Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Natural Sciences
Department of Mathematical Sciences

Sindre Henriksen

Tracking of Horizons in Seismic Data Using a Hidden Markov Model

Master's thesis in Applied Physics and Mathematics

Supervisor: Henning Omre

July 2020

NTNU
Norwegian University of
Science and Technology

Sindre Henriksen

Tracking of Horizons in Seismic Data Using a Hidden Markov Model

Master's thesis in Applied Physics and Mathematics
Supervisor: Henning Omre
July 2020

Norwegian University of Science and Technology
Faculty of Natural Sciences
Department of Mathematical Sciences

NTNU
Norwegian University of
Science and Technology

Preface

This Master's thesis concludes my Master of Science degree in physics and mathematics at the Norwegian University of Science and Technology (NTNU), where I have specialized in statistics. The thesis builds upon my work in the specialization course TMA4500.

Apart from writing this report, many hours have been spent implementing the algorithms developed in this thesis and experimenting with them. A significant part of the work leading to the final form of the algorithms, as presented in this thesis, is not discussed here. The codebase in its entirety can be found at my personal GitHub page (see Henriksen, n.d.).

I have worked part-time in the start-up company RagnaRock Geo (see RagnaRock Geo, n.d.) developing seismic interpretation software, mostly based on deep learning methods. RagnaRock Geo has been helpful by allowing the use and extension of RagnaRock's computer code base for generating simple synthetic seismic data.

I want to thank my supervisor Henning Omre for patiently working with me on this project, pushing me when I needed to be pushed and giving credit when credit was due. I also want to thank my fellow students for six great years at NTNU and in Trondheim, and my partner for her support.

Sindre Henriksen, June 2020

Abstract

Automated seismic data interpretation facilitates the creation of precise maps of the underground at low costs. In this thesis, a hidden Markov model for geological horizon propagation is defined, and two tracking algorithms based on the model are developed. One of the tracking algorithms, which builds on the Viterbi algorithm (Viterbi, 1967), shows promise when experimenting with synthetic data. However, experiments on real seismic data prove that improvements need to be implemented if the algorithm is to challenge state of the art algorithms.

Sammendrag

Automatisert tolkning av seismisk data muliggjør kartlegging av undergrunnen til lave kostnader. I denne tesen definerer vi en skjult Markov model for å forklare hvordan geologiske horisonter propagerer gjennom undergrunnen, og vi utvikler to tolkningsalgoritmer basert på modellen. En av algoritmene bygger på Viterbi-algoritmen (Viterbi, 1967). Denne algoritmen gir lovende resultater ved forsøk med syntetiske data. Forsøk med ekte seismiske data viser at den må utvikles ytterligere for å potensielt konkurrere med tolkningsalgoritmer som brukes i dag.

Contents

List of Algorithms

Chapter 1

Introduction

Interpretations and maps of the underground are essential for industries like construction, mining, and the petroleum industry. To gain insights about the underground, we can use prior knowledge about the area, drill tests, seismic data, or other sources of information. In this thesis, we work with seismic data. First, in this chapter, we present an introduction to exploration geophysics and the problem we attempt to solve. More material on exploration geophysics can be found in e.g. Simm et al. (2014) and Landrø and Amundsen (2018), while Hobbs et al. (1976) contains an introduction to structural geology.

Seismic data is obtained by emitting seismic waves into the ground and measuring the amplitudes of reflected waves. One kind of seismic waves is sound waves. Sound waves are partly reflected, and partly transmitted, when they hit a wall. This behavior is general for all seismic waves; they are partly reflected at boundaries between different materials. A boundary which reflects seismic signals is called a reflector. More specifically, we call the surface separating two layers of different rock in the underground a horizon.

There are many properties to map in the underground, like horizons, rock types, salt bodies, faults, injectites, and fluids. Mapping of horizons and faults are usually among the first steps of interpreting offshore seismic data. Usually, the ultimate goal is to locate hydrocarbons. However, other industries depend on underground maps offshore as well, like the offshore wind industry and the potentially booming industry of deep-sea mining (Letman, 2018).

In this report, we discuss mapping, or tracking, of horizons in offshore seismic data. Seismic data is usually represented as data cubes termed seismic volumes. A seismic volume can consist of several terabytes of data and contain billions of data points, and there are generally many horizons in each cube. Consequently, tracking of horizons can be a complex and time-consuming task.

Due to the complexity of tracking seismic horizons, there is a significant market for algorithms used to semi-automate the task. We refer to computer programs implementing such algorithms as autotrackers. The primary user group of seismic interpretation software is domain experts with backgrounds in geology or geophysics. In this context, we refer to them as seismic interpreters.

When tracking a horizon the interpreter selects one or more locations that belong to the horizon, often called seed points. Based on the seed points and the seismic data, the task of the autotracker is to map the horizon throughout the seismic volume.

Autotrackers significantly improve the efficiency of interpreters. However, it is challenging to precisely autotrack complex seismic horizons, like horizons spanning over many faults and horizons at depths where the data quality is poor. The number of faults increases as the depth increases, and the quality of the data decreases with depth. Since accurate mappings at depths of up to several kilometers beneath the seabed may be required to find hydrocarbons, precise and efficient autotrackers are valuable.

Several companies and researchers are working on automating horizon tracking. Different approaches have been suggested, spanning from deep learning approaches to more traditional approaches. Wu and Fomel (2018) develop a tracking algorithm based on a common approach where the first step consists of estimating slopes of reflectors throughout a seismic volume. The slope estimates are utilized to track horizons across the volume. Yu et al. (2013) use a similar two-step approach. First, they find orientations of reflectors, which is equivalent to finding slopes, by applying a log Gabor filter on frequency data. Then they iteratively expand a horizon in three dimensions from an initial point, at each iteration expanding to the location maximizing a confidence measure. Figueiredo et al. (2014) attempt to track all horizons in a volume automatically using a clustering approach. The similarity measure used in the clustering algorithm considers local similarities around voxels. Dyrendahl (2018) describes a more traditional approach based on local search methods. Autotrackers based on deep learning methods are developed by e.g. RagnaRock Geo.

All of the approaches mentioned above are fundamentally different from the one developed in this thesis. In the following chapters, a probabilistic approach to horizon tracking is presented. Relevant background information is introduced in Chapter 2. In Chapter 3, we develop a hidden Markov model for seismic horizon propagation. Two tracking algorithms are described in Chapter 4, where one is based on the Viterbi algorithm (Viterbi, 1967), and the other is a greedy algorithm. In Chapter 5 we illustrate capabilities and limitations of the model and the algorithms, on both synthetic and real seismic data. The results and different paths for further development are discussed in Chapter 6. In Chapter 7, we conclude our work.

Chapter 2

Background

In this chapter, we present some background information essential to understanding horizon autotracking challenges and to develop our model and algorithms. Section 2.1 covers seismic data collection and preprocessing procedures. In Section 2.2, we present an overview of the physics describing propagation of seismic waves through the underground. For more on seismic data collection and seismic wave propagation, see Simm et al. (2014) and Landrø and Amundsen (2018). Section 2.3 introduces relevant mathematical notation. Finally, in Section 2.4, we demonstrate that non-probabilistic inversion is unsuitable for mapping of seismic reflectors.

2.1 Seismic Data Collection and Preprocessing

There are two main groups of seismic waves: surface waves and body waves. Surface waves mostly travel along the surface of the earth; thus they are not relevant when studying seismic data. Body waves can travel through geological bodies. Body waves comprise primary waves (p-waves), which are compressional waves, and secondary waves (s-waves), which are shear waves; see Figure 2.1. S-waves do not propagate through fluids, as fluids do not support shear stresses. At seismic reflectors, part of the energy carried by p-waves is transformed into s-waves. Conversely, motion caused by s-waves induces p-waves.

As mentioned in Chapter 1, seismic data is acquired by emitting seismic signals and measuring the amplitudes of the reflected seismic waves. On land, explosives are usually used to create the seismic signals. This approach is not preferable offshore. Instead, air guns, water guns, and other sources are used. A ship tows the acoustic source and multiple receivers in a zig-zag pattern over a large area to acquire the seismic data. Two-dimensional cross sections of the seismic data parallel to the travel direction of the boat are called inlines, while cross section perpendicular to inlines are called crosslines.

Seismic waves propagating from the acoustic sources are partly reflected and partly transmitted at each geological horizon. The reflected waves travel

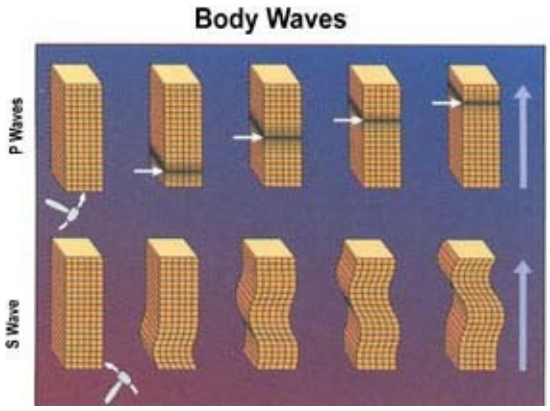


Figure 2.1: Body waves comprise p-waves and s-waves. Source: Unknown author / Public Domain.

back to the surface, and the receivers measure their amplitudes. Hence seismic data comprises two-way travel times and amplitudes. The collection of amplitudes recorded by a receiver following a single source signal emission is called a seismic trace. Seismic traces recorded by receivers under the ocean surface only encompass p-wave oscillations, since s-waves do not propagate through water. Therefore, the data considered in this study is exclusively p-wave data.

The huge amount of raw seismic data collected by the receivers at sea needs to be structured and processed to be interpretable for humans. The first processing challenge that arises is the positioning problem, illustrated in Figure 2.2. The distance in time to the location where the wave was reflected only determines the half-sphere of potential locations. The half-sphere above sea level is not considered since air itself does not reflect seismic waves. The positioning issue is handled in a process called migration. During this process, reflectors apparent in adjacently collected traces are positioned consistently on their respective half-spheres of possible locations.

Since multiple receivers are used when collecting seismic data, there are multiple data points for each discretized location in the subsurface. The data points corresponding to the same location in the subsurface are collected with the seismic signal travelling at different angles from the acoustic source; see Figure 2.3. Two-way travel times increase with the angle so that they form a hyperbola. The curvature of the hyperbola holds useful information about the speed of the seismic waves in the different subsurface layers. Furthermore, aggregating multiple reflections from each horizon location in a vertical column of the underground helps filter out noise from the data. Such an aggregation is called a common midpoint gather (CMP).

Noise in seismic data can arise from e.g. turbulence in the water around the receivers; other sources of seismic signals; and mixed rock types and otherwise heterogeneous layers. As the seismic waves travel deeper into the underground,

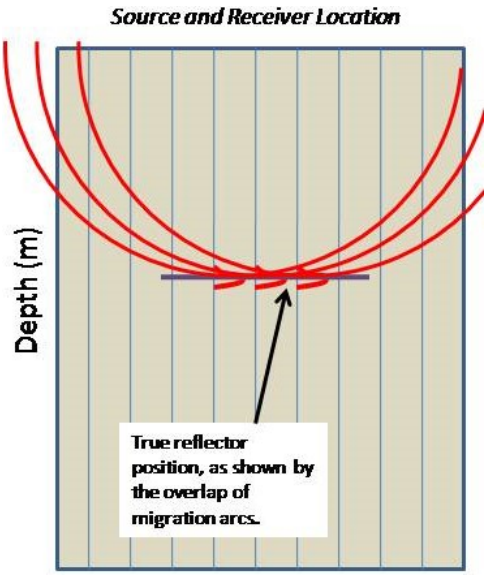


Figure 2.2: Migration of seismic data. Several measurements made close to each other are needed to determine the locations of a reflector. Source: User:NebulousAether / Wikimedia Commons / Public Domain.

they are weakened. The number of noise sources, on the other hand, increases. Thus the signal to noise ratio decreases.

Multiples and ghosts are further unwanted artifacts present in raw seismic data, as are p-waves induced by s-wave oscillations. A multiple is a seismic signal that has been reflected more than once by the same surface. See Figure 2.4 for an illustration of multiples and ghosts. One type of ghost is the source ghost; the reflection of the source signal from the surface of the sea right after transmission. The source ghost inevitably emerges since the seismic signal spreads out in all directions from the source. Ghosts are also reflected by the underground horizons. The numerous reflections of multiples and ghosts make it challenging to sort out the reflections of interest, particularly since reflections arriving simultaneously at the receivers interfere with each other.

The receivers measure seismic amplitude, and the amplitude is a result of a superposition of different waves at different frequencies. The different frequencies should ideally be separated to make the seismic data interpretable for humans, which is a complex task. However, it is practically impossible to emit only one frequency from the source, and the frequencies can change as the waves propagate. Using different frequencies also has its advantages: High frequencies lead to high resolution, while lower frequency waves reach deeper down into the underground. Furthermore, differences in behavior of low and high frequent waves while propagating through fluids can be useful to detect the fluids.

As the seismic signal propagates deeper into the underground, the signal

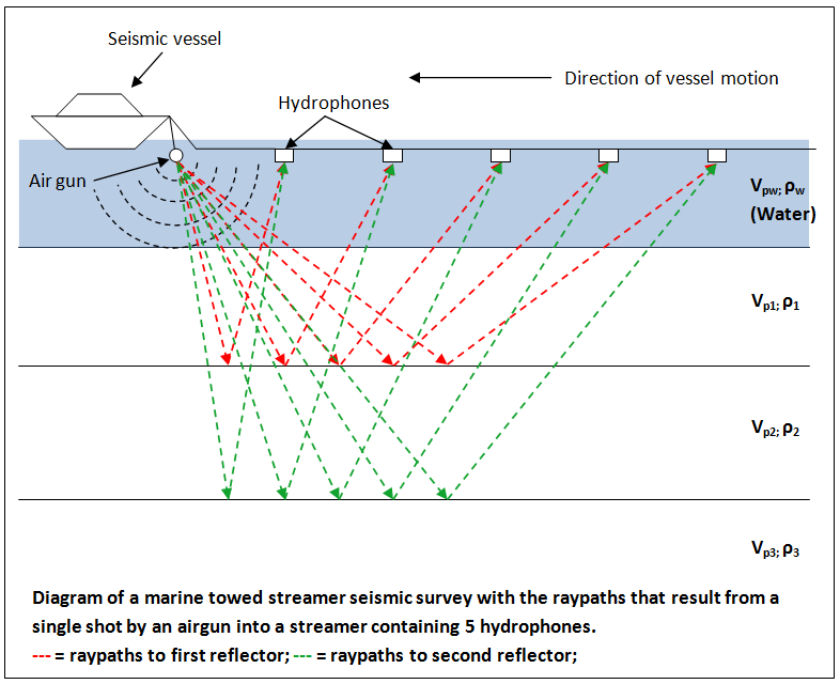


Figure 2.3: Diagram of a marine seismic survey. A streamer is a collection of cables connecting the receivers (hydrophones) and transmitting seismic signals to the vessel. Here, V is the acoustic velocity in a layer and ρ is the density of the rock. Source: User:Nwhit / Wikimedia Commons / Creative Commons BY-SA 3.0.

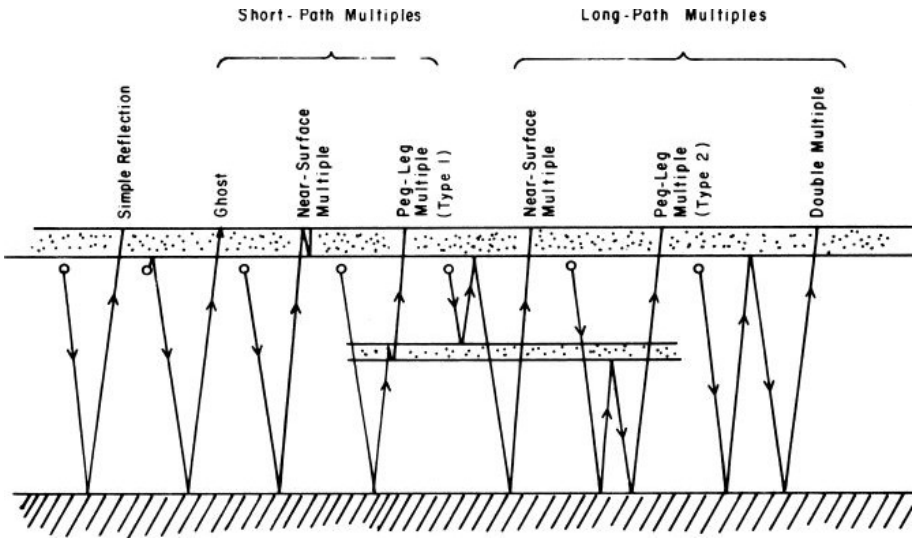


Figure 2.4: Illustration of seismic multiples and ghosts. Source: Unknown Author / SEG wiki / Creative Commons BY-SA 3.0.

becomes weaker since energy is lost on the way, e.g. to heat production, and because the area of the wavefront increases. This energy dissipation of seismic waves is called seismic attenuation. Ususally, it is attempted to increase the measured amplitudes to compensate for seismic attenuation during raw seismic data processing. We want the seismic data to look as if the seismic waves have the same energy and amplitudes as they impact each reflector, independently of the reflector's depth. This compensation makes it easier to interpret seismic data.

Handling the issues mentioned in this section, amongst others, by migrating, stacking, and cleaning the data is usually referred to as preprocessing. Every part of preprocessing is intertwined, so preprocessing is an iterative process that is run until a satisfactory result is obtained.

Clearly, there are many potential sources of error when collecting and processing seismic data. Nonetheless, seismic data is valuable since it offers large spatial coverage and is relatively cheap to acquire. Drilling wells results in more precise data, but offers low spatial coverage and is expensive. Therefore, interpretation of seismic data is used extensively as a first step of mapping the underground. Well logs can be used to give precise point data, which can be combined with seismic data. Well logs are also used to detect hydrocarbons directly after their presence is indicated by seismic data.

In this section, we have discussed some high-level properties of the seismic data collection procedure. In the following section, we give a rough outline of the geophysics behind seismic wave propagation.

2.2 Seismic Wave Propagation

Seismic waves are often treated as plane waves, i.e. waves that are constant across the plane perpendicular to the direction of travel. Since seismic waves spread out in all directions from a source, they are not truly plane waves. But as the waves travel far from the source the curvature approaches zero, so locally the plane wave assumption is a good approximation.

For a plane wave, the acoustic impedances of the rocks above and below a horizon, along with the incidence angle, determine the reflection coefficient. Generally, acoustic impedance, denoted z, is the ratio of acoustic pressure p to acoustic (oscillating) volume flow u. For plane waves, the acoustic impedance is equivalently given by the product of the rock density ρ and the plane wave acoustic velocity v. The density generally increases with depth, as the pressure from above increases. Increased density, other things equal, leads to increased acoustic impedance. Consequently, a single layer of rock can have different acoustic impedances at different locations.

The reflection coefficient of a horizon, which we denote q, determines the amplitudes of reflected waves relative to the amplitudes of incident waves. The energy of the incident waves that is not reflected carries on in transmitted waves. If the incident angle is not 90 degrees, the transmitted waves are

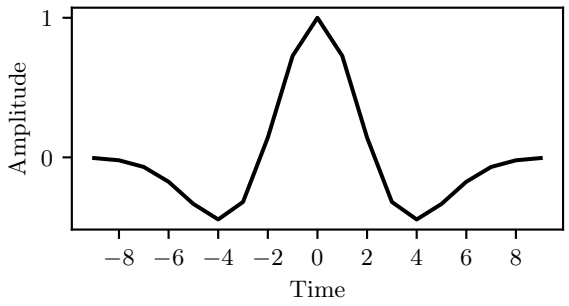


Figure 2.5: Normalized Ricker wavelet.

refracted, meaning that their direction changes. The reflection coefficient of a horizon for a plane wave at normal incidence is

$$q = \frac{z_2 - z_1}{z_2 + z_1}, \quad -1 < q < 1,$$

where z_1 and z_2 are the acoustic impedances of the medium the wave is traveling in before incidence and the second medium, respectively. A negative reflection coefficient implies an inversion of the reflected wave.

The signal emitted from a seismic source along a straight line is termed a wavelet. We denote the source wavelet by $\boldsymbol{w}$. See Figure 2.5 for an example of a so-called Ricker wavelet, also called a Mexican hat wavelet. In this study, we model seismic traces as convolutions of a wavelet $\boldsymbol{w}$ over vectors representing vertical columns of discretized reflection coefficients.

If the incidence angle of a wave is not normal, the picture gets more complicated than described above. In that case, some of the energy in the incident p-wave is reflected and transmitted as s-waves, and there is a refraction effect. The Zoeppritz equations, which we do not discuss here, describe this process. In practice, we often work with stacked traces when interpreting seismic data. A stacked trace is a combination of all the angle traces from a CMP into a single trace. When working with stacked seismic data where migration and other processing steps are readily performed, energy loss to s-waves and refraction effects can be ignored.

2.3 Notation

In this section, we define the mathematical notation necessary to develop our model and algorithms. We denote all vectors by bold lower case letters, e.g. $\boldsymbol{v}$. Matrices are denoted by bold and, with some exceptions, capitalized letters. For instance, the Wavelet matrix defined later is denoted $\boldsymbol{W}$. The zero vector is denoted $\mathbf{0}$ and the identity matrix $\boldsymbol{I}$. The context they are used in implicitly define their dimensions.

In this study, we work with three-dimensional seismic volumes. Consider a volume with domain $\mathcal{D} \subset \mathbb{R}^3$. We discretize $\mathcal{D}$ onto a regular grid $\mathcal{L} = \{(i,x,t)\,;\, i = 1,\ldots,I\,, x = 1,\ldots,X\,, t = 1,\ldots,T\}$ to simplify the problem. Here, I is the length, X is width and T the height of $\mathcal{L}$.

