concise synthesis of the convolutional neural networks (CNNs) addressed in this research. Section 2.4 explains Generative Adversarial Networks (GANs), detailing their architecture and operational principles.

2.1 Seismic and TL Seismic

The process of generating and recording seismic data to create an underground map is known as seismic acquisition. Seismic acquisition is one of the steps for seismic exploration, involving the controlled generation of elastic waves and the recording of their response to characterize subsurface properties (Yilmaz 2001*a*). This geophysical technique explores natural resources, particularly oil and gas. The method involves utilizing a source to produce seismic waves and employing receivers to detect the reflected waves. This approach enables the creation of detailed subsurface maps for resource identification. Figure 2.1 is an example of the acquisition process. As information is collected, processed, and analyzed to obtain information on the geological features below the surface, there are two types of acquisition: land seismic and marine seismic.

In this study, we will focus on the application of our methods to marine seismic, specifically to Ocean Bottom Node (OBN), which uses Nodes positioned on the bottom of the sea to obtain seismic data applied in TL seismic (TL), which consists of carrying out two different seismic acquisitions, acquired in the same position at intervals sufficient to monitor changes in reservoir fluids, often called *Baseline* (First acquisition or original acquisition) and *Monitor* (duplicated acquisition with a time interval). Figure 2.2 represents an example for *Baseline* and *Monitor* seismograms, which visually do not present differences but are evident when making the difference between *Baseline* and *Monitor*.

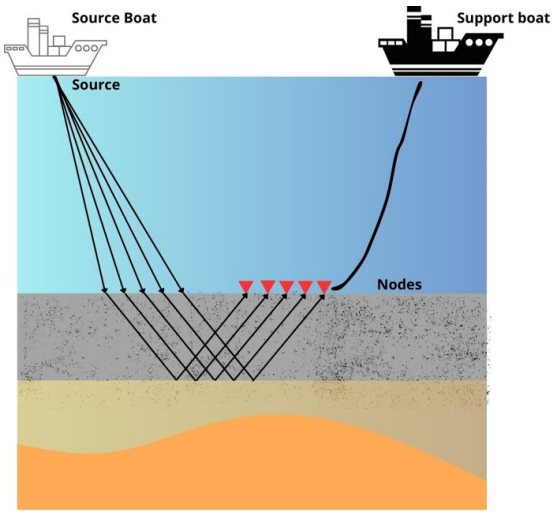

Figure 2.1: Image example of seismic acquisition OBN..

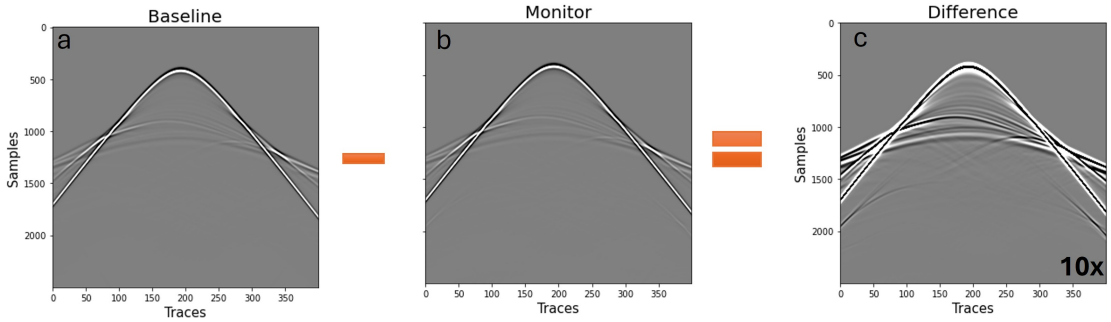

Figure 2.2: Example of two different seismograms,(a)the baseline seismogram,(b), both visually have no differences or non-repeatability effects,(c) represents the difference between baseline and a monitor amplified 10 times, in which we can see the effects of non-repeatability.

TL seismic (TL), or 4D seismic, monitors reservoir changes over time, providing information that can influence reservoir management decisions. However, one of the main challenges of this technique is the presence of non-repeatability effects in the data. These effects, resulting from environmental variations, equipment positioning shifts, and other external influences, can significantly hinder achieving a *match* between different acquisitions. This issue emerges because non-repeatable variations can hide or modify the actual changes within the reservoir over time. Consequently, adjusting or eliminating these effects from the data is essential to enable seismic imaging techniques that better reflect the actual reservoir changes, allowing for precise, business-impacting decisions.

2.1.1 Non-repeatability effects in 4D Seismic

As mentioned in the previous section, TL seismic plays a key role in decision-making for producing reservoirs, as experts rely on it to interpret residual differences that can

significantly affect outcomes (Kragh & Christie 2002). These residual differences can also be considered as TL noise or non-repeatability effects.

Non-repeatability effects are variations that occur between seismic surveys taken in the same area at different times. These variations can result from differences in shot positioning, water velocity, and water layer height, among other factors, leading to discrepancies in data when compared across surveys. Such variations may mask or distort the actual changes within the reservoir, making it challenging to interpret the data accurately. Consequently, non-repeatability reduces the reliability of TL seismic; therefore, it is necessary to align the data, mitigate these effects, and enhance the precision of reservoir monitoring analyses.

To quantify the similarity between TL seismic datasets, we used the Normalized Root Mean Square (NRMS) difference, a widely adopted metric for quantifying non-repeatability effects in 4D seismic acquisition. Following the approach described by Kragh & Christie (2002), the analysis computes the NRMS within a shallow window of the seismogram, where repeatability issues typically dominate due to acquisition and near-surface effects.. By restricting the analysis to this window, we aim to isolate acquisition-related discrepancies from subsurface production effects. A threshold NRMS value of 0.30, as suggested by Kragh & Christie (2002), is adopted as the upper limit for acceptable repeatability. This criterion allows us to determine whether a given baseline-monitor pair is suitable for further processing, such as data matching. For the inversion stage, however, we rely on a dataset that is free from non-repeatability effects to ensure that the predicted velocity changes are not biased by acquisition-related discrepancies.

$$NRMS = \frac{200 \times RMS(MONITOR - BASE)}{RMS(MONITOR) + RMS(BASE)} \tag{2.1}$$

2.1.2 Full Waveform Inversion (FWI)

Full Waveform Inversion (FWI) has emerged as one of the most advanced methodologies in seismic imaging for reconstructing high-resolution models of subsurface physical properties, such as acoustic wave velocity and density. This technique formulates the inversion as an optimization problem, where the objective is to minimize a cost function that quantifies the misfit between observed seismic data and synthetic data simulated via numerical solutions to the wave equation, typically through finite-difference or spectral methods. By exploiting the full content of the seismic waveform, including travel times, amplitudes, and phase information, FWI provides superior resolution compared to conventional inversion techniques.

However, due to the inherently nonlinear and ill-posed nature of the inverse problem, Full Waveform Inversion (FWI) is typically formulated as a local optimization problem. This formulation requires not only the evaluation of the objective (misfit) function but also the computation of its gradient concerning the model parameters, usually via the adjoint-state method (Fagua 2021). The high dimensionality of the parameter space and the presence of multiple local minima make the inversion highly sensitive to the choice of the initial model. Therefore, FWI often demands sophisticated regularization techniques and careful model parameterization to improve convergence stability and to mitigate the

risk of non-uniqueness in the recovered solution.

Given that Full Waveform Inversion (FWI) is a powerful tool, several methodologies have been developed to adapt it for TL seismic applications. When applied to monitor subtle differences between repeated acquisitions, often affected by non-repeatable errors and varying acquisition conditions, a more robust approach is required. In this context, the Double-Difference Full Waveform Inversion (DDFWI) workflow has proven particularly effective, as it significantly reduces the impact of common-mode errors by minimizing the misfit between differences in observed and simulated data from both baseline and monitor surveys. Studies such as (Asnaashari, Brossier, Garambois, Audebert, Thore & Virieux 2014) highlight the effectiveness of DDFWI in improving the resolution of temporal subsurface changes, even in the presence of noise or uncertainties in the seismic source. In our specific application, figure 2.3 highlights the stage at which the adjusted data are integrated into the DDFWI workflow..

The double-difference strategy offers enhanced accuracy in TL variations due to its focus on inverting the difference data. However, this approach requires a pre-processing step for datasets, such as TL binning, to ensure similar source-receiver locations between two surveys (Asnaashari et al. 2014). Due to the computational cost of performing an FWI iteration, an initial step to evaluate the data in terms of seismic image quality is necessary.Carrying out this process involves considerable effort and high computational cost, which motivates the use of a Deep Learning method to accelerate its execution.

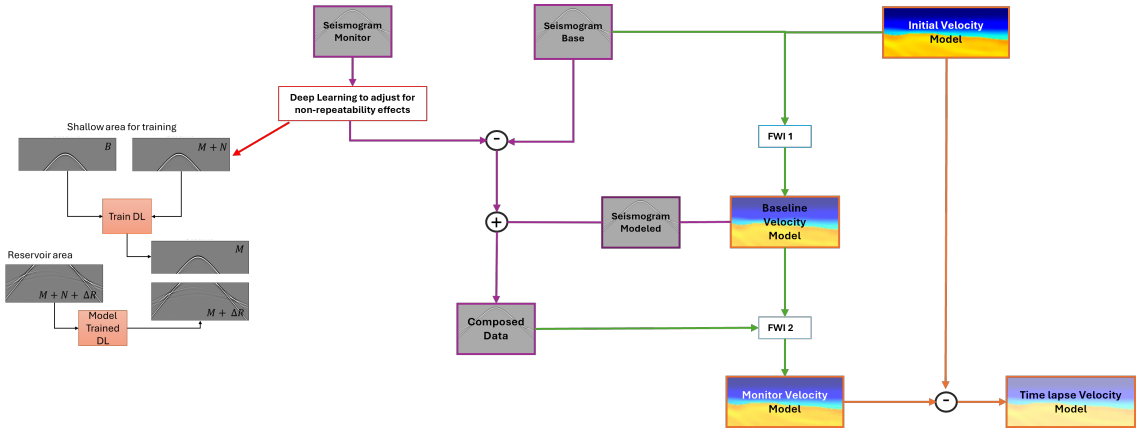

Figure 2.3: The figure illustrates the implementation of non-repeatability corrections with deep learning within a time-lapse inversion workflow. The monitor seismogram is adjusted with a trained model to mitigate these effects, and together with the baseline seismogram, it is used within the Double-Difference Full Waveform Inversion (DDFWI) framework to obtain time-lapse velocity models.

In this section, we talk about some fundamental concepts of geophysics relevant to this Thesis, including seismic acquisition, TL analysis, and Full Waveform Inversion (FWI). We have highlighted both the capabilities and the limitations of traditional physics-based methods in resolving subsurface properties and monitoring reservoir changes. In the following section, we shift our focus to the core concepts of Machine Learning (ML) and Deep Learning (DL), which have gained increasing attention as powerful alternatives to

conventional geophysics processes. These data-driven approaches can be integrated either as components within traditional workflows or as complete end-to-end solutions, offering new possibilities for enhancing the efficiency, robustness, and automation of seismic processing and inversion tasks.

2.2 Fundamentals of Deep Learning

Since its origins, Artificial Intelligence (AI) has aimed to endow machines with the ability to mimic human cognitive functions. One of the earliest contributions to the field was made by Arthur Samuel, who, in the 1950s, developed one of the first programs capable of learning autonomously, specifically a computer system that improved its ability to play checkers through experience. His pioneering work laid the foundation for what we now call Machine Learning (ML), which he famously defined as "the field of study that gives computers the ability to learn without being explicitly programmed" (Samuel 1959).On the other hand, in Computer Science, AI means the study of intelligent agents of any device that perceives its environment and takes actions that maximize its probability of successfully achieving its objectives (Shinde & Shah 2018). In this way, taking the line of algorithms, the ML concept is based on algorithms that can extract knowledge from sample data, with learning defined as performance improved through the repeated execution of a specific problem-solving task as a subfield of AI. Machine learning can also be viewed as a branch of applied statistics, using computer models to statistically approximate an unknown, often complex, function that correlates given inputs to outputs. Therefore, Deep Learning represents a subset of machine learning that uses multi-layered neural networks as the computational model (Bishop 2006). Figure 2.4 illustrates the interconnection between Artificial Intelligence, Machine Learning, and Deep Learning, with specific examples in geophysics.

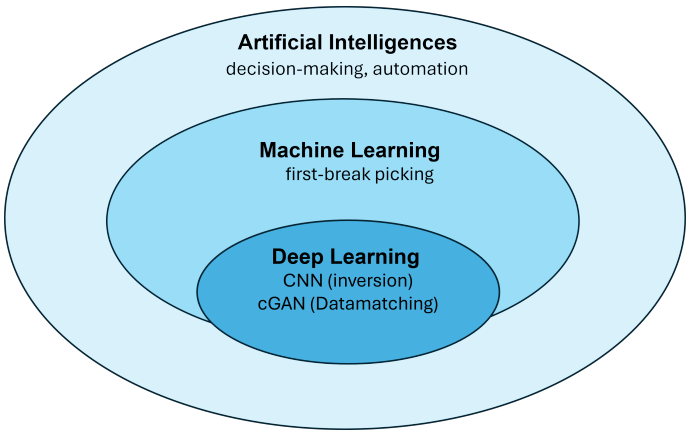

Figure 2.4: Conceptual hierarchy of Artificial Intelligence, Machine Learning, and Deep Learning, adapted from Goodfellow et al. (2016) and contextualized for geophysics applications. Deep Learning, as a subset of Machine Learning, enables the direct inversion of seismic data using neural networks such as CNNs, as well as data matching through cross-equalization techniques implemented with GANs.

DL models use Artificial Neural Networks (ANN), which are computational models inspired by the functioning of the human brain. Designed Models to recognize complex patterns and learn from data by tuning their internal parameters. The concept of ANN began with the work of Warren McCulloch and Walter Pitts in 1943, who proposed a mathematical model for artificial neurons (McCulloch & Pitts 1943). This work served as the basis for developing more advanced neural models, such as the perceptron introduced by Frank Rosenblatt in 1958, which is considered one of the first and most important ANN models (Rosenblatt 1958).

Thus, with countless layers and parameters, DL models process information numerically, where each layer applies mathematical operations to the input data, transforming parts of it into new representations. Essentially, DL incorporates a series of multiple layers containing nonlinear processing units to extract and transform features. Layers closer to the input data acquire simpler features; intermediate layers contain hidden data representations that are not visible, while those at higher levels learn more complex features derived from the outputs of lower layers. The hierarchical nature of DL models allows them to capture intricate patterns and representations within data. This layered approach allows deep neural networks to automatically learn and extract relevant features from raw input, reducing the need for manual feature engineering. As a result, deep Learning has achieved remarkable success in several domains, including geophysics, where it is increasingly applied in various research projects. For Example, multi-layered artificial neural networks are used to model complex patterns in large volumes of data. When applying the different networks, a careful definition of the approach is essential as this allows the optimization of parameters and the selection of the most appropriate architectures, ensuring that the model can extract relevant characteristics and provide accurate predictions. Figure 2.5 illustrates the process, starting with understanding the problem, followed by data analysis to define the DL algorithm for training and testing.

Figure 2.5: General workflow of a deep learning application, which begins with understanding the problem, identifying and preparing the data, selecting an appropriate DL algorithm, training the model, and testing its performance. This process outlines the typical pipeline followed in seismic and geophysical DL applications.

2.3 Convolutional Neural Networks (CNNs)

With the advancement of technology and the increase in computational capacity, more sophisticated architectures emerged, such as Convolutional Neural Networks (CNN), a

figure 2.6 representing the internal structure of a CNN proposed by (Lecun, Bottou, Bengio & Haffner 1998).CNNs process data with a grid structure, including images and, in this context, seismograms. They use Convolutional layers to extract local features from data, making them extremely useful for pattern recognition and classification tasks in large seismic data sets.

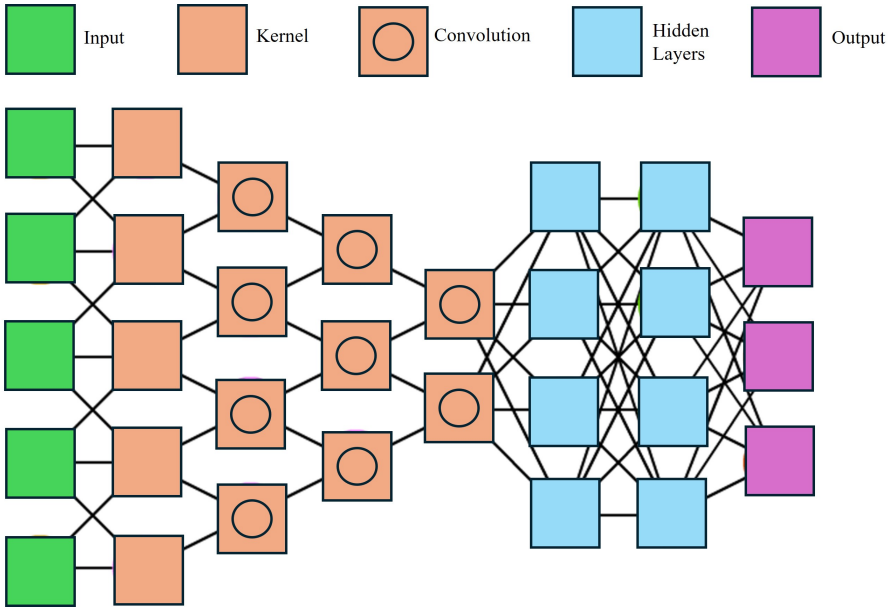

Figure 2.6: CNN structure, where the input receives the data, the kernel which is the filters run through the input data, the convolution operation combines the input data as the kernels to produce new images with features, then the hidden layers process the information generated by convolution, and finally the output provides the network prediction. Adapted from Van Veen (2016).

2.3.1 2D convolution operation

The convolutional operation is defined as:

$$g[i,j] = \sum_{u=-\infty}^{\infty} \sum_{v=-\infty}^{\infty} K[u,v], x[i-u, j-v] \tag{2.2}$$

where x is the input matrix, and u and v refer to the dimensions of the convolution kernel K. While this is a strict definition of convolution, the actual operation performed in convolutional neural networks more closely resembles a cross-correlation. Despite this, the convention is to refer to it as a convolution, and thus we adopt the term "convolution process." The algorithm below illustrates how the convolutions work.

Given an image I of size $N \times M$, the process involves computing a dot product between a kernel K (a matrix of size $n \times m$) and each corresponding region of the image I of the same size. The result of each dot product is stored in the central position of the region,

Algoritmo 1: Discrete convolution process for an image I of size $N \times M$ with a kernel K of size $n \times m$

for $r = 1$ to $M - m + 1$ **do**
  for $t = 1$ to $N - n + 1$ **do**
    $I_{\text{conv}}(r,t) = \sum_{i=1}^{n} \sum_{j=1}^{m} K(i,j) \cdot I(r+i-1, t+j-1)$
  end for
end for
Note: r and t represent the spatial positions in I

and the kernel is shifted across I to repeat the process until all valid positions are covered. The resulting output, I_{conv}, has dimensions $(N-n+1, M-m+1)$.

This convolution process is independently applied to each channel when dealing with multi-channel images. Figure 2.7 illustrates the convolution operation for a 5×5 image using a 3×3 kernel, showing how the kernel moves across the image to compute the values of the output matrix.

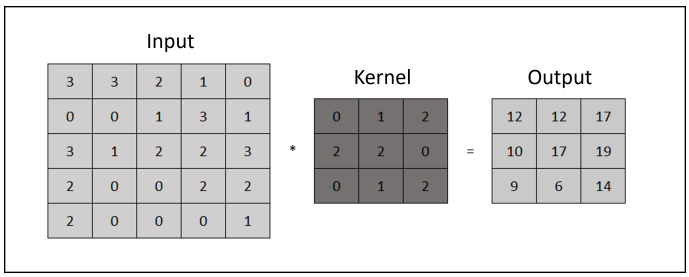

Figure 2.7: Convolution of a 5×5 matrix with a 3×3 kernel. The output is a 3×3 matrix.

Based on Algorithm 1, we can define several important concepts commonly used in convolutional networks that modify or extend this basic operation, including Pooling, Padding, and Stride, which influence how the convolution is applied and the resulting output size.

2.4 Generative Adversarial Networks (GANs)

Generative Adversarial Networks (GANs) constitute a class of deep machine learning algorithms designed for unsupervised Learning. Developed by Goodfellow, Pouget-Abadie, Mirza, Xu, Warde-Farley, Ozair, Courville & Bengio (2014), GANs comprise two neural networks: a generator and a discriminator, where the concept of adversariality is applied, the objective of which is to ensure that the classification made during training is the most accurate (Goodfellow et al. 2014). GANs involve a competitive process, analogous to a zero-sum game, in which the generator strives to produce as realistic data samples as possible. At the same time, the discriminator attempts to distinguish between authentic samples and those made by the generator. Since their inception, GANs have played a significant role in both industry and academia, based on results from scientific

databases such as (Scopus 2024), where, as of July 2025, more than 50,000 works have applied GANs in various areas. More than 5,000 works applied to geophysics, which is our focus of study, demonstrating that GANs have become a reliable tool for applying machine learning to geophysical problems. according to (Zhang & LeCun 2018).Advances in DL techniques, particularly Generative Adversarial Networks (GANs), have opened up new possibilities to mitigate these problems. GANs have stood out for their ability to generate realistic synthetic data, making them a promising tool for adjusting non-repeatability effects in seismic data.

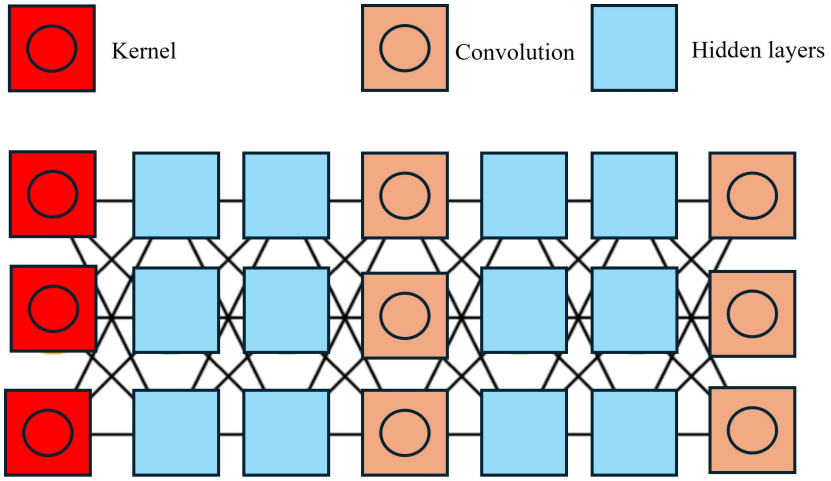

Figure 2.8: Internal Structure of GAN, Kernels (in red) are applied to the input images to extract relevant elements using convolution operations (in orange). Hidden layers (in blue) process and refine these elements, combining information from different image parts. Adapted from Van Veen (2016)

Different types of GAN algorithms are classified based on their application domains, such as image processing, computer vision, and sequential data (Shinde & Shah 2018). Among the various types of GANs applied to image processing, some interesting ones include Deep Convolutional GANs (DCGANs) (Radford, Metz & Chintala 2016*a*), which use convolutional layers to generate high-quality images; Wasserstein GANs (WGANs) (Arjovsky, Chintala & Bottou 2017), which improve training stability using the Wasserstein distance; and Conditional GANs (cGANs), which generate data conditioned on specific input information, among others (Mirza & Osindero 2014). For the application and development of adjustments in this research, we will use cGANs.

The architecture of a network GAN comprises two types, where the training process involves a generator and a discriminator network that compete against each other.

Generator (G)

Generates synthetic seismic data that is indistinguishable from real data. Generator G generates data $G(z)$ from a random input z, and D tries to identify whether $G(z)$ is false. D learns from the target data x to identify true label data and then tries to identify false G. As explained by (Goodfellow et al. 2014), the GAN model can be expressed as a function

minmax $V(G,D)$. The loss function, binary cross-entropy, is used as *minmax*. During training, first, D is trained to maximize $log(D(x))$ and at the same time, G is trained to minimize $log(1-D(G(z)))$.

$$\min_G \max_D V(G,D) = E_x[\log(D(x))] + E_z[\log(1-D(G(z)))] \tag{2.3}$$

Where E_x is the expected value from training, E_z is the expected value of all inputs to G.

Discriminator (D)

Distinguishing between real and generated seismic data. $D(x)$ is the discriminator estimate if x is real, and $D(G(z))$ is the forecast if the data generated from G is real.D is trained to maximize the probability of identifying the correct label of G. At the same time, model G is trained to minimize error.

2.4.1 Conditional Generative Adversarial Networks(cGAN)

Conditional GAN is a network GAN in which the generator is conditioned by an additional input (in this case, the original dataset) to produce corrected data.

cGANs perform well in scenarios where the output needs to be influenced by specific input parameters (Isola, Zhu, Zhou & Efros 2018). This feature is central to the development of our method. In geophysics, cGANs are especially valuable as they enable the generation of subsurface models conditioned on seismic or other geophysical data inputs. One of the main advantages of GANs over other generative algorithms is their ability to parallelize data generation, which significantly improves efficiency (Gui, Sun, Wen, Tao & Ye 2023). This capacity is essential for accurately simulating realistic images and, in turn, producing a trained model capable of learning and adjusting TL differences effectively. By using cGANs, we leverage their ability to generate realistic, conditioned outputs, enhancing the precision and reliability of our geophysical models.

Generator U-net

In the context of GAN, the U-Net architecture functions effectively as a generator due to its capability to capture and reconstruct detailed image features (Ronneberger, Fischer & Brox 2015). Originally designed for medical image segmentation, U-Net employs a U shaped structure that integrates encoding and decoding layers. This structure enables the network to learn both contextual information and local details. By combining learned features at different scales, U-Net generates high-quality images that maintain structural accuracy and pattern complexity.

