Important Notice

2024-07-02

Machine Learning alternative to sparseness: a Radon transform application for multiple and ground roll attenuation

Lira Fontes, Paloma Helena

UNIVERSITY OF CALGARY

Machine Learning alternative to sparseness: a Radon transform application for multiple and ground roll attenuation

by

Paloma Helena Lira Fontes

A THESIS
SUBMITTED TO THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE
DEGREE OF MASTER OF SCIENCE

GRADUATE PROGRAM IN GEOSCIENCE

CALGARY, ALBERTA
JULY, 2024

Abstract

Radon transform (RT) allows mapping different seismic events using different basis functions. Integrating RTs with machine learning (ML) presents an innovative approach to addressing non-linear problems in seismic data. By incorporating deep learning (DL) techniques into the framework, the objective is to address challenges encountered in seismic data processing, particularly in separating signal from coherent noise within the model space. A central idea of this work is the utilization of ML as an alternative to traditional sparseness techniques, which can struggle with complex and overlapping seismic events. Through a pixel-by-pixel approach, ML-based approaches leverage DL's capability to discern non-linear patterns in images, enabling effective segregation of multiples and ground roll from reflections in the RT domain. This approach becomes particularly valuable in scenarios where achieving complete spatial separation is challenging, for example, in multiple or ground roll overlapping primary reflections.

This study conducts numerical experiments to assess the U-Net's effectiveness in discerning ground roll and multiples, employing various workflows to predict RT panels and maximizing RT utility by incorporating multiple channels of information like RGB colour channels in an image. Experiments examined the efficiency of different RT types, such as Hyperbolic RT (HRT) and Parabolic RT (PRT), for training a U-Net model to predict multiples. While U-Net partly succeeded in predicting multiples and highlighting the importance of label selection, it also faced challenges. Transform artifacts linked to input geometry, like truncation and sampling, hampered inference, lowering generalization. Furthermore, tests deploying a Hybrid Linear-Parabolic RT methodology for ground roll suppression in a field dataset helped analyze crosstalk problems between different RT spaces. Different channels provided insights into the leakage of ground roll among the RT used, with the three-channel approach showing promising results to forecast ground roll attenuated RT panels but challenges persisting in fully disentangling ground roll from reflection data. Continued research efforts are crucial to address these challenges and unlock the full potential of ML with seismic.

Acknowledgements

Research often means spending countless hours alone, staring at a computer screen. However, even though this thesis is a machine learning application, I could only have done it with people. In my humble opinion, the human beings around me are still the most critical resource, so thank you to all the unique individuals who were part of this journey.

My sincere thanks to my supervisor, Dr. Daniel Trad. His direction and encouragement during the steep learning curve called graduate school were fundamental for my development. Daniel's expertise, way of thinking, vision and enthusiasm for applied research will always inspire me. Being his seismic processing TA was also a privilege. As a Brazilian, it was an honour to have had an Argentinian supervisor (even during the 2022 World Cup) and to be able to call him a friend. I extend my gratitude to Dr. Kristopher Innanen, Director of the CREWES (Consortium for Research in Elastic Wave Exploration Seismology) Project, for his support and words of encouragement throughout my research journey. Under Kris's guidance, CREWES has provided exceptional research opportunities for students like myself. Further, thanks to Dr. Hersh Gilbert for being part of my examination committee and for his careful suggestions for this work. I would also like to thank the CREWES project sponsors, NSERC, and the CSEG Foundation Scholarship for their financial support during my degree.

I want to thank my family for their lifetime of support, being my safe harbour and for their unconditional love: my parents, Ivete Lira Fontes and Julio Fontes, my brothers, Caio Lira Fontes and Julio Fontes Filho, as well as my niece Maya Fontes. You were the foundation of me being the person I am today!

A special thanks to Jinji Li, David Emery, Marcelo Guarido and Kai Zhuang: this journey would not have been enjoyable without our moments together, friendship, research and life conversations. Many thanks to CREWES faculty, staff and students for their help, office presence and support (in alphabetic order): Anton Ziegon, Brian Russel, Carla Acosta, Chioma Chineke, Ivan Sanchez, Kevin Bertram, Kevin Hall, Kimberly Pike, Lukas Sadownyk, Mariana Lume, Ninoska Amundaray, Scott Hess, Shang Huang, Tianze Zhang, Ziguang Su. Thanks to the geologists in the 2022 IBA team, Thomas Kenzie and Nicole Virginillo, for a fantastic experience. Thanks to Katie Biegel, Jesus Rojas, Tais Fontes, Tom Peploe, Tayo Aleshe, and Fernando Berumen for the fun moments as a UofC student. Thanks to Patricia Gavotti, Nilanjan Ganguly, Nanna Eliuk, Austin Springer, Fuju Chen and Mark Jeroncic for all the professional guidance and pieces of advice during my Summer internships at Canacol Energy, Chevron and the CSEG mentorship program.

Moving to a new country during a global pandemic was a challenge that became easy and enjoyable with the help of essential friends I now consider family: Morgana Genn, Victoria Barbosa and Rafael Manenti. Thanks to Josmar Cristello, Amanda Lima, Anna-Marie Lewrenz and Felipe Kajimoto for all the support and good moments in Calgary. Thanks to my American family, David and Michelle Chiu, who have always encouraged and supported my plans to attend grad school abroad.

As Google Scholar advised, I have been standing on giants' shoulders.

To my parents, Ivete and Julio, who always believed in the transformative power of education

Table of Contents

List of Tables

List of Illustrations, Figures and Graphics

List of Symbols, Abbreviations and Nomenclature

Symbol or abbreviation	Definition	Comments
1D	1 dimensions in space	
2D	2 dimensions in space	
ADAM	Adaptive Moment Estimator	
AGC	Automatic Gain Control	
ANN	Artificial Neural Network	
AVO	Amplitude Variation with Offset	
BN	Batch Normalization	
CMP	Common Midpoint	
CNN	Convolutional Neural Network	
DL	Deep Learning	
FFT	Fast Fourier Transform	
F-K	Frequency wavenumber filter	
HRT	Hyperbolic Radon Transform	
ISS	Inverse Scattering Theory	
LMO	Linear Moveout	
LRT	Linear Radon Tranform	
MAE	Mean Absolute Error	
ML	Machine Learning	
MLP	Multilayer Perceptron	
MSE	Mean Squared Error	
NMO	Normal Moveout	
NN	Neural Network	
PRT	Parabolic Radon Transform	
ReLU	Rectified Linear Unit	type of activation function
RGB	Red Green Blue	
RMS	root mean square	
RMSE	Root Mean Square Error	
RT	Radon Transform	
SVD	Singular Value Decomposition	
SRME	Surface-Related Multiple Elimination	
TLU	threshold logic unit	
TanH	Tangent Hyperbolic	type of activation function

s	slowness	mainly used for HRT
p	apparent slowness	slope parameter in LRT
q	Radon parameter	mainly used as slowness for PRT
τ	intercept time	two way travel time at zero offset
t	time in seconds	
x	offset	distance between source and receiver
x_{max}	maximum offset	
x_{min}	minimum offset	
p_{min}	minimum ray parameter	
p_{max}	maximum ray parameter	
v	velocity	
v_S	stacking velocity	
v_{RMS}	RMS velocity	
f	temporal frequency in Hertz (Hz)	
w	angular frequency (radian/seconds)	
θ	angle of incidence	angle between a ray and the perpendicular to a particular reflector
$\mathbf{m}$	model	data after transformation
M	model in the frequency domain	
$\tilde{\mathbf{m}}$	reconstructed model	
$\mathbf{d}$	data	time-offset domain
D	data in the frequency domain	
$\tilde{\mathbf{d}}$	reconstructed data	
$\tilde{\mathbf{D}}$	reconstructed data in the frequency domain	
$\mathbf{z_{ref}}$	reference depth	
$\mathbf{L}$	forward operator	operator that maps model to data
$\mathbf{L^H}$	adjoint operator	operator that maps data to model
$\mathbf{L^{-1}}$	inverse operator	operator that maps data to model
$\mathbf{W_d}$	matrix of data weights	
$\mathbf{W_m}$	matrix of model weights	
dr	Dropout rate	
ρ	rho filter	
ϕ	cost function for sparse RT	
λ	trade-off parameter	between data misfit and model constraint
σ	sigmoid	type of activation function (Data Science notation)

$\hat{y}$	predicted value	Data Science notation
$\boldsymbol{\beta}$	model's parameter vector	Data Science notation
st	seismic trace	
wl	wavelet	
r	reflectivity	
δ	noise	

Epigraph

Know all the theories, master all the techniques, but as you touch a human soul be just another human soul.

Carl Jung

Chapter 1

Introduction and theoretical background

In the context of exploration geophysics, the seismic method plays an important role, mainly due to its great capacity for subsurface investigation and its high level of resolution. It can be described in three main stages: acquisition, processing, and interpretation.

Seismic acquisition involves the generation of controlled seismic waves by a specialized type of source (explosives or vibroseis, among others) that travel into the subsurface through different rock layers. These waves encounter different structures and/or rock properties, and usually, part of the energy is reflected and recorded by receivers. Specialized receivers called geophones or hydrophones (onshore or offshore acquisition, respectively) capture these reflected waves, culminating in the generation of a detailed seismic record.

Following the acquisition, seismic processing is vital to turn raw seismic data into a representative subsurface image. For this to be achieved, the seismic data is subjected to a series of operations and manipulations which follow a certain workflow depending on the type of data.

Finally, interpretation utilizes geological knowledge of the study area and concepts such as seismic stratigraphy to analyze the processed data. Through seismic reflection, exploration

geophysicists can map features associated with the accumulation of hydrocarbons, identify lithologies, and visualize geological structures in the subsurface.

The aim of processing seismic reflection data is to maintain relevant events for interpreters, such as the primary reflections while attenuating unwanted events referred to as noise. Multiples, a form of seismic noise, result from energy reflecting more than once and being recorded by the receivers. They are periodic in the slowness (reciprocal of velocity) domain and have a larger moveout than the primary reflections, which makes it possible to separate them in the transformed domain.

Ground roll is a different type of seismic noise typically found on land seismic data. It is known for its elliptical movement and is primarily composed of Rayleigh waves. Its dispersive nature further complicates its attenuation, requiring specialized techniques for effective noise attenuation. This type of noise can usually be seen in the short offsets of the shot domain, and it has this name since it is found in the form of a cone. Different from multiple, which has an approximately hyperbolic shape, the ground roll has a linear shape, which permits a different approach while separating them from the primary reflections.

1.1 Radon Transforms

Radon transform (RT) is a mathematical tool that maps different shaped curves from the data space into a transformed domain, commonly known in geophysics as $\tau - p$ transform (τ represents the transformed time and, in this case, p is the slowness (or ray parameter)) or simply RT domain. It is similar to the 2D Fourier transform, where the wavefield is decomposed into its plane wave components, each with a specific frequency and angle (Yilmaz, 2001).

There are advantages to applying this new domain in certain seismic processing steps. In noise attenuation, for example, the primaries and multiples can be distinguished due to the difference in velocity and moveout. In the case of ground roll, it is due to reflections

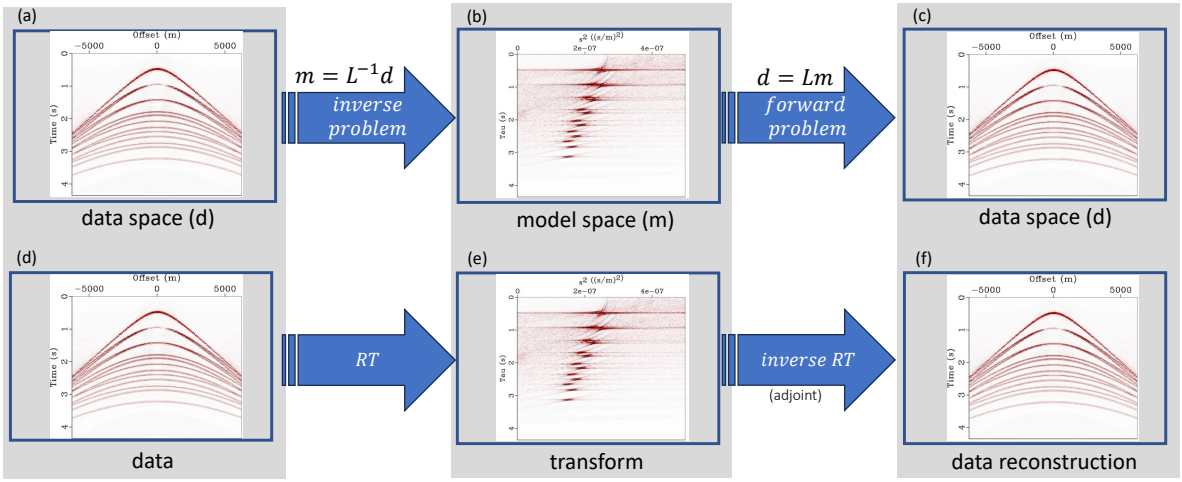


Figure 1.1: Radon Transform: hyperbolic example

having a different shape than this noise. Consequently, the RT domain can be conveniently manipulated to mute unwanted noise while keeping the primary reflections.

First introduced by Johann Radon (1917), the RT has been widely applied in different geophysics fields. Beylkin (1983), Thorson and Claerbout (1985) and Beylkin (1987) are some of the first publications that showcased an RT application in seismic processing. It was also applied in inversion (Thorson and Claerbout, 1985), multiple attenuation (Hampson, 1986), interpolation (Sacchi and Ulrych, 1995c), among other signal processing and inversion geophysical subjects. In seismic data processing, RTs are applied to map events in seismic gathers using a set of line integrals that can follow curves, usually lines, hyperbolas and parabolas. These integrals produce projections that are manifested in a new domain (Trad, 2001).

A seismic data set $d(x,t)$ presented in terms of offset (x) and travel time (t) for a continuous array can be represented as the RT:

$$m(p,\tau) = \int_{-\infty}^{+\infty} d\left(x, t = f_{t \leftarrow \tau}\left(\tau, p, x\right)\right) dx, \tag{1.1}$$

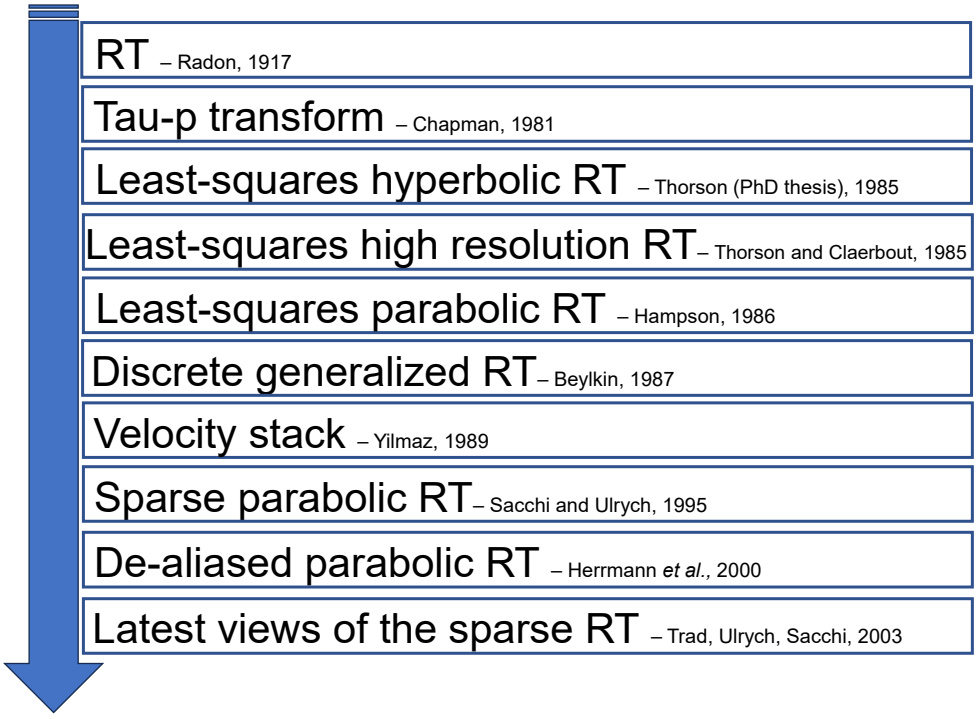


Figure 1.2: General RT timeline: Radon (1917), Chapman (1981), Thorson (1984), Thorson and Claerbout (1985), Hampson (1986), Beylkin (1987), Yilmaz (1989), Sacchi and Ulrych (1995c), Herrmann et al. (2000), Trad et al. (2003).

where $m(p, \tau)$ is the correspondent model in the RT domain with p as the parameter related to the shape of the curve (also called ray parameter) and τ as the zero-offset intercept time. The relationship between data d and model m spaces is defined by the integration pathway along the curve $t = f_{t \leftarrow \tau}(\tau, p, x)$. We refer to this operation as an RT.

Equation 1.1 represents what is known in geophysics as an inverse problem since it shows a mapping from data (Figure 1.1(a)) to the model (Figure 1.1(b)) space. A subtle distinction is that in processing the workflow for RTs usually goes from the data (Figure 1.1(d)) to the model, in this thesis, the RT space (Figure 1.1(e)). This aligns with the comparison between inversion and processing made by Claerbout (1985b) in his book Earth Sounding Analysis: Processing versus Inversion.

To get the inverse RT (or forward problem) a map from the model, $m(p, \tau)$, to the data space, $d(x, t)$, can be done by inverting the RT in Equation 1.1, such as:

$$\tilde{d}(x, t) = \int_{-\infty}^{+\infty} m(p, \tau = f_{\tau \leftarrow t}(t, p, x))dp, \tag{1.2}$$

where $\tilde{d}(x, t)$ is ideally the inverse or reconstructed data (Figure 1.1 (c) and (f)) from the model $m(p, \tau)$.

The symbol $\sim$ in Equation 1.2 means that the data is not fully restored, thus representing the adjoint pair of Equation 1.1. Furthermore, Claerbout (1985b) shows that if the adjoint operator is insufficient as an approximation for the inverse, then one should employ fitting and optimization techniques. This involves the iterative use of the modelling operator and its adjoint to reduce the misfit between original and reconstructed data, for instance, by applying the weighted least-squares approach (Sacchi and Ulrych (1995a), Trad et al. (2003)).

Thorson and Claerbout (1985) made a point that the incompleteness in the spatially sampled seismic field data is conveniently classified into three categories: limited aperture (cable length truncation), sparse sampling (aliasing), and irregular gaps in the recording array (dead traces). When (x, t) and (p, τ) are discretized the functions $d(x, t)$ and $m(p, \tau)$ are equivalents to vectors. Then, Equation 1.1 can be seen as:

$$m(p,\tau) = \int_{x_{min}}^{x_{max}} d(x, t = f_{t \leftarrow \tau}(\tau, p, x))dx, \tag{1.3}$$

where x_{min} and x_{max} refer to the minimum and maximum offset, respectively.

Similarly, Equation 1.2 can be seen as:

$$\tilde{d}(x,t) = \int_{p_{min}}^{p_{max}} m(p, \tau = f_{\tau \leftarrow t}(t, p, x))dp, \tag{1.4}$$

where p_{min} and p_{max} refer to the minimum and maximum ray parameters, respectively

Beylkin (1987) describes the RT in seismic signal processing as "a seismogram (a multi-dimensional signal array) can be viewed as a superposition of different events with energy concentrated along straight lines (at least locally). The RT maps these events into points, thus allowing identification and separation".

The RT (Equation 1.1) has its adjoint (Equation 1.2) as the first approximation to the inverse operator (Claerbout, 2004). Since seismic data is digitally recorded, and so sampled discretely in time and space, it is necessary to discretize the previous integrals by replacing it with summation (Sacchi, 2002) and imposing finite limits (offset). Therefore Equation 1.3 would be:

$$m(p,\tau) = \sum_{x_{min}}^{x_{max}} d(x, t = f_{t \leftarrow \tau}(\tau, p, x)), \tag{1.5}$$

and equation 1.4 will be:

$$\tilde{d}(x,t) = \sum_{p_{min}}^{p_{max}} m(p, \tau = f_{\tau \leftarrow t}(t, p, x)). \tag{1.6}$$

In general, RT can be classified by the curve used in the line integral paths. The most used in seismic processing are straight line, parabola, and hyperbola, which represent the linear, parabolic, and hyperbolic RT, respectively.

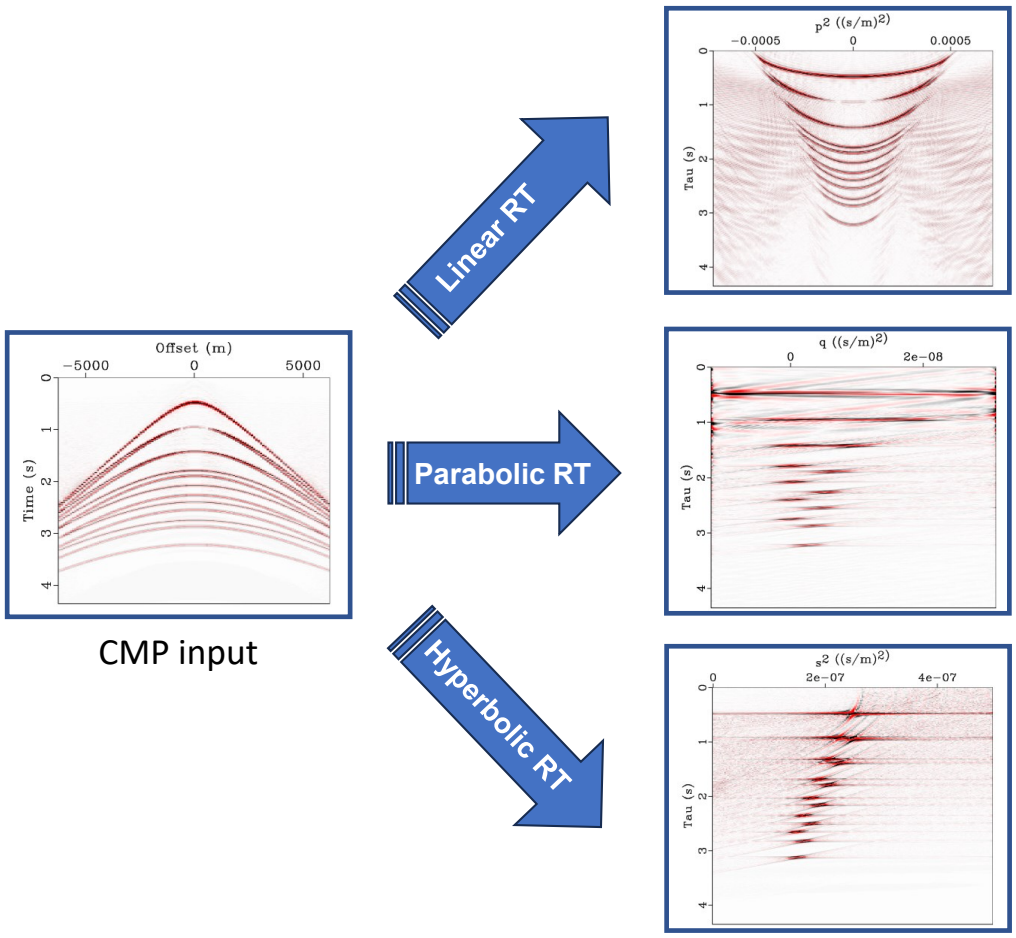


Figure 1.3: Different basis functions can be used to go from data (in this case, Common Midpoint) to the transformed space. Examples of Radon Transform (RT): are parabolic, linear and hyperbolic.

1.1.1 Hyperbolic RT (HRT)

The hyperbolic RT, also known as velocity-stack (Thorson and Claerbout (1985), Yilmaz (1989)), is the most suitable to map seismic gathers since on CMP gathers the reflection events are shaped as hyperbola. When the area's geology has a simple structure with flat layers, the shot domain can be used instead of the CMP since the reflection events will still be shaped as hyperbolas.

From a geometrical point of view, the HRT maps nearly hyperbolic events in the seismic gather (data space) to points in the hyperbolic RT space by using the hyperbolic moveout equation (Yilmaz, 2001):

$$t = \sqrt{\tau^2 + \frac{x^2}{v_s^2}}, \tag{1.7}$$

where t is travel time, τ is the zero-offset intercept time, x is offset and v_s is the stacking velocity. Thus, the HRT can be calculated by summing up the amplitudes over the hyperbolas. In the discretized case, the equation that maps from the data to the HRT domain is given by:

$$m(v_s, \tau) = \sum_{x_{min}}^{x_{max}} d\left(x, t = \sqrt{\tau^2 + \frac{x^2}{v_s^2}}\right). \tag{1.8}$$

with x_{min} and x_{max} refering to the minimum and maximum offset, respectively.

Then, the forward problem that maps back to the data space will be given by:

$$\tilde{d}(x, t) = \sum_{v_{min}}^{v_{max}} m\left(v_s, \tau = \sqrt{t^2 - \frac{x^2}{v_s^2}}\right), \tag{1.9}$$

with v_{max} and v_{min} as the maximum and minimum stacking velocities, respectively.

Since the offset axis contains relevant velocity information the ray parameter (s) is con-

veniently used as the reciprocal of the stacking velocity. Then, slowness s is given by:

$$s = \frac{1}{v_s}, \tag{1.10}$$

therefore equations 1.8 and 1.9 can also be given in terms of s (Equation 1.10). Because hyperbolas are time-variant curves, the primaries and multiples will not be exactly parallel in the RT space but will reproduce a similar trend.

1.1.2 Linear RT (LRT)

The linear RT, also known as slant-stack (Treitel et al. (1982), Claerbout (1985b)) or $\tau - p$, is calculated by applying linear moveout (LMO) to the seismic gather (shot for the case of a flat layered earth model, CMP for other cases) and summing amplitudes over the offset x such as:

$$m(p, \tau) = \sum_{x_{min}}^{x_{max}} d(x, t = \tau + px), \tag{1.11}$$

with p representing the apparent (or horizontal) slowness along the surface, in which:

$$p = \frac{sin(\theta)}{v_{RMS}}, \tag{1.12}$$

where θ is the incident angle (between the ray being reflected and the vertical axis), and v is the RMS velocity.

A hyperbola and a line in the CMP domain map ideally into an ellipse and a point in the LRT domain, respectively. The standard processing workflows are usually done in the midpoint domain, but the processor can choose to carry a full processing workflow with the data in the ray parameter (p) domain. The LRT can be applied for different goals, with the one in this thesis being to separate ground roll (geometrically linear events in the shot domain) from reflections. The LRT is often used to calculate dispersion curves for the near-surface.

1.1.3 Parabolic RT (PRT)

Even though the normal moveout (NMO) corrected data exhibit hyperbolic characteristics (hence the HRT), their curvature can be approximated by parabolas, especially when dealing with small amounts of curvature. To make the reflection events have a parabolic shape Hampson (1986) took CMP gathers and applied the NMO correction using the hyperbolic moveout (Equation 1.7) with the stacking velocity v of the primaries to get the PRT. Consequently, the primary events ideally become flat, and the multiples still have an approximately parabolic moveout (Yilmaz, 2001), therefore having different shapes. This will allow the summation along the parabola traveltime curve that can be represented in the discretized case by:

$$m(q,\tau) = \sum_{x_{min}}^{x_{max}} d(x, t = \tau + qx^2), \tag{1.13}$$

where τ is the intersection with the zero offset and t is the time after NMO correction. In the PRT case, a curvature parameter can be created and be described as:

$$q = \frac{moveout}{x^2}. \tag{1.14}$$

The PRT in the velocity domain was described by Yilmaz (1989). While changing the value of q, the basis function will match with multiples that also have a strong signature in the PRT space. Since those events have different curvatures (velocities), it is possible to map them separately. Low values of q allow the mapping of flattened events (reflections in the CMP domain after NMO correction), whereas higher values of q would map multiples.

There is also the pseudo-hyperbolic RT (Foster and Mosher, 1992), which is a different way of decomposing hyperbolic events in the time-invariant basis functions (Trad, 2001). The summation will be along the hyperbola, but it will depend on the offset and a specific depth, such as:

$$m(q,t) = \sum_{x_{min}}^{x_{max}} d\left(x, t = \tau + q\left[\sqrt{z_{ref}^2 + x^2} - z_{ref}\right]\right), \tag{1.15}$$

where z_{ref} is the specific depth.

1.1.4 Hybrid RT

In the case of land seismic, the reflections can be approximated to parabolic-shaped events after being sorted by CMP, and NMO correction is applied. Ground roll is a geometrically linear-shaped event. Since these two events usually appear superimposed in the data but do not simultaneously focus on an RT, then Trad et al. (2001) introduced the concept of hybrid RT.

The hybrid RT is based on the fact that the linear and parabolic RTs are time-invariant; therefore, they are calculated in the frequency domain. Because they have the time parameter τ in the vertical axis, it is possible to put the two RTs side by side on the same horizontal axis.

Because of the time invariance, the transform can be calculated independently for every frequency, which is also computationally attractive since it does many small operations instead of a big one in the whole time domain.

1.1.5 RT in time and frequency domain: a comparison

By applying the Fourier transform in a time-invariant system, equation 1.1 can then be seen in the temporal-frequency domain as the RT:

$$M(p,\omega) = \int_{-\infty}^{+\infty} D\left(x,\omega\right) e^{i\omega p x} dx, \tag{1.16}$$

where ω is the frequency, $M(p,\omega)$ and $D(x,\omega)$ are the Fourier transforms of, respectively, the model $m(p,\tau)$ and the data $d(x,t)$. Its operator notation after discretization is given by the inverse problem:

$$\tilde{\mathbf{m}} = \mathbf{L}^H \mathbf{d}, \tag{1.17}$$

where vector $\mathbf{d}$ indicates the data $D(x, \omega)$ at discrete values of x and fix frequency ω, $\mathbf{L^H}$ is the adjoint RT operator that maps the data $\mathbf{d}$ to the model $\tilde{\mathbf{m}}$. This *tilde* is due to the non-orthogonality of the RT, indicating that the reconstruction has been modified by projecting it onto the non-orthogonal basis functions of the model space. Therefore, further adjustments are necessary for the adjoint model $\tilde{\mathbf{m}}$ to achieve $\mathbf{m}$.

The inverse RT that makes a pair with Equation 1.16 is:

$$\tilde{D}(x, \omega) = \int_{-\infty}^{+\infty} M(p, \omega) e^{-i\omega p x} dp, \tag{1.18}$$

where $\tilde{D}(x, \omega)$ is the Fourier transform of the reconstructed data $\tilde{d}(x, t)$. After discretization, this can also be represented in operator notation as the forward problem:

$$\mathbf{d} = \mathbf{L}\mathbf{m}, \tag{1.19}$$

where vector $\mathbf{m}$ indicates the RT space $M(p, \omega)$ at discrete values of p and fix frequency ω, $\mathbf{L}$ is the RT operator that maps from the model $\mathbf{m}$. Since L is non-orthogonal, the operators $\mathbf{L}$ and $\mathbf{L^H}$ are an adjoint pair and not an inverse.

Thorson (1984) published his PhD thesis showing a technique that tried to model uncorrected shots or CMPs using hyperbolic events as basis function. Hampson (1987) mentioned that this is an extension of the classic slank stack or simply linear RT with two major differences: rather than linear, the author summed along hyperbolic (reflection-shaped) paths and then mapped to distinct points in the RT domain. By discretizing and limiting the offset, he showed that a generalized inverse could be formulated with a least-squares solution (Beylkin, 1987), such as:

$$\mathbf{m} = \left(\mathbf{L^H L}\right)^{-1} \mathbf{L^H d}. \tag{1.20}$$

To obtain the result for equation 1.18 we can substitute Equation 1.16 to then obtain

(Zhou and Greenhalgh (1994), Sacchi (2002)):

$$\tilde{D}(x,\omega) = D(x,\omega) * \rho(x,\omega), \tag{1.21}$$

with the ρ filter being:

$$\rho(x,\omega) = \int_{-\infty}^{+\infty} e^{-i\omega px} dp. \tag{1.22}$$

Equation 1.21 represents the relationship between the reconstructed data $\tilde{D}(x,\omega)$ and the data $D(x,\omega)$, with the latter being convolved $(*)$ with the filter $\rho(x,\omega)$ for every frequency. That means that to calculate $D(x,\omega)$ from $\tilde{D}(x,\omega)$, the filter ρ needs to be deconvolved frequency by frequency.

An important thing to note is that the ρ filter depends on the shape of the data (everything that has to do with the offset x) and the integration path (line, parabola, hyperbola). Consequently, having a non-dense sampling and offset boundaries makes the inversion a challenging task since this requests a deconvolution using regularization. Performing inversion or deconvolution in the RT domain is beneficial as it is the domain where high resolution is required (Trad, 2001).

The linear and parabolic RT are time-invariant, meaning that the different basis functions are parallel, so usually, they are calculated in the frequency domain. On the contrary, the hyperbolic RT is time-variant and calculated in the time domain.

By solving several small problems, one per frequency, instead of solving one large problem for all time-offset-velocity samples at the same time, the process of inversion for RTs is easier done in the frequency than in the time domain (Sacchi, 2002).

While solving in the frequency domain offers computational advantages and better preservation of the waveform, there are inherent benefits to exploring the time domain RT (Trad, 2001). The seismic reflections have a hyperbolic shape (and not linear or parabolic); consequently, in the case of the PRT, if the NMO velocities are not accurate, this can drastically affect the quality of this transform. In these cases, the HRT might be a better option.

Understanding that both time-variant (time domain) and time-invariant (frequency domain) RTs have their pros and cons, this thesis suggests an alternative approach that incorporates more information (time and frequency, sparse and non-sparse) while applying the RT. The methodology suggested here (further explained in Chapter 3) is done to better understand what are the benefits of using more channels of information that would enhance the application of RT in the seismic noise attenuation workflow.

1.1.6 The RT and sparseness

The RT operator is not orthogonal; therefore, effectively applying forward and inverse operators without data loss becomes challenging. Unless the operator is orthogonal, its adjoint is not the same as its inverse. Unlike the inverse Fourier transform that completely restores the data, the inverse RT has some restrictions (Claerbout, 2004). As seen in the limits of integration of Equation 1.1, the ideal case would require unlimited data to obtain a completely invertible RT, which is never the case for seismic data. These are truncated within maximum and minimum offset and possibly missing traces, thus affecting the resolution of the RT.

Compression refers to the process of distilling a large amount of information into a more compact form by focusing on essential details, which results in a smaller image with fewer pixels. Sparseness, by contrast, involves a small or large image where the majority of pixel values are zero or near zero, concentrating the significant information in a few non-zero values. While both methods aim to represent data efficiently, compression reduces the image size, whereas sparseness can maintain a larger size with most values being zero, thereby concentrating the information.

In linear algebra, a sparse matrix means that the matrix has lots of zeroed or close to zero elements, which might computationally help to make calculations in systems of linear equations. In some geophysics exploration fields, authors discuss sparse acquisition (or sparse sampling) which does not necessarily have the same meaning as the sparseness in RTs.