We denote the $(I \times X \times T)$ sesimic data matrix by $\boldsymbol{d}$. The i'th inline of $\boldsymbol{d}$ is denoted $\boldsymbol{d}_i$, and is a matrix of size $(X \times T)$. Column x of inline i is denoted $\boldsymbol{d}_{ix}$. The columns are stacked seismic traces. Element t of trace x is denoted d_{ixt}, where $d_{ixt} \in \mathbb{R}\ \forall (i,x,t) \in \mathcal{L}$. The matrix consisting of the x_1'th through x_2'th inlines is denoted $\boldsymbol{d}_{i_1:i_2}$, i.e. $\boldsymbol{d}_{i_1:i_2} = (\boldsymbol{d}_{i_1} \ldots \boldsymbol{d}_{i_2})$. Trace x_1 through x_2 is denoted $\boldsymbol{d}_{i,x_1:x_2}$. Furthermore, we let $\boldsymbol{d}_{-ix}$ be the set of all traces of $\boldsymbol{d}$ except trace ix and $\boldsymbol{d}_{-i}$ be the set of all inlines except inline i. The $(I \times X \times T)$ matrix of discretized reflection coefficients corresponding to $\boldsymbol{d}$ is denoted $\boldsymbol{r}$, with elements $r_{ixt} \in (-1,1)$. Subscript notation is analogous to what is defined for $\boldsymbol{r}$.

Throughout this report, we let $\mathcal{C}$ denote the clique system comprising all closest neighbor coordinate pairs in the first two dimensions of our grid, i.e. pairs on the form $\{(i,x),(i,x-1)\}, \{(i,x),(i,x+1)\}, \{(i,x),(i-1,x)\}$ and $\{(i,x),(i+1,x)\}$. We do not use periodic or reflective boundary conditions; we consider only the neighbor pairs actually present in the grid. We denote the set of all cliques including (i,x) by $\mathcal{C}_{ix}$ and the set of all cliques covering inline i by $\mathcal{C}_i$. Furthermore, we let $\mathcal{C}_{-ix} = \mathcal{C} \setminus \mathcal{C}_{ix}$ and $\mathcal{C}_{-i} = \mathcal{C} \setminus \mathcal{C}_i$. A clique is typically denoted c, and we let $\boldsymbol{r}_c = \{\boldsymbol{r}_x\,;\, x \in c\}$. Our clique system implicitly defines a neighborhood system; the set of neighbors of (i,x) is defined as $\mathcal{N}_{ix} = \cup_{c \in \mathcal{C}_{ix}} c \setminus (i,x)$.

Next, let $n_w > 0$ be an integer such that the wavelet used in a seismic study is a $(2n_w+1)$-vector denoted $\boldsymbol{w}$, where $\boldsymbol{w}$ is indexed relative to the center index. That is, w_0 is the center element of $\boldsymbol{w}$, w_{-n_w} is the first element and w_{n_w} is the final element. We have implicitly assumed that wavelets are of odd length.

Finally, we need some probability-related notation. Let $\boldsymbol{x}$ be a random vector with discrete sample space, and let $\boldsymbol{y}$ be a random vector with continuous sample space. Then $\mathrm{p}(\boldsymbol{x})$ is the probability mass function of $\boldsymbol{x}$ and $\mathrm{p}(\boldsymbol{y})$ is the probability density function of $\boldsymbol{y}$; i.e. the variables implicitly define the meaning of $\mathrm{p}(\cdot)$. We denote by $\mathrm{p}(\boldsymbol{x} \,|\, \boldsymbol{y})$ the conditional probability of $\boldsymbol{x}$ given $\boldsymbol{y}$. Analogous notation is used for conditional densities. Finally, $\boldsymbol{y} \sim \mathcal{N}_T(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ means that $\boldsymbol{y}$ is a T-variate normally distributed vector with mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. Implicitly, it is given that $\boldsymbol{\mu}$ is a T-vector and $\boldsymbol{\Sigma}$ is a $(T \times T)$ matrix.

2.4 Matrix Inversion Instability

In this study, we define our data model as

$$\boldsymbol{d}_{ix} = \boldsymbol{W}(\boldsymbol{r}_{ix} + \boldsymbol{\delta}_{ix}) + \boldsymbol{\epsilon}_{ix}\,, \quad i = 1,\ldots,I,\ x = 1,\ldots,n\,,$$

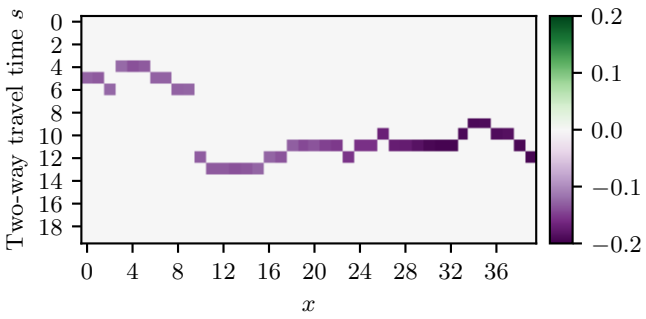


Figure 2.6: Illustration of a reflection coefficient matrix $\boldsymbol{r}_i$.

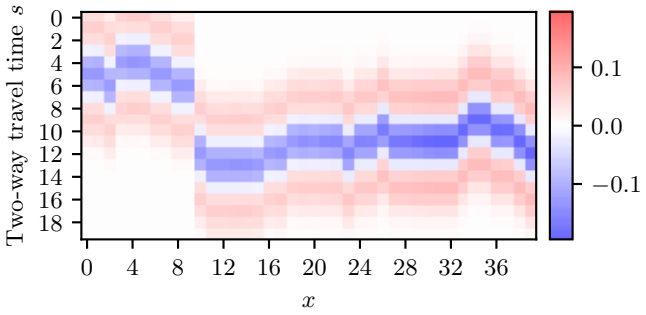


Figure 2.7: Noise-free synthetic seismic data generated using the reflection coefficient matrix illustrated in Figure 2.6.

where $\boldsymbol{d}_{ix}$ is a vector of observations, $\boldsymbol{W}$ is the wavelet matrix, $\boldsymbol{r}_{ix}$ is a vector of reflection coefficients and $\boldsymbol{\delta}_{ix}$ and $\boldsymbol{\epsilon}_{ix}$ are noise vectors. The model is described in more detail in Chapter 3.

Let us assume that the noise terms are small. Then

$$\boldsymbol{d}_{ix} \approx \boldsymbol{W}\boldsymbol{r}_{ix}\,.$$

Let $\boldsymbol{W}^{-1}$ be the inverse, or a pseudo-inverse, of $\boldsymbol{W}$. It is tempting to use

$$\hat{\boldsymbol{r}}_{ix} = \boldsymbol{W}^{-1}\boldsymbol{d}_{ix}$$

as a predictor of $\boldsymbol{r}_{ix}$. Matrix inversion is, however, an unstable operation. That is, matrix inversion is highly sensitive to noise. We illustrate this instability through an example.

Consider the two-dimensional reflection coefficient matrix $\boldsymbol{r}_i$ illustrated in Figure 2.6. First, we invert noise-free synthetic seismic data arising from $\boldsymbol{r}_i$. The data is presented in Figure 2.7. Using a pseudo-inverse of the wavelet matrix, we obtain the estimate presented in Figure 2.8. The estimate is highly accurate.

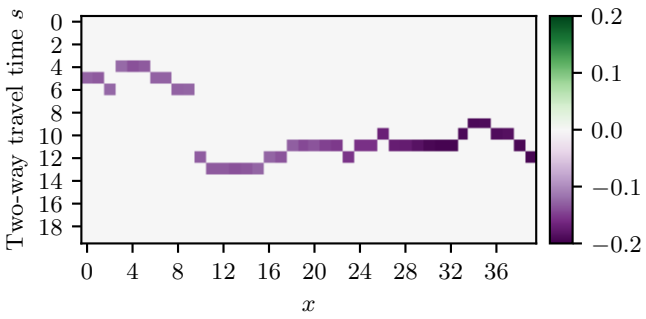


Figure 2.8: Estimate of the reflection coefficient matrix $\boldsymbol{r}_i$ from the synthetic seismic data in Figure 2.7 obtained using a pseudo-inverse of the wavelet matrix $\boldsymbol{W}$.

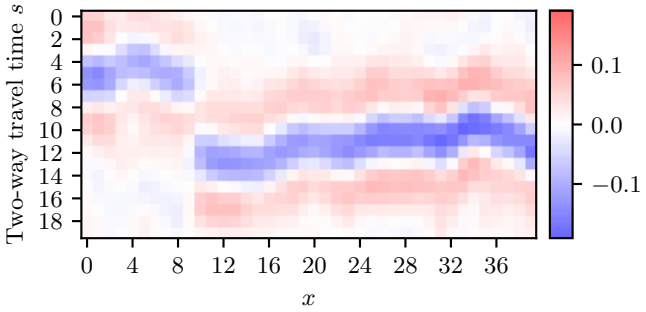


Figure 2.9: Noisy and blurred synthetic seismic data generated using the reflection coefficient matrix illustrated in Figure 2.6.

Figure 2.9 shows noisy and blurred synthetic seismic data arising from $\boldsymbol{r}_i$. We calculate the signal to noise ratio as the ratio of the sum of squares of the signal to the sum of squares of the noise, but only in a band between five points above the horizon and five points below it. This gives a signal to noise ratio of 3.91. Inverting this data using the pseudo-inverse of $\boldsymbol{W}$, we obtain the estimate presented in Figure 2.10. This estimate can hardly be said to contain any valuable information about $\boldsymbol{r}_i$. In the following chapter, we develop a probabilistic model that can be used to provide estimates less sensitive to noise in the data.

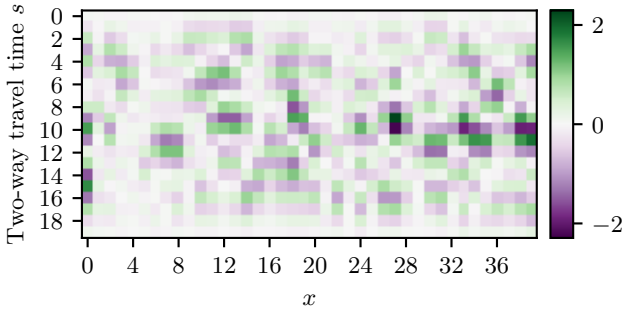


Figure 2.10: Estimate of the reflection coefficient matrix $\boldsymbol{r}$ from the synthetic seismic data in Figure 2.9 obtained using a pseudo-inverse of the wavelet matrix $\boldsymbol{W}$.

Chapter 3

Bayesian Inversion Model

We define our model in a Bayesian iversion framework. The seismic traces $\boldsymbol{d}_{ix}$, $i = 1, \ldots, I$, $x = 1, \ldots, X$ are modelled as discrete convolutions of $\boldsymbol{r}_{ix}$ and $\boldsymbol{w}$ plus noise terms $\boldsymbol{\epsilon}_{ix}$;

$$\boldsymbol{d}_{ix} = \boldsymbol{W}\boldsymbol{r}_{ix} + \boldsymbol{\epsilon}_{ix}\,, \quad i = 1, \ldots, I,\ x = 1, \ldots, X,$$

where $\boldsymbol{\epsilon_{ix}} = (\epsilon_{ix1}, \ldots, \epsilon_{ixT})$, and $\boldsymbol{W}$ is the $(T \times T)$ Toeplitz matrix with $w_{-n_w}, \ldots, w_{n_w}$ on the diagonals corresponding to the indices and zeros elsewhere. That is,

$$\boldsymbol{W} = \begin{pmatrix} w_0 & w_1 & \cdots & w_{n_w} & & & & 0 \\ w_{-1} & w_0 & \ddots & & \ddots & & & \\ \vdots & \ddots & \ddots & & & & & \\ w_{-n_w} & & & & & & \ddots & \\ & \ddots & & & & & & w_{n_w} \\ & & & & & \ddots & \ddots & \vdots \\ & & & \ddots & & \ddots & w_0 & w_1 \\ 0 & & & & w_{-n_w} & \cdots & w_{-1} & w_0 \end{pmatrix}.$$

The convolution matrix $\boldsymbol{W}$ is symmetric if $\boldsymbol{w}$ is symmetric.

To make the method developed in the following sections computationally feasible, and to get exact analytical solutions, we discretize the domain of each r_{ixt}, $(i, x, t) \in \mathcal{L}$. That is, we let $r_{ixt} \in \Omega_q$, where Ω_q is a discrete subset of $(-1, 1)$.

To further reduce the complexity of the method, we make some additional simplifying assumptions. We model the layers of rock surrounding a horizon as homogeneous layers, so that the only non-zero reflection points in $\boldsymbol{r}$ are at the horizons. That is, horizons appear as spikes in each $\boldsymbol{r}_{ix}$, $i = 1, \ldots, I$,

$x = 1, \ldots, X$. Furthermore, we consider only one horizon at a time. Thus each $\boldsymbol{r}_{ix}$ consists of one, and only one, non-zero value. Let $s_{ix} \in \Omega_s = \{1, \ldots, T\}$ be the depth of the non-zero value, and let $q_{ix} \in \Omega_q$ be the reflection coefficient of the horizon in column ix. Formally, we reparametrize $\boldsymbol{r}_{ix}$ as follows;

$$\boldsymbol{r}_{ix} = \boldsymbol{r}_{ix}(s_{ix}, q_{ix}) = q_{ix}\boldsymbol{i}(s_{ix})\,, \quad i = 1, \ldots, I, x = 1, \ldots, X\,, \tag{3.1}$$

where $\boldsymbol{i}(s_{ix})$ is a T-vector with value 1 at position s_{ix} and 0 elsewhere. We denote the domain of each $\boldsymbol{r}_{ix}$ by Ω_r. The domain is defined by Equation (3.1), Ω_s and Ω_q.

With these assumptions, there is a one to one relationship between $\boldsymbol{r}_{ix}$ and $\{s_{ix}, q_{ix}\}$. An example of a reflection coefficient matrix fulfilling the assumptions is illustrated in Figure 2.6.

By simplifying the reflection coefficients $\boldsymbol{r}$ as described above, we have introduced a modeling error. In an attempt to account for this error, we let

$$\begin{aligned} \boldsymbol{d}_{ix} &= \boldsymbol{W}(\boldsymbol{r}_{ix} + \boldsymbol{\delta}_{ix}) + \boldsymbol{\epsilon}_{ix} \\ &= \boldsymbol{W}(q_{ix}\boldsymbol{i}(s_{ix}) + \boldsymbol{\delta}_{ix}) + \boldsymbol{\epsilon}_{ix}\,, \quad i = 1, \ldots, I, x = 1, \ldots, X\,, \end{aligned} \tag{3.2}$$

where the $\boldsymbol{\delta}_{ix} = (\delta_{ix1}, \ldots, \delta_{ixT})$ represents the modelling error. See Buland and Omre (2003) for an extended discussion of this model.

We develop our model in a Bayesian inversion framework. Bayes' rule states that the posterior

$$\mathrm{p}(\boldsymbol{r} \,|\, \boldsymbol{d}) = const \times \mathrm{p}(\boldsymbol{d} \,|\, \boldsymbol{r})\mathrm{p}(\boldsymbol{r})\,, \tag{3.3}$$

where $\mathrm{p}(\boldsymbol{d} \,|\, \boldsymbol{r})$ is the likelihood, $\mathrm{p}(\boldsymbol{r})$ is the prior probability and $const = [\mathrm{p}(\boldsymbol{d})]^{-1}$ is a normalizing constant. The likelihood is the probability of the observations $\boldsymbol{d}$ given the reflection coefficients $\boldsymbol{r}$. Since $\boldsymbol{r}$ is the unknown variable matrix, $\mathrm{p}(\boldsymbol{d} \,|\, \boldsymbol{r})$ is not a density. The prior should incorporate prior domain knowledge and experience.

In Section 3.1 we define our likelihood and prior models. In Section 3.2 we derive the resulting posterior model. The derivations in these subsections are inspired by Moja et al. (2019). All results are derived considering one inline at a time. Analogous results hold for crosslines and can be obtained by rotating the system or rewriting the equations.

3.1 Likelihood and Prior Models

In this section, we define our likelihood and prior models. As stated in Equation (3.2),

$$\boldsymbol{d}_{ix} = \boldsymbol{W}(\boldsymbol{r}_{ix} + \boldsymbol{\delta}_{ix}) + \boldsymbol{\epsilon}_{ix}\,, \quad i = 1, \ldots, I, x = 1, \ldots, X\,.$$

We let both the model error terms $\boldsymbol{\delta}_{ix}$ and the noise terms $\boldsymbol{\epsilon}_{ix}$ be independent identically distributed (i.i.d.) normal random variables with expectation zero

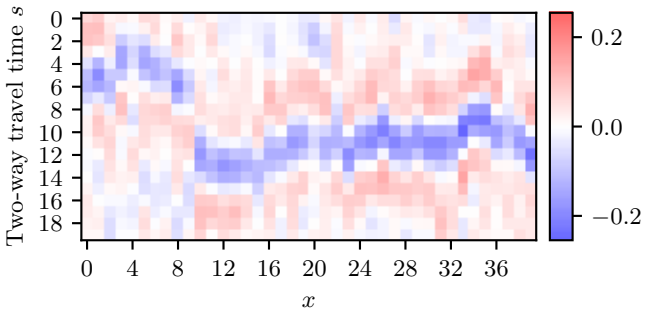


Figure 3.1: Synthetic seismic data generated according to the model described in Section 3.1 with $\sigma_\delta = 0.02$ and $\sigma_\epsilon = 0.002$, using the reflection coefficient matrix illustrated in Figure 2.6 and the wavelet in Figure 2.5.

and constant variance. Let $\boldsymbol{\delta}_{ix} \sim \mathcal{N}_T(\mathbf{0}, \sigma_\delta^2 \boldsymbol{I})$ and $\boldsymbol{\epsilon}_{ix} \sim \mathcal{N}_T(\mathbf{0}, \sigma_\epsilon^2 \boldsymbol{I})$, $i = 1, \ldots, I$, $x = 1, \ldots, X$. This implies that $[\boldsymbol{d}_{ix} \,|\, \boldsymbol{r}_{ix}] \sim \mathcal{N}_T(\boldsymbol{W}\boldsymbol{r}_{ix}, \sigma_\delta^2 \boldsymbol{W}\boldsymbol{W}^\intercal + \sigma_\epsilon^2 \boldsymbol{I})$. Figure 3.1 shows an example of synthetic seismic data generated using the model we have defined. The signal to noise ratio for the data is 4.84.

Note that independence of $\boldsymbol{\delta}_{ix}$ and $\boldsymbol{\epsilon}_{ix}$, $i = 1, \ldots, I$, $x = 1, \ldots, X$, implies that we assume the seismic traces are collected independently, i.e. that $\mathrm{p}(\boldsymbol{d}_{ix} \,|\, \boldsymbol{r}) = \mathrm{p}(\boldsymbol{d}_{ix} \,|\, \boldsymbol{r}_{ix})$. Through Equation (3.2) we have further assumed that a single-site response (ssr) model is suitable, so the likelihood becomes

$$\mathrm{p}(\boldsymbol{d} \,|\, \boldsymbol{r}) \overset{\text{indp}}{=} \prod_{i=1}^{I} \prod_{x=1}^{X} \mathrm{p}(\boldsymbol{d}_{ix} \,|\, \boldsymbol{r}) \overset{\text{ssr}}{=} \prod_{i=1}^{I} \prod_{x=1}^{X} \mathrm{p}(\boldsymbol{d}_{ix} \,|\, \boldsymbol{r}_{ix}) \tag{3.4}$$

where we have used conditional independence in the first equality. These assumptions simplify our model considerably.

The marginal likelihoods $\mathrm{p}(\boldsymbol{d}_x \,|\, \boldsymbol{r}_x)$, $x = 1, \ldots, X$, for the two-dimensional cross-section of synthetic seismic data $\boldsymbol{d}$ shown in Figure 3.1, are presented in Figure 3.2. With some exceptions, most of the weight of the likelihoods for each trace is located along the horizon.

We define the prior model as

$$\mathrm{p}(\boldsymbol{r}) = const \times \prod_{c \in \mathcal{C}} \phi(\boldsymbol{r}_c)\,, \tag{3.5}$$

where $\mathcal{C}$ is the clique system defined in Chapter 2 and $\phi(\cdot) > 0$ are called clique potentials. The constant is $[\sum_{\boldsymbol{r}'} \prod_{c \in \mathcal{C}} \phi(\boldsymbol{r}'_c)]^{-1}$. Calculation of the constant is not tractable since the domain of $\boldsymbol{r}$ is too large to sum over all possible values.