PatchGAN Discriminator

Patch GAN is a discriminator network within GANs that focuses on evaluating images in terms of "patches" or local regions rather than assessing the entire image. In this setup, the Patch GAN discriminator (Isola et al. 2018) operates on small segments of the image, assessing whether each patch is real or generated. This approach allows the

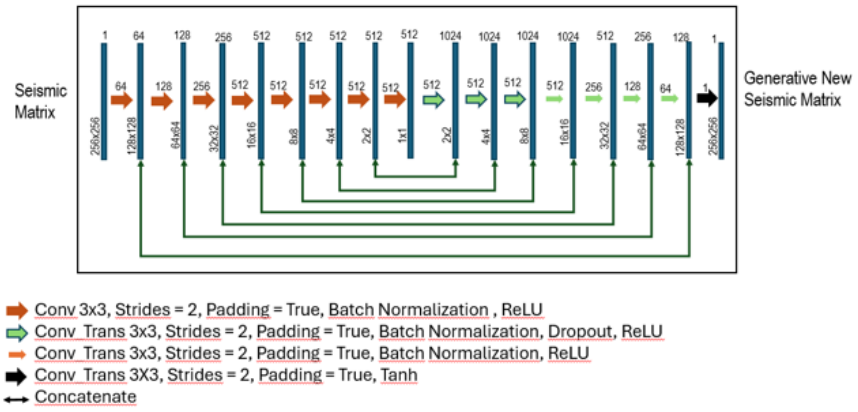

Figure 2.9: Schematic representation of the Pix2Pix U-Net generator architecture. The left path corresponds to the encoder, which progressively reduces the spatial resolution while increasing the number of feature channels, and the right path corresponds to the decoder, which reconstructs the output seismic matrix. Skip connections link encoder and decoder layers at matching scales, preserving structural details and improving reconstruction fidelity

discriminator to capture fine details and textures within the image, enhancing the quality of the generated images. Patch GAN facilitates faster model convergence and produces more realistic outcomes by concentrating on location and context at the patch level. This mainly benefits image synthesis, style transfer, and image restoration applications.

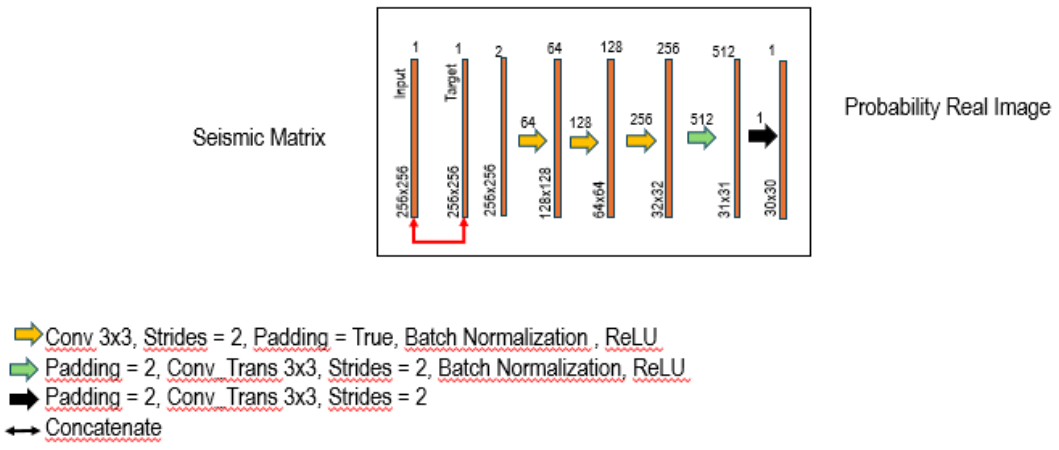

Figure 2.10: Diagram of the PatchGAN discriminator architecture. Instead of classifying the entire seismic section as real or fake, the discriminator evaluates overlapping patches, providing a matrix of authenticity scores. This patch-based strategy enforces high-frequency consistency and sharpness, allowing the generator to preserve local seismic textures while learning global structure.

This chapter presents key concepts from both geophysics and deep Learning to provide the theoretical foundation for the developments explored in the following chapters. These upcoming sections focus on the application of Deep Deep techniques to address two specific geophysical challenges: the data matching chapter 3 and the seismic inversion chapter 5. In particular, they demonstrate how deep Learning can be employed to reduce computational costs and enhance the efficiency of traditional geophysical workflows.

Chapter 3

Data Matching

In this chapter, we will detail the procedures for applying the Data Matching. First, we will address data collection and preprocessing. Next, we will discuss the architecture of the Deep Learning (DL) model, followed by a description of the model training phase. Figure 3.1 illustrates the described workflow.

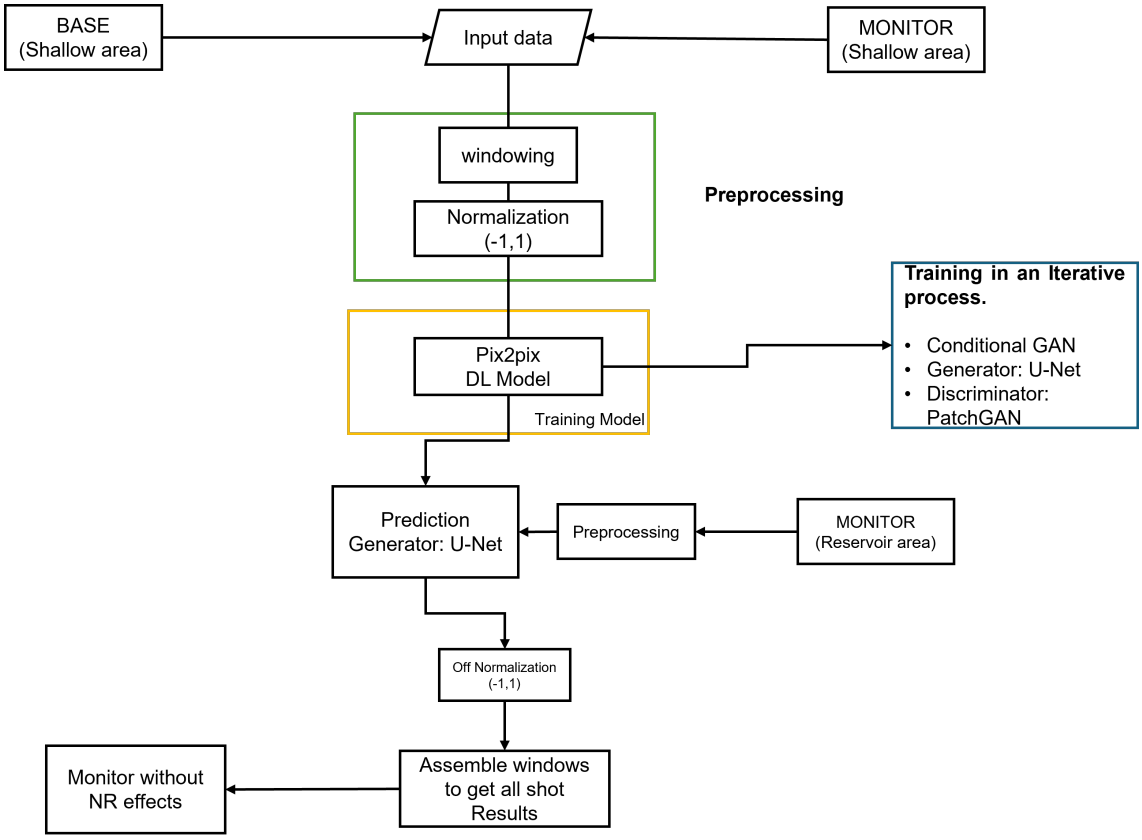

Figure 3.1: Methodology Workflow. The figure shows the developed flow, which is divided into three major areas: preprocessing, training, and prediction.

In geophysics, the concept of data matching (Lumley 2001*b*) refers to the process of adjusting seismic data from different baselines and monitor surveys to minimize non-geological differences caused by factors such as changes in acquisition geometry or other

environmental conditions (noise, pressure, temperature, etc.). These changes, called non-repeatability effects need to be adjusted. During the development of this research. A systematic methodology exhibited in figure 3.1 was designed to correct non-repeatability artifacts in TL seismic data by employing DL with a GAN-based global network. This methodology is based on the use of a *Monitor* acquisition made up of a set of fifty seismograms *Monitor* as data *input* and a *Baseline* acquisition made up of a set of fifty seismograms *Baseline* as data *target* for training, ensuring learning to effectively identify 4D noise and increasing isolation from monitoring changes in the reservoir.

To apply the methodology, each seismogram is divided into two distinct areas: the shallow area and the reservoir area. The shallow part of the Baseline acquisition and the Monitor acquisition are used to train the model. The model learns to identify variations in the monitor data (non-repeatability effects) from the baseline data (data without non-repeatability effects). The shallow portion of the Baseline is used as the target input, since it corresponds to the area that does not change. Figure 3.2 graphically illustrates the methodology's application.

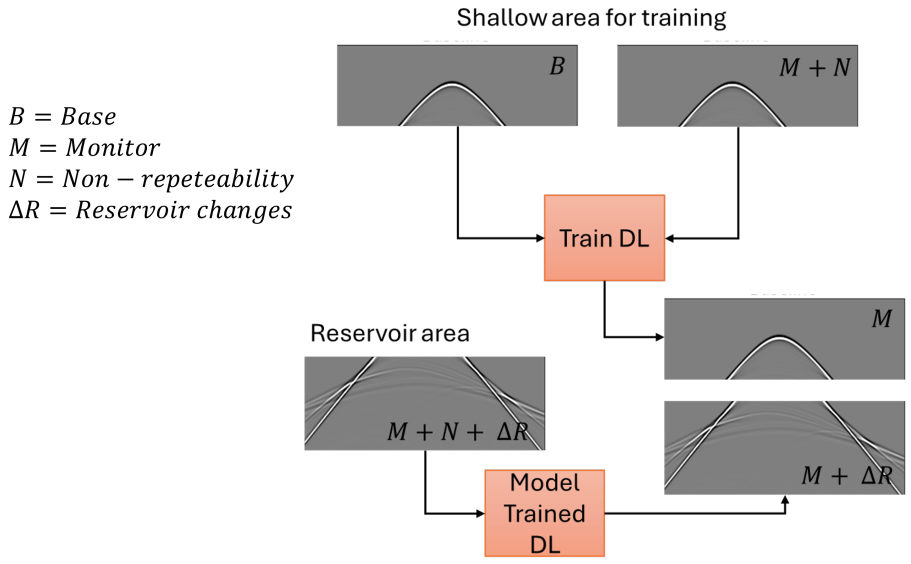

Figure 3.2: Graphical representation of the application of the methodology using a seismogram

In other words, in the proposed workflow, a shallow region is selected to provide training data for the model, focusing on the characterization of timelapse noise patterns.". Once the model becomes an expert at recognizing and aligning the 4D noise from the shallow area, it is then applied to the reservoir area. This application allows you to filter 4D noise effectively, improving the accuracy of monitoring changes within the reservoir and contributing to a more reliable and efficient geophysical interpretation.

Our study uses a single monitor acquisition to train and make predictions. Each seismogram is divided into two distinct areas. The first area, the shallow area, is used to train the model because it exhibits non-repeatability effects. The second area is the reservoir,

which not only has non-repeatability effects, but also changes within the reservoir itself, see figure 3.2.

The goal is to enhance the shallow area to train the model on 4D noise patterns. Once the model is adept at recognizing and removing 4D noise from the shallow area, it is then applied to the reservoir area to filter 4D noise effectively, thereby increasing the accuracy of monitoring changes within the reservoir.

3.1 Data Modeling

A synthetic time-lapse (TL) seismic database was generated using a representative velocity model of the Brazilian pre-salt (López, Neto, Cabrera, Cooke, Grandi & Roehl 2020). The *baseline* survey was simulated as a 2D acoustic OBN (ocean-bottom node) acquisition, with sources spaced every 50 m and receivers (OBN) distributed within a 400 m range. The OBN gathers were computed by propagating a 5 Hz Ricker wavelet (Gholamy & Kreinovich 2014) for 10 s, with a temporal sampling interval of 4 ms. The computational grid employed a uniform discretization of $\Delta x = 25$ m and $\Delta z = 25$ m. Figure 3.3 illustrates an example of the velocity model used to construct the database. To simulate the monitor survey, a Gaussian perturbation corresponding to 3% of the baseline velocity model was introduced in a localized region (red rectangle in Figure 3.3a). The perturbation extended 300 m vertically and 1500 m laterally, mimicking a significant fluid-front displacement near the injection well. In addition, the monitor scenario included a 20 m/s increase in water velocity and a 100 m thick water layer modification. As a result of data modeling, we obtained seismograms composed of 400 traces and 2,500 samples.

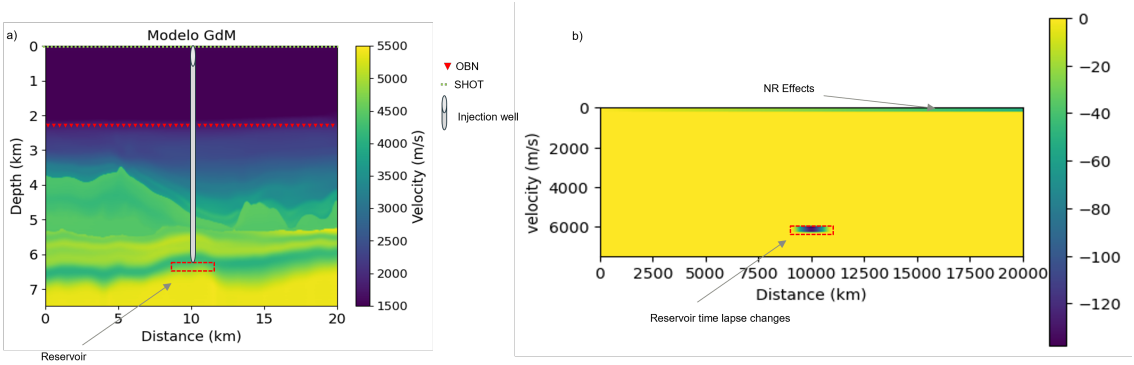

Figure 3.3: Velocity models used in the synthetic time-lapse experiment. (a) Initial velocity model employed to generate the baseline dataset and, after including non-repeatability effects, the monitor dataset. The model shows the acquisition geometry with OBN receivers and shot positions, as well as the reservoir location (red dashed line). (b) Difference model obtained by subtracting the baseline velocity model (without non-repeatability effects and without reservoir changes) from the monitor velocity model (with non-repeatability effects and reservoir perturbations).

We choose the data for the experiments based on the NRMS value criteria described in Chapter 2. According to the computed NRMS values, the TL variation did not exceed

10% relative to the Baseline, ensuring the consistency and quality of the input data for cross-equalization analysis (Johnston 2013).

Figure 3.4 illustrates an Example of the data used for all experiments. In total, 50 baseline shot gathers and 50 monitor shot gathers were used. The monitor gathers contain both non-repeatability effects and TL changes within the reservoir area.

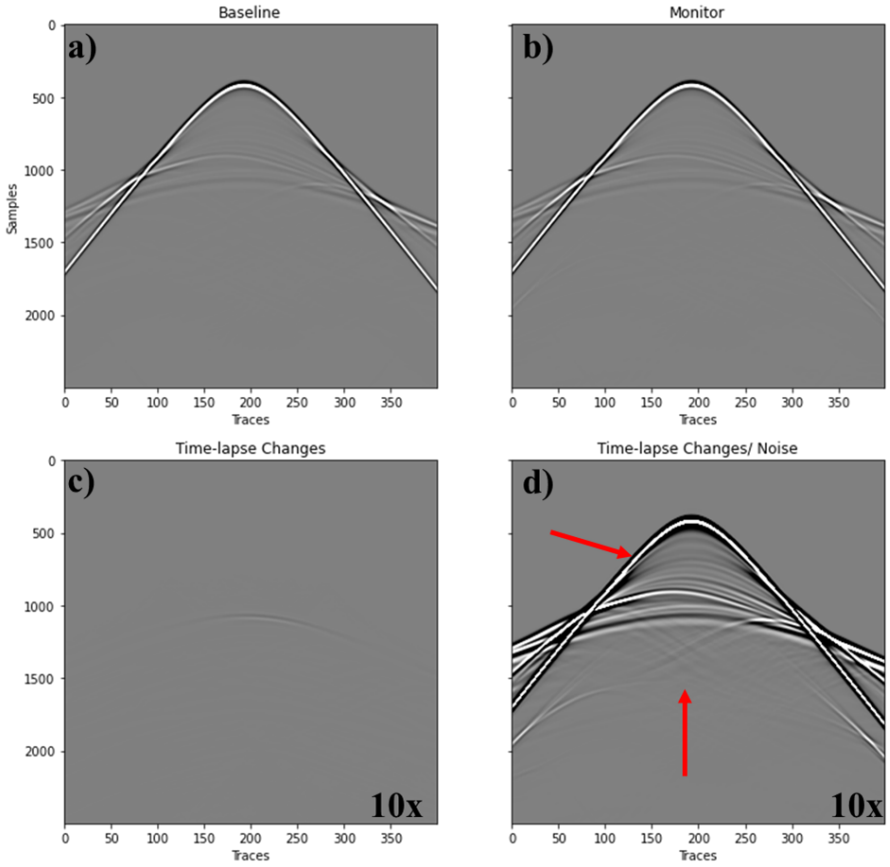

Figure 3.4: Shot gather base line(a), shot gather monitor (b), TL changes in the reservoir area, (c) TL changes in the reservoir area include and non-repeatability effects (d). An Example of data used for the experiments

3.2 Data Pre-processing

3.2.1 Windowing

Seismograms are composed of 400 traces and 2500 samples. In this context, windowing refers to dividing the seismogram into 256×256 pixel segments for input into the neural network. This segmentation facilitates data management and processing, enabling the DL model to handle more tractable portions of the seismogram. Throughout the experiments, we concluded that maintaining the default Pix2Pix window size was the most effective strategy to achieve reliable results without increasing training time.

When testing smaller dimensions—always defined as powers of two—we observed that reducing the window size consistently increased memory consumption and made data handling more complex. The chosen 256×256 segments are sufficiently large to capture the main characteristics and 4D noise patterns present in the shallow monitor data, yet compact enough for the GAN to effectively learn and generalize these patterns. This balance between detail and generalization supports the training of a robust model capable of accurately representing the features of interest (Goodfellow et al. 2014), (Radford, Metz & Chintala 2016*b*).

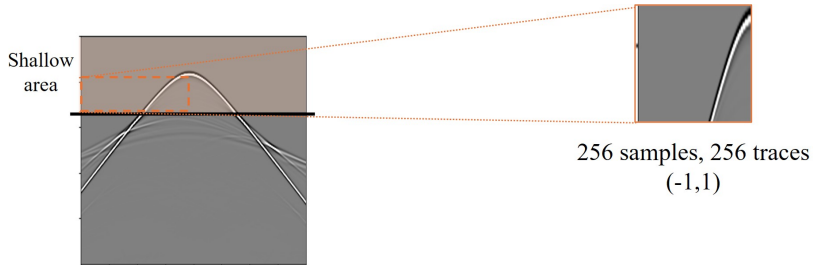

Figure 3.5: illustrates the process of creating windows in the shallow area of the seismogram. Each window has a size of 256 × 256 samples and was modeled to capture localized variations. Given that the original seismogram consists of 400 traces and 2500 time samples, the extracted windows appear rectangular.

3.2.2 Data normalization

Data normalization is essential in DL methods, especially in convolutional neural networks (CNNs).By normalizing data, we ensure that the inputs to the network are on a similar scale, preventing large amplitude variations from disproportionately influencing the network's weight adjustments. This step helps accelerate training convergence and enhances the model's overall performance. (Krizhevsky, Sutskever & Hinton 2012),(Ioffe & Szegedy 2015).

The pseudocode presented in algorithm 3 applies a global normalization to the entire seismic Shot gather. This procedure computes the maximum absolute amplitude across all traces, whether positive or negative, and scales the whole gather by this value. As a result, the amplitudes are constrained within the interval $[-1, 1]$, which helps stabilize the training of neural networks. However, this approach neglects local amplitude variations that may carry geophysical significance, particularly in 4D seismic analysis, where subtle amplitude differences can reflect changes in fluid saturation or pressure over time.

In contrast, the algorithm 2 function performs a local normalization based on root-mean-square (RMS) energy calculated within fixed-length time windows along each trace. The regional energy is interpolated over time and further smoothed using a Gaussian filter to minimize abrupt transitions. This smoothed energy is then inverted and used as a scaling factor, attenuating high-energy regions while preserving relative amplitude patterns in low-energy areas. Finally, the entire normalization matrix is scaled by the

maximum absolute amplitude of the normalized gather to ensure a consistent amplitude range across different inputs. This method preserves local amplitude relationships and enhances signal clarity.

Figure 3.6 shows an Example of the initial predicted seismograms produced by the network under two normalization scenarios: one applied before windowing and the other after windowing (see Section 3.2.1). The differences highlight the impact of the normalization stage on the quality and consistency of the predicted results.

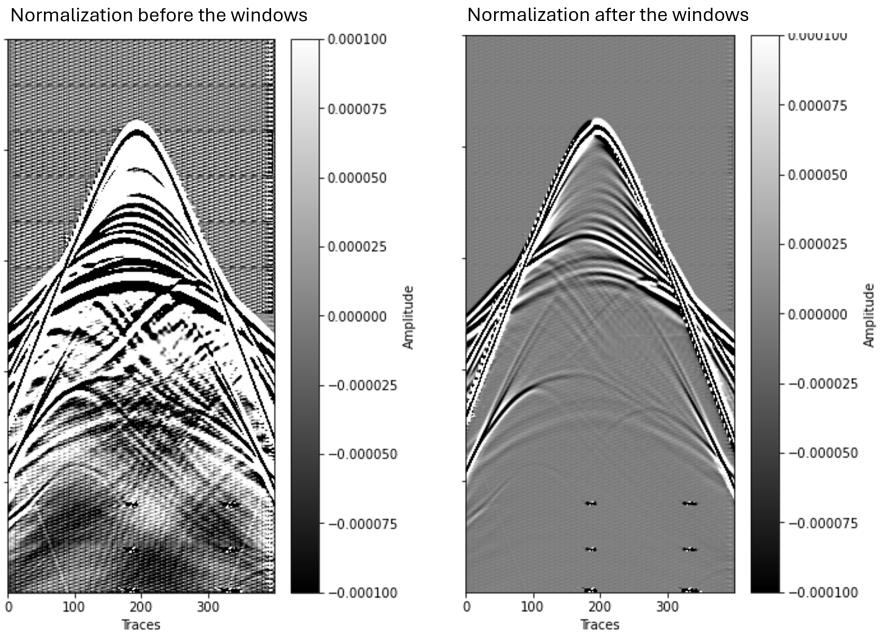

Figure 3.6: Comparison of amplitude normalization strategies applied to a seismic shot gather. **(a)** Global normalization applied to the entire gather using a single scaling factor. This method suppresses local amplitude variations and may obscure geophysically relevant details. **(b)** Local RMS-based normalization applied after windowing.

We distinguish the two approaches based on the scale at which normalization is applied. While `Amplitude Normalization for Windows` uses a single global factor for the entire gather, `Normalize Seismic Shot` adapts the scaling dynamically over time and per trace. This adaptability is especially valuable for DL tasks involving architectures such as pix2pix or CNNs, which rely on learning local spatial and temporal patterns. By maintaining meaningful amplitude relationships at finer scales, window-based normalization improves the model's ability to detect subtle signals and increases its generalization capacity, an essential advantage in applications such as data matching and seismic inversion in 4D settings.

3.2.3 Data segmentation

For training data, we used windows created from the shallow part of both the monitor and the Baseline seismogram. From the total number of windows generated, 80% were

Algoritmo 2: Amplitude Normalization for windows

Require: Shot gather S, window length op, smoothing parameter ops
Ensure: Normalized amplitude array `norma`

$nt, ntraces \leftarrow$ shape of S
Initialize `norma` as zeros of same shape as S
$nwin \leftarrow nt \div op$
Initialize `temp` as zero array of length $nwin$
$cenwin \leftarrow [i \cdot op + op//2 \text{ for } i \in [0, nwin)]$
$timeinter \leftarrow [0, 1, \ldots, nt-1]$
for $j = 0$ to $ntraces - 1$ **do**
  for $i = 0$ to $nwin - 1$ **do**
    $rms \leftarrow \text{mean}(|S[i \cdot op : (i+1) \cdot op, j]|^2)$
    if $rms == 0$ **then**
      $\texttt{temp}[i] \leftarrow 0$
    else
      $\texttt{temp}[i] \leftarrow \sqrt{rms}$
    end if
  end for
  $\texttt{norma}[:, j] \leftarrow \text{interpolate}(\texttt{timeinter}, \texttt{cenwin}, \texttt{temp})$
end for
Apply Gaussian filter to `norma` with standard deviation ops
for $j = 0$ to $ntraces - 1$ **do**
  for $i = 0$ to $nt - 1$ **do**
    if $\texttt{norma}[i, j] == 0$ **then**
      $\texttt{norma}[i, j] \leftarrow 0$
    else
      $\texttt{norma}[i, j] \leftarrow 1/\texttt{norma}[i, j]$
    end if
  end for
end for
$max\_amp \leftarrow \max(|\texttt{norma} \cdot S|)$
$min\_amp \leftarrow \min(|\texttt{norma} \cdot S|)$
$maxi \leftarrow \max(max\_amp, min\_amp)$
$\texttt{norma} \leftarrow \texttt{norma}/maxi$
return `norma`

allocated for training and 20% for validation. This division allows the GAN to effectively learn 4D noise patterns and generalize this learning, ensuring robust performance when applied to the reservoir. For testing, we used monitor data within the reservoir, providing a representative and challenging scenario for model validation.