Note the amplitude smearing over the ray parameter axis in the RTs on Figure 1.3. This happens due to the finite offset aperture and sampling issues in seismic data, leading to a lack of resolution. So, to reduce the smearing problem, one can apply regularization by using model prior information, such as assuming continuity of reflections inside the gaps, which is a valid assumption in field data. This "focalizes" the information in the model domain; thus, the concept of sparse RT (Thorson and Claerbout (1985), Sacchi and Ulrych (1995b), Trad et al. (2003)) helps to address the resolution.

A sparse Radon model implies higher resolution and helps to separate events in the model space. When going from the sparse model to the data reconstructed domain, the sparse model is consistent with the continuous reflections, whereas the non-sparse model will treat the lack of data (an inherent problem in field data) as zeros (Trad, 2001).

One suitable solution can be found by minimizing a cost function penalizing smooth Radon models using iterative re-weighted least squares – IRLS (Thorson and Claerbout, 1985). A possible cost function to select the more appropriate solution from all the possible ones can be (Trad et al., 2003):

$$\phi = \left\| \mathbf{W_d}(\mathbf{d} - \mathbf{L}\mathbf{W_m^{-1}}\mathbf{W_m}\mathbf{m}) \right\|^2 + \lambda \left\| \mathbf{W_m}\mathbf{m} \right\|^2, \tag{1.23}$$

where $\mathbf{W_d}$ represents the matrix of data weights, $\mathbf{W_m}$ is the matrix of model weight (related to resolution and smoothness), and λ is a trade-off parameter between data misfit and model constraints. The most suitable solution can be found by minimizing the cost function (Equation 1.23) using interactively re-weighted least squares (Sacchi and Ulrych (1995b), Trad et al. (2003)).

1.2 Machine learning based method

Scientific programming is a powerful tool for solving various tasks, but if the task has a lot of complex rules and is based on a vast data set, this approach can become time-consuming.

Géron (2019) defines Machine Learning (ML) as "the science (and art) of programming computers so they can learn from data". So, given a certain data set, a machine can learn how to solve a problem of your needs, having its performance described by its prediction quality.

ML methods are grouped in many different ways. Whether they have human-supervised (labelled) training or not, they can be called supervised (such as classification and regression) or unsupervised (such as clustering and dimensionality reduction), and they also have other types, such as semisupervised and reinforcement learning. These criteria are not mutually exclusive and can be combined, providing a diverse scope of ML methodologies.

In the context of supervised tasks, ML can be broadly grouped into solving two main categories of problems: classification and regression. Classification involves assigning predefined categories (or labels) to the input data, making it suitable for scenarios where the output is categorical. Regression is used when the problem involves predicting a target numerical value, being a valuable tool for tasks where understanding trends in the data is relevant.

Following Géron (2019), a general ML workflow can be described as:

1. Select and understand the rules of the problem;
2. Select a data set that is meaningful for your problem;
3. Pre-process the data set if needed;
4. Shuffle (to randomize the order – optional) and split the data into training and test set (validation set – optional);
5. Choose a type of model (e.g., linear regression, logistic regression, etc.) that works for your problem;
6. Train the model[1] with the training set in such a way that minimizes a chosen cost function;

[1]train the model: running an algorithm to determine parameters for the model that best fit the training set.

7. Perform predictions on the test set and compare with the ground truth (training set);

8. Evaluate if the model generalizes well while predicting using a new data set by comparing the performance using appropriate evaluation metrics of the training set against the metrics of the test set;

9. If they are similar, it is a good indicator (no overfitting). If not, the model needs to be fine-tuned by changing its hyperparameters.

10. If the model exhibits poor robustness or significant discrepancies between training and test metrics, fine-tune the model by adjusting its hyperparameters or applying regularization techniques.

The chosen model should produce good predictions if everything happens as described above. Although, in real life, things do not usually go as planned. ML projects can go wrong for various reasons, such as having a non-representative, poor quality, insufficient, irrelevant training set or even having a simplified or too complex of a model.

In many cases, the data available for a project may not be sufficient to obtain the information desired to solve a posed question. However, the data could have meaningful insights to address a significant portion of the problem. In this regard, while using more data, ML positively contributes to a project by providing diverse perspectives on the problem, which can provide information that is not apparent with a strict set of data or even raise answers to questions beyond the project's main scope. This thesis does not aim to present a superior noise attenuation method. Instead, it explores and analyzes whether ML can offer a data-driven answer, even if only partially, to the noise attenuation problem.

An ML approach can provide an evidence-based solution that can help speed the processing workflow. In classical geophysics seismic processing, the physics of the events are somewhat taken into account through the choice of different basis functions. ML approaches, such as Deep Learning (DL), initially allow for every potential output while trying to find patterns in the data used for training. Trad (2022) noted that the redundancy of information

in the training data is the facilitator and limitation factor on what ML can achieve. Moreover, owing to its flexibility and black-box nature, caution should be taken when employing this approach, as it can be modified to suit different problems but may lack interpretability regarding the decision-making process.

ML has been applied in geophysics for a while (McCormack (1991), Röth and Tarantola (1994)). In the past five years, ML methods have been widely applied in seismic interpretation (Wu et al., 2020), faults and salt bodies detection (Zeng et al. (2019), Alali and Alkhalifah (2023)), time-lapse (Alali et al., 2021), among other topics.

In regards to seismic processing and inversion, some other applications have been made such as deblending (Sun et al., 2020c), interpolation (Saad et al., 2023), incoherent noise attenuation (Wu et al. (2019), Yang et al. (2020)), velocity analyzes (Park and Sacchi (2020), Li et al. (2022)), velocity model building (Cameron and Vestrum, 2022), migration (Wu et al. (2022), Huang and Trad (2023)), and full waveform inversion (Sun et al. (2020a), Sun et al. (2023), Zhang et al. (2023)).

1.2.1 Neural Networks (NN)

In ML, choosing the model is an important part of the project. In regression problems, some common models, for instance, are linear regression and polynomial regression. Logistic regression and softmax are examples of classification models.

An Artificial Neural Network (ANN) is a more complex model inspired by the structure and function of the human brain, which consists of interconnected layers of neurons (or nodes). McCulloch and Pitts (1943) proposed a simplified version of the biological neuron, also called an artificial neuron. An evolution of that is the threshold logic unit (TLU) used in the simplest ANN architecture: the Perceptron (Rosenblatt, 1958). Each input connection has an associated weight, and a weighted sum of its inputs gets applied to a step function.

The Multilayer Perceptron (MLP) (Figure 1.4) is a stacking of multiple perceptrons in which every layer (except the output) and a bias neuron is fully connected to the next layer.

The basic structure of an MLP consists of neurons organized into:

- input layer: receives the input (usually a multidimensional vector of) features. Neurons in the input layer do not perform computations but instead, relay the raw input data to neurons in the first hidden layer for subsequent processing,
- hidden layer(s): in between input and output layers, and it is where the interconnected neurons perform computations and the learning happens,
- output layer: produces the final output features of the network. The number of neurons in this layer depends on the task (e.g., regression, multi-class or binary classification).

A more general term to call an MLP is simply Neural Networks (NN), which, for simplicity, is the term I chose to use in this thesis. There are many NN architectures, such as Feedforward Neural Network (FNN), Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Autoencoder, Transformers, etc. For the multi-layer NN case, the output of one layer becomes the input of the next one. Also, if an NN has two or more hidden layers (which is the case for Figure 1.4) stacked up, it is called deep NN, with which the term DL concept is usually associated. In Chapter 3, I will further explain how I used the APIs *TensorFlow* and *Keras* to build the deep CNN I used in this thesis.

Each connection between neurons has associated weights, and the network learns by adjusting these weights during training to make accurate predictions. It can be used for both regression and classification problems, and its architecture allows the capture of complex and non-linear patterns within the data.

Weights

Take linear regression as an example of an ML model. It can be described, in vectorized form, as the dot product (Géron, 2019):

$$\hat{y} = \boldsymbol{\beta_n} \cdot \boldsymbol{\chi} \tag{1.24}$$

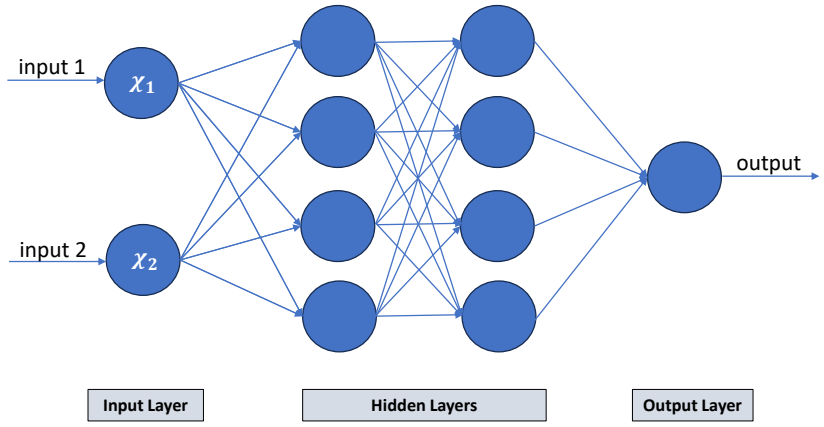


Figure 1.4: Example of a multilayer perceptron architecture. Each circle represents a neuron (bias neuron is implicit) and each arrow represents a connection with a weight associated with it. The input layer has two neurons with two inputs. There are two hidden layers, with four neurons each. The output layer contains one output neuron, resulting in one output. Since the information flows only from the input to the output, this is an example of a Feedforward NN. And because it has at least two hidden layers, it is an example of a deep NN.

where $\hat{y}$ represents the predicted value, $\boldsymbol{\beta}$ is the vector of the model's parameters where the feature weights go from β_1 to β_n and the bias is β_0, with n as the number of features. $\boldsymbol{\chi}$ represents the vector of input features.

In NN, a neuron takes multiple input signals, each multiplied by a corresponding weight. These weighted inputs are summed up, and a bias term is added. The result is then passed through an activation function, and the output becomes the input for the next layer in the network. Therefore, taking Equation 1.24 and applying an activation function could be represented as:

$$\hat{y} = \sigma(\boldsymbol{\beta}, \boldsymbol{\chi}) \tag{1.25}$$

where σ is the activation function (in this case, the sigmoid).

So, each connection (represented by the blue arrows in Figure 1.4) between nodes has its weight, which controls the strength of the connections. The learning process done over training involves adjusting these weights. Each node usually also has an associated bias, allowing the model to represent patterns even when inputs are zero.

Activation Function

When linear functions (layers) are chained together in an NN the output will consequently be linear (Géron, 2019). Other activation functions introduce non-linearity to the model, enabling it to tackle complicated problems that a linear function would not effectively address. These functions can be used in shallow (for example, Perceptron) or deep NNs. The choice of activation function (Figure 1.5) depends on the specific problem and the characteristics of the data being modelled. Some examples of activation functions commonly used in NNs are:

- Sigmoid (or logistic) function: it makes the output values ranging from 0 to 1. σ on equation 1.25 corresponds to the sigmoid function given by $\sigma(a) = \frac{1}{1+e^{-a}}$, where a is the linear combination of the neuron's inputs (each multiplied by its corresponding weight, plus a bias term).

- Hyperbolic tangent (TanH) function: it makes the output values ranging from -1 to 1, being useful in situations where the data is centred around zero. σ on equation 1.25 represents the activation function of choice, which in this case would be TanH, given by $tanh(a) = 2\sigma(2a) - 1$. Note that the sigmoid activation function is within it.

- Rectified Linear Unit (ReLU) function: $ReLU(a) = max(0, a)$ is a simple threshold operation in which for any input a, the output is the maximum of zero and a. The function has advantages over sigmoid and hyperbolic tangent since it just returns zero for negative values and the input value itself when positive values of a, so no calculations are required.

Cost Function

The cost function, also known as loss, is a measurement of the difference between the predicted and the ground truth (training set) values. When applied to the regression problem,

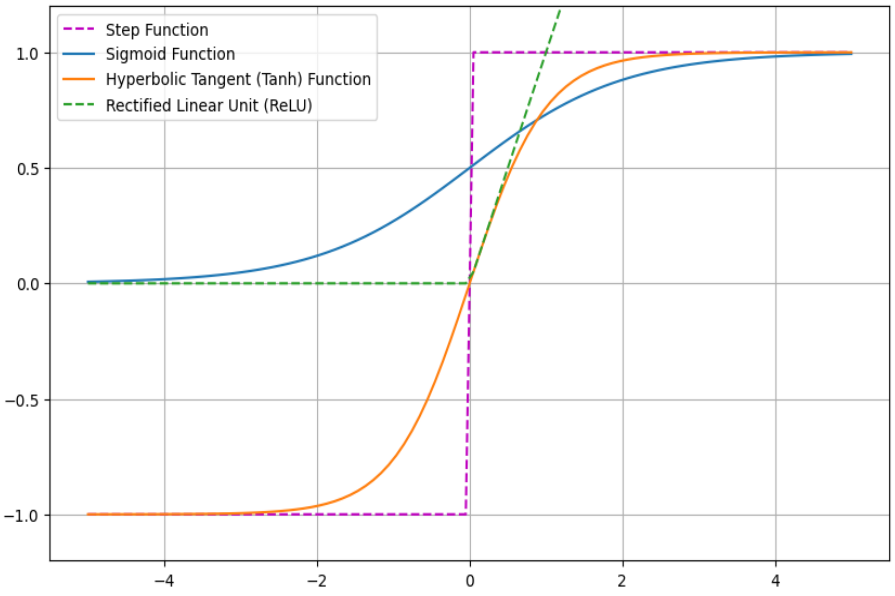


Figure 1.5: Activation functions: Step Function (dashed magenta), Sigmoid function (blue), Hyperbolic tangent function (orange) and Rectified Linear Unit function (dashed green).

NN models commonly use MSE as the loss because it is a smooth function, simplifying the optimization process.

It can be formulated as (Géron, 2019):

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(\hat{y}^{(i)} - y^{(i)}\right)^2, \tag{1.26}$$

where $\hat{y}$ is the vector of predictions for the i^{th} sample prediction, y is the vector of its true model (label), and n is the total number of samples in the training set. Usually, the lower the MSE, the better the model's performance on the training set. The loss will show the number of errors the network makes while making predictions by measuring the distance between the prediction and the target vector. In the case of the MSE, it corresponds to the Euclidean (l_2) norm.

1.2.2 Backpropagation

Backpropagation (Rumelhart et al., 1986) is a fundamental training step in NN. It uses a two-step approach to efficiently compute the gradient of the network's error for every model parameter: forward pass and reverse pass. The first makes predictions and measures the error; the second goes through each layer in reverse to measure the error contribution from each connection. Everything is done iteratively by adjusting weights and bias using an optimizer algorithm that minimizes prediction errors and converges the network to the best possible solution.

The purpose of training an NN is to learn the optimal weights and biases and use the backpropagation until the result is satisfactory for the project's need. Initializers are a method used to set the initial values of the weights and biases in the network's layers. The weights are usually initialized as random numbers within a bound. This bound can be Gaussian or uniform, based on the scale of your data. In deep NNs, a commonly used initializer in *TensorFlow* is the He (He et al., 2015), especially while using the ReLU activation function.

Another way to update the weights is by using inference. In that case, the network does not begin with completely random weights. Instead, it starts by initializing the weights with values from a previous training session. That way, it will build upon previous knowledge and try to deepen the understanding of the problem to improve the predictions. This approach aims to refine the network's understanding of the problem domain, ultimately enhancing its predictive capabilities.

Optimizers

In the process of training an NN, the goal is to minimize the cost function, which is achieved by using optimizers. The optimization process entails identifying the optimal parameters (weights and biases in NNs) that minimize the cost function over the training set. Some examples of optimizer algorithms for regression tasks are gradient descent (batch, stochastic or mini-batch) and Adaptive Moment Estimation (Kingma and Ba, 2015), also known as

ADAM.

Gradient descent is a fundamental optimization technique that iteratively updates the model parameters in the direction opposite to the gradient of the cost function, minimizing it. These updates of the weights are given by the equation:

$$\beta_{updated} = \beta_{old} - (\alpha.g), \tag{1.27}$$

where $\beta_{updated}$ is the updated weights after one interaction, β_{old} the current weights, α is the learning rate, and g is the gradient of the cost function for the weights.

ADAM extends the capabilities of gradient descent by adjusting learning rates for each iteration, a feature beneficial for optimizing non-convex functions. Its adaptive nature is rooted in analyzing gradients' magnitudes and their historical behaviour. ADAM achieves this by maintaining two key moving average estimations: the exponentially decaying average of past gradients (first moment) and the exponentially decaying average of past squared gradients (second moment). These estimations enable ADAM to adaptively scale the learning rates for individual parameters, prioritizing larger updates for parameters exhibiting large gradients or those without significant changes. Moreover, ADAM incorporates bias correction terms to mitigate the initialization bias of the moving averages, contributing to more stable and effective optimization processes. This adaptive learning rate mechanism enhances ADAM's performance across various optimization tasks, making it a popular choice for training deep NNs and addressing complex optimization challenges.

Learning rate

The learning rate, also known as step size, is an important hyperparameter during optimization. The learning rate is a scalar number (α in Equation 1.27) that will control how much the model parameters should be updated. If the learning rate is too small, the algorithm will interact many times until its convergence. If too big, the algorithm might jump across

the minimum, making it diverge and failing to find a good solution. Therefore, this hyperparameter needs to be tuned based on the characteristics of each project. Grid or random search can be used to find an optimal learning rate during hyperparameter tuning. Some optimizers, such as ADAM, adjust the learning rate during training to improve convergence.

In the context of DL, an epoch refers to one complete pass through the whole training set while training the model. In each epoch (or iteration), the weights and biases are updated using the optimizers to minimize the chosen cost function. It is common to use multiple epochs until the model reaches an acceptable level of performance since the learning rate is applied in each epoch to fasten the convergence process, as the gradient is a small number.

Metrics

There are ways to evaluate the model's performance based on how well the model parameters fit the training, validation and test sets. The metrics for regression problems are usually a measure of the distance (norm) between the (vectors of) predictions and ground truth values. It is important to choose a metric that matches the characteristics of the project's problem. For a regression model, Root Mean Square Error (RMSE), Mean Square Error (MSE) and Mean Absolute Error (MAE) are commonly used metrics. They can be described as (Géron, 2019):

$$RMSE(\mathbf{X}, h) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(h(\mathbf{x}^{(i)}) - y^{(i)}\right)^2}, \tag{1.28}$$

$$MSE(\mathbf{X}, h) = \frac{1}{n}\sum_{i=1}^{n}\left(h(\mathbf{x}^{(i)}) - y^{(i)}\right)^2, \tag{1.29}$$

$$MAE(\mathbf{X}, h) = \frac{1}{n}\sum_{i=1}^{n}\left|h(\mathbf{x}^{(i)}) - y^{(i)}\right|, \tag{1.30}$$

where n is the number of samples in the evaluation set, $\mathbf{x}^{(i)}$ is the vector of all the feature values of the $i^{(th)}$ sample, and $y^{(i)}$ is its label (or ground truth), h is the system prediction

function, $\mathbf{X}$ is the matrix of feature values of all samples, $RMSE(\mathbf{X}, h)$ and $MAE(\mathbf{X}, h)$ are the cost function that describes the distance between predicted and ground truth.

The RMSE is also known as the square root of the $l2$-norm, having also the Mean Square Error (MSE) as a similar metric but being the $l2$-norm. RMSE and MSE are sensitive to outliers and should be used accordingly. The MAE is the $l1$-norm, which is usually used in the presence of outliers in the data.

1.2.3 U-Net: a Convolutional Neural Network (CNN) case

Convolutional neural network (CNN) is a type of NN model that started its development from studies of the visual cortex and motivated the neocognitron (Fukushima, 1980). The CNNs have been used in semantic segmentation problems to classify each pixel according to the class of the object it belongs to (Géron, 2019). It is an example of a DL model and uses labelled data (supervised) to train the model while running convolutional windows to extract features from the image and classify or predict it.

Two important building blocks for the CNNs are the convolutional and pooling layers (LeCun et al., 1998). Convolutional layers perform operations of a given input data with a collection of weights (also known as filter or kernel), outputting a feature map (Goodfellow et al., 2016). To make this method more beneficial, it is possible to add more layers after the input layer, each associated with different weights to be able to extract different features from the given image (Albawi et al., 2017). During training, the convolutional layer will automatically learn the most useful weights for its task, and the previous layers will learn to combine them into more complex patterns. Consequently, a convolutional layer can apply multiple trainable weights, having multiple feature maps as an output, one per filter. (Géron, 2019).

A pooling layer is very similar to the convolutional layer but without the weights (just slide windows). Their main goal is to shrink the input image to reduce memory use and introduce invariance by aggregating the inputs (Géron, 2019). To preserve only the strongest

features within a window, the Max Pooling (maximum aggregation function) was used in the present work. There are other types of layers, such as batch normalization – BN (Ioffe and Szegedy, 2015), which are used to normalize its inputs to speed up and stabilize the training by avoiding vanishing or exploding gradients, and help with the regularization of the model.

An input-coloured image, in the context of ML, consists of multiple layers, also known as channels. Like an RGB image with three channels as an input, Figure 1.6 depicts an example of having two channels: sparse RT and non-sparse RT. In coloured images, each pixel carries colour information, represented by three channels of colours (red, green, and blue, or simply RGB) that collectively form the final image. Each colour channel in an image represents a distinct aspect that contributes to the overall visual content. Although individual channels may seem similar, merging the red, green, and blue channels produces a more comprehensive and meaningful image representation. Thus, in this thesis, I intend to use more channels (of information) of the same data, such as represented in Figure 1.6, set and analyze how this benefits the noise attenuation normally done in the RT domain.

The nomenclature of Convolutional Neural Networks (CNNs) is somewhat misleading, as the operation executed by the convolutional layers is, in fact, cross-correlation rather than convolution. In mathematical terms, convolution entails flipping one of the input signals before sliding it over the other and computing the integral of their product at each point. However, in the context of CNNs, what is actually performed is cross-correlation, a similar operation but without the flipping of the kernel (Goodfellow et al., 2016). Thus, from a Computer Vision perspective, convolution is essentially considered the mathematical operation of cross-correlation, which measures similarity. Despite this technical distinction, in the machine learning community, the term "convolution" is accepted to denote the cross-correlation operation performed in CNNs.

One of the most applied CNN architectures is the U-Net (Ronneberger et al., 2015), and it can be applied in different fields, such as cell detection in microscope image (Falk et al., 2019) and salt interpretation in seismic data (Zeng et al., 2019). The U-Net structure can

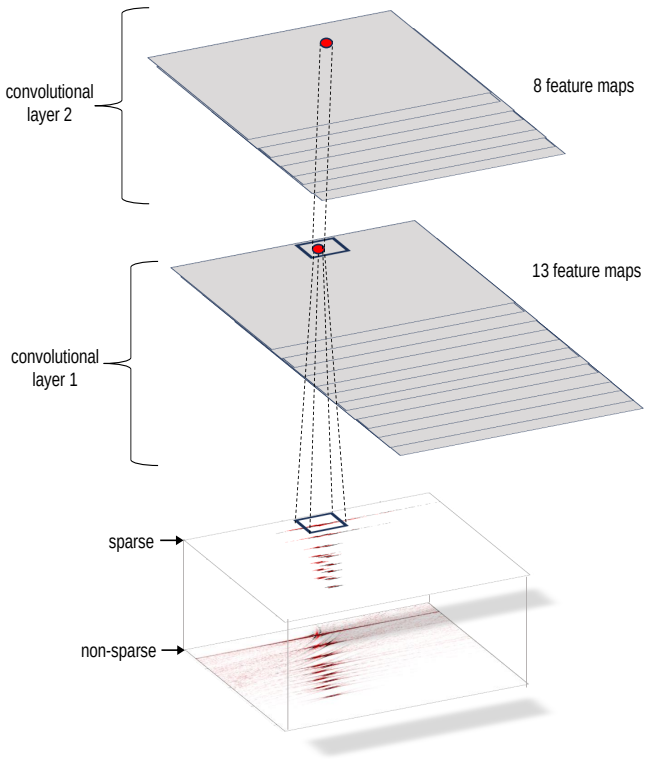


Figure 1.6: An RGB (red, green and blue) image contains three layers (or channels); in this case, there are two channels: sparse and non-sparse with its convolutional layers and feature maps.

be described as an assembly of convolutional and pooling layers within an encoder-decoder process. In the encoder part, the network down-samples the input data into a smaller size while going deeper, increasing the number of feature maps (having the possibility to use more than one channel). The decoder part up-samples the data and updates the weights by concatenating them with its correspondent piece from the encoder part to localize the relevant extracted features from the encoder part. This occurs symmetrically, forming a U-shape scheme.

It is important to note that the original design Ronneberger et al. (2015) used the max-pooling layers with a fixed pooling window size (2x2), which may not be suitable for non-square input images. Furthermore, labels are an important parameter in this method since they are the information the network will use to learn to identify a specific feature in an image.

1.3 Motivation for this thesis

In this thesis, I consider the application of DL to transforms used in seismic processing. DL is a good candidate for doing this because of its capability to recognize non-linear patterns in images, which is relevant while working in signal processing transformations. The transformations: allow the separation of components in the data and, allow the conversion of non-regular sampling data into regular sampling. Hence, the separation of multiple or ground roll from reflections in the RT domain can be done even when there is no complete spatial separation in the model domain.

Although applying ML methodologies to solve seismic processing tasks is not something new to geophysics (further discussed in Chapter 2), the motivation is to tackle challenges in traditional seismic data processing tools while applying a deep NN technique.

The core idea of this thesis is the application of ML as an alternative to traditional sparseness techniques. Traditional applications of RT would use line boundary mute or mute by amplitude, whereas NNs can introduce non-linearity (inherent characteristic of field data) and a data-driven approach to this process. While sparseness works well for idealized data, it needs help with complicated and overlapping events, such as field seismic data. Furthermore, an interesting analyzes is to evaluate if the concept of sparseness still holds as having a higher resolution while using ML.

By exploring the pixel-by-pixel (in this thesis, window-by-window) approach, the key is to make the network intelligent enough to handle overlapping information and other data complexities to effectively convey how this approach complements or supersedes traditional sparseness techniques.

Furthermore, I aim to create a methodology that maximizes the utility of RTs by utilizing more channels of information. This approach is analogous to using multiple colour channels in an RGB image, where each channel provides distinct information and composes a better final image when used together. The intention is to evaluate how this multi-channel approach benefits the noise attenuation portion of the seismic processing workflow.

1.4 Thesis scope and objectives

This thesis is organized as follows:

- In Chapter 1, I go over an introduction to Radon Transforms and Neural Networks.
- In Chapter 2, I do an overview of multiple and ground roll as seismic noises and how standard geophysical seismic data processing has been tackling the noise attenuation problem.
- In Chapter 3, I bridge the knowledge explained in Chapters 1 and 2 to build the workflow for the methodology presented in this thesis.
- In Chapter 4, I illustrate some applications of multiple attenuation using the methodology suggested in Chapter 3.
- In Chapter 5, I explore some applications of ground roll attenuation using the methodology suggested in Chapter 3.
- In Chapter 6, I present my conclusions and recommendations for further work.

Some of the work shown in this thesis has been presented and shown in other reports and conference abstracts. Chapters 1-5 were largely described on CREWES reports (Fontes et al. (2022), Trad (2022), Fontes and Trad (2023a)) and GeoConvention abstracts (Fontes and Trad (2023b), Fontes and Trad (2024)).

Chapter 2

Seismic noise attenuation

In the context of an ML project, it is important to understand the problem in order to interpret and evaluate the current solutions. This can serve as a reference point to provide insights into potential solutions and compare performance to a baseline. Therefore, in this chapter, I will go through a general timeline of methodologies and the state of the art in geophysics for seismic noise attenuation.

In general terms, any unwanted signal is considered noise. To understand the wanted signal, we need to consider the goals of the project and the data available. The main goal of a processing workflow is to remove noise but keep intact primary reflection events since the geological information we aim to interpret lies in these physical responses.

Sheriff (2002) divides noise into two groups: coherent noise, including non-reflection coherent events, and random noise, such as instrument noise and other non-coherent energy. In the context of seismic data processing, one of the most common uses of the RTs is to attenuate coherent noise, such as ground roll and multiples, while most of the random noise is naturally attenuated by the stacking or migration processes.

Once the processor chooses the noise attenuation methodology, there are generally two approaches to go with it. The first approach is to predict the noise and subtract it from the original data. Seismic data is a composition of various seismic traces, and the convolutional

model describes a seismic trace such as:

$$st = wl * r + \delta, \tag{2.1}$$

where the seismic trace st is the result of the convolution ($*$) between the source signal or wavelet wl and the reflectivity r plus an associated noise δ.

The noise is usually assumed to be random and uncorrelated with the signal. Looking at this formula, it might seem that simply subtracting the noise is easy. However, noise can be presented in different ways and forms (not just random). The processor should be careful in this subtraction (usually called mute) while maintaining as much reflection information as possible.

The second approach is to model the reflections so as not to consider the noise in this process. Both ways have advantages and disadvantages associated with it, which I will further discuss in this chapter.

Depending on the data type, marine or land seismic, the processor will deal with a different noise type (multiples or ground roll, respectively). Consequently, the characteristics of the data directly influence the processing workflow and dictate the methodology applied to address the noise problem. As outlined in Chapter 1, the ML workflow typically starts with selecting and understanding the problem. In this chapter, I aim to look into the geophysical challenge posed by noise, explore the methods utilized for its attenuation, review traditional approaches adopted by the geophysical community to tackle this problem and introduce some ML methodologies.

2.1 Multiples

In the seismic reflection method, multiples can be defined as the seismic energy reflected more than once at the subsurface that the receivers record. This energy that bounces back and forth is a coherent noise and will remain in the stacked section unless appropriately

addressed. If not effectively attenuated (while minimally affecting the primary events), multiples can lead to errors in the processing, inversion and interpretation steps of the seismic analyzes, which can be very costly, for example, for hydrocarbon exploration projects. Thus, attenuating multiples is an essential step of the seismic processing workflow.

The occurrence of multiples is based on the nature of the subsurface, usually happening in areas with strong velocity contrasts or complex structures, such as dipping layers and water bottoms. Multiples are a type of noise that usually have a frequency and amplitude in the range of similar close-by primary reflections. However, depending on how much energy it loses as it bounces around more than one layer or in a long wave path, it can diminish its frequency and amplitude range.

Multiples are a very common noise in marine seismic data. One way to group the multiples types is according to where the ray path reflection occurs (Dragoset and Jeričević, 1998). Surface (or free-surface) multiples originate with at least one downward reflection forming at the water surface, while internal multiples (also known as inter-bed multiples or peg-legs) have all of their downward reflections starting at the subsurface. The surface multiples are usually the strongest in amplitude for a sedimentary basin without salt bodies and igneous rocks because the reflection coefficient at the water surface is usually higher than the rest of the events.

Another way to group multiples is by the size of their reverberation path (Peacock and Treitel, 1969). Short-path multiples have a small ray path, which arrives soon after the primary reflection, whereas long-path multiples have an extended ray path, introducing an increase in travel time. They are also known as short and long-period multiples, respectively, and have the potential for interference with primary reflections.

2.1.1 Methods of attenuating multiples

In the seismic data processing workflow, attenuation of multiples is typically performed across various stages, each addressing specific characteristics in the seismic domain. For

instance, stacking NMO-corrected gathers is an effective way to attenuate multiples, but not all its types. During the pre-stack stage, attention is directed toward low-velocity energy clusters within the velocity analyzes spectrum. Furthermore, in the post-stacking setting, multiples can be discerned in the stack section by analyzing their similar travel times, and their identification is done by analyzing the structures of the strong reflection interfaces associated with multiples and the underlying robust reflection interface.

Weglein (1999) remarks two broad categories to approach the problem of multiples attenuation: explore a feature that differs among multiples and primary, or predict (and then subtract) multiples from the data. The first is based on modelling or inversion of the acquired wavefield. The latter category attacks two main feature differences: periodicity, where multiples are periodic but primaries are not (Taner, 1980); and separability, where a transform is utilized to separate and mute the events.

Similarly Xiao et al. (2003) organized the multiple attenuation methodologies into three main groups: deconvolution, wavefield prediction with subtraction and filtering. Deconvolution utilizes the periodicity of multiples to create an operator based on assumptions like the data being zero-offset and in a flat-layer Earth model without lateral variation. The effectiveness of this method decreases in scenarios where the interface that generates the multiples is not simple (complex seabed, for example) or in the case of long-period multiples (not helpful for the periodic requirements) (Xiao et al., 2003). This methodology attenuates short and long-path multiple based on the filter length, data window, and prediction distance. Using the Linear RT domain can be beneficial since multiples become periodic in the new domain, and predictive deconvolution (Peacock and Treitel, 1969) can be applied.

The second group is the wave equation-based techniques. An example of this methodology is the ISS – inverse scattering series (Weglein et al., 1997), which uses data and background models to predict multiples and then subtracts it from the original data. The subtraction of the predicted events is usually done adaptively, based on the wavelet of the reflectivity prediction (Xiao et al., 2003).

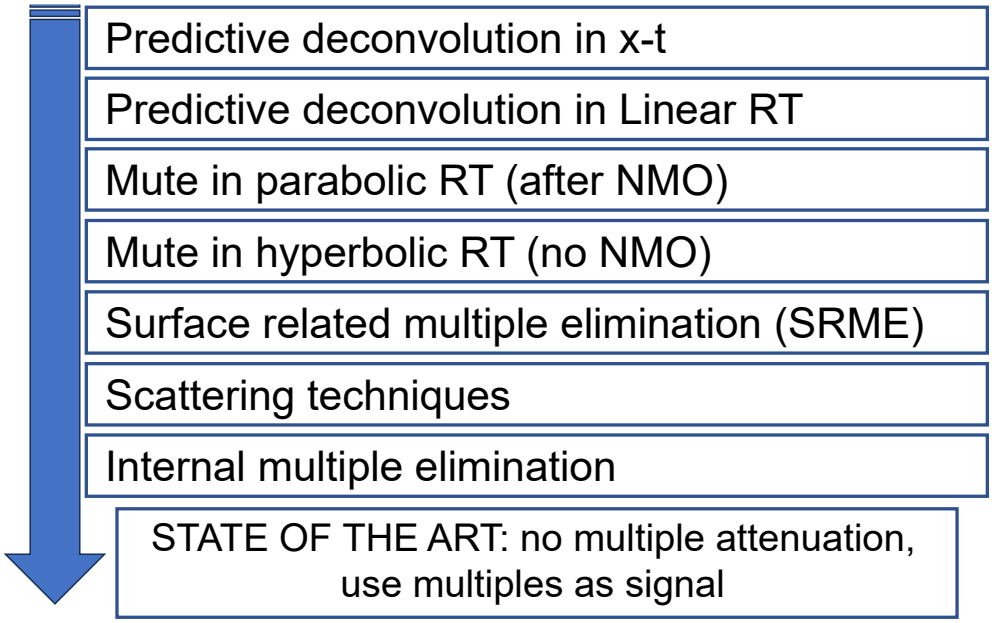


Figure 2.1: A general multiple attenuation timeline of methods

The third group, filtering methods, is based on the fact that in certain transformed domains, the difference in NMO velocities for primary and multiples is more evident. For example, primaries typically have less moveout than multiples (Yilmaz, 2001), so based on the difference in NMO velocities, the processor would use this convenient domain to help in muting multiples. The success of this method depends on sufficient moveout differences, which does not happen in certain cases such as near offset traces and peg-leg generated in shallow water (Xiao et al., 2003).