Through Equation (3.5), we have defined our prior model as a Gibbs ran-

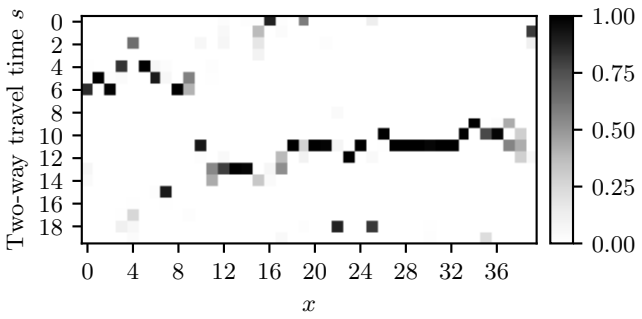


Figure 3.2: Relative (column-wise) marginal likelihoods for the synthetic seismic data presented in Figure 3.1.

dom field. Let $\phi(\boldsymbol{r}_c) > 0 \; \forall c \in \mathcal{C}$. Then, for all $i = 1, \ldots, I$, $x = 1, \ldots, X$,

$$\begin{aligned}
\mathrm{p}(\boldsymbol{r}_{ix} \,|\, \boldsymbol{r}_{-ix}) &= \frac{\mathrm{p}(\boldsymbol{r})}{\sum_{\boldsymbol{r}'_{ix}} \mathrm{p}(\boldsymbol{r}'_{ix}, \boldsymbol{r}_{-ix})} \\
&= \frac{\cancel{const \times \prod_{c \in \mathcal{C}_{-ix}} \phi(\boldsymbol{r}_c)} \prod_{c \in \mathcal{C}_{ix}} \phi(\boldsymbol{r}_c)}{\cancel{const \times \prod_{c \in \mathcal{C}_{-ix}} \phi(\boldsymbol{r}_c)} \sum_{\boldsymbol{r}'_{ix}} \prod_{c \in \mathcal{C}_{ix}} \phi(\boldsymbol{r}'_c)} \\
&= \frac{\prod_{c \in \mathcal{C}_{ix}} \phi(\boldsymbol{r}_c)}{\sum_{\boldsymbol{r}'_{ix}} \prod_{c \in \mathcal{C}_{ix}} \phi(\boldsymbol{r}'_c)} \\
&= \mathrm{p}(\boldsymbol{r}_{ix} \,|\, \boldsymbol{r}_{jy} : (j, y) \in \mathcal{N}_{ix}) \,.
\end{aligned}$$

That is, our prior Gibbs random field is also a Markov random field. Furthermore, we note that $\{\boldsymbol{r}_i\}_{i=1}^n$, can be represented by a Markov random chain;

$$\begin{aligned}
\mathrm{p}(\boldsymbol{r}_i \,|\, \boldsymbol{r}_{1:i-1}) &= \frac{\mathrm{p}(\boldsymbol{r}_{1:i})}{\sum_{\boldsymbol{r}'_i} \mathrm{p}(\boldsymbol{r}_{1:i-1}, \boldsymbol{r}'_i)} \\
&= \frac{\sum_{\boldsymbol{r}'_{i+1:I}} \mathrm{p}(\boldsymbol{r}_{1:i}, \boldsymbol{r}'_{i+1:I})}{\sum_{\boldsymbol{r}'_{i:I}} \mathrm{p}(\boldsymbol{r}_{1:i-1}, \boldsymbol{r}'_{i:I})} \\
&= \frac{\cancel{const \times \prod_{c \in \mathcal{C} \setminus \mathcal{C}_{i:I}} \phi(\boldsymbol{r}_c)}}{\cancel{const \times \prod_{c \in \mathcal{C} \setminus \mathcal{C}_{i:I}} \phi(\boldsymbol{r}_c)}} \\
&\qquad \times \frac{\prod_{c \in \mathcal{C}_i \setminus \mathcal{C}_{i+1}} \phi(\boldsymbol{r}_c) \sum_{\boldsymbol{r}'_{i+1:I}} \prod_{c \in \mathcal{C}_{i+1:I}} \phi(\boldsymbol{r}'_c)}{\sum_{\boldsymbol{r}'_{i:I}} \prod_{c \in \mathcal{C}_{i:I}} \phi(\boldsymbol{r}'_c)} \\
&= \mathrm{p}(\boldsymbol{r}_i \,|\, \boldsymbol{r}_{i-1}) \quad i = 2, \ldots, I, \,.
\end{aligned}$$

Finally, we show that $[\boldsymbol{r}_i \,|\, \boldsymbol{r}_{-i}]$, $i = 2, \ldots, I-1$ are also Markov chains. First, we rewrite $\mathrm{p}(\boldsymbol{r}_i \,|\, \boldsymbol{r}_{-i})$ to a recursive formulation;

$$\mathrm{p}(\boldsymbol{r}_i \,|\, \boldsymbol{r}_{-i}) = \mathrm{p}(\boldsymbol{r}_{i1} \,|\, \boldsymbol{r}_{-i}) \prod_{x=2}^{X} \mathrm{p}(\boldsymbol{r}_{ix} \,|\, \boldsymbol{r}_{i,1:x-1}, \boldsymbol{r}_{-i}) \,.$$

Now, for $x = 2, \ldots, X$,

$$\begin{aligned}
\mathrm{p}(\boldsymbol{r}_{ix} \,|\, \boldsymbol{r}_{i,1:x-1}, \boldsymbol{r}_{-i}) &= \frac{\mathrm{p}(\boldsymbol{r}_{i,1:x}, \boldsymbol{r}_{-i})}{\sum_{\boldsymbol{r}'_{ix}} \mathrm{p}(\boldsymbol{r}_{i,1:x-1}, \boldsymbol{r}'_{ix}, \boldsymbol{r}_{-i})} \\
&= \frac{\sum'_{\boldsymbol{r}_{i,x+1:X}} \mathrm{p}(\boldsymbol{r}_{i,1:x}, \boldsymbol{r}'_{i,x+1:X}, \boldsymbol{r}_{-i})}{\sum_{\boldsymbol{r}'_{i,x:X}} \mathrm{p}(\boldsymbol{r}_{i,1:x-1}, \boldsymbol{r}'_{i,x:X}, \boldsymbol{r}_{-i})} \\
&= \frac{const \times \prod_{c \in \mathcal{C} \setminus \mathcal{C}_{i,x:X}} \cancel{\phi(\boldsymbol{r}_c)}}{\cancel{const \times \prod_{c \in \mathcal{C} \setminus \mathcal{C}_{i,x:X}} \phi(\boldsymbol{r}_c)}} \\
&\qquad \times \frac{\phi(\boldsymbol{r}_{ix}, \boldsymbol{r}_{i,x-1}) u_{ix}}{\sum_{\boldsymbol{r}'_{ix}} \phi(\boldsymbol{r}'_{ix}, \boldsymbol{r}_{i,x-1}) u_{ix}} \\
\mathrm{p}(\boldsymbol{r}_{i1} \,|\, \boldsymbol{r}_{-i}) &= \frac{u_{i1}}{\sum_{\boldsymbol{r}'_{i1}} u_{i1}} \,,
\end{aligned} \tag{3.6}$$

where, for $x = 1, \ldots, X-1$,

$$\begin{aligned}
u_{ix} &= u_{ix}(\boldsymbol{r}_{ix}, \boldsymbol{r}_{i-1,x:X}, \boldsymbol{r}_{i+1,x:X}) \\
&= \phi(\boldsymbol{r}_{ix}, \boldsymbol{r}_{i-1,x}) \phi(\boldsymbol{r}_{ix}, \boldsymbol{r}_{i+1,x}) \sum_{\boldsymbol{r}'_{i,x+1:X}} \prod_{c \in \mathcal{C}_{i,x+1:X}} \phi(\boldsymbol{r}'_c) \,, \\
u_{iX} &= u_{iX}(\boldsymbol{r}_{iX}, \boldsymbol{r}_{i-1,X}, \boldsymbol{r}_{i+1,X}) \\
&= \phi(\boldsymbol{r}_{iX}, \boldsymbol{r}_{i-1,X}) \phi(\boldsymbol{r}_{ix}, \boldsymbol{r}_{i+1,X}) \,.
\end{aligned} \tag{3.7}$$

We have, for $x = 1, \ldots, X-1$ the recursive relation

$$u_{i,x-1} = \phi(\boldsymbol{r}_{i,x-1}, \boldsymbol{r}_{i-1,x-1}) \phi(\boldsymbol{r}_{i,x-1}, \boldsymbol{r}_{i+1,x-1}) \sum_{\boldsymbol{r}'_{ix}} \phi(\boldsymbol{r}_{i,x-1}, \boldsymbol{r}'_{ix}) u_{ix} \,.$$

From Equation (3.6) and Equation (3.7), we see that

$$\begin{aligned}
\mathrm{p}(\boldsymbol{r}_{ix} \,|\, \boldsymbol{r}_{i,1:x-1}, \boldsymbol{r}_{-i}) &= \mathrm{p}(\boldsymbol{r}_{ix} \,|\, \boldsymbol{r}_{i,x-1}, \boldsymbol{r}_{i-1,x:X}, \boldsymbol{r}_{i+1,x:X}) \,, \\
\mathrm{p}(\boldsymbol{r}_{i1} \,|\, \boldsymbol{r}_{-i}) &= \mathrm{p}(\boldsymbol{r}_{i1} \,|\, \boldsymbol{r}_{i-1}, \boldsymbol{r}_{i+1}) \,,
\end{aligned}$$

which shows that $[\boldsymbol{r}_i \,|\, \boldsymbol{r}_{-i}]$, $i = 2, \ldots, I-1$ are Markov chains. Furthermore, we have shown that

$$\mathrm{p}(\boldsymbol{r}_i \,|\, \boldsymbol{r}_{-i}) = \mathrm{p}(\boldsymbol{r}_i \,|\, \boldsymbol{r}_{i-1}, \boldsymbol{r}_{i+1}) \,.$$

Similarly, $[\boldsymbol{r}_1 \,|\, \boldsymbol{r}_{-1}]$ and $[\boldsymbol{r}_I \,|\, \boldsymbol{r}_{-I}]$ are Markov chains with

$$\begin{aligned}
\mathrm{p}(\boldsymbol{r}_1 \,|\, \boldsymbol{r}_{-1}) &= \mathrm{p}(\boldsymbol{r}_1 \,|\, \boldsymbol{r}_2) \,, \\
\mathrm{p}(\boldsymbol{r}_I \,|\, \boldsymbol{r}_{-I}) &= \mathrm{p}(\boldsymbol{r}_I \,|\, \boldsymbol{r}_{I-1}) \,.
\end{aligned}$$

The prior Markov model combined with our single-site response likelihood form a two-dimensional hidden Markov model. Furthermore, $[\boldsymbol{r}_i \,|\, \boldsymbol{r}_{-i}, \boldsymbol{d}]$, $i = 1, \ldots, I$ can be represented by one-dimensional first-order hidden Markov models. See Figure 3.3 for an illustration of a one-dimensional hidden Markov model corresponding to the case $I = 1$.

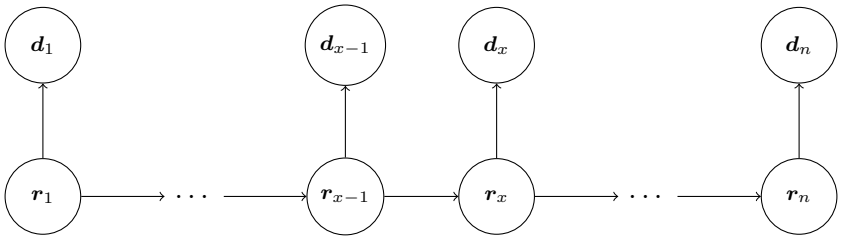


Figure 3.3: Illustration of a hidden Markov model.

3.2 Posterior Model

In this section, we assess the posterior probability $\mathrm{p}(\boldsymbol{r} \,|\, \boldsymbol{d})$. First, we derive some useful results. Then we show how to find the following: The probability of any realization $\boldsymbol{r}^*$; the prediction maximizing the posterior; all marginal probabilities $\mathrm{p}(r_{ixt} \,|\, \boldsymbol{d})$, $(i, x, t) \in \mathcal{L}$; and the expectation $\mathbf{E}[\boldsymbol{r}]$. We also describe how to obtain simulations from the posterior distribution.

Using Equation (3.4) and Equation (3.5), we get the posterior

$$\begin{aligned}\mathrm{p}(\boldsymbol{r} \,|\, \boldsymbol{d}) &= const \times \mathrm{p}(\boldsymbol{d} \,|\, \boldsymbol{r})\mathrm{p}(\boldsymbol{r}) \\ &= const \times \prod_{i=1}^{I} \prod_{x=1}^{X} \mathrm{p}(\boldsymbol{d}_{ix} \,|\, \boldsymbol{r}_{ix}) \prod_{c \in \mathcal{C}} \phi(\boldsymbol{r}_c)\,,\end{aligned}$$

where *const* is the normalizing constant $\sum_{\boldsymbol{r}'} \mathrm{p}(\boldsymbol{d} \,|\, \boldsymbol{r})\mathrm{p}(\boldsymbol{r})$. Thus the posterior is also a Gibbs random field, with a clique system including cliques of single nodes. The clique potential (i, x) is the marginal likelihood $\mathrm{p}(\boldsymbol{d}_{ix} \,|\, \boldsymbol{r}_{ix})$.

We state a couple of results without proof;

$$\begin{aligned}\mathrm{p}(\boldsymbol{r}_{ix} \,|\, \boldsymbol{r}_{-ix}, \boldsymbol{d}) &= \frac{\mathrm{p}(\boldsymbol{d}_{ix} \,|\, \boldsymbol{r}_{ix}) \prod_{c \in \mathcal{C}_{ix}} \phi(\boldsymbol{r}_c)}{\sum_{\boldsymbol{r}'_{ix}} \mathrm{p}(\boldsymbol{d}_{ix} \,|\, \boldsymbol{r}'_{ix}) \prod_{c \in \mathcal{C}_{ix}} \phi(\boldsymbol{r}'_c)} \\ &= \mathrm{p}(\boldsymbol{r}_{ix} \,|\, \boldsymbol{r}_{jy} : (j, y) \in \mathcal{N}_{ix}, \boldsymbol{d}_{ix})\,,\end{aligned}$$

and, for $i = 2, \ldots, I$,

$$\begin{aligned}\mathrm{p}(\boldsymbol{r}_i \,|\, \boldsymbol{r}_{1:i-1}, \boldsymbol{d}) &= \prod_{y=1}^{X} \mathrm{p}(\boldsymbol{d}_{iy} \,|\, \boldsymbol{r}_{iy}) \prod_{c \in \mathcal{C}_i \setminus \mathcal{C}_{i+1}} \phi(\boldsymbol{r}_c) \\ &\quad \times \frac{\sum_{\boldsymbol{r}'_{i+1:I}} \prod_{j=i+1}^{I} \prod_{y=1}^{X} \mathrm{p}(\boldsymbol{d}_{jy} \,|\, \boldsymbol{r}'_{jy}) \prod_{c \in \mathcal{C}_{i+1:I}} \phi(\boldsymbol{r}'_c)}{\sum_{\boldsymbol{r}'_{i:I}} \prod_{j=i}^{I} \prod_{y=1}^{X} \mathrm{p}(\boldsymbol{d}_{jy} \,|\, \boldsymbol{r}'_{jy}) \prod_{c \in \mathcal{C}_{i:I}} \phi(\boldsymbol{r}'_c)} \\ &= \mathrm{p}(\boldsymbol{r}_i \,|\, \boldsymbol{r}_{i-1}, \boldsymbol{d}_{i:I})\,.\end{aligned}$$

The proofs are similar to the proofs of the analogous statements in Section 3.1.

As in Section 3.1, we rewrite the row-wise full-conditionals in a recursive form. For $i = 2, \ldots, I-1$,

$$\mathrm{p}(\boldsymbol{r}_i \,|\, \boldsymbol{r}_{-i}, \boldsymbol{d}) = \mathrm{p}(\boldsymbol{r}_{i1} \,|\, \boldsymbol{r}_{-i}, \boldsymbol{d}) \prod_{x=2}^{X} \mathrm{p}(\boldsymbol{r}_{ix} \,|\, \boldsymbol{r}_{i,1:x-1}, \boldsymbol{r}_{-i}, \boldsymbol{d})\,. \tag{3.8}$$

We have

$$\begin{aligned}
\mathrm{p}(\boldsymbol{r}_{ix} \,|\, \boldsymbol{r}_{i,1:x-1}, \boldsymbol{r}_{-i}, \boldsymbol{d}) &= \frac{\mathrm{p}(\boldsymbol{d} \,|\, \boldsymbol{r}_{i,1:x}, \boldsymbol{r}_{-i})\mathrm{p}(\boldsymbol{r}_{i,1:x}, \boldsymbol{r}_{-i})}{\sum_{\boldsymbol{r}'_{ix}} \mathrm{p}(\boldsymbol{d} \,|\, \boldsymbol{r}'_{ix}\boldsymbol{r}_{i,1:x-1}, \boldsymbol{r}_{-i})\mathrm{p}(\boldsymbol{r}'_{ix}\boldsymbol{r}_{i,1:x-1}, \boldsymbol{r}_{-i})} \\
&= \frac{\phi(\boldsymbol{r}_{ix}, \boldsymbol{r}_{i,x-1})v_{ix}}{\sum_{\boldsymbol{r}'_{ix}} \phi(\boldsymbol{r}'_{ix}, \boldsymbol{r}_{i,x-1})v_{ix}}\,, \quad x = 2, \ldots, X\,, \qquad (3.9) \\
\mathrm{p}(\boldsymbol{r}_{i1} \,|\, \boldsymbol{r}_{-i}, \boldsymbol{d}) &= \frac{v_{i1}}{\sum_{\boldsymbol{r}'_{i1}} v_{i1}}\,,
\end{aligned}$$

where, for $x = 1, \ldots, X-1$,

$$\begin{aligned}
v_{ix} &= v_{ix}(\boldsymbol{r}_{ix}, \boldsymbol{r}_{i-1,x:X}, \boldsymbol{r}_{i+1,x:X}, \boldsymbol{d}_{i,x:X}) \\
&= v_{ix}(\boldsymbol{r}_{ix}, \boldsymbol{d}'_{i,x:X}) \\
&= \mathrm{p}(\boldsymbol{d}_{ix} \,|\, \boldsymbol{r}_{ix})\phi(\boldsymbol{r}_{ix}, \boldsymbol{r}_{i-1,x})\phi(\boldsymbol{r}_{ix}, \boldsymbol{r}_{i+1,x}) \\
&\qquad \times \sum_{\boldsymbol{r}'_{i,x+1:X}} \prod_{y=x+1}^{X} \mathrm{p}(\boldsymbol{d}_{iy} \,|\, \boldsymbol{r}'_{iy}) \prod_{c \in \mathcal{C}_{i,x+1:X}} \phi(\boldsymbol{r}'_c)\,, \qquad (3.10) \\
v_{iX} &= v_{iX}(\boldsymbol{r}_{iX}, \boldsymbol{r}_{i-1,X}, \boldsymbol{r}_{i+1,X}, \boldsymbol{d}_{i,X}) \\
&= v_{iX}(\boldsymbol{r}_{iX}, \boldsymbol{d}'_{iX}) \\
&= \mathrm{p}(\boldsymbol{d}_{iX} \,|\, \boldsymbol{r}_{iX})\phi(\boldsymbol{r}_{iX}, \boldsymbol{r}_{i-1,X})\phi(\boldsymbol{r}_{ix}, \boldsymbol{r}_{i+1,X})\,.
\end{aligned}$$

The simplifying notation

$$\begin{aligned}
\boldsymbol{d}'_{i,x:X} &= (\boldsymbol{r}_{i-1,x:X}, \boldsymbol{r}_{i+1,x:X}, \boldsymbol{d}_{i,x:X}), \\
\boldsymbol{d}'_{i,X} &= (\boldsymbol{r}_{i-1,X}, \boldsymbol{r}_{i+1,X}, \boldsymbol{d}_{i,X})
\end{aligned}$$

is introduced in Equation (3.10), and is used going forward.

As with the analogous result for the prior, we can express $v_{ix}(\cdot)$, $x = 1, \ldots, X-1$, recursively;

$$\begin{aligned}
v_{i,x-1} = \mathrm{p}(\boldsymbol{d}_{i,x-1} \,|\, \boldsymbol{r}_{i,x-1})&\phi(\boldsymbol{r}_{i,x-1}, \boldsymbol{r}_{i-1,x-1})\phi(\boldsymbol{r}_{i,x-1}, \boldsymbol{r}_{i+1,x-1}) \\
&\times \sum_{\boldsymbol{r}'_{ix}} \phi(\boldsymbol{r}_{i,x-1}, \boldsymbol{r}'_{ix}) v_{ix}(\boldsymbol{r}'_{ix}, \boldsymbol{r}_{i-1,x:X}, \boldsymbol{r}_{i+1,x:X})\,.
\end{aligned}$$

Furthermore, we see that

$$\begin{aligned}
\mathrm{p}(\boldsymbol{r}_{ix} \,|\, \boldsymbol{r}_{i,1:x-1}, \boldsymbol{r}_{-i}, \boldsymbol{d}) &= \mathrm{p}(\boldsymbol{r}_{ix} \,|\, \boldsymbol{r}_{i,x-1}, \boldsymbol{d}'_{i,x:X}) \\
\mathrm{p}(\boldsymbol{r}_{i1} \,|\, \boldsymbol{r}_{-i}, \boldsymbol{d}) &= \mathrm{p}(\boldsymbol{r}_{i1} \,|\, \boldsymbol{d}'_i)\,,
\end{aligned}$$

where $\boldsymbol{d}'_i = (\boldsymbol{r}_{i-1}, \boldsymbol{r}_{i+1}, \boldsymbol{d}_i)$, and so

$$\mathrm{p}(\boldsymbol{r}_i \,|\, \boldsymbol{r}_{-i}, \boldsymbol{d}) = \mathrm{p}(\boldsymbol{r}_i \,|\, \boldsymbol{d}'_i)\,.$$

Corresponding results hold for $i = 1$ and $i = I$.

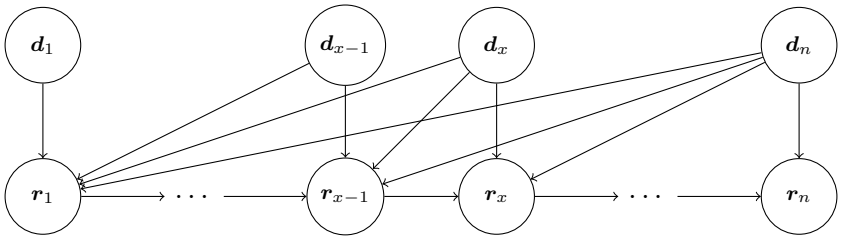


Figure 3.4: Posterior Markov chain. Seismic data $\boldsymbol{d}$, reflection coefficients $\boldsymbol{r}$.

The row-wise full-conditionals $\mathrm{p}(\boldsymbol{r}_i \,|\, \boldsymbol{d}'_i)$ can be represented by a Markov chain with transition probabilities $\mathrm{p}(\boldsymbol{r}_{ix} \,|\, \boldsymbol{r}_{i,x-1}, \boldsymbol{d}'_{i,x:X})$. The transition probabilities are, in general, not constant with respect to x, i.e. the Markov chains are non-stationary. The Markov chain corresponding to the two-dimensional case $I = 1$ is illustrated in Figure 3.4.