The selection of the DL model for this investigation was based on these networks' abilities to classify data and attenuate noise and artifacts. Models using CNNs and GANs have shown substantial efficacy in these areas. Research by (Goodfellow et al. 2014)

Algoritmo 3: Normalize Seismic Shot

Input : Shot gather matrix *shot*
Output: Normalized shot $shot_{norm}$, and scaling factor *maxi*

$max\_amp \leftarrow |\max(shot)|$
$min\_amp \leftarrow |\min(shot)|$
$maxi \leftarrow \max(max\_amp, min\_amp)$
if $maxi \neq 0$ **then**
    $shot_{norm} \leftarrow shot / maxi$
    Convert $shot_{norm}$ to `float32`
end
else
    $shot_{norm} \leftarrow shot$
    Convert $shot_{norm}$ to `float32`
end
return $shot_{norm}, maxi$

demonstrated that GANs are proficient in generating new data from complex distributions. The U-Net, a CNN architecture, preserves salient details while eliminating noise, thus improving the quality of processed seismic data. Furthermore, studies by (Kaur, Pham & Fomel 2020) and (Gonzalez, Da Costa, Pinheiro, Rincon, Gebre, de Araújo & Lopez 2023) applied GANs for seismic data interpolation, noting significant improvements in data quality, supporting the suitability of GANs for addressing non-repeatability challenges in TL seismic data. Additionally, the adaptability of DL models allows for modifications across various data types and specific requirements, optimizing performance in diverse geological scenarios.

3.3 Hyperparameter Tuning

To achieve stable and accurate performance in the data matching task, we conducted a thorough hyperparameter tuning process. Due to the inherently unstable and competitive dynamics of Generative Adversarial Networks (GANs), hyperparameter selection plays a crucial role in convergence and output quality. Unlike standard convolutional networks, GANs require a delicate balance between the generator and discriminator losses to prevent issues such as mode collapse or vanishing gradients.

We initially adopted a manual tuning strategy, see figure 3.7, focusing on key hyperparameters such as the reconstruction loss weight (**lambda**), convolutional **kernel size**, and **learning rate**. This approach involved running controlled experiments while analyzing the evolution of generator and discriminator loss curves. For each training session, we carefully monitored how changes in these parameters affected convergence, stability, and output realism. This strategy follows recent practices in the field, where metaheuristic approaches such as Variable Neighborhood Search (VNS) (Pinheiro, Gonzalez, Corso,

Gebre, da Costa, Xavier-de Souza & Barros 2024) and Genetic Algorithms (GA) (Alarsan & Younes 2020) have been explored to optimize GAN performance in seismic tasks.

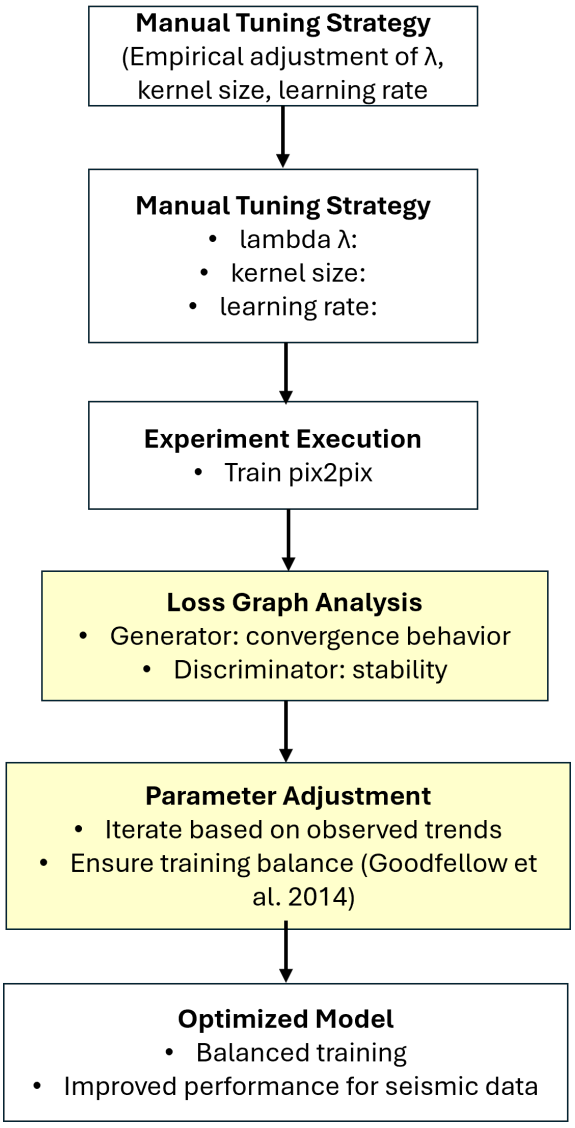

Figure 3.7: Manual hyperparameter tuning workflow adopted in this study. The process involved iterative experimentation and training curve analysis for selecting the optimal learning rate, kernel size, and loss weighting parameter λ.

The λ parameter, which balances the adversarial loss and the L_1 reconstruction loss in the generator objective, was varied across several orders of magnitude (10^4, 5×10^4, 10^5, and 10^6). The goal was to determine how this balance affects the realism and amplitude fidelity of the predicted gathers. While the normalized root mean square (NRMS) error remained stable across configurations (≈ 1.15–1.17), both the mean absolute error (MAE) and signal-to-noise ratio (SNR) improved significantly when $\lambda \geq 10^5$. The MAE dropped from 0.0034 to 0.0003, and SNR increased from 4.0 dB to nearly 5.0 dB. These results confirm that higher λ values help preserve subtle amplitude differences, which are

essential in seismic interpretation.

These outcomes, derived from the initial manual hyperparameter tuning phase, provided valuable insights into the influence of λ on key performance metrics. In particular, they highlighted that larger λ values enhanced MAE and SNR without compromising NRMS or SSIM. However, It is important to emphasize that these values stem from preliminary tests. In the final experiments shown in the Results section, we recalculated all metrics using improved configurations and a complete validation protocol, thereby ensuring a more rigorous and representative assessment of the model's performance.

To complement manual experimentation, Figure 3.8 presents a parallel coordinates plot generated using the Weights & Biases platform to visualize the impact of multiple hyperparameter combinations on the final model performance. The parameters analyzed include the learning rates for both the discriminator (DLR) and generator (GLR), the regularization parameter λ, and the convolutional kernel size. The color scale represents the final cost value for each experiment, where darker lines (deep purple) indicate better performance (lower cost) and lighter lines (yellow) indicate poorer results.

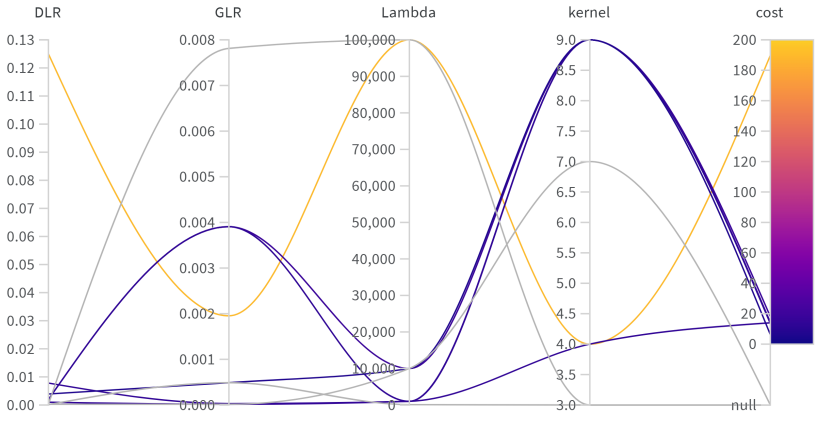

Figure 3.8: Parallel coordinates plot generated with Weights & Biases to compare the effect of different hyperparameter configurations on model performance. Each line represents one training run, and the color indicates the final cost, with darker colors corresponding to better performance. Lower learning rates (DLR and GLR), higher λ values, and larger kernel sizes are associated with lower cost values, highlighting the importance of proper hyperparameter tuning in achieving stable and accurate data matching.

From this plot, it becomes evident that configurations with lower learning rates (DLR and GLR), larger λ values (around 100,000), and larger kernel sizes (8 or 9) tend to achieve significantly better results. In contrast, higher learning rates and smaller kernel sizes correlate with increased cost values. This analysis confirms that proper tuning of λ and kernel size plays a critical role in model performance and that overaggressive learning rates can hinder convergence. Consequently, the best-performing configurations strike a balance between stable training and sufficient model complexity.

Based on both manual analysis and the use of Weights & Biases, we observed considerable differences related to the kernel size. Although the configuration with a kernel

size of 9 achieved slightly better performance in terms of accuracy and output fidelity, it resulted in a dramatically larger number of trainable parameters, over 330 million, compared to just under 30 million with a kernel size of 3. This tenfold increase imposes a substantial computational burden, requiring significantly more memory, training time, and hardware resources. In contrast, using a kernel size of 3 provides a much more efficient trade-off between performance and computational cost. The model remains effective while being more scalable and accessible for practical applications, particularly when dealing with large-scale seismic data or limited computational infrastructure. Therefore, a kernel size of 3 was selected as the optimal configuration to ensure a balance between predictive accuracy and computational feasibility..

Equation 3.1 provides the general formula for computing the number of trainable parameters in a 2D convolutional layer. Here, K represents the kernel size, C_{in} is the number of input channels, and C_{out} is the number of output channels (filters). The additional C_{out} term accounts for the biases associated with each filter.

$$\text{Parameters} = (K \times K \times C_{\text{in}}) \times C_{\text{out}} + C_{\text{out}} \tag{3.1}$$

The number of parameters was calculated for each convolutional layer in the encoder and decoder of the generator, as well as in the discriminator layers, for both scenarios: using a kernel size of 3 and a kernel size of 9. Table 3.1 summarizes the result of calculating the total parameters using the equation 3.1, and Figure 3.9 summarizes the number of parameters computed based on the two kernel sizes.

Kernel size	**Generador**	**Discriminador**	**Total**
3	28.26M	1.55M	**29.8M**
9	318.87M	13.99M	**332.86M**

Table 3.1: Number of parameters per kernel size setting

To further illustrate the computational implications of kernel size selection, we compared the total number of trainable parameters in the pix2pix-based cGAN for two configurations: using a kernel size of 3 and a kernel size of 9. While a larger kernel can increase the receptive field and potentially improve performance, it also drastically inflates the model complexity. This comparison highlights the trade-off between model expressiveness and computational feasibility.

In summary, the final hyperparameter configuration selected for use in our experiments consists of a kernel size of three and a loss weighting factor of $\lambda = 100{,}000$. This combination yielded the best compromise between accuracy, structural fidelity, and computational cost. Consequently, it was adopted for the remainder of the data matching experiments.

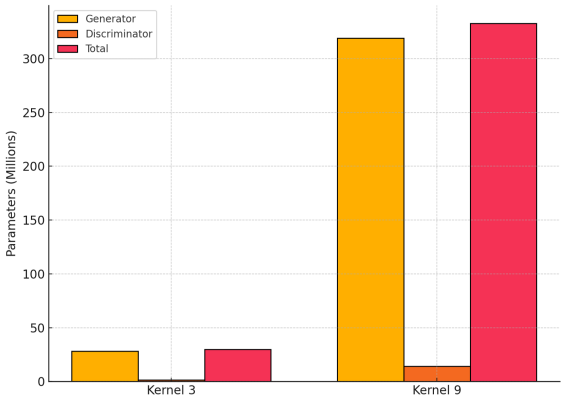

Figure 3.9: Comparison of trainable parameters in the generator, discriminator, and overall pix2pix cGAN model for kernel sizes of 3 and 9. The total parameter count increases more than tenfold when using a kernel size of 9, making it significantly more computationally expensive. Despite slight improvements in accuracy, the kernel size of 3 provides a better balance between efficiency and model performance.

3.4 Training and Validation Model

3.4.1 Training Process

A Conditional Generative Adversarial Network (cGAN) based on the pix2pix architecture was employed to adjust non-repeatability (NR) effects in TL seismic data. In this framework, the input to the network is the shallow section of the monitor gather, which contains both meaningful geophysical signals and NR-induced noise. The corresponding target is the shallow section of the Baseline gather, assumed to be free from NR distortions. These shallow regions were chosen for training because NR effects are typically more pronounced near the surface and can distort seismic interpretation.

The cGAN model consists of two neural networks: a generator $\boldsymbol{G}$ and a discriminator $\boldsymbol{D}$. The generator is implemented as a U-Net convolutional network, which receives 256×256-pixel windows from the monitor data and attempts to transform them into clean, baseline-like images. The discriminator, designed as a PatchGAN, evaluates the generator's outputs and classifies them as either real (Baseline) or fake (generated). This adversarial setup creates a dynamic competition: the generator learns to produce realistic seismic images, while the discriminator aims to distinguish them from real ones (Goodfellow et al. 2014).

Figure 3.10 illustrates the structure of the cGAN and its role within our workflow. During training, model $\boldsymbol{D}$ seeks to maximize classification accuracy, while model $\boldsymbol{G}$ is optimized to minimize reconstruction error and generate outputs that deceive $\boldsymbol{D}$.

This adversarial training process forms a zero-sum game where each network is optimized with opposing objectives. The generator minimizes a composite loss combining a pixel-wise L_1 reconstruction term and an adversarial term. The discriminator is trained with a binary cross-entropy loss to improve classification. As training progresses, the generator becomes increasingly capable of removing NR effects while preserving the un-

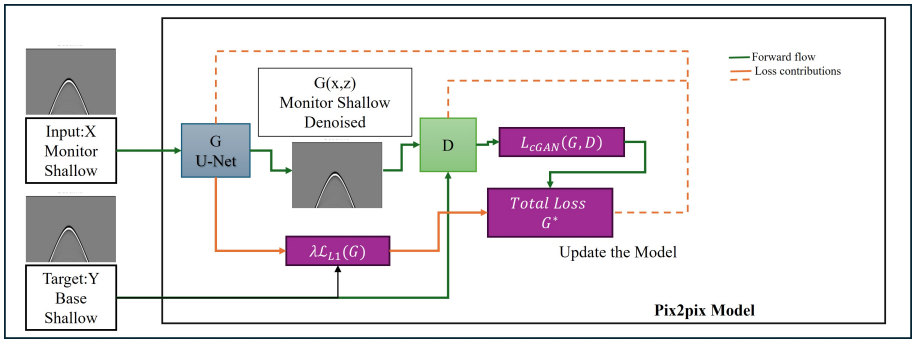

Figure 3.10: Illustrative diagram of how the pix2pix model is applied in our method.

derlying 4D changes related to reservoir dynamics.

To ensure training stability, we applied techniques such as batch normalization and label smoothing. Batch normalization standardizes feature distributions across mini-batches, which accelerates convergence and improves generalization. Label smoothing prevents the discriminator from becoming overconfident, fostering a more stable and balanced adversarial learning environment (Salimans, Goodfellow, Zaremba, Cheung, Radford & Chen 2016).

Training Configuration for Data Matching

Training was performed using the Adam optimizer with a learning rate of 2×10^{-6}, $\beta_1 = 0.5$, and $\beta_2 = 0.999$—common hyperparameters for stable GAN training. Models were trained for up to 50000 epochs with a batch size of 1.

Each input-output pair consisted of normalized seismic windows extracted from the monitor and baseline gathers. As detailed in Section 3.2.2, normalization was applied after windowing to preserve local amplitude variations and ensure consistent input statistics.

To address border artifacts caused by patch-wise prediction and reconstruction, a ramp function was introduced at the final stage of the data pipeline as detailed in section 3.5. This function applied a smooth amplitude taper with 50-pixel overlaps in both horizontal and vertical directions. This tapering ensured seamless merging of overlapping windows and improved continuity and realism in the reconstructed gathers, particularly at patch boundaries.

The generator's loss function included both adversarial and L_1 components:

$$\mathcal{L}_{\text{cGAN}}(G,D) = E_{x,y}[\log D(x,y)] + E_x[\log(1 - D(x,G(x)))] \tag{3.2}$$

$$\mathcal{L}_{L_1}(G) = E_{x,y}[||y - G(x)||_1] \tag{3.3}$$

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{cGAN}} + \lambda \mathcal{L}_{L_1} \tag{3.4}$$

3.4.2 Cross Validation

To evaluate the generalization performance of the pix2pix model, we employed a cross-validation approach. Due to the synthetic nature of the dataset, we followed a random split strategy, dividing the full dataset into training and testing subsets with an 80/20 ratio. This ensures that the model is trained on a diverse set of data while preserving a portion for independent evaluation.

We used k-fold cross-validation with $k = 5$ to reduce the risk of overfitting and assess the robustness of the model across different data partitions. For each fold, the model was trained on 80% of the data and validated on the remaining 20%, rotating the subsets across the folds. To see more in detail, the following algorithm 4 explains the implementation.

Cross-validation results confirmed the stability of the network's learning process, as performance variations across folds remained minimal. Table 3.2 summarizes the cross-validation experiments. These results further validate the model's ability to generalize to unseen input data, an essential requirement for DL applications.

Algoritmo 4: Pseudocode of the 5-fold

Data: Dataset of input seismic images (X) and target baseline images (Y)
Parameters: Number of folds $k = 5$, number of training steps, batch size, loss function parameters

```
begin
    Initialize KFold with k folds and shuffle;
    foreach fold i in 1 to k do
        Split data into training set (X_train, Y_train) and validation set (X_val, Y_val);
        Build TensorFlow datasets from (X_train, Y_train) and (X_val, Y_val);
        foreach training step do
            Perform one training iteration of the pix2pix model;
            Every n steps, compute validation loss and save to file;
        end
        Save the final model checkpoint and loss history for fold i;
    end
    Aggregate results: compute mean and standard deviation of validation loss
     across all folds;
end
```

Experiment	λ	Skip	Epochs / Steps
Set 1	100	1	50,000
Set 2	20	1	50,000
Set 3	20	10	50,000
Set 4	10,000	1	50,000
Set 5	100,000	1	50,000

Table 3.2: Hyperparameter settings used for each cross-validation experiment.

The following results correspond to different cross-validation tests performed using various hyperparameter settings, including different values for the regularization parameter λ and the skip connection configuration. These experiments were conducted to analyze the impact of these hyperparameters on the model's generalization ability and overall performance on the seismic data matching process.

Figure 3.11 shows the results from the first set of cross-validation experiments, performed with a regularization parameter of $\lambda = 100$, and training for 50,000 epochs. This configuration served as a baseline to evaluate the model's learning behavior and its ability to generalize under these conditions.

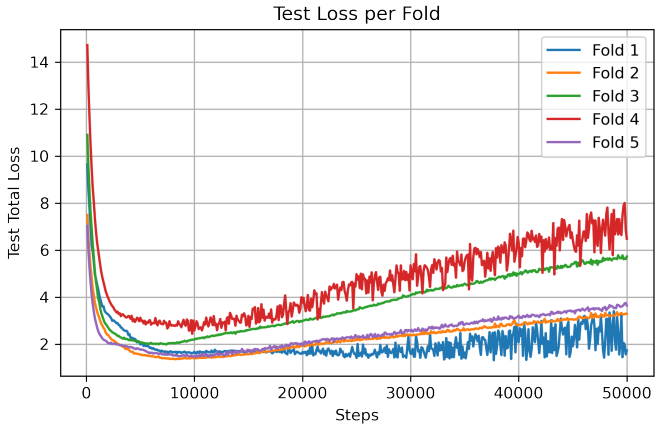

Figure 3.11: Test loss evolution across the five folds for the first cross-validation experiment, performed with $\lambda = 100$ and 50,000 epochs. The curves illustrate the learning dynamics and generalization behavior of the model under this baseline configuration

.

For $\lambda = 100$, the model exhibited fast initial convergence across all folds. However, signs of overfitting emerged in later stages, particularly in Folds 1 and 4, where the test loss began to increase significantly. This behavior suggests that the chosen regularization weight was insufficient to prevent the adversarial component from dominating, leading to reduced generalization in some validation subsets. The results highlight the sensitivity of the model to fold partitioning and motivate further tuning of λ and early stopping criteria.

The figure 3.12 displays the total test loss for each fold during cross-validation as a function of training steps. The curves show that while there is an initial decrease in test loss — indicating some level of learning — this trend quickly reverses. All five folds exhibit a progressive and sustained increase in test loss beyond approximately 5,000 to 10,000 steps. This consistent upward trend across all folds indicates a lack of generalization and the onset of overfitting, where the model begins to memorize the training data rather than learning meaningful patterns. Notably, none of the folds reaches a stable or low loss region, and no convergence behavior is observed. Fold 4, in particular, shows a significantly steeper increase in loss, suggesting higher sensitivity to overfitting or more complex data characteristics in that partition.

A $\lambda = 20$ remains too low to balance the adversarial objective adequately. The skip of

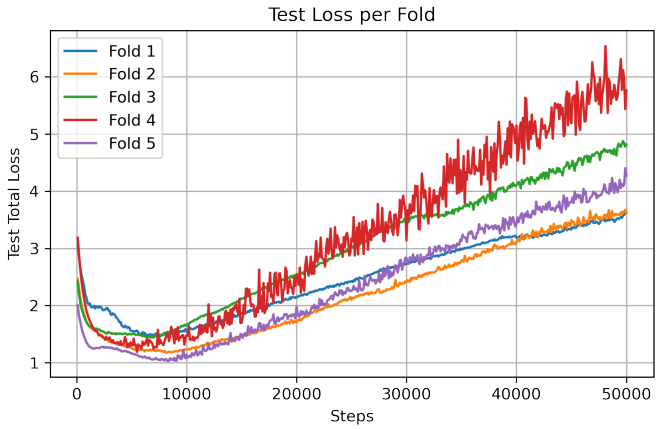

Figure 3.12: Test loss evolution across the five folds for the cross-validation experiment performed with $\lambda = 20$ and skip connection enabled (skip $= 1$). The curves show an initial reduction in test loss during early training stages, followed by a progressive increase, particularly in folds 3, 4, and 5, suggesting overfitting under this configuration. The results indicate the sensitivity of the training dynamics to the chosen λ value and skip connection settings.

10 probably further degraded the reconstruction capacity, eliminating part of the spatial coherence that L_1 intends to preserve. This process produces a model that tends to generate "realistic" but inconsistent adversarial structures compared to the target, which results in an increasing test loss.

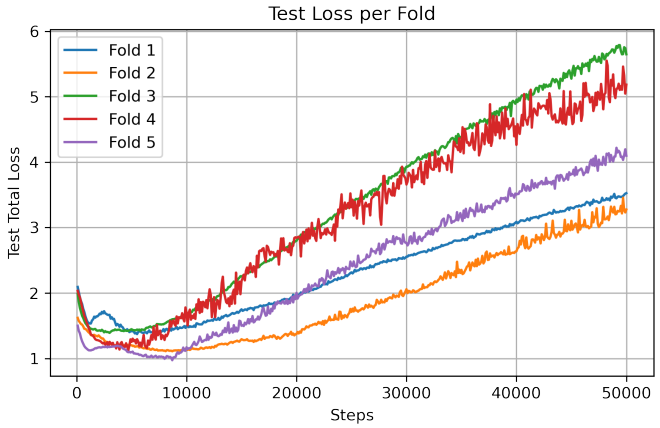

Figure 3.13: Test loss curves for each fold in the cross-validation experiment using $\lambda = 20$ and a skip value of 10. Although all folds show initial convergence within the first 5,000 steps, a consistent increase is observed in test loss across all folds, particularly in Folds 3 and 4. This behavior indicates severe overfitting and highlights the limitations of using a low λ combined with a high skip value, which may hinder the model's ability to reconstruct structurally consistent outputs.

In the setting of $\lambda = 20$ and skip = 10, the model shows consistent signs of overfitting across all folds. Figure 3.13. The shallow regularization and the reduction of low-level feature propagation due to the large skip distance likely contributed to poor reconstruction quality and loss instability. The overall performance deteriorates significantly over time, indicating that this configuration is not suitable for robust cross-equalization learning.

The 3.14 results of the 5-fold cross-validation provide strong evidence of the generalization and stability of the model. The consistent convergence of test loss across all folds suggests that the proposed pix2pix architecture can learn the underlying mapping required for effectively adjusting nonrepeatability effects in TL OBN seismic data, even when exposed to different subsets of the dataset. The low variability in the minimum test loss values between folds indicates that the model is not overly sensitive to the particular training and validation splits, further confirming its robustness. These findings support the applicability of the model to real-world scenarios where unseen variations in seismic data may occur.

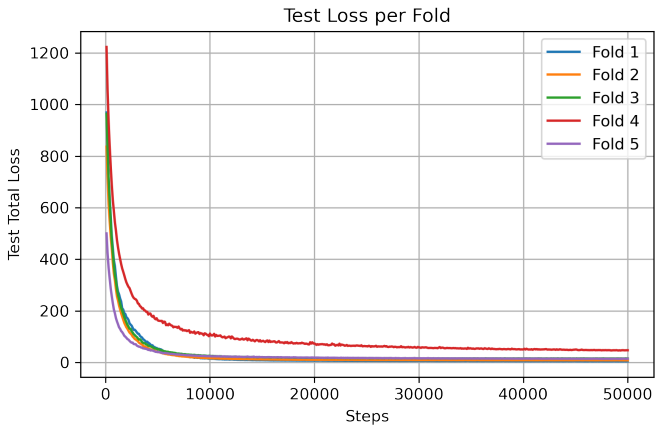

Figure 3.14: Test loss evolution across the five folds for the cross-validation experiment performed with $\lambda = 10{,}000$. The curves show consistent convergence across all folds, with the test loss steadily decreasing and stabilizing without evidence of overfitting, indicating that this configuration promotes robust generalization of the model on unseen data.

Based on the analysis presented in the hyperparameter tuning section, the best-performing was obtained using this configuration a regularization parameter $\lambda = 10^5$, a kernel size of 3, and a learning rate of 2×10^{-6}. We use this combination of hyperparameters to perform the final cross-validation test of the model.