Figure 2.1 shows a timeline of some of the most important steps toward developing the multiple attenuation methods. Taner et al. (1995) used a multichannel predictive deconvolution performed in the $x-t$ domain applied to multiple attenuation. Some relevant approaches for multiples prediction have been developed, such as the surface-related multiple elimination – SRME (Verschuur (1992), Dragoset and Jeričević (1998), Verschuur and Berkhout (2015)), the inverse scattering series – ISS (Weglein et al., 1997), among others. The current state of the art is to treat not only primaries but also multiples as signal (Berkhout and Verschuur (1994), Weglein (2016)).

2.1.2 RT applications of multiples attenuation

Events shaped as parabola and hyperbola in the data domain will be ideally represented as an isolated point in the parabolic and hyperbolic RT domain, respectively. Also, after applying the Linear RT, multiples are periodic in the slowness (reciprocal of apparent horizontal velocity) but not in the time domain. Therefore, they have a larger moveout than primaries, which makes it possible to separate them in certain domains, such as the RT space.

In a way, the RT can be seen as a methodology that belongs to the deconvolution and filtering categories in which multiples are estimated in the RT space. Some authors, such as Berndt and Moore (1999), make the prediction of multiples in the RT, go back to the data (time/space) domain with these multiples only and then subtract that from the original data. Another way is to predict primaries only in the RT domain and then return to the data space (Kabir and Marfurt, 1999). The processor can even predict and subtract the multiples in the RT and return the data space with (hopefully) primaries only.

Using the multiple's periodicity, some authors have applied RTs aiming for multiple attenuation. Some examples of multiple attenuation via RT can be seen in Figure 1.2. Treitel et al. (1982) applied the slant stacks to attenuate multiples. Hampson (1986) showed that the RT with parabolic basis functions is a convenient domain to filter out multiples. Yilmaz (1989) applied the hyperbolic RT to attenuate multiples by introducing the concept of a stretched domain to mitigate the amplitude smearing along the velocity axis. Foster and Mosher (1992) discussed examples and extended multiple suppression using target-oriented parabolic RT. Trad et al. (2003) demonstrated the relevance of high resolution for multiple attenuation in different types of RTs. Trad (2003) and Trad et al. (2004) applied the hyperbolic RT handling the dipping layers with apex-shifted events.

Timeline of methods - attenuating multiples in the RT domain

The simplest example of utilizing a mute filter in the RT space to remove unwanted noise (such as ground roll and multiples) is the case of a line boundary (Figure 2.2a). In this case,

the fact that multiples and primary reflections are separated in the RT space since they have different velocities makes the mute as simple as a line boundary with primaries on one side (Figure 2.2(a)), where the filter value is set to one, and the coherent noise on the other side with the filter valued as zero, using a "taper". In a perfect scenario (Figure 1.1(b)), signal and noise are well mapped and sampled. In reality, this is not what happens with field data; therefore, the signal is extended from what would have been a point (Figure 1.1(b)) to an "area of information" (Figure 1.1(e)). These aliasing artifacts (Moore and Kostov, 2002) result from poor sampling and limited aperture on the data domain. Thus, it will not have a well-defined RT panel causing an increase in the amplitude of aliased events (Marfurt et al., 1996) that fall outside the slowness analyzes window. In this regard, the sparse RT (Sacchi and Ulrych (1995b), and Trad et al. (2003)) tries to address that issue.

To have a more accurate separation of the noise and useful events and then apply the mute, the processor would need to manually create a boundary to convey each dataset's complexity, which is difficult and time-consuming. Having this in mind, Trad (2003) introduced the idea of a mute ("smart mute") in which instead of only relying on the localization of the noise (line boundary) in the RT domain, the amplitude is used as a third axis (Figure 2.2(b)). With this approach, events that overlap in the data domain could be separated in the RT domain even if they fall into the same model parameter.

This idea was used before in Harlan et al. (1984) to remove noise using focusing as the guiding decision maker. In fact, from a broad perspective, sparse transforms also use this concept while trying to enforce spatial localization by softly penalizing (muting) low-amplitude components of the inverted model. Regarding the concept of penalizing low amplitudes, it is important to notice that small amplitudes can be essential to reproduce weak events and complicated waveforms. Essentially, that is an issue when using a line boundary alone because these low amplitudes are visually overwhelmed by the stronger mapping from other events. Muting by amplitude alone would also suffer from the same problem.

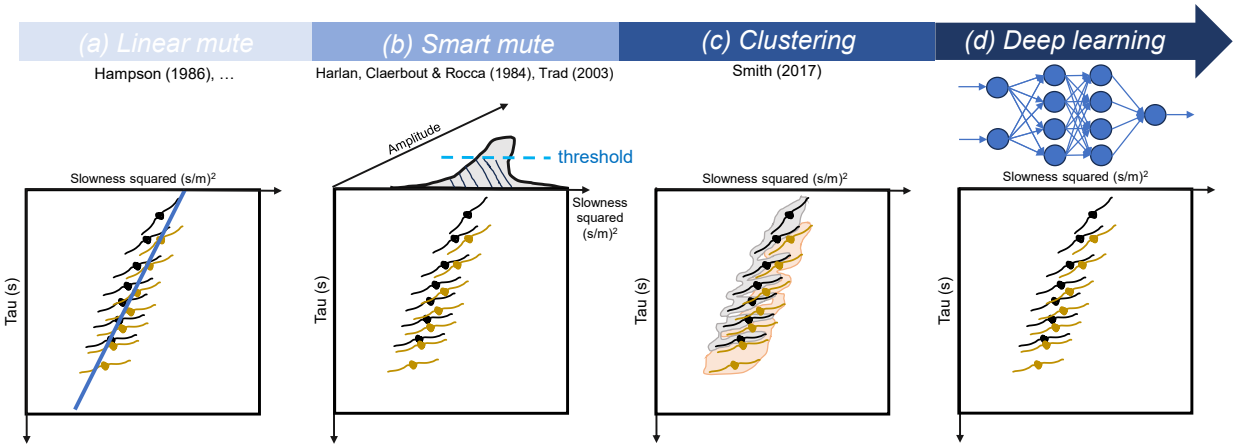


Figure 2.2: Timeline of methods for primaries (black) and multiples (yellow) separation: (a) Linear mute (Hampson, 1986), (b) Smart mute (Harlan et al. (1984), Trad (2003)), (c) Clustering (Smith, 2017), and (d) Deep learning. The RT used to illustrate the timeline is the Parabolic RT.

2.1.3 ML and RT applied to multiples attenuation

As noted by Russell (2019), in the pursuit of applying physics to real-world problems, we often find ourselves simplifying complex geophysical scenarios. However, it is essential to acknowledge that the geophysical solutions generated from these simplifications often over-simplify the inherent complexities of the real world. Consequently, employing tools like ML can be critical in finding the otherwise unnoticed nonlinearities within these solutions, offering insights beyond the scope of traditional theory.

The clustering technique (Figure 2.2(c)) can be related to mute by spatial localization and amplitude simultaneously. We could think of the clustering method as being more intelligent than the mute by amplitude. For example, the idea of clustering has been applied to assist velocity auto-picking (Smith, 2017) to reduce the time spent doing velocity analyzes. However, clustering can be very data-dependent since it is an unsupervised ML technique, so scientists seek alternatives with more generalization power.

An alternative that is more flexible than clustering would be Deep Learning - DL (Figure 2.2(d)), which provides a methodology that can help to understand noise nonlinearity better. Although in classical processing, the physics of the events are taken into account, in the ML

approach, the network tries to find patterns and predict them based on the data used for training (supervised method). The idea behind this process is similar to clustering but leveraging the flexibility and ability to incorporate vast amounts of not-well-behaved data.

As mentioned in Chapter 1, ML is widely applied in geophysics for topics like seismic interpretation and processing. While seismic interpretation, for example, fault mapping, usually involves spatially regular grids in migrated sections, seismic processing faces challenges of multidimensionality and irregularly sampled groups of seismic traces, so patterns and coherence are extremely affected by these factors (Trad, 2022). In seismic processing tasks, the data are used as a collection of time series, whereas DL techniques, developed in the field of computer vision, treat seismic as images. A DL approach is a methodology that can speed up the process, although because of its focus on images with regular sampling, flexibility, and black-box nature, it should be used with caution.

DL, particularly the U-Net architecture, has been used as an alternative approach to address the attenuation of multiples. The training portion is a vital step while using Neural Networks since the machine will learn from the provided examples, so an appropriate training set is necessary.

Bugge et al. (2021) trained on synthetic randomized pairs of prestack seismic, with and without multiples, preserving the amplitude response in near offsets, even in cases of overlapping events and far offset. Durall et al. (2022) also used synthetic-only pairs of multiple-infested and multiple-free NMO corrected seismic gathers, subtracting the multiples while maintaining the primary energy crosspoint, far offsets and high frequencies. Another application for multiple attenuation is to use the U-net in the adaptive subtraction step (Li et al., 2021).

Bugge et al. (2021) and Durall et al. (2022) compared their final DL results with an industry-standard RT, therefore suggesting that their method is an alternative to conventional RT. Other authors used RT and applied U-Net using the same methodology to attenuate multiples. Based on the fact that sparsity facilitates the separation of primaries

and multiples Xue et al. (2022) used the U-Net to predict a sparse version of the multiples-only HRT panels training set, showcasing that through multiple attenuation. Fernandez et al. (2023) explored multiple strategies of inputs and labels (primaries only, multiples only and both), training on synthetic seismic gathers and comparing against high-resolution RT, demonstrating competitive results on real data.

In this thesis, I will apply RTs in seismic data that has multiples and use these RT panels as input to train the U-Net to predict multiples-only RT panels and later subtract that from the original. It is also possible to predict primaries only RT panels. The goal is to analyze how the hyperparameters used affect the effectiveness of this task and if the result is geophysically meaningful.

2.2 Ground roll

Ground roll is considered a coherent linear noise and is commonly found on land seismic data. This type of Rayleigh wave occurs because of the coupling of compressional (P) waves and the vertical component of the shear wave (SV) that propagates along the free surface. It can have strong backscattered components because of lateral heterogeneity in the near-surface component (Yilmaz, 2001).

This type of surface wave usually has a low frequency and low velocity, resulting from the dispersive nature of surface waves, where different frequency components travel at different velocities due to the spread of the waveform over time. The weathered materials and/or unconsolidated sediments in the near-surface influence the ground roll to have a strong amplitude. This is also the result of a 2D propagation pattern since they travel mostly in the near-surface.

2.2.1 Methods of attenuating ground roll

This coherent noise superposes the shallow reflections in short offsets and deep reflections in short and larger offsets, especially in the center of the shot gathers. A common goal in the denoising step of the land seismic processing workflow while in the shot gather domain is to decompose the data into reflection and ground roll components (Claerbout, 1985a). Coherent noise attenuation methods are usually based on filtering and prediction. Several methods have aimed to attenuate and predict ground roll in the geophysics literature.

One of the simplest and most widely used techniques for separating ground roll and seismic reflections is the $F-K$ filter since this noise has relatively low-frequency content and low velocity. It uses a 2D Fourier transform in the seismic data to go from the space-time $(x-t)$ to frequency-wavenumber $(F-K)$ domain and a slice can be drawn with a 2D band-pass filter being employed to selectively retain or reject specific parts of the $F-K$ spectrum (Embree et al. (1963), Wiggins (1966)).

For instance, in the VISTA software (Schlumberger, 2015), the $F-K$ analyzes window uses a 2D Fourier transform on shot gathers, where the ground roll is coherent, to obtain the $F-K$ spectrum to be able to design a filter window to isolate the desired portion of the data for retention or subtraction from the original shot gathers. This involves multiplying each sample of the data by the corresponding $F-K$ point of the filter, followed by an inverse 2D Fourier transform to reconstruct the attenuated data.

Linear events that have coherence in the $t-x$ domain can be distinguished based on their dips in the $F-K$ domain, easing the removal of specific types of unwanted energy from the dataset (Yilmaz, 2001). Furthermore, ground roll typically occupies the low range of the frequency band in the $F-K$ spectrum, making it a convenient domain for identifying this noise, whereas in the $t-x$ domain, ground roll is often widely spread. However, filtering in the $F-K$ domain may attenuate low frequencies associated with relevant reflections, so caution is necessary on its application due to this trade-off.

Additionally, this technique encounters challenges when dealing with irregularly sampled

data in space, a common scenario in field seismic data. The spatially aliased and dispersive nature of this coherent noise, coupled with potential wrap-around effects induced by the Fourier transform, can limit the effectiveness of this methodology (Perkins and Zwaan (2000), Yilmaz (2001)).

Another technique for ground roll attenuation is the polarization filters. It is assumed that in the wave propagation through the subsurface, the particles move linearly for body waves (reflections) and elliptically for ground roll, and while using multicomponent geophones rather than an array of single components, this characteristic can be utilized (Kragh and Peardon, 1995). Building on this concept, another method for distinguishing reflections from ground roll utilizes polarization filters, leveraging seismic waves' orientation and polarization traits to differentiate between ground roll and primary reflections. One of the challenges of this filtering method is that it breaks for the case of not pure Rayleigh waves and reflections or conversions in not planar reflectors since its first assumption fails (Sánchez-Galvis et al., 2016).

Singular Value Decomposition (SVD) techniques provide a means of separating linear events from desired seismic reflection signals, and it has been used as a coherent filter (Freire and Ulrych, 1988). This is done by decomposing the original data matrix into a multiplication of three matrices: two orthogonal with autovectors of the original data and one diagonal with singular values of the original data matrix. In NMO-corrected CMP gathers, the reflections will usually be in some of the initial eigenimages since they will be laterally coherent, whereas the ground roll will tend to be in the last eigenimages.

What is expected when dealing with field seismic data is that because the events are located in different bands of the eigenimages, attenuation of the ground roll can be employed during the reconstruction (Bekara and Van der Baan (2007), Porsani et al. (2010)). So, while using SVD as a filtering method, what is expected is to keep the reflections and, while doing that, leave the ground roll out. SVD can also be used to predict the ground roll and then apply an adaptive subtraction to obtain the attenuated version (Cary and Zhang, 2009).

The curvelet transform (Starck et al. (2002), Candes and Donoho (2005)) is a methodology that can be used to capture seismic signals and their geometric structure through curved edges and features. This transformation decomposes the input signal into curvelet coefficients, which encode information about these features' presence, scale, orientation, and location. This permits the sparse representation of seismic data, concentrating coefficients on relevant features and facilitating tasks like denoising. By transforming seismic data into the curvelet domain, filtering can be applied to attenuate ground roll noise effectively. This process involves suppressing or removing coefficients associated with ground roll while retaining those corresponding to desired seismic signals.

Moreover, curvelets combine orientation features in space with multiresolution in frequency and wavenumber domains, enabling a good feature representation similar to techniques involving multi-input multi-label (frequency and time, for example) in ML processes. Curvelet transforms enable a better resolution and quality of seismic images, particularly in identifying subsurface structures with directional features.

2.2.2 RT applications of ground roll attenuation

Events shaped as lines in the data domain will ideally represent isolated points in the linear RT domain. For land seismic data, the ground roll can be approximately described as a cone of lines, and then the LRT can be applied to separate this coherent noise from the rest of the data. This methodology should be applied in the domain that this noise is coherent, which is the case for shot gathers.

Ground roll isolation using Radon Transform (RT) is typically approached in two manners. Some researchers employ the noise prediction in the model space (RT), go back to the data domain and then subtract it from the original data, ideally resulting in a ground roll-free version of the data. Another way would be to use the RT to go to the model space so that this domain will be the attenuated version already, also known as filtering.

Some examples of ground roll attenuation via RT exist, including the work of Hu et al.

(2016), who highlighted the importance of high resolution in ground roll extraction using linear RT. Seismic data features events with different shapes, such as linear noise events overlapping the hyperbolic reflections, leading to a significant decrease in the effectiveness of mapping the reflection events through the RT. A possibility to overcome this challenge is to combine two or more operators as a combined RT approach, which Trad et al. (2001) named as the Hybrid RT. They used a linear and pseudo-hyperbolic RT to achieve a sparse data representation within the model space. Their approach assumes that linear events characterize ground roll, while reflections are approximated as hyperbolas.

2.2.3 ML applied to ground roll attenuation

Employing tools like ML can present an alternative way of addressing heavily aliased coherent noise, as it has the capability to identify overlooked nonlinear features. This approach has the potential to provide insights beyond the scope of conventional ground roll attenuation methodologies. The CNN architecture has been used as an alternative method to tackle ground roll attenuation. For instance, Pham and Li (2022) used the ground roll characteristics for its suppression using a 2D CNN with specific DL blocks, not requiring the selection of a frequency threshold and masks like in traditional separation techniques. The authors developed their own training set and corresponding labels demonstrating the efficacy of their supervised training.

Yu et al. (2019) applied a CNN to attenuate multiples, but they also attenuated linear noise that is not ground roll conducting predictions on both field and synthetic seismic datasets. They also performed predictions for attenuation of scattered ground roll in field seismic data using the same CNN framework. Data preprocessing involved segmenting the datasets into patches of 40 x 40, and a comparative analyzes was done between the CNN approach and the industry-standard F-K filtering technique, with the latter serving as the reference label for the original field dataset containing ground roll. The CNN method had similar results to the F-K-filtered one as the network was trained using the F-K-filtered data

as labels. However, the CNN-filtered version kept some of the frequencies below $15Hz$, which the F-K filtering technique did not effectively capture. The designed network can handle more complex tasks if fed with enough training samples.

Furthermore, some authors have been using unsupervised DL methodologies. Liu et al. (2023a) explored the potential of an unsupervised DL approach for ground roll and scattered noise attenuation. It compares the results with a high-resolution hyperbolic RT application, which usually cannot handle scattered noise that does not have linear apparent features. Liu et al. (2023b) implemented a self-supervised approach to suppress ground roll using the prior knowledge that this coherent noise is source-generated, and based on that, they used masks for the noisy labelled data step without requiring a clean label to help the training process.

A recent advancement in seismic processing involving DL is StorSeismic, as introduced by Harsuko and Alkhalifah (2022). This approach draws inspiration from the encoder component of the Transformer architecture proposed by Vaswani et al. (2017) and uses a Bidirectional Encoder Representation from Transformers (BERT) (Devlin et al., 2018), allowing a framework that pre-trains the NN to learn the relation between traces and then fine-tune for specific processing tasks.

Zhang and van der Baan (2021) employed a supervised CNN application in which the training strategy of the network integrates synthetic and field data examples, incorporating a radial transform in both low and high-frequency domains to create a preliminary separation.

Compared with the multiple attenuation case, where many published methodologies use ML and RT to mitigate that noise, the number of proposed methodologies is much less for ground roll attenuation. Therefore, I will apply RTs and use these panels as input and labels to the U-Net as a methodology to predict RT panels of ground roll attenuated seismic and analyze how the hyperparameters used affect the effectiveness of this task and if the result is physically meaningful.

Chapter 3

Methodology and workflow of tests

In Chapter 1, I explained a broad overview of the diverse applications of DL models across various geophysical areas, whereas Chapter 2 focuses on the task of seismic processing noise attenuation. Building upon that foundation, the current chapter goes deeper into the architecture of the U-Net, central to the experiments employed in this thesis. Here, I analyze this architecture, including its parameters, and show how it was tailored to tackle the challenge of noise attenuation

Furthermore, I provide an outline of the workflow adopted in the subsequent chapters, which integrates the utilization of RTs with the U-Net predictions: data generation (synthetic case) or field data loading and pre-processing, data preparation for ML, training and prediction using the U-Net DL model, and visualization of the seismic data results sorted by CMP or shot gathers. I also explore the collaborative relationship between ML and seismic processing, shedding light on the generation of synthetic seismic data and highlighting the major role of data preparation in both ML and seismic processing projects.

As previously discussed, CNNs represent an example of supervised learning techniques, characterized by the use of convolutional and pooling layers. Moreover, to qualify as a DL process and be able to "see" complex and non-linear patterns within the data, a CNN must have two or more hidden layers, a criterion met by the architecture employed in this

thesis. Throughout the experiments showcased in this thesis, a U-Net is used to address a regression problem, treating seismic data as images rather than as time series, as typically done in seismic processing.

3.1 U-Net architecture

The architecture of the U-Net model was utilized for the experiments conducted in this thesis. As highlighted in Chapter 1, this is a CNN structure used for regression tasks specifically aimed at predicting values in the RT domain. As a supervised method, the labels are an important parameter since it is the information that the network will use to learn how to identify specific features in an image.

DL processes deal with NNs attempting to train machines through several layers of logic. A neuron aggregates multiple input signals by multiplying each with a weight, adding a bias term, and then passing the result through an activation function. This transformed output serves as the input for the subsequent layer in the network. During training, the learning process happens, and, for each neuron, the number of connections within the network will dictate how often the weights will be updated. The amount of hidden layers is the factor that tells how "deep" this learning process will be.

The black-box nature of this process lies in the fact that the hidden layers have information that is not interpretable by humans. The choice of hyperparameters will strongly rely on the specific problem the processor wants to solve as well as the characteristics of the data being used so all of the parameters used while building an architecture will be relevant for the analyzes.

The U-Net architecture is an encoder-decoder structure designed to capture hierarchical features from input data (images) while building predictions from the features. The encoder section of the U-Net is composed of convolutional layers responsible for extracting essential features from the input image and pooling layers that downsample the spatial dimensions

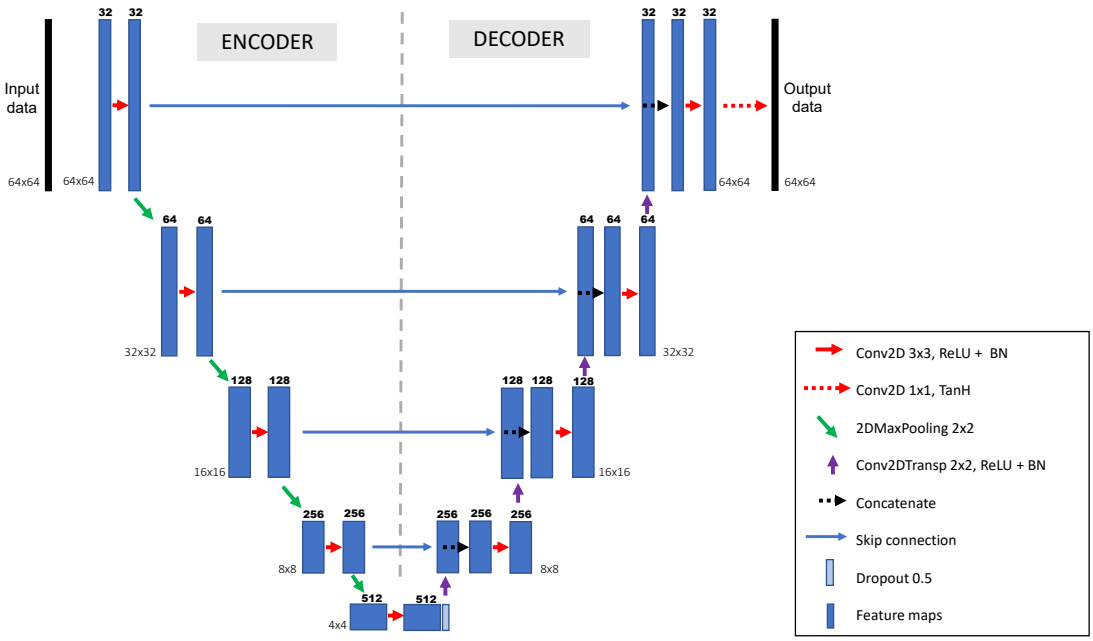


Figure 3.1: Schematic representation of the modified U-Net architecture (Fontes et al., 2022).

of the input feature maps while retaining the most important information (LeCun et al., 1998). Conversely, the decoder section utilizes transpose convolutional layers to increase back the spatial dimension of the feature maps to match the size of the ones in the encoder part while decreasing the number of feature maps. Skip connections play a crucial role in helping capture local and global information by transferring information from the encoder to the decoder part, preserving spatial information and enhancing gradient flow during the training process.

Figure 3.1 shows the U-net architecture used in the present thesis. In the encoder part, the network uses four layers to down-sample the input data (initially with a 64x64 spatial dimension) into a smaller size (4x4) while going deeper increasing the number of feature maps. Each of these layers sequentially contains a 2D convolutional layer (filter size of 3x3, and padding) with a ReLU activation function, followed by Batch Normalization (BN) and a Max Pooling layer (sized 2x2). Each filter learns how to detect different features in the input data, and in this architecture, the number of filters starts at 32 in the first layer with

the output of this layer having 32 feature maps, and the number of filters doubles after each Max Pooling layer, getting to 512 in the central layer.

The central portion of the architecture contains a single convolutional block, which compresses the spatial information into a high-dimensional representation using a 2D convolutional layer (filter sized 3x3) followed by Batch Normalization, ReLU activation, and Dropout (rate of 0.5).

During the decoder part, the network uses four layers to up-sample the feature maps while decreasing the number of filters. The weights are updated through the concatenation with corresponding feature maps from the encoder part, forming an interconnected U-shaped structure. Each layer in the decoder part sequentially contains a 2D transpose convolutional layer (sized 3x3) with ReLU activation and stride 2x2, followed by a Batch Normalization. The skip connections are concatenated with the corresponding feature maps from the encoder before each 2D convolutional layer in the decoder.

To finalize the output layer, a single 2D convolutional layer (1x1 filter size) with a TanH activation function and one filter to produce the final output. The model takes an image of sixty-four traces by sixty-four samples of time (RT domain) as input and predicts an output image of the same size.

A 2x2 Max Pooling layer means that the size of the feature maps is halved in each dimension after applying that operation. Max Pooling layers serve to downsample the feature maps, reducing their spatial dimensions while retaining the most important features. This downsampling, represented in Figure 3.4, helps to reduce computational complexity, improve computational efficiency, and enhance the network's ability to learn hierarchical features.

Data normalization and scaling ensure uniformity and compatibility across different layers of the network, while padding helps maintain consistent input dimensions throughout the architecture. The stride parameter means that in the Conv2DTranspose, the filter will move 2 pixels at a time, horizontally and vertically.

Convolutional layers apply non-linear transformations to the input data through com-

monly used activation functions in DL models. ReLU (Rectified Linear Unit) is a widely chosen activation function for its simplicity, computational efficiency, and effectiveness in training deep neural networks. By setting negative values to zero and leaving positive values unchanged, ReLU helps to capture complex patterns and relationships within the dataset during the training process, which may be challenging for humans to discern. Given the high number of parameters in the model architecture (Table 4.1), ReLU's computational efficiency, where it simply sets negative values to zero, is advantageous for faster training. The Hyperbolic tangent (TanH) is used in the output layer, and it can be appropriate depending on the specific requirements and characteristics of the task. Since the data preparation normalizes as scales the data before entering the training process, this activation function can be a good option.

Furthermore, all the 2D Convolutional and Max Pooling layers used He normal (He et al., 2015) initializer, which is common for DL models. This is the parameter used to initialize the weights randomly from a Gaussian distribution (mean of zero and standard deviation proportional to the square root of the number of input units).

Training deep NN is challenging due to the changing distribution of inputs in each layer during training since the input of one layer is the output of the previous layer. This slows down the training process, an issue called internal covariate shift Ioffe and Szegedy (2015), and the models with high nonlinearity become more challenging. Therefore the Batch Normalization (Ioffe and Szegedy, 2015) step has been used to address this problem by normalizing the layer inputs.

Droup out (Hinton et al. (2012), Srivastava et al. (2014)) is a way to better generalize a NN and avoid overfitting. This mechanism prevents nodes from over-specializing and serves as a form of regularization (James et al., 2023). During training only, it randomly ignores inputs by making them equal to zero and scaling the remaining inputs by the keep probability $(1 - dr)$ (Géron, 2019) when fitting the model. The dropout rate, dr, is its hyperparameter and it represents the probability of each input to be ignored, in this case 50%.

The integration of *TensorFlow* (Abadi et al., 2015) API facilitates the implementation and training of the U-Net model, providing researchers with a flexible and efficient framework for experimentation. Leveraging these powerful tools, researchers can explore various architectural configurations and optimization strategies to address complex problems in seismic processing and beyond. In this thesis, the architecture was not modified through the experiments to make fair comparisons among the tests.

Hyperparameter	**Description**
number of layers	9
dimension of input data	64x64
dimension of input in each layer	64x64, 32x32, 16x16, 8x8; 4x4; 8x8; 16x16; 32x32; 64x64
number of filters	32, 64, 128, 256; 512; 256; 128; 64; 32
2D convolutional layer filter size	3x3 (output layer 1x1)
Max Pooling 2D layer filter size	2x2
kernel initializer	He normal
activation function	ReLU (output layer: TanH)
stride	2
Dropout	rate of 0.5
optimizer	ADAM
learning rate for training	0.01
batch size for training	32
validation split	0.2
shuffle	True
dimension of output data	64x64

Table 3.1: U-Net and training hyperparameters.

Hyperparameter tuning significantly influences the performance and behaviour of the U-Net model and it is summarized in Table 3.1. All of the U-Net architecture and hyperparameters were kept the same throughout all the example tests in the following chapters.

3.2 Machine Learning and Seismic Processing

In the geophysics world, ML finds application primarily in two domains: interpretation and processing. In interpretation, the algorithms are fed with relevant data for prediction, often in the form of stacked seismic sections, which can be seen in ML as a 1D prediction. However, in

processing tasks, the success of ML has been somewhat limited. Despite extensive efforts, the field has yet to achieve significant breakthroughs in this area in the sense that the processing companies in the oil and gas industry have not fully used them. Nevertheless, there is an effort to comprehensively understand the applicability of ML in various seismic processing tasks, as mentioned in Chapters 1 and 2. This involves exploring the reasons behind the successes and failures of ML approaches to discern which problems are susceptible to ML solutions and which are not.

It is worth noting that while ML practitioners typically apply statistical techniques to seismic data, seismic processing involves a lot of physical tricks such as NMO correction, band limitation in seismic data, and multidimensional adjustments, which ML practitioners may not inherently understand. When selecting a problem to address using ML techniques, it is crucial to consider how it aligns with its corresponding ground truth. Thus, a nuanced understanding of the problem, its strengths and limitations is essential for making informed decisions in geophysical applications. This is the reasoning behind the choice of this thesis topic.

Extending this discussion to a specific challenge, consider the denoising problem in seismic processing. Despite having the codes and functions for the standard seismic processing workflow, in the case of trying an ML approach, there is a need to integrate that into the ML pipelines, and also produce labels (for the supervised learning cases). For example, having *C++* codes for RT and *Python* codes for the ML prediction are not necessarily easily connected into the same workflow. This is where the open-source Madagascar (Fomel et al., 2012) environment comes into play. It facilitates this workflow by combining *Python* as module-glue dataflow with standard DL tools like *Tensorflow*, therefore favouring the test of different inputs and label combinations (Trad, 2022), allowing this to be a suitable environment for experimentation. Overall, this thesis envisions incorporating the strengths of DL and seismic processing and understanding how the denoising process can be analyzed using both lenses.

An important aspect of the discussion will also involve the concept of using separate channels for different types of RT spaces. However, the practical challenge lies in ensuring that each channel is of the same magnitude, or at least close enough quantification for the ML model understanding.

As emphasized by Sun et al. (2020b), seismic data differs from conventional images in several aspects. Seismic data typically exhibits a dynamic range of $-3x10^4$ to $3x10^4$ and has a frequency content ranging from 5 to 100Hz, whereas conventional images have a dynamic range of 0 to 255 and cover a broader frequency spectrum (Sun et al., 2020b). Despite the potential of machine learning in seismic processing, challenges persist, as noted by Alkhalifah et al. (2022), especially concerning the implementation of neural networks with field data. Consequently, the utilization of synthetic data remains relevant for conducting tests.

3.2.1 Generating synthetic seismic data

When training a network, reference data is necessary. To have these ground truths easily generated as well as to have better control over the tests, synthetic seismic data were created to run experiments. To train the U-Net (Figure 3.1), I created synthetic data and labels using several forward modelling programs described in (Trad, 2022), using Finite Differences for ground roll examples and the convolutional method for the multiples examples. These modules were implemented in the Madagascar framework (Fomel et al., 2012), which permits combining standard seismic modules with DL libraries in a common data flow.

Acquisition parameters such as the number of receivers, shot interval, total record time, temporal sampling, wavelet and dominant frequency will be explained for each test in Chapters 4 and 5. Velocity models were also created using simple (velocity increasing with depth) or varying geology. In this thesis, I will show examples of multiples and ground roll, therefore the way this noise was added to the synthetic data varies and also will be further discussed in the chapters to follow. Figure 3.2 is an example of the generation of synthetic data for the case of multiple. Using the velocity model (a), the generation of shot gathers (convolutional

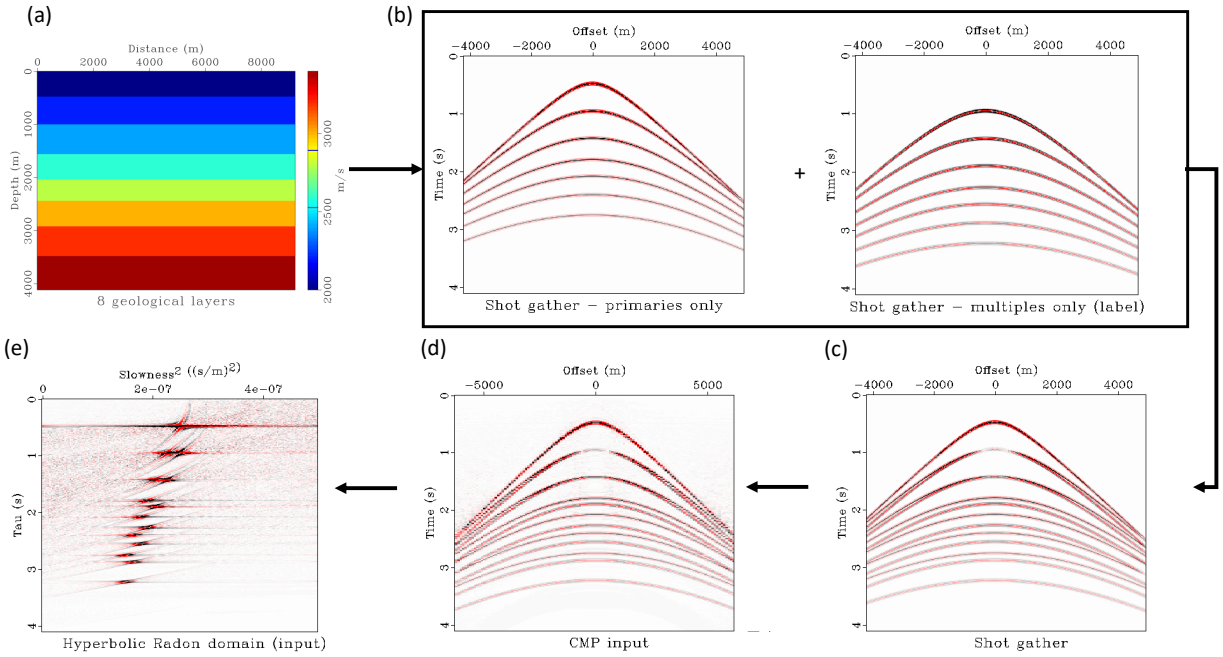


Figure 3.2: Schematic representation of the generation of synthetic seismic: an example of data with multiples. The velocity model (a), with increasing velocity from top to bottom, is used as an input in the convolutional model algorithm to generate multiples as primaries in separate shots (b). These are then concatenated to (c) the shot with multiples and primaries, then sorted by (d) CMP. The last step is to apply the RT to have the (e) RT panel, where the multiples are aligned to the right and primaries are to the right.

model) with reflections and multiples separately (b) is possible, making it possible to create the input data (c). Then, these shot gathers (c) are sorted by CMP (d), and an RT, in this case HRT, is applied to obtain the HRT images (e).