We find the full-conditional posterior probabilities for an inline $\mathrm{p}(\boldsymbol{r}_i \,|\, \boldsymbol{d}'_i)$ for any $\boldsymbol{r}_i$, $\boldsymbol{r}_{ix} \in \Omega_s$, $x = 1, \ldots, X$ by inserting the transition probabilities into Equation (3.8). The (full-conditional) maximum a posteriori (MAP) prediction

$$\hat{\boldsymbol{r}}_{i,MAP} = \mathrm{argmax}_{\boldsymbol{r}_i} \mathrm{p}(\boldsymbol{r}_i \,|\, \boldsymbol{d}'_i)\,, \tag{3.11}$$

can be found using the Viterbi algorithm, which we describe in detail in Chapter 4. The MAP prediction is the most likely sequence of reflection coefficients given all the data and the assumptions that are made in the introduction of this chapter and in Section 3.1.

The posterior marginal probabilities for an inline conditioned on the neighboring inlines $\mathrm{p}(\boldsymbol{r}_{ix} \,|\, \boldsymbol{d}'_i)$, $i = 2, \ldots, I-1$, $x = 2, \ldots, n$, $s_x \in \Omega_s$, can be found recursively;

$$\begin{aligned} \mathrm{p}(\boldsymbol{r}_{ix} \,|\, \boldsymbol{d}'_i) &= \sum_{\boldsymbol{r}'_{i,x-1}} \mathrm{p}(\boldsymbol{r}'_{i,x-1}, \boldsymbol{r}_{ix} \,|\, \boldsymbol{d}'_i) \\ &= \sum_{\boldsymbol{r}'_{i,x-1}} \mathrm{p}(\boldsymbol{r}_{ix} \,|\, \boldsymbol{r}'_{i,x-1}, \boldsymbol{d}'_i) \mathrm{p}(\boldsymbol{r}'_{i,x-1} \,|\, \boldsymbol{d}'_i)\,. \end{aligned} \tag{3.12}$$

The distribution $\mathrm{p}(\boldsymbol{r}_{i1} \,|\, \boldsymbol{d}'_i)$ is given in Equation (3.9). Corresponding results hold for $i = 1$ and $i = I$. The posterior marginals indicate the uncertainty of predictions. If the marginal is high in other locations in a column than the MAP location, then the certainty of that specific MAP location is, all other things equal, lower.

From the marginals, we can find the marginal maximum a posteriori (MMAP) prediction

$$\hat{\boldsymbol{r}}_{i,MMAP} = (\mathrm{argmax}_{\boldsymbol{r}_{i1}} \mathrm{p}(\boldsymbol{r}_{i1} \,|\, \boldsymbol{d}'_i),\, \ldots,\, \mathrm{argmax}_{\boldsymbol{r}_{iX}} \mathrm{p}(\boldsymbol{r}_{iX} \,|\, \boldsymbol{d}'_i))^{\mathsf{T}}\,.$$

The marginals also give the expectation

$$\begin{aligned}\mathbf{E}[\boldsymbol{r}_i \mid \boldsymbol{d}'_i] &= \big(\mathbf{E}[\boldsymbol{r}_{i1} \mid \boldsymbol{d}'_i],\, \ldots,\, \mathbf{E}[s_X \mid \boldsymbol{d}'_i]\big)^\intercal \\ &= \left(\sum_{\boldsymbol{r}'_{i1}} \mathrm{p}(\boldsymbol{r}'_{i1} \mid \boldsymbol{d}'_i)\boldsymbol{r}'_{i1},\, \ldots,\, \sum_{\boldsymbol{r}'_{iX}} \mathrm{p}(\boldsymbol{r}'_{iX} \mid \boldsymbol{d}'_i)\boldsymbol{r}'_{iX}\right)^\intercal .\end{aligned}$$

The MMAP prediction and the expectation vector can be used as predictors. However, the MAP predictor, by definition, maximizes the posterior probability. Thus it is the preferred predictor if it can be found efficiently.

Finally, we can obtain simulations from the posterior $\mathrm{p}(\boldsymbol{r}_i \mid \boldsymbol{d}'_i)$. This is done by first drawing a sample from $\mathrm{p}(\boldsymbol{r}_{i1} \mid \boldsymbol{d}'_i)$, then iteratively drawing $\boldsymbol{r}_{ix}$, $x = 2, \ldots, n$ from $\mathrm{p}(\boldsymbol{r}_{ix} \mid \boldsymbol{r}_{i,x-1}, \boldsymbol{d}'_i)$ using the previously drawn vector $\boldsymbol{r}_{i,x-1}$. Simulations can be used obtain different realizations that can be used as predictors.

Chapter 4

Autotracking Algorithms

In this chapter, we develop two different horizon autotracking algorithms based on the hidden Markov model developed in Chapter 3. Recall that an autotracking algorithm predicts a horizon in a seismic volume based on seismic data and one or more seed points. The first of the two algorithms we develop finds approximations of MAP predictions for one inline at a time; see Section 4.1. The second is a greedy autotracker with lower computational complexity; see Section 4.2. The greedy autotracker also has the advantage of simultaneously operating in the entire three-dimensional seismic volume.

Both autotracking algorithms are designed to be interactive. A human interpreter choses one or more seed points to mark the horizon in question and to aid the tracking algorithm. Furthermore, we estimate the horizon depths $\boldsymbol{s}$ and the reflection coefficients $\boldsymbol{q}$ sequentially, as described in detail in the following sections.

For a tracking algorithm to outperform another, it should either require fewer seed points to track a horizon with sufficient precision or run faster. In this thesis, we mostly focus on precision.

4.1 Viterbi Autotracker

In this section, we develop an autotracking algorithm based on the fact that we can find the full-conditional maximum a posteriori (MAP) prediction for a horizon in an inline. The MAP prediction is defined in Equation (3.11). We find it using the Viterbi algorithm (Viterbi, 1967), which we describe later. For now, assume we know how to find the MAP prediction. The algorithm estimates the MAP prediction one inline at a time.

When tracking a horizon, the interpreter starts by selecting one or more seed points in an inline, say inline $i-1$. When selecting a seed point, the interpreter adds information. We incorporate this information into the likelihood. Let u_{ix} be a seed point given at column (i,x). We assume that the horizon passes through the seed point, i.e. that $u_{ix} = s_{ix}$. The data at (i,x) is now $\{\boldsymbol{d}_{ix}, u_{ix}\}$. Recall that $\boldsymbol{r}_{ix} = q_{ix}\boldsymbol{i}(s_{ix})$, q_{ix} being the reflection coefficient

and s_{ix} the depth of the horizon at (i, x). We define the likelihood as follows;

$$\mathrm{p}(\boldsymbol{d}_{ix}, u_{ix} \,|\, \boldsymbol{r}_{ix}) = \mathrm{p}(u_{ix} \,|\, s_{ix}) = \begin{cases} 1, & u_{ix} = s_{ix} \\ 0, & \text{otherwise} \end{cases} .$$

Hence, any horizon not containing all seed points has probability 0. This ensures that the autotracked horizon covers all seed points.

To find $\hat{\boldsymbol{r}}_{i-1,MAP}$ as defined in Equation (3.11), we would need to know $\boldsymbol{r}_{i-2}$ and $\boldsymbol{r}_i$ to find the exact transition probabilities. The issue is that $\boldsymbol{r}_{i-2}$ and $\boldsymbol{r}_i$ are unknown, and there is no information except the seed point(s) in inline $i-1$ which could lead to initial guesses. Therefore, for the initial inline $i-1$, we use the analog of Equation (3.9) corresponding to a two-dimensional seismic volume to estimate the transition probabilities. The interpreter, if necessary, adds seed points until the estimate $\hat{\boldsymbol{r}}_{i-1}$ is sufficiently accurate.

Next, we wish to find an estimate $\hat{\boldsymbol{r}}_i$ for $\boldsymbol{r}_i$ (and $\hat{\boldsymbol{r}}_{i-2}$) by replacing $\boldsymbol{r}_{i-1}$ in $\boldsymbol{d}'_i = (\boldsymbol{r}_{i-1}, \boldsymbol{r}_{i+1}, \boldsymbol{d}_i)$ with the estimate $\hat{\boldsymbol{r}}_{i-1}$ in Equation (3.11). However, we still need an estimate for $\boldsymbol{r}_{i+1}$. A reasonable initial guess would be $\tilde{\boldsymbol{r}}_{i+1} = \hat{\boldsymbol{r}}_{i-1}$. From Equation (3.9) we know that three factors in our estimate of the posterior transition probabilities $\mathrm{p}(\boldsymbol{r}_{ix} \,|\, \boldsymbol{r}_{i,x-1}, \boldsymbol{d}'_{i,x:X})$ are $\phi(\boldsymbol{r}_{ix}, \boldsymbol{r}_{i,x-1})$, $\phi(\boldsymbol{r}_{ix}, \hat{\boldsymbol{r}}_{i-1,x})$ and $\phi(\boldsymbol{r}_{ix}, \tilde{\boldsymbol{r}}_{i+1,x})$. However, since $\tilde{\boldsymbol{r}}_{i+1} = \hat{\boldsymbol{r}}_{i-1}$, we have $\phi(\boldsymbol{r}_{ix}, \hat{\boldsymbol{r}}_{i-1,x}) = \phi(\boldsymbol{r}_{ix}, \tilde{\boldsymbol{r}}_{i+1,x})$. Thus the weight of the contribution of $\hat{\boldsymbol{r}}_{i-1,x}$ is squared compared to the contribution of $\boldsymbol{r}_{i,x-1}$, which is undesirable. However, we are free to define the clique potentials suitably. In effect, we define them such that we ignore $\phi(\boldsymbol{r}_{ix}, \tilde{\boldsymbol{r}}_{i+1,x})$.

In the two following subsections, we develop different parts of the algorithm. First, in Section 4.1.1, we develop an algorithm to find the posterior transition matrices for our hidden Markov model. In Section 4.1.2, we describe an algorithm to find the MAP prediction based on the posterior transition matrices. In Section 4.1.3, we put it all together.

4.1.1 Posterior Transition Matrices — Reverse Algorithm

In this subsection, we use an algorithm based on Equation (3.9) to sequentially find the posterior transition matrices $\mathrm{p}(\boldsymbol{r}_{ix} \,|\, \boldsymbol{r}_{i,x-1}, \boldsymbol{d}'_{i,x:X}), x = 2, \ldots, n$. Algorithm 1, the Reverse Algorithm, implements this procedure, with a subtle change due to the handling of small and large numbers in computers. We explain this change at the end of this subsection.

In the following paragraphs we discuss the complexity of Algorithm 1. The complexity is important due to the large size of seismic datasets. Recall that $\boldsymbol{r}_{ix} = q_{ix}\boldsymbol{i}(s_{ix})$, where $q_{ix} \in \Omega_q$. There are $T|\Omega_q|$ possible values for $\boldsymbol{r}_{ix}$ for each $x \in \{1, \ldots, X\}$, where $|\Omega_q|$ is the cardinality of Ω_q. Thus there are $(X-1)(T|\Omega_q|)^2$ transition probabilities, as well as $T|\Omega_q|$ probabilities for the possible values of $\boldsymbol{r}_{i1}$. The complexity of the algorithm itself is $O((X-1)(T|\Omega_q|)^2 + T|\Omega_q|) = O(X(T|\Omega_q|)^2)$. The space complexity is also $O(X(T|\Omega_q|)^2)$, since all transition probabilities are stored.

Algorithm 1: Reverse Algorithm

input : Data $\boldsymbol{d}'_i$ with seismic data matrix $\boldsymbol{d}_i \subset \boldsymbol{d}'_i$ of size $(X \times T)$; likelihood $\mathrm{p}(\boldsymbol{d}_{ix} \,|\, \boldsymbol{r}_{ix})$, $\boldsymbol{r}_{ix} \in \Omega_r$, $x = 1, \ldots, X$; and clique potentials $\phi(\cdot, \cdot)$

output: Posterior marginal probability $\mathrm{p}(\boldsymbol{r}_{i1} \,|\, \boldsymbol{d}'_i)$, posterior transition matrices $\mathrm{p}(\boldsymbol{r}_{ix} \,|\, \boldsymbol{r}_{i,x-1}, \boldsymbol{d}'_{i,x:X})$, $\boldsymbol{r}_{ix}, \boldsymbol{r}_{i,x-1} \in \Omega_r$, $x = 2, \ldots, X$

forall $\boldsymbol{r}_{iX} \in \Omega_r$ **do**
  $v_{iX}(\boldsymbol{r}_{iX}, \boldsymbol{d}'_{iX}) = \mathrm{p}(\boldsymbol{d}_{iX} \,|\, \boldsymbol{r}_{iX}) \phi(\boldsymbol{r}_{iX}, \boldsymbol{r}_{i-1,X})$
end
$q_X = \sum_{\boldsymbol{r}_{iX}} v_{iX}(\boldsymbol{r}_{iX}, \boldsymbol{d}'_{iX}) / |\Omega_r|$
forall $\boldsymbol{r}_{iX} \in \Omega_r$ **do**
  $v^*_{iX}(\boldsymbol{r}_{iX}, \boldsymbol{d}'_{iX}) = v_{iX}(\boldsymbol{r}_{iX}, \boldsymbol{d}'_{iX}) / q_{iX}$
end
for $x \leftarrow X$ **to** 2 **do**
  forall $\boldsymbol{r}_{ix} \in \Omega_r$ **do**
    forall $\boldsymbol{r}_{i,x-1} \in \Omega_r$ **do**
      $\mathrm{p}(\boldsymbol{r}_{ix} \,|\, \boldsymbol{r}_{i,x-1}, \boldsymbol{d}'_{i,x:X}) = \frac{\phi(\boldsymbol{r}_{ix}, \boldsymbol{r}_{i,x-1}) v^*_{ix}(\boldsymbol{r}_{ix}, \boldsymbol{d}'_{i,x:X})}{\sum_{\boldsymbol{r}'_{ix}} \phi(\boldsymbol{r}'_{ix}, \boldsymbol{r}_{i,x-1}) v^*_{ix}(\boldsymbol{r}'_{ix}, \boldsymbol{d}'_{i,x:X})}$
      $v_{i,x-1} = \mathrm{p}(\boldsymbol{d}_{i,x-1} \,|\, \boldsymbol{r}_{i,x-1}) \phi(\boldsymbol{r}_{i,x-1}, \boldsymbol{r}_{i-1,x-1}) \times \sum_{\boldsymbol{r}'_{ix}} \phi(\boldsymbol{r}_{i,x-1}, \boldsymbol{r}'_{ix}) v_{ix}(\boldsymbol{r}'_{ix}, \boldsymbol{r}_{i-1,x:X}, \boldsymbol{r}_{i+1,x:X})$
    end
    $q'_{x-1} = \sum_{\boldsymbol{r}_{i,x-1}} v'_{i,x-1}(\boldsymbol{r}_{i,x-1}, \boldsymbol{d}'_{i,x-1}) / |\Omega_r|$
    forall $\boldsymbol{r}_{i,x-1} \in \Omega_r$ **do**
      $v^*_{i,x-1}(\boldsymbol{r}_{i,x-1}, \boldsymbol{d}'_{i,x-1:X}) = v'_{i,x-1}(\boldsymbol{r}_{i,x-1}, \boldsymbol{d}'_{i,x:X}) / q'_{x-1}$
    end
  end
end
forall $\boldsymbol{r}_{i1} \in \Omega_r$ **do**
  $\mathrm{p}(\boldsymbol{r}_{i1} \,|\, \boldsymbol{d}'_i) = \frac{v^*_{i1}(\boldsymbol{r}_{i1}, \boldsymbol{d})}{\sum_{\boldsymbol{r}'_{i1}} v^*_{i1}(\boldsymbol{r}'_{i1}, \boldsymbol{d})}$
end

To calculate $\mathrm{p}(\boldsymbol{r}_{ix} \,|\, \boldsymbol{r}_{i,x-1}, \boldsymbol{d}'_{i,x:X}), x = 2, \ldots, n$, for each $\boldsymbol{r}_{ix}, \boldsymbol{r}_{i,x-1}$ one needs $v_x(\boldsymbol{r}_{ix}, \boldsymbol{d}'_{i,x:X})$. Calculating $v_x(\boldsymbol{r}_{ix}, \boldsymbol{d}'_{i,x:X})$ involves the calculation of $\mathrm{p}(\boldsymbol{d}_{ix} \,|\, \boldsymbol{r}_{ix})$; recall Equation (3.10). Since $[\boldsymbol{d}_{ix} \,|\, \boldsymbol{r}_{ix}]$ is multivariate normal, and the multivariate normal pdf involves finding the determinant and the inverse of the covariance matrix $\boldsymbol{\Sigma} = \sigma_\delta^2 \boldsymbol{W}\boldsymbol{W}^\intercal + \sigma_\epsilon^2 \boldsymbol{I}$, the time complexity generally is $O(T^{2.373})$ (Williams, 2012). However, since $\boldsymbol{\Sigma}$ is constant with respect to i and x, the inverse and determinant only need to be calculated once. The time complexity of computing the density with the inverse and determinant of $\boldsymbol{\Sigma}$ known is $O(T^2)$ nonetheless since it involves multiplication of a vector with a matrix. So the total time complexity of Algorithm 1, including internal calculations, is $O(X(T|\Omega_q|)^2 T^2) = O(X|\Omega_q|^2 T^4)$.

Since T can be large, the dependency on T^4 is problematic. However, as horizons are usually relatively flat, we do not have to consider the entire depth of a seismic volume when tracking each horizon. Instead, we consider a horizontal cross section of depth m, in effect letting $|\Omega_s| = m$. The value m can be changed from inline to inline, but for simplicity, we keep it constant in this study. While T can be larger than 1000, a much smaller m will mostly be sufficient; we use $m \approx 100 \approx T^{2/3}$. Similarly, we can consider values of q_{ix} in some range only. Letting $|\Omega_q| \approx m$, the complexity of Algorithm 1 is approximately $O(Xm^6) \approx O(XT^4)$.

Although m is relatively small the m^6 dependency is still an issue, as indicated by the approximate T^4 dependency. We can further reduce the complexity by estimating the horizon depths $\boldsymbol{s}_i$ and reflection coefficients $\boldsymbol{q}_i$ separately. First, assume that $\boldsymbol{q}_i$ is known. Then we can find $\mathrm{p}(s_{ix} \,|\, s_{i,x-1}, \boldsymbol{q}_{i,x:X}, \boldsymbol{d}'_{i,x:X}) = \mathrm{p}(s_{ix} \,|\, s_{i,x-1}, \boldsymbol{q}_{i,x-1:x}, \boldsymbol{d}'_{i,x:X}), x = 2, \ldots, X$ analogously to how we find $\mathrm{p}(\boldsymbol{r}_{ix} \,|\, \boldsymbol{r}_{i,x-1}, \boldsymbol{d}'_{i,x:X})$. Specifically, we replace the sums over $\boldsymbol{r}'_{ix}$ and $\boldsymbol{r}'_{i,x+1:X}$ in Equation (3.9) and Equation (3.10) with sums over s'_{ix} and $\boldsymbol{s}'_{i,x+1:X}$, respectively. Similarly, we can find $\mathrm{p}(q_{ix} \,|\, q_{i,x-1}, \boldsymbol{s}_{i,x-1:x}, \boldsymbol{d}'_{i,x:X})$. To utilize these facts, we would need an initial guess for either $\boldsymbol{s}_i$ or $\boldsymbol{q}_i$. Since small errors in s_{ix} might lead to large amplitude errors, and thus to difficulties in estimating q_{ix}, it makes sense to start with an initial guess for $\boldsymbol{q}_i$. Furthermore, this way the interpreter can focus solely on $\boldsymbol{s}$. And the depth of the horizon $\boldsymbol{s}$ is usually the unknown of interest.

By first estimating $\boldsymbol{s}_i$ and then using the estimate of $\boldsymbol{s}_i$ to estimate $\boldsymbol{q}_i$, we have reduced the complexity of estimating $\boldsymbol{r}_i$ to $O(Xm^4) + O(X(m|\Omega_q|)^2) \approx O(Xm^4) \approx O(XT^{8/3})$. This procedure can, of course, be run iteratively. However, running an average of k iterations at each inline increases time complexity by a factor of k. Furthermore, assuming the interpreter ensures a precise estimate of $\boldsymbol{s}_i$ is found and that this leads to a good approximation of $\boldsymbol{q}_i$, we should be able to obtain good results with just one run for each inline. To keep the number of parameters low, we let $k = 1$ in this thesis.

One final step is taken to reduce the complexity even further. We consider only the data at $n < 10$ depths around a point when calculating the likelihood instead of at all m depths, reducing the complexity to $O(Xm^2n^2) \approx$

$O(Xm^3) \approx O(XT^2)$. The effects of using this method, with $n = 7$, are illustrated in Figure 4.1. The likelihoods are calculated using a true reflection coefficient from the middle horizon. Notice how the two top horizons, which have negative reflection coefficients, stand more out from the area around them when using the local likelihood method. This is due to the large spikes at the horizons appearing at the wrong index when calculating likelihoods for depths close to the true depths. For depths further away from the horizons, these spikes are not included. When considering entire traces, the spikes are always included, and as a result the background is more uniform.

The second thing to notice from Figure 4.1, is that using the full likelihoods method generally yields higher likelihoods at the top horizon than the middle horizon, even though the reflection coefficient is from the middle one. This is due to the large negative seismic data from the top horizon appearing at an unexpected location when assuming the correct location for the middle horizon. Further experiments, which we do not discuss in this report, confirm that the local method is not only notably faster; it tends to yield better results.

Applying Algorithm 1 for horizon depths $\boldsymbol{s}_i$ on the synthetic seismic illustrated in Figure 3.1, we find the posterior transition matrix $\mathrm{p}(s_{i,10} \mid s_{i,9}, \boldsymbol{q}_i, \boldsymbol{d}_i)$ shown in Figure 4.2. Here the estimate of the reflection coefficient vector $\boldsymbol{q}_i$ is just a constant, and i is the index of the illustrated inline. Even though there is a fault between $x = 9$ and $x = 10$, the most likely transition from any depth at $x = 9$ is to depth 11 at $x = 10$, close to the true depth 12.