Figure 3.15 presents the evolution of the total test loss across all five folds of the cross-validation experiment using the selected hyperparameters ($\lambda = 10^5$, kernel size = 3, and learning rate = 2×10^{-6}). All curves exhibit a steep decrease in loss during the initial training phase (0–10,000 steps), followed by a smooth convergence toward minimal values as training progresses. The low variance between folds indicates strong consistency and stable generalization across different data splits. This behavior confirms that the model is not only learning effectively but is also generalizing well to unseen data. Overall, the cross-validation results validate the selected configuration as robust and re-

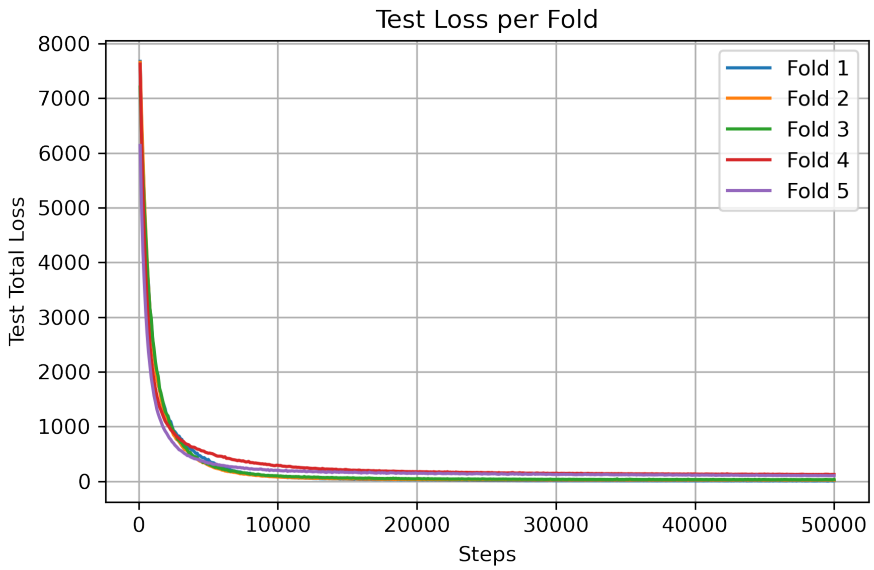

Figure 3.15: Test loss evolution across the five folds of the cross-validation experiment performed with $\lambda = 100{,}000$, selected based on the hyperparameter tuning results. The curves exhibit consistent convergence, with the test loss decreasing steadily and stabilizing across all folds. This behavior suggests robust generalization without signs of overfitting for this regularization setting.

liable, making it suitable for deployment in further analysis and seismic interpretation tasks.

3.5 Assembling from predicted windows

As the final step in the data matching methodology, for assembling windows, a ramp function was implemented to apply a gradual amplitude taper along the edges of each seismic window. This step reduces edge effects that can arise from abrupt amplitude transitions at window boundaries. To smoothly merge overlapping windows during reconstruction, side-specific ramp functions were defined, applying gradual weights depending on the horizontal position x within each window.

The ramp functions for the left ($ramp_L$) and right ($ramp_R$) sides are defined as follows:

$$ramp_L(x) = \begin{cases} 1 & \text{if } x < 206 \\ \frac{1}{50}(256 - x) & \text{if } x \geq 206 \end{cases} \tag{3.5}$$

$$ramp_R(x) = \begin{cases} \frac{1}{50}(x) & \text{if } x < 50 \\ 1 & \text{if } x \geq 50 \end{cases} \tag{3.6}$$

Here, x denotes the horizontal index within the window. The function $ramp_L(x)$ applies a decreasing weight toward the right edge, while $ramp_R(x)$ applies an increasing

weight toward the left edge. Use the ramp function to ensure a smooth transition when overlapping adjacent windows.

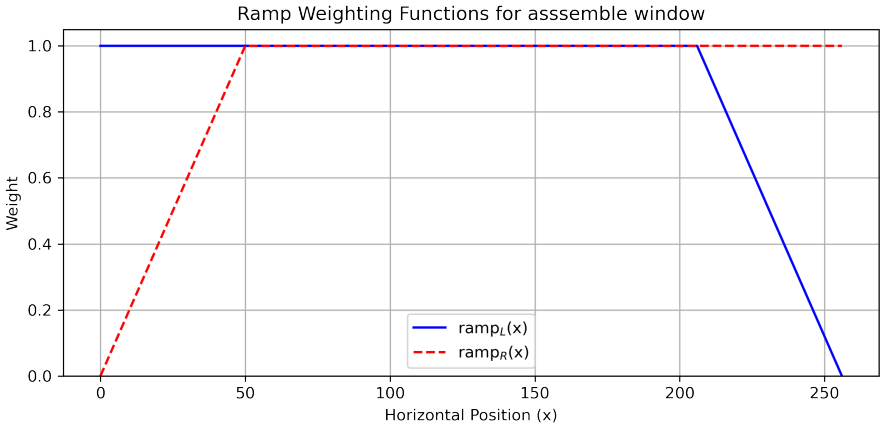

Figure 3.16: Ramp weighting functions used to merge adjacent windows. $ramp_L(x)$ applies a decreasing weight on the right side, while $ramp_R(x)$ increases weight on the left side.

The figure illustrates a comparison between a normalized seismic gather (see Section 3.2.2) and the effects introduced during the windowing process (see Section 3.2.1). These effects included border artifacts caused by the patch-wise assembly of the predicted windows. We implemented the ramp function to address this issue, with 50-pixel overlaps in both horizontal and vertical directions. As a result, the image on the right shows a predicted seismic gather with significantly improved continuity and a more realistic amplitude behavior along the window edges.

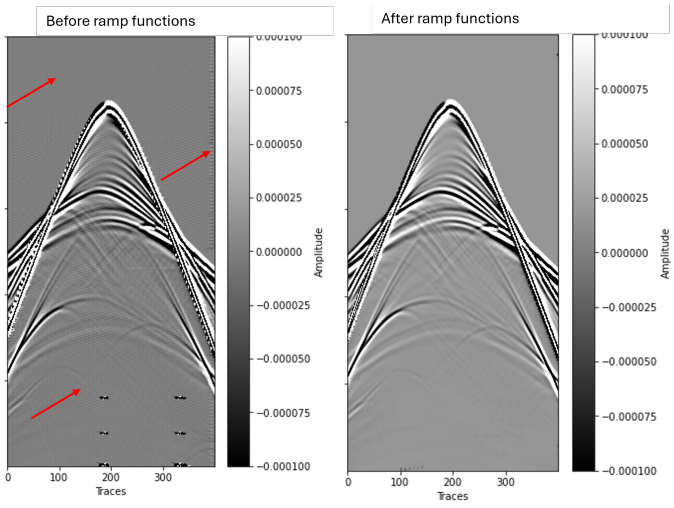

Figure 3.17: Ramp weighting functions used to merge adjacent windows. $ramp_L(x)$ applies a decreasing weight on the right side, while $ramp_R(x)$ increases weight on the left side.

3.6 Results

Before presenting the predictions, we examine the behavior of the loss function during training. Figure 3.18 illustrates the training behavior of the pix2pix model for seismic data matching by showing the evolution of four key loss components:Generator loss (L_G), Discriminator loss(L_D), L1 loss (L_{L1}), and Total GAN loss function (G^*). In each subplot, the green curve represents the training loss, while the red curve corresponds to the validation loss.

Generator loss (L_G) in figure 3.18 exhibits some oscillations in training and a gradually increasing validation curve. This behavior is typical of GAN training, where the generator attempts to improve realism under the dynamic feedback from the discriminator. Although fluctuations are expected, the increasing validation loss may indicate a tendency toward overfitting or a generator that becomes overly confident in fooling the discriminator.

Discriminator loss (L_D) in figure 3.18b, which decreases rapidly and stabilizes at a low value for both training and validation. These actions indicate that the discriminator becomes proficient early in the training and maintains its generalization capability across data splits, suggesting stable adversarial interplay between the two networks.

L1 loss (L_{L1}) 3.18c, a direct measure of the pixel-wise difference between the generated and target seismograms. This loss decreases smoothly and stabilizes, with training and validation curves closely aligned. The convergence toward low L1 values confirms that the generator learns to reproduce the overall structure of the seismic response with high fidelity.

Total GAN loss function (G^*) in figure 3.18d, which integrates adversarial and L1 losses into a single optimization objective. The curve exhibits a smooth downward trend for both training and validation, with the training loss slightly lower. This indicates effective learning with minimal overfitting and a good balance between structural accuracy and realism.

Overall, the figure confirms that the pix2pix model successfully optimizes both reconstruction and adversarial objectives. The discriminator remains stable and generalizes well, while the generator learns to produce seismograms that are both accurate and realistic. This validates the use of GAN-based architectures for correcting non-repeatability effects in TL seismic data.

Following the training of the DL model, the monitor data is used as input to correct for non-repeatability effects in the shallow area. Figures 3.19 present examples of the central Shot gather for the Baseline, the monitor, and the predicted monitor. Visually, the predicted monitor appears well correlated with the Baseline: the overall waveform shape, amplitude magnitude, and frequency content are highly similar, with no significant differences evident through visual inspection alone.

However, qualitative assessment is not sufficient to fully evaluate prediction accuracy. Therefore, several quantitative metrics were computed, including NRMS (Normalized Root Mean Square error), SNR (Signal-to-Noise Ratio), MAE (Mean Absolute Error), and SSIM (Structural Similarity Index Measure). Among these, the MAE was found to be particularly low, indicating that the predicted image closely approximates the target

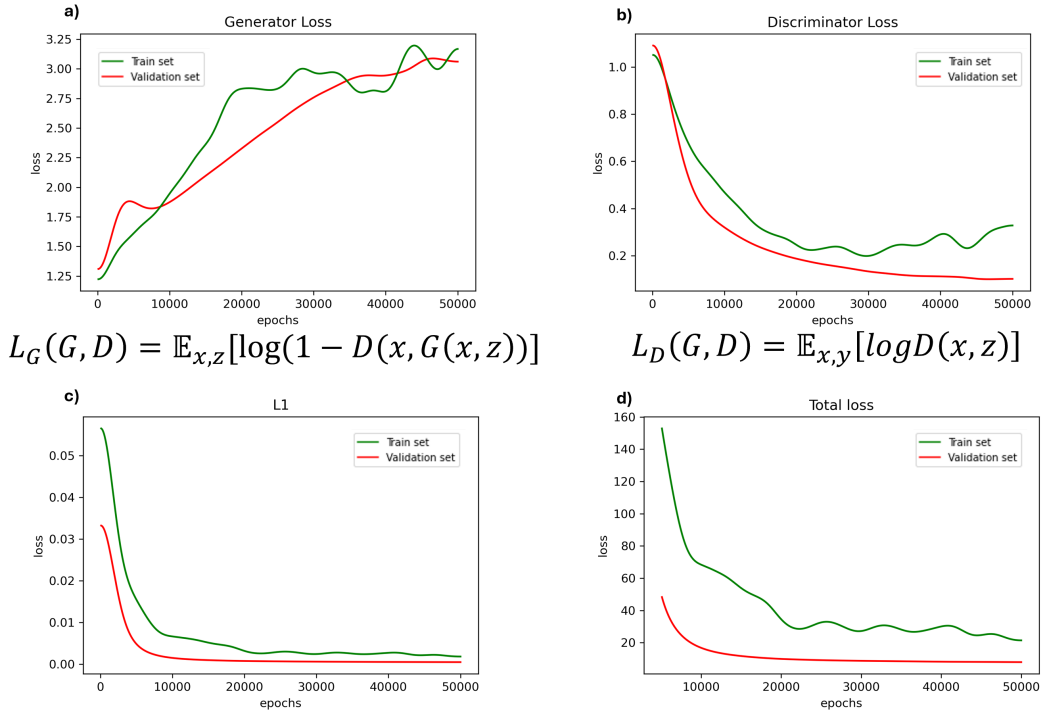

Figure 3.18: Overview of the loss functions used in the conditional GAN (cGAN) framework: (a) Generator loss function L_G, (b) Discriminator loss function L_D, (c) L_1 loss function, which enforces similarity between the generated and target data (conditioning term), and (d) total generator objective G^*. The weight for the L_1 loss is set to $\lambda = 10{,}000$, following hyperparameter selection to balance adversarial training and pixel-wise accuracy.

monitor in absolute amplitude terms. Additionally, the SSIM score reached 0.9, suggesting that the predicted output retains a high degree of structural and perceptual similarity to the reference.

These quantitative results reinforce the visual observation, confirming that the DL model effectively mitigates non-repeatability effects and produces monitor data that is both physically consistent and perceptually coherent.

Figure 3.20 focuses on the central trace extracted from the shot gather sequence illustrated in 3.19. In Figure 3.20a, we observe the behavior of the Baseline, monitor, and predicted traces in the shallow area. The predicted trace (green line) resembles the Baseline (blue line), indicating that the prediction adjusts the non-repeatability effects. However, this similarity is difficult to evaluate by simply plotting the traces, line by line. For this reason, Figure 3.20c presents a difference-based analysis. The **predicted difference** (magenta line) is computed as the predicted monitor minus the Baseline; the **real difference** (black line) corresponds to the actual difference between the monitor (with non-repeatability effects) and the Baseline; and the **perfect difference** (green line) repre-

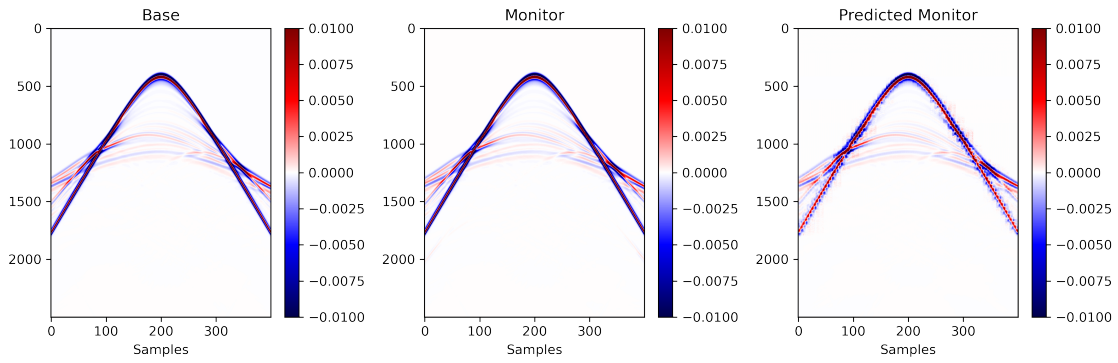

Figure 3.19: Seismograms from the central Shot gather: Baseline, monitor (with non-repeatability effects), and predicted monitor generated by the DL model.$\lambda = 10{,}000$

sents the difference between a clean synthetic monitor (without non-repeatability effects) and the Baseline. Although Figure 3.20 shows a substantial similarity between the predicted and baseline images, and metrics like SSIM indicate near-perfect performance, the geophysical trace-level analysis in Figure 3.20c suggests that the network does not reproduce the expected physical behavior with high accuracy. This performance highlights a limitation in relying solely on image-based metrics or visual inspection when evaluating seismic data predictions.

On the other hand, analyzing the reservoir area in the same way as we did for the shallow area, Figure 3.20b shows the behavior of the central trace, and Figure 3.20d presents the difference analysis focused on the reservoir region. As a result, we observe that the **predicted difference** exhibits a similar pattern to the **real difference**. However, one key observation emerges when we consider the amplitude ranges involved: as shown in Figure 3.4, the amplitude values used to train the network are up to ten times larger than those present in the reservoir zone. These results lead to the assumption that the network may struggle to learn or generalize patterns corresponding to such low-amplitude variations, which could explain its limited performance in accurately modeling the reservoir response.

In conclusion, for the experiment, the analysis reveals that while the DL model performs well in the shallow region where amplitudes are higher and non-repeatability effects are more pronounced, it shows limitations in the reservoir zone, where amplitude variations are significantly lower. Although standard image similarity metrics and visual inspection suggested a successful prediction, the trace analysis and amplitude-specific analysis exposed discrepancies in the network's ability to reproduce physically meaningful differences in low-energy regions. These findings emphasize the importance of incorporating geophysical validation beyond visual and statistical metrics, especially when evaluating model performance in reservoir characterization tasks. It also suggests that future training strategies should include amplitude balancing or specialized loss functions to improve sensitivity in low amplitude regions like the reservoir.

The previously analyzed results demonstrate the model's performance and stability,

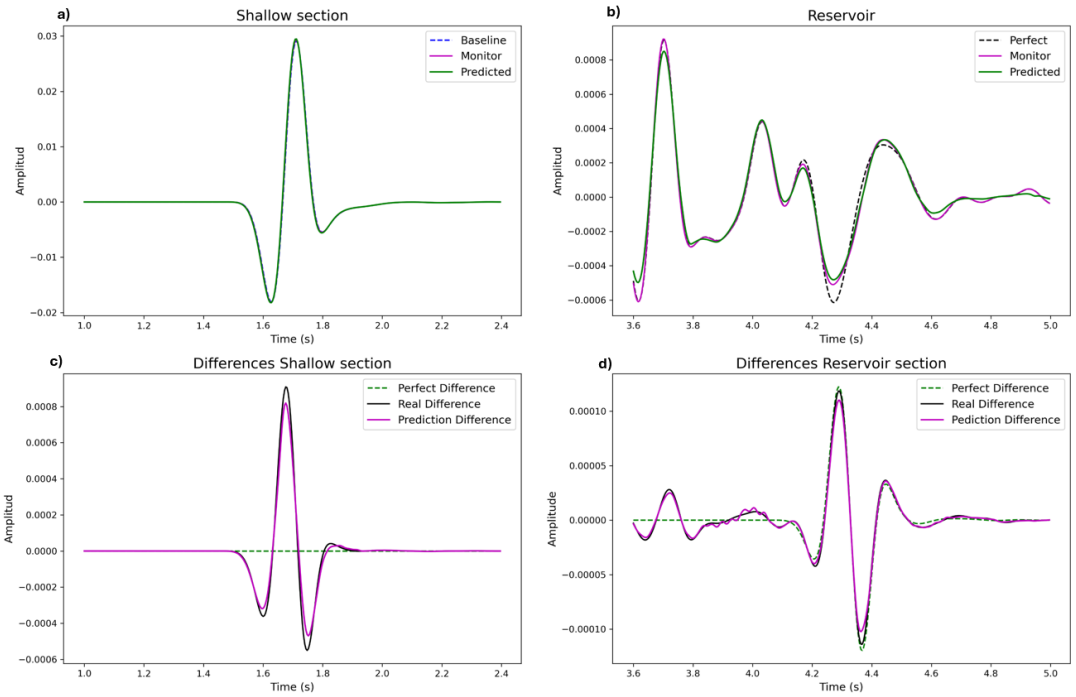

Figure 3.20: Central trace comparing. (a) Shallow area, (b) Reservoir area, (c) Difference in the shallow area, (d) Difference in the Reservoir area

as evidenced by the loss curves. Additionally, we evaluated the geophysical response of the experiment by examining whether the signal was corrected or adjusted according to the main objective: minimizing the effects of non-repeatability in the monitor data using a cGAN or a DL-based methodology, without affecting the reservoir signal which is precisely what we aim to interpret. This objective is the core motivation of this work.

As previously discussed, once we adjust the monitor data, we can integrate it into a DDFWI workflow. This integration can reduce computational costs and improve processing speed, enabling the practical application of this seismic processing tool.

We obtained the best results after conducting multiple experiments, which led to the final methodology and hyperparameter configuration as mentioned in section 3.3. First, Figure 3.21 presents the analysis of the Generator loss L_G, Discriminator loss L_D, L1 loss L_1, and Total GAN loss function G^* curves. When compared with the previous test shows in figure 3.18, a considerable improvement can be observed. The L_G shows no evidence of overfitting, unlike in the previous (Figure 3.18) where there was some overlap between the training and validation curves. Meanwhile, the discriminator converges more stably and never drops to zero, which is the expected behavior. Consequently, the L1 and Total GAN losses also show better performance, as both the generator and the discriminator directly influence them.

Figure 3.22 shows the comparison between the original Baseline and monitor data and the result inferred by our network. Visually, the outcome is within expectations: it preserves the shape of the monitor seismogram and remains within the expected amplitude

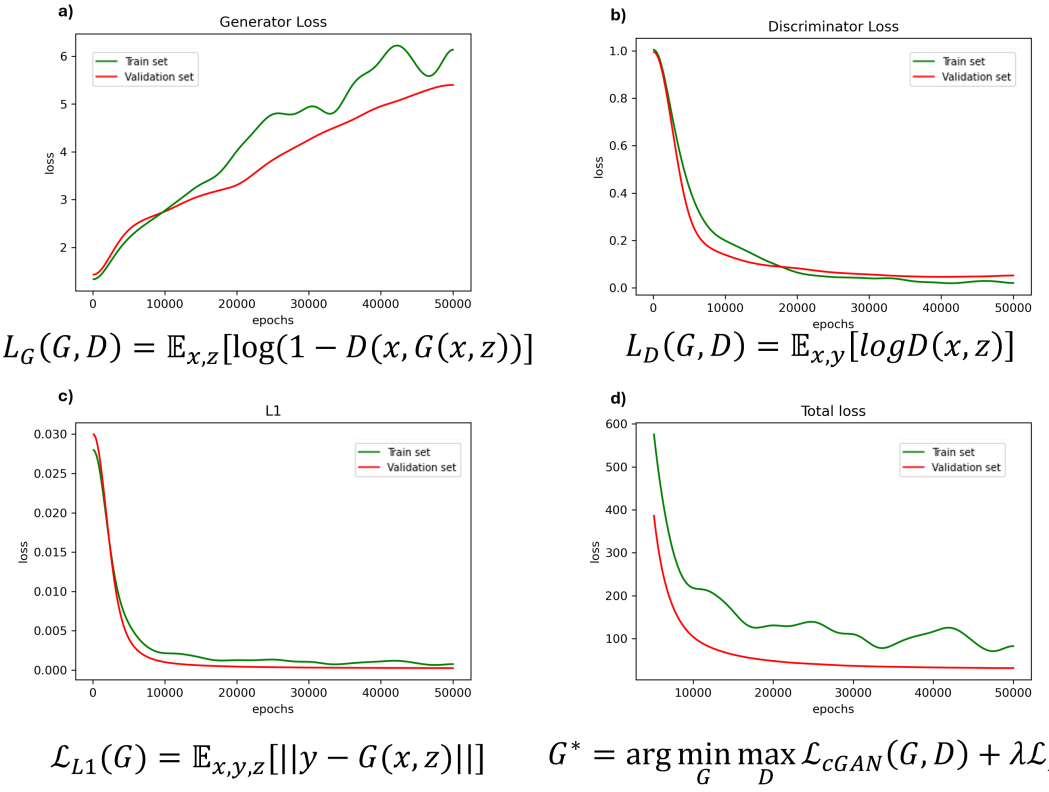

Figure 3.21: Overview of the loss functions used in the conditional GAN (cGAN) framework: (a) Generator loss function L_G, (b) Discriminator loss function L_D, (c) L_1 loss function, which enforces similarity between the generated and target data (conditioning term), and (d) total generator objective G^*. The weight for the L_1 loss is set to $\lambda = 100{,}000$, following hyperparameter selection to balance adversarial training and pixel-wise accuracy.

range. This is also supported by the metrics shown in Table 3.3, where all values fall within the acceptable range for each of the quantitative indicators that assess the similarity between the original image and the predicted result.

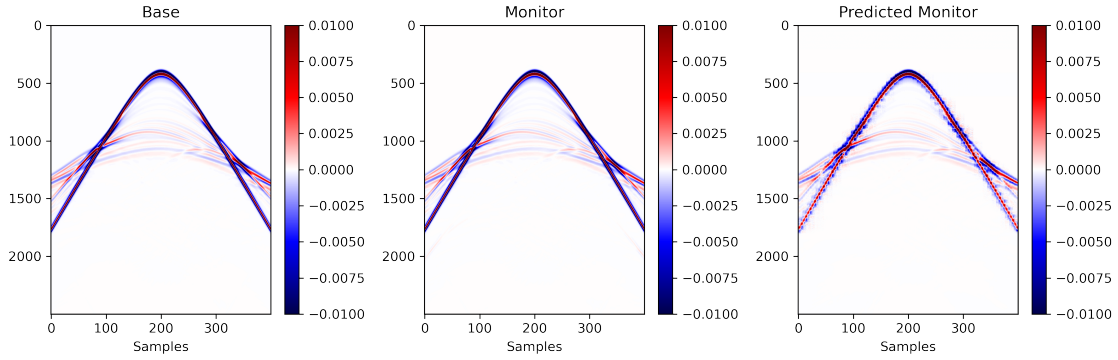

Figure 3.22: Seismograms from the central Shot gather: Baseline, monitor (with non-repeatability effects), and predicted monitor generated by the DL model.$\lambda = 100{,}000$

Finally, from a geophysical perspective and following the same analysis criteria used previously, Figure 3.23 shows the behavior of the central trace from the central Shot of the acquisition predicted by our network, both in the shallow and reservoir regions. When compared to the previous result (Figure 3.20), a significant improvement can be observed in the reservoir area.

The upper panels in Figure 3.23 present the direct comparison between the predicted trace and the baseline and monitor signals. In both the shallow and deep (reservoir) zones, the predicted trace closely matches the monitor trace, reflecting high fidelity in waveform reconstruction. In particular, the alignment of the main reflection events indicates that the network successfully learned to correct non-repeatability artifacts without distorting the underlying geophysical signal.

The lower panels display the difference traces: the ideal **perfect difference**(green line), the **real difference** (black line) between monitor and Baseline, and the **predicted difference**. It is noteworthy that the **predicted difference** (magenta line) follows the shape and amplitude of the **real difference** (black line) with great accuracy, especially in the reservoir area. This similarity suggests that the model is not only removing acquisition-related artifacts but also preserving meaningful TL changes. The fact that the predicted difference trace remains structurally coherent and aligned with the real difference confirms the model's capacity to reconstruct reservoir-related signal changes. These results reinforce its applicability in reservoir monitoring workflows, particularly in enhancing the input quality for DDFWI.