Depending on which domain is more convenient, and the type of RT being used, sorting the shots by common midpoint (CMP) was necessary, and then the desired Radon (linear, parabolic, hyperbolic or hybrid) transform was applied. Furthermore, the RT panel of the label was also generated for the training; for each noise, this was done differently and will be further explained in Chapters 4 and 5. The RT domain images have τ (intercept time) and s^2 (slowness squared) axis, and these get passed through data preparation steps.

3.2.2 Data preparation

Much of the ML data pipeline involves extensive data preparation or data cleaning, which is similar to the preprocessing stages in seismic processing (Trad, 2022). It is a really important step in both workflows to provide the quality and reliability of the subsequent data analyzes. Although the construction of the network typically varies depending on the specific task, data generation and network training are typically universal processes across various domains (Yu and Ma, 2021).

In ML projects, preprocessing steps such as normalization and scaling are important techniques used before feeding a dataset into an ML model, ensuring the data is appropriately prepared for the learning process, especially while using optimization processes like gradient descent. When features are scaled or normalized, they converge faster as they help the optimization algorithm minimize the loss function more efficiently, leading to faster convergence.

Normalization, also known as standardization, involves transforming the dataset's features to a standard scale. I used a normalize function in Python, which computes the mean and standard deviation of the input data and then scales the data to have a mean of zero and a standard deviation of one. This process ensures that all features contribute equally to

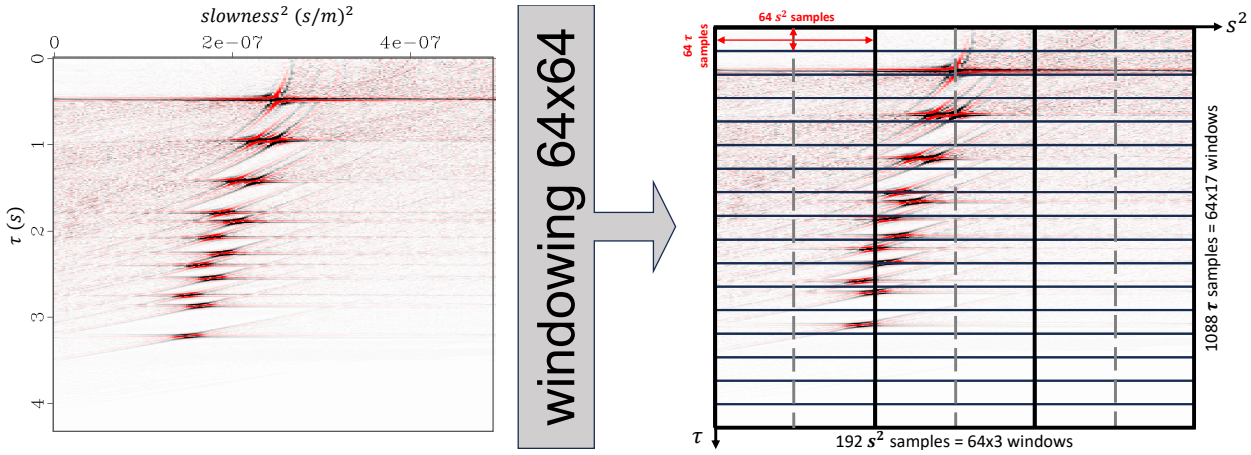


Figure 3.3: Schematic representation of the windowing process: the HRT panel is subdivided into patches of 64 (s^2 axis) by 64 (τ axis) samples. The grey dashed lines represent the overlapping factor. In this specific example, this RT image will have 51 times 4 patches of 64x64 images, therefore, a total of 204 windows.

the model training process and prevents features with larger scales from dominating those with smaller scales. Normalization is particularly important when the features have different units or scales, which is relevant while using different channels of information.

Scaling, however, involves adjusting the range of the features without changing their distribution. I used a scaling function in Python known as percentile scaling, which involves centring the data around its median, clipping outliers beyond the specified percentile threshold (pclip=99), and subsequently normalizing the clipped data to fit within the range [0, 1] by dividing by the maximum value. This multi-step process ensures that the data is appropriately scaled for model training, contributing to improved numerical stability and performance.

The DL workflow begins by reading seismic data in the RT domain, as well as input and label data, followed by normalization and scaling. Statistics parameters like mean, standard deviation, maximum value, and pclip were saved to facilitate data restoration post-prediction. Subsequently, the normalized and scaled data undergoes a windowing (Figure 3.3) process, a crucial step before inputting it into the U-Net model. The windowing process partitions RT panels into smaller patches, also known as the input size (Table 3.1), that the network

processes at each iteration, enhancing memory efficiency. As DL models often require large amounts of data for effective training this technique enables the capture of features effectively while also addressing irregular geometry acquisition challenges.

Windowing not only enables NNs to capture local context within seismic data, allowing them to focus on specific geological features or structures present in each window, but also preserves the spatial information inherent in the data. By maintaining spatial relationships between data points across different windows, this technique facilitates accurate analyzes of seismic data. For instance, when the seismic source is not centred in the image or irregular geometry acquisition poses a challenge, windowing helps mitigate these issues. The chosen window size of 64x64 in the RT panel ensures uniform sampling, with each window representing 64 samples of transformed time (τ) and 64 samples of the ray parameter (s, p or q).

The chosen window size of 64x64 in the RT panel ensures efficient processing (enables the U-Net downsampling), although careful attention is required to ensure all data samples are appropriately windowed. It is important to perform this process meticulously, as samples inputted for training must be a multiple of 64 to ensure all data is properly windowed and seen by the network (Figure 3.3). While this adjustment can be made in synthetic data, ensuring alignment with the window size in field data can be challenging.

3.2.3 Training process

The training process is when the machine learns the input data patterns based on its labels. The network refines its predictions by adjusting the weights associated with each connection between neurons to optimize the predictions' efficiency. The training pipeline for the tests (Chapters 4 and 5) used *TensorFlow* libraries. To guide the training process, the U-net model is compiled with the optimizer, batch size, shuffle, validation split and learning rate as described in Table 3.1. The upcoming chapters will specify and discuss the loss function, metrics and number of epochs (iterations) used for each test.

Batch size is the number of samples the model processes in each epoch. A batch size of 32 is often chosen because it strikes a balance between memory usage and training efficiency on GPUs commonly employed in DL tasks, while still providing enough samples to effectively update the model parameters. Over time, researchers have found that a batch size of 32 tends to perform well across various DL tasks and architectures (Bengio, 2012). Furthermore, knowing that the windowing process was done using 64x64, the training set was divided into batches, each containing 32 samples. These batches are sequentially fed into the model for computation of gradients and updating of the model's parameters (weights and biases) through backpropagation.

The windowing process extracts 64x64 patches, with an overlap of 32 samples (in both x and y dimensions), from the input data, as seen in Figure 3.3. This means that each window contains a 64x64 pixels area of the original data in an overlapping manner. Furthermore, during each training iteration (epoch), the model processes a batch of 32 images.

The convolution kernel or filter represents the weights that connect the units within the feature map, and once a feature is identified, its precise location becomes less critical, and only its general position relative to other features matters (LeCun et al., 1998). The order of samples in each batch is randomized (shuffle), preventing the model from learning any sequential patterns that might exist in the data. By making shuffling the training set before each epoch prevents the learning from being biased by the order of samples, promotes better generalization, makes the model learn from a diverse set of examples in each epoch, and prevents the model from overfitting to specific patterns present in the training data.

A layer filled with neurons employs the same filter, generating a feature map. The latter will highlight the areas in an image where the filter is the most active (Géron, 2019). Throughout the filters that compose each convolutional layer in the U-Net training phase, each layer will naturally acquire useful filters for its prediction, while consecutive layers learn to integrate this information into complex patterns.

Each filter produces a feature map by convolving the filter across the input data. In

this process, all neurons in a feature map share the same set of weights (the filter), but their outputs differ based on the region of the input data they process. A convolutional layer usually includes multiple filters, each producing a different feature map with its unique set of weights. This architecture enables the extraction of multiple features at each spatial location. While filters contain valuable network information, they are relatively small. In contrast, feature maps are larger and offer insights into the network by representing intermediate components of the NN.

Figure 3.4 illustrates one feature map of each one of the nine convolutional layers of the U-Net architecture in Figure 3.1, which used RT images as input data (windowed 64x64). The deepest layer, called latent space, can be considered a compressed version of the information. In the latent space, the information continuously makes sense by putting important information close to each other in such a way that the network can move from one type of information to another and all the information that navigates through is meaningful.

The training set is utilized to train the model, while the validation set assists in adjusting hyperparameters and monitoring the model's performance throughout training. The *TensorFlow* validation split parameter determines the portion of training data reserved for validation during model training. A validation split of 0.2 allocates 20% of the training data for validation, with the remaining 80% used for training. After each training epoch, the model's performance is tested on the validation set to indicate its learning and generalization capabilities. Once all the epochs are completed, the trained model is saved to a specified file for use while predicting.

3.2.4 General workflow - applying the U-net and RT to attenuate seismic noise

After training, the U-Net model *TensorFlow* can save the model's architecture, weights, biases and optimizer. That is what the machine has learned about the provided training and validation sets. Once all of these parameters are saved, they can be used to make predictions.

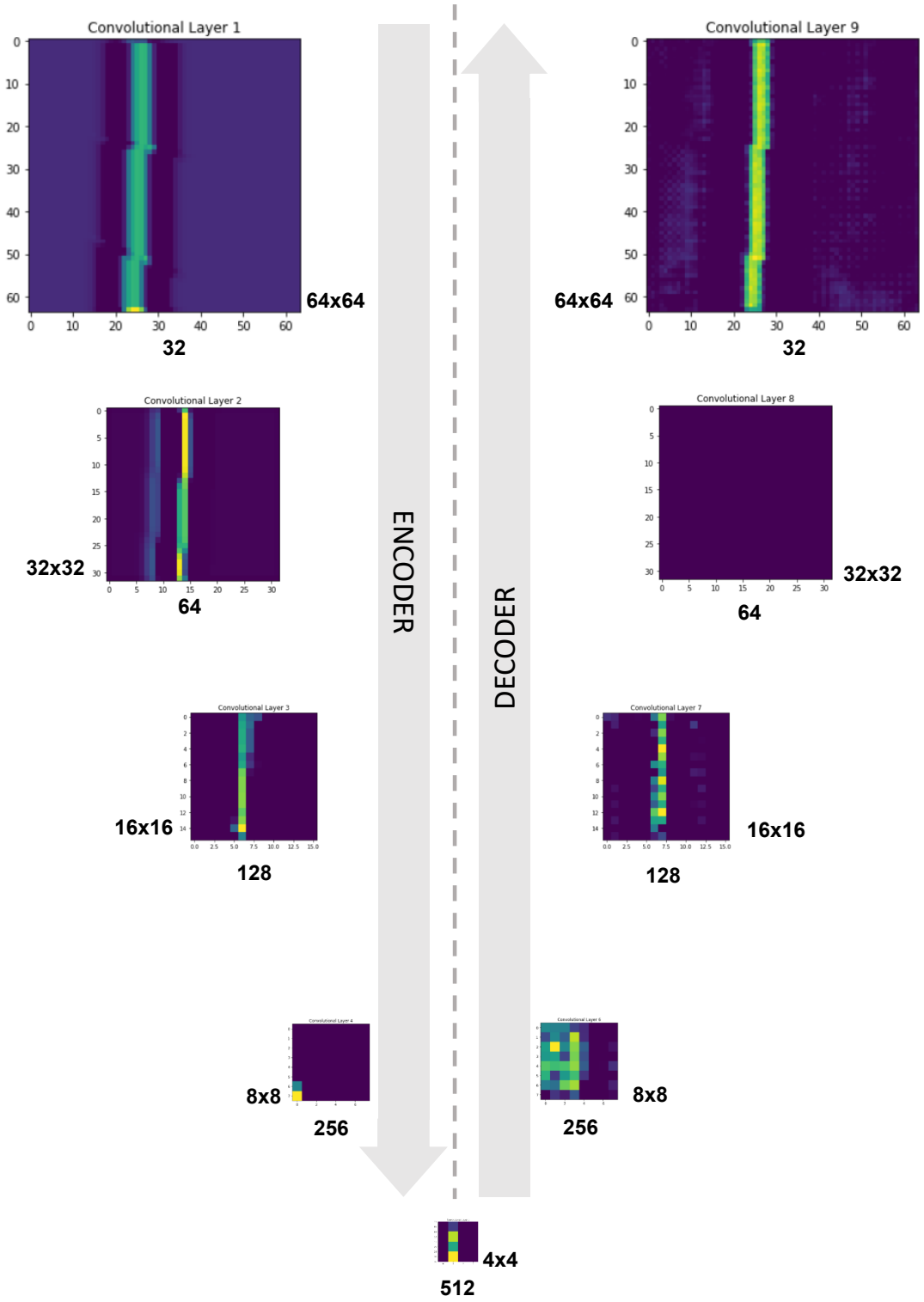


Figure 3.4: Schematic representation of the resulting feature map of each 2D convolutional layer. The convolutional layer 1, for instance, has a total of 32 feature maps, whereas the convolutional layer 5, also known as latent space, has 512 feature maps.

To predict the images in the RT domain, all of the U-Net architecture hyperparameters mentioned in table 3.1 were kept the same throughout all the example tests in the upcoming chapters.

Figures 3.5 and 3.6 exemplify the general workflow I will use in the upcoming chapters for multiples and ground roll testing, respectively. The synthetic data are generated, or field data are loaded, and in each case, the corresponding label is generated. This seismic data, shot gathers for ground roll workflow (Figure 3.6) or sorted by CMP for the multiples workflow (Figure 3.5), undergo the selected RT (linear, hyperbolic, parabolic or hybrid).

The RT input and RT label then go under the data preparation processes: normalization, scaling and windowing. These 64x64 patches of data are entered into the U-net to be trained. The prediction of denoised RT (Figure 3.6) or the noise-only RT (Figure 3.5) is made. The last step of the workflow is to take the predicted RT and apply the inverse RT using Least Squares or simply the adjoint RT to reconstitute the data from the Radon domain to the shot domain (Figure 3.6) or the CMP domain and then back to the shot domain again (Figure 3.5).

Note that the main difference between these two workflows is that the noise acts differently depending on its domains. While using RT for multiple, this noise is better analyzed in the CMP domain, whereas ground roll is coherent in the shot domain and, therefore, better. Figures 3.5 and 3.6 are only generic workflows, but some variations of these were applied and will be further explained in the tests section (Chapters 4 and 5).

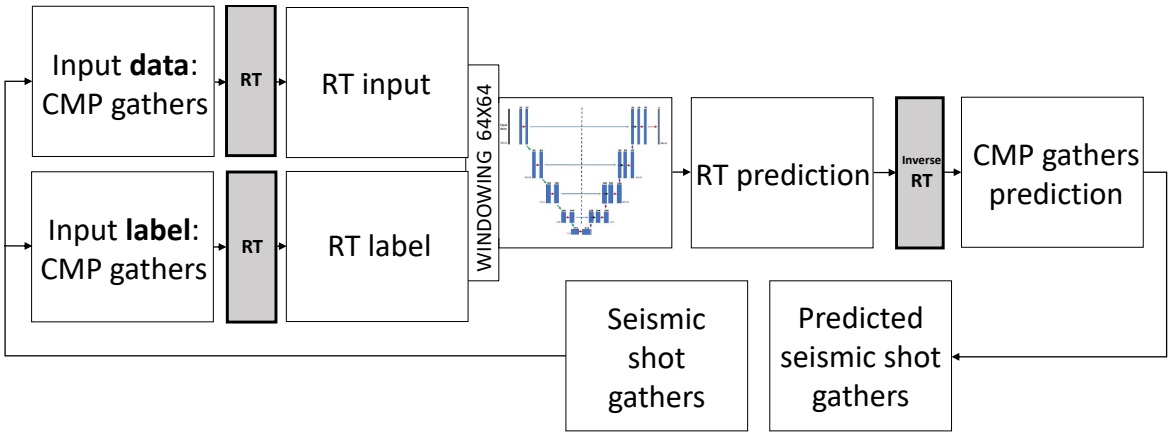


Figure 3.5: Multiples general workflow for numerical experiments.

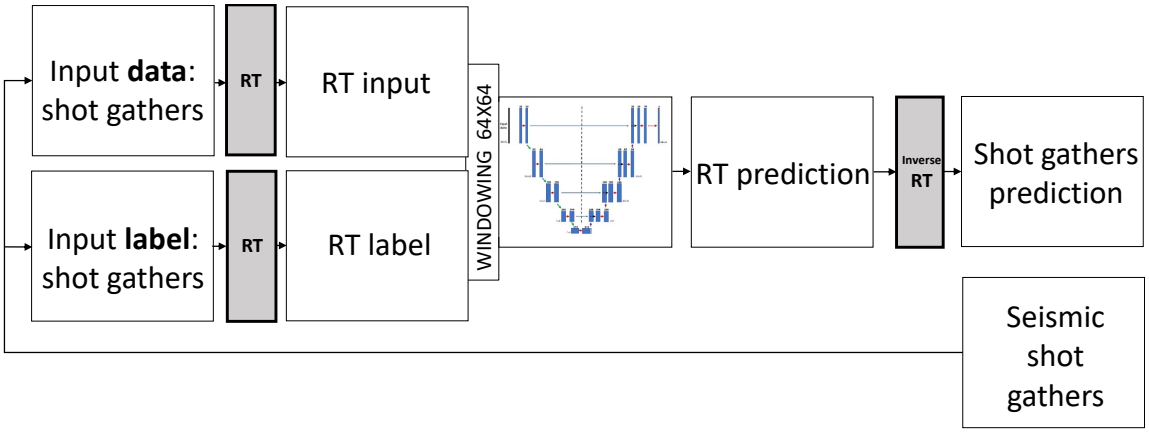


Figure 3.6: Ground roll general workflow for numerical experiments.

Chapter 4

Training and prediction - multiples

In Chapter 2, I introduced various methodologies used to attenuate multiples, a common noise type in marine seismic data. In Chapter 3, I explained a general workflow using images of RTs to produce predictions through DL. This chapter presents the results of applying the workflow (Figure 3.5) shown in Chapter 3 to predict this coherent noise.

With its flexibility, DL can offer a methodology that helps understand noise nonlinearity while incorporating vast amounts of not-well-behaved data. Unlike classical processing, which relies on the physics of the events, in this chapter, I used seismic CMP gathers represented in the RT domain as images to train the network to find patterns and make predictions based on the data used for training.

The target is to address the more easily detectable surface multiples, where energy from the source descends, strikes an interface, rebounds, reaches the surface, and is recorded. Essentially, this creates another source of reflectivity, resulting in a mix of double-bounced energy, leading to longer travel times. As this thesis focuses on an initial DL analyzes, it is relevant to mention that internal multiples will not be covered here due to their complexity.

In seismic data processing, the characteristics of the data dictate the workflow and methodologies used to address noise-related challenges. During the pre-stack stages, the focus is put on identifying low-velocity energy clusters within the velocity analyzes spec-

trum. Unlike primaries, multiples exhibit periodic behaviour (Taner, 1980) and as seen in Chapter 2, various techniques, including RT, are able to predict and attenuate multiples based on that.

Basis functions display different appearances in the RT domain, guiding their application for specific tasks. In the shot domain, multiples and primaries share a hyperbolic shape, whereas, in the CMP NMO-corrected domain, primaries flatten out while multiples display an approximately parabolic moveout. The first refers to the Hyperbolic RT, and the latter is what Hampson (1986) called Parabolic RT. This periodicity, with multiples showcasing a larger moveout compared to primaries, helps in their separation within RT spaces.

In this chapter, I further describe the synthetic data generation process of the data used. The purpose of training an NN is to learn the weights and biases and use backpropagation until satisfactory results are achieved. However, the main focus of this discussion lies not on hyperparameter selection but rather on the enhancement achieved by employing non-sparse in comparison to sparse seismic RT while using DL for the prediction of multiples, despite the standard geophysical applications favouring sparse RT.

4.1 Hyperbolic RT tests with synthetic seismic - multiple channels and inference

To better understand the U-Net's performance in multiple attenuation, tests using various Hyperbolic RT (HRT) panels were conducted and analyzed. Initially, some geological models were used to gain insight into the inference process. Synthetic data generated from simplified earth models served as the groundwork for these experiments, allowing for controlled manipulation of the types of multiples present in the data. Yu et al. (2019), for example, demonstrated the feasibility of demultiple applications using synthetic seismic gathers for training.

Two distinct datasets were prepared: one containing both primaries and multiples and

the other with only multiples. These datasets were then transformed into RT panels. The dataset containing both multiples and primaries served as inputs, while the multiples-only panels were used as labels. The network was trained using these to differentiate between primaries and multiples. Subsequently, the inverse RT was applied to generate RT panels with predicted multiples. A last step should be taken to have the multiple attenuated version, which is to subtract the predicted multiple from the input data, leaving only the primary reflections.

To train the U-Net architecture (Figure 3.1), I used implementations of the convolutional model to generate 120 synthetic shot gathers from each velocity model (Figures 4.1a, 4.2a and 4.3a). Each shot gather comprised 380 receivers, with a shot interval of 10 meters and a total record time of approximately 4.3 seconds, with a sample rate of 0.004 seconds, resulting in 1088 samples per trace.

Taking the example from Figure 3.2, the workflow involves generating shots containing only primary reflections, followed by separate shots containing only first-order multiples. These data sets were then combined in the shot domain to simulate acquisition scenarios capturing both primaries and their first-order multiples. Figures 4.1b, 4.2b, and 4.3b show shot 98 from each velocity model. It is important to recognize that certain primaries and multiples almost overlap in the shot domain (Figure 4.4). Consequently, they will be closer together in the HRT domain, which is something to keep close attention to ensure the network can discern and accurately map these distinctions.

Subsequently, I sorted them by CMP (Figures 4.1c, 4.2c and 4.3c) with a total of 578 CMPs and the HRT was applied (Figures 4.1d, 4.2d and 4.3d). In the HRT panels, primaries and multiples are separated, with primaries aligned to the left and multiples to the right since they have higher travel times. I also generated the labels for training, which are 578 HRT panels of multiples only (Figures 4.7e, 4.9e and 4.10e). Then, these 578 HRT images (1088 (τ) x 192 (s^2)) passed through the data preparation steps explained in Chapter 3, having 64x64 patches as the input images. These are the data inputted (input and label)

into the U-Net (Figure 3.1) for training and prediction.

After prediction, the network output will be the patches of the 578 HRT images with the predicted multiples. The last step of the workflow is to apply the inverse HRT using Least Squares to reconstitute the data back to the CMP domain (and, after, be able to sort by shot). Figures 4.5 and 4.6 summarize the workflows used in the following numerical examples.

Table 4.1 presents a summary of the U-Net model architecture generated using *TensorFlow*, providing information about the number of parameters in each layer. The model begins with an input and output size layer (64, 64, 1), representing a 64x64 image. The total number of parameters in the U-Net model for the example shown in Figure 4.5 contains a total of 3,839,841 parameters, with 3,835,937 parameters being trainable and 3,904 parameters being non-trainable. This summary helps in understanding the complexity of the model architecture and, consequently, its capacity to learn from the input data.

Various tests were conducted using different HRT panels to gain deeper insight into the U-Net's performance in predicting multiples. Initially, the inference process was explored using different geological models (Test I) to understand the U-Net's behaviour during inference.

- Train and predict with HRT for 3 geological layers;
- Train with 3 geological layers and predict for 5 geological layers, both with HRT;
- Train with 3 geological layers and predict for 8 geological layers, both with HRT;
- Train with 3 and 5 geological layers and predict for 8 geological layers, both with HRT;
- Train with 3, 5, and 8 geological layers and predict for 8 geological layers, both with HRT.

Next, to assess the advantages of applying higher-resolution HRT, tests were conducted using sparse HRT. Furthermore, the network was used with two channels (workflow in Figure

4.6), sparse and non-sparse HRT panels, to further constrain the training and comprehend how input labels influence the training process (Test II):

- Training was performed with 3 and 5 geological layers, predicting for 8 geological layers, both using sparse HRT;
- Training with 3, 5, and 8 geological layers, predicting for 8 geological layers, all using sparse HRT.
- Training with 3, 5, and 8 geological layers, predicting for 8 geological layers, using sparse and non-sparse HRT as input channels, with non-sparse HRT as the label;
- Similarly, training sessions with 3, 5, and 8 geological layers, predicting for 8 geological layers, utilizing sparse and non-sparse HRT as input channels, with sparse HRT as the label.

4.1.1 Test I: Inference learning

After analyzing the data and choosing the U-Net model, the training phase starts by using a training dataset. Throughout this process, the learning algorithm adjusts the model's parameters to minimize a predetermined cost function (in this case, MSE), thereby enhancing its predictive capabilities. With the model appropriately trained, it is then ready to make predictions on new cases, a step commonly referred to as inference.

Inference is a method that involves using a trained model to make predictions on new data. This section will show the results of training the U-Net and saving the weights for its best-predicted model. This analyzes sheds light on how the network builds on previously learned knowledge to predict multiples effectively. Throughout the examples presented, the focus remains primarily on a qualitative analyzes.

First, I trained the network with a simple model having three geological layers (Figure 4.1a using 120 shots, which will then be turned into 578 HRT, following the workflow on

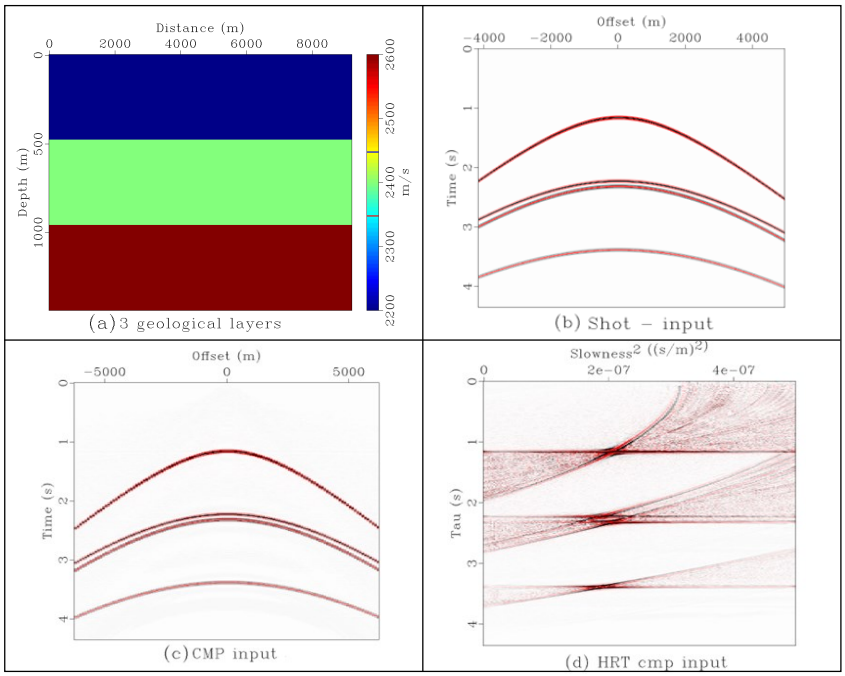


Figure 4.1: (a) Velocity model using 3 geological layers with a thickness of 160 meters each. (b) Shot 98 with primaries and multiples and (c) after sorting by CMP. (d) HRT panel, after applying the HRT operator in the CMP.

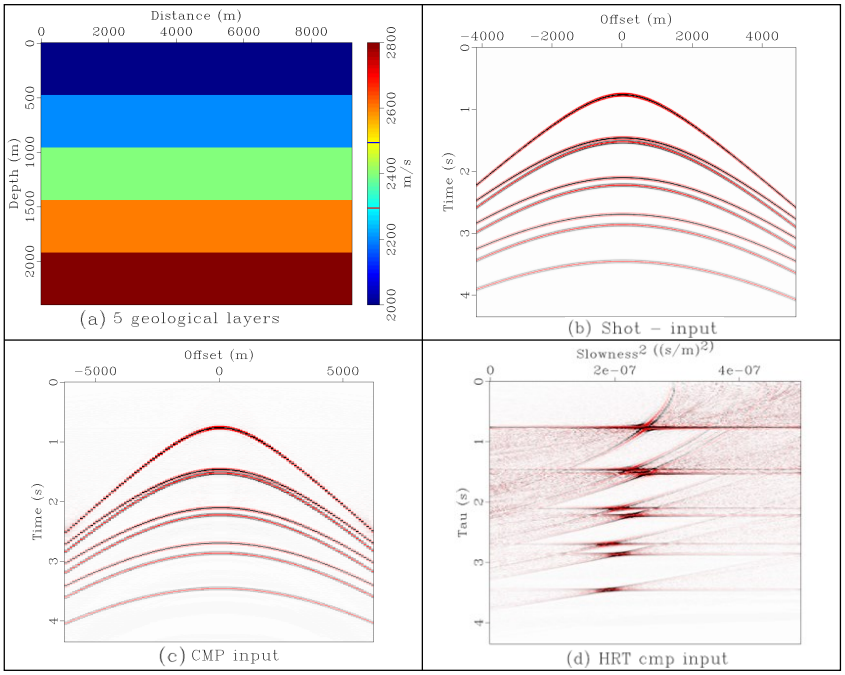


Figure 4.2: (a) Velocity model using 5 geological layers with a thickness of 96 meters each. (b) Shot 98 with primaries and multiples and (c) after sorting it by CMP. (d) HRT panel, after applying the HRT operator in the CMP.

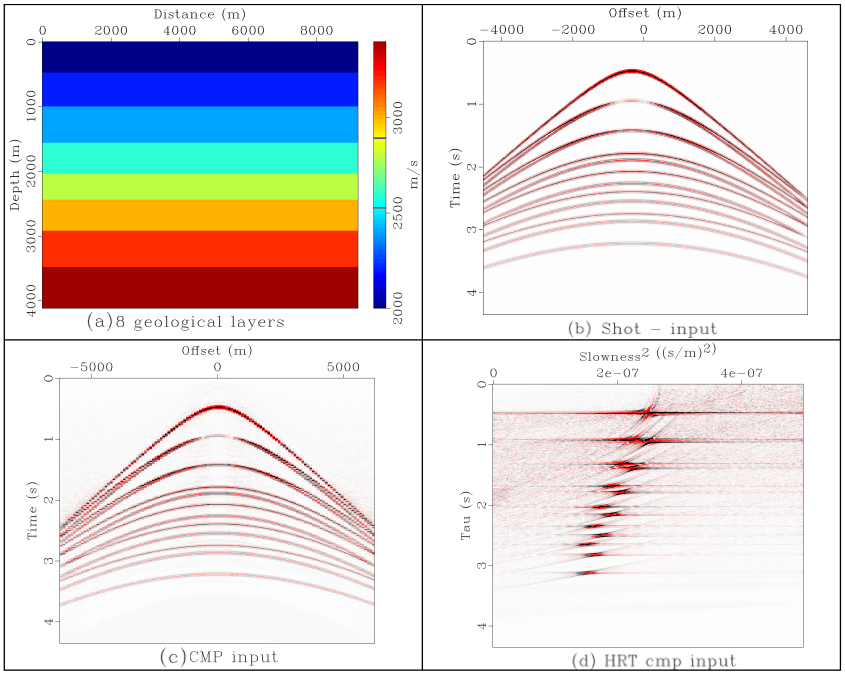


Figure 4.3: (a) Velocity model using 8 geological layers with a thickness of 60 meters each. (b) Shot 98 with primaries and multiples and (c) after sorting it by CMP. (d) HRT panel, after applying the HRT operator in the CMP.