Algorithm technicalities

Notice that if the marginal likelihoods $\mathrm{p}(\boldsymbol{d}_{ix} \mid \boldsymbol{r}_{ix})$ are mostly small or mostly large, $v_x(\cdot, \cdot)$ will become too small or large, respectively, to represent numerically as Algorithm 1 iterates towards smaller x. In practice, we solve this issue by letting

$$v^*_{ix}(\boldsymbol{r}_{ix}, \boldsymbol{d}'_{i,x:X}) = v_{ix}(\boldsymbol{r}_{ix}, \boldsymbol{d}'_{i,x:X})/q_{ix}\,, \quad x = 1, \ldots, X\,,$$

where $q_{ix} = \sum_{\boldsymbol{r}'_{ix}} v_{ix}(\boldsymbol{r}'_{ix}, \boldsymbol{d}'_{i,x:X})/|\Omega_r|$. Then Equation (3.9) can be written

$$\mathrm{p}(\boldsymbol{r}_{ix} \mid \boldsymbol{r}_{i,x-1}, \boldsymbol{d}'_{i,x:X}) = \frac{\phi(\boldsymbol{r}_{ix}, \boldsymbol{r}_{i,x-1}) v^*_{ix}(\boldsymbol{r}_{ix}, \boldsymbol{d}'_{i,x:X})}{\sum_{\boldsymbol{r}'_{ix}} \phi(\boldsymbol{r}'_{ix}, \boldsymbol{r}_{i,x-1}) v^*_{ix}(\boldsymbol{r}'_{ix}, \boldsymbol{d}'_{i,x:X})}\,,$$
$$\mathrm{p}(\boldsymbol{r}_{i1} \mid \boldsymbol{d}'_i) = \frac{v^*_{i1}(\boldsymbol{r}_{i1}, \boldsymbol{d})}{\sum_{\boldsymbol{r}'_{i1}} v^*_{i1}(\boldsymbol{r}'_{i1}, \boldsymbol{d})}\,.$$

When considering $\boldsymbol{s}_i$ and $\boldsymbol{q}_i$, an analogous adaption is made.

4.1.2 Cross section MAP Prediction — Viterbi Algorithm

We have not yet discussed how to find $\hat{\boldsymbol{r}}_{i,MAP}$, $i = 1, \ldots, I$, defined in Equation (3.11). As mentioned earlier in this section, we do not actually find true

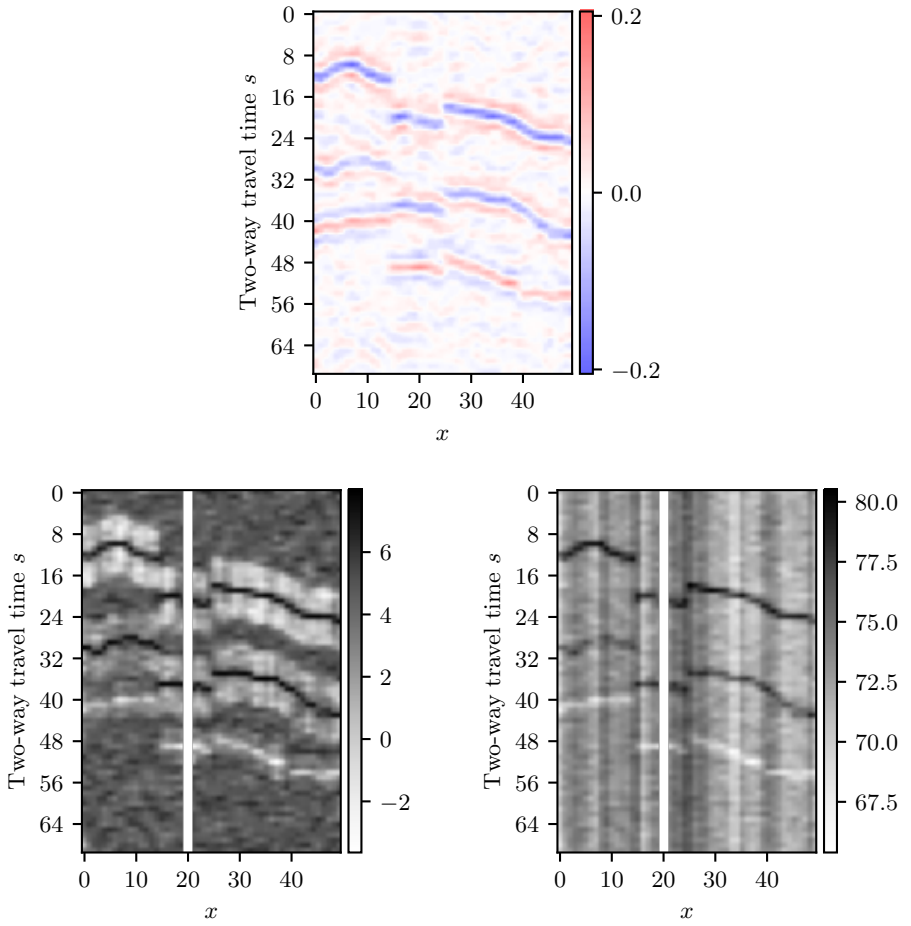


Figure 4.1: Top figure: Synthetic seismic data. Bottom left: Likelihoods for each data point when considering three points above and below the points. Bottom right: Likelihoods when considering the entire trace for each point. The white column in the likelihood figures is due to a seed point in that column.

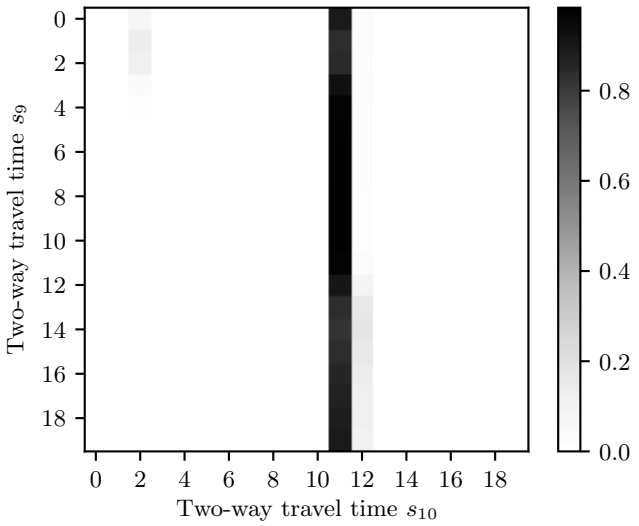


Figure 4.2: Posterior transition matrix $\mathrm{p}(s_{13} \mid s_{12}, \boldsymbol{d}_{13:n}, q)$ for the synthetic seismic data in Figure 3.1.

MAP predictions, since we use initial guesses and approximations as inputs. Therefore, and for simplicity, we denote our estimates $\hat{\boldsymbol{r}}_i$. Also, we estimate $\boldsymbol{s}_i$ and $\boldsymbol{q}_i$ sequentially — which we come back to later. To find $\hat{\boldsymbol{r}}_i$, we use the Viterbi algorithm (Viterbi, 1967); see Algorithm 2. The Viterbi algorithm is a dynamic programming algorithm for hidden Markov models; we break the problem down to sub-problems and solve them optimally in a recursive manner. Since the problem possesses the optimal substructure property, this yields the global optimum.

The statement about optimal substructure can be proven inductively. We present an informal proof. The marginal probability of $\boldsymbol{r}_{i1}$, $\mathrm{p}(\boldsymbol{r}_{i1} \mid \boldsymbol{d}'_i)$, is given by Equation (3.9). Equation (3.9) is also used directly to find $\operatorname{argmax}_{\boldsymbol{r}'_{i1}} \mathrm{p}(\boldsymbol{r}_{i2} \mid \boldsymbol{r}'_{i1}, \boldsymbol{d}'_i)$ for all possible values of $\boldsymbol{r}_2$. The most likely path to $\boldsymbol{r}_3$, i.e. the most likely sequence $\{\boldsymbol{r}_1, \boldsymbol{r}_2, \boldsymbol{r}_3\}$, has to go through the most likely path to $\boldsymbol{r}_2$, since

$$\mathrm{p}(\boldsymbol{r}_3, \boldsymbol{r}_2, \boldsymbol{r}_1 \mid \boldsymbol{d}) = \mathrm{p}(\boldsymbol{r}_3 \mid \boldsymbol{r}_2, \boldsymbol{d}_{3:n}) \mathrm{p}(\boldsymbol{r}_2, \boldsymbol{r}_1, \mid \boldsymbol{d}) \,.$$

That is, the most likely path to $\boldsymbol{r}_3$ is the path maximizing the probability of the sub-path to $\boldsymbol{r}_2$ multiplied by the probability of going from $\boldsymbol{r}_2$ to $\boldsymbol{r}_3$. So we do not have to consider suboptimal paths to $\boldsymbol{r}_2$. An analogous argument holds for $\boldsymbol{r}_x$, $x = 4, \ldots, n$.

As discussed before, we define $\boldsymbol{r}_i$ through $\boldsymbol{s}_i$ and $\boldsymbol{q}_i$ and estimate them separately. For the MAP prediction of $\boldsymbol{s}_i$, the algorithm needs to store the optimal path to each of the m possible values of s_{ix}, $x = 1, \ldots, X$, along with the probabilities of those paths. The paths can be stored in an $X \times m$ matrix,

Algorithm 2: Viterbi Algorithm

input : Posterior marginal probability $\mathrm{p}(\boldsymbol{r}_{i1} \,|\, \boldsymbol{d}'_i)$; posterior transition matrices $\mathrm{p}(\boldsymbol{r}_{ix} \,|\, \boldsymbol{r}_{i,x-1}, \boldsymbol{d}'_{x:X}),\ \boldsymbol{r}_{ix}, \boldsymbol{r}_{i,x-1} \in \Omega_r,\ x = 2, \ldots, X$

output: Estimated MAP horizon prediction $\hat{\boldsymbol{r}}_i$

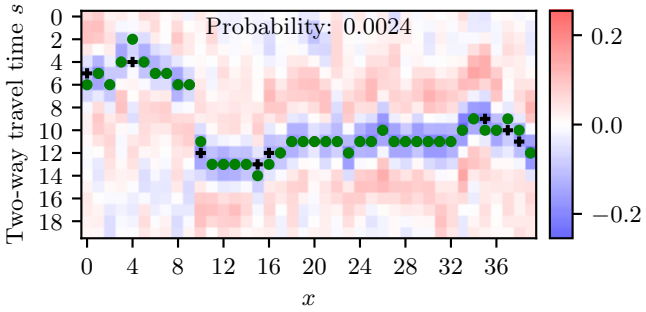
forall $\boldsymbol{r}_{i2} \in \Omega_r$ **do**
  $\max_{\boldsymbol{r}_{i1}} \mathrm{p}(\boldsymbol{r}_{i,1:2} \,|\, \boldsymbol{d}'_i) = \max_{\boldsymbol{r}_{i1}} \left[\mathrm{p}(\boldsymbol{r}_{i2} \,|\, \boldsymbol{r}_{i1}, \boldsymbol{d}'_{i,2:X})\, \mathrm{p}(\boldsymbol{r}_{i1} \,|\, \boldsymbol{d}'_i)\right]$
end
for $x \leftarrow 3$ **to** n **do**
  forall $\boldsymbol{r}_{ix} \in \Omega_r$ **do**
    $\max_{\boldsymbol{r}_{i,1:x-1}} \mathrm{p}(\boldsymbol{r}_{i,1:x} \,|\, \boldsymbol{d}'_{i,x:X}) =$
    $\max_{\boldsymbol{r}_{i,x-1}} \left[\mathrm{p}(\boldsymbol{r}_{ix} \,|\, \boldsymbol{r}_{i,x-1}, \boldsymbol{d}'_{i,x:X}) \times \max_{\boldsymbol{r}_{i,1:x-2}} \mathrm{p}(\boldsymbol{r}_{i,1:x-1} \,|\, \boldsymbol{d}'_i)\right]$
  end
end
$\max_{\boldsymbol{r}_i} \mathrm{p}(\boldsymbol{r}_i \,|\, \boldsymbol{d}'_i) = \max_{\boldsymbol{r}_{i,X}} \left[\max_{\boldsymbol{r}_{i,1:X-1}} \mathrm{p}(\boldsymbol{r}_i \,|\, \boldsymbol{d}'_i)\right]$
$\hat{\boldsymbol{r}}_i = \mathrm{argmax}_{\boldsymbol{r}_i} \mathrm{p}(\boldsymbol{r}_i \,|\, \boldsymbol{d}'_i)$

while the probabilities can be stored in a m-vector. The time complexity of the algorithm is $O(Xm^2)$; one needs to check the probability of going from m depths to m depths for all the X traces. Constructing the path from the stored array is done by backtracking and requires indexing into the array X times, an $O(X)$ operation. Similarly, the Viterbi algorithm's space complexity for estimating $\boldsymbol{q}_i$ is $O(X|\Omega_q|)$, and the time complexity is $O(X|\Omega_q|^2)$. The total time complexity is $O(Xm^2) + O(X|\Omega_q|^2) \approx O(Xm^2) \approx O(XT^{4/3})$.

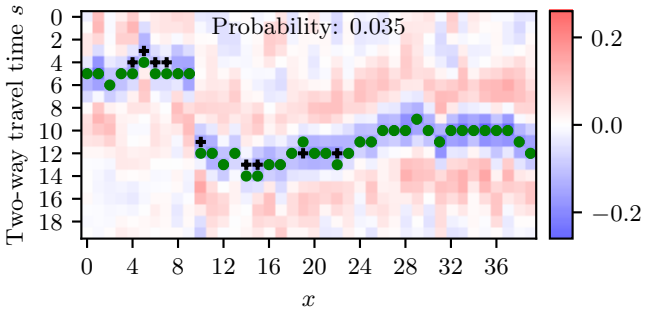
The estimated MAP horizon prediction for the seismic data shown in Figure 3.1 is presented in Figure 4.3a. Again, we have used a constant as initial guess for the reflection coefficient vector $\boldsymbol{q}_i$. A single seed point is used, at $x = 20$. The analog of Equation (3.8) for $\boldsymbol{s}$ is used to calculate the posterior probability the horizon prediction $[\hat{\boldsymbol{s}} \,|\, \boldsymbol{q}_i]$. The horizon prediction is close to the true horizon.

Figure 4.3b shows the horizon prediction for a neighboring inline. Notice that the posterior probability for this prediction is an order of magnitude larger than for the initial inline. There are two main reasons for this increase. The first reason is that the horizon prediction for the initial inline is utilized when finding predictions for the neighboring inlines, as discussed in the introduction of this chapter. The second reason is that after predicting the depths of the horizon for the initial inline, we estimate the reflection coefficients of this inline and use those estimates as initial guesses for the neighboring inlines. These inital guesses are more precise than the constant used for the initial inline. This illustrates that the tracking algorithm becomes more powerful after being kickstarted by the interpreter with one or more seed points.

Using the analog of Equation (3.12) for $\boldsymbol{s}$, we find the marginal probabilities



(a)



(b)

Figure 4.3: Posterior probabilities for horizon predictions in two neighboring synthetic seismic inlines. The green dots show the locations of the predictions. The black crosses show where the predicted depths differ from the true horizon depths.

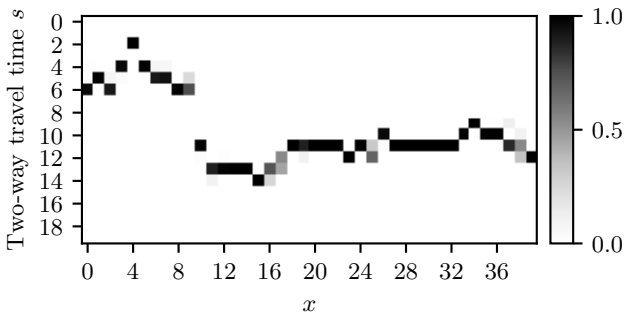


Figure 4.4: Posterior marginals for the synthetic seismic shown in Figure 4.3a.

for a seismic data inline. The marginal probabilities for the data shown in Figure 4.3a are presented in Figure 4.4. Due to the seed point and the prior, the posterior marginals are more closely centered around the horizon than the marginal likelihoods shown in Figure 3.2.

4.1.3 3D Volume Prediction — Autotracking Algorithm

In this subsection, we define the Viterbi based autotracking algorithm. We have outlined how the tracking algorithm works, but we have yet to define the clique potential function $\phi(\cdot)$. Since we condition on the reflection coefficients $\boldsymbol{q}_i$ when estimating the depths $\boldsymbol{s}_i$ and vice versa, we only need to define $\phi_s(s_{ix}, s_{jy} \,|\, q_{ix}, q_{jy})$ and $\phi_q(q_{ix}, q_{jy} \,|\, s_{ix}, s_{jy})$, $i = 1, \ldots, I$, $x = 1, \ldots, X$, $(j, y) \in \mathcal{N}_{ix}$. We let $\phi_s(s_{ix}, s_{jy} \,|\, q_{ix}, q_{jy}) = \phi_s(s_{ix}, s_{jy}) = \phi_s(|s_{ix} - s_{jy}|)$ and $\phi_q(q_{ix}, q_{jy} \,|\, s_{ix}, s_{jy}) = \phi_q(q_{ix}, q_{jy}) = \phi_q(|q_{ix} - q_{jy}|)$. That is, $\phi(\cdot)$ depends only on the absolute value of the difference between the inputs. We consider q_{ix} in steps of 0.01, i.e. $\Omega_q \subset \{-1, -0.99, \ldots, 0.99, 1\}$, around the initial guesses, and set $\phi_s = \phi_q$.

We let

$$\phi(s_{ix}, s_{jy}) = \mathrm{p}_t^*(\tau; \nu) = \left(1 + \frac{k\tau^2}{\nu}\right)^{-\frac{\nu+1}{2}}, \quad x = 2, \ldots, n\,,$$

where $p_t^*(\cdot\,; \nu)$ is an unnormalized version of the pdf of the t-distribution with ν degrees of freedom and $\tau = |s_{ix} - s_{jy}|$, and with k as a scaling factor. For the results presented this report, $\nu = 5$ and $k = 1$ is used unless otherwise specified. The reasoning behind using the t-distribution is that heavy tails are desired, since larger jumps occur at faults. Figure 4.5 illustrates a normalized version of the clique potential function for $m = 20$. The value marked by an arrow illustrates that although the probability of each big change is relatively small, the total probability of any big change is not negligible.

As we have seen, it can be sufficient to use a constant as intial guess for the reflection coefficients at the initial inline $\boldsymbol{q}_i$. At least for simple seismic data.

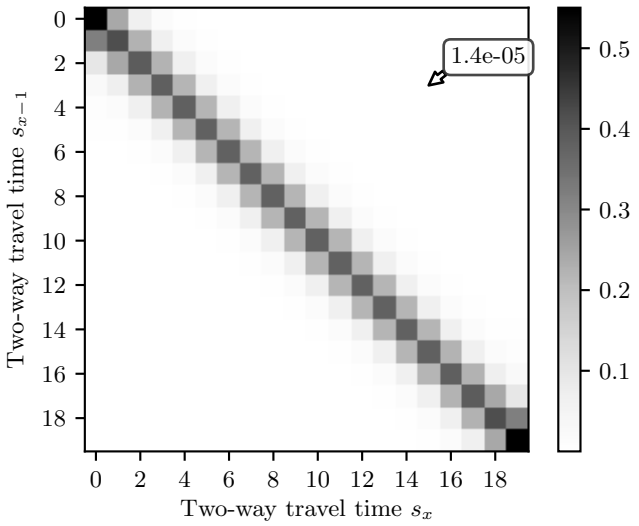


Figure 4.5: Example of a clique potential function $\phi(s_x \mid s_{x-1})$.

Let u_{ix} be the depth of the first seed point chosen by the interpreter. A naive estimate of q_{ix} is $\tilde{q}_{ix} = d_{ixu_{ix}}/w_0$, which can be used as the constant initial guess. Since the seismic data at each point is somewhat smoothed, a better estimate may be $\hat{q_{ix}} = c\tilde{q}_{ix}$, with $c \in (1.5, 2)$. Alternatively, the interpreter can manually estimate the reflection coefficient together with the horizon depth. If several seed points are provided, we use linear interpolation for the initial reflection coefficient guesses between the columns with seed points.

With the clique potentials and a procedure to generate initial reflection coefficient guesses defined, all the pieces of the Viterbi-based autotracking algorithm are in place. The algorithm, used to track seismic horizons in three dimensions, is summarized in Algorithm 3. Within Algorithm 3, Algorithm 1 and Algorithm 2 are run $2I$ times each. Thus the complexity of the autotracking algorithm is approximately $O(IXT^{4/3}) + O(IXT^2) = O(IXT^2)$.

Multiple horizon tracking

To track multiple horizons, we can run Algorithm 3 as it is once for each horizon. However, we can make some improvements. First, we utilize the fact that horizons generally can not cross each other. We incorporate this constraint by setting the likelihoods to 0 for depths that violate it. Secondly, after tracking each horizon, we use our model to subtract the theoretical expected effects of the horizons on the seismic data. That is, we subtract the expected values of $[\boldsymbol{d}_{ix} \mid \hat{\boldsymbol{r}}_{ix}] \sim \mathrm{p}(\boldsymbol{d}_{ix} \mid \hat{\boldsymbol{r}}_{ix})$, $i = 1, \ldots, I$, $x = 1, \ldots, X$ from $\boldsymbol{d}$.