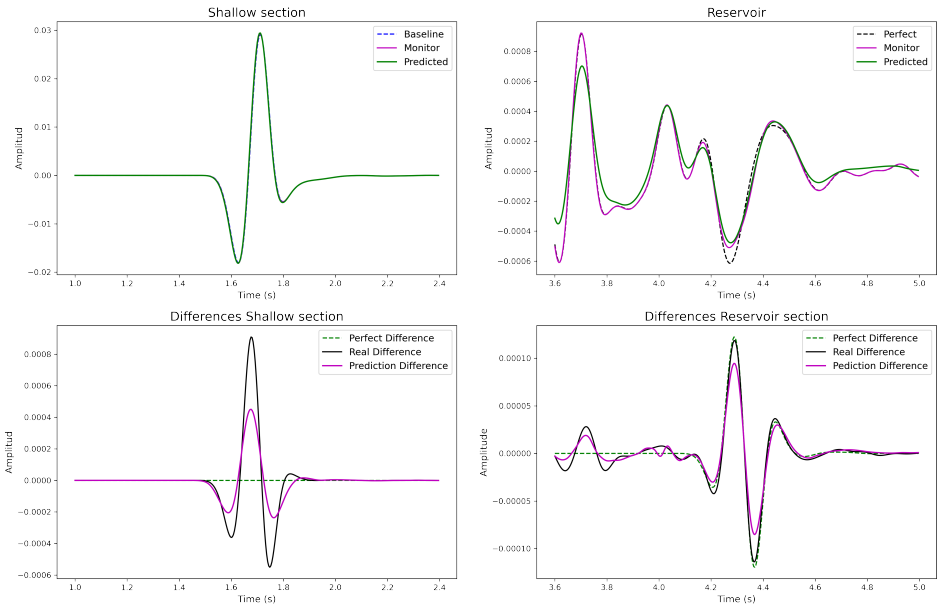

Figure 3.23: Central trace comparing. (a) Shallow area, (b) Reservoir area, (c) Difference in the shallow area, (d) Difference in the Reservoir area

For these two final results, we computed quantitative metrics to assess the behavior between the planned and predicted data, thereby providing a numerical evaluation of the model's performance. Table 3.3 presents the quantitative evaluation of the data matching performance for two different configurations of the regularization parameter λ. Test 1 uses $\lambda = 10{,}000$, while Test 2 increases this value to $\lambda = 100{,}000$.

Metric	NRMS (%)	MAE	MSE	SNR (dB)	PSNR (dB)	SSIM
Test 1	5.87	0.0003	3.3×10^{-6}	2.1	30	0.9
Test 2	5.84	0.0003	3.3×10^{-6}	2.1	30	0.9

Table 3.3: Quantitative evaluation metrics for data matching performance. The results indicate high similarity between the predicted and reference seismic windows.

The results show remarkable consistency across both settings. The NRMS values remain virtually unchanged (5.87% vs. 5.84%), as do the MAE and MSE, both indicating very low amplitude differences. PSNR values hold steady at 30 dB, reflecting high signal fidelity, and SSIM remains at 0.9, suggesting considerable structural similarity between the predicted and reference windows.

These observations highlight that increasing λ from 10,000 to 100,000 leads to a more stable training process, particularly when evaluated through cross-validation. While the individual performance metrics remain consistent, the model's convergence behavior improves notably with the higher regularization value. The results demonstrate that the choice of λ is fundamental to achieving stable and generalizable performance in the pix2pix-based data matching framework.

3.7 Technical Overview of the Data Matching Implementation

The implementation of our DL model to adjust non-repeatability effects in TL seismic data leverages advanced technological tools. First, we used the TensorFlow library to build and train the pix2pix model (Isola et al. 2018). TensorFlow, a widely adopted open-source platform for machine learning and DL, provided flexibility and efficiency for neural network development.

For preprocessing and normalizing the seismogram windows, we employed the NumPy library, which provides robust support for mathematical operations and large-dimensional array manipulation. We also used Keras, an API built on top of TensorFlow, to construct and experiment with deep learning models efficiently. Since pix2pix is a conditional generative adversarial network (cGAN), it requires specific adversarial training strategies. These were implemented using TensorFlow and Keras to ensure proper configuration of loss functions, optimizers, and training dynamics..

For result visualization and training monitoring, we employed Matplotlib, which was essential for generating plots of the seismogram windows, enabling visual inspection of the model's improvements. We carried out the Data Matching processing experiments on a node in a computational cluster. This node runs a Linux operating system (kernel

version 3.10.0-1160.49.1.el7.x86_64) and is equipped with 36 CPU cores across 2 physical sockets, with 8 CPUs allocated during the training job. It also provides access to a GPU-enabled environment, which is essential for the efficient training of GAN-based models such as pix2pix. The node offers a substantial amount of memory, with 360 GB of total RAM, of which 32 GB were allocated during the experiment. These specifications provided sufficient computational resources for both the adversarial training and the data-intensive preprocessing steps involved in normalizing and aligning synthetic seismogram pairs. The use of this GPU environment significantly accelerated the training process, making it feasible to run multiple epochs over high-resolution input windows within reasonable time frames.

The table 3.4 summarizes the key aspects of the data matching implementation developed to mitigate non-repeatability effects in TL seismic data.

Aspect	**Data Matching**
Objective	Reduce non-repeatability effects
Model Architecture	pix2pix (Conditional GAN)
Input Data	Seismogram pairs (Baseline and Monitor), shallow area
Target Output	Adjusted Monitor (noise-corrected)
Training Data	Synthetic noisy-clean seismogram pairs
Learning Type	Image-to-image translation with adversarial loss
GPU Usage	Required (GANs are computationally intensive)
Training Time	$\sim$150 sec / 1000 epochs
Inference Time	$\sim$150 seconds per data set
Main Libraries	TensorFlow, Keras, NumPy, Matplotlib
Preprocessing	Windowing and normalization
Application Role	Acts as a noise filter before implementation in a workflow for DDFWI

Table 3.4: Summary of the Data Matching implementation

Chapter 4

Velocity Inversion Target-Oriented

This chapter corresponds to the article **"Single-shot time-lapse target-oriented velocity inversion using machine learning"**, published in the journal Applied Sciences MDPI..

During the development of this thesis, I explored the application of deep learning techniques to seismic inversion. In the published work, I demonstrated how a neural network can be trained to perform target-centered inversion using a single shot gather. This approach shows that the inversion process can be slightly faster than conventional methods, potentially reducing computational costs.

4.1 Introduction

Time-lapse seismic is an important tool to unveil reservoir property changes due to oil/gas production (Johnston 2013). Indeed, during the production life cycle of a reservoir, the displacement of water, oil, and gas in the reservoir changes the impedance properties of the rocks,which are reflected as differences in the seismic record from one survey to another (Nguyen, Nam & Park 2015*b*). In this context, the 4D, or time-lapse, seismic has been stated as an important tool to support decision-making of oil/gas reservoir production. Moreover, in the current context of climate change (Mac Dowell, Fennell, Shah & Maitland 2017), 4D seismic is also a tool to monitor and manage CO_2 storage in geologic formations (Aldakheel, Pevzner, Gurevich & Glubokovskikh 2021).

In this study, we explore a straightforward 4D seismic inversion methodology. In our method, we assume acoustic propagation, and we invert seismic data to obtain velocity models. To standardize the notation along the paper, we consider that we have seismic acquisition from a baseline reservoir and a subsequent, during production, monitor acquisition. To be precise, we use the time-lapse difference (monitor minus baseline) of seismic data to invert the velocity model perturbation (monitor minus baseline), we call anomaly the change, or perturbation, in the reservoir region.

Target-oriented time-lapse inversion is a challenging task due to the complex physics relating the velocity changes in the reservoir to the seismic response (Cao & Roy 2017). In this context, a statistical inversion approach like the ML methodology seems to be appro-

priate to deal with uncertainty and lack of information. Some studies have proposed ML inversion techniques to produce velocity models directly from seismic data (Zhang, Si, Wu & Yan 2022), but the huge amount of data limits the success of this endeavor. Several studies on 4D seismic and ML focus on preprocessing of seismic data, to diminishes non-repeatability noise (Alali, Kazei, Sun & Alkhalifah 2022*b*, Alali, Smith, Nivlet, Bakulin & Alkhalifah 2021), or to find the water velocity layer (Duan, Yuan, Hatchell, Vila & Wang 2020, Araujo, Corso, Nascimento, Souza, Araujo & Barros 2024). In time-lapse seismic, some authors use seismic data and ML to estimate reservoir properties like water saturation or change in reservoir pressure (Rollmann, Soriano-Vargas, Almeida, Davolio, Schiozer & Rocha 2022, Xue, Araujo, Lopez, Wang & Kumar 2019, Kaur, Zhong, Sun & Fomel 2022). These studies generally use not only seismic data as input but also well-log data, reservoir parameters, or physics rock parameters (Dramsch, Christensen, MacBeth & Lüthje 2022, Corte & MacBeth 2021, Li, Alkhalifah & Guo 2021, Kim, Park, Seol & Byun 2023). The novelty of this paper is the simplicity: we employ ML to perform a target-oriented velocity inversion using only pre-stack data. Moreover, our method is target-oriented, computationally economical, and needs low seismic data.

In fact, we invert the reservoir anomalies using seismic data from a single shot.We notice that the usual 4D inversion assumes a well-known baseline acquisition. In special, the 4D target-oriented is focused on the inversion of a restricted reservoir region. Moreover, in the study area of our problem, the Brazilian pre-salt, the overburden issue caused by geomechanical movements above the target area is negligible. Therefore, the 4D target-oriented problem shares information with adjacent areas about the location of major geologic structures like horizontal layers or faults. In this restricted context of the 4D target-oriented problem, the ML inversion approach can be successful because several major issues of the geophysical inversion are minimized (Johnston 2013).

The seismic inversion based on one-shot acquisition is not feasible in standard 2D or 3D inversion problems. The reason for the unfeasibility of the one-shot inversion is due to the ambiguity in the location of the seismic events (Yilmaz 2001*b*). In fact, a seismic event caused by, for instance, a reflection in a layer interface does not present a well-defined position but rather can be localized at any point along an isochronous line centered in the seismic source (Wang 2016). In order to solve the ambiguity in the location of the seismic event, more shots are necessary. However, the location ambiguity is not a central issue in target-oriented 4D inversion once the overall layer positions are already well-defined for the baseline acquisition (Nguyen et al. 2015*b*). In this context, we believe that the 4D target-oriented ML inversion proposed in this paper can surpass the limitation of the single-shot acquisition and become a fast and economical option in time-lapse inversion.

The outline of the study is the following: in the methodology we present the basics of the seismic model used in the study, the methodology to construct synthetic models used in the ML training, an illumination study, and the Machine Learning method for seismic inversion. In the results, we show the inversions performed using the ML technique for two cases without and with 4D noise.

Finally, in the conclusion we present the main results, discuss the limitations of the method and point out future perspectives.

4.2 Methodology

4.2.1 Synthetic modeling

The proposed inversion methodology was tested and validated with synthetic seismic data. We used a realistic 20 km wide, 7.5 km deep P-wave velocity model (Figure 4.1) estimated from a pre-salt oil field (named Gato do Mato) located at the Santos basin, offshore Brazil (da Silva, Fagua Duarte, Almeida, Ferreira, Moura & Araujo 2021, Lopez, Neto, Cabrera, Cooke, Grandi & Roehl 2020). The work of (Cypriano, Yu, Ferreira, Huard, Pereira, Jouno, Khalil, Urasaki, Cruz, Yin, Clarke & Jesus 2019) provides more details about the geology of this area. Figure 4.1 also shows the acquisition geometry, formed of 1 shot and 49 ocean bottom nodes (OBNs). The source is located at 8 m depth and at the center of the analysis region in the lateral direction, and the receivers are at 2 km depth with 400 m spacing. Also, the source wavelet is a Ricker with peak energy at 5 Hz frequency (15 Hz maximum bandwidth), this frequency range covers the basic frequency band employed in seismic exploration. The reservoir region is located at about 6 km depth, as indicated in Figure 4.1,the depth of the reservoir region is usual in a pre-salt formation.

The above parameters correspond to the baseline survey. We also considered 5000 monitor surveys, which differ from the baseline by the added velocity perturbations in the reservoir region to represent the time-lapse effect of production-induced fluid substitution. Different velocity perturbations were applied to the monitor surveys, to simulate a number of possible fluid movements in the reservoir region due to oil and gas production. The velocity perturbations representing reservoir changes are described in more detail in section 4.2.2.

The dataset is composed of 5001 shot gathers in total, one associated to the baseline and the others with each of the monitor surveys. The seismograms were obtained by calculating the finite differences approximation of the acoustic wave equation in 2D for constant-density and isotropic media. The main numerical parameters were: time step of 4 milliseconds, total recording time of 10 seconds, and regular grid spacing of 20 meters.

In addition, to avoid reflections at the borders, we used as absorbing boundary condition 25 convolutional perfectly matched layers (CPML) added around the domain of analysis. Surface-related multiples were not modeled.

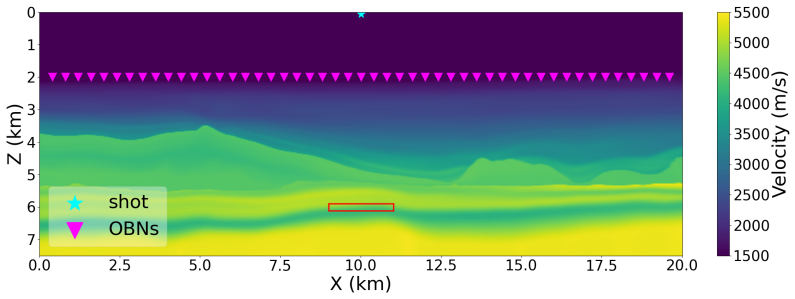

Figure 4.1: P-wave velocity model and acquisition geometry used to model the synthetic database.The rectangle indicates the reservoir region.

4.2.2 Reservoir Simulation

To test the inversion methodology, we produce a set of synthetic reservoir models based on the presalt velocity model. The simulated injection well is located below the salt formation at 6000 m depth and at 10000 m horizontal position. The region of the reservoir, the target region for inversion, is indicated with a red rectangle in Figure 4.1. Empirical experience from the industry suggests that, for the Brazilian pre-salt, the maximal production-induced velocity perturbation in the reservoir is around 3%. In addition, the average dimension of the reservoir is 2000 m in the horizontal direction and 200 m in the vertical direction.

To perform an inversion task using ML, we need an adequate training set, which in our case is a set of reservoir velocity models with a good variability in shape and velocity. In our algorithm, we consider the initial velocity model m_{init} as presented in Figure 4.1 and the reservoir velocity perturbation model m_{pert} that represent the changes in the reservoir due to the oil and gas production. The perturbations in the reservoir are related to water, oil, or gas displacement in the media, all these factors induce changes in the physical properties of the rocks and consequent changes in acoustic velocity (Nguyen et al. 2015*b*). To model the reservoir anomaly we employ a Gaussian statistical distribution centered around the injection well.

The standard deviation of the Gaussian follows an aleatory distribution from 300m to 500m. The center of the Gaussian was sampled around the point $(x,z) = (10\text{ km}, 6\text{ km})$.

4.2.3 Illumination study

In order to assess whether the single-shot geometry is able to extract some information from the reservoir region, we conducted an illumination analysis. Ray tracing (Wang 2014) is the conventional technique for illumination, but representing the wavefronts as rays is an approximation given the finite frequency of seismic signals. Instead, we performed the illumination analysis by migrating the "rabbit ear" component of the image condition of reverse-time migration (RTM). Higher migration magnitudes correspond to better illuminated regions of the subsurface.

We show in Figure 4.3 the illumination maps obtained by migrating different subsets of traces at a time. The target reservoir region is depicted in the Figure by a rectangle. In Figure 4.3(a) we depict the illumination that corresponds to traces 11-20, the left side. In Figure 4.3(b) we show the illumination corresponding to the five smallest-offset traces (traces 22 to 27). Figure 4.3(c) corresponds to the illumination of traces 21 to 30, and Figure 4.3(d) to traces 31 to 40. We do not show the illumination corresponding to the far-offset traces 1 to 11 or 41 to 49, because these traces weakly contribute to illuminating the target area. The conclusion from these three panels is that the reservoir region is well illuminated by the employed single-shot acquisition geometry.

4.2.4 Machine Learning training and testing

As mentioned in section 4.2.1, we simulated a database of 5000 synthetic anomalies in total. These data were randomly split into disjoint 80% train 20% test so that 4000

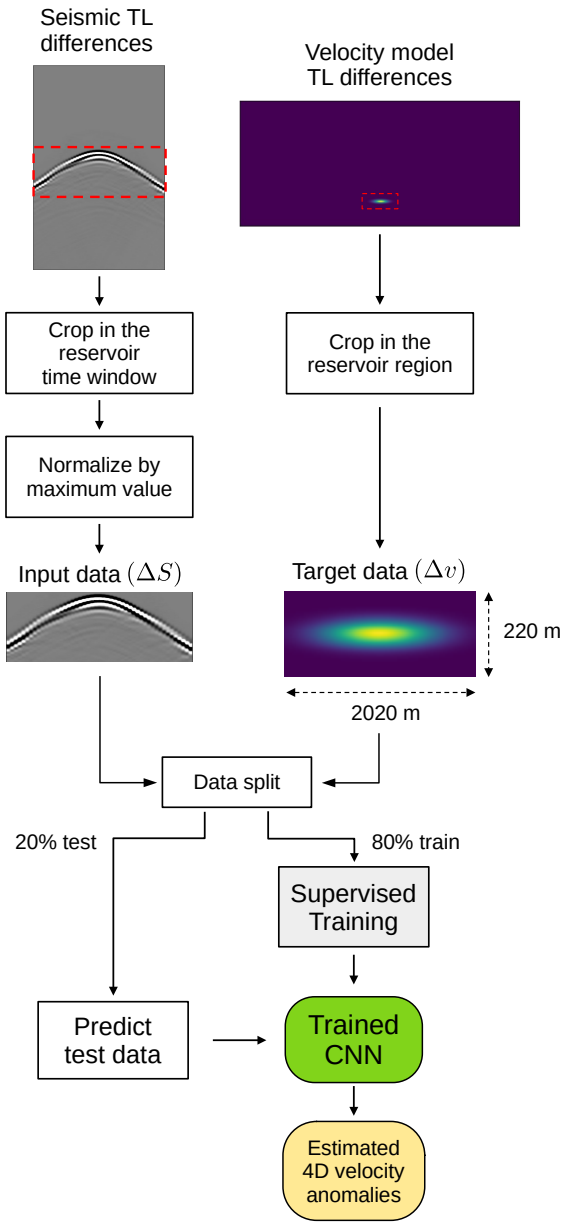

Figure 4.2: Block diagram of the proposed ML inversion methodology. The input set consists of the seismic time-lapse (TL) difference and the output the inverted reservoir anomaly.

anomalies were used for training the network and 1000 for testing.

In this work, we train a machine learning algorithm to invert the time-lapse difference of P-wave velocity in the reservoir region from the time-lapse difference of pre-stack seismic data. The proposed ML inversion methodology is illustrated in Figure 4.2. First, we calculate the differences of all monitor pre-stack seismograms relative to the baseline seismogram. Then we extract (crop) from each time-lapse seismic difference the subset of data within the time interval that contains most of the energy reflected from the

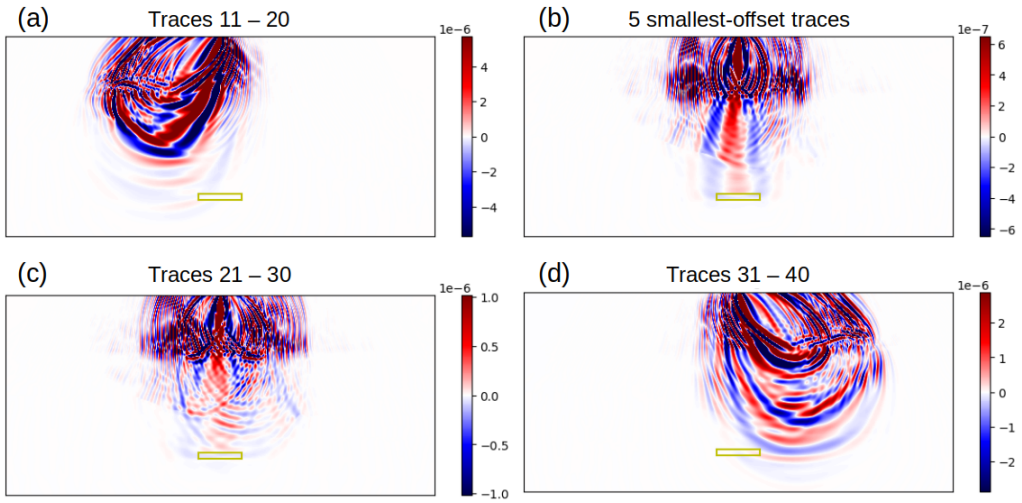

Figure 4.3: Illumination map produced by a single seismic source. This color map should be interpreted as areas where the energy of the seismic wave pass by and is captured by the receptors: (a) Corresponds to traces of sequence numbers from 11 to 20 (inclusive), (b) to the 5 smallest-offset traces, (c) to traces from 21 to 30, and (d) to traces from 31 to 40. The yellow rectangle indicates the target reservoir region as in Figure 4.1. The illumination map reveals that this simple acquisition geometry provides reasonable information to invert the target region.

reservoir. To avoid biasing the algorithm by the specific amplitude range of seismic data which depends on factors unrelated to the reservoir such as source wavelet amplitude and distances between source and receivers – we normalize each cropped difference by its maximum magnitude, resulting in the normalized time-cropped seismic data ΔS. Regarding the velocity model, the preprocessing steps are used to calculate the time-lapse velocity differences and extract the subset within the 220 m high, 2020 m wide reservoir region, resulting in the velocity anomaly Δv in the target region. Finally, ΔS and Δv are respectively passed as inputs and desired outputs (targets) for the supervised training of a neural network.

The core of the inversion methodology is formed by the neural network shown in Figure 4.4. The network has a novel architecture, inspired by the temporal convolutional network (TCN) (Bai, Kolter & Koltun 2018). The first step is to disregard from ΔS the large-offset traces 1 to 7 and 42 to 49, because the illumination study (section 4.2.3) revealed that those traces carry little reflection information from the reservoir region; keeping them could induce the network to learn spurious input-output associations. The remaining traces pass through a sequence of five convolutional layers arranged in a novel way. Recall that the standard convolutional layer applies a sliding 2D kernel to inputs that are neighboring to each other (unit dilation) in both dimensions. The layers we use apply the sliding 2D kernel to neighboring inputs in the horizontal axis, and further apart from each other in the vertical axis across layers. In other words, the convolutional layers use unit dilation in the space dimension, and exponentially increasing dilations in the time dimension. This way, we capture longer-duration time relationships between seismic

samples and spatially localized relationships across traces. The values of kernel size and dilation increasing rate were chosen such that each element at the output of the fifth layer is a function of all input elements (full history coverage). A series of convolutional layers with increasing dilations is at the core of the TCN architecture, widely used for sequence modeling; the difference here is that we combine this concept with the standard convolutions across traces.

The next layer is a global max pooling layer that retains only the maximum value of each feature map outputted by the convolutional layers, in order to reduce dimensionality and constrain the neural network complexity. With this pooling, only the strongest patterns make it through the rest of the network, and it also introduces some invariance to subtle changes in the input (Géron 2022), resulting in better generalization capabilities. The pooling outputs are submitted to a dropout layer with a 2% rate, and then to a fully connected layer of 1111 neurons. The next layer then applies a linear shifting and scaling to the 0–100 range, which is the interval of variation of the anomaly velocities. By doing that, we force the data in the previous layers to be within the [-1, 1] interval for better conditioning of the error gradient during backpropagation, increasing the stability and convergence of training. Finally, the data are reshaped as an 11×101 matrix to form the predicted velocity anomaly Δv. The proposed network architecture has about 2.39×10^5 parameters. For training, we used Adam optimizer with mean squared error (MSE) loss function, adaptively decreasing learning rate (starts at 10^{-3}, reduces by 80% whenever validation loss does not improve for 25 consecutive epochs), batch size 32 and early stopping as the training stopping criterion. Dropout and early stopping are regularization measures to reduce the overfitting issue (Bishop 1995, Géron 2022). Figure 4.3 shows the MSE loss curves computed for the training and the validation sets. The loss function indicated in the figure shows a good convergence during the neural network training and validation process. The curves for the training and validation subsets are very close to one another, an evidence that the training converged and that the network is not overfitting.

To compare the accuracy of the inverted velocity reservoir anomaly with the true anomaly, we employ the Structural Similarity Index Measure (SSIM) (Wang, Bovik, Sheikh & Simoncelli 2004*a*), which is a scalar metric that quantifies the perceptual similarity between images. SSIM varies in the [0, 1] range; one indicates that the compared images have good similarity, and zero means that they are very different.In addition, we employ a Mean Average Error (MAE) to quantify uncertainty in our analysis. The MAE estimate the difference pixel-to-pixel between the ML estimated velocity anomaly and the real anomaly.

4.3 Results

4.3.1 Inversion for perfect repeatability

The ML target-oriented inversion methodology is tested in the synthetic data set introduced in section 4.2.1. Initially, we tested the ML method in noise-free data. The test was performed for the 1000 test samples, which corresponds to 20% of the total of 5000

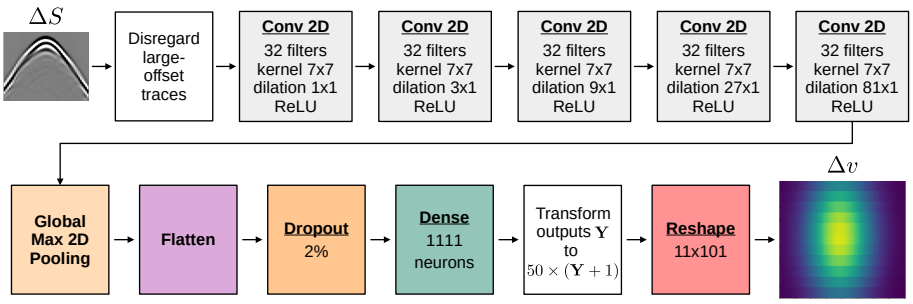

Figure 4.4: Proposed neural network architecture. The input of the ML is the seismic time-lapse difference within the reservoir time window and the output is the velocity anomaly of the target region.