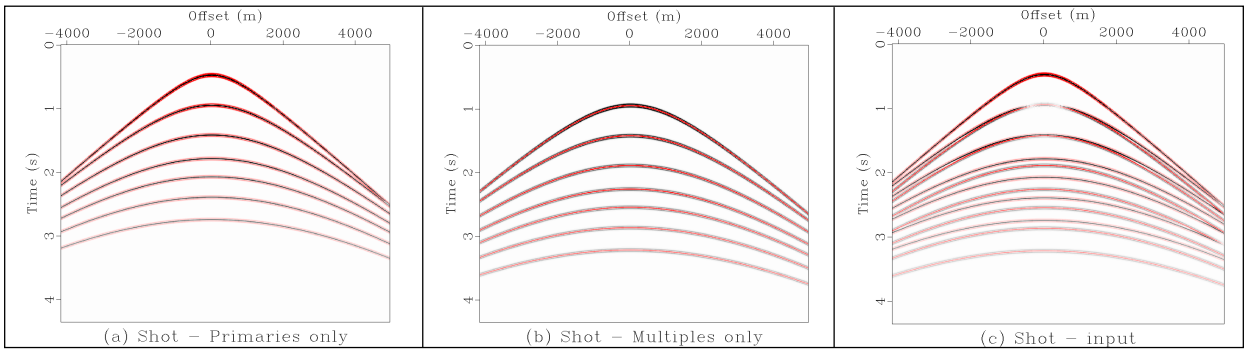


Figure 4.4: Example of 8 geological layers earth model: (a) Shot 98 with primaries only, (b) with first-order multiples only, and (c) with multiples and primaries, having some events overlapping

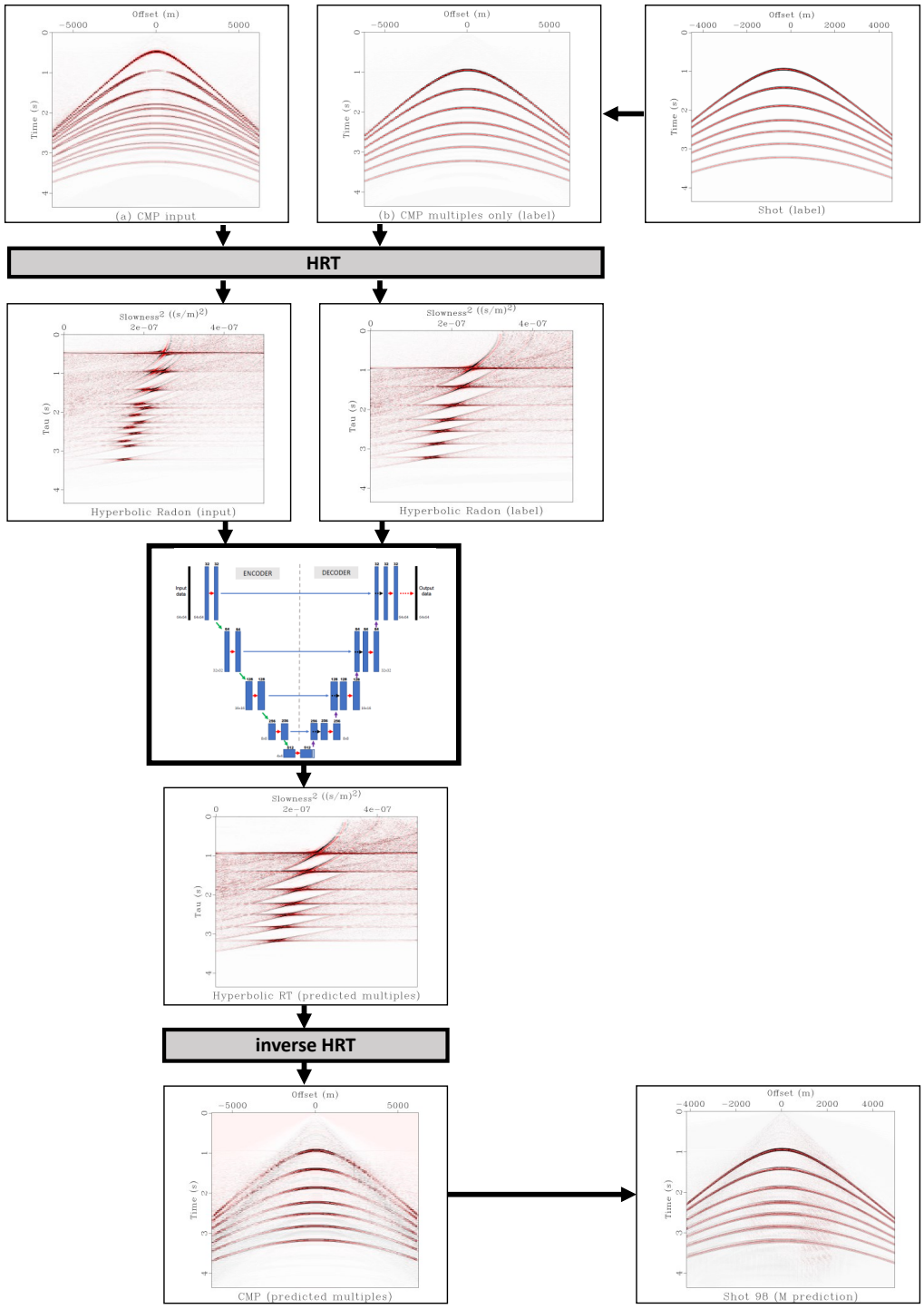


Figure 4.5: Workflow of the 8 geological layers case using 1 channel. Synthetic shot gathers are sorted by CMP, resulting in the input data (a), with multiples and primaries and the input label (b), with multiples only. Then, the HRT (inverse operator) is applied to generate the hyperbolic Radon panels of the input (c) and labels (d) to feed the U-Net (e). The network then predicts, after training, the HRT panels of only multiples (f). The inverse HRT is then applied to return the data to the CMP domain (g).

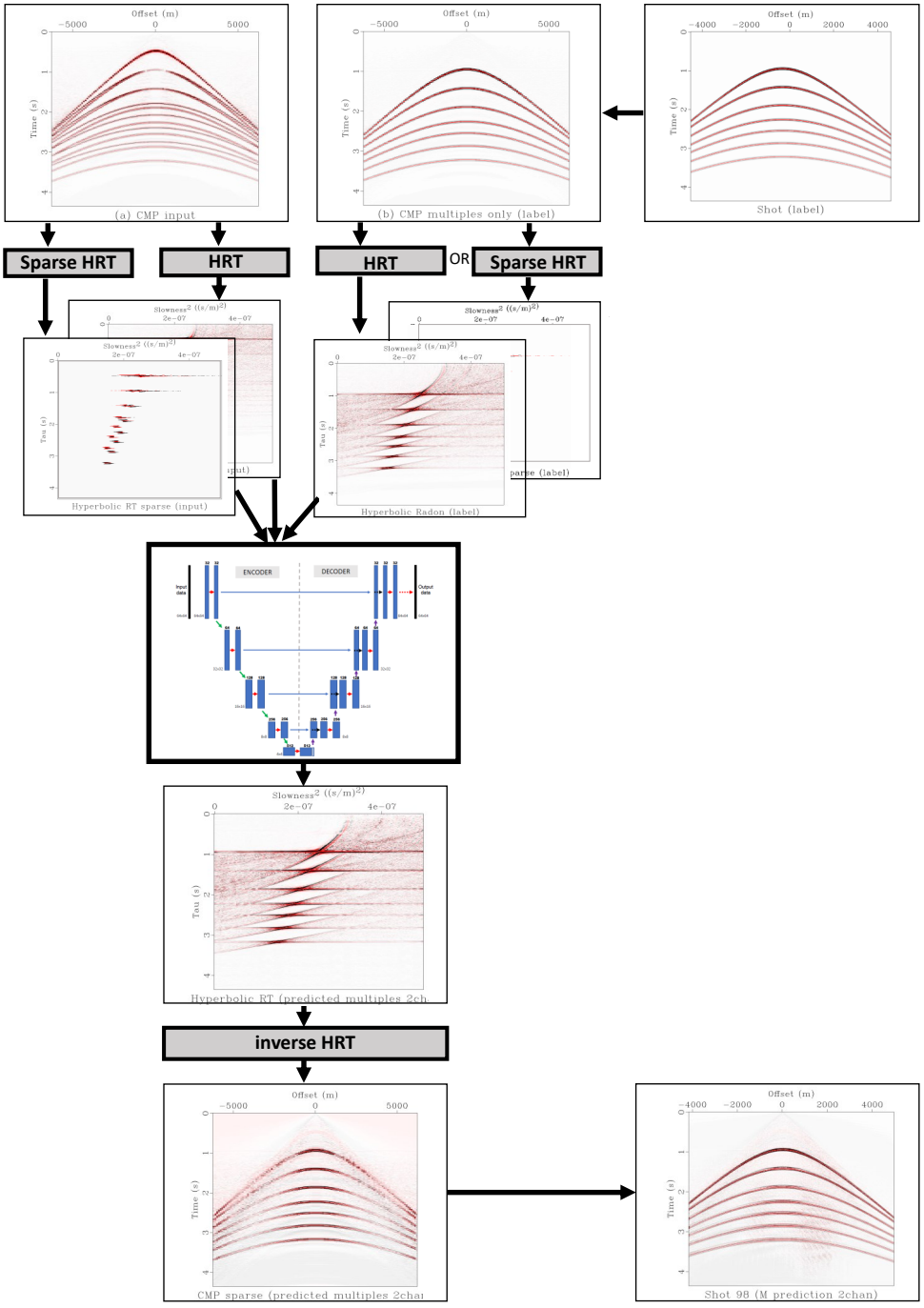


Figure 4.6: Workflow of the 8 geological layers case using 2 channels. Synthetic shot gathers are sorted by CMP, resulting in the input data (a), with multiples and primaries and the input label (b), with multiples only. Then, the HRT is applied to generate the HRT panels for input (c) and labels (d). Also, the sparse HRT is applied to generate the sparse HRT panels of the sparse input (c') and sparse labels (d'). The input data and one of the labels (in this case, non-sparse) will be fed into the U-Net (e). The network then predicts, after training, the HRT panels of only multiples (f). The inverse HRT is then applied to return the data to the CMP domain (g).

Layer (type)	**Output Shape**	**Number of Parameters**
InputLayer	(None, 64, 64, 1)	0
Conv2D	(None, 64, 64, 32)	320
BatchNormalization	(None, 64, 64, 32)	128
MaxPooling2D	(None, 32, 32, 32)	0
Conv2D	(None, 32, 32, 64)	18496
BatchNormalization	(None, 32, 32, 64)	256
MaxPooling2D	(None, 16, 16, 64)	0
Conv2D	(None, 16, 16, 128)	73856
BatchNormalization	(None, 16, 16, 128)	512
MaxPooling2D	(None, 8, 8, 128)	0
Conv2D	(None, 8, 8, 256)	295168
BatchNormalization	(None, 8, 8, 256)	1024
MaxPooling2D	(None, 4, 4, 256)	0
Conv2D	(None, 4, 4, 512)	1180160
BatchNormalization	(None, 4, 4, 512)	2048
Dropout	(None, 4, 4, 512)	0
Conv2DTranspose	(None, 8, 8, 256)	524544
BatchNormalization	(None, 8, 8, 256)	1024
Concatenate	(None, 8, 8, 512)	0
Conv2D	(None, 8, 8, 256)	1179904
BatchNormalization	(None, 8, 8, 256)	1024
Conv2DTranspose	(None, 16, 16, 128)	131200
BatchNormalization	(None, 16, 16, 128)	512
Concatenate	(None, 16, 16, 256)	0
Conv2D	(None, 16, 16, 128)	295040
BatchNormalization	(None, 16, 16, 128)	512
Conv2DTranspose	(None, 32, 32, 64)	32832
BatchNormalization	(None, 32, 32, 64)	256
Concatenate	(None, 32, 32, 128)	0
Conv2D	(None, 32, 32, 64)	73792
BatchNormalization	(None, 32, 32, 64)	256
Conv2DTranspose	(None, 64, 64, 32)	8224
BatchNormalization	(None, 64, 64, 32)	128
Concatenate	(None, 64, 64, 64)	0
Conv2D	(None, 64, 64, 32)	18464
BatchNormalization	(None, 64, 64, 32)	128
Conv2D	(None, 64, 64, 1)	33

Table 4.1: U-Net Model summary using TensorFlow helps us to understand the size of a simple training process. For instance, in the first test, training and predicting three geological layers were done, and the network had a total of 3839841 parameters, with 3835937 trainable and 3904 non-trainable parameters.

Figure 4.5. Figure 4.7a shows shot 98, and Figure 4.7b shows the same shot with multiples only (label). Something to notice is that a multiple nearly overlaps a primary reflection (Figure 4.7a); therefore, the HRT helps to separate multiples and primaries, making it important for the network to learn that these represent different events. The U-Net was trained using the three geological layers HRT models (Figures 4.7d and 4.7e) and made predictions for three geological layers case, with results shown in Figure 4.7f, having the two multiples mostly predicted.

However, it is important to note that there are a lot of artifacts, known as the butterfly effect, on the left and right sides of the HRT panels, which result from poor sampling and limited aperture on the CMP domain. They also need to be predicted by the network, not only the punctual information (in this case localized in the center of the panel) representing the multiples. When returning to the CMP and then the shot domain (Figure 4.7c), the HRT needs this information to reconstruct the data. The U-Net did not fully predict these artifacts as they are not coherent, and consequently, the shot domain contains some undesired information, as seen in Figure 4.7c.

A cost function, often called loss, is commonly used to evaluate the model performance by measuring the error between predicted and true output. Figure 4.8 illustrates the Mean Squared Error (MSE) values across 20 epochs for both the training (in blue) and validation (in orange) datasets. By inspecting the loss curve, it is possible to gain insights into the trend of the loss function over time. Ideally, the aim is for a decreasing trend that minimizes the loss in both training and validation, which indicates effective learning and generalization by the model.

Throughout 20 epochs, the MSE decreased between the first and last predictions, as seen in Figure 4.8a. Despite occasional peaks in the validation portion, the overall trend of the error decreased. This test demonstrates the model's ability to predict a simple geological scenario despite occasional outliers observed during validation predictions. These outliers may be attributed to artifacts present in certain data patches, such as the butterflies on

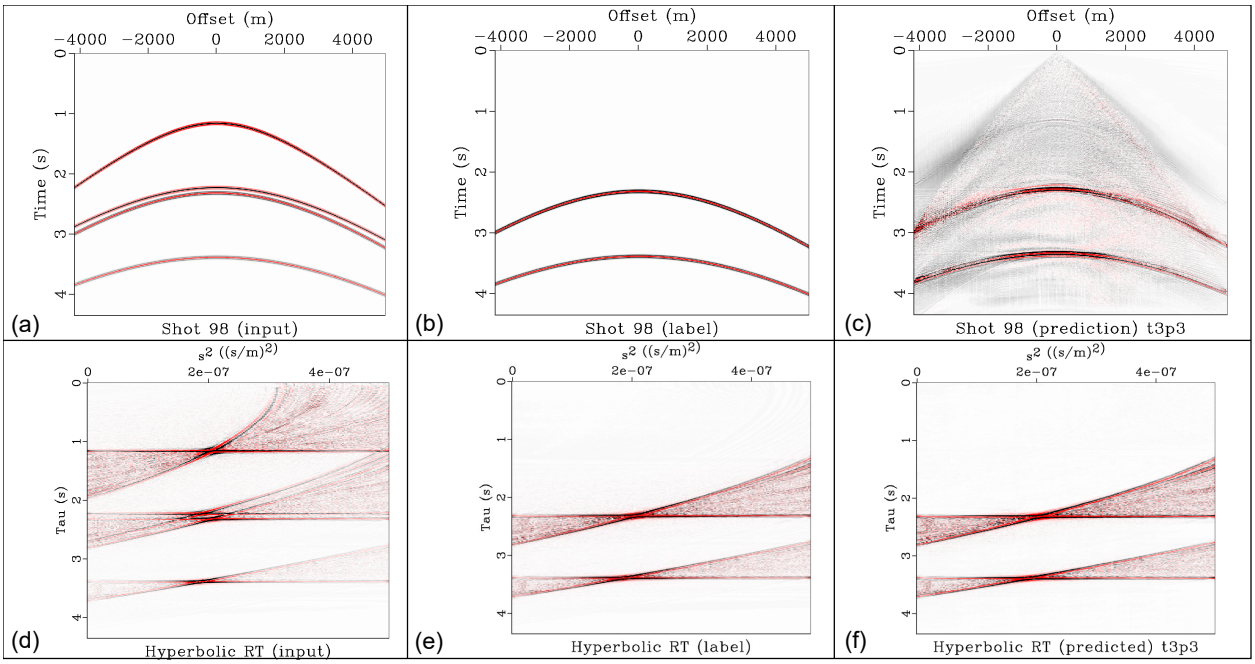
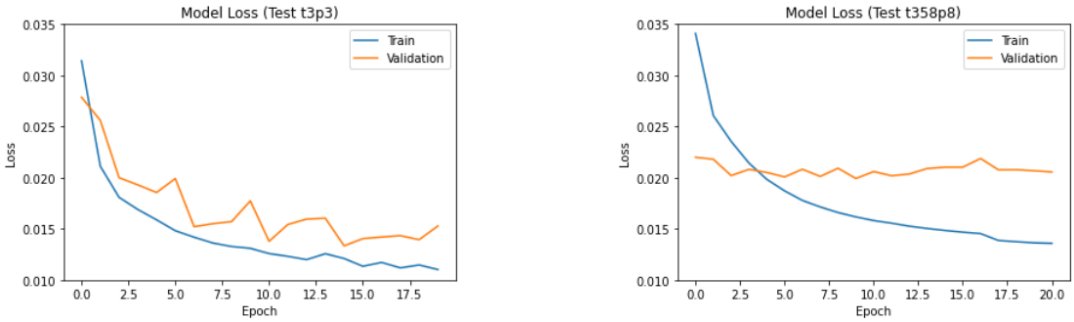


Figure 4.7: Three geological layers case: shot 98 with multiples and primaries (a), just with multiples (b) and after the U-Net prediction using the three geological layers training, subsequent inverse HRT and sorting by shot. HR with multiples and primaries (d), just with multiples (e) and after the network prediction using the three geological layers training (f).

the right and left sides of the HRT panels, which can challenge the U-Net's learning and prediction process. Although the MSE consistently decreased as expected, extending the training with more epochs could potentially further reduce the error, but it would also increase the time of prediction and perhaps overfitting.

Then, using the weights derived from training predictions were made for a five-layer geological model. Here, the three-layer geological model served as the training and validation sets, while the five-layer case was used as the test set. Figure 4.9f illustrates the prediction of multiples in certain regions of the HRT panel. This demonstrates that through prior training on a simpler model, the network could approximate the prediction of multiples without explicit training on the new geological model.

Nonetheless, it is important to note that there are many artifacts on the HRT panels (Figure 4.9f). These artifacts, along with the punctual information representing the multiples,



(a) (b)

Figure 4.8: U-Net loss curve: mean square error (MSE) of the overall prediction and the validation portion of the data. (a) Case of training in the 3 geological layers data, (b) training with 3,5 and 8 geological layers data.

must be predicted by the network. The HRT also relies on this information to reconstruct the data when returning to the CMP and then the shot domain. However, the U-Net could not fully predict these artifacts, resulting in undesired features. In addition to the U-Net not having been trained on a more complex geological model yet at this point, the windowing process may also contribute to this limitation, as the network fed 64x64 patches rather than the entirety of the HRT panels. This can also be seen in Figure 4.9c since some background information (butterfly effect) remains in the shot domain from some primary reflections, some in the short and others in the long offset.

To achieve better predictions, additional information for training is likely needed. Another test was performed (Figure 4.10) to assess whether the network could predict the eighth geological layers model solely based on the training with three geological layers. In this instance, it is evident that a primary reflection overlaps with a multiple (as shown in Figure 4.10a) in short offsets. This observation is also apparent on the HRT panel (Figure 4.10(d)) around $\tau = 1$ and $\tau = 1.5$, but I expect that the network will be capable of distinguishing them. Figure 4.10f illustrates the prediction of some multiples, although certain artifacts remain unresolved. Additionally, in the shot domain (Figure 4.10c), remnants of some primaries persist, despite not being explicitly evident in Figure 4.10f, showing that the

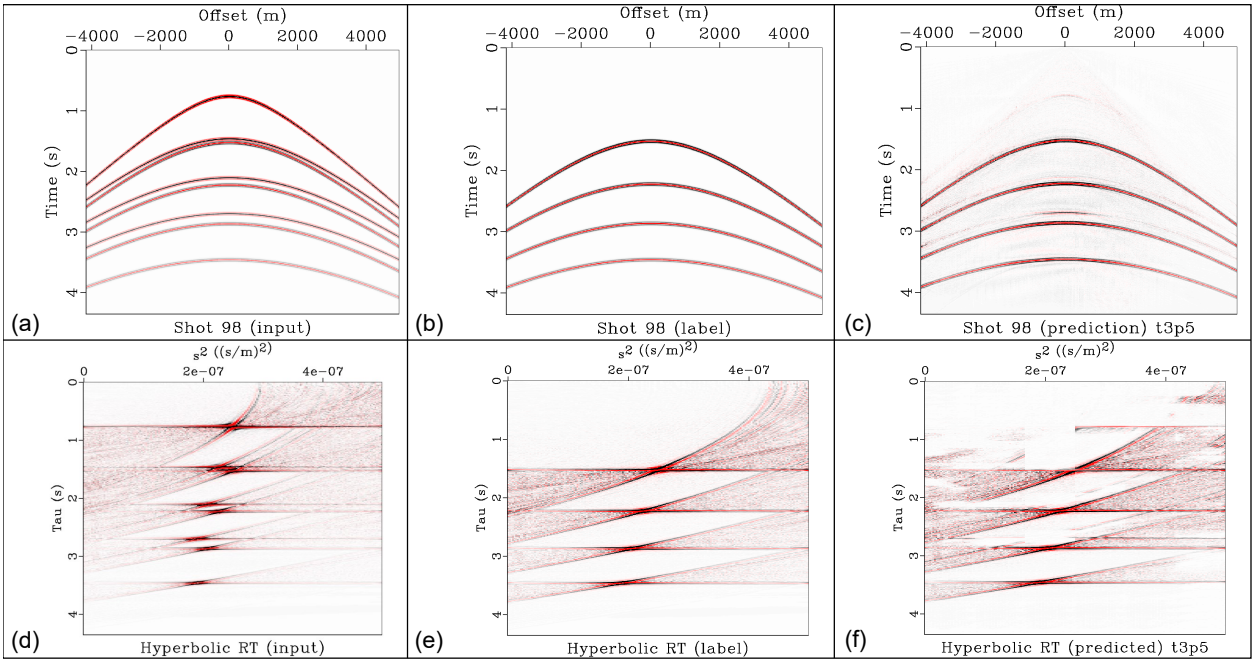


Figure 4.9: Five geological layers case: shot 98 with multiples and primaries (a), just with multiples (b) and after the U-Net prediction using the three geological layers training, subsequent inverse HRT and sorting by shot. HR with multiples and primaries (d), just with multiples (e) and after the network prediction using the 3 geological layers training (f).

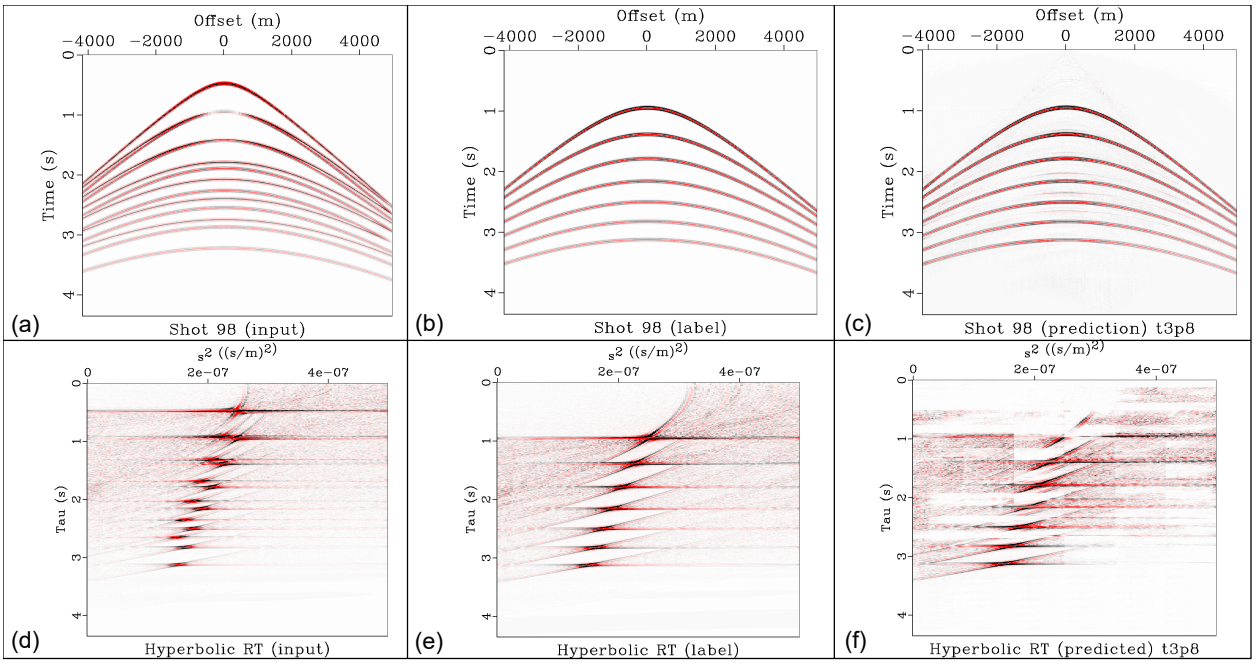


Figure 4.10: Eight geological layers case: shot 98 with multiples and primaries (a), just with multiples (b) and after the U-Net prediction using the three geological layers training, subsequent inverse HRT and sorting by shot. HR with multiples and primaries (d), just with multiples (e) and after the network prediction using the three geological layers training (f).

not coherent artifacts (butterfly effect) are important for the prediction.

Seeking improved results, the network was trained with three and then five geological layers, saving its weights to serve as a foundation for predicting the eight geological layers scenario. The outcome of this prediction in the HRT domain (Figure 4.11f) shows that all the multiples were predicted, having minor missing artifacts. This test's final outcome can be observed in the shot domain (Figure 4.11c) and compared with its input label (Figure 4.11b). Although room for improvement remains in the prediction, particularly regarding the completion of missing artifacts, the network predicts multiples, marking an enhancement over previous tests.

In the subsequent test (Figure 4.12), the network was trained on three, five, and eight geological layers and was applied to predict the eight geological layers case. The result in the shot domain (Figure 4.12c) demonstrates successful prediction of all multiples, alongside the

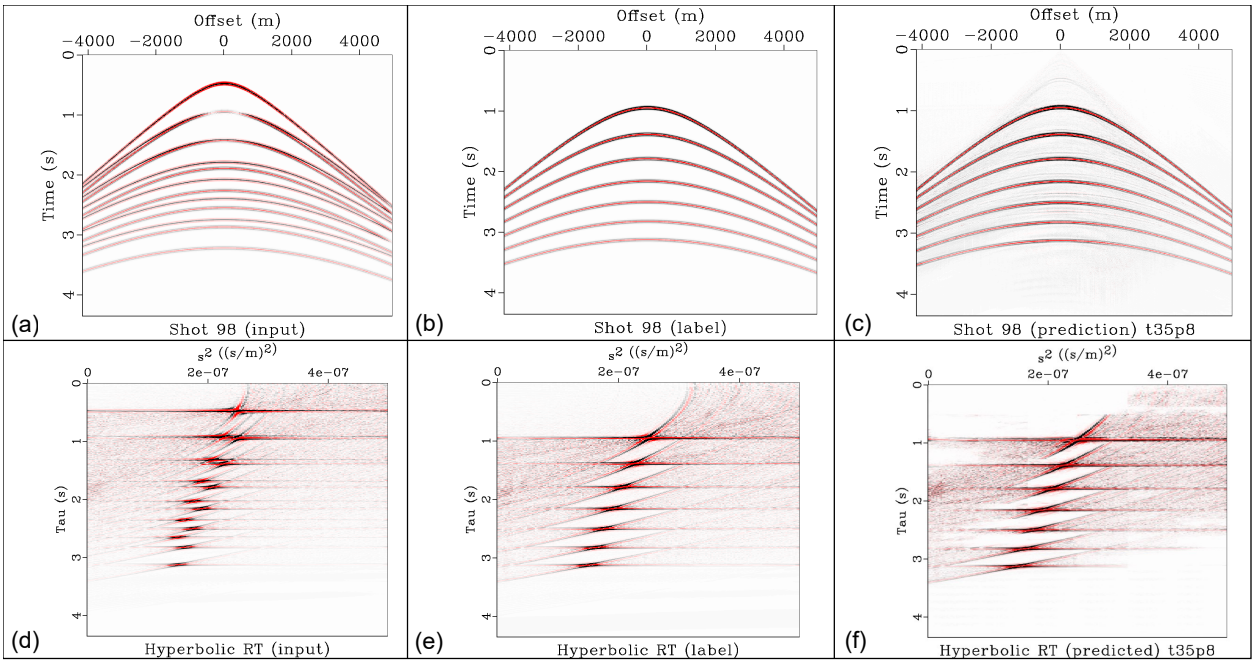


Figure 4.11: Eight geological layers case: shot 98 with multiples and primaries (a), just with multiples (b) and after the U-Net prediction using the three and five geological layers training, subsequent inverse HRT and sorting by shot. HRT with multiples and primaries (d), just with multiples (e) and after the network prediction using the three and five geological layers training (f).

prediction of a substantial portion of the artifacts. Qualitative comparison between Figures 4.12e and 4.12f reveals that they are similar.

Additionally, Figure 4.8b indicates a similar decreasing trend in MSE during prediction compared to the case with less training (Figure 4.8a). The validation portion presents a stable MSE, possibly because the weights saved by prior training were being used, therefore contributing to a plateau performance. When the training loss decreases while the validation loss plateaus, it indicates potential overfitting. This means the model is becoming overly focused on fitting the training data precisely but struggles to generalize to new data. So, even though the result in Figure 4.12f seems similar to Figure 4.12e, this pattern in curves (Figure 4.8b) suggests that while the model may have learned the specific patterns in the training set, in this case is no longer learning new and relevant information during training. It also faces difficulty applying them to unseen examples (generalization power).

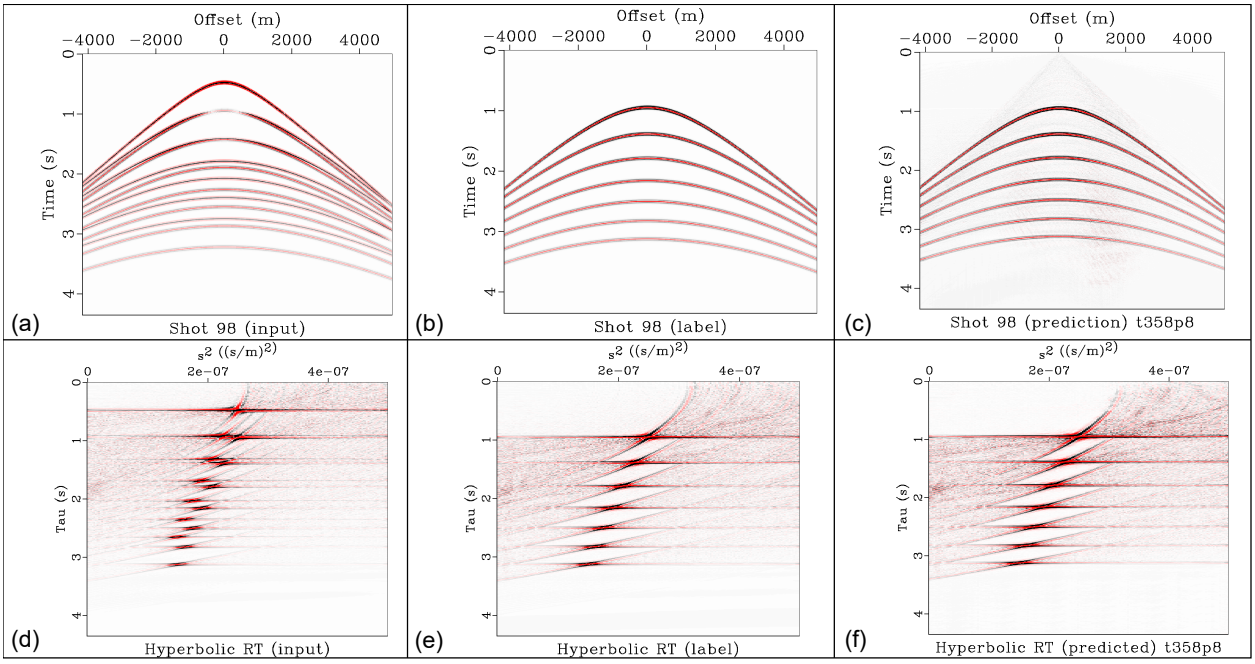


Figure 4.12: Eight geological layers case: shot 98 with multiples and primaries (a), just with multiples (b) and after the U-Net prediction using the three, five and eight geological layers training, subsequent inverse HRT and sorting by shot. HRT with multiples and primaries (d), just with multiples (e) and after the network prediction using the three, five and eight geological layers training (f).

To be able to conduct a quantitative interpretation analyzes, employing an AVO-compliant dataset would be ideal. However, it is crucial to acknowledge that the proposed methodology may compromise true amplitudes, particularly for long offsets. Notably, when utilizing the RT, it is important to consider that the data suffer from aliasing artifacts when too coarsely sampled in offset (Moore and Kostov, 2002). Such aliasing artifacts can adversely affect the quality of the RT panel, leading to an increase in the amplitude of aliased events that extend beyond the s analyzes window (Marfurt et al., 1996). In this regard, sparse RT techniques (Thorson and Claerbout (1985), Sacchi and Ulrych (1995b), and Trad et al. (2003)) have been developed trying to address that, having the chance to improve the multiple prediction. With that in mind, some tests were conducted to assess whether the network could leverage this higher-resolution information to improve prediction quality.

4.1.2 Test II: advantages and disadvantages of applying sparse RT and two input channels

The sparse HRT (Sacchi and Ulrych (1995b), Trad et al. (2003)), known for its higher resolution, employs an iterative reweighted Least Squares algorithm with external iterations. To better understand how the network learns features with varying input data and labels, I conducted a test using both sparse and non-sparse HRT.

In one test, Figure 4.13, the U-Net was trained with three and five geological layers and predicted for the eight geological layers case. Figure 4.13f shows that there are still some artifacts in the background, even though the train was done in the sparse HRT. Comparing 4.13c with 4.11c the far offset, the first one is weaker.

Then, the U-Net was trained with three, five and eight geological layers and tasked with predicting the eighth geological layer scenario. Figures 4.14a and 4.14b show shot 98 and its corresponding multiples-only version, respectively. After applying the sparse HRT, seen in Figures 4.14d and 4.14e, the resulting prediction in Figure 4.14f revealed some attenuated multiples alongside persistent background artifacts.

Examining the shot domain representation of the U-Net result in Figure 4.14c, some remains of primary reflections are still present. Comparing this result with the non-sparse prediction (Figure 4.12c) indicates that the U-Net using sparse HRT did not enhance the final prediction while using the DL approach, particularly in the long offset amplitudes are attenuated if compared with the non-sparse prediction. While the U-Net, using a non-sparse approach (Figure 4.12c), predicts multiples, it suffers from background noise in the prediction. On the other hand, the U-Net using a sparse HRT (Figure 4.14c) prediction demonstrates reduced background noise, thereby enhancing resolution but low amplitudes on the far offsets.

An alternative approach to address the challenge of near-overlapping primary and multiple events involves employing two channels, sparse and non-sparse HRT, a workflow represented in Figure 4.6. This approach aims to provide the network with more informative data while minimizing background noise, thereby facilitating better differentiation of near-overlapping events and, consequently, improving multiples prediction.

Figure 4.15c shows the result of U-Net using panels of both sparse and non-sparse HRT (Figure 4.15d) as inputs and HRT non-sparse (Figure 4.15e) as the label. Figure 4.16c shows the result using both HRT sparse and non-sparse as inputs (Figure 4.16d) and HRT sparse (Figure 4.16e) as the label. Qualitatively comparing the results of the U-Net prediction in the HRT panels (Figures 4.15f and 4.16f), it can be noted that the label has a large influence on the predictions. Surprisingly, using HRT sparse as the label does a poor job of predicting the multiples compared to the non-sparse label. Furthermore, as shown in Figure 4.16f, the background HRT is still present. Therefore, the prediction using the non-sparse HRT shows a better U-Net multiple prediction.