Finally, since there is some correlation between how different horizons propagate through a particular seismic volume, we can use this to adapt the clique potentials for neighboring horizons. Let us assume that the interpreter be-

Algorithm 3: Viterbi Autotracking Algorithm

input : Seismic data matrix $\boldsymbol{d}$ of size $(I \times X \times T)$; likelihood $\mathrm{p}(\boldsymbol{d}_{ix} \,|\, \boldsymbol{r}_{ix})$, $\boldsymbol{r}_{ix} \in \Omega_r$, $x = 1, \ldots, X$; clique potentials $\phi(\cdot, \cdot)$; and seed points on the form (i, x, s_{ix}, q_{ix})

output: Horizon prediction $\hat{\boldsymbol{r}}(\hat{\boldsymbol{s}}, \hat{\boldsymbol{q}})$

Choose any seed point (i, x, s_{ix}, q_{ix})

Use all seed points in inline i to create inital guess $\hat{\boldsymbol{q}}_i$ for the inline reflection coefficients $\boldsymbol{q}_i$ by interpolating

$\mathrm{p}(s_{i1} \,|\, \boldsymbol{d}'_i)$; $\mathrm{p}(s_{ix} \,|\, s_{i,x-1}, \boldsymbol{d}'_{i,x:X}) \leftarrow$ output of Algorithm 1 for input $\boldsymbol{d}'_i = (\hat{\boldsymbol{q}}_i, \boldsymbol{d}_i)$; $\mathrm{p}(\boldsymbol{d}_{ix} \,|\, s_{ix}, \hat{q_{ix}})$; $\phi(\cdot, \cdot)$

$\hat{\boldsymbol{s}}_i \leftarrow$ output of Algorithm 2 for input $\mathrm{p}(s_{i1} \,|\, \boldsymbol{d}'_i)$; $\mathrm{p}(s_{ix} \,|\, s_{i,x-1}, \boldsymbol{d}'_{i,x:X})$

$\mathrm{p}(q_{i1} \,|\, \boldsymbol{d}'_i)$; $\mathrm{p}(q_{ix} \,|\, q_{i,x-1}, \boldsymbol{d}'_{i,x:X}) \leftarrow$ output of Algorithm 1 for input $\boldsymbol{d}'_i = (\hat{\boldsymbol{s}}_i, \boldsymbol{d}_i)$; $\mathrm{p}(\boldsymbol{d}_{ix} \,|\, \hat{s}_{ix}, q_{ix})$; $\phi(\cdot, \cdot)$

$\hat{\boldsymbol{q}}_i \leftarrow$ output of Algorithm 2 for input $\mathrm{p}(q_{i1} \,|\, \boldsymbol{d}'_i)$; $\mathrm{p}(q_{ix} \,|\, q_{i,x-1}, \boldsymbol{d}'_{i,x:X})$

for $j \leftarrow i + 1$ **to** I **do**

  $\mathrm{p}(s_{j1} \,|\, \boldsymbol{d}'_j)$; $\mathrm{p}(s_{jx} \,|\, s_{j,x-1}, \boldsymbol{d}'_{j,x:X}) \leftarrow$ output of Algorithm 1 for input $\boldsymbol{d}'_j = (\hat{\boldsymbol{s}}_{j-1}, \hat{\boldsymbol{q}}_{j-1}, \boldsymbol{d}_j)$; $\mathrm{p}(\boldsymbol{d}_{jx} \,|\, s_{jx}, \hat{q}_{jx})$; $\phi(\cdot, \cdot)$

  $\hat{\boldsymbol{s}}_j \leftarrow$ output of Algorithm 2 for input $\mathrm{p}(s_{j1} \,|\, \boldsymbol{d}'_j)$; $\mathrm{p}(s_{jx} \,|\, s_{j,x-1}, \boldsymbol{d}'_{j,x:X})$

  $\mathrm{p}(q_{j1} \,|\, \boldsymbol{d}'_j)$; $\mathrm{p}(q_{jx} \,|\, q_{j,x-1}, \boldsymbol{d}'_{j,x:X}) \leftarrow$ output of Algorithm 1 for input $\boldsymbol{d}'_j = (\hat{\boldsymbol{s}}_j, \hat{\boldsymbol{q}}_{j-1}, \boldsymbol{d}_j)$; $\mathrm{p}(\boldsymbol{d}_{jx} \,|\, \hat{s}_{jx}, q_{jx})$; $\phi(\cdot, \cdot)$

  $\hat{\boldsymbol{q}}_j \leftarrow$ output of Algorithm 2 for input $\mathrm{p}(q_{j1} \,|\, \boldsymbol{d}'_j)$; $\mathrm{p}(q_{jx} \,|\, q_{j,x-1}, \boldsymbol{d}'_{j,x:X})$

end

for $j \leftarrow i - 1$ **to** 0 **do**

  $\mathrm{p}(s_{j1} \,|\, \boldsymbol{d}'_j)$; $\mathrm{p}(s_{jx} \,|\, s_{j,x-1}, \boldsymbol{d}'_{j,x:X}) \leftarrow$ output of Algorithm 1 for input $\boldsymbol{d}'_j = (\hat{\boldsymbol{s}}_{j+1}, \hat{\boldsymbol{q}}_{j+1}, \boldsymbol{d}_j)$; $\mathrm{p}(\boldsymbol{d}_{jx} \,|\, s_{jx}, \hat{q}_{jx})$; $\phi(\cdot, \cdot)$

  $\hat{\boldsymbol{s}}_j \leftarrow$ output of Algorithm 2 for input $\mathrm{p}(s_{j1} \,|\, \boldsymbol{d}'_j)$; $\mathrm{p}(s_{jx} \,|\, s_{j,x-1}, \boldsymbol{d}'_{j,x:X})$

  $\mathrm{p}(q_{j1} \,|\, \boldsymbol{d}'_j)$; $\mathrm{p}(q_{jx} \,|\, q_{j,x-1}, \boldsymbol{d}'_{j,x:X}) \leftarrow$ output of Algorithm 1 for input $\boldsymbol{d}'_j = (\hat{\boldsymbol{s}}_j, \hat{\boldsymbol{q}}_{j+1}, \boldsymbol{d}_j)$; $\mathrm{p}(\boldsymbol{d}_{jx} \,|\, \hat{s}_{jx}, q_{jx})$; $\phi(\cdot, \cdot)$

  $\hat{\boldsymbol{q}}_j \leftarrow$ output of Algorithm 2 for input $\mathrm{p}(q_{j1} \,|\, \boldsymbol{d}'_j)$; $\mathrm{p}(q_{jx} \,|\, q_{j,x-1}, \boldsymbol{d}'_{j,x:X})$

end

gins by tracking the top horizon of interest. In general, the top horizon is the simplest to track, since the seismic data from smaller depths is less noisy. Furthermore, faults have had less time to develop. When tracking the second horizon from top, we center the clique potentials for depths around the changes in depth from the top horizon, and so on. That is, if $s_{ix} - s_{i-1,x} = k$ for the top horizon, then we let $\phi_s(s_{ix}, s_{i-1,x}) = \phi_s(s_{ix} - s_{i-1,x})$ for the next horizon have its largest value for $\phi_s(k)$ instead of at 0. This adaptation of the clique potentials is equivalent to shifting the seismic data so that the tracked horizon appears as flat. We are also more certain about the changes, so we increase ν by a factor of 10. This increase moves the t-distribution closer to the normal distribution, reducing the weight of the tails.

The Viterbi based autotracking algorithm is demonstrated in Chapter 5. In the next section, we develop a greedy autotracking algorithm.

4.2 Greedy Autotracker

In this section, we develop a greedy autotracking algorithm based on the hidden Markov model developed in Chapter 3. The algorithm is greedy in the sense that at each step it expands the horizon to a depth at some column, and the depth is chosen based on what seems optimal locally. No global properties, or properties for entire inlines, are considered.

The algorithm takes one or more seed points as input. Certain depths in columns neighboring the seed point columns are added to a list termed the frontier. The horizon is expanded to one new column at each step of the algorithm, where the new column is one of the columns in the frontier. When the depth in a new column is determined, the column is removed from the frontier. Next, all columns neighboring the new column, and for which we have not yet determined the depth of the horizon, are added to the frontier. This procedure is repeated until the frontier is empty, at which point the horizon depth is determined for all columns. See Figure 4.6 for an illustration of a frontier at a certain time step.

At each step, we need to determine which column and depth to expand the horizon to. Assume that the depth of the horizon at column (i, x) is set to s_{ix}. Assume further that the depth is not set for column $(i+1, x)$, implying that this column must be in the frontier. To decide which depth to expand the horizon to in this column, we consider the product $\psi(s_{i+1,x}, s_{ix}, q_{ix}, \boldsymbol{d}_{i+1,x}) = \phi(s_{ix}, s_{i+1,x})\mathrm{p}(\boldsymbol{d}_{i+1,x} \mid s_{i+1,x}, q_{ix})$, which we call the score of $s_{i+1,x}$. We use the reflection coefficient for the neighboring point when expanding the horizon. After the depth at $(i+1, x)$ is determined, we use $\hat{q}_{i+1,x} = \mathrm{argmax}_{q_{i+1,x}} \mathrm{p}(\boldsymbol{d}_{i+1,x} \mid s_{i+1,x}, q_{i+1,x})$ as an estimate for the coefficient. The score is considered for depths $s_{i+1,x}$ not to distant from s_{ix}, e.g. the interval $\{s_{ix} - m/2, s_{ix} - m/2 + 1, \ldots, s_{ix} + m/2 - 1, s_{ix} + m/2\}$, where m is an even integer. Here, m serves about the same function as for the Viterbi based autotracking algorithm. However, m can generally be smaller here, since it

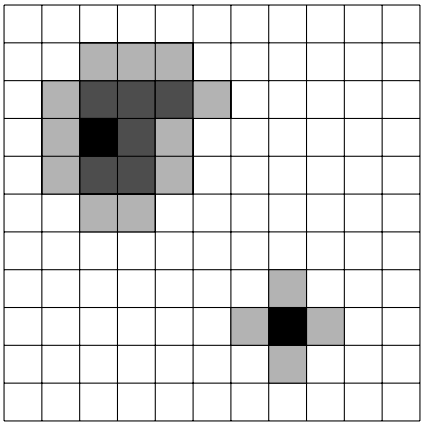


Figure 4.6: Illustration of frontier at a certain time step when using the greedy tracking method. Black: Seed columns. Dark grey: Tracked columns. Light grey: Frontier columns.

limits the differences as opposed to putting a limit on the depths of the entire horizon. At each step, we expand the horizon to the depth with the highest score, considered across all columns in the frontier.

It is possible that a column is added to the frontier multiple times before the algorithm expands the horizon to that particular column — in theory up to four times. To account for this, we let the score be

$$\psi_1(s_{ix}, s_{ix}^{(1)}, \boldsymbol{d}_{ix}) = \psi(s_{ix}, s_{ix}^{(1)}, \boldsymbol{d}_{ix})^4$$

the first time s_{ix} is added to the frontier, where $s_{ix}^{(1)}$ is the predicted depth in the first column resulting in s_{ix} being added to the frontier. Furthermore, we let

$$\begin{aligned}
\psi_2(s_{ix}, s_{ix}^{(2)}, \boldsymbol{d}_{ix}) &= \psi(s_{ix}, s_{ix}^{(2)}, \boldsymbol{d}_{ix})^2 \psi(s_{ix}, s_{ix}^{(1)}, \boldsymbol{d}_{ix})^2, \\
\psi_3(s_{ix}, s_{ix}^{(3)}, \boldsymbol{d}_{ix}) &= \prod_{k=1}^{3} \psi(s_{ix}, s_{ix}^{(k)}, \boldsymbol{d}_{ix})^{4/3}, \\
\psi_4(s_{ix}, s_{ix}^{(4)}, \boldsymbol{d}_{ix}) &= \prod_{k=1}^{4} \psi(s_{ix}, s_{ix}^{(k)}, \boldsymbol{d}_{ix}).
\end{aligned}$$

This way the scores are comparable across columns and depths added to the frontier more than once.

The greedy autotracking algorithm is summarized in Algorithm 4. It is not straightforward to find the algorithm's complexity , and we will not attempt to do it exactly. However, assuming a somewhat arbitraty average of IXm locations are in the frontier on average, we can find an estimate. If we maintain a list of locations sorted by the score, finding the location with maximum score is an $O(1)$ operation. Calculating each score in the first place is an $O(n^2)$

operation since it involves a likelihood calculation. Estimating $\hat{q}_{ix}$ involves calculating the likelihood $\mathrm{p}(\boldsymbol{d}_{ix} \mid s_{ix}, \tilde{q}_{ix})$ for about m values of $\tilde{q}_{ix}$. Thus this step has time complexity $O(mn^2)$. Inserting each of the at most $4m$ locations into the sorted list is an $O(\log(IXm))$ operation, using e.g. a binary search to find the position of each location in the list. The complexity of the final step of the loop is relatively low. There are IX columns, each of which are added to the horizon once (except the seed points). Thus the estimated complexity is $O(IX[mn^2 + \log(IXm)]) \approx O(IX[T^{4/3} + \log(IXT^{2/3})])$. Assuming $I \approx X \approx T$, we arrive at $O(IX[T^{4/3} + 8/3\log(T)]) = O(IXT^{4/3})$.

Algorithm 4: Greedy Autotracking Algorithm

input : Seismic data matrix $\boldsymbol{d}$ of size $(I \times X \times T)$; likelihood $\mathrm{p}(\boldsymbol{d}_{ix} \mid \boldsymbol{r}_{ix})$, $\boldsymbol{r}_{ix} \in \Omega_r$, $x = 1, \ldots, X$; clique potentials $\phi(\cdot, \cdot)$; and seed points on the form (i, x, s_{ix}, q_{ix})

output: Horizon prediction $\hat{\boldsymbol{r}}(\hat{\boldsymbol{s}}, \hat{\boldsymbol{q}})$

forall seed points (i, x, s_{ix}, q_{ix}) **do**
  Add (j, y, s) to frontier for all $(j, y) \in \mathcal{N}_{ix}$, $s \in \{\max(s_{ix} - m/2, 1), \ldots, \min(s_{ix} + m/2, T)\}$ if $\hat{s}_{jy}$ is not set
end
repeat
  Find (i, x, s) from frontier with maximum score
  Let $\hat{s}_{ix} = s$
  Estimate $\hat{q}_{ix}$
  Add (j, y, s) to frontier for all $(j, y) \in \mathcal{N}_{ix}$, $s \in \{\max(\hat{s}_{ix} - m/2, 1), \ldots, \min(\hat{s}_{ix} + m/2, T)\}$ if $\hat{s}_{jy}$ is not set
  Remove entire column (i, x) from frontier
until frontier is empty

Comparing the estimated complexity with the complexity of the Viterbi based tracking algorithm, it seems that the greedy algorithm is, asymptotically, faster. The greedy algorithm also has the advantage of working in three dimensions, compared to one inline at a time for the Viterbi algorithm. The downside is that the greedy approach is less holistic, in the sense that it only considers local optimality. In practice, one can imagine the greedy algorithm overcoming a fault by tracking around it. However, it might be problematic to track a horizon across a fault running through the entire seismic volume.

To track multiple horizons, we implement the same adaptations as discussed at the end of Section 4.1.3. In the next chapter, we present and discuss experiments for both algorithms.

Chapter 5

Experimental Results

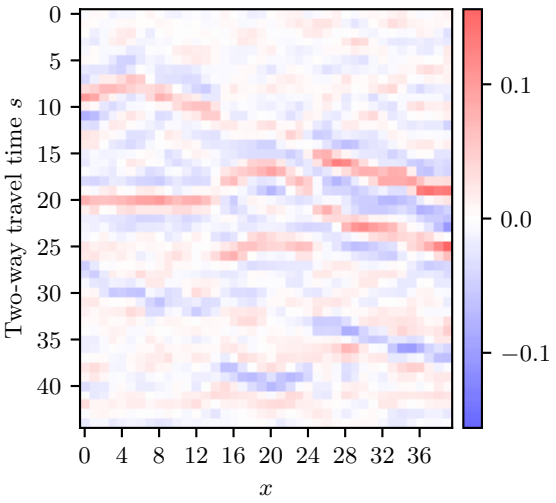
In this chapter, we demonstrate the horizon tracking algorithms developed in Chapter 4, and discuss some limitations. First, in Section 5.1, we present results from experimenting with synthetic seismic data. In Section 5.2 we show results for real seismic data.

5.1 Synthetic Data

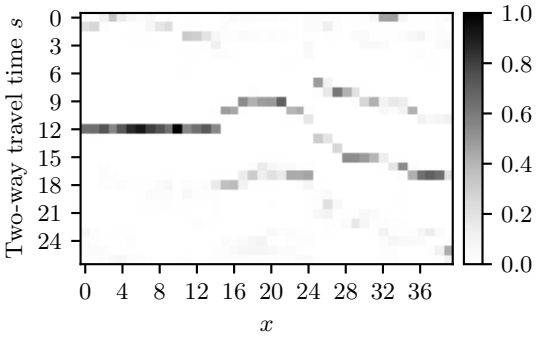
The synthetic seismic data used in this report, even though simple, is useful since we can freely change it to test the performance of tracking algorithms on specific cases. In this section, we consider a three-dimensional synthetic data $\boldsymbol{d}$ of size $(40 \times 40 \times 45)$. Inline $i = 10$ of the seismic data is presented in Figure 5.1a. The seismic data is created from three synthetic horizons using the model described in Chapter 3. Furthermore, we have used Gaussian convolution along the inlines and crosslines to introduce lateral correlations, while also smoothing the data. The reflection coefficients of the two top horizons are similar. Using a constant reflection coefficient from the middle horizon as initial guess for the reflection vector $\boldsymbol{q}_i$, we get the likelihoods shown in Figure 5.1b. As discussed in Chapter 4, we do not need to consider all depths when tracking a horizon. Here, we consider depths $t = 8$ to $t = 34$.

The top left plot of Figure 5.2 shows the horizon prediction for the middle horizon at inline $i = 10$ obtained using the Viterbi based autotracker. The posterior marginals are presented below the prediction. We see that the autotracker tracks the wrong horizon between the faults. The right plots show that the error has propagated to $i = 20$.

To obtain the results shown in Figure 5.2, we have used a single seed point at $(i, x) = (10, 10)$. Figure 5.3 shows the corresponding results when using a single seed point at $(10, 20)$, i.e. between the faults, instead. Here, the correct horizon is tracked throughout inline $i = 10$, and this also propagates to $i = 20$. We have deliberately created a challenge for the tracking algorithm by setting similar reflection coefficients for the two top horizons, and through the positioning and sizes of the faults. With that in mind, the results obtained



(a) Synthetic seismic data.



(b) Likelihoods for the synthetic seismic data shown above from depth $t = 8$ to $t = 34$.

Figure 5.1: Synthetic seismic data and likelihoods.

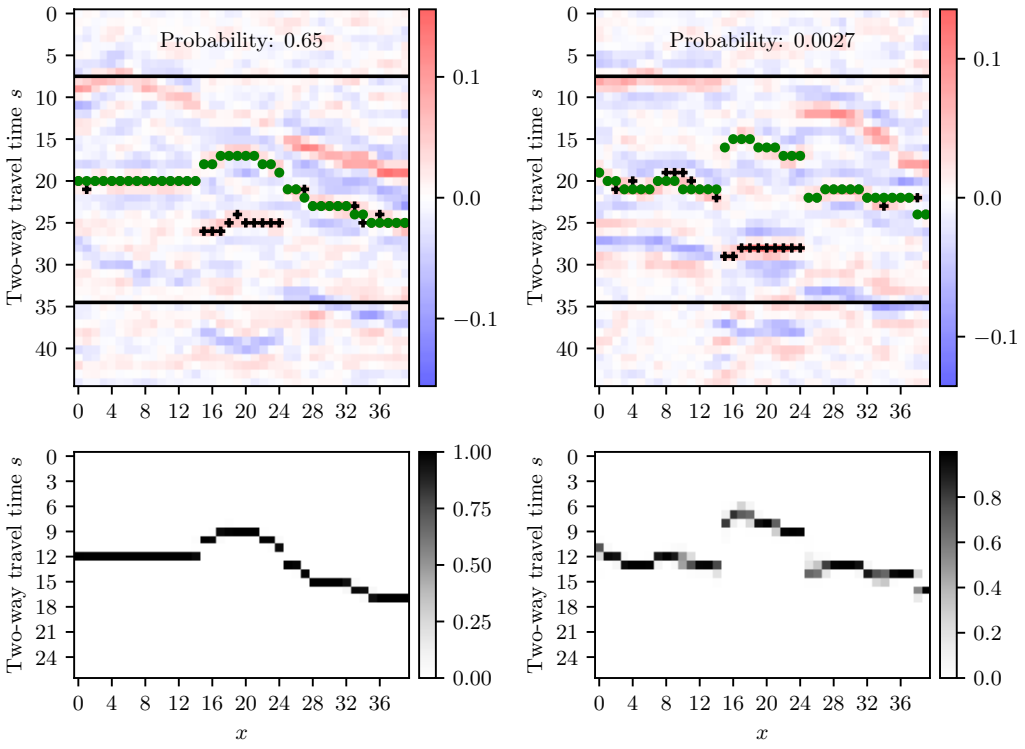


Figure 5.2: Top left plot: Tracked seismic horizon at inline $i = 10$ with seed point at crossline $x = 10$. The green dots show the locations of the predictions. The black crosses show where the predicted depths differ from the true horizon depths. Top right: Tracked horizon at $i = 20$. The lower plots show the posterior marginals for the horizon in the corresponding inlines from depth $t = 8$ to $t = 34$.