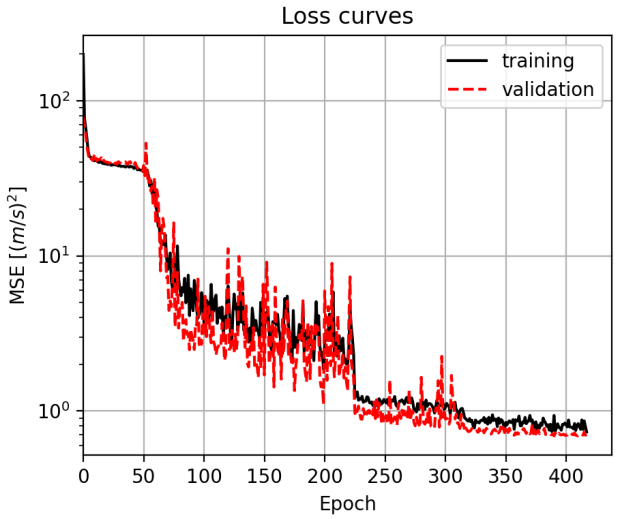

Figure 4.5: Loss curves computed on the training and validation subsets during the training of the neural network employed for velocity inversion. The loss function is calculated with velocities in the scale of the 4D anomalies (0–100 m/s range).

samples. The obtained values of the SSIM index for the testing inversions are inside the interval $[0.9727; 0.9999]$, which is an excellent result for image reproducibility. Moreover, the outcomes of the ML method are illustrated in the first columns of Figure 4.6 in which we compare true and predicted images for five cases according to SSIM criteria. The illustrated examples are located at key percentiles of the SSIM distribution on the test set: minimum (worst case), the three quartiles, and maximum (best case).

A more detailed view of the reservoir anomaly is depicted in the last two rows of Figure 4.6, which shows the central vertical and central horizontal velocity profiles. We notice in these figures a very good agreement between true and predicted velocity profiles for the noiseless case. Indeed, even in the worst scenario, the predicted profiles are very close to the true profiles. As we shall see in section 4.3.2, the predicted anomalies are impacted by non-repeatability noise.

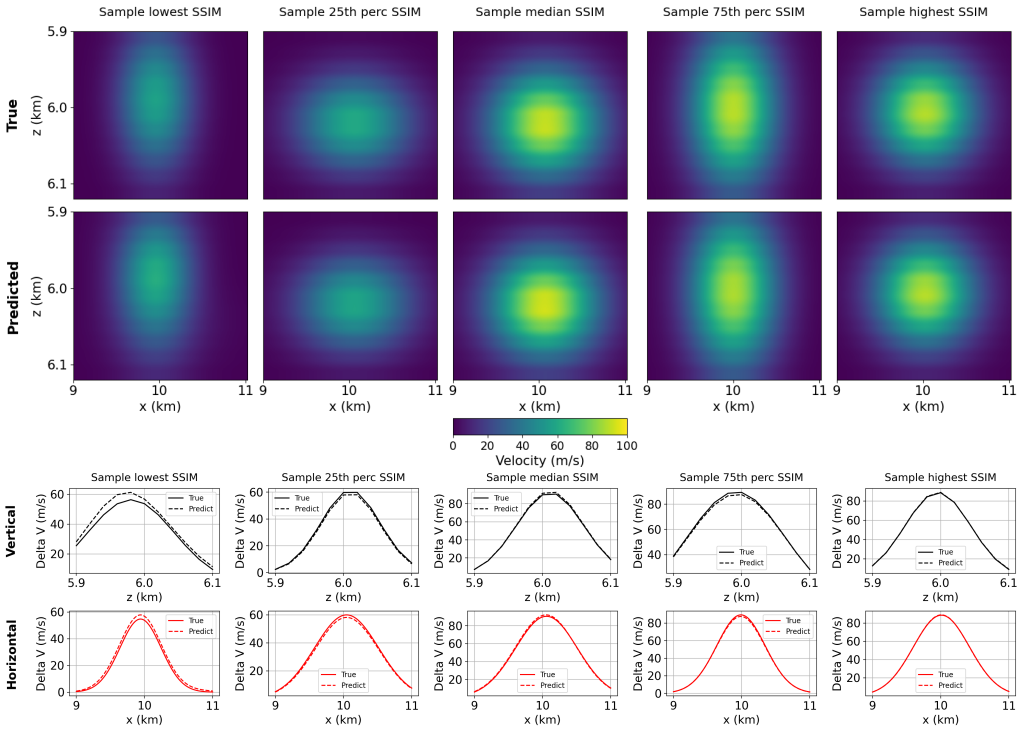

Figure 4.6: Inversion results of individual samples for the perfect repeatability scenario. The yellow pattern in the panels represent the velocity anomaly, the horizontal and vertical dimensions reproduce the reservoir region indicated in the rectangle of Figure 4.1.Comparison of true (first row) and inverted (second row) reservoir anomaly for five samples at key points of the SSIM distribution on the test subset: minimum (worst case), 25th percentile, median, 75th percentile and maximum (best case). The last two rows compare, respectively, the central vertical and central horizontal velocity profiles of the velocity anomalies.

4.3.2 Inversion in non-repeatability scenarios

The ML inversion methodology developed in this paper was designed to work on a perfect repeatability scenario. The ML inversion input is the seismic data difference, which in practice is contaminated by non-repeatability effects. In the industry, data non-repeatability is treated using several techniques such as 4D binning (Nguyen et al. 2015*b*), matched filtering (Rickett & Lumley 2001, Lumley, Adams, Meadows, Cole & Wright 2003), amplitude balancing (Rickett & Lumley 1998) and time warping (Hale 2013). However, the complete removal of non-repeatability effects is impracticable. In this way, we test our methodology in this more challenging scenario in which some treatment to erase 4D noise was implemented, but a residual non-repeatability effect is still present. We model such non-repeatability residual by introducing uniform-distributed random perturbations to the positions of receivers in the lateral direction. The new data testing set was created similarly to the synthetic seismic data presented in section 4.2.1, the difference is the aleatory shaking in the receiver position. We employ two non-repeatability noises according to the maximal aleatory variation in the receiver position: +/- 0.1 m and

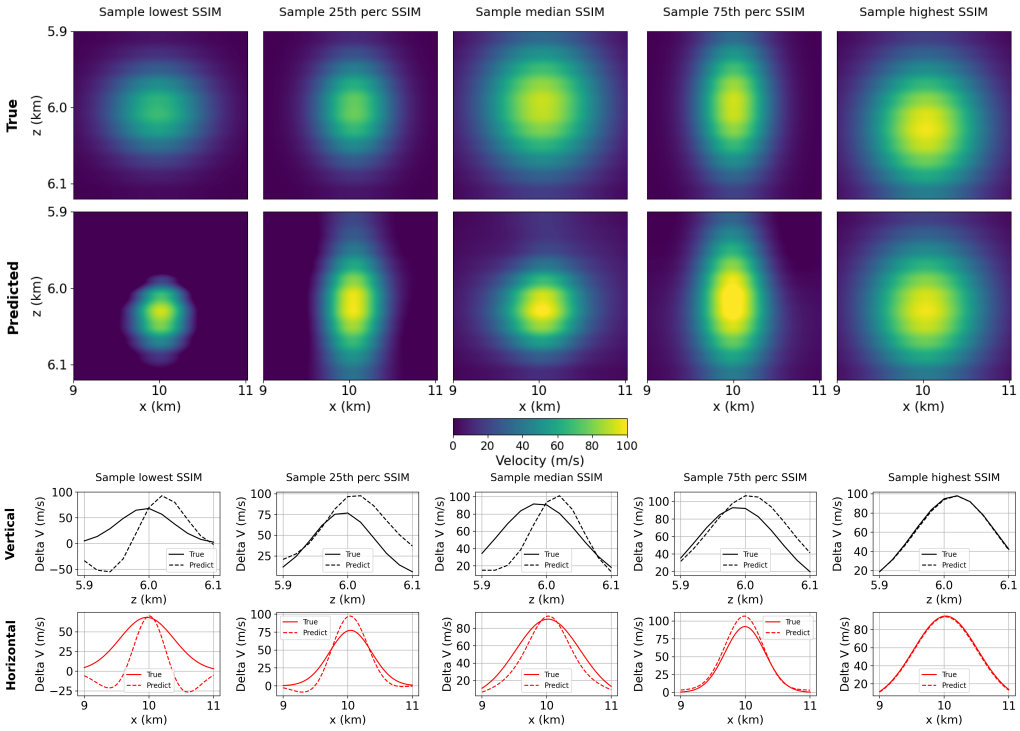

Figure 4.7: Inversion results of individual samples for a scenario with non-repeatability, modeled by randomly moving the receivers in the lateral directions, with maximum perturbations equal to +/- 0.1 m. The yellow pattern in the panels represent the velocity anomaly, the horizontal and vertical dimensions reproduce the reservoir region indicated in the rectangle of Figure 4.1. Comparison of true (first row) and predicted (second row) time-lapse velocity anomalies in the target region for specific samples of a test dataset contaminated with geometry non-repeatability noise. The 4D noise was modeled by randomly shifting the receivers in the lateral direction, with maximum perturbations equal to +/- 0.1 m. The shown samples are located at key percentiles of the SSIM distribution on the referred dataset: minimum (worst case), three quartiles, and maximum (best case). The last two rows compare, respectively, the central vertical and central horizontal velocity profiles of the velocity anomalies.

+/- 0.5 m.

The testing of ML inversion methodology with non-repeatability noise is shown in Figures 4.7 and 4.8 for maximum receiver variation equal to +/- 0.1 m and +/- 0.5 m, respectively. The basic schema of these two figures repeats the previous Figure 4.6 that shows the inversion for the ideal case of perfect repeatability. The two upper rows show the true and the predicted velocity anomaly according to key percentiles of the SSIM distribution on the dataset: minimum (worst case), median case, maximum (best case), and two intermediary quartiles. The two lower rows of these Figures show the central vertical and central horizontal velocity profiles of the velocity anomalies. The medians of the estimated SSIM index of ML inversion are 0.69 and 0.48 for the +/- 0.1 m and +/- 0.5 m cases, respectively.The Kruskal-Wallis test for the SSIM shows significant difference

among the three scenarios ($\chi^2 = 2082$, df = 2, p-value < 0.00001). An analysis of the MAE reveals the same pattern of the SSIM quantifier. The median of MAE corresponding to the noiseless case is 0.38 while to the noisy case the medians are 230 and 550 for the +/- 0.1 m and +/- 0.5 m cases, respectively. The Kruskal-Wallis test for the MAE also reveals significant difference among the three treatments ($\chi^2 = 2144.9$, df = 2, p-value < 0.00001).

The general overview of Figures 4.6, 4.7 and 4.8 reveals that non-repeatability noise degrades inversion performance, as expected. The worsening of the ML methodology is observed by visual inspection of the velocity anomalies of the mentioned figures. In addition, the velocity profiles, both vertical and horizontal, also are affected by the non-repeatability noise. The difference between true and predicted velocities profiles of Figure 4.6 contrasts with profiles of Figures 4.7 and, in especial, Figure 4.8 that has a strong noise. In the horizontal direction, the peak of the anomaly is centered at 10 km and this position is not perturbed with noise. However, in the vertical direction, the peak shifts from 6 km to 6.04 in the worst scenario. In this way, the non-repeatability noise produces a shift depth in the peak of the anomaly.

In order to quantify the uncertainty of the inversion methodology, we compute the error in the prediction of the velocity anomaly. Figure 4.9 estimate the pixel-to-pixel difference between ML predicted and true velocity anomaly. Indeed, Figure 4.9 is an error color map where horizontal and vertical axis reproduce the same dimensions of previous Figures 4.6, 4.7 and 4.8. The first panel, Figures 4.9 shows the error for the perfect repeatability case, while Figures 4.9 and (c) for the repeatability noise cases corresponding to +/- 0.1m and +/-0.5 m, respectively. The error magnitude is much larger in the noise scenarios, the error bars in the panels are not in the same scale. The general pattern with and without noise is distinct, the noiseless case is characterized by an underestimation in the peak of the anomaly while the noisy cases are characterized by an overestimation in this peak and an underestimation in the region above the anomaly. This information is important for the geophysical interpreter that in the field case will deal with a noisy environment. In this situation, the interpreter should consider that the anomaly peak, whose velocity anomaly is produced by a fluid replacement, could be overestimated.

4.4 Conclusion

The 4D seismic inversion encompasses a myriad of different techniques (Johnston 2013, Corte, Dramsch, Amini & MacBeth 2021). In this work, we make the option for methodological simplicity, instead of introducing rock physics, production features, or detail reservoir characterization (Rollmann et al. 2022, Dramsch, Corte, Amini, Lüthje & MacBeth 2019). we employ seismic data produced from an acoustic wave and invert the reservoir velocity using just one shot and dozens of receivers.

The results are satisfactory when we consider the low computational cost and the economical use of input seismic data. The computational time for training and testing the 5000 samples used in this paper was around 4 hours in a Center for high-performance computing of our University (NPAD/UFRN). In the same computing facility, a similar

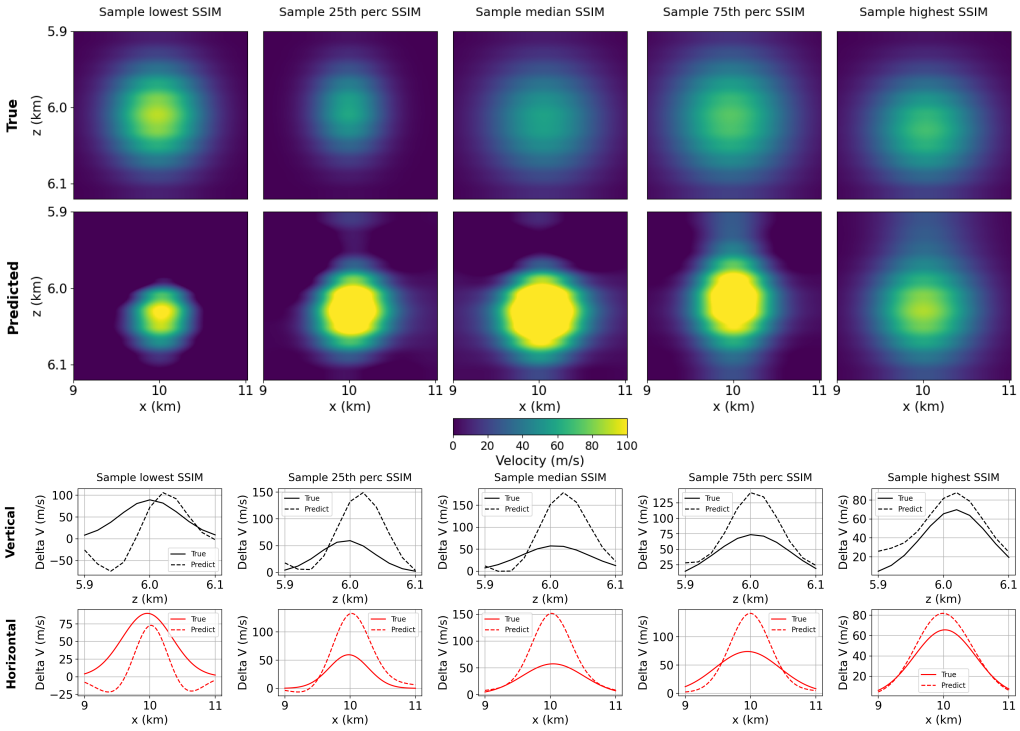

Figure 4.8: Inversion results of individual samples for a scenario with non-repeatability, modeled by randomly moving the receivers in the lateral directions, with maximum perturbations equal to +/- 0.5 m. The yellow pattern in the panels represents the velocity anomaly, the horizontal and vertical dimensions reproduce the reservoir region indicated in the rectangle of Figure 4.1.Comparison of true (first row) and predicted (second row) time-lapse velocity anomalies in the target region for specific samples of a testing dataset contaminated with geometry non-repeatability noise. The illustrated samples are located at key percentiles of the SSIM distribution on the referred dataset: minimum (worst case), three quartiles, and maximum (best case). The last two rows compare, respectively, the central vertical and central horizontal velocity profiles of the velocity anomalies.

4D inversion (da Silva et al. 2021) took one hundred hours to perform a similar inversion using full-waveform inversion (FWI) methodology. The high computational cost of FWI is related with the huge memory usage, the wave propagation associated with one iteration in the optimization procedure use data from a hundred seismic shots. Moreover, we emphasize that in our inversion effort, we employed a single shot, which in part explains the comparative low computational cost. Indeed, a usual seismic inversion can use a hundred shots to provide good illumination to perform an acceptable inversion.

We point out an important limitation of our methodology: the diversity and quality of the training dataset. The success of the ML inversion is strongly dependent on the quality of the training dataset. In fact, in a field case, if the actual velocity anomaly significantly differed from the anomalies presented in the training dataset, the inversion could not be successful. In this way, a successful application of our methodology in a field case would depend on a good simulation of the reservoir changes over time, that means, reservoir

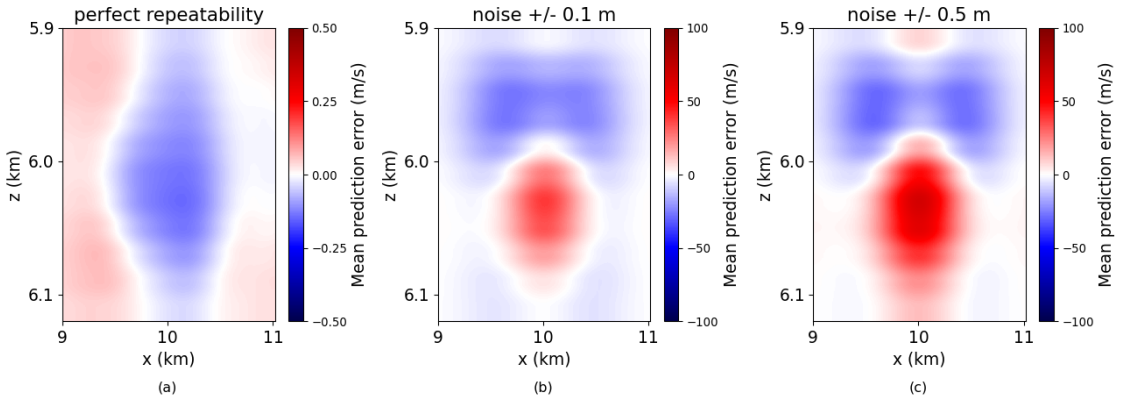

Figure 4.9: Spatial distribution of Δv prediction errors for the test scenarios with (a) perfect repeatability, (b) +/- 0.1 m geometry 4D noise and (c) +/- 0.5 m geometry 4D noise.

simulations with an ample production scenarios.

The present study comprises a simple target-oriented ML inversion methodology based on one-shot acquisition geometry and a novel neural network architecture. In future work, we intend to expand our methodology in several directions. For instance, to introduce in the acquisition geometry more shots for each survey to test the method with greater illumination in the reservoir region. Another open line of research is to improve the ML training with seismic data contaminated by non-repeatability noise. We stress that in our methodology, we initially suppose that we can erase the non-repeatability noise, but we are aware that in practical situations this is impossible. In this way, we tested the methodology with a residual non-repeatability noise and the results are acceptable. To conclude, we present in this paper an ML method to perform 4D seismic inversion that has a low computational complexity when compared with traditional FWI inversion.

Chapter 5

DL Seismic Inversion vs. DDFWI

This chapter presented the methodology and application of single-shot inversion for a presalt velocity model, drawing on previously published work. In this chapter, we build upon the same neural network architecture and methodological principles, specifically the strategy of generating multiple monitor velocity models. However, the key difference lies in the design of the two new scenarios, which incorporate more geologically realistic anomalies. These anomalies aim to simulate subsurface conditions where the behavior of the velocity perturbations more closely resembles reservoir dynamics, allowing for a more robust evaluation of the model's generalization and interpretability.

Additionally, the created scenarios were inverted using the trained CNN and compared against results obtained through Double-Difference Full Waveform Inversion (DDFWI), serving as a benchmark. This study conducted a comparative analysis to evaluate whether the DL-based approach offers practical advantages over traditional physics-driven methods, particularly in terms of performance, computational efficiency, and interpretational value.

5.1 Methodology

5.1.1 Syntethic Data Generation

The first step was to construct a synthetic dataset designed to replicate realistic 4D seismic anomalies in the Brazilian pre-salt context. This involved using a different pre-salt velocity model than the one employed in the previous chapter and generating 2D synthetic reference and monitor data. The acquisition was simulated using an ocean bottom node (OBN) configuration in a single-shot setup. Figure 5.1 represents the flowchart used for data generation. Two different data sets were created. In the first case, 1000 monitor seismograms were generated by introducing random Gaussian perturbations, simulating potential anomalies within the reservoir. These anomalies represent a *hardening* reservoir, where the velocity in the monitor model is higher than in the baseline. Figure 5.2 is an example of the data generated for the first scenario.

In the second case, another 1000 monitor seismograms were generated, this time simulating two distinct types of Gaussian perturbations for each monitor: one representing a *hardening* anomaly and the other a *softening* corresponding to production effects such as gas breakout or pressure reduction anomaly Figure 5.3 is an example of the data generated

for the second scenario. In both scenarios, the seismic data were modeled using acoustic wave propagation and the acoustic wave equation. As a result, seismograms composed of 560 traces and 1500 samples were generated.

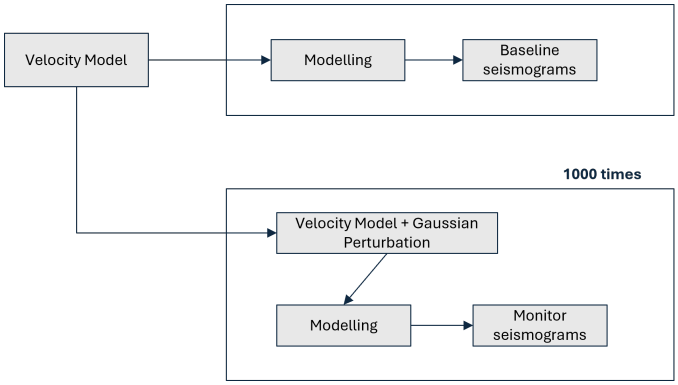

Figure 5.1: Flowchart illustrating the synthetic data generation process. Using a pre-salt velocity model.

Simulating a Ocean Bottom Node (OBN) geometry, the Baseline and the monitor seismograms were generated using the following parameters in our simulations:

- **Receiver (Node) Spacing:** 1.75 km, ranging from 1 km to 14 km along the horizontal direction. This is coarser than conventional acquisition setups, which typically use 500 m spacing.
- **Source Spacing:** 50 m, covering the entire 14 km horizontal extent.
- **Grid Discretization:** The computational grid uses a uniform spacing of $\Delta x = 25$ m and $\Delta z = 25$ m.
- **Grid Size:** The total number of grid points is 560 in the horizontal (x) direction and 280 in the vertical (z) direction.
- **Survey Types:** A baseline survey was modeled, along with 1000 monitor surveys that simulate time-lapse changes in the reservoir.

Synthetic time-lapse anomalies were modeled using random Gaussian perturbations within the velocity model, confined to a specific spatial window. The following random parameters were used for the case with a single anomaly:

- $\sigma_x \sim \text{Uniform}(150\text{ m}, 250\text{ m})$: standard deviation in the horizontal direction.
- $\sigma_z \sim \text{Uniform}(180\text{ m}, 220\text{ m})$: standard deviation in the vertical direction.
- $x_{\text{center}} \sim \text{Uniform}(4.7\text{ km}, 5.2\text{ km})$: horizontal coordinate of the anomaly center.
- $z_{\text{center}} \sim 5\text{ km} + \text{Uniform}(0\text{ m}, 10\text{ m})$: vertical coordinate of the anomaly center.
- $\Delta V_{\text{max}} \sim \text{Uniform}(50\text{ m/s}, 100\text{ m/s})$: maximum change in velocity inside the anomaly.

Two Gaussian-shaped anomalies were introduced to simulate time-lapse variations. The random parameter distributions used for each anomaly are as follows:

Left Anomaly:

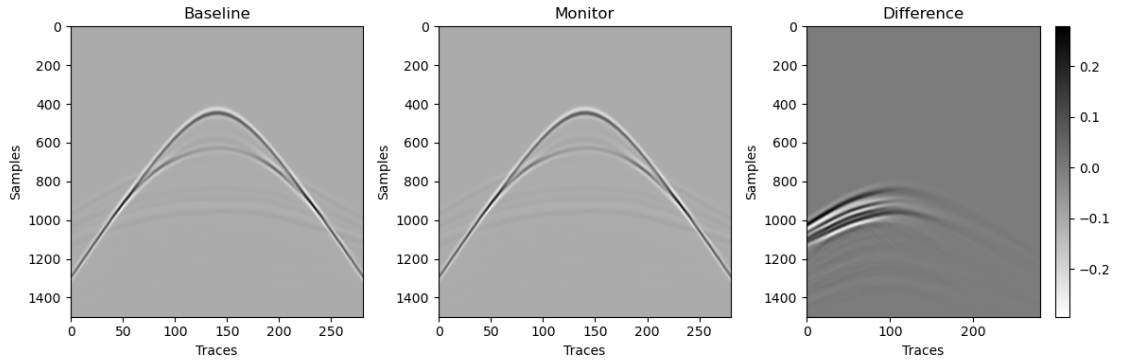

Figure 5.2: Example of central shot seismogram modelling for the database. The figure shows, from left to right: the baseline seismogram (Baseline), the monitoring seismogram (Monitor), and their difference (Difference), which highlights temporal anomalies. These examples are part of the dataset used to train and evaluate the learning models.

- $\sigma_x \sim \text{Uniform}(150\text{ m}, 250\text{ m})$: standard deviation in the x direction.
- $\sigma_z \sim \text{Uniform}(50\text{ m}, 100\text{ m})$: standard deviation in the z direction.
- $x_{\text{center}} = \text{Uniform}(5\text{ km})$: center position along x.
- $z_{\text{center}} = 4.9$ km: fixed depth.
- $\Delta V_{\max} \sim \text{Uniform}(0\text{ m/s}, 100\text{ m/s})$: positive velocity perturbation.