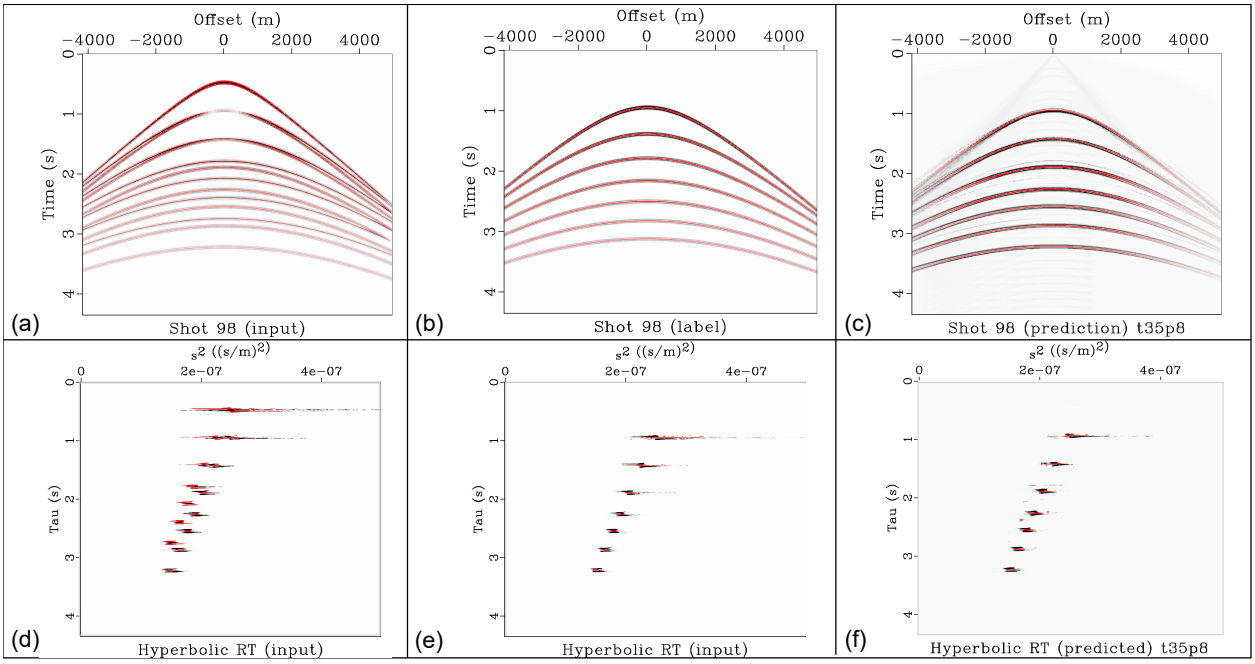


Figure 4.13: Eight geological layers case: shot 98 with primaries and multiples (a) and its sparse HR panel (d). Shot 98 just with multiples only (b) and its sparse HR panel (e). Shot 98 after the U-Net prediction for the 8 geological layers case using the training of the three and five geological layers and inverse HRT (c) and its eight geological layers HR panel prediction (f).

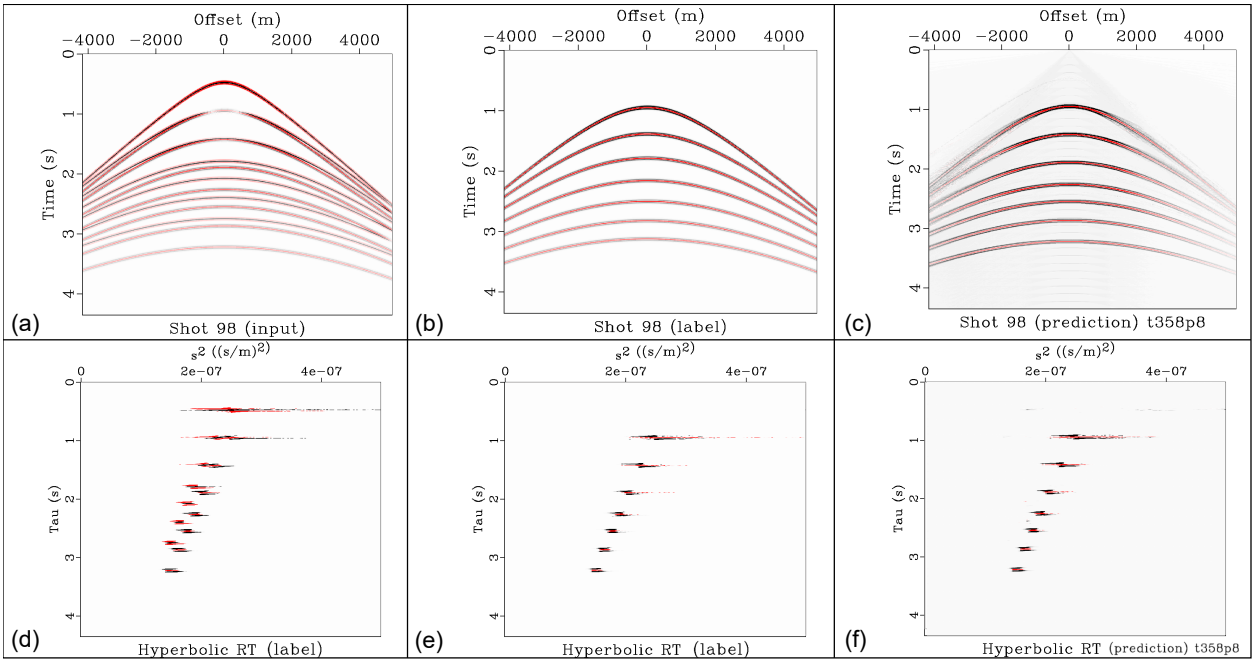


Figure 4.14: Eight geological layers case: shot 98 with primaries and multiples (a) and its sparse HR panel (d). Shot 98 just with multiples only (b) and its sparse HR panel (e). Shot 98 after the U-Net prediction for the 8 geological layers case using the training of the three, five and eight geological layers and inverse HRT (c) and its 8 geological layers HR panel prediction (f).

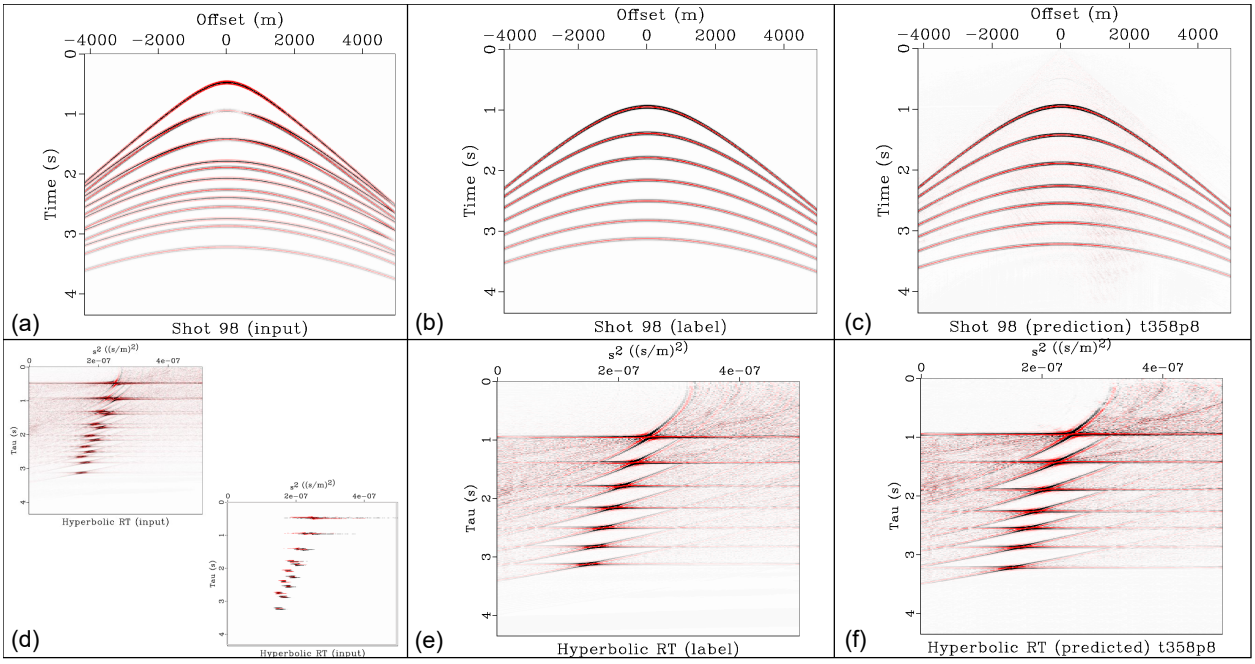


Figure 4.15: Eight geological layers case using 2 channels and non-sparse HRT as the label. Shot 98 with primaries and multiples (a), its sparse and non-sparse HRT panel (2 channels) used as inputs (d). Shot 98 just with multiples (b) and its non-sparse HR panel used as the label (e). Shot 98 after the U-Net prediction for the 8 geological layers case using the three, five and eight geological layers training (2 channels) and inverse HRT (c) and its HR panel result from the prediction (f).

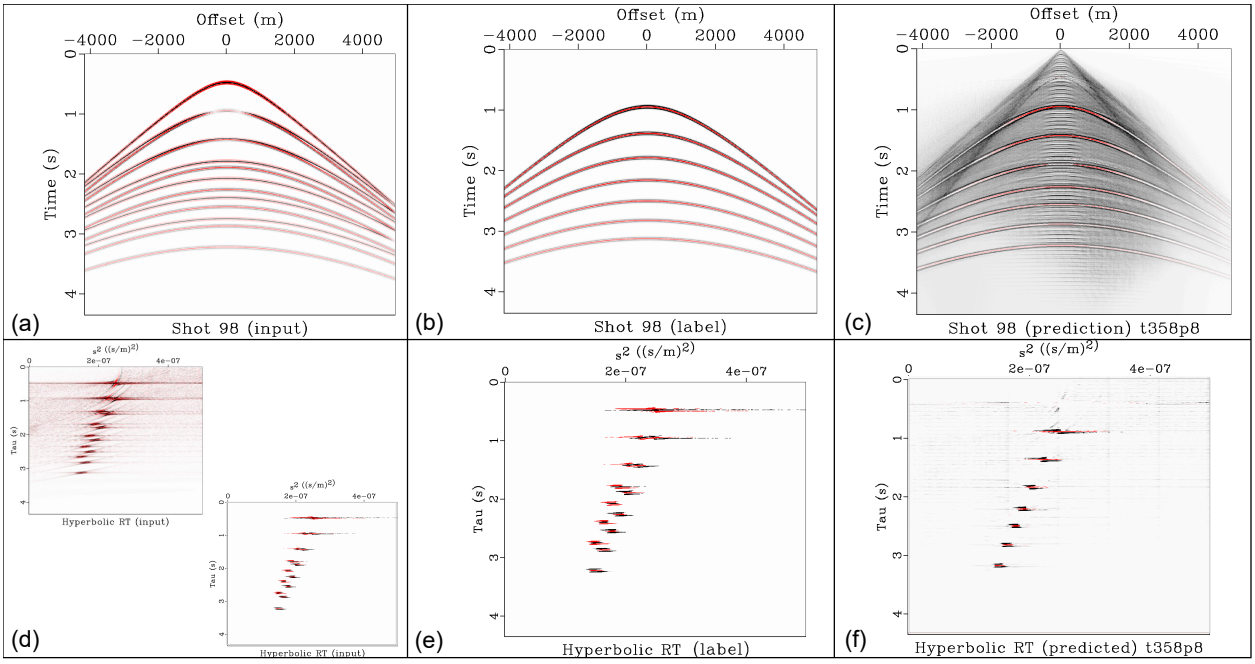


Figure 4.16: Eight geological layers case using 2 channels and sparse HR as the label. Shot 98 with primaries and multiples (a), its sparse and non-sparse HRT panel (2 channels) used as inputs (d). Shot 98 just with multiples (b) and its sparse HRT panel used as the label (e). Shot 98 after the U-Net prediction for the 8 geological layers case using the three, five and eight geological layers training (2 channels) and inverse HRT (c) and its HR panel result from the prediction (f).

4.2 Parabolic and Hyperbolic RT tests with synthetic seismic - Bridge

In the previous section, experiments investigating the efficacy of sparse and non-sparse HRT channels revealed promising outcomes, surprisingly having the latter demonstrating superior capabilities in predicting multiples. Building upon that, further tests were done to train the network with multiple channels incorporating different features, such as the Parabolic Radon Transform (PRT), to refine the understanding of multiple prediction.

Exploring the integration of additional labels, including sparse or non-sparse HRT, offers valuable insights into the network's adaptability to diverse input scenarios. The choice of labels used in training is important, impacting the workflow's outcome. If one label is used, that would be the primary information, and the second channel would be the supporting knowledge. Thus, experiments were done to try to answer the question: when having more than one label, which one enables the best prediction? When using a DL methodology, questions of this nature remain unanswered until put to the test.

Another relevant point in this thesis is that I try to understand the problem by analyzing how the workflow, mixing seismic processing and ML processes, is built for the posed problem. Thus, some experiments were done using workflows similar to those of previous tests. To train the U-Net (Figure 3.1), I used implementations of the convolutional model to generate 120 synthetic shot gathers, with a total record time of approximately 4 seconds with 0.004 seconds of temporal sampling from the same eight geological layers velocity model (Figure 4.3d).

Figure 4.17 shows the input data (multiples and primaries) and input label (multiples only) for training, exemplified by shot 98. Then the HRT (Figure 4.17a), sparse Hyperbolic RT (Figure 4.17b), and Parabolic (Figure 4.17c) RT are applied, generating RT panels. In the RT panels, primaries and multiples are spatially separated, with primaries aligned to the left and multiples to the right since they have higher travel times.

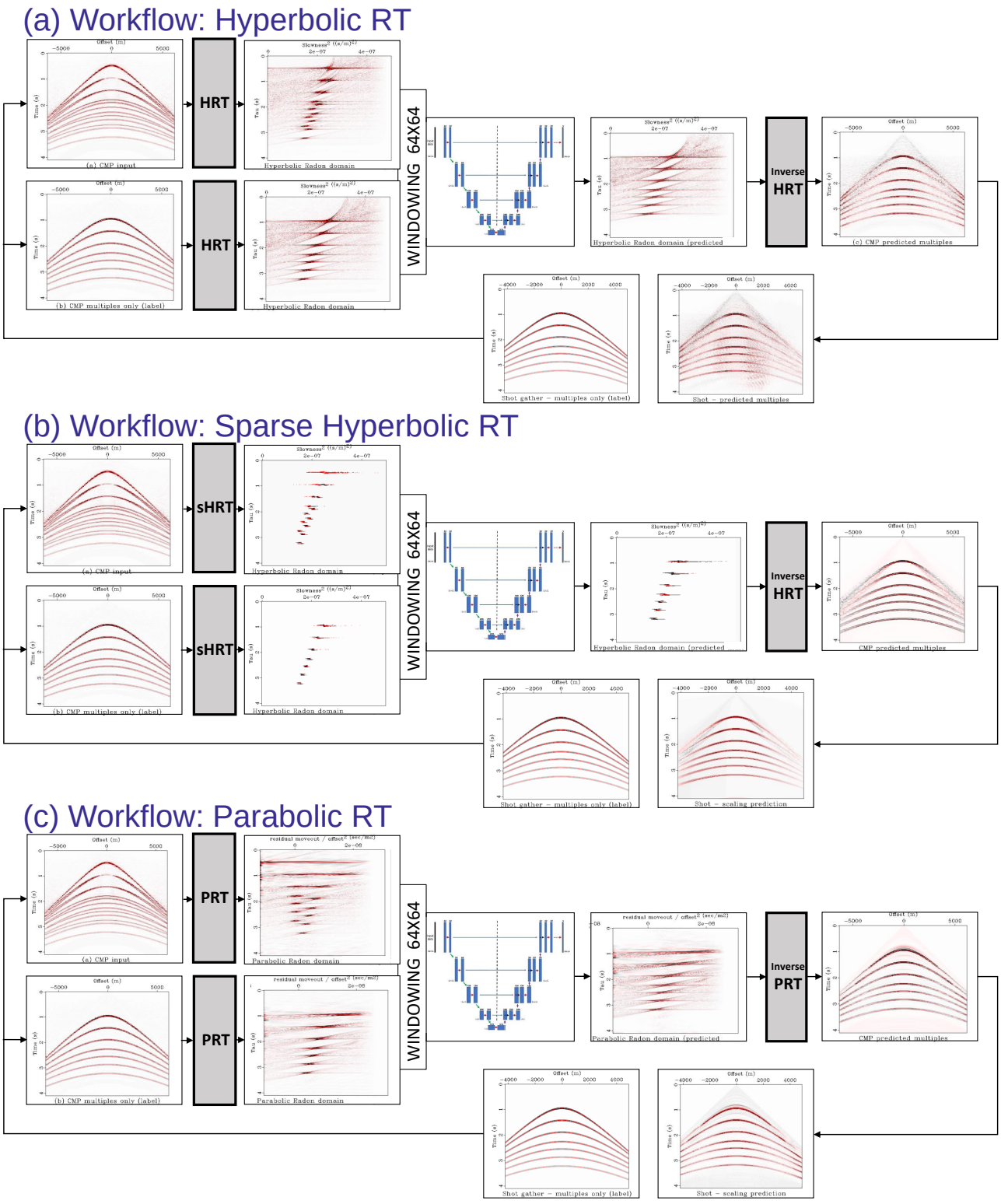


Figure 4.17: Workflow for the (a) Hyperbolic RT, (b) sparse Hyperbolic RT and (c) Parabolic RT.

Then, these HRT (τ x s^2) or PRT (τ x q) images go through data preparation steps: normalization, scaling, and windowing. These are then the data inputted into the U-Net, which will be assigned to train and predict multiples. It is important to note that some primaries and multiples are overlapping in the shot domain and, therefore, will be close in the RT domain, so that is something to keep close attention to if the network will understand and map this difference.

The network's output consists of the RT panels, ideally containing the predicted multiples. The final stage involves applying the inverse RT using Least Squares to reconstruct the data from the RT domain back to the CMP domain and subsequently to the shot domain, depending on the stage of the processing workflow. Upon analyzing the tests depicted in Figure 4.17, several observations can be made:

1. The choice of label significantly influences the U-Net's predictions. Since the prediction relies on training, the selected label dictates the overall appearance of the prediction.

2. Contrary to expectations, the sparse version does not consistently yield superior predictions. Still, while using an RT model with fewer details in the image and more pixels with content close to zero (sparse), the U-Net seems not to learn as much as in the other case (non-sparse). Furthermore, it is important to note that the example uses synthetic data, so these effects could be even stronger, considering that field data can be affected by so many other factors.

3. Comparing the HRT (Figure 4.17a) with the sparse HRT (Figure 4.17b), the former demonstrates better performance in predicting the long offset portion. However, the sparse HRT better controls the background noise. Yet, the PRT (Figure 4.17c) achieves an overall superior multiple prediction, having a balance between preserving long offsets (despite the application of stretch mute) and minimizing background noise.

Following these tests, the objective is to assess whether Hyperbolic and Parabolic RT, operating in time-variant and invariant domains, respectively, can complement each other.

However, since one is done in the time and the other in the frequency domain, the two channels are not pixel-by-pixel equivalent to each other. To explore the potential co-action between these two RTs, I employed some further tests to try to build an intermediate transformation (bridge) towards enforcing sparseness and, therefore, having a higher resolution prediction.

Figure 4.18 illustrates the workflow for the bridge 1 test. By having PRT as input data and HRT as the label, it is possible to predict an "intermediary1" output, which does not have physical meaning since the input and the label have different x-axis (s^2 and q, respectively). This intermediary output bridges as an input, and by using the HRT as a label, I can predict the RT with multiples only and convert it back to the data domain.

Figure 4.19 shows the workflow for the Bridge 2 test, which has a similar idea to Bridge 1, but now I am trying to understand if using the PRT as the label is better. By comparing the results of the prediction of both of the bridges, the predicted CMP that presents, qualitatively, a better prediction is the Bridge 2 test (Figure 4.19).

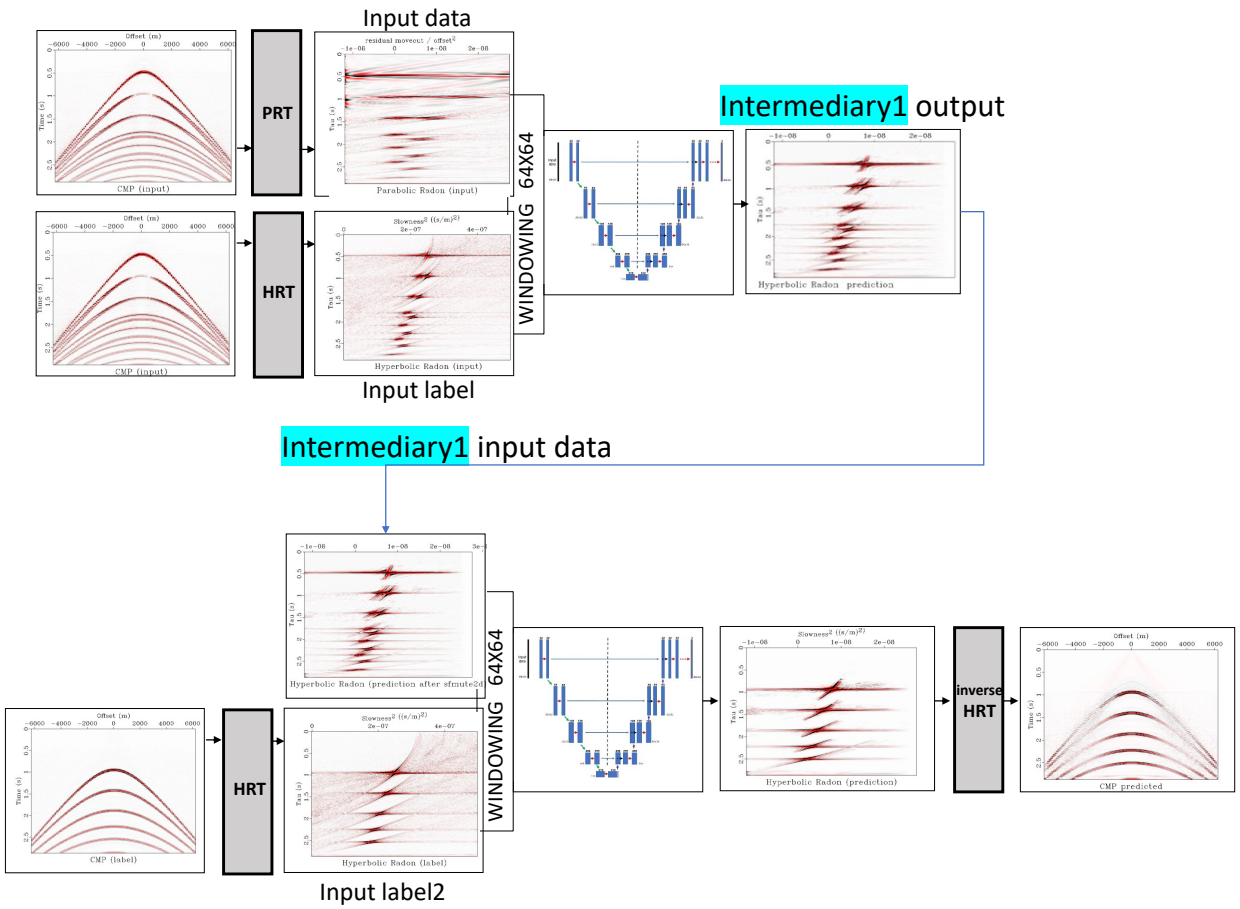


Figure 4.18: Bridge 1: a workflow that uses Hyperbolic RT as label

Bridge 2: Input Hyperbolic RT, Label Parabolic RT

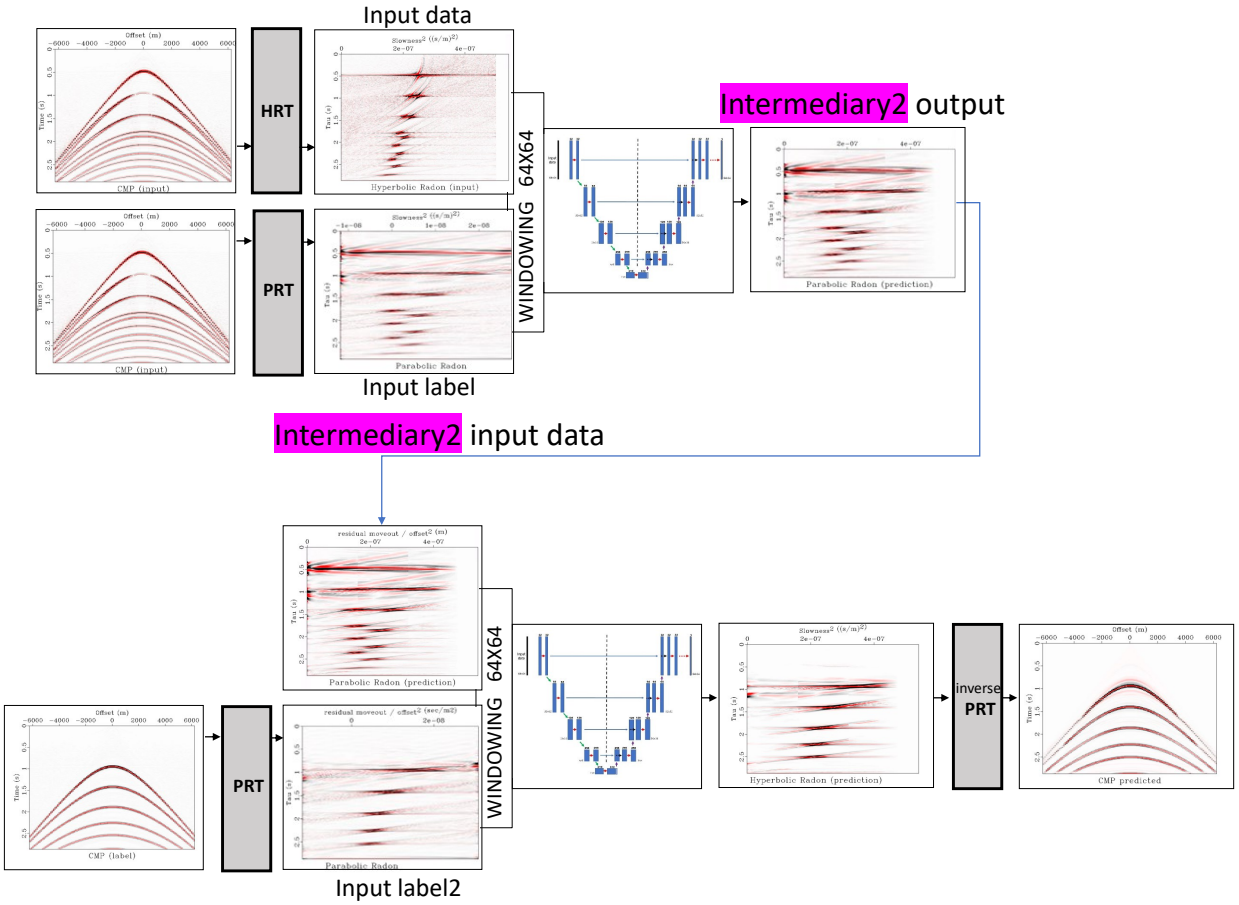


Figure 4.19: Bridge 2: a workflow that uses Parabolic RT as label

Alternatively, Figure 4.20 shows a different way to do bridge 1 between HRT and PRT. In this case, the input data 1 is the Parabolic RT, and the input label 1 is the Hyperbolic RT. This will produce an intermediary RT output that will serve as input for a 2-channel U-Net workflow with the input data and label. That is a way to connect the same information in different channels. After prediction, the inverse HRT is applied to evaluate the prediction result in the data domain. Note that in Bridge 1, the U-Net predicted an RT panel with a stronger concentration of amplitude in the center if compared with alternative bridge 1.

A similar workflow is shown in Figure 4.21 as an alternative to the bridge 2 workflow represented previously (Figure 4.19). In a way, the U-Net is trying to enforce sparseness in the predicted RT, but since it uses different types of RT, the way that this sparseness is translated in the prediction will not necessarily help.

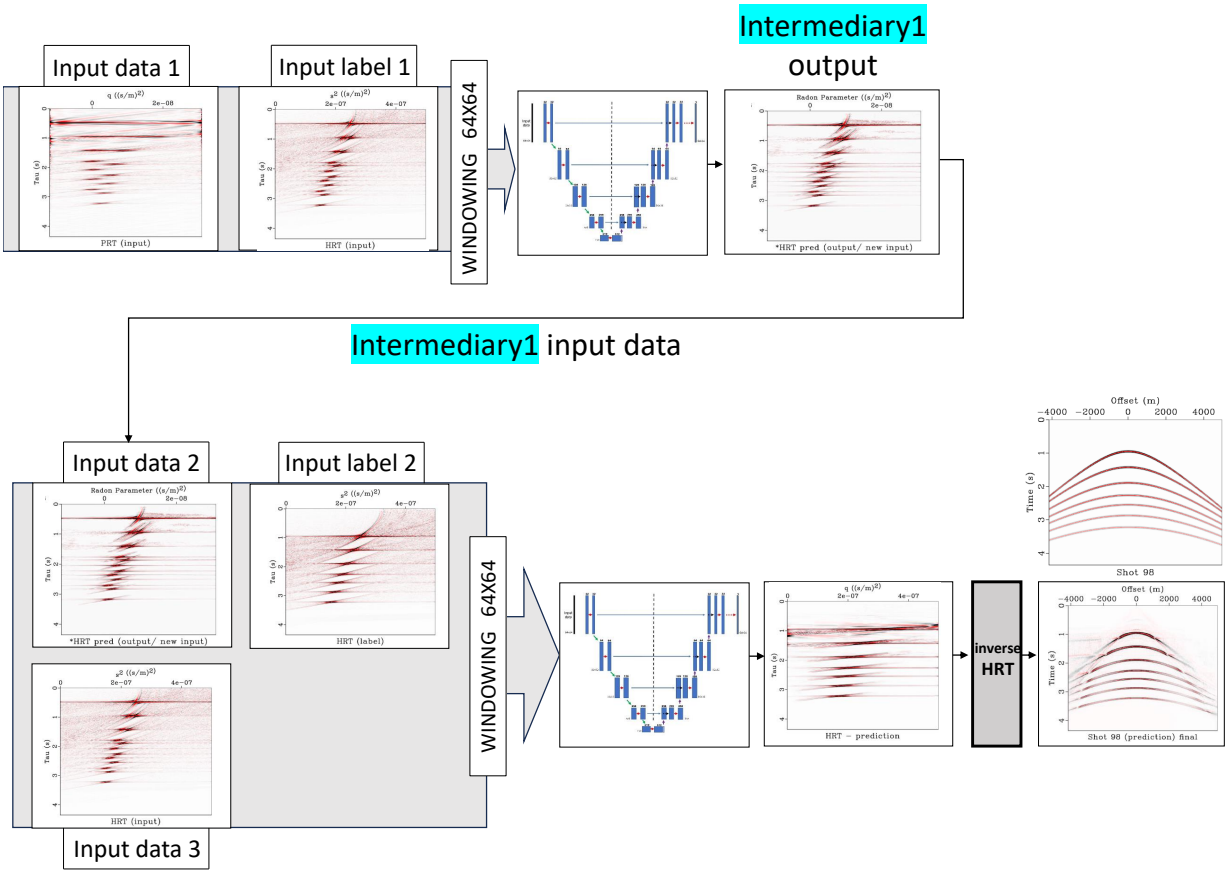


Figure 4.20: Alternative Bridge 1: a workflow that uses Hyperbolic RT as the label and an intermediary1 as a bridge to a two-channel prediction.

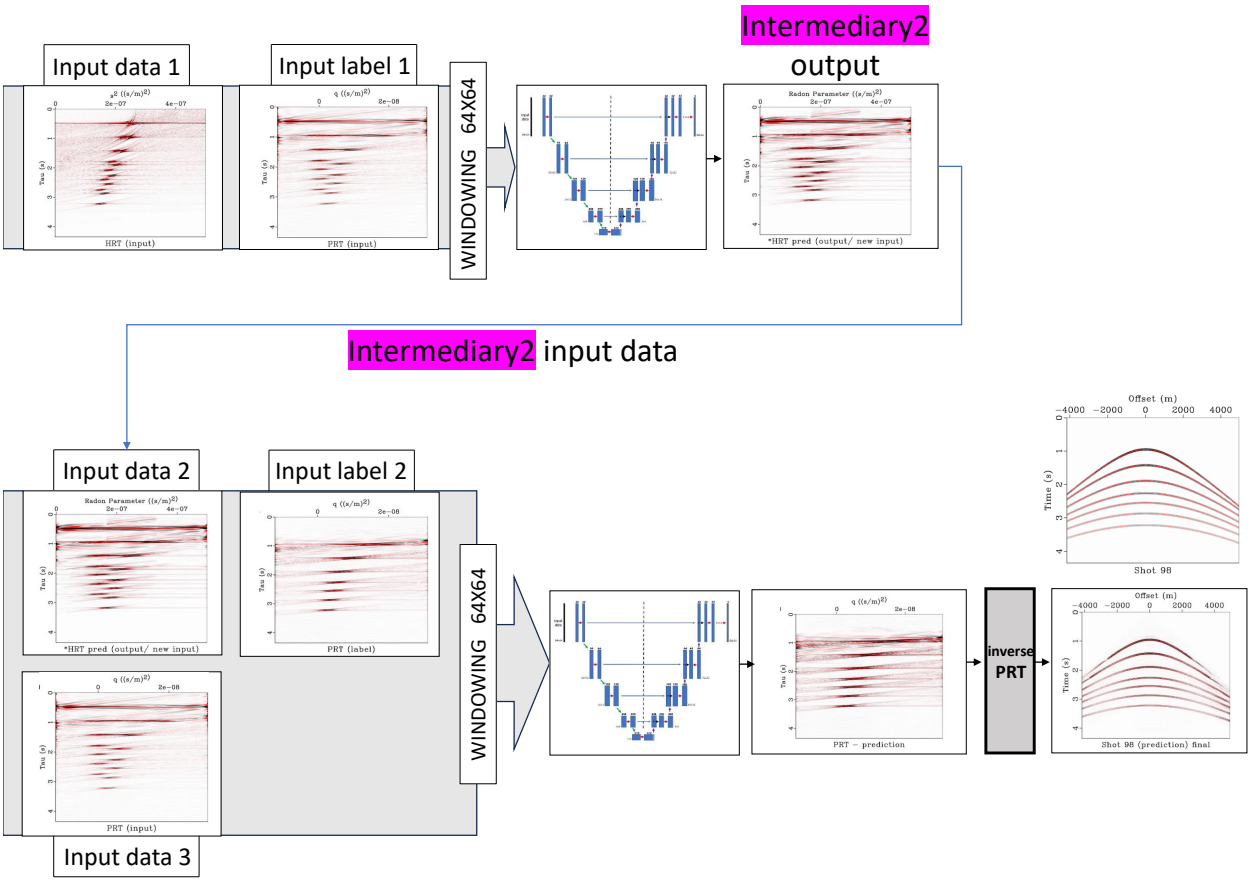


Figure 4.21: Alternative Bridge 2: a workflow that uses Parabolic RT as the label and an intermediary2 as a bridge to a two-channel prediction.