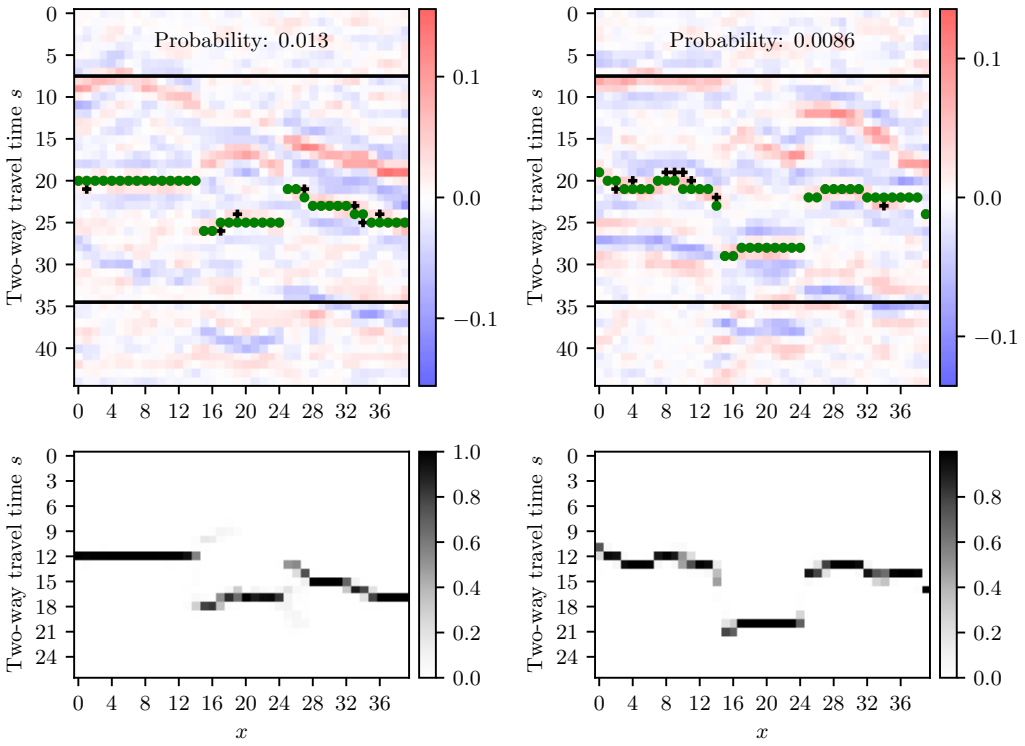


Figure 5.3: Top left plot: Tracked seismic horizon at inline $i = 10$ with seed point at crossline $x = 20$. The green dots show the locations of the predictions. The black crosses show where the predicted depths differ from the true horizon depths. Top right: Tracked horizon at $i = 20$. The lower plots show the posterior marginals for the horizon in the corresponding inlines from depth $t = 8$ to $t = 34$.

here illustrate the capabilities of the tracking algorithm well.

To help the algorithm track the middle horizon correctly, as shown in Figure 5.3, we chose a particularly suitable seed point. However, as discussed in Chapter 4, it should generally be beneficial to track the top horizon first and then perform the steps discussed in Section 4.1.3 for tracking multiple horizons.

In the top left plot of Figure 5.4 we see the prediction for the top horizon obtained using suitable seed points. Using the model described in Chapter 3, we remove the expected effects of the horizon on the seismic data. The result of this operation is shown in the top right plot in Figure 5.4. Next, we change the clique potential so that, in practice, we shift the seismic data as shown in the bottom left plot, making the top horizon flat. The combined result of the two operations is shown in the bottom right plot of Figure 5.4.

After tracking the top horizon and adapting the seismic data and the clique potential, it should be easier to track the middle horizon. Figure 5.5 shows

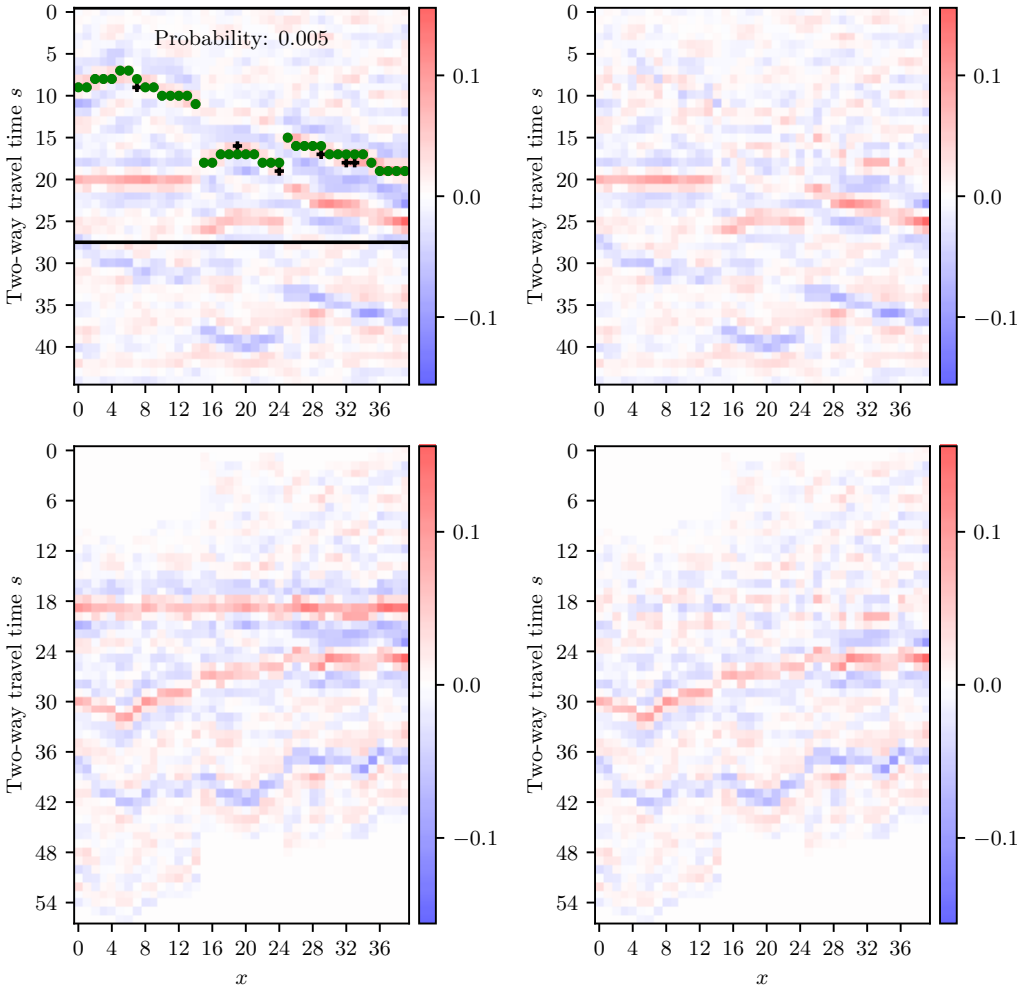


Figure 5.4: Top left plot: Tracked seismic horizon at inline $i = 10$. The green dots show the locations of the predictions. The black crosses show where the predicted depths differ from the true horizon depths. Top right: Seismic data with predicted effects of horizon removed. Lower left: Seismic data shifted according to depth changes of predicted horizon. Lower right: Combined result of the two aforementioned operations.

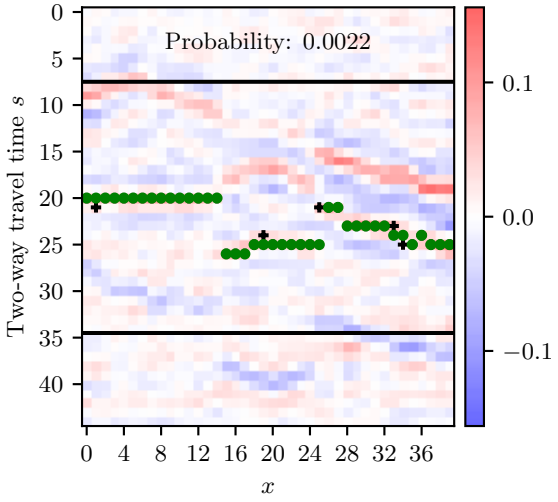


Figure 5.5: Tracked seismic horizon at inline $i = 10$ with seed point at crossline $x = 10$. The green dots show the locations of the predictions. The black crosses show where the predicted depths differ from the true horizon depths.

the result. Now the correct horizon is tracked even though we use a seed point at $(10, 10)$, which did not result in satisfactory results at the first attempt of tracking the horizon.

The left plots in Figure 5.6 illustrate the tracked horizons in three dimensions. The right plots illustrate the true synthetic horizons. We see that the true horizons are somewhat more volatile than the predictions. However, the predictions are relatively precise throughout the seismic volume.

In the examples used so far, the direction of the faults has been perpendicular to the inlines. This is beneficial for the Viterbi based tracking algorithm since it works by propagating the horizon along the inlines. The right plot of Figure 5.7 shows the top horizon of the same data as before, but now rotated so that the faults are parallel to the inlines instead of perpendicular to them. The left plot shows an attempt to track the horizon. The tracking algorithm does not track the horizon well across the faults. In Chapter 6, we discuss possible ways of amending this issue.

Finally, we present the result of attempting to track the top horizon of the synthetic seismic data illustrated in Figure 5.1 using the greedy algorithm. See Figure 5.8. Even though we have used a seed point at every plateau, i.e. before the first fault, between the faults, and after the second fault, the results are not convincing. Before the first fault, at $x = 15$, the middle horizon is tracked instead of the top one.

Since the results obtained using the greedy algorithm are poor compared to the results obtained using the Viterbi based autotracker, we do not consider the greedy algorithm further. This difference in performance illustrates that

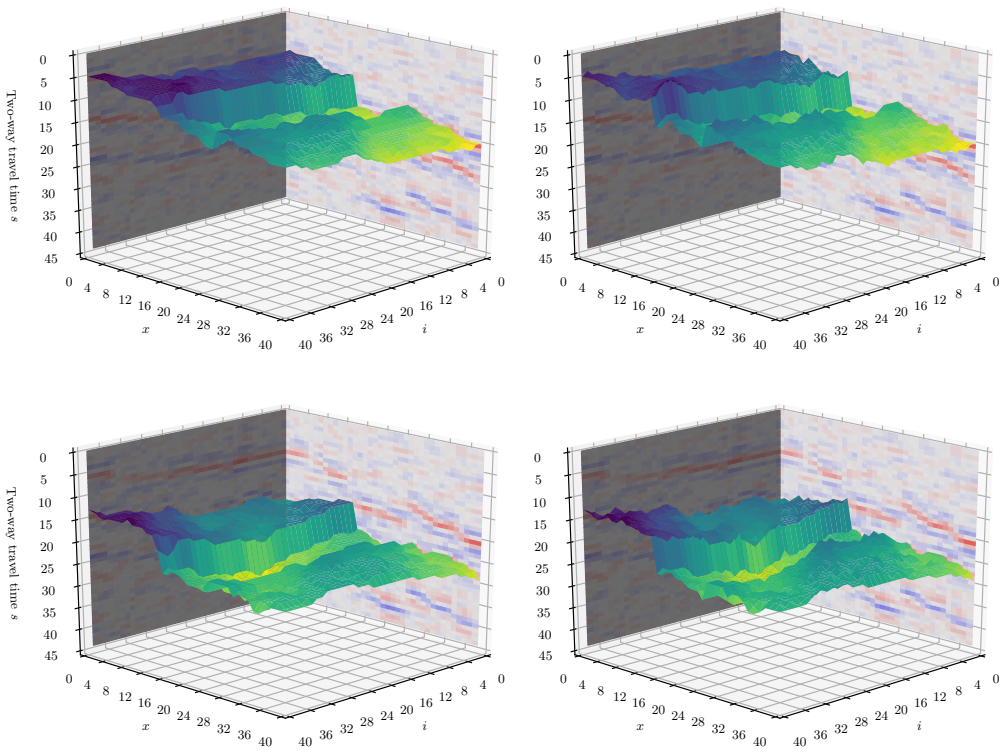


Figure 5.6: Left: Horizon predictions. Right: True horizons.

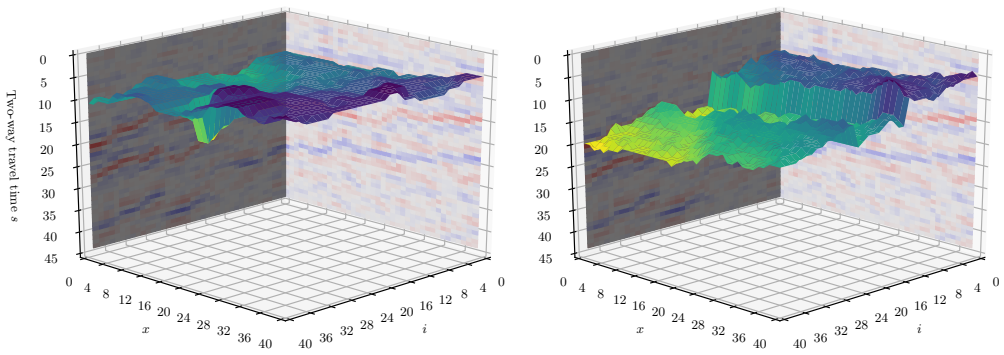


Figure 5.7: Left: Horizon prediction. Right: True horizon.

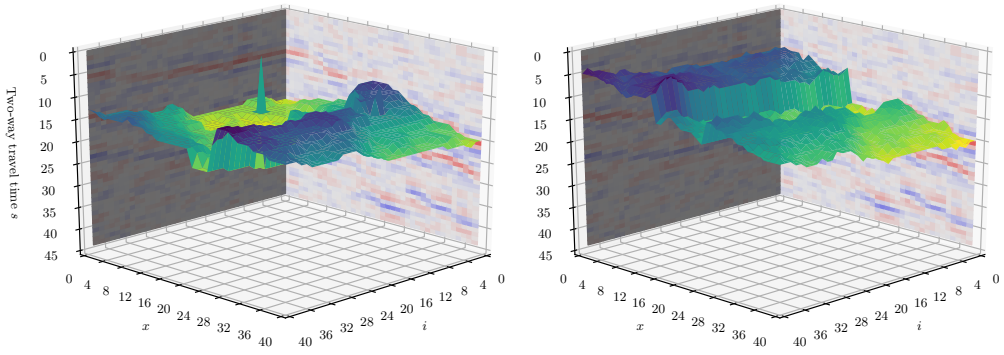


Figure 5.8: Left: Horizon prediction obtained using a greedy method. Right: True horizon.

the large scale properties considered when optimizing one inline at a time are important. In the next section, we present some results on real seismic data obtained using the Viterbi based autotracker.

5.2 Real Data

Next, we test the Viterbi based autotracking algorithm developed in Section 4.1.3 on real synthetic, seismic data. We need an estimate of the wavelet used to acquire the data, and we need estimates for the structured noise variable σ_δ^2 and the white noise variable σ_ϵ^2. When working with synthetic data, these are known variables. In this section, we consider the open-source Volve dataset (2020). There are some open-source horizon interpretations available for the Volve dataset. Unfortunately, we have not been able to acquire a wavelet estimate created by an expert.

Figure 5.9 shows an inline of the Volve seismic data. Some areas in the plot are magnified, illustrating some challenges real data may present. The first magnified area from the top illustrates that real seismic data can have areas with missing data. This might stem from processing or the data collection procedure itself. Note also that the spacial coverage increases with depth in this dataset. The second magnified area illustrates that there may be lots of small and large faults. Even though faults are often considered to exist in two-dimensional planes, it is more precise to think of them as thin volumes where different layers are mixed by the movements that create faults. This makes it all the more challenging to track horizons across faults.

The estimated wavelet we use to track a horizon is shown in Figure 5.10. The method used to estimate the wavelet is described in SEG Wiki (2020). We do not describe the method in this report.

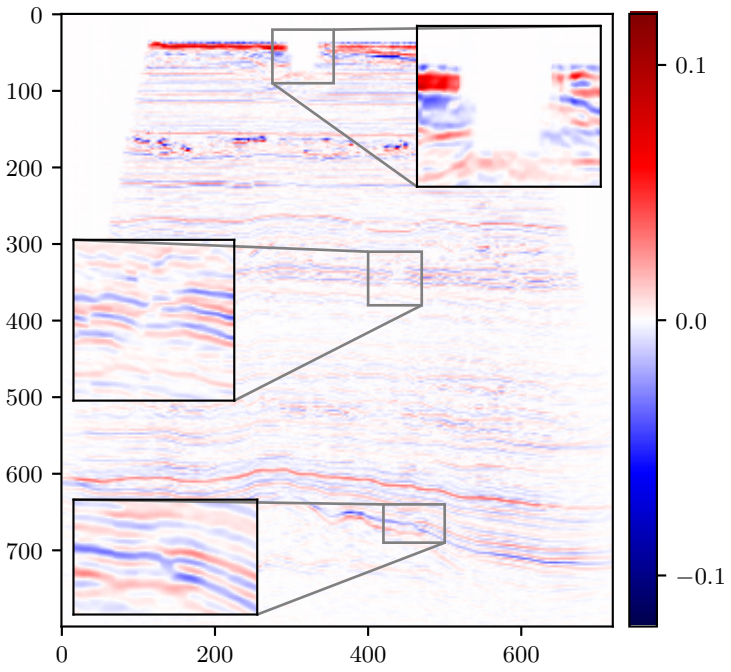


Figure 5.9: Seismic data cross section from the open-source Volve dataset (2020) released by Equinor under the *Creative Commons BY-NC-SA 4.0* license.

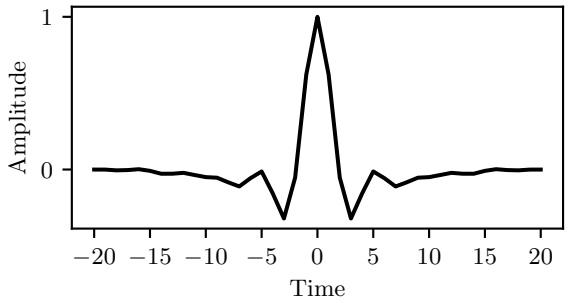
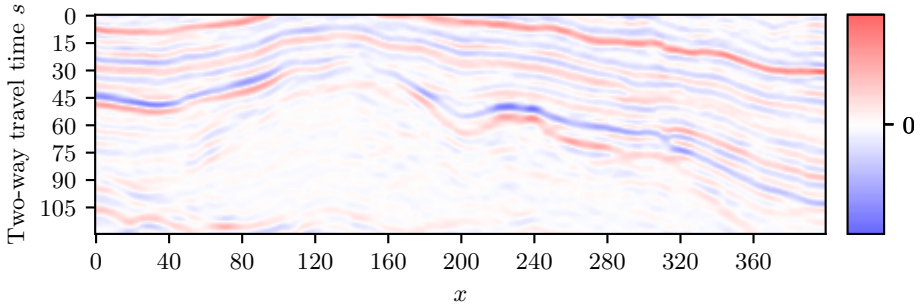


Figure 5.10: Estimated wavelet for the Volve seismic data.

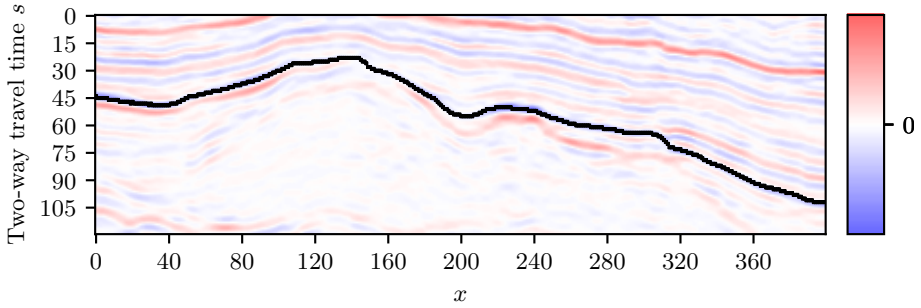
For our experiment, we consider the seismic data from crosslines $x = 150$ to $x = 550$ and depths from $t = 600$ to $t = 719$. The data is shown in Figure 5.11a. The horizon we focus on is shown in Figure 5.11b. Using the constant -0.06 as initial guess for the reflection coefficient vector $\boldsymbol{q}_i$, and letting $\sigma_\delta = 0.005$ and $\sigma_\epsilon = 0.0005$, we get the likelihoods shown in Figure 5.11c. The noise variables are chosen so that the likelihoods stand out as much as possible at the true horizon depths. The likelihoods stand out along the horizon at some crosslines, but at other crosslines they do not.

Figure 5.12a shows the result of attempting to track the horizon with a single seed point at $x = 55$. The result is not impressive, as one might expect considering the likelihoods from Figure 5.11c. We illustrated in Section 5.1 how the tracking algorithm can become more powerful after tracking the initial inline. To test this effect on the real data, we provide enough seed points to track the horizon well in the initial inline; see Figure 5.12b. Figure 5.12c shows the tracked horizons 5 inlines later. Again, the results are not convincing.

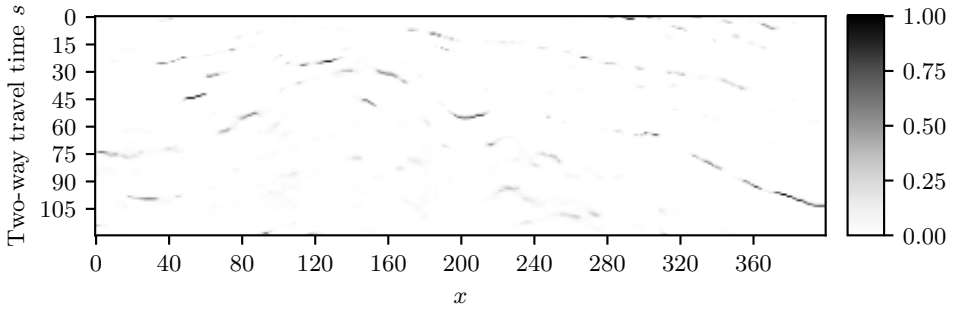
The tracked horizon illustrated in Figure 5.12a is too volatile compared to the true horizon. It is natural to attempt tracking the horizon using a stricter prior, i.e. a prior restricting the volatility of the horizon more. Figure 5.13 shows the results of tracking the horizon using a single seed point with $\nu = 100$ and $k = 2$. These parameter values give a very strict prior. However, the tracked horizon is still not smooth enough. Using even stricter priors results in numerical issues since the likelihoods are close to zero at some of the true horizon locations while large jumps also have close to zero probabilities. Furthermore, such strict priors would make it challenging to track horizons with faults. In Chapter 6, we discuss possible reasons for the challenges illustrated here and in Section 5.1, as well as potential ways to improve the performance of the algorithm.