Right Anomaly:

- $\sigma_x \sim \text{Uniform}(200\text{ m}, 300\text{ m})$: standard deviation in the x direction.
- $\sigma_z \sim \text{Uniform}(50\text{ m}, 100\text{ m})$: standard deviation in the z direction.
- $x_{\text{center}} = \text{Uniform}(9.5\text{ km})$: center position along x.
- $z_{\text{center}} = 4.9$ km: fixed depth.
- $\Delta V_{\max} \sim \text{Uniform}(-100\text{ m/s}, 0\text{ m/s})$: negative velocity perturbation.

5.1.2 Deep Learning-Based Inversion

The second stage of the methodology involves training a convolutional neural network (CNN) to perform 4D seismic inversion. The CNN is trained to map seismic amplitude differences to the underlying velocity changes in the subsurface. Figure 5.4 and 5.5 resume the workflow and the network.

The input to the network is a 40 ×100 image representing the difference between the monitor and baseline seismograms for the first scenario and 40 ×280 for the second one. The target is a velocity map of the same size as the region where the anomaly was introduced. This formulation enables the network to learn the spatial correlation between seismic amplitude variations and physical changes in the subsurface. The inversion process was made shot by shot for a total of nine shots.

The architecture consists of four convolutional layers, each followed by max-pooling layers with a stride of 2. These convolutional blocks extract spatial features at different scales. After the final pooling layer, the feature maps are flattened and passed through two

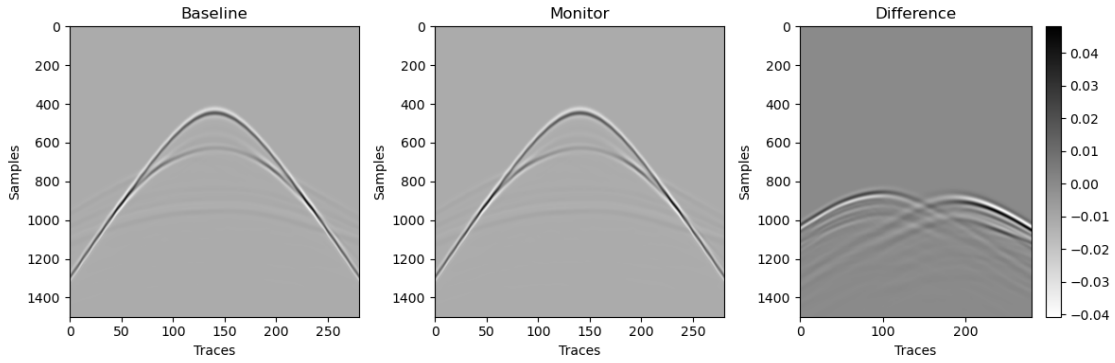

Figure 5.3: Example of seismograms modelling for the database. The figure shows, from left to right: the baseline seismogram (Baseline), the monitoring seismogram (Monitor), and their difference (Difference), which highlights both temporal anomalies. These examples are part of the dataset used to train and evaluate the learning models.

fully connected layers. The last layer reshapes the output into a 40 ×100 or 40 ×280 velocity map according to each experiment. ReLU activation functions are used throughout the network except in the final layer.

The network is trained using Mean Squared Error (MSE) as the loss function. Early stopping is implemented to halt training when performance on the validation set no longer improves. Training is carried out using an 80/20 split of the dataset, with 20% reserved for validation. Although each training session is configured for up to 20,000 epochs, early stopping automatically interrupts the process if the validation loss fails to improve by at least 0.01 over 60 consecutive epochs.

5.1.3 Inversion normalization

For the inversion task using a CNN, the complete set of input seismograms (Baseline and Monitor) was first normalized globally using z-score normalization, which centers the data distribution by subtracting the mean and scaling by the standard deviation. This normalization is particularly appropriate for regression tasks, where the output (i.e., the velocity model) must capture absolute physical variations.

After normalization, we compute the time-lapse difference between each Monitor seismogram and the Baseline. From each difference gather, we extract a subset of time samples defined by fixed cropping indices (`irw_min` and `irw_max`), corresponding to the interval where most of the energy reflected from the reservoir is concentrated.

To reduce the influence of acquisition-related amplitude variations—such as differences in source strength or source-receiver geometry—each cropped difference is further normalized by dividing by its maximum absolute amplitude. This final step produces the input data used for training, denoted as the normalized time-cropped seismic difference ΔS, as detailed in Algorithm 5.

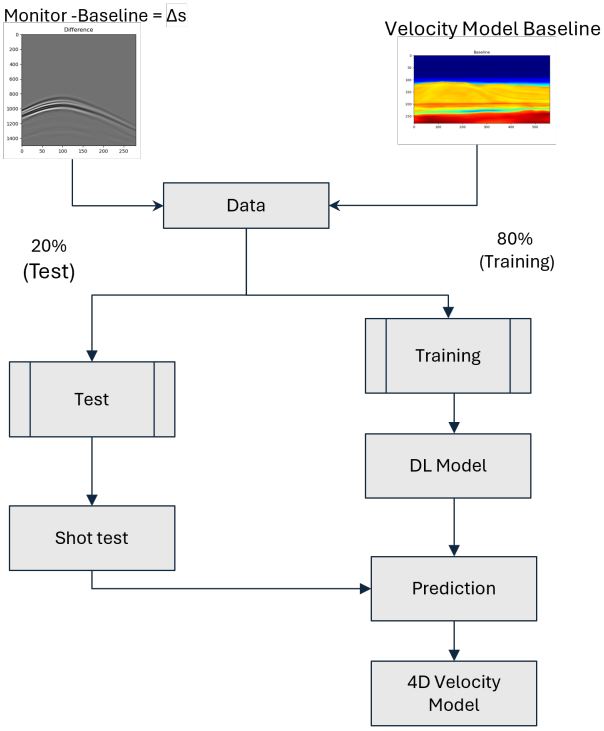

Figure 5.4: Flowchart of the convolutional neural network (CNN) architecture used for seismic inversion. The network takes seismic data differences and velocity model, processes it through a series of convolutional and pooling layers to extract hierarchical features, and outputs a predicted subsurface velocity model. The model is trained in a supervised learning framework using synthetic seismic–velocity pairs

.

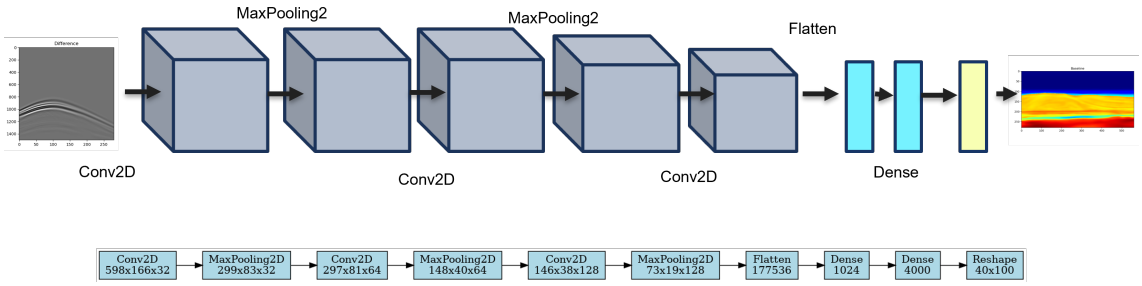

Figure 5.5: Architecture of the convolutional neural network (CNN) used for seismic inversion.

5.1.4 DDFWI Implementation

We applied conventional 4D Full Waveform Inversion using the double-difference (DDFWI) approach to the same synthetic scenarios for comparison. FWI is a physics-based inversion method that iteratively updates the velocity model by minimizing the misfit between observed and simulated seismic data.

In each case, we used the baseline velocity model as the initial model and treated the

Algoritmo 5: Time-lapse difference normalization

Require: Baseline seismogram `baseseism`, monitor seismograms `monitorseisms`, crop indices `irw_min`, `irw_max`

Ensure: Normalized cropped time-lapse differences `outseisms`

1: **for** $i = 0$ to $Nmonitors - 1$ **do**
2: $tldiff \leftarrow monitorseisms[i] - baseseism$
3: $crop \leftarrow tldiff[irw\_min : irw\_max, :]$
4: $outseisms[i] \leftarrow crop / \max(crop)$
5: **end for**

monitor seismic dataset as the observation to be inverted. We performed the inversion using acoustic forward modeling based on the wave equation, following the same principles used during synthetic data generation. Through an iterative optimization procedure, we adjusted the velocity values to reduce the difference between the recorded and simulated wavefields.

The results obtained from FWI-DD serve as a benchmark to evaluate the quality, resolution, and generalization capability of the CNN-based inversion. The inversion was performed using information from nine shot gathers.

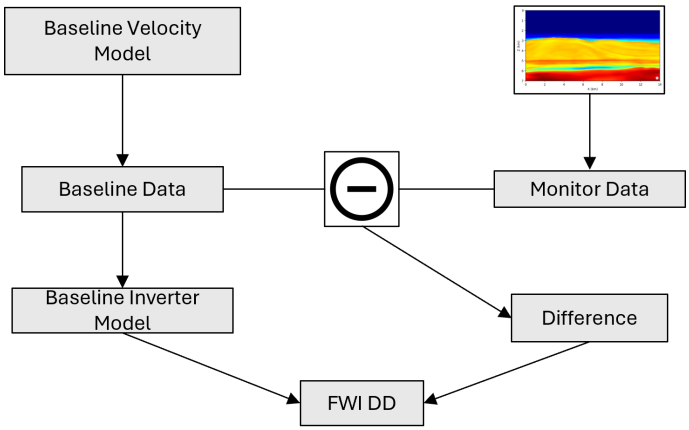

Figure 5.6: Flowchart of the proposed Deep Domain Adaptation Full Waveform Inversion (DDFWI) workflow.

5.2 Results

In this section, we present the results of the experiments conducted under the two different scenarios. The following figure 5.7 shows the inversion result for a single position, corresponding to the central shot position from the entire modeling domain. The figure on the left represents the initial velocity model for the first scenario, where the reservoir zone containing the designed anomaly is highlighted in a black box window of 40 × 100 grid points, which corresponds to the input area used for training the network. The figure on

the right shows the setup for the second scenario, where two anomaly regions are marked with black boxes: the hardening anomaly on the right and the softening anomaly on the left. Each anomaly was designed independently, but the input to the network covers the whole region containing both anomalies, corresponding to a window of 40 × 280 grid points.

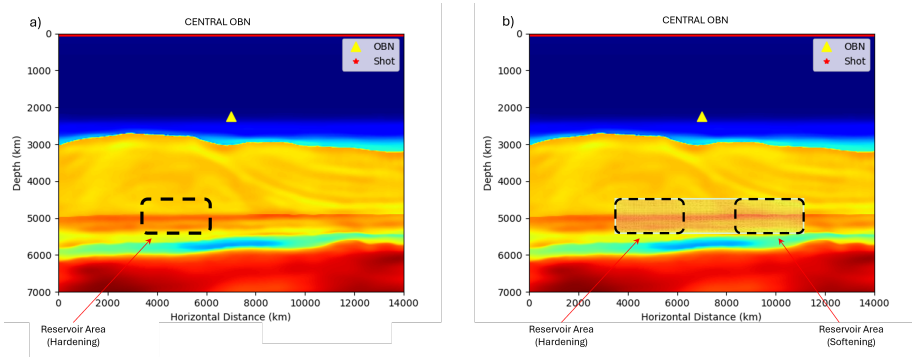

Figure 5.7: Initial velocity models and anomaly design for the two experimental scenarios. (a) The initial model for Scenario 1 shows the reservoir region where a single hardening anomaly was introduced. The anomaly is located within a 40 × 100 grid window, which serves as the input domain for the CNN. (b) The initial model for Scenario 2 includes two independent anomalies: a hardening anomaly on the right and a softening anomaly on the left. Both are highlighted with black boxes. The total input window used for training in this case spans 40 × 280 grid points, encompassing both anomalies.

5.2.1 First scenario: single anomaly

In this first scenario, a realistic anomaly was designed within the hardening reservoir area, according to the parameters described in 5.1.1. Based on this setup, 1000 variations of the anomaly were generated, serving as the input dataset for training the neural network. In the following figure, two images are shown using the same spatial and color scales: on the left, the true anomaly, and on the right, the result obtained after the network prediction.

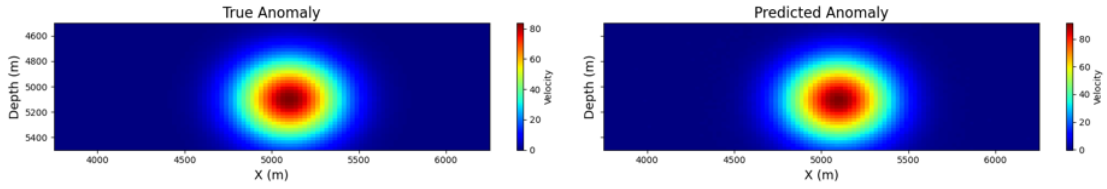

Figure 5.8: Comparison between the True Anomaly and predicted time-lapse anomaly. The left panel shows the true Gaussian-shaped velocity perturbation used in the simulation, while the right panel displays the corresponding prediction obtained using the DL-based inversion.

Figure 5.9 shows the training behavior and validation loss curves for Shot 5, measured using the mean squared error (MSE), demonstrating effective learning behavior across approximately 300 epochs. Initially, both training and validation losses decrease rapidly during the first 100 epochs, indicating that the model quickly captures the primary features of the data. From epoch 100 to around 240, the learning rate slows down, but both curves continue to decrease with reasonable stability, showing no signs of overfitting. A sharp drop in training loss is observed around epoch 250, which is not fully reflected in the validation loss, suggesting the beginning of slight overfitting. After this point, the training loss continues to decrease and stabilizes near 0.01, while the validation loss levels off around 0.03–0.05. This behavior suggests that the model has reached convergence and is beginning to specialize on the training data. Despite minor fluctuations in the validation curve, the overall trend remains consistent, and the model generalizes reasonably well, see figure 5.9. Early stopping around epoch 240 could be considered to preserve generalization. The sudden drop may indicate a learning rate adjustment or optimizer behavior. Overall, the CNN demonstrates robust performance in reconstructing velocity perturbations for this central shot.

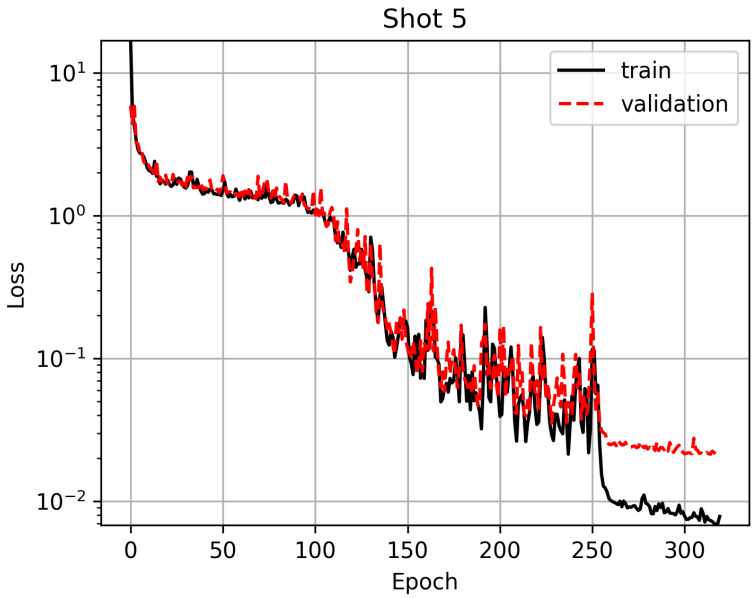

Figure 5.9: Training and validation loss curves for the CNN inversion model. The loss steadily decreases across epochs, indicating successful learning. The validation loss stabilizes after approximately 250 epochs, suggesting convergence and no significant overfitting. The use of early stopping around this point helped avoid unnecessary training and preserved model generalization.

To obtain meaningful results from the FWI process, it was necessary to invert the entire modeled dataset. This is because the resolution of FWI is highly dependent on the amount and completeness of the information provided. A higher data volume improves the inversion's ability to reconstruct fine-scale velocity variations. The result shown below

corresponds to 100 iterations of FWI, carried out using the same propagation parameters used in the data modeling stage.

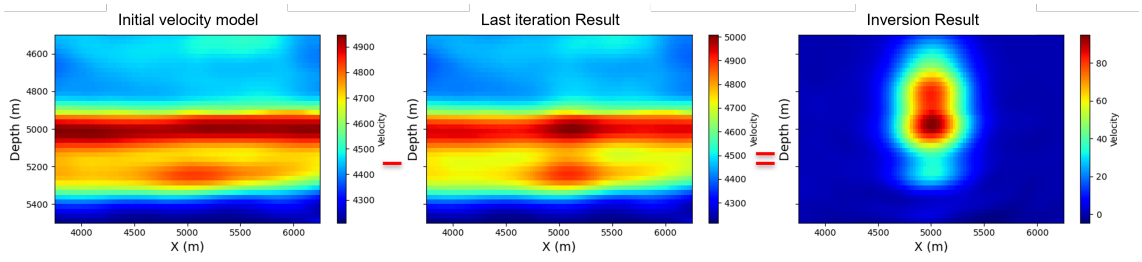

Figure 5.10: Results of the DDFWI for whole propagation (9 shots). From left to right: the Initial velocity model, the last iteration result, and their difference, which highlights temporal anomalies.

One of the main objectives of this work is not only to present inversion results but also to analyze and compare, at least qualitatively, the outputs of the deep learning-based inversion and the conventional full-waveform inversion (FWI) method. This comparison aims to highlight the potential advantages of using AI-based computational methods. It is important to emphasize that the deep learning inversion result under analysis was generated using data from a single-shot position. In contrast, the FWI result was obtained using data from all nine shot positions designed during the synthetic data generation phase. Additionally, we recall that the velocity anomalies were synthetically introduced with variations of approximately (50 m/s, 100 m/s). The following figure presents a side-by-side comparison of three images: the true velocity model, the predicted model obtained from our deep neural network, and the FWI benchmark result. All three images are displayed using the same color scale to allow a direct and meaningful comparison under consistent visual and quantitative criteria.

Figure 5.11 presented below encapsulates the core contribution of this work: demonstrating that deep learning–based seismic inversion can provide competitive results to traditional Full Waveform Inversion (FWI), even under more constrained data conditions.

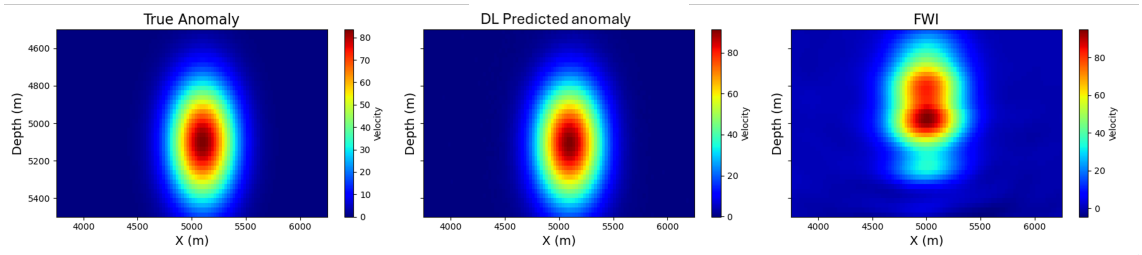

Figure 5.11: Comparison of the true velocity anomaly (left), the prediction from the deep learning-based inversion using a single shot (center), and the FWI result using nine shots (right). All results are shown using the same color scale.

On the left, the true velocity anomaly acts as the reference model, showing a clean, compact perturbation that was synthetically introduced with a velocity contrast of ap-

proximately 50–100 m/s. This target structure was designed as a benchmark for testing inversion performance.

The central figure shows the prediction from the deep convolutional neural network trained using data from a single-shot position only. Remarkably, the predicted anomaly exhibits high spatial fidelity, correctly identifying the lateral position, depth range, and approximate amplitude of the true anomaly. The neural network achieves this using only a fraction of the data and without access to the whole acquisition geometry used during data modeling. This result highlights the generalization capability and efficiency of the AI-based approach. The output is smooth yet well-resolved and free from artefacts, indicating a stable learning process and strong prior modeling encoded during training. The integration of in training contributes to better generalization and more reliable inference, even without exhaustive iteration.

The result on the right corresponds to the output from FWI. We obtained it after 100 iterations using a conventional iterative inversion process that incorporated information from all nine shot positions designed in the synthetic acquisition. As expected, the FWI result resolves the anomaly with relatively high fidelity. However, it also exhibits vertical smearing and low-frequency artifacts below the target. These issues are commonly observed in FWI and are typically associated with local minima, limited data bandwidth, or insufficient inversion regularization. During the execution of the FWI, the gradient and the evolution of the cost function across iterations were carefully analyzed. Notably, while FWI benefits from richer data, it incurs significantly higher computational costs and greater sensitivity to the initial model and acquisition parameters.

This comparative analysis leads to several important conclusions:

- The DL inversion approach is capable of producing high-quality predictions with drastically reduced input data, making it attractive for scenarios with sparse acquisition or real-time processing constraints.
- While FWI delivers accurate reconstructions with complete datasets, it remains computationally expensive and more sensitive to inversion artefacts.
- The proposed AI-based method generalizes well to unseen test data and shows strong promise as a fast, flexible alternative or complement to conventional inversion in practical applications.

Overall, this study shows that DL can perform robust, low-data seismic inversion, paving the way for more scalable, efficient reservoir monitoring workflows in real-world 4D seismic scenarios.

5.2.2 Second scenario with two anomalies

In the second scenario, we designed two distinct anomalies within the reservoir zone to simulate contrasting subsurface changes. Specifically, we introduced a positive anomaly representing a hardening effect on the right side of the reservoir, and a negative anomaly representing a softening effect on the left. We generated each anomaly independently using its own set of random Gaussian perturbations to reflect realistic variations in subsurface properties. This setup aimed to test the neural network's ability to simultaneously

detect and distinguish multiple types of anomalies within a single input domain. For training, we provided the CNN with a 40×280 grid window that fully encompassed both anomaly regions, enabling the model to learn the complex spatial patterns associated with each type of change.

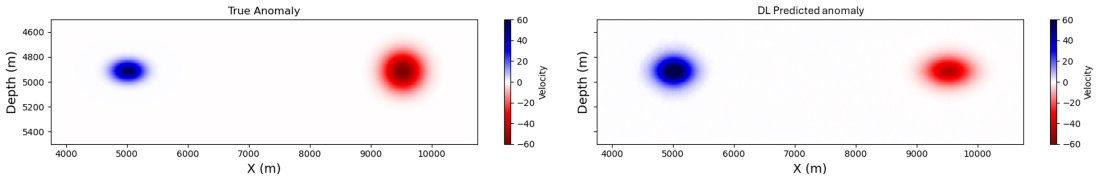

Figure 5.12: Comparison between the True Anomaly and predicted time-lapse anomaly. The left panel shows the true Gaussian-shaped velocity perturbation used in the simulation, while the right panel displays the corresponding prediction obtained using the DL-based inversion.

The following figure 5.13 presents a set of prediction results corresponding to different percentiles based on the mean squared error (MSE) between the predicted and true velocity models. These samples represent the [25, 50, 75] percentiles of the MSE distribution, from left to right. The figure illustrates how the quality of the predictions varies across the error spectrum, from the lowest error (left) to the highest error (right), as measured by MSE.

The following figure 5.14 presents the result obtained in the third step of the proposed methodology, corresponding to the DDFWI. This result integrates information from all nine shot gathers, allowing for a more robust reconstruction of the velocity model and providing a benchmark to assess the effectiveness of the deep learning-based initialization in comparison to conventional FWI procedures.

This final experiment provides a conclusive demonstration of the potential of deep learning–based seismic inversion in recovering complex subsurface structures. The figure presented compares the true model containing two contrasting velocity anomalies (one positive and one negative), the result from the neural network trained with data from a single shot, and the conventional Full Waveform Inversion (FWI) result using data from nine shot positions.

The left figure includes two anomalies that are symmetric in spatial extent but opposite in contrast, representing realistic scenarios of velocity increase and decrease within the reservoir zone. This setup was deliberately chosen to test the sensitivity and resolution capacities of both inversion methods.

The central figure captures both anomalies with high spatial accuracy. Although slightly smoothed, the predicted structures were well centered with the correct polarity and depth. Notably, this level of reconstruction was achieved using seismic data from only a single shot. This highlights the model's generalization power and the efficiency gained from learning strong prior knowledge during training. The network was trained with a maximum of 20,000 epochs but included early stopping to avoid overfitting, further contributing to its robustness.