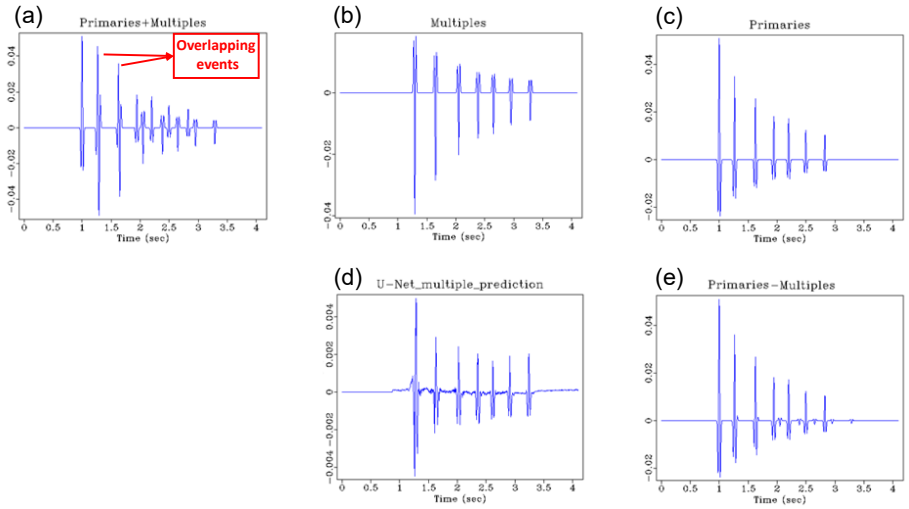


Figure 4.22: Amplitude of one seismic trace: (a) primaries and multiples, (b) multiples, (c) primaries only, (d) U-net prediction, and then (e) the result of the subtraction (d) from (a). Comparing (b) and (d), the network prediction qualitatively maps the location of the events, but it does not fully honour the amplitude. Scaling and matching filter were needed to be applied to calculate the difference.

Furthermore, since these workflows (Figure 4.17) predict only multiples, one last step should be taken to have the final attenuated data set. This is done by taking the CMP domain of the multiple prediction done by the U-Net and subtracting it from the original CMP input to ideally have only primaries left. Usually, an adaptive subtraction is used, but since the predictions do not produce an AVO-compliant amplitude, the subtraction would not necessarily work. Figure 4.22 shows an example of the adaptive subtraction done in one trace of the U-Net prediction (Figure 4.22d) from the original data (Figure 4.22a). The result is shown in Figure 4.22e, with scaling and matching filter being necessary for this adaptive subtraction.

4.3 Conclusions

The RT is an important tool for separating multiple and primary reflection events. This chapter examined the efficiency of employing various methodologies, particularly using HRT and PRT panels for training the U-Net model and using that for prediction. While the U-Net demonstrated the capability to partially predict multiples through inference, challenges arose due to sampling artifacts. Even though the windowing process had overlapping windows, attempting to predict multiples in a more complex geologic model (8 geological layers) using the training from a simpler model (3 geological layers) was possible, but with some artifacts showing some windowing footprint. To address this, incorporating diverse geological models during training could enhance prediction accuracy. Surprisingly, the network output using only sparse HRT presented less background noise but did not attenuate overlapping multiples as expected. Training with two channels, sparse and non-sparse HRT, and utilizing non-sparse RT panels as labels yielded improved multiple predictions.

Additionally, it is imperative to acknowledge the challenges posed by noise and sampling artifacts, where the network may struggle to distinguish between coherent noise and signal. Moreover, overlapping primaries and multiples in the shot domain present a significant hurdle, emphasizing the importance of attention to detail during network training. The predictions generated by the network could potentially be used for qualitative interpretation rather than solely serving as intermediate processing products. This qualitative approach to interpretation highlights the practical utility of the network's outputs.

Subtracting the U-Net predicted multiples from the original input is not a simple task (matching filter, scaling) since the U-Net does not seem to produce an amplitude-friendly prediction. This is due to the pixel-by-pixel nature of this methodology and the fact that I am treating the RT model space as an image. An alternative way to do that would be to predict the attenuated version instead of predicting the multiples (to then subtract), but further tests would be necessary to assess that possibility.

It was challenging to window the RT panels as the window size is small and goes through

non-stationary issues. This is why time-variant RT could be considered less useful, even though it produced fair results.

The contribution to generalization is not clear. It is relevant to emphasize that the network lacks an understanding of the underlying physics of the experiment. Thus, solely enhancing the quality of input data, such as employing a higher-quality RT, may not necessarily improve prediction outcomes. That is the reason why the key is to conduct systematic experiments to understand the most effective input information for the network. This can be time consuming, but resources such as GPU can expedite the experimentation process.

Chapter 5

Training and prediction - ground roll

In Chapter 4, I presented a series of experiments to predict multiples using the U-Net through the workflow outlined in Figure 3.5, along with some variations. Building upon this, the current chapter focuses on addressing a distinct type of noise: ground roll. Given the different characteristics between multiples and ground roll, a slightly different workflow depicted in Figure 3.6 is employed to tackle ground roll effectively. This chapter investigates how nonlinear characters of real seismic data are handled while predicting the ground roll attenuated version of the seismic dataset.

Furthermore, central to the investigations in this chapter is examining how various RT spaces interact with one another, a relationship referred to as crosstalk. By inputting different RT panels into the network, these tests aim to provide deeper insights into how the proposed ML approach handles the intricacies of seismic data, particularly regarding sparseness and nonlinear characteristics. This comprehensive exploration seeks to enhance our understanding of how DL techniques can effectively contribute to tasks such as multiple prediction and ground roll attenuation in the seismic data processing workflow. Throughout all numerical tests conducted, consistency was maintained in both the architecture of the U-Net model (depicted in Figure 3.1) and its hyperparameters, as outlined in Table 3.1.

Ground roll, a type of surface wave commonly found in land seismic data, is defined

by its strong amplitude influenced by weathered materials and unconsolidated sediments in the near-surface. It is a coherent noise in the shot domain, having a relatively low-frequency content and low velocity compared to other seismic events. As seen in Chapter 2, various techniques, including RT, can predict and attenuate ground roll. Basis functions display different appearances in the RT domain, guiding their application for specific tasks. Primaries exhibit a hyperbolic shape in the shot domain, whereas ground roll has a linear shape. This difference in shape helps separate them within RT spaces.

In Chapter 4, the experiments presented promising results while using information from two types of RT (parabolic and hyperbolic) separately or combined using a bridge. From this groundwork, I tried to expand this workflow to tackle ground roll, which has different challenges, going even further by using field data. Through various tests, the present chapter aims to analyze the use of the DL methodology in field data while employing hybrid RT and determine which channel of information proves to be more advantageous in the prediction process. Moreover, I further describe the field data acquisition parameters from the dataset used for tests and some of its pre-processing steps.

5.1 Hybrid RT - Crosstalk

The depiction of a hybrid RT panel is illustrated in Figures 5.1b, 5.1b', 5.2b, and 5.2b'. The x-axis represents the Radon parameter, denoting a fusion of the Linear Radon Transform (LRT) on the left side and the Parabolic Radon Transform (PRT) on the right side of the hybrid RT panel. As explained in Chapter 1, each of these transforms has its representation of slowness: p for LRT and q for PRT.

Usually, seismic data processors do not visualize the RT space (Figure 5.1b and 5.1b') while applying RT. What is usually done is a check in the reconstruction of the data (Figure 5.1c and 5.1c'). However, it can be helpful to visualize seismic events mapping into different regions of the RT domain. Intending to understand whether it is possible to map different

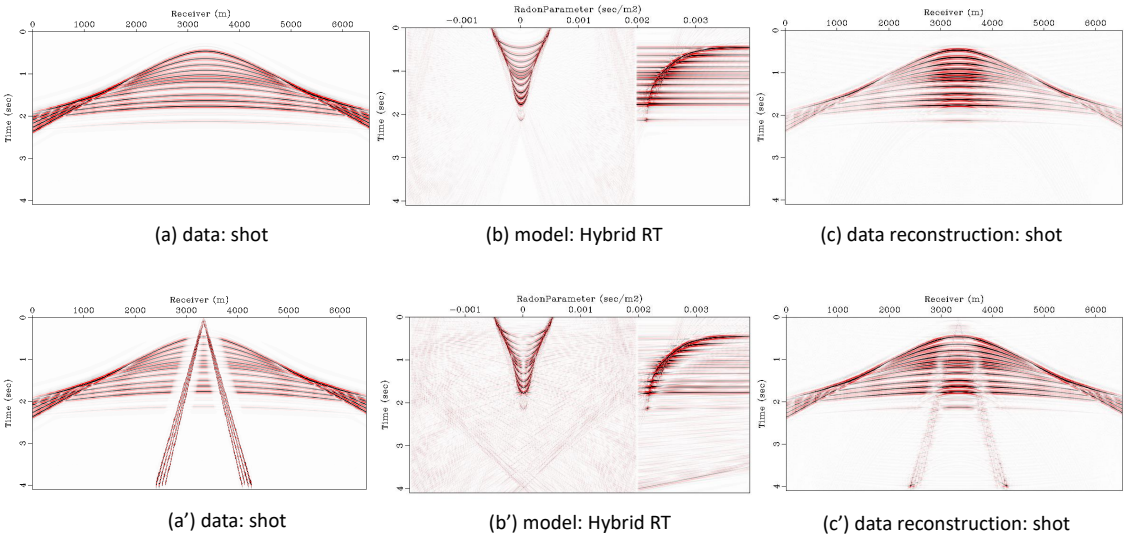


Figure 5.1: Synthetic shot (using the convolutional model) using AGC: with reflections only (a) and with ground roll (a') using velocity model and geometry from Spring Coulee field data; Hybrid (Linear and Parabolic) RT of reflections only (b) and its reconstruction (c). Hybrid RT of reflections and ground roll (b') and its reconstruction (c'). The reconstruction was done using the adjoint RT

events into different RT spaces Trad et al. (2001) applied the hybrid operator for ground roll attenuation. It shows the possibility of such mapping if the two basis functions are different. That is the case for ground roll and reflections (data space, in the shot domain) since they are mapped into lines and approximately hyperbolic shapes, respectively, in the RT (model) space.

In Chapter 4, the application of the DL methodology for multiple prediction allowed for the utilization of sparseness, primarily due to the use of NMO correction. While enforcing sparseness in the RT domain facilitated the separation process, its applicability is not possible in the current context because of the unique characteristics of ground roll coherence in the shot domain due to the dispersion (the NMO can be used in simple structures by converting to linear, but this distorts the reflections). Therefore, to build the models shown in Figures 5.1b and 5.1b' sparseness is not enforced because the transforms inherently do not exhibit

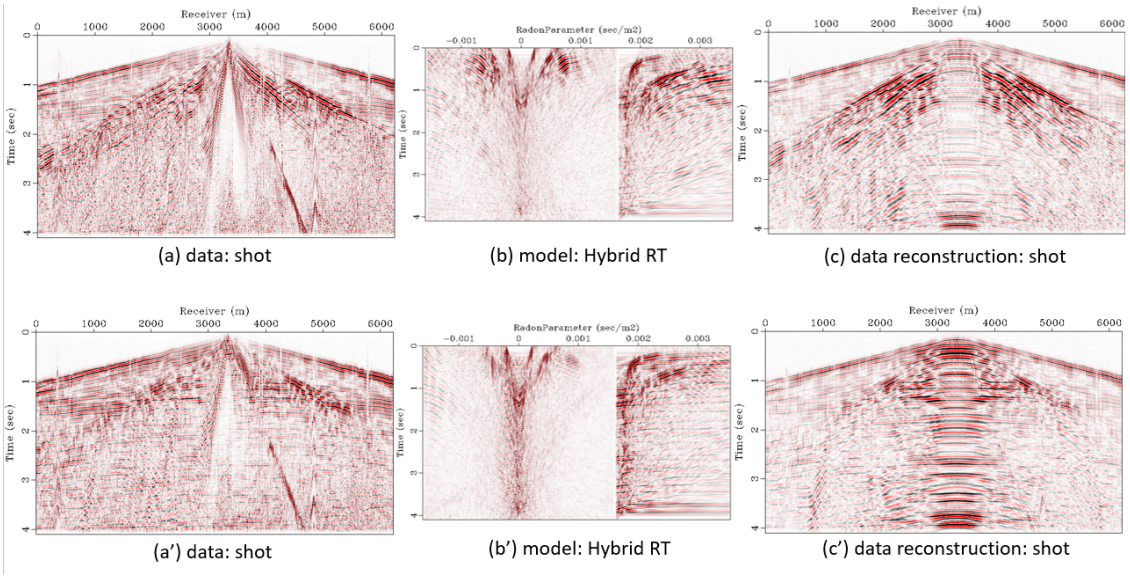


Figure 5.2: Field seismic data: (a) original Spring Coulee shot with a visible ground roll and some reflections, its (b) Hybrid (Linear and Parabolic) RT and its (c) shot reconstruction. (a') Spring Coulee's original shot after being FK filtered in Vista with more visible reflections, its (b') Hybrid (Linear and Parabolic) RT and its (c') shot reconstruction.

sparse characteristics.

The transformation in the PRT involves mapping hyperbolic shapes from the data space to parabolic shapes in the model space, while the LRT maps hyperbolic reflections to ellipses and linear ground roll to aliased lines, so neither of them has sparseness by definition. Moreover, these two spaces have crosstalk because they both have the capability to explain the same hyperbolas.

It becomes even more challenging when dealing with aliased events, where coherence exists in the frequency slowness (q for PRT and p for LRT) domain but not in the time offset domain. Consequently, all aliased events manifest as background noise, as evidenced in Figure 5.1b'. It is important to remember that spatial aliasing plays a big role in the case of ground roll, as we can see in Figure 5.1a' if compared with when the ground roll is not present (Figure 5.1a). While a sparsity constraint could be used to mitigate aliasing, the non-sparse nature of the events introduces interference, particularly as the ground roll lines become steeper and spread widely in the RT domain, amplifying aliasing effects. As well as

the band-limited nature of seismic data and irregularity in geometry, which is usually the case for field data.

One of the advantages of working with synthetic data is that the processor can organize the seismic data in a regular grid, whereas field data have irregularities, such as variable offsets, among others. For example, Ma and Li (2023) tried to address challenges posed by varying acquisition settings using DL models. Somewhat experimenting with synthetic seismic that mimics a real case geometry can try to overcome this barrier. In practice, it is vital to run experiments in field data to see how the methodology handles all the irregularities of a real data set, and in the DL world, in fact, it might learn patterns that the human eyes are not able to. The synthetic data set shown in Figure 5.1 mimics the field data named Spring Coulee, having all the same parameters.

5.1.1 Spring Coulee: a field data example

The Spring Coulee dataset was acquired by CREWES in 2008, and it contains 54 shots using a dynamite source, with a total of 34857 traces. It has a geometry of acquisition split-spread, with around 600 receivers per shot, 0.002 sample rate and 2001 samples per trace. Figures 5.1a and 5.1a' are an example of generating synthetic data using the same geometry as the Spring Coulee dataset and approximate velocity model of the mentioned field data.

Figure 5.2 shows a similar representation seen in Figure 5.1 but now using the field seismic shot gathers (Figure 5.2a). While using synthetic data, the mapping of the ground roll can be easily seen in the RT. Also, the aliasing situation is usually worse in the field data (Figure 5.2b and 5.2b') compared with the synthetic one (Figure 5.1b and 5.1b').

I used the VISTA (Schlumberger, 2015) software to generate the FK-filtered version of the original Spring Coulee dataset. Figure 5.3 shows the application of this workflow, with Figure 5.3c being the input label and Figure 5.3a the input data for the following experiments.

Figure 5.2 illustrates data where Automatic Gain Control (AGC) has been applied. AGC can enhance weaker signals' visibility while preventing stronger signals' saturation in seismic

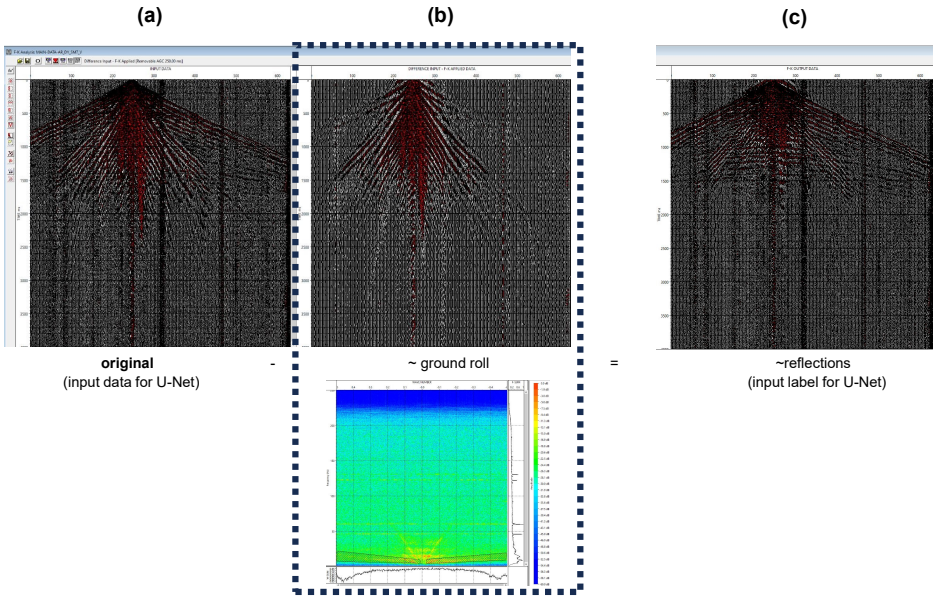


Figure 5.3: Using the workflow (Using VISTA (Schlumberger, 2015) software to generate the FK-filtered version of the Spring Coulee dataset. The ground roll shot (b) is subtracted from the original shot (a), generating the shot with enhanced reflections only, which will serve as the input label.

data. This enhances the visibility of reflections submerged beneath ground roll, and it can help methodologies that treat seismic data as images rather than time series, like DL and RT. Thus, AGC aids in equalizing signal intensities, thereby facilitating a clearer interpretation of seismic patterns for the U-Net model. In practice, AGC is typically computed internally and removed before writing the output. However, in Figure 5.2, I am showing the data with AGC for display purposes.

It is revealing to visualize the data patches after the windowing process (Figure 3.3) will look like since this can provide insights into what the U-Net will actually be exposed for training. Figure 5.4 illustrates the windowing process (64x64) done in the input data and the input label for the case of the synthetic seismic dataset. In this case, the first window will have very similar (to the human eyes) patterns in the input and the label, but this will not be the case for where the relevant information is located (the central part of the hybrid panel).

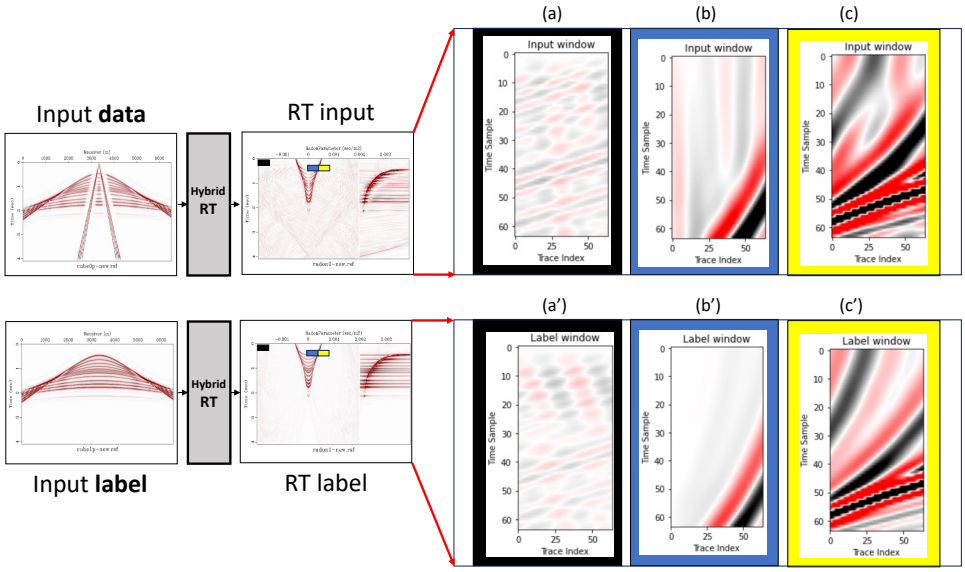


Figure 5.4: Illustrating the windowing process on the synthetic seismic data. The Hybrid RT goes through a windowing process (64,64), and three of these windows are visualized in (a) as the first window, (b) and (c) for the RT input as well as its respective RT labels in (a'), (b') and (c).

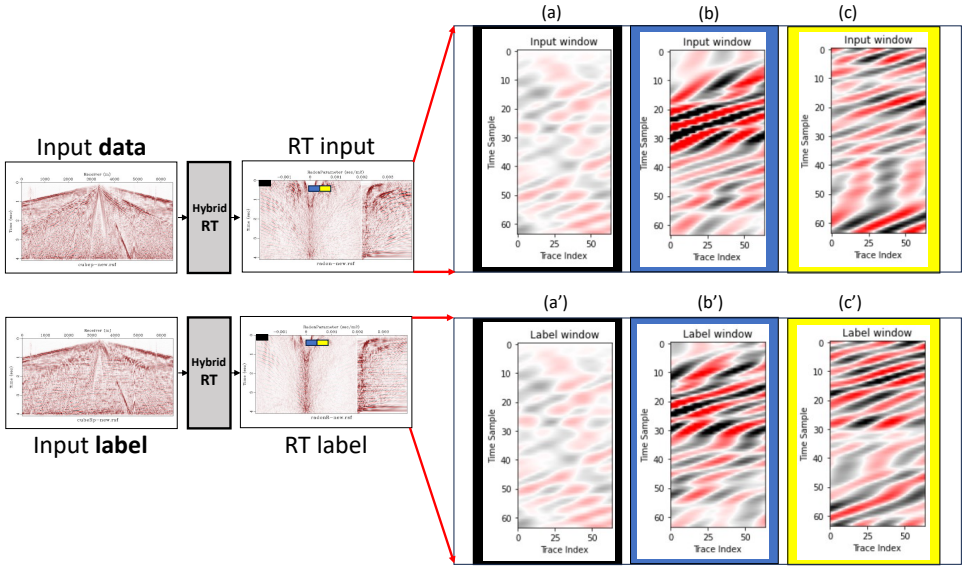


Figure 5.5: Illustrating the windowing process on the input field seismic data (Spring Coulee). The Hybrid RT goes through a windowing process (64,64), and three of these windows are visualized in (a) as the first window, (b) and (c) for the RT input as well as its respective RT labels in (a'), (b') and (c).

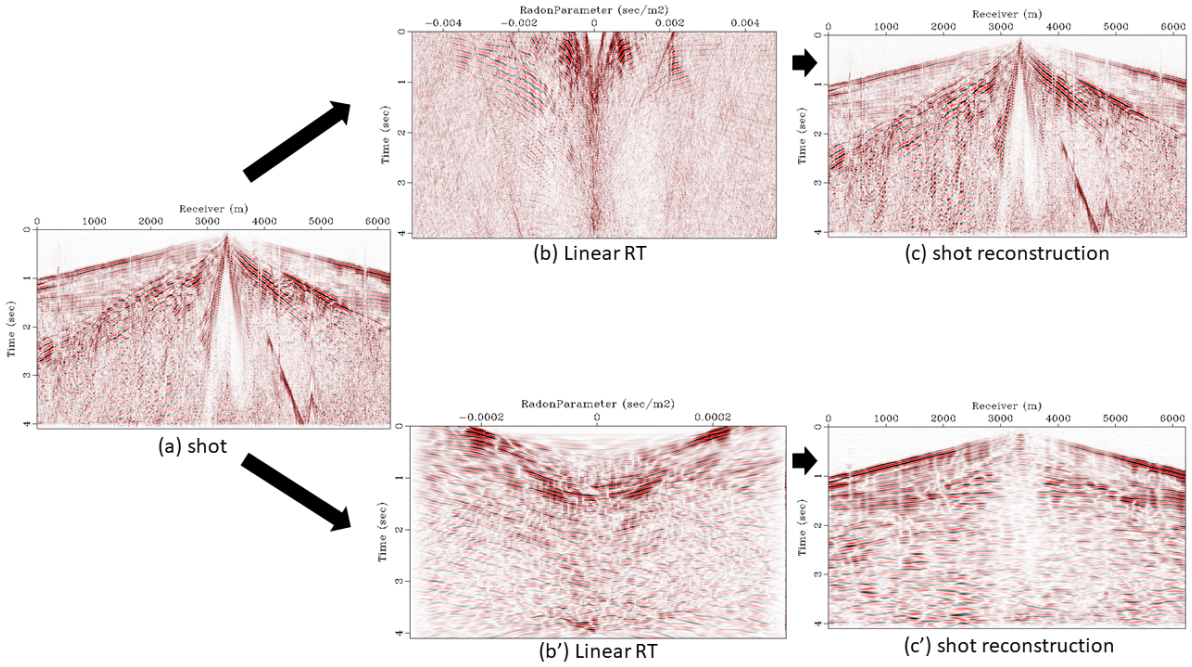


Figure 5.6: (a) original Spring Coulee shot, (b) Linear RT with a wide x-axis, and its (c) shot reconstruction. (b') Linear RT with a restricted x-axis focusing on the ellipses and its (c') shot reconstruction.

Moreover, Figure 5.5 illustrates the same situation but using the Spring Coulee data. However, in both cases, the artifacts' behaviour might be unpredictable because of their non-linear nature; therefore, they are different in every window, which can be challenging to analyze. Ideally, the window would cover the whole RT space, but then there is a trade-off between having many windows, which helps the U-Net model to avoid overfitting, versus having bigger windows, which help to avoid non-stationarity artifacts but require too many parameters for the model.

Another challenging characteristic of the RT is that many areas in the RT have weak but meaningful signals. These are part of the truncation artifacts (so-called butterfly effect), which are necessary for reconstructing the data. An option would be not to consider them, but that will damage the RT reconstruction. Trad et al. (2001) makes a point that different seismic events can be mapped into different basis functions, and signal or coherent noise can be extracted from the data while only selecting a region of the RT that maps the basis

function of your choice (Figure 5.6).

For instance, the processor can restrict the Linear RT x-axis (Figure 5.6b') and only select an area that has ellipses (seismic reflections) and reconstruct the data using only the desired operator by taking a smaller portion of the p (apparent slowness) axis (Trad et al., 2001). This would technically map back mainly the reflections (Figure 5.6c'). However, the complexity of the mapping in field data complicates this nonlinear filtering process and can also lead to the loss of useful signal, as we can see in Figure 5.6c.

In the standard hybrid RT, the separation and focusing (sparseness) involve modelling, but here, I am exploring the application of DL to achieve the same goal. Hence, I am endeavouring to discover an ML-based alternative to sparseness. This exploration aims to determine whether, with sufficient quality labels, it is possible to distinguish between coherent noise and reflections effectively.

As seen in Chapter 2, there are several methods for ground roll removal in the geophysics literature, and this thesis does not aim to suggest a new method. Rather, I would like to discuss how to apply the Hybrid RT to a field data set in a DL methodology of predicting a clean model space. While doing that, I also use the idea that the two RT (linear and parabolic) spaces map different events and, therefore, can work as one, two (one LRT and one PRT) and three (two LRT and one PRT) channels in the U-net training process.

5.1.2 Test I: one channel, one label and one output prediction

Figure 5.7 shows the Hybrid RT one-channel workflow using the synthetic data. Similarly, Figure 5.8 shows the same workflow but using the field data. This experiment used the original field data set as the input data and its FK-filtered version as the input label. For real data, the choice of label introduces uncertainty for DL methods since perfect labels with reflections only are challenging to create.

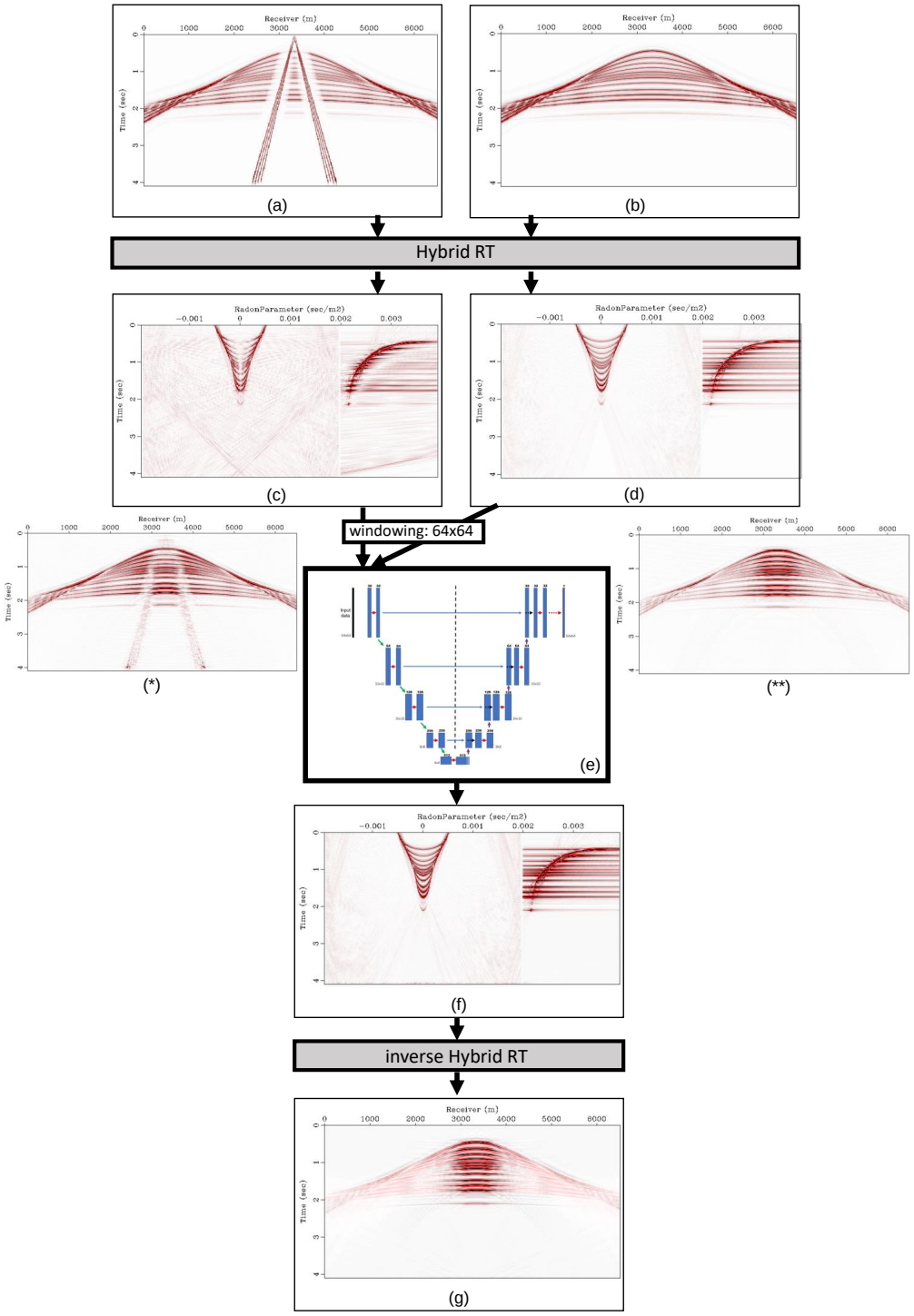


Figure 5.7: Hybrid RT one channel workflow: synthetic shot gathers are the input data, with the ground roll and primaries (a) and, the input label, with the label with reflections only (b). Then, the Hybrid RT is applied to generate the hybrid panels of the input (c) and label (d) to feed the U-Net (e). The network then predicts, after training, the hybrid panels with enhanced reflections only (f), attenuating the ground roll. The inverse Hybrid RT using the adjoint operator is then applied to return the data to the shot domain (g).

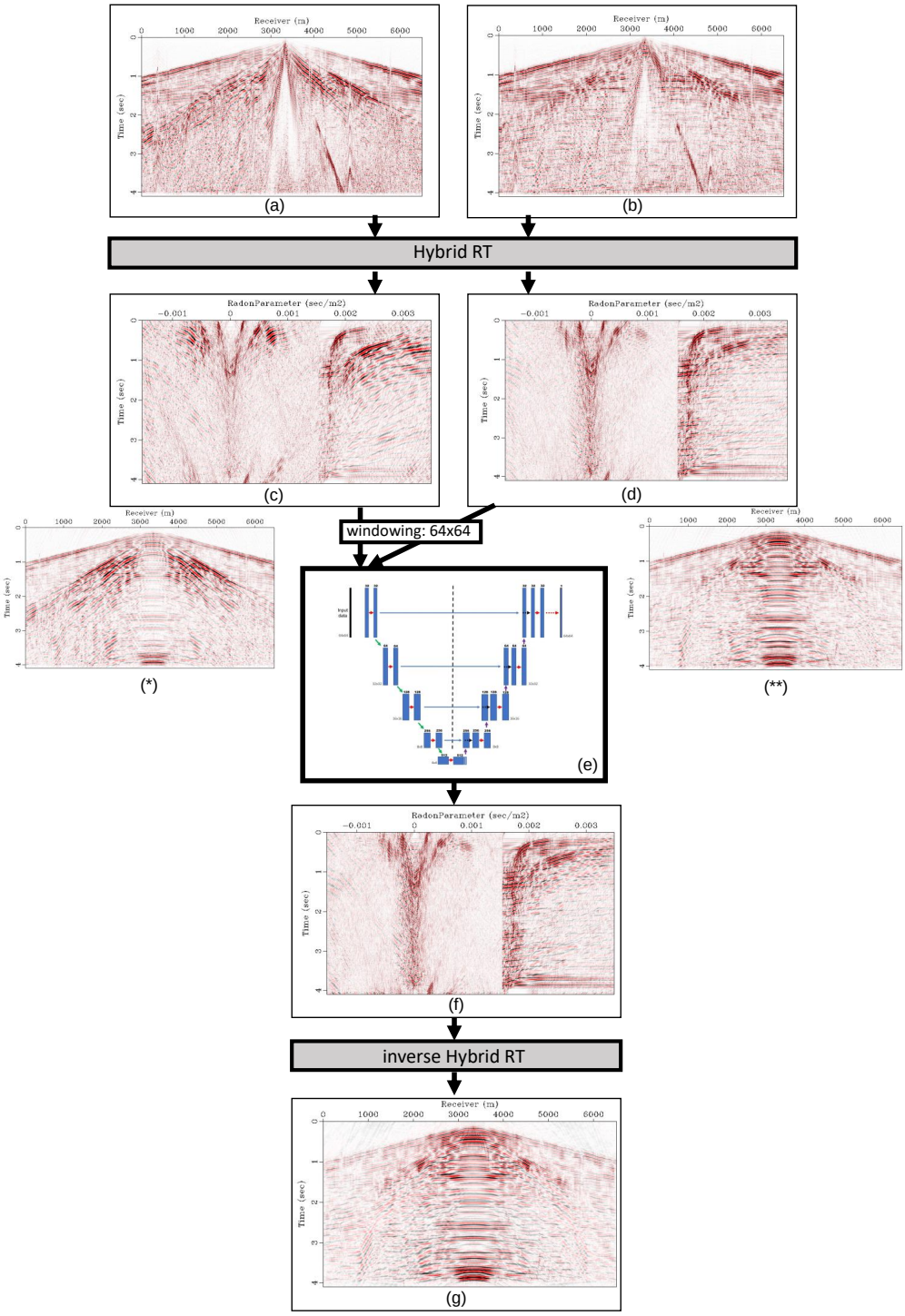


Figure 5.8: Hybrid RT one channel workflow: Spring Coulee shot gathers are the input data (a), with the ground roll and primaries and the input label (b), with the original being FK-filtered to approximate a label with approximate reflections only. Then, the Hybrid RT is applied to generate the hybrid panels of the input (c) and label (d) to feed the U-Net (e). The network then predicts, after training, the hybrid panels with enhanced reflections only (f), attenuating the ground roll. The inverse Hybrid RT using the adjoint operator is then applied to return the data to the shot domain (g).