(a) Subset of Volve seismic data inline.

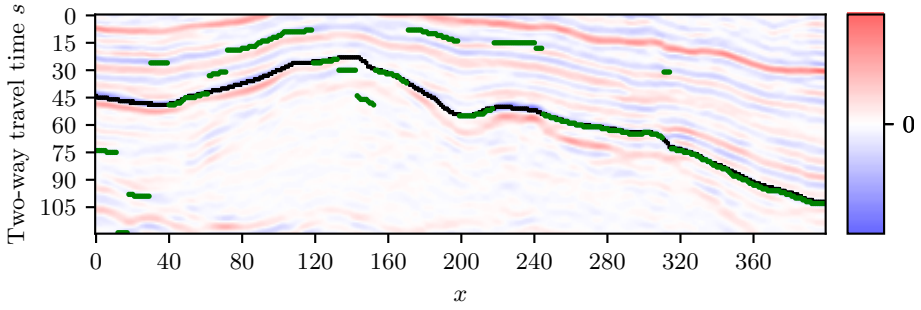


(b) Subset of Volve seismic data inline with true horizon in black.

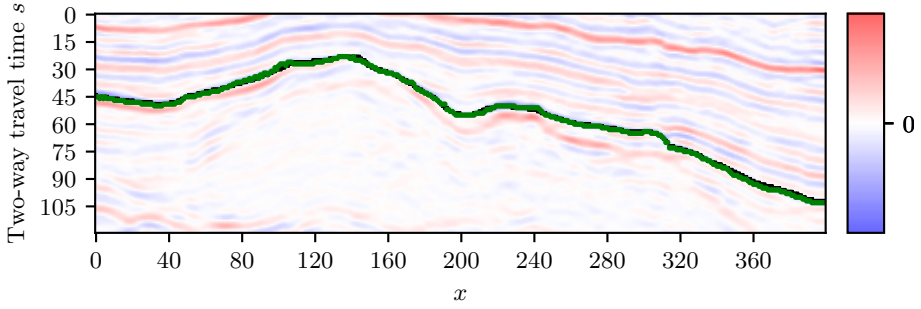


(c) Likelihoods for the seismic data above using the constant -0.06 as initial guess for the reflection coefficient vector $\boldsymbol{q}_i$.

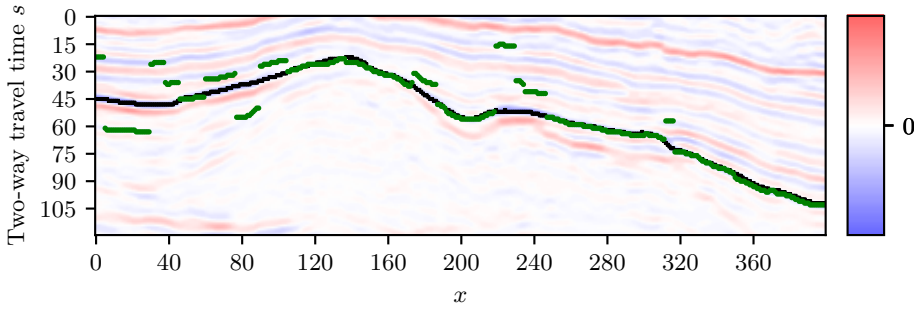
Figure 5.11: Subset of Volve seismic data inline, true horizon and likelihoods.



(a) Green: Tracked horizon using a single seed point.



(b) Green: Tracked horizon using several seed points.



(c) Green: Tracked horizon 5 inlines after initial inline.

Figure 5.12: The top two plots show attempts to track the horizon marked with black points in the initial inline, first with a single seed point and then with several seed points. The lower plot shows the tracked horizons five inlines after the initial inline.

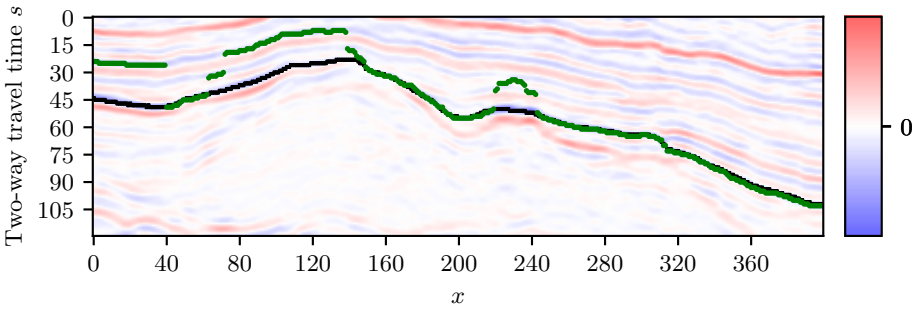


Figure 5.13: Green: Tracked horizon using a single seed point and a relatively strict prior.

Chapter 6

Discussion

In this chapter, we summarize and discuss the results from Chapter 5. Furthermore, we compare some aspects of the Viterbi based autotracking algorithm developed in Chapter 4 with approaches that may be considered state of the art of horizon autotracking. Finally, we discuss potential ways to improve the Viterbi based autotracker.

We demonstrated in Section 5.1 that the Viterbi based tracking algorithm performs well on synthetic data with properties closely resembling those of the model the algorithm is based on. The greedy algorithm did not perform well, indicating that using global properties for each inline pays off. The problem is that when the greedy algorithm encounters a fault, the jump may be too large to be considered worthwhile when considering local properties only.

The Viterbi based autotracking algorithm also fell short when attempting to track a horizon across faults propagating parallelly to the inlines. Real data can contain faults propagating in different directions, so rotating the data is not a universal solution. The interpreter could add seed points at multiple inlines, but this would require more effort from the interpreter. Another alternative is to use the algorithm iteratively, going back and forth sequentially or randomly, and possibly considering crosslines and not only inlines. We could also use Markov chain Monte Carlo sampling to simulate the horizon in one inline or crossline at a time and see if the result converges towards the true horizon.

Testing the Viterbi based autotracker on real data revealed further challenges, as the results are not convincing. One reason may be a poor wavelet estimate. This hypothesis is strengthed by the fact that removing the effects of the predicted horizon on the seismic data is not successful; see Figure 6.1. It seems like the estimated wavelet is not smooth enough. Since the model behind the method is not an exact representation of reality, we hypothesize that the best way to estimate the wavelet would be to extend the algorithm to a three-step algorithm. As described in Chapter 4, we first estimate the horizon depths, and then use those estimates to find an estimate for the reflection coefficients. Based on those estimates, we could estimate the wavelet.

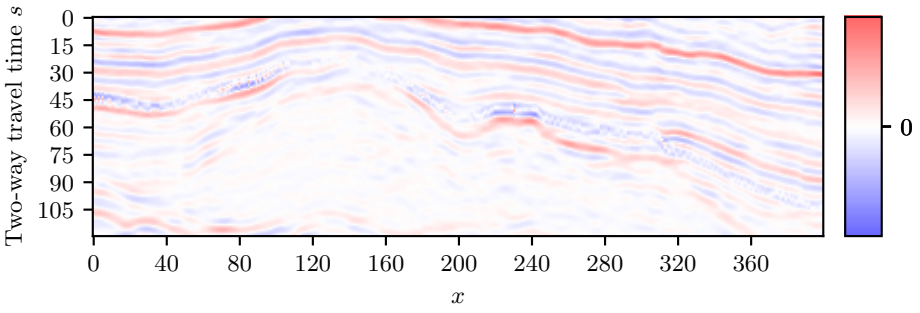


Figure 6.1: Subset of Volve seismic data inline, where we have removed the predicted effects of the tracked horizon illustrated in Figure 5.12b.

The real data experiments also revealed that even with a precise estimate of the horizon at the initial inline, the algorithm did not properly propagate the horizon to nearby inlines. As mentioned in the previous paragraph, a poor wavelet estimate may partly be to blame. However, it is likely that changing from a 4-neighborhood system as we have used here to an 8-neighborhood system will increase the performance of the algorithm. In practice, this would mean including the clique potentials $\phi(s_{ix}, \hat{s}_{i-1,x-1})$ and $\phi(s_{ix}, \hat{s}_{i-1,x+1})$ as factors when estimating the posterior transition probabilities $\mathrm{p}(s_{ix} \mid s_{i,x-1}, \boldsymbol{q}_{i,x:X}, \boldsymbol{d}'_{i,x:X})$, and an analogous adaption for the reflection coefficients. Such a neighborhood system should be particularly helpful for smooth horizons, as all three neighboring horizon points from the previous inline are usually at about the same depth. The computational cost of this change is relatively small.

Real seismic data is more complex than the synthetic seismic data considered in this thesis. One specific difference is that the number of seismic data voxels above and below the depth of a horizon having the same sign as at the horizon is more variable in real data; see Figure 5.11. Thus a better wavelet estimate might not be sufficient to track a real horizon precisely. A potential solution is using a univariate likelihood model, e.g. by letting d_{ixs_x} be normally distributed around the maximum amplitude of the wavelet multiplied by the reflection coefficient q_{ix}. This would involve only using the center index of the wavelet, and dismissing the rest of the information contained in the wavelet when tracking a horizon. Nonetheless, it is a possibility that may be worthwhile exploring.

Figure 6.2 presents an inline from the open-source Netherlands offshore F3 dataset (2020). We have magnified two particularly complex areas. The F3 data is clearly more complex than the Volve data. Due to the suboptimal results when testing the autotracking algorithm on the relatively simpler Volve data, and since there are limited open-source expert interpretations, we do not discuss tracking of horizons in this complex dataset.

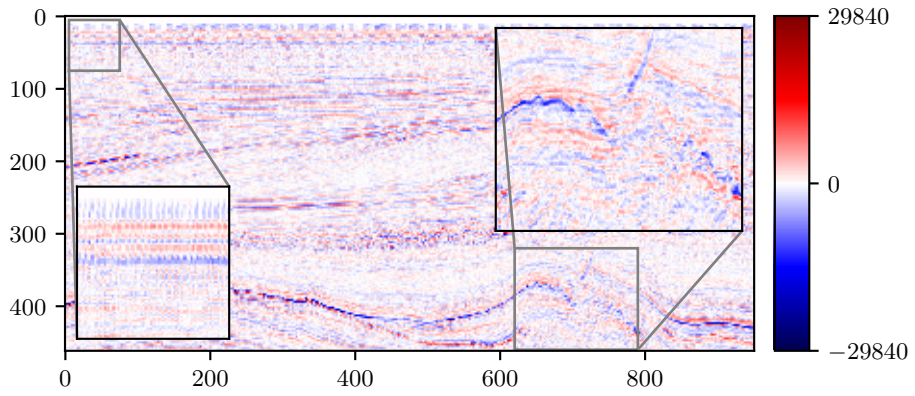


Figure 6.2: Seismic data cross section from the open-source F3 dataset (2020) released under the *Creative Commons BY-SA 3.0* license.

In Chapter 1, we briefly discussed some of the state of the art autotracking methods. Wu and Fomel, 2018 develop a method consisting of two steps, the first of which is to estimate slopes in the seismic volume. The method developed in this study also consists of two steps; the horizon depths and the reflection coefficients are estimated separately. However, these steps can be run iteratively, each time benefitting from each other. Furthermore, the method is developed with the possibility of interaction in mind, which might be difficult when using a clustering-based approach such as the one developed by Figueiredo et al., 2014.

It might also be a challenge to allow for interactivity when using machine learning-based approaches. One advantage of machine learning based methods is that, assuming there is enough data covering all frequent edge cases, one does not need to think of and find solutions for all those edge cases. In this context, edge cases refers to irregular behaviour or shape of a horizon. Examples of edge cases are irregularities caused by to faults, salt bodies, injectites, etc.

We have not discussed the runtime of the algorithms. Since little effort is spent on optimizing the implementation of the algorithms, and since our focus has not been on efficiency, we do not discuss runtimes in detail. However, it can be said that tracking the horizon in six inlines from the Volve data, providing the results presented in Figure 5.12, takes less than one minute on an average laptop.

Future work

As mentioned above, implementing a step for wavelet estimation into the autotracking algorithm seems likely to be important to improve the performance of the algorithm. Or we could go the other way and use only the center index of the wavelet. Testing the use of an 8-neighborhood system would be a simple first step going forward. In the following paragraphs, we discuss other

potential ways to improve the tracking algorithm.

When considering the closest neighbors only at each step of the algorithm, we do not look at larger-scale trends in horizon depths and reflection coefficients. For instance, we would not recognize if a horizon is trending upwards or downwards along a particular direction. It might be worthwhile to examine such trends when tracking a horizon. Another aspect we have not considered in this study is that horizons may vanish due to a layer of rock being completely eroded away in some areas. Theoretically, the algorithm we have developed might highlight such areas by estimating a reflection coefficient close to zero. However, it may be that including the possibility of a horizon vanishing directly as an option when estimating the depths, i.e. adding the possibility to Ω_s, is a better approach.

Interactivity is a key aspect of a tracking algorithm. Utilizing a combination of human expertise and the speed and thoroughness of an algorithm is often beneficial when working with complex data like seismic data. As described in this report, the Viterbi based autotracking algorithm would have to run a complete new process if new seed points are added after an initial attempt to track a horizon. This process can be performed more elegantly, e.g. by starting tracking from the inline with the new seed point and stopping when the horizon coincides with the initial prediction to a certain degree.

Other interesting extensions to explore include: using expectation maximization to estimate parameters like σ_δ and σ_ϵ, as well as ν used in the prior model; starting with a smaller grid containing a subset of the data points to reduce complexity and increase efficiency, i.e. a multi-grid model; and exploring the feasibility of using a multi-site response model.

Chapter 7

Conclusion

In this thesis, after relevant background information is discussed, we define a hidden Markov model for propagation of seismic horizons through seismic volumes. We develop two different algorithms for tracking seismic horizons. The algorithms are both interactive, in the sense that they take seed points as input. Seed points are horizon locations selected by a human interpreter. The horizon is tracked so that all seed points are included in the result.

One of the two algorithms is based on the Viterbi algorithm (Viterbi, 1967). The Viterbi algorithm finds maximum a posteriori (MAP) predictions for one-dimensional hidden Markov models. The Viterbi based tracking algorithm finds estimated MAP predictions for one two-dimensional seismic cross-section at a time, resulting in predicted depths of the horizon throughout the seismic volume. This method shows promise when working with synthetic data. However, we illustrate that there are issues when tracking horizons in real seismic data.

The second autotracking algorithm developed is a greedy algorithm that utilizes local information only at each step when expanding the horizon. However, this method exhibits stronger weaknesses than the Viterbi based algorithm when experimenting with relatively simple synthetic data.

We hypothesize that the main reason for the shortcomings of the Viterbi based algorithm when working with real seismic data is a poor wavelet estimate. Subsequently, we propose adding a step in the algorithm for wavelet estimation as a first step for building on the work presented in this thesis.

Bibliography

Books and Articles

Buland, Arild and Henning Omre (Jan. 2003). "Bayesian linearized AVO inversion". In: *Geophysics* 68.1, pp. 185–198.

Dyrendahl, Monika Maria (2018). "A better Horizon Auto Tracker - Powered by Machine Learning". MA thesis. NTNU.

Figueiredo, A M, P M Silva, M Gattass, F B Silva, and Ruy L Milidiú (Aug. 2014). "A seismic faces analysis approach to map 3D seismic horizons". In: *SEG Technical Program Expanded Abstracts 2014*. Denver, Colorado: Society of Exploration Geophysicists, pp. 1501–1505.

Hobbs, Bruce E, Winthrop Dickinson Means, and Paul Frederick Williams (1976). *An outline of structural geology*. Wiley.

Landrø, Martin and Lasse Amundsen (2018). *Introduction to exploration geophysics with recent advances*. Bivrost.

Letman, Jon (Aug. 2018). "The Race Is On to Mine the Deep Ocean". In: *National Geographic*.

Moja, Selamawit Serka, Zeytu Gashaw Asfaw, and Henning Omre (May 2019). "Bayesian Inversion in Hidden Markov Models with Varying Marginal Proportions". In: *Math. Geosci.* 51.4, pp. 463–484.

Simm, Rob, Mike Bacon, and Michael Bacon (Apr. 2014). *Seismic Amplitude: An Interpreter's Handbook*. en. Cambridge University Press.

Viterbi, A (Apr. 1967). "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm". In: *IEEE Trans. Inf. Theory* 13.2, pp. 260–269.

Williams, Virginia Vassilevska (2012). "Multiplying matrices faster than Coppersmith-Winograd". In: *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pp. 887–898.

Wu, Xinming and Sergey Fomel (July 2018). "Least-squares horizons with local slopes and multigrid correlations". In: *Geophysics* 83.4, pp. IM29–IM40.

Yu, Yingwei, Cliff Kelley, and Irina Mardanova (Aug. 2013). "A pattern recognition-based horizon auto-tracking algorithm". In: *13th International Congress of the Brazilian Geophysical Society & EXPOGEF, Rio*

de Janeiro, Brazil, 26–29 August 2013. Vol. 3. Rio de Janeiro, Brazil: Society of Exploration Geophysicists and Brazilian Geophysical Society, pp. 1612–1616.

Other Sources

Creative Commons BY-NC-SA 4.0 (n.d.). https://creativecommons.org/licenses/by-nc-sa/4.0/. Accessed: 2020-1-28.

Creative Commons BY-SA 3.0 (n.d.). https://creativecommons.org/licenses/by-sa/3.0/. Accessed: 2020-1-28.

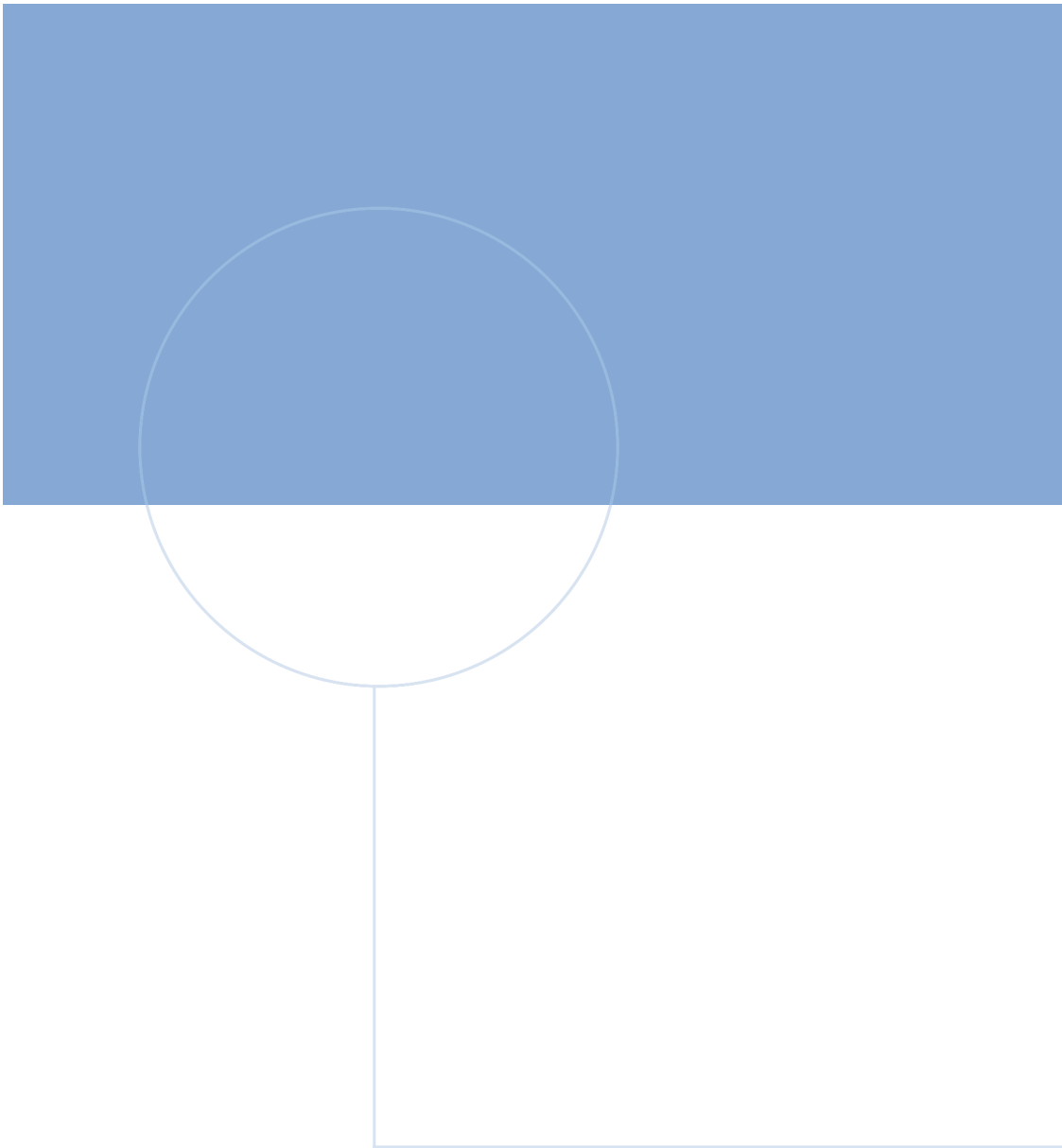
Henriksen, Sindre (n.d.). https://www.github.com/sindrehenriksen/mamosa.

RagnaRock Geo (n.d.). https://www.ragnarockgeo.com/.

SEG Wiki (n.d.). https://wiki.seg.org/wiki/Wavelet_estimation_for_well_ties. Accessed: 2020-1-23.

TerraNubis - Data Info of F3 Block (n.d.). https://terranubis.com/datainfo/Netherlands-Offshore-F3-Block-Complete. Accessed: 2020-1-28.

Volve field data village (n.d.). https://www.equinor.com/en/how-and-why/digitalisation-in-our-dna/volve-field-data-village-download.html. Accessed: 2020-1-28.



NTNU

Norwegian University of
Science and Technology