The FWI result computed with full access to data from nine shots also identifies the

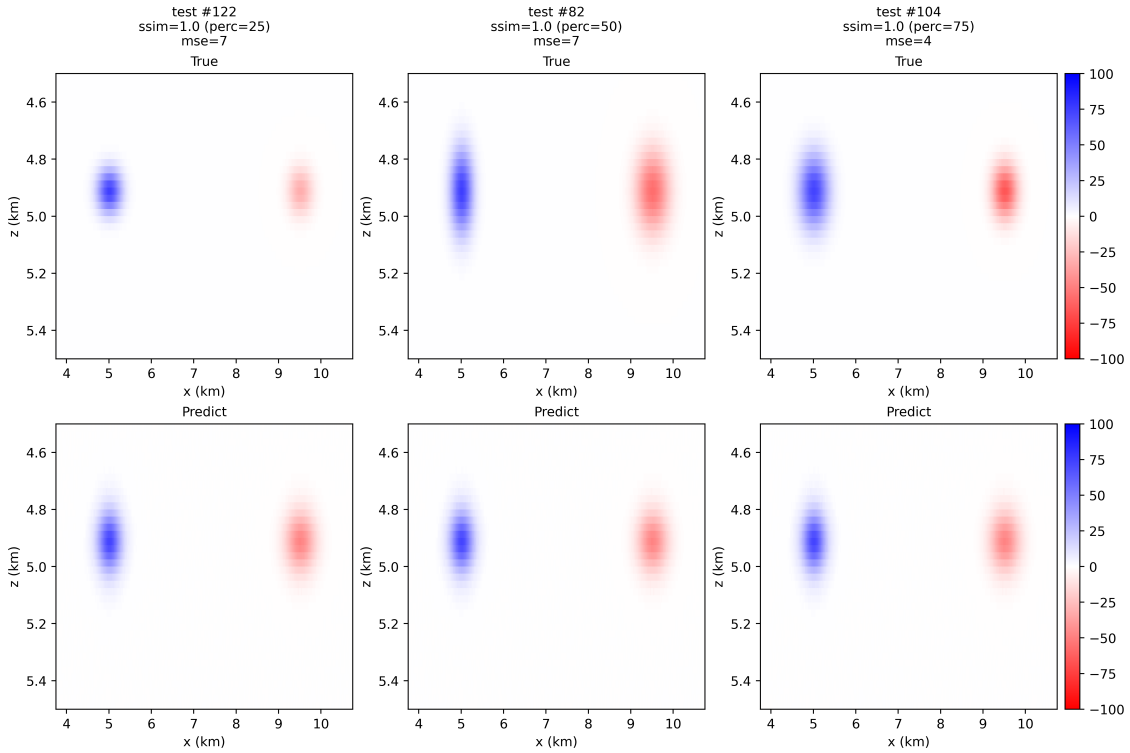

Figure 5.13: Predicted velocity models corresponding to the 25th, 50th, and 75th percentiles of the mean squared error (MSE) distribution. From left to right, the predictions illustrate increasing error relative to the ground truth, highlighting the variability in the network's performance across different samples.

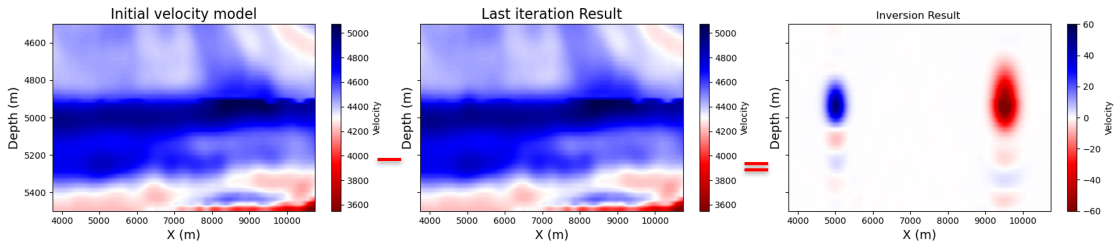

Figure 5.14: Results of the DDFWI. From left to right: the baseline velocity model, the monitor velocity model, and their difference, which highlights temporal anomalies.

two anomalies. Still, it introduces visible artifacts around them, especially in the form of vertical smearing and secondary oscillations. These artifacts are a well-known limitation of conventional inversion when dealing with limited frequency content or when local minima are present in the solution space.

This comparison reveals a clear trade-off: while FWI benefits from a more complete acquisition geometry, it remains computationally expensive and prone to instability. In contrast, the DL model, even when trained with fewer data, provides interpretable and geologically plausible outputs with minimal artifacts and significantly lower computational

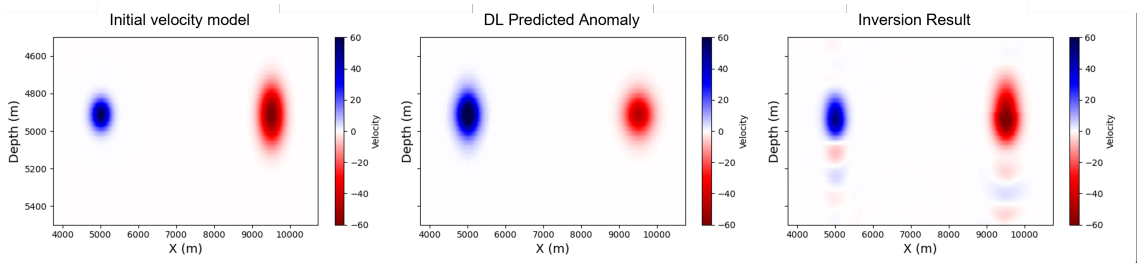

Figure 5.15: Comparison of the true velocity anomaly (left), the prediction from the deep learning-based inversion using a single shot (center), and the FWI result using nine shots (right). All results are shown using the same color scale.

costs.

In conclusion, this chapter demonstrated that DL can successfully perform seismic inversion for both single and multiple anomalies, recovering high-resolution subsurface information from sparse data. The results highlight the potential of AI-based methods to complement or even replace traditional inversion workflows in scenarios where acquisition is limited, computational cost is a concern, or rapid reservoir monitoring is required.

5.2.3 Time resources

To gain a comprehensive understanding of the computational requirements associated with the various stages of the proposed workflow, Table 5.1 presents a summary of the execution times and hardware resources necessary for each process. This breakdown encompasses propagation, network training, and conventional full-waveform inversion (FWI), facilitating a direct comparison of performance and scalability across these methods.

Process	Resources	Execution Time	Observations
Synthetic Data generation	CPU (Ogbon)	12.5 min.	Each shot
DL inversion	GPU (Ogbon)	4 min	Each shot
DDFWI	CPU (Local)	4.5 hours	9 shots / t 8 / 100 iterations

Table 5.1: Execution time and computational resources for each methodology stage.

5.2.4 Technical Overview of the DL based Inversion

The CNN was trained in a supervised fashion, using a large dataset generated through 2D acoustic forward modeling implemented in Julia, ensuring that each input seismic difference corresponded precisely to its velocity model. The model training and evaluation were carried out in Python using TensorFlow and Keras, which allowed modular design and GPU acceleration. NumPy was employed for data handling and normalization, ensuring consistency with preprocessing steps. Visualization of results and monitoring of training metrics were managed using Matplotlib.

The following table summarizes the core aspects of the deep learning-based inversion approach designed to reconstruct 2D velocity models from synthetic seismic data. This stage of the workflow uses a supervised convolutional neural network (CNN) to map time-lapse seismogram differences into subsurface velocity distributions. Key details such as model architecture, input and output formats, training configuration, memory and GPU usage, and software tools are outlined below. The inversion model plays a central role in estimating physical property changes in the reservoir area and serves as a benchmark for evaluating the effectiveness of learning-based inversion strategies.

Aspect	**Inversion**
Objective	Inversion DL-based
Model Architecture	Convolutional Neural Network (CNN)
Input Data	Seismogram difference (Monitor–Baseline), reservoir area
Target Output	2D velocity model
Training Data	Synthetic seismogram–velocity model pairs
Learning Type	Supervised
GPU Usage	Required
Training Time	$\sim$ 11 min. / each shot
Inference Time	$\sim$ 10 seconds per inversion case
Main Libraries	TensorFlow, Keras, NumPy, Matplotlib
Preprocessing	Amplitude differencing and velocity map labeling
Application Role	Acts as Inversion tool comparable with a benchmark

Table 5.2: Summary of the Inversion implementation

Chapter 6

Conclusions

The proposed data matching workflow, which includes splitting each seismic shot gather into overlapping windows and applying per-window normalization, has proven effective in preparing the input data for neural network training. While it could be argued that normalization should be applied globally before windowing, the local normalization approach offers clear advantages: it preserves relative amplitude dynamics within each window and allows the model to learn in a scale-consistent manner, particularly beneficial for convolutional neural networks operating on local spatial features.

Empirically, the use of k-fold cross-validation showed consistent training stability and an apparent reduction in test loss across folds, confirming that this normalization strategy does not hinder generalization but instead supports it by standardizing input domains across varying shot gathers.

This approach aligns well with practices in related fields such as medical imaging and geophysical inversion, where patch-wise normalization is commonly applied to manage varying amplitude scales across large datasets. Thus, the proposed preprocessing strategy is both technically sound and methodologically justified.

However, results also indicate that while the model can fit the reference data structure, the predicted amplitudes tend to be significantly overestimated. This issue negatively impacts the precision of the matching process and highlights the need for further refinement of both the training data and model constraints. A logical next step is to focus the training set on cases where the initial RNMS mismatch is below 10%, allowing for a more focused evaluation of the method's reliability under realistic conditions.

The convolutional neural network (CNN)-based inversion method demonstrates strong potential for retrieving subsurface velocity models from seismic data. In the synthetic experiments conducted, the model was able to reconstruct velocity profiles that preserved key geological features, particularly within the reservoir zone. Quantitative metrics confirmed the visual quality of the inverted images, although the recovered amplitude values still deviated from expected physical scales.

One of the strengths of this framework is its computational efficiency. Once trained, the model is capable of producing high-resolution velocity predictions in a fraction of the time required by traditional full-waveform inversion methods. These performance characteristics make it particularly attractive for applications where approximate models are acceptable or where prior reservoir knowledge is available, such as in 4D seismic monitoring scenarios.

Nonetheless, residual 4D noise in the predicted outputs and the amplitude mismatches suggest that further optimization is needed. Although manual hyperparameter tuning provided a reasonable balance during training, incorporating automated or data-driven optimization strategies could improve convergence and model robustness, particularly across diverse geologies and acquisition setups.

To enhance the generalization capacity of the model, future work should expand the training set to include a broader range of velocity anomalies, acquisition geometries, and noise conditions. Moreover, exploring more sophisticated network architectures, potentially with attention mechanisms or multi-scale feature extraction, may lead to improved inversion fidelity.

Overall, this work establishes a solid foundation for CNN-based seismic inversion and its integration with data-driven preprocessing strategies. It demonstrates that, with appropriate design and training protocols, DL can serve as a viable and scalable complement to conventional inversion techniques in seismic exploration.

Bibliography

Alali, A. E., V. Kazei, B. Sun, R. Smith, P. Nivlet, A. Bakulin & T. Alkhalifah (2020), Cross-equalization of time-lapse seismic data using recurrent neural networks, *em* 'SEG Technical Program Expanded Abstracts 2020'.

Alali, A., R. Smith, P. Nivlet, A. Bakulin & T. Alkhalifah (2021), Time-lapse seismic cross-equalization using temporal convolutional networks, *em* 'SEG/AAPG/SEPM First International Meeting for Applied Geoscience and Energy'.

Alali, A., V. Kazei, B. Altaf, X. Zhang & T. Alkhalifah (2020), 'Time-lapse cross-equalization by deep learning', *King Abdullah University of Science and Technology Repository* .

Alali, A., V. Kazei, B. Sun & T. Alkhalifah (2022*a*), 'Time-lapse data matching using a recurrent neural network approach', *Geophysics* **87**, V405–V417.

Alali, A., V. Kazei, B. Sun & T. Alkhalifah (2022*b*), 'Time-lapse data matching using a recurrent neural network approach', *Geophysics* **87**(5), V405–V417.

Alarsan, F. & M. Younes (2020), 'Best selection of generative adversarial networks hyper-parameters using genetic algorithm', Research Square Preprint.

Aldakheel, M., R. Pevzner, B. Gurevich & S. Glubokovskikh (2021), 'Seismic characterization of co2 storage driven by time-lapse images of a prior injection using the artificial neural network', *Interpretation* **9**(3), T911–T925.

Anjom, F. K., F. Vaccarino & L. V. Socco (2023), 'Machine learning for seismic exploration: Where are we and how far are we from the holy grail?', *Geophysics* **89**, WA157–WA178.

Araujo, R. C. F., G. Corso, H. A. D. Nascimento, S. X. Souza, J. M. Araujo & T. Barros (2024), 'Estimation of water velocity and receiver position changes in time-lapse seismic using machine learning', *Brazilian Journal of Geophysics* **42**(2).
URL: *https://sbgf.org.br/revista/index.php/rbgf/article/view/2313*

Araya-Polo, M., J. Jennings, A. Adler & T. Dahlke (2018), 'Deep-learning tomography', *The Leading Edge* **37**(1), 58–66.

Arjovsky, M., S. Chintala & L. Bottou (2017), 'Wasserstein gan'.

Asnaashari, Amir, Romain Brossier, Stéphane Garambois, Francois Audebert, Pierre Thore & Jean Virieux (2014), 'Time-lapse seismic imaging using regularized full-waveform inversion with a prior model: Which strategy?', *Geophysical Prospecting* **63**.

Bai, S., J. Z. Kolter & V. Koltun (2018), 'An empirical evaluation of generic convolutional and recurrent networks for sequence modeling'.

Bishop, C. M. (1995), *Neural Networks for Pattern Recognition*, Oxford University Press.

Bishop, C. M. (2006), *Pattern recognition and machine learning*, Springer, New York.

Cao, J. & B. Roy (2017), 'Time-lapse reservoir property change estimation from seismic using machine learning', *The Leading Edge* **36**(3), 234–238.

Chai, T. & R. R. Draxler (2014), 'Root mean square error (rmse) or mean absolute error (mae)? – arguments against avoiding rmse in the literature', *Geoscientific Model Development* **7**(3), 1247–1250.
URL: *https://gmd.copernicus.org/articles/7/1247/2014/*

Corte, G. & C. MacBeth (2021), A physics-based loss function to constrain neural network inversion of 4d seismic data, *em* '82nd EAGE Annual Conference & Exhibition', pp. 1–5.

Corte, G., J. Dramsch, H. Amini & C. MacBeth (2021), Keynote 5: Informing neural networks with fluid flow consistent property correlations: A 4d seismic inversion application, *em* 'EAGE Conference and Exhibition 2021', pp. 1–5.

Cypriano, L., Z. Yu, D. Ferreira, B. Huard, R. Pereira, F. Jouno, A. Khalil, E. Urasaki, N. Cruz, A. Yin, D. Clarke & C. C. Jesus (2019), Obn for pre-salt imaging and reservoir monitoring–potential and road ahead, *em* '16th International Congress of the Brazilian Geophysical Society', Vol. 318, Rio de Janeiro, Brazil.

da Silva, D., E. Fagua Duarte, W. Almeida, M. Ferreira, F. A. Moura & J. M. Araujo (2021), 'Target-oriented inversion using the patched green's function method', *Geophysics* **86**(6), R811–R823.

Dramsch, J. S., A. N. Christensen, C. MacBeth & M. Lüthje (2022), 'Deep unsupervised 4-d seismic 3-d time-shift estimation with convolutional neural networks', *IEEE Transactions on Geoscience and Remote Sensing* **60**, 1–16.

Dramsch, J. S., G. Corte, H. Amini, M. Lüthje & C. MacBeth (2019), Deep learning application for 4d pressure saturation inversion compared to bayesian inversion on north sea data, *em* 'Second EAGE Workshop Practical Reservoir Monitoring 2019', pp. 1–5.

Duan, Y., S. Yuan, P. Hatchell, J. Vila & K. Wang (2020), Estimation of time-lapse timeshifts using machine learning, *em* 'SEG Technical Program Expanded Abstracts 2020', pp. 3724–3728.

Fagua, E. (2021), 'Métodos otimizados para inversão completa das formas de onda em 3d', Online. Available at: `https://bdtd.ibict.br/vufind/Record/UFRN_a1308eebdc77e6389d7227541219fdf2` (Accessed: 2025-06-04).

Géron, A. (2022), *Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow*, 3ª edição, O'Reilly Media.

Gholamy, Afshin & Vladik Kreinovich (2014), Why ricker wavelets are successful in processing seismic data: Towards a theoretical explanation, *em* '2014 IEEE Symposium on Computational Intelligence for Engineering Solutions (CIES)', pp. 11–16.

Gonzalez, J. C., C. A. N. Da Costa, D. Pinheiro, K. Rincon, M. G. Gebre, J. M. de Araújo & J. Lopez (2023), 'Seismic data interpolation with an iterative workflow and generative adversarial networks', *First Break* **41**(1), 1–5.

Goodfellow, I., J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville & Y. Bengio (2014), 'Generative adversarial networks'.

Goodfellow, I., Y. Bengio & A. Courville (2016), *Deep learning*, MIT Press, Cambridge, MA.

Gui, J., Z. Sun, Y. Wen, D. Tao & J. Ye (2023), 'A review on generative adversarial networks: Algorithms, theory, and applications', *IEEE Transactions on Knowledge and Data Engineering* **35**(4), 3313–3332.

Hale, D. (2013), 'Dynamic warping of seismic images', *Geophysics* **78**(2), S105–S115.

Horé, Alain & Djemel Ziou (2010), Image quality metrics: Psnr vs. ssim, *em* '2010 20th International Conference on Pattern Recognition', pp. 2366–2369.

Huynh-Thu, Q. & M. Ghanbari (2008), 'Scope of validity of psnr in image/video quality assessment', *Electronics Letters* **44**, 800.

Ioffe, S. & C. Szegedy (2015), 'Batch normalization: Accelerating deep network training by reducing internal covariate shift'.

Isola, Phillip, Jun-Yan Zhu, Tinghui Zhou & Alexei A. Efros (2018), 'Image-to-image translation with conditional adversarial networks'.
URL: *https://arxiv.org/abs/1611.07004*

Johnston, D. H. (2013), *Practical Applications of Time-Lapse Seismic Data*, Society of Exploration Geophysicists.

Jun, H. & Y. Cho (2021), 'Repeatability enhancement of time-lapse seismic data via a convolutional autoencoder', *Geophysical Journal International* **228**(2), 1150–1170.

Kaur, H., N. Pham & S. Fomel (2020), 'Seismic data interpolation using deep learning with generative adversarial networks', *Geophysical Prospecting* **69**.

Kaur, H., Z. Zhong, A. Sun & S. Fomel (2022), 'Time-lapse seismic data inversion for estimating reservoir parameters using deep learning', *Interpretation* **10**(1), T167–T179.

Kim, S., J. Park, S. Seol & J. Byun (2023), 'Machine learning-based time-lapse 1d seismic full-waveform inversion with efficient training data generation in a carbon capture and storage monitoring', SSRN Preprint.

Kingma, D. & J. Ba (2014), Adam: A method for stochastic optimization, *em* 'International Conference on Learning Representations'.

Kragh, E. & P. Christie (2002), 'Seismic repeatability, normalized rms, and predictability', *The Leading Edge* **21**, 640–647.

Krizhevsky, A., I. Sutskever & G. E. Hinton (2012), 'Imagenet classification with deep convolutional neural networks', *Communications of the ACM* **60**, 84–90.

Lecerf, D. & M. Besselievre (2018), 'A new approach to compensate for illumination differences in 4d surveys with different individual acquisition geometries', *First Break* **36**(2), 71–76.

Lecun, Y., L. Bottou, Y. Bengio & P. Haffner (1998), 'Gradient-based learning applied to document recognition', *Proceedings of the IEEE* **86**(11), 2278–2324.

Lee, S., J. Won & H. Jun (2024), 'Cross-equalization for time-lapse sparker seismic data', *Geophysical Prospecting* .

Li, Y., T. Alkhalifah & Q. Guo (2021), 'Target-oriented time-lapse waveform inversion using deep learning-assisted regularization', *Geophysics* **86**(4), R485–R495.

Lopez, J., F. Neto, M. Cabrera, S. Cooke, S. Grandi & D. Roehl (2020), Refraction seismic for pre-salt reservoir characterization and monitoring, *em* 'SEG Technical Program Expanded Abstracts 2020', pp. 2365–2369.

Lumley, D., D. C. Adams, M. Meadows, S. Cole & R. Wright (2003), 4d seismic data processing issues and examples, *em* 'SEG Technical Program Expanded Abstracts 2003', pp. 1394–1397.

Lumley, D. E. (2001*a*), 'Time-lapse seismic reservoir monitoring', *Geophysics* **66**(1), 50–53.

Lumley, D. E. (2001*b*), 'Time-lapse seismic reservoir monitoring', *Geophysics* **66**(1), 50–53.

López, J. L., F. Neto, M. Cabrera, S. Cooke, S. Grandi & D. Roehl (2020), Refraction seismic for pre-salt reservoir characterization and monitoring, *em* 'SEG Technical Program Expanded Abstracts 2020'.

Mac Dowell, N., P. Fennell, N. Shah & G. Maitland (2017), 'The role of co2 capture and utilization in mitigating climate change', *Nature Climate Change* **7**, 243–249.

McCulloch, W. S. & W. Pitts (1943), 'A logical calculus of the ideas immanent in nervous activity', *The Bulletin of Mathematical Biophysics* **5**, 115–133.

Mirza, M. & S. Osindero (2014), 'Conditional generative adversarial nets'.

Nguyen, P. K. T., M. J. Nam & C. Park (2015*a*), 'A review on time-lapse seismic data processing and interpretation', *Geosciences Journal* **19**, 375–392.

Nguyen, P. K. T., M. J. Nam & C. Park (2015*b*), 'A review on time-lapse seismic data processing and interpretation', *Geosciences Journal* **19**(2), 375–392.

Pinheiro, D. N., J. C. Gonzalez, G. Corso, M. G. Gebre, C. A. N. da Costa, S. Xavier-de Souza & T. Barros (2024), 'Hyperparameter determination for gan-based seismic interpolator with variable neighborhood search', *Computers & Geosciences* **192**, 105689.

Plotnitskii, P., O. Ovcharenko, V. Kazei, D. Peter & T. Alkhalifah (2022), Dl-fused elastic fwi: Application to marine streamer data, *em* 'Second International Meeting for Applied Geoscience and Energy', pp. 1945–1950.

Radford, A., L. Metz & S. Chintala (2016*a*), 'Unsupervised representation learning with deep convolutional generative adversarial networks'.

Radford, A., L. Metz & S. Chintala (2016*b*), 'Unsupervised representation learning with deep convolutional generative adversarial networks'.

Rickett, J. & D. E. Lumley (1998), A cross-equalization processing flow for off-the-shelf 4d seismic data, *em* 'SEG Technical Program Expanded Abstracts 1998', pp. 16–19.

Rickett, J. E. & D. E. Lumley (2001), 'Cross-equalization data processing for time-lapse seismic reservoir monitoring: A case study from the gulf of mexico', *Geophysics* **66**(4), 1015–1025.

Rollmann, K., A. Soriano-Vargas, F. Almeida, A. Davolio, D. J. Schiozer & A. Rocha (2022), 'Convolutional neural network formulation to compare 4-d seismic and reservoir simulation models', *IEEE Transactions on Systems, Man, and Cybernetics: Systems* **52**(5), 3052–3065.

Ronneberger, O., P. Fischer & T. Brox (2015), U-net: Convolutional networks for biomedical image segmentation, *em* N.Navab, J.Hornegger, W. M.Wells & A. F.Frangi, eds., 'Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015', Springer, Cham, pp. 234–241.

Rosenblatt, F. (1958), 'The perceptron: A probabilistic model for information storage and organization in the brain', *Psychological Review* **65**(6), 386–408.

Salimans, T., I. Goodfellow, W. Zaremba, V. Cheung, A. Radford & X. Chen (2016), 'Improved techniques for training gans'.

Samuel, A. L. (1959), 'Some studies in machine learning using the game of checkers', *IBM Journal of Research and Development* **3**(3), 210–229.

Scopus, Elsevier (2024), 'Portal periódicos capes'. Available at: `https://www-scopus-com.ez18.periodicos.capes.gov.br` (Accessed: 2024-10-22).

Shinde, P. P. & S. Shah (2018), A review of machine learning and deep learning applications, *em* '2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)', IEEE, pp. 1–6.

Waage, M., S. Bünz, M. Landrø, A. Plaza-Faverola & K. A. Waghorn (2019), 'Repeatability of high-resolution 3d seismic data', *Geophysics* **84**, B75–B94.

Wang, Y. (2014), 'Seismic ray tracing in anisotropic media: A modified newton algorithm for solving highly nonlinear systems', *Geophysics* **79**(1), T1–T7.

Wang, Y. (2016), *Seismic Inversion: Theory and Applications*, Wiley Blackwell.

Wang, Z., A. C. Bovik, H. R. Sheikh & E. P. Simoncelli (2004*a*), 'Image quality assessment: From error visibility to structural similarity', *IEEE Transactions on Image Processing* **13**(4), 600–612.

Wang, Zhou, A.C. Bovik, H.R. Sheikh & E.P. Simoncelli (2004*b*), 'Image quality assessment: from error visibility to structural similarity', *IEEE Transactions on Image Processing* **13**(4), 600–612.

Willmott, C. & K Matsuura (2005), 'Advantages of the mean absolute error (mae) over the root mean square error (rmse) in assessing average model performance', *Climate Research* **30**, 79.

Wu, Y., G. A. McMechan & Y. Wang (2022), 'Adaptive feedback convolutional-neural-network-based high-resolution reflection-waveform inversion', *Journal of Geophysical Research: Solid Earth* **127**(6), e2022JB024138.

Xue, Y., M. Araujo, J. Lopez, K. Wang & G. Kumar (2019), 'Machine learning to reduce cycle time for time-lapse seismic data assimilation into reservoir management', *Interpretation* **7**(3), SE123–SE130.

Yang, F. & J. Ma (2019), 'Deep-learning inversion: A next-generation seismic velocity model building method', *Geophysics* **84**(4), R583–R599.

Yilmaz, O. (2001*a*), *Seismic data analysis*, SEG Books, Tulsa, OK.

Yilmaz, Ö. (2001*b*), *Seismic Data Analysis: Processing, Inversion, and Interpretation of Seismic Data*, Society of Exploration Geophysicists.

Yuan, C., X. Zhang, X. Jia & J. Zhang (2019*a*), 'Time-lapse velocity imaging via deep learning', *Geophysical Journal International* **220**(2), 1228–1241.

Yuan, C., X. Zhang, X. Jia & J. Zhang (2019*b*), 'Time-lapse velocity imaging via deep learning', *Geophysical Journal International* **220**(2), 1228–1241.

Zhang, S.-B., H.-J. Si, X.-M. Wu & S.-S. Yan (2022), 'A comparison of deep learning methods for seismic impedance inversion', *Petroleum Science* **19**(3), 1019–1030.

Zhang, X. & Y. LeCun (2018), 'Adversarially-trained normalized noisy-feature auto-encoder for text generation'.