The U-Net prediction using the synthetic data presents promising results, having its hybrid RT space (Figure 5.7f) qualitatively similar to the label (Figure 5.7d). After applying the inverse Hybrid RT, the data space can be seen in Figure 5.7g. It resembles its label reconstruction (Figure 5.7(**)) but without the background noise. However, it appears that the small offsets are given more importance compared to the longer offsets.

The prediction of the U-Net (Figure 5.8g) has some similarities with the label reconstruction (Figure 5.8(**)), which shows that the DL is doing fair work during the prediction. Also, the RT panels from the label (Figure 5.8d) and the prediction (Figure 5.8f) are similar, therefore having little connection with the ground roll aliasing nature.

5.1.3 Test II: three channels, three labels and three output predictions

After confirming the satisfactory performance of the U-Net model with one channel, I conducted further tests using three input channels. As seen in Figure 5.9b, the workflow can also be done using three input data, three labels, and three outputs. The three channels would be negative LRT, positive LRT (since the data has a split-spread geometry) and PRT. To do that I split the Hybrid RT (Figure 5.10c) panel into three: $-$ Linear (Figure 5.10a), $+$ Linear (Figure 5.10b) and Parabolic (Figure 5.10c) RT. Each one of these three portions of the Hybrid RT (each as one channel) will be the input for the U-Net. The same is done for the labels (Figures 5.10a', 5.10b' and 5.10c'). In order to do that, some adjustments need to be made on the x-axis (p and q) to ensure that all three panels have the same size since the DL methodology process is done pixel-by-pixel.

Since in field data, the left and right sides of a split-spread shot do not have exactly the same features, therefore the $-$ LRT (Figure 5.10a') and the $+$ LRT (Figure 5.10b') do not have symmetrically the same features. Thus, it is interesting to see that the U-Net predicted of them, seen respectively in Figures 5.10a"and 5.10b" are also not similar. Furthermore, these predictions do not look like their input (Figures 5.10a and 5.10b), which still have

ground roll in it, showing that the U-Net was able to produce a good prediction of an attenuated version using the Hybrid RT. Figure 5.10c" shows the U-Net prediction of the PRT portion of the Hybrid RT, and apparently, the U-Net tried to fit some of the LRT in that portion of the prediction, but that is still unclear since this test had three separate inputs and three separate outputs.

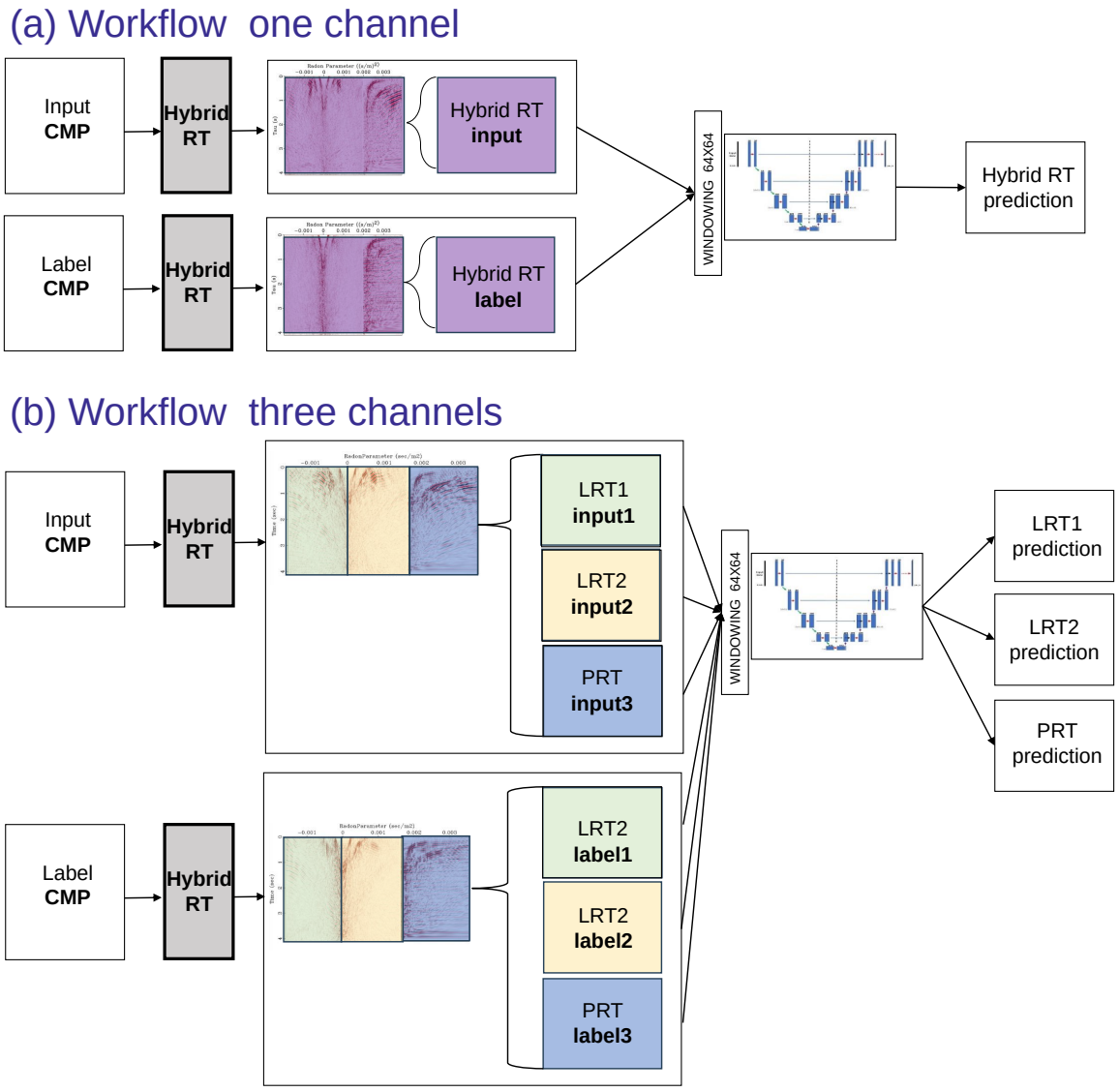


Figure 5.9: In a one-channel workflow (a), there is one input channel and correspondent label with the U-Net prediction being the one Hybrid RT panel. In the three channels workflow (b), there are three input channels (LRT1, LRT2 and PRT) and their correspondent three labels, but in this case, the U-Net will predict three RT panels as outputs: LRT1, LRT2 and PRT.

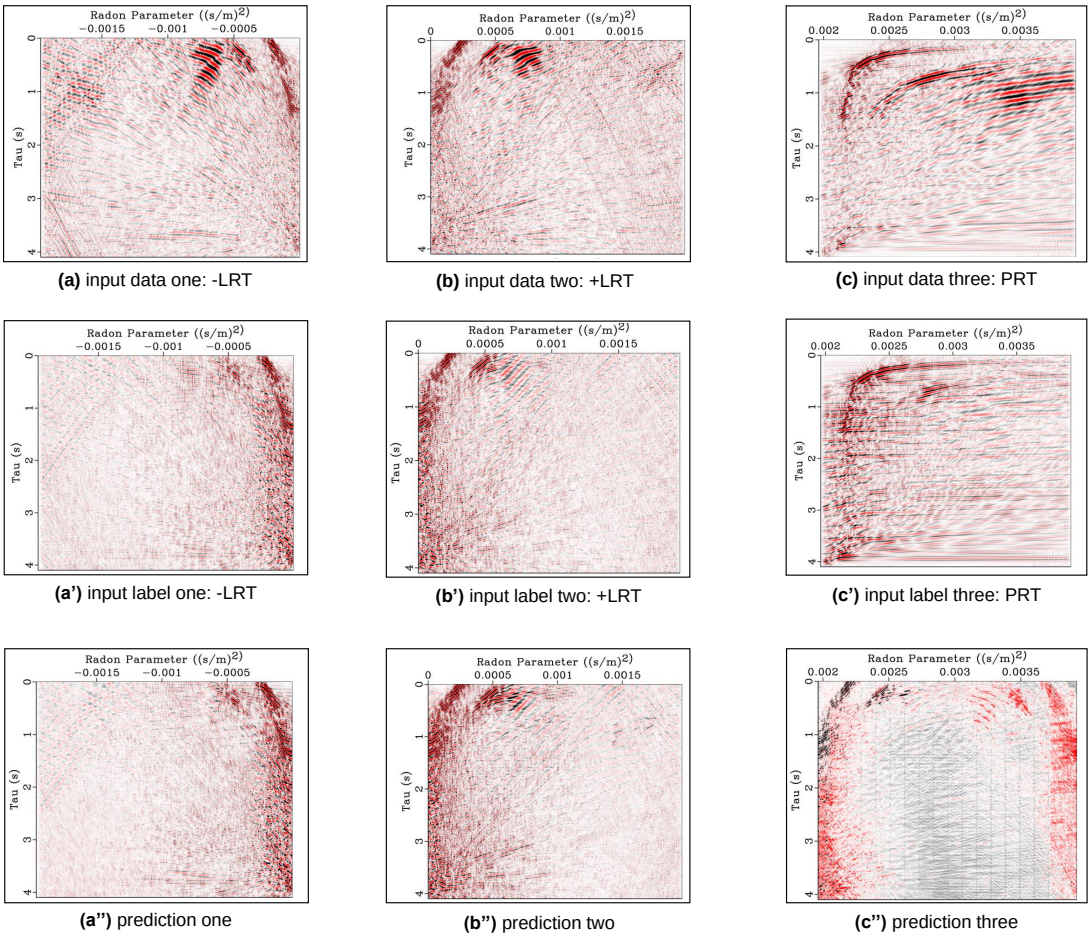


Figure 5.10: The input data, Spring Coulee, Hybrid RT, was split into 3 channels: (a) − Linear, (b) + Linear, and (c) Parabolic RT panels. The LRT has a negative side since the geometry of acquisition was split-spread. The same split was done with its input label (FK-filtered version of the input data): (a') − Linear, (b') + Linear, and (c') Parabolic RT panels of enhanced reflections only. By using 3 input data and 3 input labels, the U-Net can predict 3 output data: (a") LRT1, (b") LRT2, and (c") PRT

5.2 Conclusions

This chapter examined the efficiency of using the U-Net methodology to attenuate ground roll noise effectively. The objective was to explore how the DL-based prediction of ground roll-attenuated seismic datasets interacts with the non-linear characteristics of field seismic data.

DL can predict complicated patterns because of its non-linearity, but it is unclear if this methodology contributes to generalization. A central focus of this investigation was the relationship between different RT spaces, known as crosstalk. The study aimed to provide insights into how DL techniques handle seismic data intricacies by inputting various RT panels into the network. Consistency in the U-Net architecture and its hyperparameters was maintained across all numerical tests to make a fair comparison among all the tests.

Ground roll, characterized by its strong amplitude and low frequency compared to other seismic events, presents a unique challenge. However, this is insufficient for non-repeatable patterns like sampling artifacts. These are a consequence of the complexity of ground roll amplitudes when mapping to the Radon space. Hybrid RT, combining linear and parabolic Radon spaces, is explored to mitigate ground roll interference while retaining reflection information. The absence of sparseness enforcement in ground roll prediction models is due to the inherently non-sparse characteristics of ground roll and reflection basis functions.

I am using a supervised ML method, but while using F-K filtered as a label seems reasonable, the simple fact that I need to select a label introduces some uncertainty for the problem, as generating a label with only reflections for field data is not straightforward.

Mapping hyperbolic shapes from data to model space introduces crosstalk between different RT spaces, complicating the separation of coherent noise and reflections. Moreover, dealing with aliased events adds further complexity to the problem.

The choice of labels is really important for applying this U-Net methodology. Tests involving one and three RT channels reveal insights into the leakage of ground roll among the channels used, with the three-channel approach showing fair U-Net predictions. However,

challenges persist in fully disentangling ground roll from reflection data, given their overlap in RT spaces.

Chapter 6

Conclusions

In this thesis, I have explored applying a deep learning (DL) technique to address challenges in seismic data processing, particularly focusing on separating signal from noise in the model space. By leveraging DL's capability to recognize non-linear patterns in images, I have demonstrated the effectiveness of ML-based approaches in separating multiples/ground roll and reflections in the Radon transform (RT) domain, even in scenarios with overlapping events in the model domain.

A central idea of this work is the utilization of ML as an alternative to traditional sparseness techniques, which can struggle with complex and overlapping events encountered in field seismic data. Through a pixel-by-pixel (or window-by-window) approach, I equipped the network with the intelligence to handle overlapping information and other data complexities, effectively complementing or even surpassing traditional sparseness techniques.

This thesis' methodology also focused on maximizing the utility of RTs by incorporating multiple channels of information, analogous to using multiple colour channels in an RGB image. In Chapter 4, the experiments examined the efficiency of employing various methodologies, including Hyperbolic RT (HRT) and Parabolic RT, for training a U-Net model to predict multiples and primary reflections. While the U-Net demonstrated partial success in predicting multiples, challenges arose due to sampling artifacts and the complexity of geo-

logical models. Incorporating diverse geological models during training improved prediction accuracy, although challenges with noise and sampling artifacts persisted.

I also recognized the importance of addressing challenges posed by noise and sampling artifacts and the difficulty distinguishing between coherent noise and signal, particularly in scenarios with overlapping primaries and multiples. This analyzes highlighted the practical utility of the network's outputs for qualitative interpretation. However, there are challenges in subtracting predicted multiples from the original input due to the pixel-by-pixel nature of the methodology.

Moreover, in Chapter 5, using field data made choosing appropriate labels more prominent, considering the impossibility of generating labels with only reflections for field data. Tests involving different parts of the Hybrid RT as channels provided insights into the leakage of ground roll among the channels used, with the three-channel approach showing promising results but challenges persisting in fully disentangling ground roll from reflection data.

While machine learning holds promise for seismic data processing, systematic experimentation and a deeper understanding of the underlying physics are essential for maximizing its effectiveness. Continued research efforts are significant in addressing these challenges and unlocking the full potential of machine learning in seismic data processing.

6.1 Recommendations for future work

Looking ahead, efforts to mitigate overfitting, address windowing issues (challenges of non-stationarity in RT space) and understand the impact of the RT butterfly effect on the prediction will be important. Moreover, the transition to experiments with more complex geologic models and field data would be ideal. Analyzing how the network evaluates coherent noise that is not well-behaved would advance the understanding of how it identifies patterns in the RT panels. For example, variable geometry commonly occurring in field data can influence this workflow.

Further improvement could include incorporating data and model weights in the transforms. These would allow the mapping of specific events with a preference for particular operators (LRT or PRT). Finding a better ground truth (label) can lead to a better U-Net prediction. While this will not change the current workflow, it can positively affect the outcome.

Bibliography

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

Alali, A. and Alkhalifah, T. (2023). Integrating u-nets into a multi-scale waveform inversion for salt body building. *arXiv preprint arXiv:2304.02758*.

Alali, A., Smith, R., Nivlet, P., Bakulin, A., and Alkhalifah, T. (2021). Time-lapse seismic cross-equalization using temporal convolutional networks. In *SEG International Exposition and Annual Meeting*, page D011S060R008. SEG.

Albawi, S., Mohammed, T. A., and Al-Zawi, S. (2017). Understanding of a convolutional neural network. In *2017 international conference on engineering and technology (ICET)*, pages 1–6. IEEE.

Alkhalifah, T., Birnie, C., Harsuko, R., Wang, H., and Ovcharenko, O. (2022). Deep earth: Leveraging neural networks for seismic exploration objectives. In *SEG International Exposition and Annual Meeting*, page D011S123R004. SEG.

Bekara, M. and Van der Baan, M. (2007). Local singular value decomposition for signal enhancement of seismic data. *Geophysics*, 72(2):V59–V65.

Bengio, Y. (2012). Practical recommendations for gradient-based training of deep architectures. In *Neural networks: Tricks of the trade: Second edition*, pages 437–478. Springer.

Berkhout, A. and Verschuur, D. J. (1994). Multiple technology: Part 2, migration of multiple reflections. In *SEG Technical Program Expanded Abstracts 1994*, pages 1497–1500. Society of Exploration Geophysicists.

Berndt, C. and Moore, G. F. (1999). Dependence of multiple-attenuation techniques on the geologic setting: A case study from offshore taiwan. *The Leading Edge*, 18(1):74–80.

Beylkin, G. (1983). Inversion of the generalized radon transform. In *Inverse Optics I*, volume 413, pages 32–39. SPIE.

Beylkin, G. (1987). Discrete radon transform. *IEEE transactions on acoustics, speech, and signal processing*, 35(2):162–172.

Bugge, A. J., Evensen, A. K., Lie, J. E., and Nilsen, E. H. (2021). Demonstrating multiple attenuation with model-driven processing using neural networks. *The Leading Edge*, 40(11):831–836.

Cameron, G. and Vestrum, R. (2022). Convolutional neural networks to augment psdm velocity model building. In *Asia Petroleum Geoscience Conference and Exhibition (APGCE)*, volume 2022, pages 1–5. European Association of Geoscientists & Engineers.

Candes, E. J. and Donoho, D. L. (2005). Continuous curvelet transform: Ii. discretization and frames. *Applied and Computational Harmonic Analysis*, 19(2):198–222.

Cary, P. and Zhang, C. (2009). Ground roll attenuation via svd and adaptive subtraction. In *Frontiers+ Innovation–2009 CSPG CSEG CWLS Convention, Calgary*, pages 372–375.

Chapman, C. (1981). Generalized radon transforms and slant stacks. *Geophysical Journal International*, 66(2):445–453.

Claerbout, J. (1985a). Ground roll and radial traces.

Claerbout, J. F. (1985b). *Imaging the earth's interior*, volume 1. Blackwell scientific publications Oxford.

Claerbout, J. F. (2004). *Earth soundings analysis: Processing versus inversion*. Blackwell Scientific Publications.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dragoset, W. H. and Jeričević, Ž. (1998). Some remarks on surface multiple attenuation. *Geophysics*, 63(2):772–789.

Durall, R., Ghanim, A., Ettrich, N., and Keuper, J. (2022). Dissecting u-net for seismic application: An in-depth study on deep learning multiple removal. *arXiv preprint arXiv:2206.12112*.

Embree, P., Burg, J. P., and Backus, M. M. (1963). Wide-band velocity filtering—the pie-slice process. *Geophysics*, 28(6):948–974.

Falk, T., Mai, D., Bensch, R., Çiçek, Ö., Abdulkadir, A., Marrakchi, Y., Böhm, A., Deubner, J., Jäckel, Z., Seiwald, K., et al. (2019). U-net: deep learning for cell counting, detection, and morphometry. *Nature methods*, 16(1):67–70.

Fernandez, M., Ettrich, N., Delescluse, M., Rabaute, A., and Keuper, J. (2023). Deep learning strategies for seismic demultiple. In *Third EAGE Digitalization Conference and Exhibition*, volume 2023, pages 1–5. European Association of Geoscientists & Engineers.

Fomel, S., Sava, P., Vlad, I., Liu, Y., Jennings, J., Browaeys, J., Bashkardin, V., Godwin, J., Song, X., and Hennenfent, G. (2012). Madagascar software package and reproducible research.

Fontes, P. H. L. and Trad, D. O. (2023a). A machine learning alternative to sparseness. *CREWES Research Report*, 35.

Fontes, P. H. L. and Trad, D. O. (2023b). The use of u-net and hyperbolic radon transform for multiple attenuation. *GeoConvention, Conference Abstract*.

Fontes, P. H. L. and Trad, D. O. (2024). A machine learning alternative to sparseness. *GeoConvention, Conference Abstract*.

Fontes, P. H. L., Trad, D. O., and Sánchez-Galvis, I. J. (2022). The use of u-net and radon transforms for multiple attenuation. *CREWES Research Report*, 34.

Foster, D. J. and Mosher, C. C. (1992). Suppression of multiple reflections using the radon transform. *Geophysics*, 57(3):386–395.

Freire, S. L. and Ulrych, T. J. (1988). Application of singular value decomposition to vertical seismic profiling. *Geophysics*, 53(6):778–785.

Fukushima, K. (1980). A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biol. Cybern.*, 36:193–202.

Géron, A. (2019). *Hands-on machine learning with Scikit-Learn, Keras, and Tensor-Flow*. O'Reilly Media, Inc.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep learning*. MIT press.

Hampson, D. (1986). Inverse velocity stacking for multiple elimination. *SEG Technical Program Expanded Abstracts*, pages 422–424.

Hampson, D. (1987). The discrete radon transform: a new tool for image enhancement and noise suppression. In *SEG Technical Program Expanded Abstracts 1987*, pages 141–143. Society of Exploration Geophysicists.

Harlan, W. S., Claerbout, J. F., and Rocca, F. (1984). Signal/noise separation and velocity estimation. *Geophysics*, 49(11):1869–1880.

Harsuko, R. and Alkhalifah, T. A. (2022). Storseismic: A new paradigm in deep learning for seismic processing. *IEEE Transactions on Geoscience and Remote Sensing*, 60:1–15.

He, K., Zhang, X., Ren, S., and Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034.

Herrmann, P., Mojesky, T., Magesan, M., and Hugonnet, P. (2000). De-aliased, high-resolution radon transforms. In *SEG International Exposition and Annual Meeting*, pages SEG–2000. SEG.

Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*.

Hu, Y., Wang, L., Cheng, F., Luo, Y., Shen, C., and Mi, B. (2016). Ground-roll noise extraction and suppression using high-resolution linear radon transform. *Journal of Applied Geophysics*, 128:8–17.

Huang, S. and Trad, D. (2023). Convolutional neural-network-based reverse-time migration with multiple reflections. *Sensors*, 23(8):4012.

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. pmlr.

James, G., Witten, D., Hastie, T., Tibshirani, R., and Taylor, J. (2023). *An introduction to statistical learning: With applications in python*. Springer Nature.

Kabir, M. M. N. and Marfurt, K. J. (1999). Toward true amplitude multiple removal. *The Leading Edge*, 18(1):66–73.

Kingma, D. P. and Ba, J. (2015). Adam: A method for stochastic optimization. In *Proc. 3rd Int. Conf. Learn. Representations*.

Kragh, E. and Peardon, L. (1995). Ground roll and polarization. *First Break*, 13(9).

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

Li, Z., Jia, J., Lu, Z., Jiao, J., and Yu, P. (2022). Seismic velocity anomalies detection based on a modified u-net framework. *Applied Sciences*, 12(14):7225.

Li, Z., Sun, N., Gao, H., Qin, N., and Li, Z. (2021). Adaptive subtraction based on u-net for removing seismic multiples. *IEEE Transactions on Geoscience and Remote Sensing*, 59(11):9796–9812.

Liu, D., Sacchi, M. D., Wang, X., and Chen, W. (2023a). Unsupervised deep learning for ground roll and scattered noise attenuation. *IEEE Transactions on Geoscience and Remote Sensing*.

Liu, S., Birnie, C., Bakulin, A., Dawood, A., Silvestrov, I., and Alkhalifah, T. (2023b). A self-supervised scheme for ground roll suppression. *arXiv preprint arXiv:2310.13967*.

Ma, Y. and Li, W. (2023). Robust deep learning seismic velocity inversion over varying acquisitions. *IEEE Geoscience and Remote Sensing Letters*, 20:1–5.

Marfurt, K. J., Schneider, R. V., and Mueller, M. C. (1996). Pitfalls of using conventional and discrete radon transforms on poorly sampled data. *Geophysics*, 61(5):1467–1482.

McCormack, M. D. (1991). Neural computing in geophysics. *The Leading Edge*, 10(1):11–15.

McCulloch, W. S. and Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5:115–133.

Moore, I. and Kostov, C. (2002). Stable, efficient, high-resolution radon transforms. In *64th EAGE Conference & Exhibition*, pages cp–5. European Association of Geoscientists & Engineers.

Park, M. J. and Sacchi, M. D. (2020). Automatic velocity analysis using convolutional neural network and transfer learning. *Geophysics*, 85(1):V33–V43.

Peacock, K. and Treitel, S. (1969). Predictive deconvolution-theory and practice. *Geophysics*, 34(2):155–169.

Perkins, C. and Zwaan, M. (2000). Ground roll attenuation. In *62nd EAGE Conference & Exhibition*, pages cp–28. European Association of Geoscientists & Engineers.

Pham, N. and Li, W. (2022). Physics-constrained deep learning for ground roll attenuation. *Geophysics*, 87(1):V15–V27.

Porsani, M. J., Silva, M. G., Melo, P. E., and Ursin, B. (2010). Svd filtering applied to ground-roll attenuation. *Journal of Geophysics and Engineering*, 7(3):284.

Radon, J. (1917). Uber die bestimmung von funktionen durch ihre integralwerte längs gewisser mannigfaltigkeiten. *Berichte über die Verhandlungen der Königlich-Sächsischen Akademie der Wissenschaften zu Leipzig, Mathematisch-Physische Klasse*, 69:262–277.

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

Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386.

Röth, G. and Tarantola, A. (1994). Neural networks and inversion of seismic data. *Journal of Geophysical Research: Solid Earth*, 99(B4):6753–6768.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *nature*, 323(6088):533–536.

Russell, B. (2019). Machine learning and geophysical inversion—a numerical study. *The Leading Edge*, 38(7):512–519.

Saad, O. M., Fomel, S., Abma, R., and Chen, Y. (2023). Unsupervised deep learning for 3d interpolation of highly incomplete data. *Geophysics*, 88(1):WA189–WA200.

Sacchi, M. (2002). Statistical and transform methods in geophysical signal processing. *Department of Physics, University of Alberta*.

Sacchi, M. D. and Ulrych, T. J. (1995a). High-resolution velocity gathers and offset space reconstruction. *Geophysics*, 60(4):1169–1177.

Sacchi, M. D. and Ulrych, T. J. (1995b). High-resolution velocity gathers and offset space reconstruction. *Geophysics*, 60(4):1169–1177.

Sacchi, M. D. and Ulrych, T. J. (1995c). Model re-weighted least-squares radon operators. *SEG Technical Program Expanded Abstracts*, pages 616–618.

Sánchez-Galvis, I.-J., Serrano-Luna, J.-O., Niño-Niño, C.-A., Sierra, D.-A., and Agudelo-Zambrano, W.-M. (2016). Svd polarization filter taking into account the planarity of ground roll energy. *CT&F-Ciencia, Tecnología y Futuro*, 6(3):5–24.

Schlumberger (2015). Vista. `https://www.slb.com/products-and-services/delivering-digital-at-scale/software/vista/vista-desktop-seismic-data-processing-software`.

Sheriff, R. E. (2002). *Encyclopedic dictionary of applied geophysics*. Society of exploration geophysicists.

Smith, K. (2017). Machine learning assisted velocity autopicking. In *SEG Technical Program Expanded Abstracts 2017*, pages 5686–5690. Society of Exploration Geophysicists.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

Starck, J.-L., Candès, E. J., and Donoho, D. L. (2002). The curvelet transform for image denoising. *IEEE Transactions on image processing*, 11(6):670–684.

Sun, J., Innanen, K., Zhang, T., and Trad, D. (2023). Implicit seismic full waveform inversion with deep neural representation. *Journal of Geophysical Research: Solid Earth*, 128(3):e2022JB025964.

Sun, J., Niu, Z., Innanen, K. A., Li, J., and Trad, D. O. (2020a). A theory-guided deep-learning formulation and optimization of seismic waveform inversion. *Geophysics*, 85(2):R87–R99.

Sun, J., Slang, S., Elboth, T., Greiner, T. L., McDonald, S., and Gelius, L.-J. (2020b). Attenuation of marine seismic interference noise employing a customized u-net. *Geophysical Prospecting*, 68(3):845–871.

Sun, J., Slang, S., Elboth, T., Larsen Greiner, T., McDonald, S., and Gelius, L.-J. (2020c). A convolutional neural network approach to deblending seismic data. *Geophysics*, 85(4):WA13–WA26.

Taner, M. (1980). Long-period sea-floor multiples and their attenuation. *Geophys. Prospect*, 28:30–48.

Taner, M. T., O'Doherty, R. F., and Koehler, F. (1995). Long period multiple suppression by predictive deconvolution in the x–t domain. *Geophysical Prospecting*, 43(4):433–468.

Thorson, J. R. (1984). *Velocity-stack and slant-stack inversion methods*. PhD thesis, Stanford University.

Thorson, J. R. and Claerbout, J. F. (1985). Velocity-stack and slant-stack stochastic inversion. *Geophysics*, 50(12):2727–2741.

Trad, D. (2001). *Implementations and applications of the sparse Radon transform*. PhD thesis, The University of British Columbia.

Trad, D., Hargreaves, N., verWest, B., and Wombell, R. (2004). Multiple attenuation using an apex shift radon transform. In *Offshore Technology Conference*, pages OTC–16944. OTC.

Trad, D., Ulrych, T., and Sacchi, M. (2003). Latest views of the sparse radon transform. *Geophysics*, 68(1):386–399.

Trad, D. O. (2003). Interpolation and multiple attenuation with migration operators. *Geophysics*, 68(6):2043–2054.

Trad, D. O. (2022). Combining classical processing with deep learning. *CREWES Research Report*, 34.

Trad, D. O., Sacchi, M. D., and Ulrych, T. J. (2001). A hybrid linear-hyperbolic radon transform. *Journal of Seismic Exploration*, 9(4):303–318.

Treitel, S., Gutowski, P., and Wagner, D. (1982). Plane-wave decomposition of seismograms. *Geophysics*, 47(10):1375–1401.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.

Verschuur, D. (1992). Surface-related multiple elimination in terms of huygens' sources. *Journal of Seismic Exploration*, 1:49–59.

Verschuur, D. and Berkhout, A. (2015). From removing to using multiples in closed-loop imaging. *The Leading Edge*, 34(7):744–759.

Weglein, A. B. (1999). Multiple attenuation: an overview of recent advances and the road ahead (1999). *The Leading Edge*, 18(1):40–44.

Weglein, A. B. (2016). Multiples: Signal or noise? *Geophysics*, 81(4):V283–V302.

Weglein, A. B., Gasparotto, F. A., Carvalho, P. M., and Stolt, R. H. (1997). An inverse-scattering series method for attenuating multiples in seismic reflection data. *Geophysics*, 62(6):1975–1989.

Wiggins, R. A. (1966). ω-k filter design. *Geophysical Prospecting*, 14:427–440.

Wu, H., Zhang, B., Lin, T., Li, F., and Liu, N. (2019). White noise attenuation of seismic trace by integrating variational mode decomposition with convolutional neural network. *Geophysics*, 84(5):V307–V317.

Wu, H., Zhang, B., and Liu, N. (2022). Self-adaptive denoising net: Self-supervised learning for seismic migration artifacts and random noise attenuation. *Journal of Petroleum Science and Engineering*, 214:110431.

Wu, X., Geng, Z., Shi, Y., Pham, N., Fomel, S., and Caumon, G. (2020). Building realistic structure models to train convolutional neural networks for seismic structural interpretation. *Geophysics*, 85(4):WA27–WA39.

Xiao, C., Bancroft, J., Brown, J., and Cao, Z. (2003). Multiple suppression: A literature review. *CREWES Research Report*, 15.

Xue, Y., Shen, H., Jiang, M., Feng, L., Guo, M., and Wang, Z. (2022). A fast sparse hyperbolic radon transform based on convolutional neural network and its demultiple application. *IEEE Geoscience and Remote Sensing Letters*, 19:1–5.

Yang, L., Chen, W., Liu, W., Zha, B., and Zhu, L. (2020). Random noise attenuation based on residual convolutional neural network in seismic datasets. *Ieee Access*, 8:30271–30286.

Yilmaz, Ö. (1989). Velocity-stack processing. *Geophysical Prospecting*, 37(4):357–382.

Yilmaz, Ö. (2001). *Seismic data analysis: Processing, inversion, and interpretation of seismic data*. Society of exploration geophysicists.

Yu, S. and Ma, J. (2021). Deep learning for geophysics: Current and future trends. *Reviews of Geophysics*, 59(3):e2021RG000742.

Yu, S., Ma, J., and Wang, W. (2019). Deep learning for denoising. *Geophysics*, 84(6):V333–V350.

Zeng, Y., Jiang, K., and Chen, J. (2019). Automatic seismic salt interpretation with deep convolutional neural networks. In *Proceedings of the 2019 3rd international conference on information system and data mining*, pages 16–20.

Zhang, C. and van der Baan, M. (2021). Ground-roll attenuation using a dual-filter-bank convolutional neural network. *IEEE Transactions on Geoscience and Remote Sensing*, 60:1–11.

Zhang, T., Sun, J., Trad, D., and Innanen, K. (2023). Multilayer perceptron and bayesian neural network based elastic implicit full waveform inversion. *IEEE Transactions on Geoscience and Remote Sensing*.

Zhou, B. and Greenhalgh, S. A. (1994). Linear and parabolic tau-p transforms revisited. *Geophysics*, 59(7):1133–1149.